



IMPULS
PRO KARIÉRU
A PRAXI

METODICKÉ POSTUPY

PRO REALIZACI WORKSHOPŮ
DIGITALIZACE VE ŠKOLÁCH



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání



Jhk.cz



JIHOČESKÁ
HOSPODÁŘSKÁ
KOMORA


Jihočeský kraj

Obsah

1. Pracovní postupy pro základní školy / **6**

1.1 Programování Micro:bit pro ZŠ / **7**

1.2 Grafika – práce s fotografií pro ZŠ / **29**

1.3 Tvorba digitálního obsahu pro ZŠ / **51**

1.4 Workshop Codey Rocky / **77**

1.5 Kybernetická bezpečnost pro ZŠ / **95**

1.6 Blokové programování pro ZŠ / **113**

1.7 3D modelování a 3D tisk pro ZŠ / **167**

2. Pracovní postupy pro střední školy / **204**

2.1 Programování Micro:bit pro SŠ / **205**

2.2 Tvorba digitálního obsahu pro SŠ / **227**

2.3 Kybernetická bezpečnost pro SŠ / **245**

2.4 Blokové programování pro SŠ / **263**

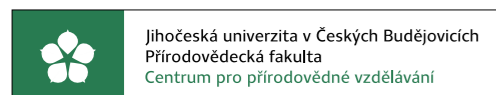
2.5 Internet věcí (IoT) pro SŠ / **317**

2.6 3D modelování a 3D tisk pro SŠ / **343**

Tato publikace „Metodické postupy pro realizaci workshopů digitalizace ve školách“, vznikla v rámci aktivity Asistenčního centra Impuls pro kariéru a praxi při Jihočeské hospodářské komoře díky realizaci projektu „Implementace Krajského akčního plánu Jihočeského kraje III“, který je spolufinancován Evropskou unií. Registrační číslo projektu CZ.02.3.68/0.0/0.0/19_078/0018246

Elektronická verze publikace je k dispozici na www.impulsprokarieru.cz

Publikaci připravila: Jihočeská univerzita v Českých Budějovicích
s kolektivem autorů z Centra pro přírodovědné vzdělávání
Přírodovědecké fakulty Jihočeské univerzity



Grafický design: Čestmír Sukdol – www.brandi.cz

Vydala: Jihočeská hospodářská komora

2021

S rozvojem nových technologií, jako je umělá inteligence nebo robotika, se naše společnost rychle proměňuje. Dopad digitální transformace na každodenní život se stále prohlubuje. Na tyto změny musí reagovat také vzdělávací systém, který by měl poskytovat odpovídající, moderní a kvalitní znalosti, dovednosti a kompetence, vedoucí k prosperitě jednotlivce i celé společnosti. Evropská rada v říjnu 2017 dokonce vyzvala k tomu, aby systémy odborné přípravy a vzdělávání byly „vhodné pro digitální věk“.

Stejně jako všechny velké technologické pokroky v minulosti má digitalizace vliv na to, jak lidé žijí, jak komunikují, jak studují a pracují. Klíčovým problémem se stává ochrana osobních údajů, kyberšikana nebo šíření falešných zpráv. Rychle se proměňuje i pracovní trh. Některá pracovní místa zmizí, mnoho pracovních míst a průmyslových odvětví bude transformováno a objeví se nové činnosti. Proto je investice do vzdělávání (celoživotního) v nových technologiích jednou z nejdůležitějších.

Inovace ve vzdělávání může být ale efektivní jen tehdy, budou-li dobře připraveni učitelé. Na vzdělávání v oblasti informatiky a informačních technologií je tak kladen stále větší důraz a učitelé čelí složitým úkolům a velkým výzvám. Musí se zorientovat v novém konceptu výuky, naučit se nové učební metody a připravit nové výukové materiály.

V rámci projektu Metodické materiály a realizace metodických workshopů digitalizace, který je realizován Jihočeskou hospodářskou komorou ve spolupráci s Centrem pro přírodovědné vzdělávání Přírodovědecké fakulty Jihočeské univerzity, bylo připraveno několik metodických materiálů pro učitele základních a středních škol.

Metodické materiály mají jednotnou strukturu, obsahují teoretickou část k dané problematice, příklady z praxe, metodickou a didaktickou část, přehled doporučených pomůcek a pracovní listy pro žáky a studenty, případně návody k jejich vytváření. Jsou připraveny v souladu s koncepcí STEM a umožňují tak učitelům nejen získat základní přehled a učební materiál ke konkrétní problematice, ale také propojit některá témata mezioborově a mezipředmětově.

Doc. RNDr. Ing. Jana Kalová, Ph.D.,
Jihočeská univerzita

Milí vyučující,

v rukách držíte publikaci s názvem Metodické postupy pro realizaci workshopů digitalizace ve školách, která je určena učitelům informatiky základních a středních škol. Tato publikace je primárně součástí workshopů digitalizace pro pedagogy, které realizuje Jihočeská hospodářská komora, ale může být používána samostatně i bez absolvování workshopů.

Publikace obsahuje celkem 13 témat zaměřených na rozvoj digitálních kompetencí rozdělených pro výuku na základních a středních školách zaměřených průřezově od programování po kybernetickou bezpečnost, 3D tisk nebo práci s fotografií. Každé téma obsahuje základní instrukce s doporučenou časovou náročností, teoretickou část, metodickou a didaktickou část a pracovní listy, které jsou určeny pro práci se žáky. Najdete zde i odkazy na další literaturu a zdroje.

Cílem této publikace je zejména přinést vyučujícím inspiraci do výuky a rozšířit učební materiály. Tento materiál je také dostupný v elektronické podobě na webu <https://www.impulsprokarieru.cz/digitalni-kompetence>.

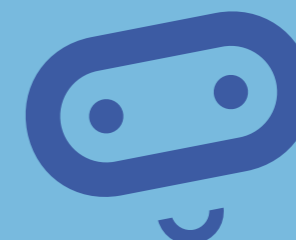
Tato publikace vznikla v rámci aktivity Asistenčního centra Impuls pro kariéru a praxi při Jihočeské hospodářské komoře díky realizaci projektu „Implementace Krajského akčního plánu Jihočeského kraje III“, který je spolufinancován Evropskou unií. Registrační číslo projektu CZ.02.3.68/0.0/0.0/19_078/0018246.

Přejeme mnoho úspěchů při rozvíjení digitálních dovedností.

Tým Jihočeské hospodářské komory

PRACOVNÍ POSTUPY PRO ZÁKLADNÍ ŠKOLY

PROGRAMOVÁNÍ MICRO:BIT PRO ZŠ



Základní instrukce

Tento kurz je doporučován pro žáky druhého stupně základní školy nebo nižší ročníky osmiletých gymnázií. Autor má zkušenosti s výukou této látky od sedmé třídy, ale je přesvědčen, že je možné toto téma učit i v nižších ročnících. Je možné tento kurs učit i na vyšších ročnících středních škol, s výjimkou technických škol a gymnázií. Pro tyto školy je určen obdobný kurz, který si místo grafického jazyka MakeCode bere za základ programovací jazyk Python.

Časová dotace tohoto kurzu nemůže být zcela jednoznačná. Závisí na tom, zda se studenti již seznámili s grafickým programováním a zda již mají nějaké zkušenosti s Micro:bitem. Učitel by měl nejprve prostudovat všechny materiály dané k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.



Cílem workshopu je představit vývojovou destičku BBC Micro:bit primárně určenou pro výuku programování na základní i střední škole. V našem kurzu se zaměříme na programovací jazyk MicroPython jako podmnožinu jazyka Python. Začneme od úplných základů, ale postupně se přeneseme až k možnosti komunikace dvou Micro:bitů a ukážeme si práci ve dvojici anebo i ve větší skupině. Finálním produktem by mělo být vytvoření zabezpečovacího zařízení tvořeného dvěma Micro:bity, z nichž jeden má funkci sensoru a druhý hlásiče.

Níže naleznete doporučený průběh a časovou dotaci. Kurz lze rozčlenit do čtyř částí:

- 1. Práce s výstupy** – výstup na displej a přehrávání zvuku.
 - a. Pokud již proběhl nějaký kurz Micro:bitu, je možné tuto část přeskočit nebo minimalizovat.
 - b. Pokud žáci již měli nějaké jiné grafické programování např. Scratch, pak postačí jedna vyučovací hodina (45 minut)
 - c. Pokud žáci ještě neměli programování a neznají Micro:bit, pak doporučuji dvě vyučovací hodiny.
- 2. Práce se vstupy** – stisk tlačítka, detekce pohybu, magnetického pole, zvuku, světla, měření teploty.
 - a. Pokud žáci znají Micro:bit a programovali, postačí jedna vyučovací hodina. Pozor, pokud znají Microbit V1, u nového typu jsou některé věci navíc.
 - b. Ve všech ostatních případech doporučuji dvě vyučovací hodiny. Je nutné probrat podmíněné příkazy.
- 3. Komunikace mezi dvěma Micro:bity** – posílání zpráv mezi dvěma Micro:bity pomocí vestavěného radio modulu. Bezpečnost této komunikace.
 - a. Pokud s tímto žáci již pracovali, je možné tuto pasáž zcela přeskočit.
 - b. Jinak doporučuji jednu vyučovací hodinu.
- 4. Samotná tvorba zabezpečovacího zařízení** je už pro všechny stejná. Doporučuji nejméně dvě vyučovací hodiny. V první hodině učitel zvolí jednu z možností zabezpečení a společně s žáky jej zkonstruuje. Ve druhé a případně dalších hodinách žáci samostatně pracují na svých projektech. Je zde možnost přidání jedné nebo více hodin, kdy žáci budou své projekty představovat ostatním.

Minimální doba, za kterou lze tento výukový balík absolvovat, je tedy **dvě vyučovací hodiny**.

– první hodina: části 1 až 3, druhá hodina: část 4. Doporučený počet je osm hodin (2, 2, 1, 3) včetně představení projektů. Je třeba počítat s možností, že za běhu bude nutné přidat či ubrat hodiny.

Pro části 1 až 3 a první hodinu části 4 je vhodnou metodou výuky výklad spojený se samostatnou prací žáků v hodině. Učitel by měl vždy část látky vysvětlit a pak nechat žáky, aby si vše vyzkoušeli a přišli případně na další možnosti.

Je třeba počítat rovněž s tím, že od části 3 dále by žáci měli pracovat v ideálně dvoučlenných týmech (pokud je málo Micro:bitů tak vícečlenných) tak, aby v každém týmu byly nejméně dva Micro:bity.

V závěrečných hodinách pak lze s úspěchem použít projektovou výuku, kdy žáci buď dostanou přidělené téma nebo si jej zcela sami zvolí. V poslední hodině by každý tým měl prezentovat své funkční zabezpečovací zařízení a předvést jeho přednosti a slabiny.

Pro výuku je doporučen Micro:bit verze 2 (Micro:bit V2). Nejlépe pro každého žáka jeden. Doporučuji vybavit se navíc USB kabely a držáky baterií pro běh Micro:bitu mimo počítač. To lze pořídit v kompletu za cca 550 Kč. Současně doporučuji vybavit se náhradními AAA bateriemi.

Máme-li Micro:bit V1, je nutné řešit audio výstup tak, jak je to popsáno v učebnici níže. V tom případě budete potřebovat dva kabely s krokodýlky pro každého a nějaký zdroj zvuku, např. sluchátka.

Pro výuku použijeme grafický programovací jazyk Microsoft MakeCode, který je podobný například jazyku Scratch. Pro tvorbu programů v MakeCode máme dvě možnosti:

1. Nemáme-li možnost instalovat programy na počítač nebo máme-li Linux, použijeme online editor na adrese: <https://makecode.microbit.org/>. Předpokladem je rovněž dostatečně kvalitní a rychlé internetové připojení.
2. Pokud můžeme instalovat do počítače a máme na něm operační systém Windows nebo OS X, pak doporučuji off-line editor MakeCode. Nebudete tak závislí na výkonu internetu a budete si jistí, že po ukončení práce neztratíte své programy. Editor stáhnete zde: <https://makecode.microbit.org/offline>

Co se týče podkladů pro výuku, máme několik možností.

Základní literaturou pro tento projekt by měla být česká učebnice vypracovaná v rámci projektu **iMyšlení** s názvem **Robotika pro základní školy: programujeme Micro:bit pomocí Makecode**.

Mnoho dalších informací a návodů; včetně mnoha zajímavých projektů lze rovněž najít na adrese microbiti.cz.

Pro ty, kdo ovládají angličtinu mohou rovněž být zdrojem inspirace původní webové stránky projektu **BBC Micro:bit**.



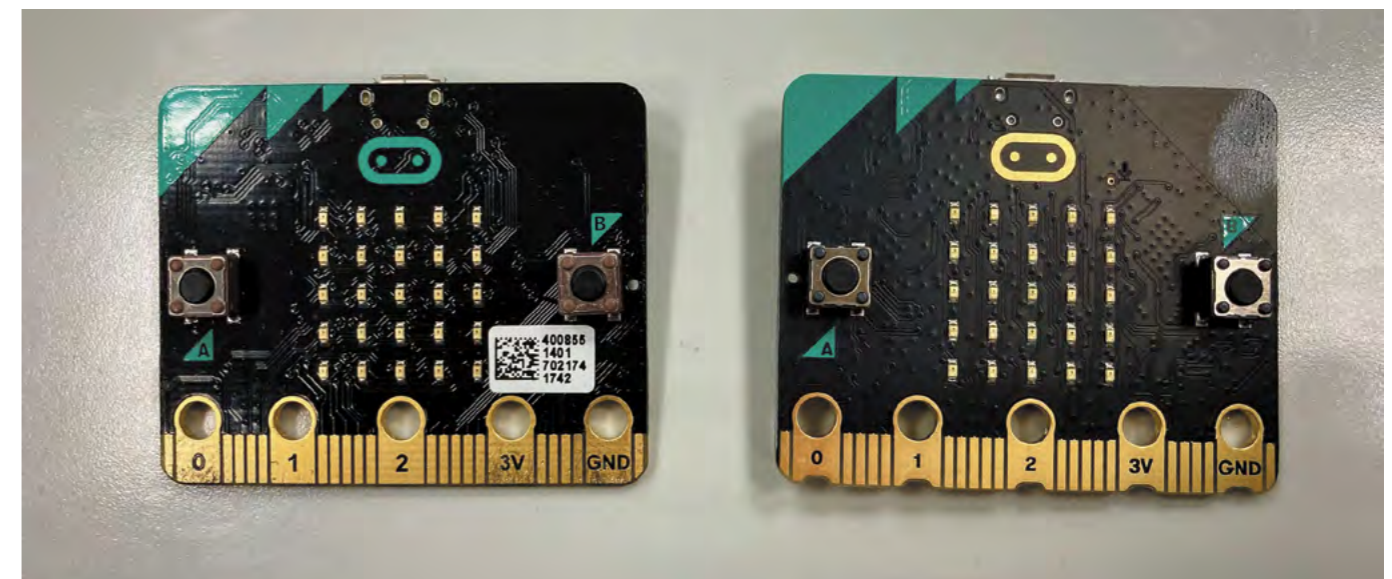
Teoretická část k dané problematice

Micro:bit je kapesní počítač, který vás uvede do světa software a hardware a jejich vzájemné spolupráce. Obsahuje LED displej, tlačítka, sensory a mnoho programovatelných funkcí. Nová verze micro:bitu obsahuje i mikrofon a buzzer. (<http://www.microbit.org>).

Vlastnosti Micro:bitu (V2 znamená pouze verze V2):

- ✓ Matice 5x5 LED diod
- ✓ Dvě programovatelná tlačítka
- ✓ Jedno dotekové tlačítko (V2)
- ✓ Akcelerátor, umožňující reagovat na pohyby
- ✓ Čidlo světla, magnetického pole a teploměr
- ✓ Mikrofon (V2)
- ✓ Buzzer (primitivní reproduktor) (V2)
- ✓ Možnost vzájemné komunikace pomocí radií
- ✓ Bluetooth
- ✓ Konektory pro připojení dalších zařízení, z nich tři pro pohodlné připojení pomocí kabelů s krokodýlky
- ✓ Možnost snadného připojení k rozšiřujícím zařízením pomocí pásu konektorů

Existují dvě verze micro:bitu. Starší verze prodáváná od roku 2016, nyní označovaná jako V1 a novější od října 2020 označovaná jako V2.



Obrázek 1: čelní strana micro:bitu, vlevo V1, vpravo V2

Rozhodně si nyní pořizujte V2, je lépe vybavená a má větší paměť pro programy při de facto stejné pořizovací ceně. Její výhodou je integrovaný buzzer a mikrofon, takže pro práci s tímto textem nepotřebujete nic jiného než dva micro:bity.

Grafické programování – je takové programování, kdy program netvoříme psaním kódu na klávesnici, ale posouváním grafických bloků po obrazovce. Mezi tyto nástroje patří například Scratch, MakeBlock nebo Open Roberta Lab.

Microsoft MakeCode – grafický programovací jazyk vyvinutý firmou Microsoft pro programování Micro:bitu.



Obrázek 2: Zadní strana micro:bitu, vlevo V1, vpravo V2

Příklady z praxe

Autora k vytvoření tohoto materiálu přivedla příhoda z praxe, která se stala, když prvním rokem ověřoval svou učebnici na základní škole v rámci testování.

Dva bratři vlastní dva micro:bity si doma vytvořili primitivní zabezpečovací zařízení. Jeden microbit ukryli na verandě domu a naprogramovali jej, aby reagoval na změnu světla posláním signálu druhému micro:bitu umístěnému v jejich pokoji. Ten měli připojený k reproduktoru, aby je na tuto skutečnost upozornil. Když přišel domů někdo z rodičů, byli o tom včas informováni a mohli odložit tablety, knihy atd. a vzít si do ruky učebnice.

Ačkoliv z pedagogického hlediska s tím samozřejmě nesouhlasím, z didaktického hlediska se jedná o zajímavý případ spojený se spontánní týmovou spoluprací a využitím znalostí získaných v kroužku.

Absolvováním tohoto kurzu žáci získají nebo si prohloubí základní programátorské návyky. Dále se dozví informace ze světa hardware a integrace software a hardware. Zde záleží pouze na učiteli, který musí odhadnout úroveň svých žáků a jak hluboko si může dovolit ponořit se do dané problematiky.

Dále lze na této problematice dobře vysvětlit možné bezpečnostní problémy bezdrátové komunikace – např. odposlouchávání, man in the middle attack atd.

V neposlední řadě lze tímto úkolem rozvíjet práci ve skupině – pro správnou funkci naprosté většiny případů jsou nutné dva micro:bity. S úspěchem lze na závěr zadat žákům projekt ve skupině, který budou pak muset před ostatními představit a obhájit.

Metodická a didaktická část

Jak bylo vysvětleno v první kapitole, kurz je rozdělen do čtyř částí. Postupně se zastavíme u všech čtyř a řekneme si, co v nich učít.

Práce s výstupy

V této části je třeba zvládnout následující úkoly:

- ✓ Seznámit se s micro:bitem
- ✓ Seznámit se s prostředím editoru MakeCode
- ✓ Naučit se nahrát vytvořený kód do micro:bitu
- ✓ Pochopit rozdíl mezi bloky Při startu a Opakuj stále
- ✓ Práce s displejem – zobrazení ikonky a textu
- ✓ Práce se zvukem

Na úvod rozdělá učitel žákům micro:bity a v krátkosti jim je představí. Upozorněte žáky na skutečnost, že micro:bit má na sobě popisky jednotlivých součástí. Vysvětlete význam jednotlivých tlačítek – A, B programovatelná tlačítka, Reset slouží k opětovnému spuštění programu od začátku anebo k vypnutí micro:bitu při podržení 3 sekundy.

Nyní si studenti spustí MakeCode program anebo se na internetu připojí k online editoru MakeCode. Seznamte je s prostředím a základními principy grafického programování, tj. práce s bloky – zavedení, kopírování, přesun a smazání.

Vysvětlete rozdíl mezi připravenými bloky „Spustit při startu“ (spustí se jednou při startu) a „Opakuj stále“ (opakuje se neustále dokola). Ukažte žákům, jak jej znovu vytvoří, pokud si jej omylem smažou.

Zkuste vytvořit jednoduchý program a nahrát jej do micro:bitu. Ukažte žákům rozdíl mezi možnostmi Uložit (program se uloží na disk) a Stáhnout (program se nahraje rovnou do micro:bitu). Vysvětlete, že po připojení je micro:bit přístupný podobně jako flash disk a uložený program do něj lze nahrát i tak, že jej na tento disk přetáhneme. Ukažte žákům, že během nahrávání bliká žlutá dioda na zadní straně. Také jim ukažte,



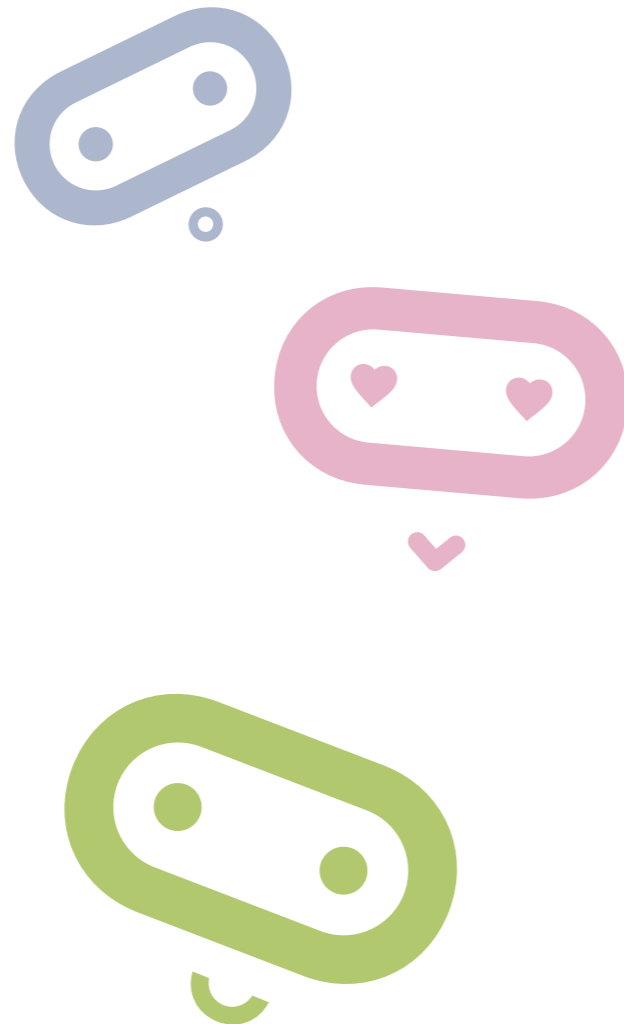
že ukončený program mohou opět spustit stiskem tlačítka Reset.

U dalších programů v této části nebudou již většinou vkládány obrázky (zdrojový kód). Naleznete je v příložených pracovních listech.

Nechte žáky upravit program tak, že zaměníte blok „při startu“ za „opakuj stále“. Zkuste přidat časovou prodlevu. Vyzkoušejte zobrazení připravených obrázků, opět s vhodnými časovými prodlevami. Vyzkoušejte případně i cyklus „Opakuj n krát“. Vysvětlete žákům rozdíl v nekonečném cyklu a cyklus pevným počtem opakování.

Alternativně můžete zavést pojem proměnná a žákům vysvětlit podmíněný cyklus.

Vyzkoušejte si přehrávání zvuku. Pokud máte micro:bit V1, musíte nejprve připojit výstupní zařízení. Vyzkoušejte bloky „hraj tón“, „hraj melodii“ a „play sound“ (pouze V2). Můžete nechat žáky složit vlastní melodii, ale v tom případě doporučuji vybavit se klapkami na uši a třídu zvukově izolovat.



Práce se vstupy

✓ Kromě potřeb z minulé kapitoly doporučuji jeden nebo více silných magnetů na třídu.

Nejjednodušším typem vstupu jsou dvě tlačítka značená A a B. Jejich nejjednodušší použití je pomocí bloku „Po stisknutí tlačítka A(B)“ v kategorii vstup. Naprogramujte akci (výstup na displej, zahrání tónu“ po stisku zvoleného tlačítka). Přiřadte aktivitu k oběma tlačítkům.

Vyzkoušejte si naprogramovat totéž pomocí bloku „Opakuj stále“ a podmíněného bloku „když, jinak když, jinak“. Pozor část „jinak když“ přidáte stiskem znaménka +. Vysvětlete na tom podmíněný příkaz. Návrh bloku zde:

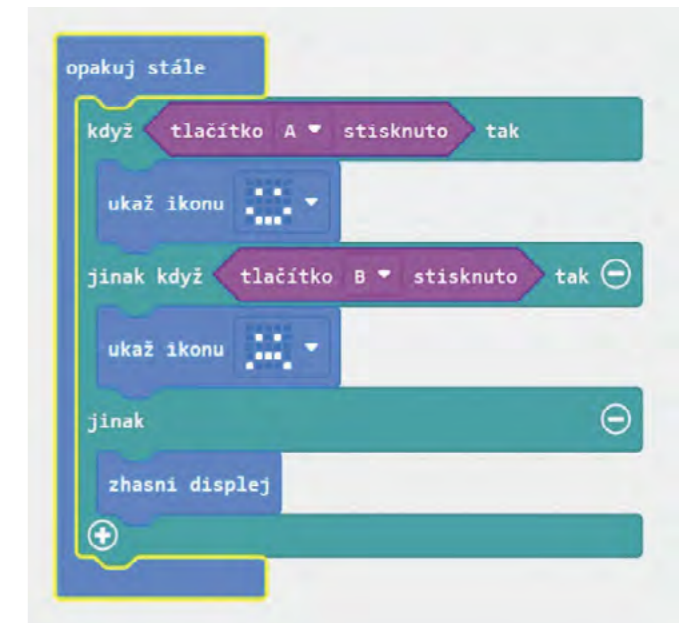
Vyzkoušejte si nyní další možnosti vstupů. Použijte bloky „Při zatřesení“ (vyzkoušejte i jiné pohyby) a máte-li V2, tak „on logo pressed“ (je stisknuto logo) a „on loud sound“ (je slyšet zvuk).

Nyní vyzkoušejte měřit další veličiny a zobrazit jejich hodnotu: intenzitu světla, magnetického pole a teplotu. Můžete vyzkoušet i azimut. Nezapomeňte, že před použitím se musí micro:bit zkalibrovat (micro:bit napíše výzvu, pak se objeví uprostřed displeje bod a natáčením micro:bitu musíme celý displej zaplnit).

Napište program, který po stisku tlačítka A zobrazí teplotu, po stisku tlačítka B intenzitu světla a současném stisku obou tlačítek zobrazí intenzitu magnetického pole. Experimentujte se zhasínáním a rozsvícením a osvětlením a dále s přiblížením magnetu.

Můžete na neznalcích vyzkoušet magii. Přiblížíte-li ruku zařatou v pěst k micro:bitu, zobrazí se smajlík. Neříkejte, že máte v té pěsti silný magnet a vyzvěte je, ať to zkusí po vás. Můžete úkol vylepšit tím, že řeknete, že obrázek se zobrazí jen tomu, kdo má magický potenciál. Slovo magický pak můžete nahradit slovem magnetický a sledujte kdy, si záměny dotyčný všimne.

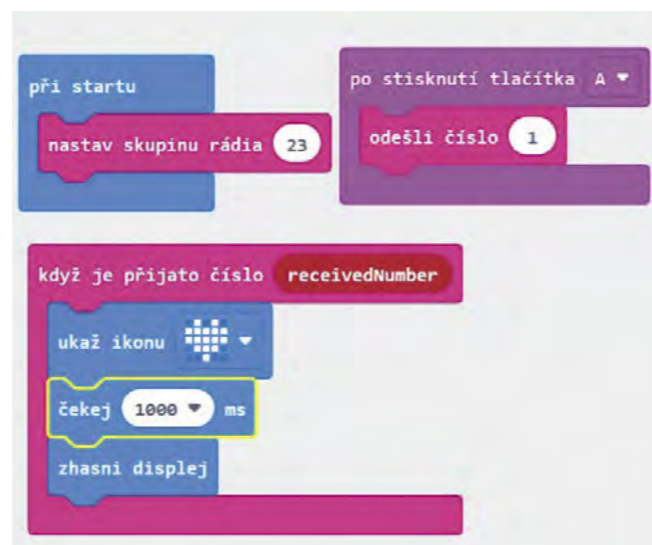
Můžete vyzkoušet ještě další magický program, při zatřesení vám dá micro:bit náhodnou odpověď ano/ne. Můžete se ho tak dotazovat na mám/nemám.



Komunikace mezi dvěma micro:bity

- ✓ Pro splnění této hodiny musí již být vytvořeny nejméně dvoučlenné skupiny vybavené dvěma micro:bity.

Na úvod naprogramujte takovýto jednoduchý programek na obou micro:bittech ve skupině. ➡



Všimněte si při testování v editoru, že po stisku tlačítka A se zobrazí druhý micro:bit (příjemce), který bude vykonávat činnost při obdržení signálu. Je důležité, aby měly různé skupiny různou skupinu rádia, jinak se vzájemně budou rušit. (Nechte nejdříve žáky, aby se přesvědčili, co se stane, když to neudělají. Bude se vám pak dobře vykládat bezpečnost bezdrátového spojení.)

Poté co si žáci rozdělí skupiny rádia, je nechte zkusit si vzájemně posílat signál. Je-li to možné (a bezpečné), vypusťte je mimo učebnu, ať si otestují dosah micro:bitů ve volném prostoru i přes zdi. Bude se to hodit při tvorbě zabezpečovacího zařízení.

Naeditujte složitější případ, kdy se posílá jiné číslo při stisku A a jiné při stisku B. Máte-li V2, tak třetí se pošle při stisku loga. Přijímající micro:bit pak podle přijatého čísla rozhodne o své činnosti. Kód příkladu je v pracovním listě.

Žáci mohou kód upravit tak, že význam při poslání signálu z jednoho micro:bitu může být např. „Půjdem na hřiště?“, „Půjdem se učit“, „Máme úkoly“ a odpovědi zpět např. „Ano“, „Ne“ a „Nevím“. Ponechte žákům ať si sami vyberou a řešení otestují. Micro:bity lze nyní označit jako tázající a odpovídající.

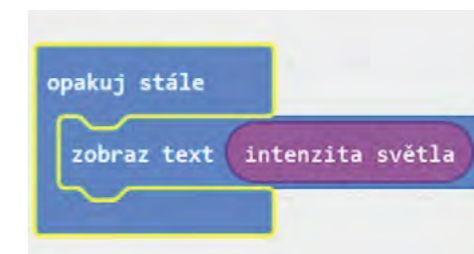
Vysvětlete žákům nebezpečí této komunikace. Pokud útočník zná použitou skupinu a význam kódů, může konverzaci odposlouchávat. Navíc může i do komunikace vstoupit a způsobit zmatení účastníků.



Tvorba zabezpečovacího zařízení

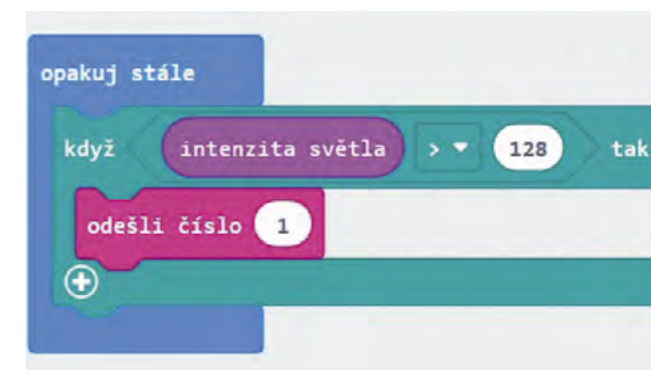
Vytvoříme jednoduché zabezpečovací zařízení, které bude reagovat na intenzitu světla.

- 📷 Vyzkoušejte si nejprve na jednoduchém programu rozdíl v hodnotě intenzity světla mezi zhasnutím a rozsvícením (zavřenými a otevřenými žaluziemi).

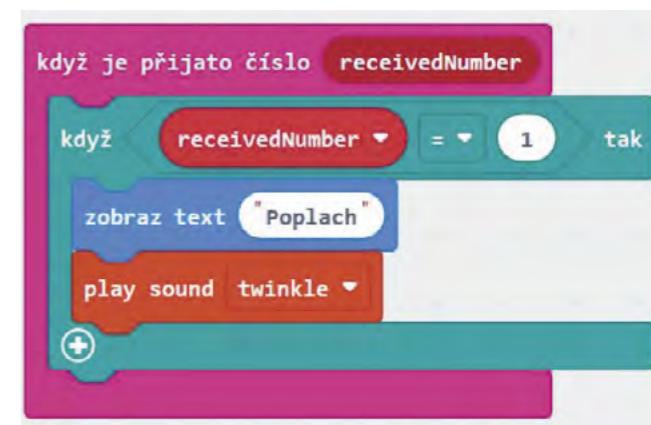


Potřebujeme dva micro:bity:

- ✓ Vysílač – hlídá intenzitu světla a při rozsvícení (zhasnutí) vyšle signál. Kód by mohl vypadat např. takto.



- ✓ Přijímač – dostane info, že došlo k incidentu a vyhlásí poplach. Kód například takto.



Nyní zadejte žákům projekt na vytvoření zabezpečovacího zařízení pomocí dvou micro:bitů. Napovězte jim, co mohou použít – stisk tlačítka (naopak i uvolnění tlačítka), změna intenzity světla, teploty, magnetického pole, audio signál, pohyb.

Doporučené pomůcky

V první řadě potřebujeme dostatek micro:bitů. Ideální stav je, když má každý žák svůj micro:bit. Ideálnější je pak, když má svůj osobní micro:bit, pokud si to může škola anebo rodiče žáků dovolit.

Ke každému micro:bitu potřebujeme připojovací USB kabel a doporučuji i přenosný držák na baterie. Ideální je koupit vše dohromady jako výhodný balíček, jak je popsáno v úvodu textu.

Dále potřebujeme počítač, který je buď připojen do internetu nebo má nainstalovanou off-line verzi editoru MakeCode. Určitě doporučuji druhou možnost, nebudeme tolik vydáni vrtochům internetového připojení.

Pro vysvětlení reakce na magnetické pole potřebujeme také silnější magnet. Např. „vypůjčený“ z nástěnky.

Pokud máme verzi jedna micro:bitu pak ještě potřebujeme externí repráčky anebo sluchátka.

Pracovní list

Přílohou tohoto materiálu jsou pracovní listy. Tyto pracovní listy jsou k dispozici v editovatelné elektronické formě, aby si je každý učitel mohl upravit, např. dle toho, co má již s žáky probráno.

Pracovní listy jsou čtyři, pro každé z výše uvedených témat jeden. V pracovních listech počítám s tím, že škola má k dispozici Micro:bit V2. Pokud má k dispozici pouze staší verzi V1, je nutné tyto listy upravit. Rozdíly jsou zejména v tom, že u V1 je nutno pro přehrání zvuku připojit externí zařízení a dále V1 nemá mikrofon a dotekové tlačítko, takže tyto dvě zařízení nelze použít jako čidlo pro vyhlášení poplachu.

Pojďme na to!



Pracovní list 1 Práce s výstupy micro:bitu

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem MakeCode anebo s přístupem na internet
- ✓ Micro:bit V2
- ✓ Propojovací USB kabel

Ajdeme na to

Vezměte si svůj micro:bit a pořádně si jej prohlédněte. Na přední straně máte pole 5x5 LED diod, které umí svítit různou intenzitou červené barvy. Po jejich stranách máte dvě programovatelná tlačítka označení A a B. Nad diodami se ještě nachází programovatelné dotekové tlačítko a vpravo nahoře nad LED je malá tečka, což je vlastně mikrofon. Otočme nyní micro:bit a podívejme se na zadní stranu. Nahoře máme dva porty microUSB pro připojení k PC a konektor pro připojení baterie packu. Mezi nimi je tlačítko RESET pro opětovné spuštění programu od začátku. Všimněte si, že další části jsou zde popsány. Pro úplnost ještě dodejme, že z obou stran dole vidíme piny pro připojení dalších periférií. Celý micro:bit je například možné zasunout do nějakého externího zařízení.

Spusťte program MakeCode nebo se na internetu připojte k **online editoru MakeCode**. Seznamte se s prostředím a základním principy grafického programování, tj. práce s bloky – zavedení, kopírování, přesun a smazání.

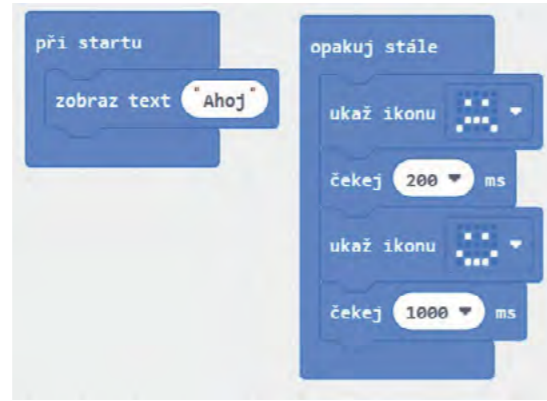
Zkuste vytvořit jednoduchý program a nahrát jej do micro:bitu. Zkuste si rozdíl mezi možnostmi Uložit (program se uloží na disk) a Stáhnout (program se nahraje rovnou do micro:bitu, je-li připojený). Po připojení je micro:bit přístupný podobně jako flash disk a uložený program do něj lze nahrát i tak, že jej na tento disk přetáhneme. Všimněte si, že během nahrávání bliká žlutá dioda na zadní straně. Vyzkoušejte si i to, že program lze spustit opakovaně pomocí tlačítka RESET (anebo odpojením a připojením micro:bitu ke zdroji energie).



Všimněte si rovněž emulátoru micro:bitu v levé horní části programu MakeCode. Programy si tak můžete zkusit i když nemáte u sebe micro:bit (např. doma).

Zaměňte nyní blok „při startu“ za „opakuj stále“. Zkuste přidat časovou prodlevu. Vyzkoušejte zobrazení připravených obrázků, opět s vhodnými časovými prodlevami.

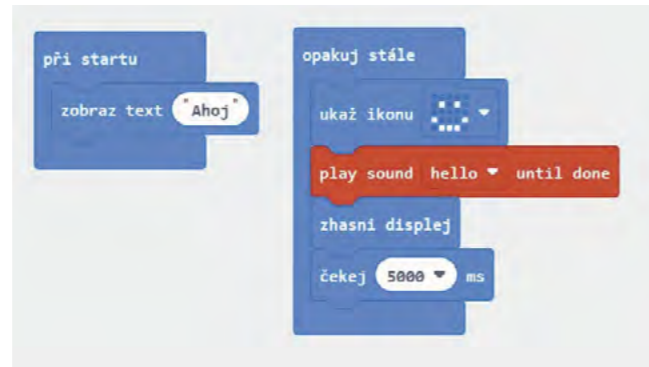
Nyní již vidíte rozdíl mezi těmito bloky. Blok „Při startu“ se spustí jednou na začátku a naopak „Opakuj dokola“ dělá přesně to, co slibuje jeho název. Místo něj lze použít i cyklus „Opakuj n krát“. Tady se jedná o cyklus s podmíněným počtem opakování = říkáme kolikrát se má provést.



Vyzkoušejte si přidat i blok „zhasni displej“ a ponechte displej zhasnutý např. 200 ms po skončení zobrazení úsměvu.

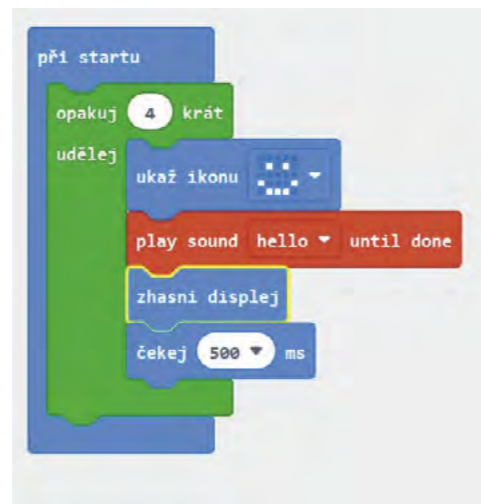
Můžete si i z nabídky displej zkusit rozsvěcet a zhasínat jednotlivé led diody.

Vyzkoušejte si přehrávání zvuku. Náš micro:bit V2 má připravené zvuky v bloku „play sound“.



Protože z opakujícího zvuku vás jistě budou poměrně brzy bolet uši, je daleko lepší použít místo této konstrukce následující.

Vyzkoušejte bloky „hraj tón“, „hraj melodii“ a „play sound“ (pouze V2). Můžete si rovněž zkusit vytvořit vlastní melodii.



Co jste se naučili

Zobrazovat obrázky a text na displeji micro:bitu. Víte, jak část kódu spustit jednou, opakovaně, několikrát. Dále umíte na micro:bitu přehrát tón, připravenou anebo vlastní melodii.

Pracovní list 2 Práce se vstupy micro:bitu

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem MakeCode anebo s přístupem na internet
- ✓ Micro:bit V2
- ✓ Propojovací USB kabel

A jdeme na to

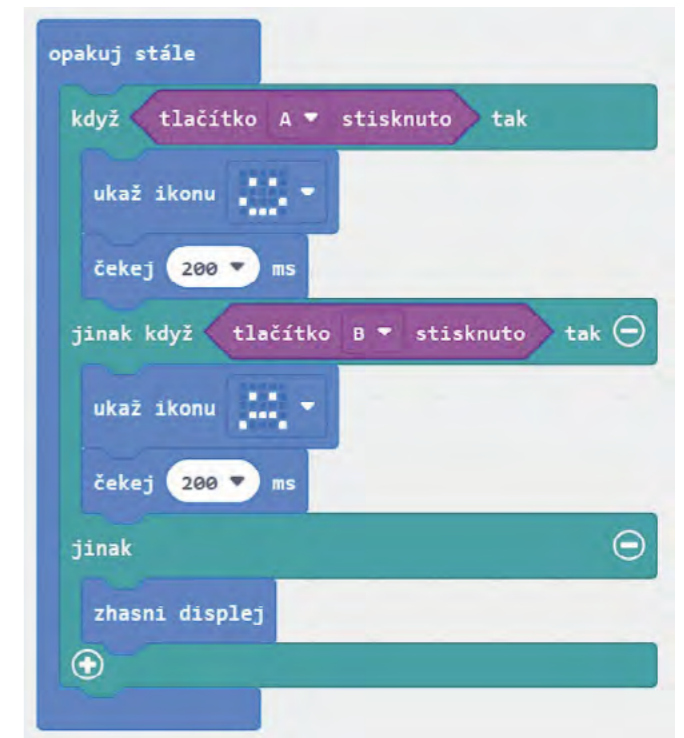
Nejjednodušším typem vstupu jsou dvě tlačítka značená A a B. Jejich nejjednodušší použití je pomocí bloku „Po stisknutí tlačítka A(B)“ v kategorii vstup. Vyzkoušejte následující kód.



Vyzkoušejte si naprogramovat totéž pomocí bloku „Opakuj stále“ a podmíněného bloku „když, jinak když, jinak“. Pozor část „jinak když“ přidáte stiskem znaménka +.

Jedná se o podmíněný příkaz. Pokud platí první podmínka, provede se a skočí se na konec bloku, a tak dále, až pokud se dojde na konec, provede se část jinak.

Vyzkoušejte si nyní další možnosti vstupů. Použijte bloky „Při zatřesení“ „on loud sound“ (je slyšet zvuk).



Nyní vyzkoušejte měřit další veličiny a zobrazit jejich hodnotu: intenzitu světla, magnetického pole a teplotu. Můžete vyzkoušet i azimut. Nezapomeňte, že před použitím se musí micro:bit zkalibrovat (micro:bit napíše výzvu, pak se objeví uprostřed displeje bod a natáčením micro:bitu musíme celý displej zaplnit).

Napište program, který po stisku tlačítka A zobrazí teplotu, po stisku tlačítka B intenzitu světla a současném stisku obou tlačítek zobrazí intenzitu magnetického pole. Experimentujte se zhasínáním a rozsvícením a osvětlením a dále s přiblížením magnetu.

Zkuste podobně experimentovat s úrovní hlasitosti v okolí.

Můžete na neznalcích vyzkoušet magii:

Přiblížíte-li ruku zaťatou v pěst k micro:bitu zobrazí se smajlík. Neříkejte, že máte v té pěstí silný magnet a vyzvěte je, ať to zkusí po vás. Můžete úkol vylepšit, tím že řeknete, že obrázek zobrazí jen tomu, kdo má magický potenciál. Slovo magický pak můžete nahradit slovem magnetický a sledujte kdy si toho dotýčný všimne.

Můžete vyzkoušet ještě další magický program, při zatřesení vám dá micro:bit náhodnou odpověď ano/ne. Můžete se ho tak dotazovat na mám/nemám.

Co jste se naučili

Programovat vstupní události např. stisk tlačítka nebo zatřesení microbitem. Naučili jste se odečítat fyzikální veličiny z reálného světa – teplota, světlo, hluk a magnetické pole. Vše toto později využijete v tvorbě zabezpečovacích zařízení.



Pracovní list 3 Komunikace mezi dvěma micro:bity

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem MakeCode anebo s přístupem na internet
- ✓ 2 Micro:bity nejlépe V2
- ✓ Propojovací USB kabel
- ✓ Doporučuji pro oba Micro:bity použít Baterý Box – pro testování spojení v „terénu“

A jdeme na to

V této a následující hodině musí být vytvořeny nejméně dvoučlenné skupiny vybavené dvěma micro:bity

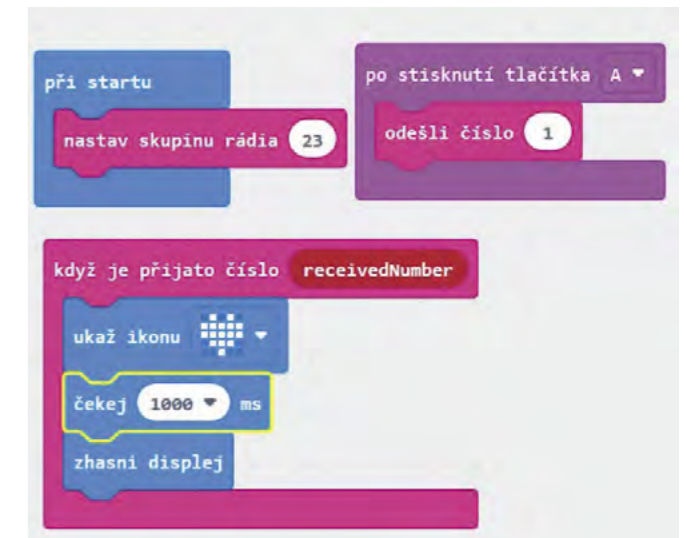
Na úvod naprogramujte takovýto jednoduchý prográmk na obou micro:bittech ve skupině.

Všimněte si při testování v editoru, že po stisku tlačítka A se zobrazí druhý micro:bit (příjemce), který bude vykonávat činnost při obdržení signálu. Je důležité, aby měly různé skupiny různou skupinu rádia v bloku „při startu“, jinak si vzájemně polezete do zelí. Signál by měly obdržet všechny micro:bity se stejnou skupinou.


Zkoušejte si posílat vzájemně signál. Připojte micro:bity na battery pack a naopak je odpojte z USB kabelu a vezměte je na procházku ven a vyzkoušejte dosah signálu.

Pro testování si také můžete nahrát na jeden micro:bit, který ponecháte připojený u počítače následující program a jít „na procházku“ jen s jedním micro:bitem na kterém ponecháte předchozí program.

Můžete zmenšit dobu po kterou je zobrazena ikona se srdcem, abyste lépe odlišili jednotlivé impulsy.



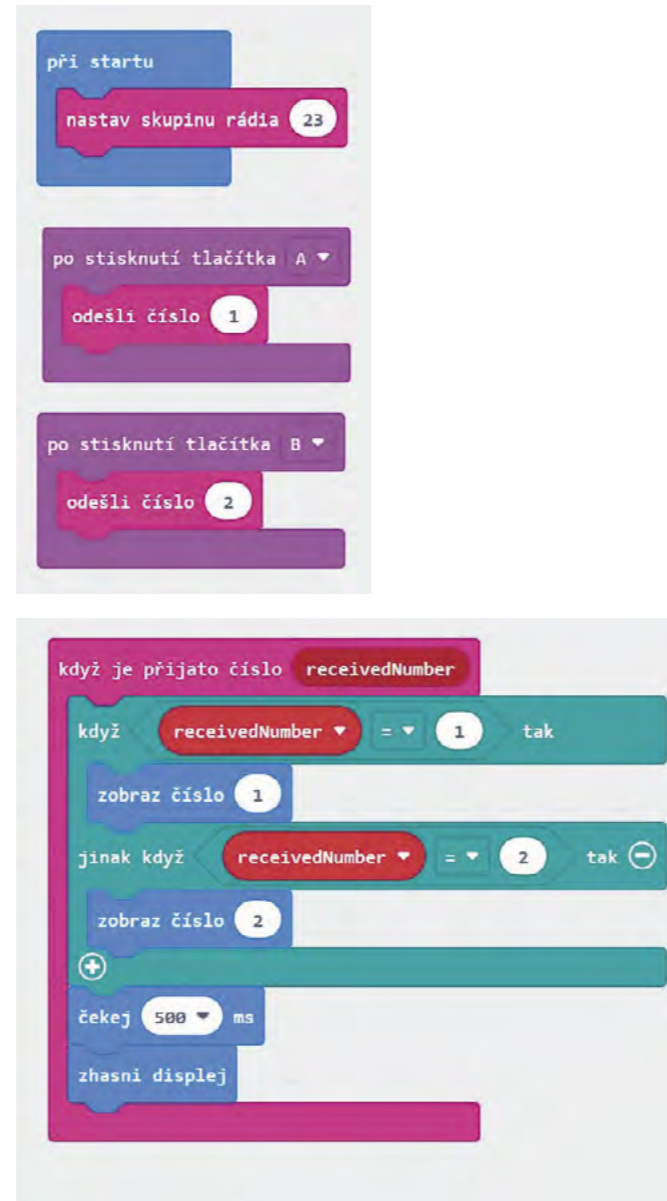
Pozor. Pokud připojujete Micro:bit k počítači, vždy nejdříve odpojte Battery Pack, aby nedošlo k poškození Micro:bitu vyšším napětím.

Naeditujte složitější případ, kdy se posílá jiné číslo při stisku A a jiné při stisku B. Přijímající micro:bit pak podle přijatého čísla rozhodne o své činnosti. 

Máte-li V2, upravte příklad tak, že třetí číslo se pošle při stisku loga.

Upravte kód tak, že význam při poslání signálu z jednoho micro:bitu může být např. „Půjdem na hřiště?“, „Půjdem se učit“, „Máme úkoly“ a odpovědi zpět např. „Ano“, „Ne“ a „Nevím“. V takovémto případě už bude nutné mít v micro:bitech různý kód.

Uvědomte si nebezpečí této komunikace. Pokud útočník zná použitou skupinu a význam kódů, může konverzaci odposlouchávat. Navíc může i do komunikace vstoupit a způsobit zmatení účastníků.



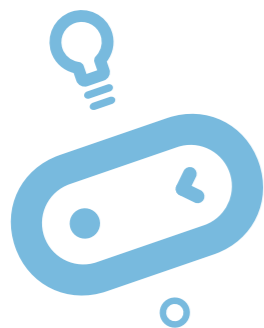
```

při startu
  nastav skupinu rádia 23

po stisknutí tlačítka A
  odešli číslo 1

po stisknutí tlačítka B
  odešli číslo 2

když je přijato číslo receivedNumber
  když receivedNumber = 1 tak
    zobraz číslo 1
  jinak když receivedNumber = 2 tak
    zobraz číslo 2
  +
  čekej 500 ms
  zhasni displej
    
```



Pracovní list 4


Tvorba zabezpečovacího zařízení

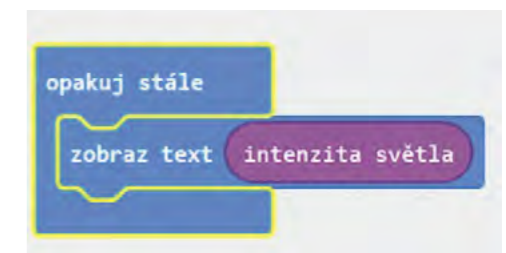
Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem MakeCode anebo s přístupem na internet
- ✓ 2 Micro:bity nejlépe V2
- ✓ Propojovací USB kabel
- ✓ Doporučuji pro oba Micro:bity použít Battery Box – možnost umístění i mimo počítač

Ajdeme na to

Vytvoříme jednoduché zabezpečovací zařízení, které bude reagovat na intenzitu světla.

Vyzkoušejte si nejprve na jednoduchém prográmku rozdíl ve hodnotě intenzity světla mezi zhasnutím a rozsvícením (zavřenými a otevřenými žaluziemi). 




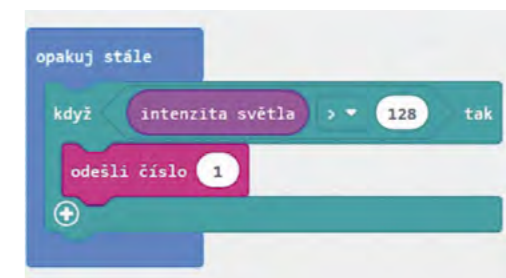
```

opakuj stále
  zobraz text intenzita světla
    
```

Stanovte si hodnotu, která pro vás bude znamenat dělicí hodnotu mezi světlem a tmou.


Nyní již vytvoříme jednoduché zabezpečovací zařízení. Hodnotu 128 můžete nahradit odpozorovanou hodnotou z předchozího programu. Tentokrát budeme nahrávat rozdílné programy na každý Micro:bit.


- ✓ Vysílač – hlídá intenzitu světla a při rozsvícení (zhasnutí) vyšle signál – poplach. Kód by mohl vypadat např. takto. 



```

opakuj stále
  když intenzita světla > 128 tak
    odešli číslo 1
  +
    
```

- ✓ Přijímač – dostane info, že došlo k incidentu a vyhlásí poplach. Kód by mohl vypadat například takto. 



```

když je přijato číslo receivedNumber
  když receivedNumber = 1 tak
    zobraz text Poplach
    play sound twinkle
  +
    
```

■ Zkuste upravit kód přijímače, tak aby opakoval varování, dokud není např. stisknuto tlačítko.

■ Nyní vytvořte libovolné zabezpečovací zařízení pomocí dvou micro:bitů.

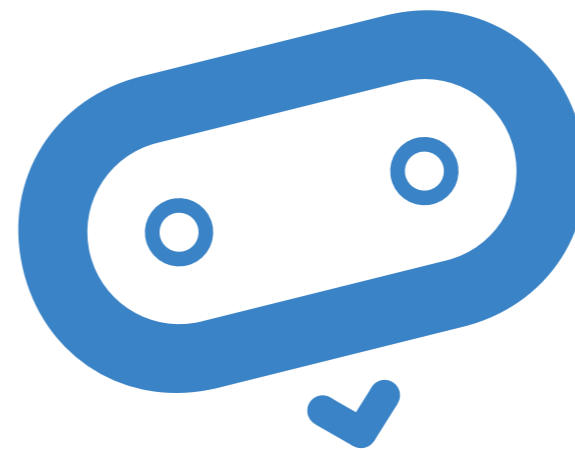
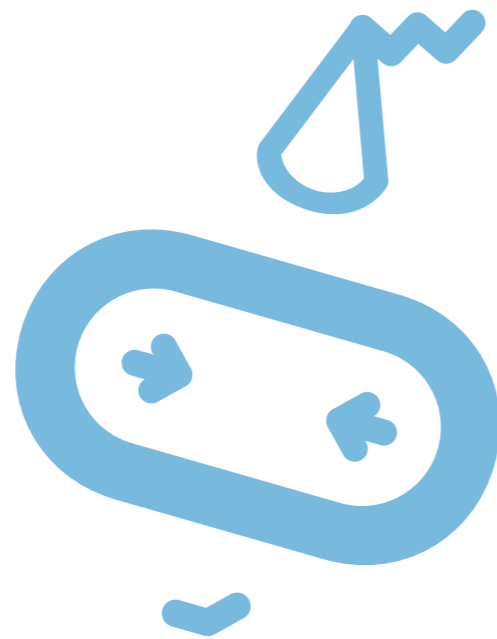
Můžete vyzkoušet různé typy událostí:

- ✓ stisk tlačítka (naopak i uvolnění tlačítka)
- ✓ změna intenzity světla
- ✓ změna teploty
- ✓ změna intenzity magnetického pole
- ✓ audio signál (zvuk)
- ✓ pohyb micro:bitu (zatřesení)

Co jste se naučili

Znalosti získané v tomto kurzu jsou vlastně znalostmi ze světa robotiky. Umíte nyní sledovat své okolí a zjišťovat jeho změny – zde se jedná o analogii s lidskými smysly. Na jednu stranu sice nemáte smysly „chuť a zrak“, ale máte zde naopak smysl intenzita magnetického pole. Na tyto události nyní umíte reagovat zprávami na displej či zvukem. Umíte rovněž komunikovat s jiným Micro:bitem na dálku.

Je třeba si uvědomit, že ačkoliv tento kurz vás mnoho naučil, záleží pouze na vás, jak se získanými dovednostmi naložíte a zda je dále budete prohlubovat pomocí dalších kurzů či další literatury.



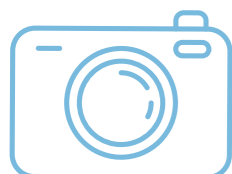


GRAFIKA – PRÁCE S FOTOGRAFIÍ PRO ZŠ

Základní instrukce

Tento kurz je určen zejména pro žáky druhého stupně základní školy nebo nižší ročníky osmiletých gymnázií. Autor kurzu je amatérským fotografem s dlouholetými zkušenostmi s analogovými i digitálními fotoaparáty. Řadu let se zabývá též CAD počítačovou grafikou a 3D technologiemi. Kurz lze použít i pro výuku na středních školách, vždy záleží na přístupu a invenci a zkušenostech lektora.

Časovou dotaci kurzu nelze jednoznačně určit. Záleží do značné míry na dosavadních zkušenostech žáků či studentů s fotografováním a s úpravou fotografií. Výslednou dotaci lektor stanoví po prostudování materiálů ke kurzu.



Většina z nás v běžném životě vytváří řadu fotografií pro pracovní či dokumentární potřeby, často jde také o zážitkové fotografie (kultura, sport, dovolená), případně i umělecké fotografování. Absolventi kurzu zjistí, že i základní nastavení parametrů při fotografování má zásadní vliv na výslednou fotografii a že ne vždy stačí jen mačkat spoušť fotoaparátu – ať použijí běžný mobilní telefon, poloautomat či drahý profesionální fotoaparát. Základní úprava digitální fotografie totiž začíná již při jejím pořizování, takže stojí za to věnovat čas nastavení fotoaparátu. Ze špatné fotografie totiž dobrou neudělá ani PhotoShop.

Finálním výstupem kurzu by tak měly být fotografie upravené pro konkrétní použití (od pouhého prohlížení v mobilu, přes sdílení na webu a sociálních sítích až po tisk fotografie na papír či jiná média).

Příklad rozdělení kurzu a doporučená časová dotace:

1. Základní fotografické pojmy.

- I když je možné každý fotoaparát používat rovnou i bez znalosti teoretických základů, striktně doporučuji seznámení žáků alespoň se základními pojmy. To umožní daleko širší využití fotoaparátu, zejména pak při fotografování v nestandardních a extrémních podmínkách.
- S přihlédnutím ke zkušenostem žáků by tak měla stačit jedna vyučovací hodina (45 minut). Pokud jsou zkušenosti minimální, bude lépe na teorii rezervovat dvě vyučovací hodiny (90 minut)

2. Pořízení fotografií a nastavení fotoaparátu v konkrétních podmínkách; fotografování „z ruky“ vs. použití stativu – praktické cvičení.

- Zde je třeba rozhodnout, zda toto praktické cvičení bude probíhat v učebně či v budově nebo jestli se vypravíme „do terénu“. Je nutné uvažovat čas přípravy a zásadní je také samotné technické vybavení žáků. Předpokládá se, že každý žák (nebo alespoň velká většina) má k dispozici smartphone s fotoaparátem. Doporučuji do cvičení zařadit i klasický fotoaparát (nejlépe školní, s opatrností lze využít i osobní techniku lektora či fotoaparát některého žáka).
- Doporučená dotace jsou dvě vyučovací hodiny (90 minut).

3. Úprava fotografií podle konkrétní potřeby použití a publikování fotografií – praktické cvičení.

- Je třeba, aby každý žák měl k dispozici samostatné počítačové pracoviště s nainstalovaným fotoeditorem, v tomto kurzu je využit **PhotoFiltre v. 6. 5. 3 v české lokalizaci**.
- Je na rozhodnutí lektora, do jaké hloubky se bude při úpravách fotografií pouštět a kolik času bude mít k dispozici. Doporučuji minimálně dvě vyučovací hodiny (90 minut), pro větší komfort je vhodné věnovat cvičení tři až čtyři vyučovací hodiny (135–180 minut).

Minimální doba absolvování kurzu je pět vyučovacích hodin, doporučený počet je osm vyučovacích hodin. Výsledná doba se může v průběhu kurzu upravit podle potřeby.

Pro část 1 je použita metoda výkladu s ukázkami, části 2 a 3 budou realizovány jako praktické cvičení v terénu, resp. v počítačové učebně. Podle možností lektora, počtu žáků a také podle dostupného vybavení mohou praktická cvičení probíhat individuálně či ve dvojicích.

Publikování fotografií lze případně realizovat formou projektové výuky, kdy každý žák nebo každá dvojice prezentuje výsledky své práce (vytištěné či publikované fotografie, fotoalbum).

Pro samotný kurz je doporučeno, aby každý žák měl k dispozici mobilní telefon s fotoaparátem (nabitý na 100 % a s dostatečným volným prostorem na paměťové kartě). Lektor by měl být vybaven též alespoň jedním digitálním fotoaparátem umožňujícím manuální nastavení parametrů, případně s výměnnými objektivy.

Dalším doporučeným vybavením je klasický stativ doplněný o držák na mobilní telefony. Dále je vhodný ministativ (s možností upevnění na různých místech – kameny, pařezy, stromy apod.) pro práci v přírodě.

Úprava fotografií probíhá v počítačové učebně vybavené dostatečným počtem počítačů s nainstalovaným fotoeditorem.

Jako podkladů pro výuku lze využít nepřeberné množství publikací či internetových zdrojů. Doporučené publikace jsou uvedeny v použitých zdrojích v závěru tohoto dokumentu.



Teoretická část k dané problematice

Fotoaparátem pro účely tohoto kurzu rozumíme digitální zařízení umožňující pořizování fotografií. Nejčastěji se tedy bude jednat o „chytrý“ mobilní telefon vybavený aplikací Fotoaparát. Teoretické pojmy však budou vysvětlovány na klasickém fotoaparátu, přičemž ty základní vycházejí ještě z analogových fotoaparátů. Předpokládá se alespoň základní znalost, takže zde nebude popisována konstrukce fotoaparátu.

Z teorie fotografování je třeba alespoň stručně vysvětlit několik skutečně základních pojmů, bez jejichž pochopení nelze správně využívat rozšířené funkce fotoaparátu. Patří sem zejména **clona, rychlost závěrky, citlivost, expozice, ohnisková vzdálenost, hloubka ostrosti a teplota světla – vyvážení bílé barvy**.

Tyto pojmy budou vysvětleny níže a na tomto základě pak bude probíhat praktické cvičení na pořizování fotografií.

Clona

Nachází se v objektivu fotoaparátu a je to vlastně otvor, jehož velikostí řídíme množství světla procházejícího objektivem, resp. dopadajícího na snímač fotoaparátu nebo na film u analogových fotoaparátů. Aby se tento údaj dal nějak konkretizovat, označujeme clonu písmenem **f**. Změnou clony o jeden stupeň dosáhneme dvojnásobku, resp. poloviny množství dopadajícího světla.



Obr. 1 – Příklad nastavení clony objektivu

Mezinárodní číselná řada s hodnotami clonového čísla:

1 – 1,4 – 2 – 2,8 – 4 – 5,6 – 8 – 11 – 16 – 22 – 32 – 45

Rychlost závěrky – doba expozice

Pro řízení množství světla dopadajícího na snímač (film) máme ještě další možnost. Můžeme nastavit dobu, po kterou bude světlo na snímač dopadat. Pojem **rychlost závěrky** je vžitý ještě z analogových fotoaparátů, můžeme se ale setkat např. i s označeními **doba expozice**, **osvitová doba** nebo **čas**.

Jako u clony, i zde je stanovena mezinárodní řada doby expozice v sekundách a ve zlomcích sekund:

1 – 1/2 – 1/4 – 1/8 – 1/15 – 1/30 – 1/60 – 1/125 – 1/250 – 1/500 – 1/1000

A stejně jako u clony platí, že změna rychlosti závěrky o jeden stupeň v řadě má za následek dvojnásobek, resp. polovinu množství dopadajícího světla na snímač (film).

Pro fotografování v noci (noční krajina, hvězdy apod.) je třeba nastavit daleko delší expoziční dobu, může jít i o desítky sekund či minuty.

Citlivost

Na správnou expozici fotografie má vliv také vhodná volba citlivosti. Označuje se číslem na stupnici ISO a původně vychází z citlivosti filmu, která bývala označována také jako ASA. Toto číslo je výrazně uvedeno na každé krabičce s filmem, protože zásadním způsobem ovlivňuje použití konkrétního filmu. Jedná se o tolik vžitě hodnoty, že se číselná řada označení citlivosti ISO používá i u digitální fotografie.

Základní číselná řada citlivosti ISO:

25 – 50 – 100 – 200 – 400 – 800 – 1600 – 3200 – 6400 – 12000

Pro jemnější nastavení existují u digitálních fotoaparátů i mezistupně, např. 250, 360. A opět platí, že dvojnásobná citlivost snižuje expoziční dobu na polovinu, tzn., že pro stejný osvit snímače je pak potřeba poloviční množství světla.

Je dobré vědět, že s nastavením vyšší citlivosti se na fotografii zvětšuje také míra tzv. šumu a zmenšuje se prokreslení detailů.



Expozice

Na základě výše uvedených pojmů můžeme pracovat s výsledným pojmem **expozice**. Ve výsledku jde tedy zejména o to, aby fotografie nebyla podexponovaná (moc tmavá) nebo přexponovaná (moc světlá). Špatnou expozici již nelze na počítači dodatečně opravit a je tak velmi důležité nastavit správnou expozici ještě před pořízením fotografie. Toho dosáhneme vhodným nastavením již dříve zmíněných parametrů, tedy clony, rychlosti závěrky a citlivosti. Platí zde tzv. reciprocita, tedy např. když otevřeme clonu, musíme zkrátit expoziční dobu nebo snížit nastavení citlivosti a obráceně.

Nelze jednoznačně říct, jak který parametr nastavit. U expozice totiž nejde jen o to, zda bude fotografie podexponovaná či přexponovaná, i když to je zcela zásadní věc. Nastavením expozice ovlivňujeme také hloubku ostrosti a ostrost objektů v pohybu. To bude vysvětleno níže.

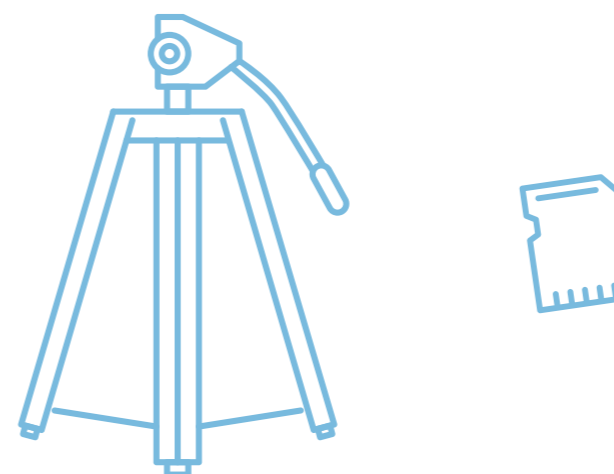
Ohnisková vzdálenost

Udává se v mm a zjednodušeně řečeno je to vzdálenost čočky objektivu od jejího ohniska. Je nutno si uvědomit, že čím větší je ohnisková vzdálenost, tím menší je **úhel záběru**. Také velikost ohniskové vzdálenosti je parametrem přejatý z analogové fotografie. Hraje zde totiž roli také velikost filmového políčka (u analogových fotoaparátů). Snímače digitálních fotoaparátů mají ale menší velikost než film a tak jsou zde i jiné hodnoty ohniskové vzdálenosti. Abychom mohli i nadále používat vžitě hodnoty ohniskové vzdálenosti, je třeba u digitálních fotoaparátů provést přepočítání (to však již přesahuje obsah tohoto kurzu).

Pro naše potřeby bude stačit, když budeme podle ohniskové vzdálenosti rozlišovat tři typy objektivů a to:

- ✓ **Základní** – ohnisková vzdálenost (přepočtená na kinofilm) je 50 mm.
- ✓ **Širokoúhlý** – s ohniskovou vzdáleností v rozsahu cca 12–40 mm – používáme hlavně pro fotografování krajin.
- ✓ **Teleobjektiv** – ohnisková vzdálenost se pohybuje ve velkém rozsahu od cca 60 mm až např. do 1000 mm. Používají se např. při fotografování vzdálených objektů v přírodě, sportu, umění a architektuře.

Naprostá většina objektivů má možnost v určitém rozsahu měnit ohniskovou vzdálenost (tzv. zoomovat), i když existují i některé objektivy s pevnou, neměnnou ohniskovou vzdáleností (např. tzv. portrétové objektivy používané ve fotoatelierech).



Hloubka ostrosti

Kromě správné expozice má na kvalitu snímku vliv také ostrost. Fotografie musí samozřejmě být ostrá, ale ne vždy musí být ostré vše, co se na fotografii nachází. Míra toho, co je ostré a co ne, se nazývá hloubka ostrosti. Velkou hloubku ostrosti je vhodné použít typicky u fotografií krajiny či zátiší, malou hloubku ostrosti naopak např. při fotografování květin, portrétů apod.

Hloubku ostrosti můžeme ovlivnit třemi způsoby:

- ✓ nízkým nastavením clony
- ✓ použitím objektivu s vyšší ohniskovou vzdáleností (teleobjektivu)
- ✓ volbou vhodné vzdálenosti, ze které pořizujeme fotografii (ne vždy je to možné – třeba u zvířat, ale velmi dobře je tento způsob použitelný např. u fotografování květin)



Obr. 2 - Příklad fotografie s malou a velkou hloubkou ostrosti

Teplota světla – vyvážení bílé (wb)

U většiny běžných fotografií se nemusíme nastavením vyvážení bílé barvy (white balance) zabývat, ale pokud fotografujeme v nestandardních světelných podmínkách (šero, osvětlení objektů různě barevnými zdroji světla, přesvětlená scéna – sníh, pláž), je vhodné upravit barevnou teplotu světla ještě před pořízením fotografie. Teplota světla se udává ve stupních Kelvina [°K]. Nelze jednoznačně říci, jak správně teplotu nastavit, časem získáte svou zkušenost. Záleží také na konkrétním fotoaparátu.

Profesionální fotografové používají k vyvážení bílé zaměření objektivu na neutrálně šedou destičku (18% šedá), kterou vždy nosí s sebou, lze použít i bílý papír. Po zaměření stiskneme na fotoaparátu tlačítko či volbu WB.

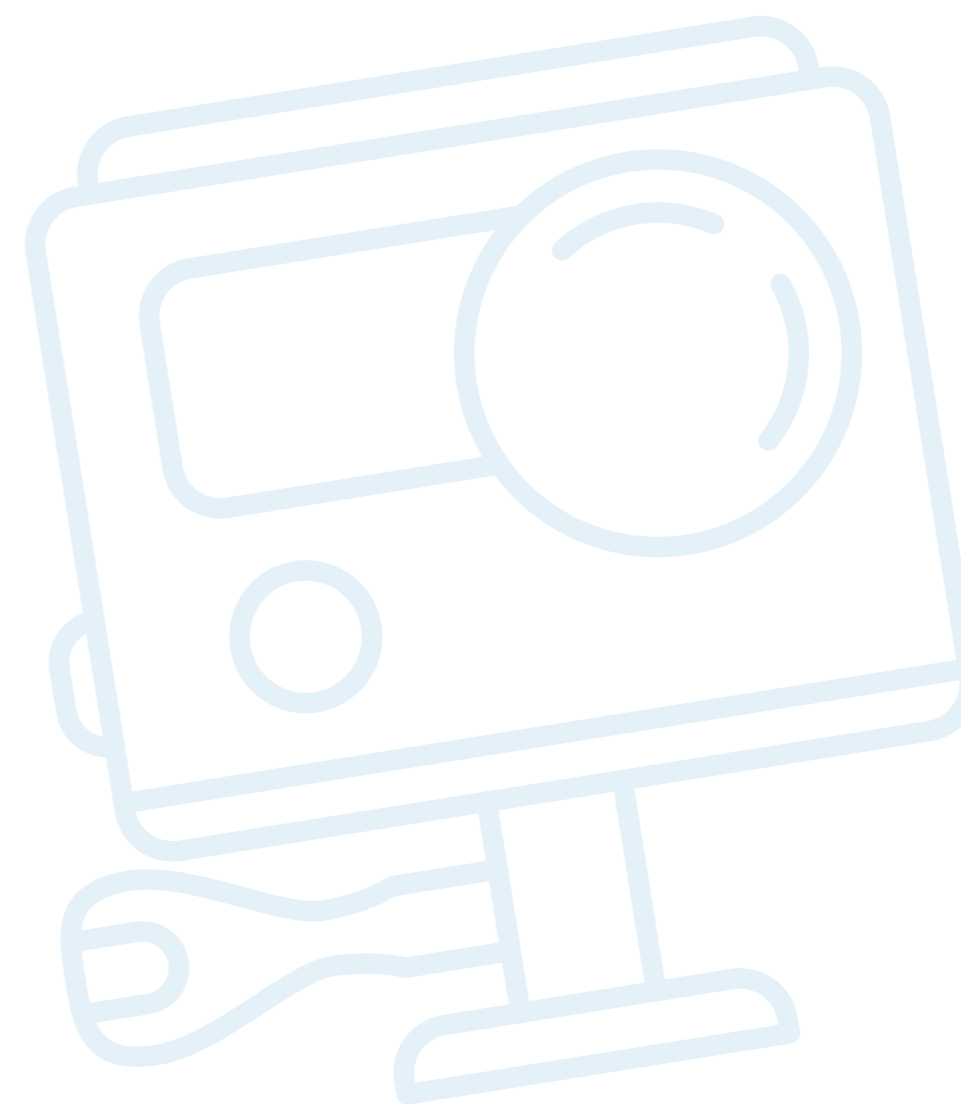
Příklady z praxe

Materiál vznikl na základě vlastních pedagogických zkušeností, zejména pak vychází z případů, kdy žáci pořizují snímky do vlastních seminárních a maturitních prací. Je s podivem, jak málo žáků je ochotno věnovat alespoň minimum času prostudování či alespoň přečtení naprostých základů fotografování. Podle toho pak také žákovské fotografie ve velké většině vypadají.

V době, kdy většina fotoaparátů funguje s automatickým nastavením, si málokdo uvědomuje, co všechno se dá nastavením parametrů ovlivnit a jak málo stačí k tomu, aby pořízená fotografie byla svou kvalitou minimálně o úroveň výše.

Absolvováním tohoto kurzu žáci mohou získat návyky, jejichž osvojením budou schopni pořizovat kvalitní fotografie a to jak klasickým digitálním fotoaparátem, tak i běžným fotoaparátem mobilního telefonu.

Zásadní je pro tento kurz invence a nasazení pedagoga a správná volba vzorových fotografovaných objektů a scénérií. Pokud se podaří žáky vtáhnout do skupinové práce, může kurz nabýt částečně i soutěžní či prezentační charakter.



Metodická a didaktická část

V tomto cvičení nejsou záměrně uvedeny konkrétní hodnoty jednotlivých parametrů. Jde o to, aby si žáci vyzkoušeli vliv nastavení základních parametrů fotoaparátu u typických příkladů a před stisknutím spouště se alespoň zamysleli nad tím, co fotografují a pro jaké účely. Zručnost lze získat pouze cvikem. V dnešní době nemusíme naštěstí řešit náklady na film a stačí nám tak pouze dostatečně velká paměťová karta a nabitá baterie fotoaparátu. Ve cvičení také neřešíme kompozici snímku, to by bylo na samostatný kurz.

Nastíněné příklady jsou ilustrativní a nic nebrání tomu, abyste si je přizpůsobili podle konkrétní situace.

Nepříliš vhodný postup je pořizování velkého množství stejných či podobných záběrů s tím, že „on alespoň nějaký snímek vyjde“ či „později si to přeberu a vymažu ty nepovedené“.

Daleko efektivnější je chvilku se zamyslet a věnovat pár vteřin nastavení. Fotografie pak bude stát za to.

Základní doporučení než začneme fotit

Co je dobré ověřit a nastavit na fotoaparátu ještě před volbou jednotlivých parametrů:

- ✓ **Datum a čas** – k fotografii nám fotoaparát „přibalí“ také informaci, kdy byl snímek pořízen. Pokud fotíme mobilem, povolte i geolokaci, budete tak mít i informaci, na jakém místě jste fotografovali.
- ✓ **Rozlišení fotografie** – zásadně ovlivní kvalitu fotografie. Pokud můžete, nastavte vyšší hodnoty. Fotografie bude mít větší objem, ale pokud budete potřebovat později rozlišení snížit, bez problémů to provedete při úpravě v počítači. Obráceně to není možné.
- ✓ **Poměr stran** – ne vždy je nejvhodnější širokoúhlý poměr 16:9, často i dnes použijeme klasický poměr 4:3, v některých případech se nám může hodit třeba i čtvercový poměr 1:1. Lze sice později fotografii oříznout, ale uvědomte si, že na širokoúhlých či ultraširokoúhlých záběrech dochází k mnohem většímu zkreslení při okrajích fotografie.
- ✓ **Výběr vhodného objektivu** – toto částečně souvisí s předchozím bodem. U fotoaparátu s výměnným objektivem je třeba předem nasadit ten správný. U novějších mobilních telefonů dnes máme na výběr zpravidla alespoň dva zadní objektivy a jeden přední (selfie). U zadních objektivů má vždy jen jeden z nich vysoké rozlišení. Druhý bývá ultraširokoúhlý, tzn., že se nám na fotografii sice vejde více (např. celá vysoká věž či široké panorama), vždy je to ale za cenu většího zkreslení a horšího rozlišení výsledné fotografie.

Poznámka: cvičení není návodem, jak používat editor PhotoFiltre. Je opět potřeba, aby si lektor všechny zamýšlené činnosti předem vyzkoušel, vybral konkrétní funkce a zároveň stanovil potřebný čas pro zvládnutí cvičení.

Doporučené pomůcky

Kurz není z pohledu žáků nikterak náročný na pomůcky. Jak již bylo řečeno, vystačíme i s mobilním telefonem. Na druhou stranu – pokud si žáci přinesou vlastní digitální fotoaparát, budou si moci vyzkoušet více nastavení.

Pro žáky:

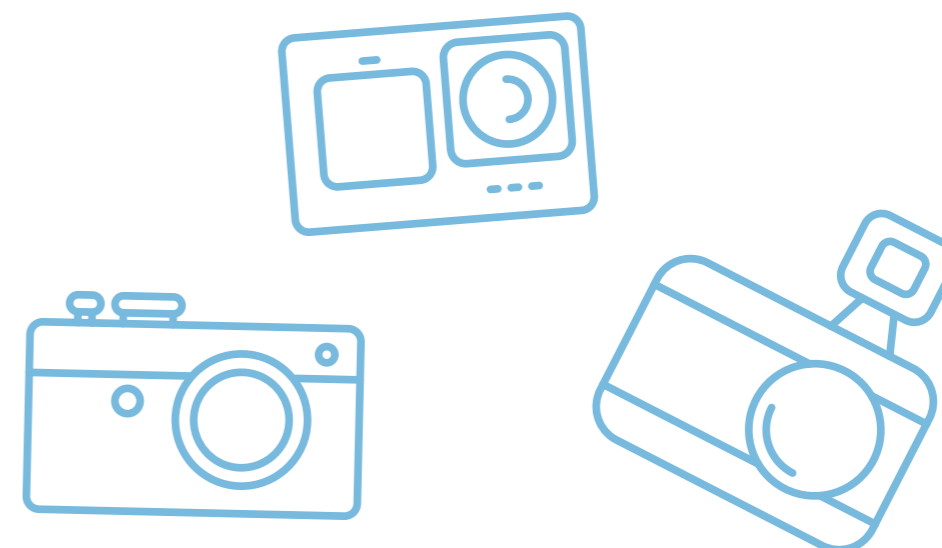
- ✓ Digitální fotoaparát nebo modernější mobilní telefon s fotoaparátem
- ✓ Dobře nabitá baterie, příp. náhradní baterie k fotoaparátu nebo power banka
- ✓ Dostatek místa na paměťové kartě

Pro učitele:

- ✓ Digitální fotoaparát
- ✓ Stativ
- ✓ Náhradní baterie

Pro zpracování pořízených fotografií:

- ✓ Počítačová učebna s dostatečným počtem osobních počítačů či notebooků
- ✓ Běžný fotoeditor – zcela postačí např. volně šiřitelný PhotoFiltre, se kterým jsou také uvedeny ukázky v tomto kurzu



Pracovní list 1

Pořízení fotografií

Při cvičení doporučuji pořizovat dva základní typy fotografií:

- ✓ **Krajina či scénérie budov** – volná příroda nebo město (vesnici)
- ✓ **Detail** – nejlépe nepohybující se objekt (rostlina, skála, zajímavá struktura dřeva...)

Konečná volba fotografovaných objektů je ale na lektorovi. Stejně tak pořadí fotografování objektů a nastavování škály konkrétních hodnot jednotlivých parametrů. Je nanejvýš vhodné si cvičení provést předem samostatně a až následně s žáky. Získáte tím orientační představu o časové náročnosti a stanovíte si počet pořizovaných snímků.

Nemá smysl fotografovat se všemi možnými nastaveními. Doporučuji vždy max. 3 záběry na změnu jednoho parametru, tzn. postupně nastavit obě krajní hodnoty (minimální a maximální) a poté nějakou hodnotu přibližně uprostřed řady.

Výsledkem by měla být vždy správně exponovaná fotografie, tedy nepřexponovaná ani nepodexponovaná. Je třeba si uvědomit, že změnou jednoho parametru se nám mohou změnit i některé ostatní parametry. Vždy ale jeden z nich bude mít prioritu.

U každého snímku si do zápisníku poznamenejte nastavené parametry – jednak ten, který jste změnili primárně a také ostatní, které mohl automaticky upravit fotoaparát. Ne vždy však všechny parametry budete moci zjistit (zejména u mobilů), některé hodnoty lze ale dodatečně v počítači vyčíst z hlavičky již pořízené fotografie.

Priorita clony

Jak již bylo řečeno, s otevřenou clonou dosáhneme malé hloubky ostrosti, se zavřenou naopak hloubky velké. Volte tedy pro stejný záběr nízké clonové číslo (např. 2,8), vysoké clonové číslo (např. 22) a poté někde mezi (např. 8). Fotoaparát sám nastaví vhodnou dobu expozice.

Je třeba si uvědomit, že příliš dlouhá doba expozice (1/20 a delší) má za následek rozmazání snímku i pouhým chvěním ruky, proto při delších expozičních dobách používáme stativ.

U mobilních telefonů bývá volba clony nahrazena nastavením jasů. Navíc, protože jsou mobilní telefony osazeny širokouhlým objektivem, nastavení clony nemá takový efekt jako u fotoaparátu. Doporučuji proto použít klasický digitální fotoaparát.

Priorita expoziční doby

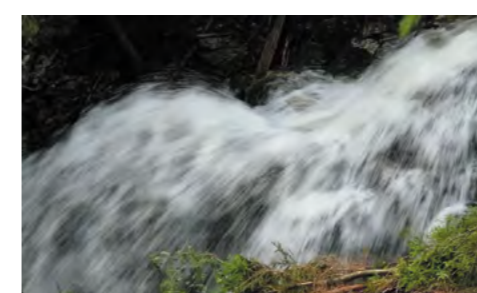
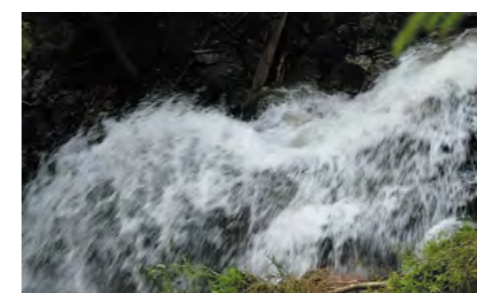
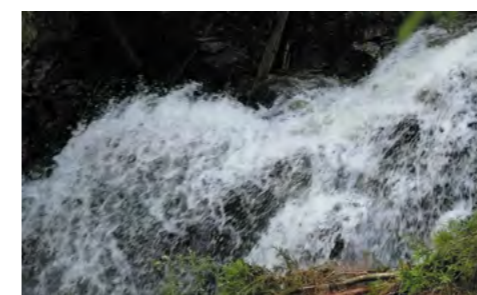
Je několik situací, kdy je lépe upřednostnit čas před nastavením clony. Jde zejména o **nastavení krátkého času** (< 1/250 s) při:

- ✓ fotografování **rychle se pohybujících objektů** (sport, rychle se pohybující zvířata)
- ✓ fotografování **tekoucí či stříkající vody, deště a sněžení** (když chceme zachytit vodu ostře)
- ✓ **použití teleobjektivu** – pokud fotíme na větší ohniskovou vzdálenost z ruky bez stativu, budeme potřebovat velmi krátký čas (< 1/500 s)

Pokud ale záměrně chceme mít objekt rozmazaný – typicky při uměleckých záběrech či při potřebě zdůraznit pohyb, nastavíme čas delší (> 1/100 s).

Zde tedy opět volte pro stejný nebo podobný záběr (voda stříkající z kašny nebo peřeje na potoce či jedoucí automobil, běžící osoba apod.) tři různé expoziční doby: např. 1/500 s, 1/100 s a 1/20 s. Při delších dobách používejte stativ.

Fotoaparát si při vámi nastavené expoziční době sám nastaví clonu tak, aby byla dosažena správná expozice fotografie.



Obr. 3 – Vliv expoziční doby na výslednou fotografii (nastavené časy 1/125, 1/60 a 1/25 s)

Vhodné nastavení citlivosti

Pro další tři stejné snímky si zkuste nastavit tři různé hodnoty citlivosti (např. ISO 100, ISO 800 a ISO 6400) – krajní hodnoty budete možná muset upravit, pokud by vás fotoaparát „nepustil“ např. s příliš velkou nastavenou citlivostí. U takto pořízených snímků budeme později při jejich zvětšení na velkém displeji počítače zkoumat vliv nastavení citlivosti na šum fotografie.

Práce s hloubkou ostrosti

Vhodným nastavením clony, ohniskové vzdálenosti a přiblížením se k fotografovanému objektu se pokuste vyfotografovat tentýž snímek s malou a poté s velkou hloubkou ostrosti. Zde má pochopitelně smysl pořídit pouze dva snímky.

Pro inspiraci – může jít třeba o snímek sochy či kašny na náměstí, kdy domy v pozadí budou ostré nebo rozostřené. Nebo se pokuste o fotografii záhonu květin, kdy na jednom snímku budou ostré všechny květiny, na druhém třeba jen jedna v popředí.

A co teď s tím?

Fotografie máme pořízeny, údaje nastavených parametrů zaznamenány, takže po návratu z terénu je třeba snímky zkopírovat do počítače pro následnou úpravu.

Z karty odstraňujeme originální fotografie až tehdy, když jsme si jisti, že už je nebudeme potřebovat a že jsou bezpečně uloženy na jiném místě.

Pravděpodobně jste si všimli, že fotoaparáty disponují i přednastavenými režimy pro fotografování různých scén (např. sport, krajina, noční krajina, portrét, sníh, pláž apod.). Tyto režimy můžeme pochopitelně s výhodou využít, když např. nemáme čas na zdouhavé nastavování parametrů a objekt našeho fotografování by mezitím zmizel nebo bychom přišli o příznivé světlo v krajině.

Vždy bychom však měli být schopni danou scénu zachytit pomocí ručně zadaných parametrů. Ne, že by se režimy scén nedaly používat trvale, to se jistě dají. Ale při ručním nastavení víme přesně předem, co jsme nastavili a jaký výsledek můžeme očekávat. Je to podobné jako u auta s automatickou převodovkou a s ručním řazením. To svezení, když si tam tu rychlost zařadím podle sebe, je prostě jiné ☺.

Co jsme se naučili?

- ✓ Rozumíme významu základních parametrů, které lze na fotoaparátu nastavit.
- ✓ Umíme pořídit fotografie pomocí fotoaparátu či mobilního telefonu s různě nastavenými parametry.
- ✓ Víme, co změna nastavení jednotlivých parametrů na fotografii ovlivňuje.

Pracovní list 2

Úprava fotografií

V tomto cvičení budeme ve fotoeditoru upravovat fotografie získané v předchozím cvičení tak, aby byly připravené pro konkrétní způsob použití.

Fotografie máme již z fotoaparátu zkopírovány do počítače a též jsme je (pro lepší orientaci při práci) vhodně přejmenovali.

Při cvičení budeme provádět:

- ✓ Základní úpravy fotografií
- ✓ Některé speciální úpravy a zvláštní efekty
- ✓ Publikování fotografií

Základní úpravy fotografií

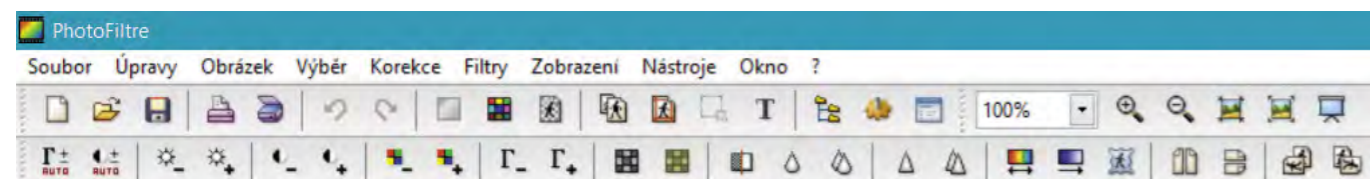
Málokterou fotografii můžeme bez úprav rovnou použít. Velmi často je potřeba provést alespoň některé základní úpravy. V další části cvičení si kromě těchto úprav ukážeme i možnosti editoru v použití řady efektů.

Obecný doporučený postup při úpravách fotografie:

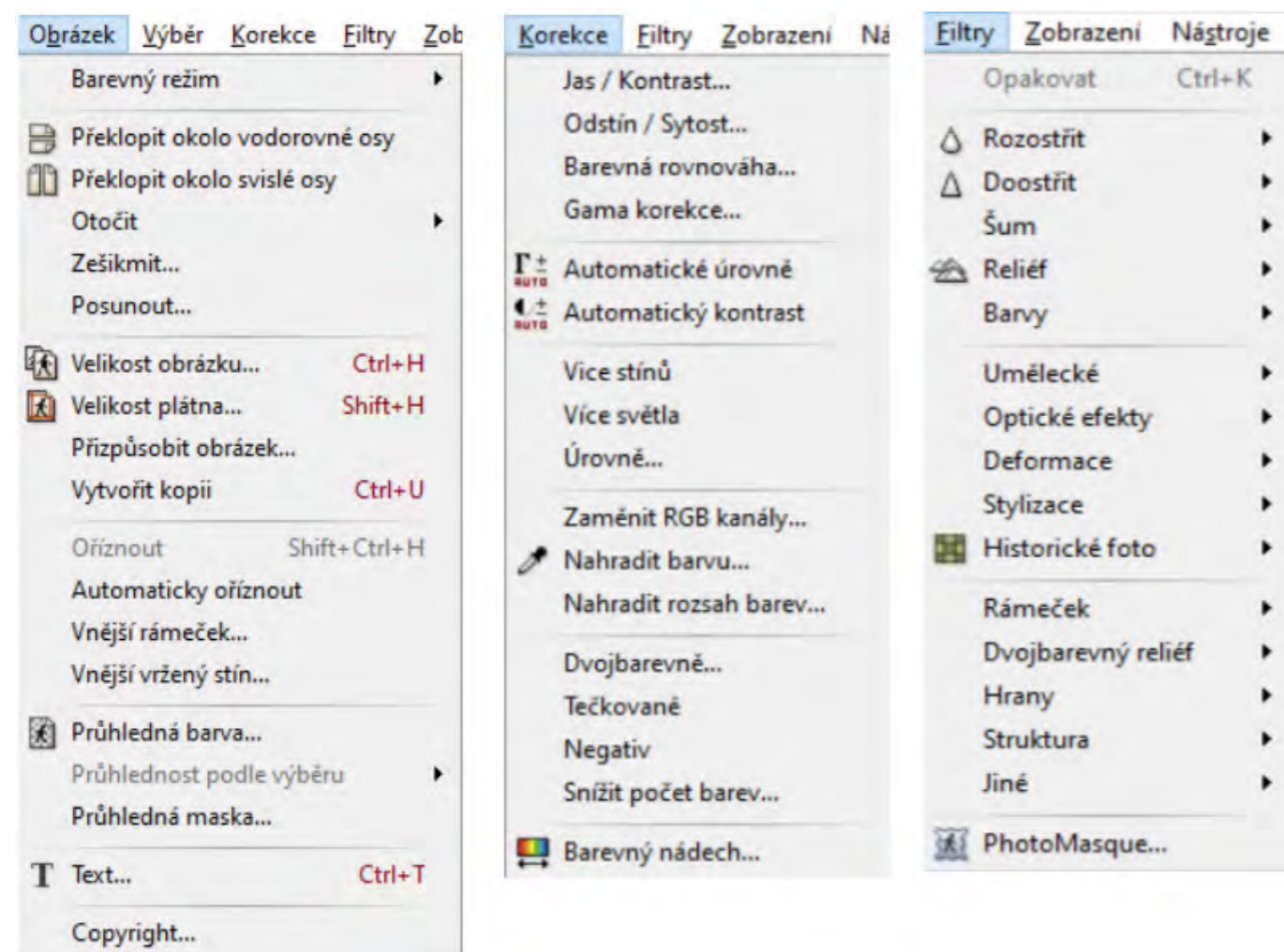
- ✓ Načtení obrázku
- ✓ Správné natočení (otočení po 90° nebo jemné natočení)
- ✓ Oříznutí obrázku na požadované rozměry
- ✓ Úprava jasu, kontrastu, gamma korekce (přizpůsobení rozložení barevných tónů snímku pro vnímání lidského zraku)
- ✓ Úprava barev, průhlednost
- ✓ Provedení dodatečných korekcí a aplikace filtrů
- ✓ Uložení a publikování obrázků

Nejčastěji budeme vybírat funkce za základního menu a z roletového menu (viz následující obrázky).

Ve spodní liště základního menu jsou přístupné často používané funkce. Jde zejména o nastavení jasu, kontrastu, sytosti barev, gamma korekce, převod do černobílé fotografie, rozostření a doostření, barevný nádech a přechod, zrcadlové překlopení, otočení o 90°.



Obr. 4 – Základní menu editoru PhotoFiltre



Obr. 5 – Roletové menu Obrázek – Korekce – Filtry editoru PhotoFiltre

Načtení obrázku

Nejčastěji pomocí příkazu **Otevřít**. Pokud máme obrázek načtený ve schránce, vložíme jej klávesovou zkratkou **Ctrl+V** nebo lépe **Ctrl+Shift+V** jako nový obrázek.

Správné natočení

Pokud je obrázek otočen o 90°, 180° nebo o 270°, použijeme ikonu otočení o 90° po nebo proti směru hodinových ručiček.

Jestliže potřebujeme již jen přesně pootočit, musíme do roletového menu **Obrázek – Otočit – Volně...**

Oříznutí obrázku na požadované rozměry

Tažením kurzoru při stisknutém levém tlačítku myši vybereme okno výřezu. Přesně jej můžeme doladit tažením za hranu již provedeného výběru. Přesnou velikost výřezu můžeme kontrolovat na stavovém řádku na spodní hraně pracovní plochy. Samotný ořez provedeme v menu **Obrázek – Oříznout** nebo klávesovou zkratkou Ctrl+Shift+H. Druhou možností je provést kopii ořezu do schránky **Ctrl+C** a klávesovou zkratkou **Ctrl+Shift+V** ji vložíme jako nový obrázek.

Úprava jasu, kontrastu, gamma korekce

S těmito funkcemi je třeba zacházet opatrně a úpravy provádět po jemných krocích. Editor sice disponuje funkcí **Zpět**, ale je třeba si uvědomit, že po několika krocích si již nemusíme pamatovat, co přesně jsme s obrázkem dělali a budeme muset úpravy provést znovu.

Úprava barev, průhlednost

Úpravou barevnosti můžeme zkorigovat např. špatně nastavené vyvážení bílé barvy na fotoaparátu. Také lze tyto funkce využít při záměrném vytvoření barevného nádechu snímku.

Průhlednost obrázku, resp. nastavení průhledné barvy pozadí obrázku využijeme zejména při vkládání obrázku (např. loga) do jiného obrázku. Pozor – obrázek s průhlednou barvou pozadí můžeme uložit jen jako GIF s 256 barvami.

Některé speciální úpravy a zvláštní efekty

Nyní máme obrázek správně natočený, ve správné velikosti a v požadovaných barvách. Pokud jej budeme chtít „doladit“ zvláštními efekty a filtry, můžeme si vybrat z poměrně velkého množství, které PhotoFiltre nabízí. Nalezneme je v menu **Korekce** nebo **Filtry**.

Nyní můžete popustit uzdu své fantazii, ale **POZOR**:

Při používání korekcí a filtrů mějte na paměti, že méně je někdy více a nesnažte se využít mnoho filtrů najednou. Vždy vycházejte z toho, k čemu bude výsledný obrázek použit.

Publikování fotografií

Pokud jsme s výsledkem spokojeni, tak bychom před publikováním měli v první řadě upravený obrázek uložit. Vždy si však ponechte původní originál. Nikdy nevíte, zda jej nebudete později potřebovat k jiným úpravám.

Uložení

Pro uložení použijte vhodný formát (nejčastěji asi JPG). A také vyberte vhodné úložiště, protože i zde platí pravidlo, že cenu svých dat zjistíte v okamžiku, kdy o ně přijdete. Často používané USB flashdisky jsou vhodnější spíše pro přenášení dat, pro trvalé uložení využijte buď lokální pevný disk, externí pevný disk, síťové úložiště či některou z možností on-line úložišť.

Lektor by měl být předem obeznámen s možnostmi ukládání dat v konkrétních podmínkách počítačové učebny a sítě.

Publikování

Způsobů publikování je celá řada. Může se jednat např. o:

- ✓ použití obrázku do elektronického dokumentu nebo prezentace
- ✓ vložení na webovou stránku
- ✓ použití obrázku na plakát
- ✓ vytištění fotografií na tiskárně a vložení do alba
- ✓ použití obrázku jako textury pro 3D nebo 2D projekt
- ✓ vytvoření pozadí počítačové plochy nebo nějaké videosekvence
- ✓ vytvoření virtuálního fotoalba či fotoarchivu s využitím některé on-line služby (např. Google Fotky, Zonerama apod.). Některé služby mohou být zpoplatněny nebo mít určitá omezení (kapacita, časové omezení...).

Parametry výsledného obrázku by měly odpovídat konkrétnímu způsobu použití. Např. rozlišení při prohlížení na monitoru stačí zmenšit pouze na 72 dpi, pokud ale budeme obrázek tisknout na velký plakát, rozhodně ho necháme v plném rozlišení. Pro takový záměr musíme již na fotoaparátu nastavit co nejlepší kvalitu snímku.

Pro potřeby tohoto cvičení předpokládáme vložení fotografie do fotoalba vytvořeného pomocí on-line služby. Výhodou je možnost prezentace pouze těm osobám, kterým svá alba zpřístupníte. Zde opět záleží na podmínkách a možnostech konkrétní školy. Doporučit se dá např. již zmíněná Zonerama.

■ *Počítejte s určitou časovou rezervou potřebnou na založení účtu.*

Co jsme se naučili?

- ✓ Pořízené fotografie umíme otevřít a upravit na PC ve fotoeditoru.
- ✓ Víme, jak obrázky upravit podle jejich budoucího využití a také je bezpečně uložit.
- ✓ Z pořízených a upravených fotografií umíme vytvořit on-line fotoalbum, které zpřístupníme vybraným osobám.



Co umíme po absolvování kurzu?

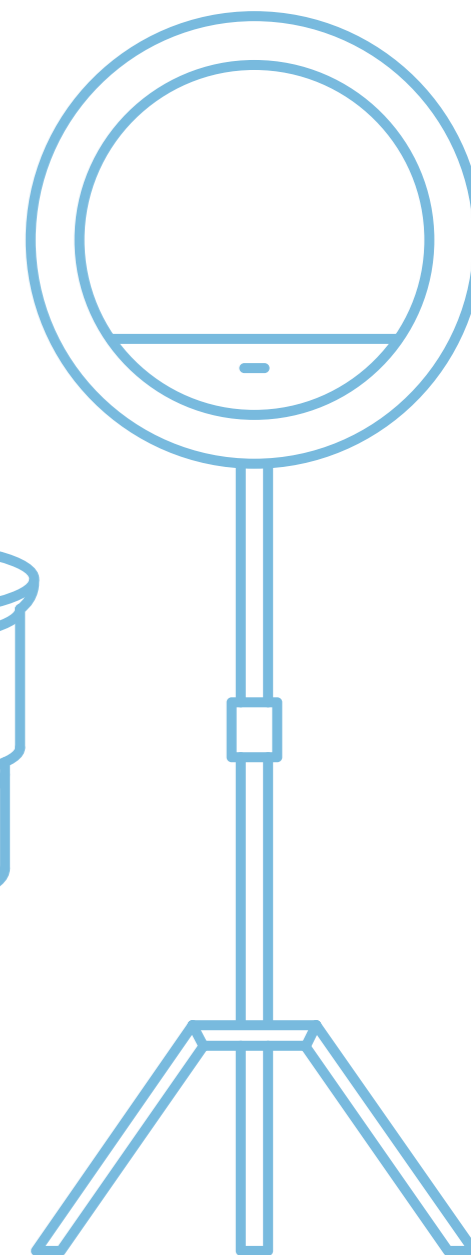
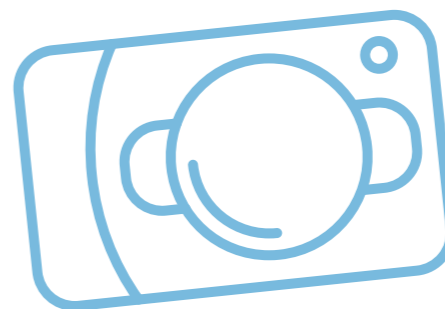
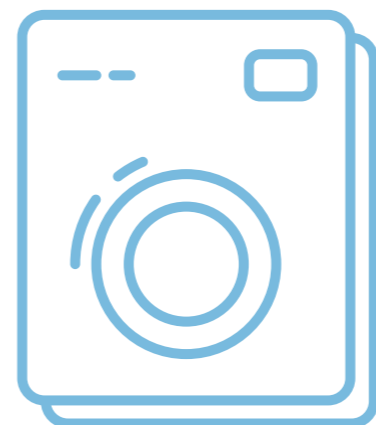
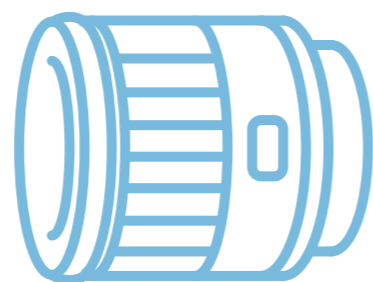
Kurz by žáky měl naučit:

- ✓ porozumět základním parametrům klasického digitálního fotoaparátu a fotoaparátu mobilního telefonu
- ✓ nastavit základní fotografické parametry na fotoaparátu a na mobilním telefonu
- ✓ pořídit fotografie pro konkrétní účel a v požadované kvalitě
- ✓ pořízené fotografie upravit a publikovat

Použité zdroje

MYŠKA, Miroslav. *333 tipů a triků pro digitální fotografie*. 2. vyd. Brno: Computer Press, 2014. ISBN 978-80-251-4308-7.

FOTOGRAFIE – archiv autora





13 TVORBA DIGITÁLNÍHO OBSAHU PRO ZŠ

INTERAKTIVNÍCH
VÝUKOVÝCH
MATERIÁLŮ
A ŽÁKOVSKÝCH
AKTIVIT

Základní instrukce

Kurz je především určen pro učitele prvního a druhého stupně základních škol nebo nižších ročníků víceletých gymnázií.

Časová dotace

6 hodin

Program kurzu

V první části se frekventanti seznámí se **zdroji hotových výukových materiálů** na internetu a především se **základními operacemi** pro vlastní tvorbu žákovských aktivit v programu **SMART Notebook** jako základního software pro plnohodnotné využití interaktivních a dotykových zařízení při výuce (kapitoly 1–5 následujícího textu).

Ve druhé části budou následovat instrukce pro využití jeho **aktuálních doplňků SMART Lab a Response**, které obsahují hotové šablony pro snadnou tvorbu žákovských aktivit včetně možnosti testování žáků. Účastníci se také seznámí i s novou online verzí **SMART Learning Suite**, která od letošního léta 2021 nese název **Lumio** a umožňuje používat žákovské aktivity při distanční výuce (kapitoly 6-7 následujícího textu). V závěru druhé části se ještě podíváme na ukázky interaktivních učebnic, včetně jejich 3D variant s rozšířenou realitou, 3D animacemi a 3D modely.

Použité metody výuky

- ✓ instruktáž
- ✓ samostatná práce
- ✓ projektová výuka

Podklady pro výuku

Kromě následujícího textu to jsou především webové zdroje, jejichž kompletní seznam uvádíme v kapitole „Užitečné webové odkazy k tématu interaktivní výuky“ a obsahují videozáznamy webinářů autora textu, ukázky hotových žákovských aktivit, e-learningový kurz a adresy webových portálů s mnoha aktivitami od učitelů našich škol.



Cílem následujícího textu je představit možnosti tvorby interaktivních a dotykově ovládaných výukových aplikací pro interaktivní tabule či displeje SMART Board, které jsou na našich školách nejrozšířenější. Seznámíme se nejen se základními funkcemi aplikace SMART Notebook z balíčku SMART software, ale i jeho doplňku SMART Lab s hotovými šablonami aktivit a zaměříme se i na velmi užitečnou aplikaci SMART Response pro tvorbu testů, kvízů a dotazníků. V závěru textu si také představíme použití hotových aktivit a testů na vlastních mobilních zařízeních formou online přístupu a tedy jejich použití i při distanční výuce.

Teoretická část k dané problematice

Interaktivní technologie

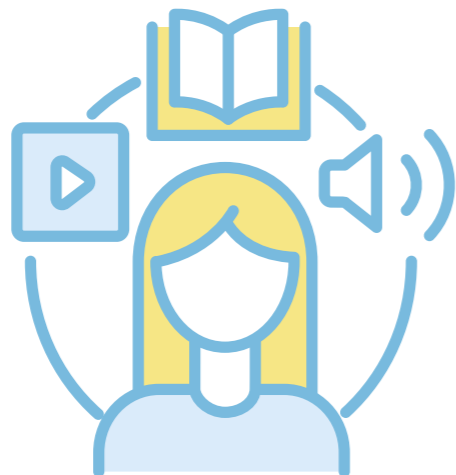
V posledních letech se stále častěji do výuky zapojují moderní technologie, mimo jiné i **interaktivní tabule** a obdobná dotyková zařízení (displeje, tablety a chytré mobilní telefony). V současné době se tyto technologie již vyskytují prakticky ve všech základních a středních školách a ty využívají dotykovou techniku od mnoha výrobců. Nejznámější jsou určitě interaktivní tabule resp. displeje **SMART Board** a **ActivBoard**.

Nejrozšířenější značkou interaktivních tabulí v České republice je SMART Board, dodávaný do našich škol společností **AV media**, který byl zároveň cca před 20 lety i první značkou interaktivních tabulí v České republice. Můžeme pracovat jednak s tradičními interaktivními tabulemi s projektořem, ale také modernějšími (i když mnohem dražšími) nástěnnými dotykovými displeji, moderními tablety či využívat i chytrých mobilních telefonů. Všechny druhy zařízení můžeme ovládat jak tzv. **stylusem** (elektronickým perem), tak i **prstem** a zdarma škola obdrží roční licenci výukového softwaru SMART Notebook i s mnoha doplňky.

Digitální učební materiály (DUM)

Z mnoha výzkumů i prosté zkušenosti nejen autora tohoto textu ale vyplývá, že tuto ne příliš levnou techniku stále velmi mnoho učitelů **neumí správně využívat** a pletou si ji s pouhým projekčním plátnem. Jinak řečeno – promítají na ní své dávno připravené a většinou pouze textové přípravy jako na klasickém zpětném projektoru. Žákyně a žáci se tedy pouze dívají nebo data z promítaných materiálů opisují do sešitu. Říkáte si, že to tak určitě již dávno není, ale opak je pravdou. Jak bylo již zmíněno, důvodem jistě není vybavení dnešních škol, které je díky mnoha dotačním programům na velmi dobré úrovni. Důvodem je fakt, že již nebyl na mnoha školách udělán druhý krok, tedy **nikdo učitelům neukázal**, k čemu vlastně interaktivní zařízení slouží a **jak se mají správně využívat**.

Jsou tři možnosti – tzv. žákovské aktivity (jak se výukovým materiálům pro interaktivní zařízení říká) si **vyrobiť sami, dostat je** od kolegyně či kolegy ze sborovny, nebo si je **stáhnout z internetu**.



Internetové portály s DUMy

Vezmeme to od konce – na internetu se sice vyskytuje mnoho materiálů vytvořených pro interaktivní tabule, bohužel kvalita těchto materiálů není nijak vysoká. Ve většině případů se navíc jedná o materiály vytvořené ve starších verzích softwaru, které zdaleka nevyužívají všechny možnosti současných technologií.

Díky různým projektům začaly před cca 15 lety vznikat webové portály, na kterých uživatelé mohli sdílet své materiály. DUMy byly ale často tvořeny učiteli, kteří neovládali interaktivní techniku správně a DUMy pak spíše vypadaly jako přípravy na hodinu nebo maximálně pracovní listy. Postupem času se samozřejmě procento těchto materiálů snížilo, ale i na velkých portálech nenajdeme ani dnes příliš interaktivních DUMů. Zmiňme např. **rvp.cz** a **dumy.cz**. Přestože jde o jedny z největších českých portálů, zaměřených na publikování materiálů k výuce.

Toto hodnocení ovšem neplatí pro portál **veskole.cz**, který je moderně zpracovaný a odráží se to i na způsobu snadného vyhledávání materiálů. Podobně jako u portálu **aktivucitel.cz** pro konkurenční tabule ActivBoard je vyhledávání jednoduché, přehledné a intuitivní. Vyhledávací formulář obsahuje kolonku pro vyhledávaný text, stupeň (mateřská škola, 1. stupeň ZŠ, 2. stupeň ZŠ, střední škola, ostatní školy), předmět a jeho specifikace, typ souboru a způsob využití (podklady pro výuku nebo interaktivní cvičení). Aktuálně je na tomto portálu cca 35 000 žákovských aktivit, většina je ovšem již lehce zastaralých, protože nevyužívají aktuálně existující softwarové aplikace resp. jejich doplňky (myšlen např. doplněk SMART Lab klasického software SMART Notebook).

Zásady tvorby DUMů

Ideální cestou je tedy vyrobit si moderní žákovské aktivity sami. Ještě přidejme poznámku k terminologii – digitální učební materiál (DUM) je veškerý učební materiál v elektronické formě určený pro výuku. Většinou se jedná o prezentace, pracovní listy, textové přípravy pro výklad nové učební látky, tabulky, audio a video ukázky apod. Mezi DUMy samozřejmě patří i žákovské interaktivní aplikace resp. žákovské aktivity. Nejedná se tedy o synonymum slova DUM, ale o jeho podmnožinu.

Jak již bylo zmíněno, DUM si může vytvořit každý učitel sám, nebo jej může stáhnout ze speciálních webových portálů. Ideální DUM ovšem nenahrazuje samotnou výuku, ale vhodně ji doplňuje. Všechny DUMy by také měly být propojeny s konkrétními očekávanými výstupy, které jsou definovány v rámcových vzdělávacích programech (RVP) a konkretizovány ve školních vzdělávacích programech (ŠVP).

Jak již bylo zmíněno, každý materiál by měl být tvořen s cílem podpořit konkrétní výstupy RVP. Vždy je třeba si uvědomit, pro jakou fázi výuky je materiál určen – jiný bude pro nové téma a jiný pro ověření znalostí. Je nutné také dbát na přehledné a logické řazení stránek. Přehledné a jednoznačné pak mají být i úkoly na jednotlivých stránkách aplikace. Je vhodné omezit počet úkolů, ideálně na 1 na každé stránce s aktivitami, vhodné je také omezit počet výukových stran s aktivitami na cca 5–10.

Jelikož chceme, aby byl DUM pro děti přínosný, bavil je, aktivizoval je a přinesl jim nové poznatky, musíme dodržovat určité zásady. Kritérií, která má dum splňovat, lze nalézt mnoho, můžeme např. zvolit kritéria, která uvádí Jan Němec*.

* NĚMEC, Jan. Metodika tvorby interaktivních DUM [online]. Brno, 2011 [cit. 2021-01-25]. Dostupné z: <https://is.mendelu.cz/zp/portal_zp.pl?prehled=vyhledavani;podrobnosti_zp=38089;zp=38089;download_prace=1>

Obsah titulní strany

Titulní strana by měla obsahovat:

- ✓ Jméno autora,
- ✓ škola,
- ✓ téma hodiny,
- ✓ předmět,
- ✓ ročník,
- ✓ klíčová slova.



Na další straně výukové aplikace je vhodné připojit metodický návod, jak s daným materiálem pracovat.

Grafická úprava

Neméně důležitou roli při tvorbě DUMů hraje grafická úprava. Každý DUM by měl splňovat následující pravidla:

- ✓ Použití bezpatkového písma (Arial, Calibri),
- ✓ vhodná velikost písma (28 a vyšší),
- ✓ jednotný motiv a barevnost - vhodná barevná kombinace s písmem,
- ✓ jednotný font,
- ✓ ne moc grafických prvků, aby se nestaly rušivými.

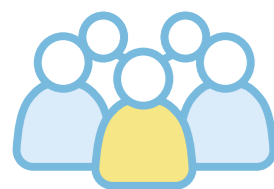
Je důležité volit úkoly tak, aby žák byl zapojen, nebo aby alespoň dával pozor při výkladu nezáživných témat. Použít multimedia a animace pro zpestření samozřejmě lze, vždy je ale třeba vybírat, aby nebyl materiál zahlcen. Je třeba si uvědomit, že DUM je pouze doplňkem výuky a i nadále je nutné zařazovat do výuky společnou diskuzi a práci ve dvojicích či skupinách. Na škodu samozřejmě není překvapit děti či studenty něčím vtipným a neočekávaným, záleží na fantazii každého učitele.

Příklady z praxe

Sada ukávek konkrétních žákovských aktivit pro různé předměty na:

<https://www.petrpexa.cz/smart/ukazky.notebook>

https://www.petrpexa.cz/smart/ukazky_lab.notebook



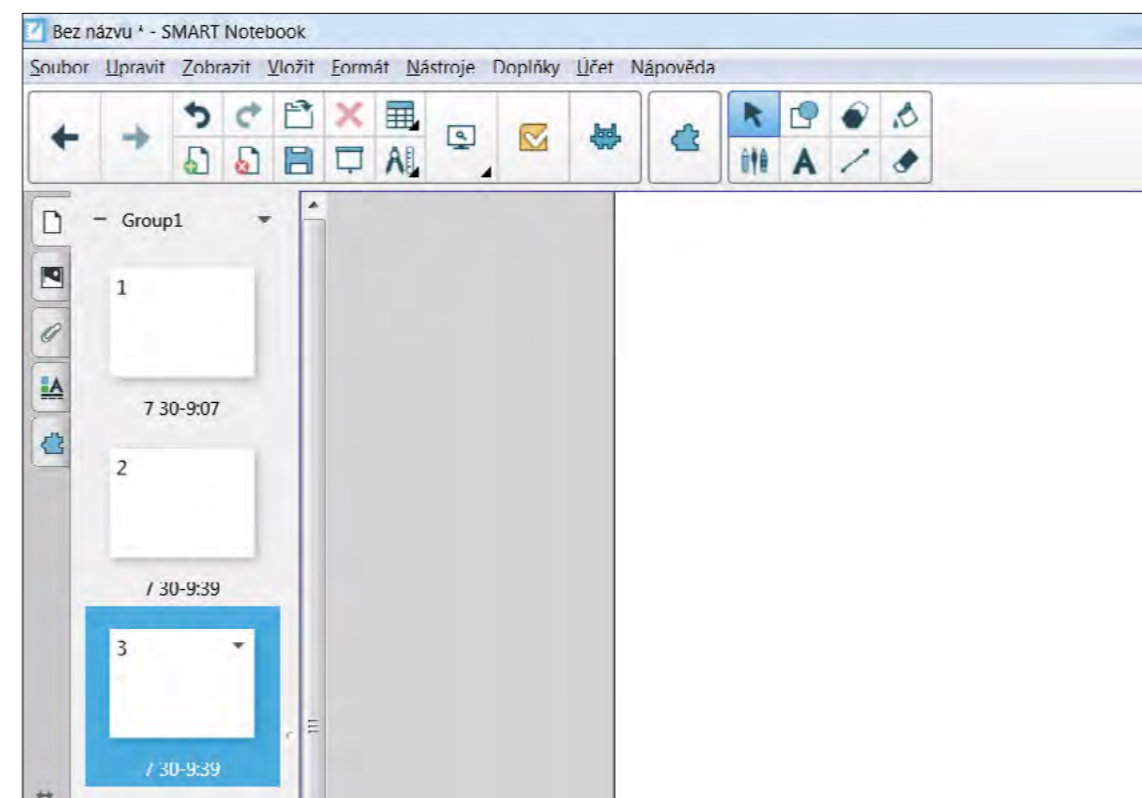
Metodická a didaktická část

SMART Notebook

Pro tvorbu interaktivních materiálů je zapotřebí určitý software. Nejrozšířenějším programem, určeným k tvorbě interaktivních materiálů resp. žákovských aktivit, je SMART Notebook. SMART Notebook je software, který využívají tisíce českých učitelů při každodenní výuce již téměř 20 let, je dodáván společně s tabulemi či displeji firmy SMART Technologies a je zdarma k dispozici pro všechny vyučující na škole, která je těmito interaktivními zařízeními vybavena. Software je možné také zdarma stáhnout v 45-denní trial verzi ze stránek <https://www.smarttech.com/products/education-software/>, nejnovější je verze 21. 0. Díky tomu si učitelé mohou připravovat hodiny doma, v kabinetě nebo kdekoli, kde mají k dispozici svůj počítač.

Po prvním spuštění programu na první hodině výuky či různých odborných kurzů a workshopů se vždy účastníci ptám, jaký jiný program jim připomíná a vždy padají stejné a správné odpovědi – PowerPoint a Malování. Z tohoto zjištění je zřejmé, že tvorba žákovských aktivit je naprosto jednoduchá, velmi se podobá tvorbě klasických prezentací a jednotlivé objekty aktivit se kreslí známými grafickými nástroji.

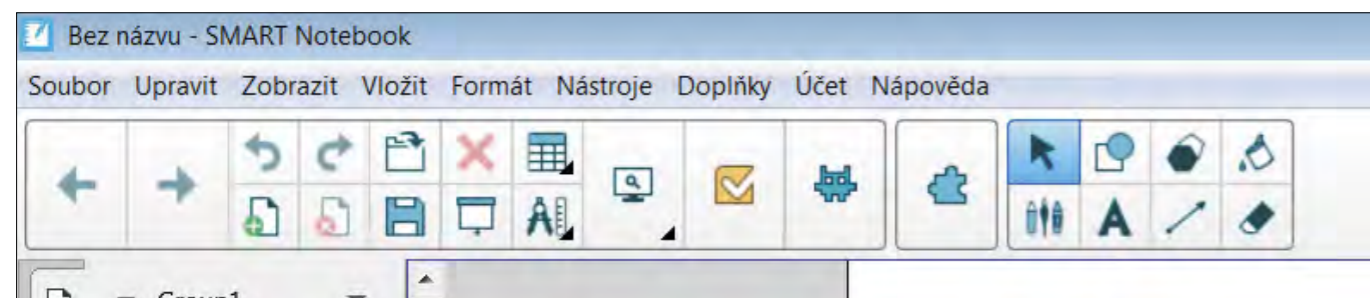
Jsou dva způsoby tvorby aktivit ve SMART Notebooku – jednak „ruční“ varianta, tedy od prvního návrhu aktivity přes tvorbu jejího obsahu až po výsledný grafický vzhled si vše vytvoříme sami. Opět se nabízí porovnání s PowerPointem, protože druhý způsob je obdobný – můžeme využít předpřipravené šablony aktivit SMART Lab, do kterých vyplníme jen promyšlený obsah aktivity, ale samotným návrhem, tedy jak a čím budeme děti aktivizovat a grafickým vzhledem aktivity se již zdržovat nemusíme. Této variantě se budeme věnovat později.



Obrázek 1: SMART Notebook

Panel nástrojů

Program SMART Notebook obsahuje nástroje pro psaní, kreslení, malování, rýsování, nástroje pro kreslení tvarů, nástroje pro matematiku včetně funkcí, grafů a geometrie. Důležitou součástí tohoto programu je tedy hlavní **panel nástrojů** a tlačítka v něm jsou organizována do dalších panelů.



Obrázek 2: Panel akcí a nástrojů

Na obrázku č. 2 můžeme vidět základní tlačítka pro obsluhu programu, zleva: tlačítka **Zpět** a **Vpřed**, pomocí kterých posouváme jednotlivé vytvořené stránky, další dvě šipky posouvají zpět a vpřed úpravy, které provádíme. Tato tlačítka uživatelé již dobře znají z dalších známých programů (Microsoft Word, PowerPoint).

Ikona bílé stránky se zeleným kroužkem a znaménkem plus slouží k přidávání dalších stran, naopak tlačítko s **červeným kroužkem a křížkem** odstraňuje stránky, které chceme vymazat.

Pro nastavení barvy či vzoru pozadí stránky stačí kliknout **pravým tlačítkem myši na prázdnou plochu stránky a zvolit možnost Nastavit pozadí**. Na postranním panelu se objeví možnosti bez výplně, plná výplň, přechodová výplň, výplň vzorkem či výplň obrázku, takže je možné kombinovat několik barev, vzorů a podobně. Důležité však je volit takové barvy, aby nezhoršovaly čitelnost textu a neničily zrak. Pokud zvolíme tmavé pozadí, je možné na stránku umístit další objekt (světlý), který vytvoří plochu, kam je možné psát.

Ikona složky využijeme tehdy, chceme-li otevřít a načíst nový soubor, tlačítko diskety, když ukládáme provedené změny. **Další tlačítka** slouží k odstranění objektu, zvětšování či zmenšování zobrazení obrazovky, vkládání tabulek a také v panelu vidíme výrazné ikony doplňků SMART Lab a SMART Response, kterým se budeme věnovat později detailně.

Vedle panelu akcí najdeme **panel kreslicích nástrojů** jako ve známém Malování, kde si můžeme vybrat nástroj podle toho, co chceme vytvořit. Po kliknutí na některý z nástrojů se ještě vedle zobrazí kontextový panel, který nabízí např. různé barvy, styly, tloušťky čar, tvary a další vlastnosti nakreslených objektů.

První ikonou je **kurzor**, který si vybíráme, když potřebujeme s objektem pohybovat, přesunovat ho, upravovat velikost apod. Dále vidíme **tlačítko s tvarem čtverce a kruhu**, které slouží k vkládání různých tvarů, jako jsou čtverec, obdélník, kruh, hvězda a další. Další ikona slouží pro **vkládání mnohoúhelníků a plechovku s barvou** využijeme, když chceme zvolit barvu výplně objektu. Ve spodní řadě nejdeme **tlačítko pro výběr pera a vkládání textu**, **ikona čáry** umožňuje volbu čar obyčejných, přerušovaných, s šipkami a dokonce čar obloukových.

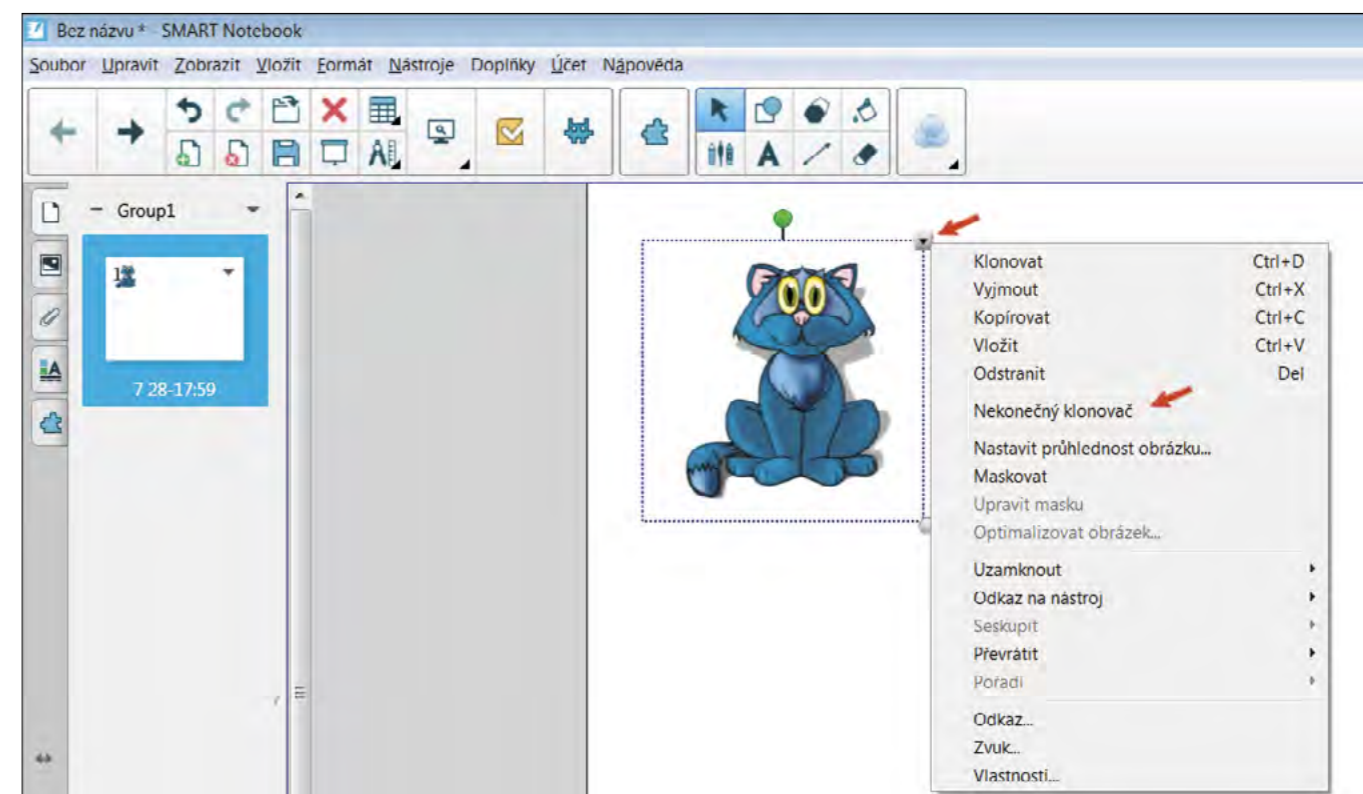
Program tedy nabízí snadnou intuitivní orientaci díky grafickým ikonám, které jsou připraveny pro většinu úprav a disponuje i řadou dalších kreativních nástrojů jako je pero textu, štětec, nekonečný klonovač (viz dále) a rozpoznávání tvarů.

Vkládání a úprava objektů v aktivitě

Základem každé aktivity jsou textové a grafické objekty, které žáci dotykem různě přesouvají, spojují čarami, zařazují na správné místo či v případě textu sami perem či prstem vkládají do chybějících míst. Do stránky s aktivitou vkládáme potřebné objekty kreslením pomocí horního panelu nástrojů, kopírováním z jiných programů (např. text z pracovních listů, vytvořených ve Wordu) a nabídkou Vložit v menu programu SMART Notebook, vše opět jako v Powerpointu.

Po vložení objektu ho můžeme různě **upravovat kliknutím na šipku v horním panelu** a poté na zvolený objekt. Základními úpravami jsou především **změna pozice a velikosti** jako ostatně v mnoha dalších známých programech. Pomocí **zeleného kolečka nad obrázkem** je možné objekt otáčet a pokud klikneme na **šipku v horní části obrázku** (nebo na objekt klepneme pravým tlačítkem myši), jsou nabídnuty další různé možnosti úprav. Pokud zvolíme položku Převrátit a poté Nahoru/dolů, můžeme otočit obrázek vzhůru nohama, stejná možnost se nabízí u převrácení Vlevo/vpravo, kdy bude obrázek zrcadlově otočen.

Další funkcí, která může sloužit k animovanému skrytí objektu, nebo naopak jeho dynamickému zobrazení, je volba **Vlastnosti** ve spodní části nabídky (jak bude ještě uvedeno později podrobně, stejnou funkci zajistí i třetí ikona v levém postranním panelu).



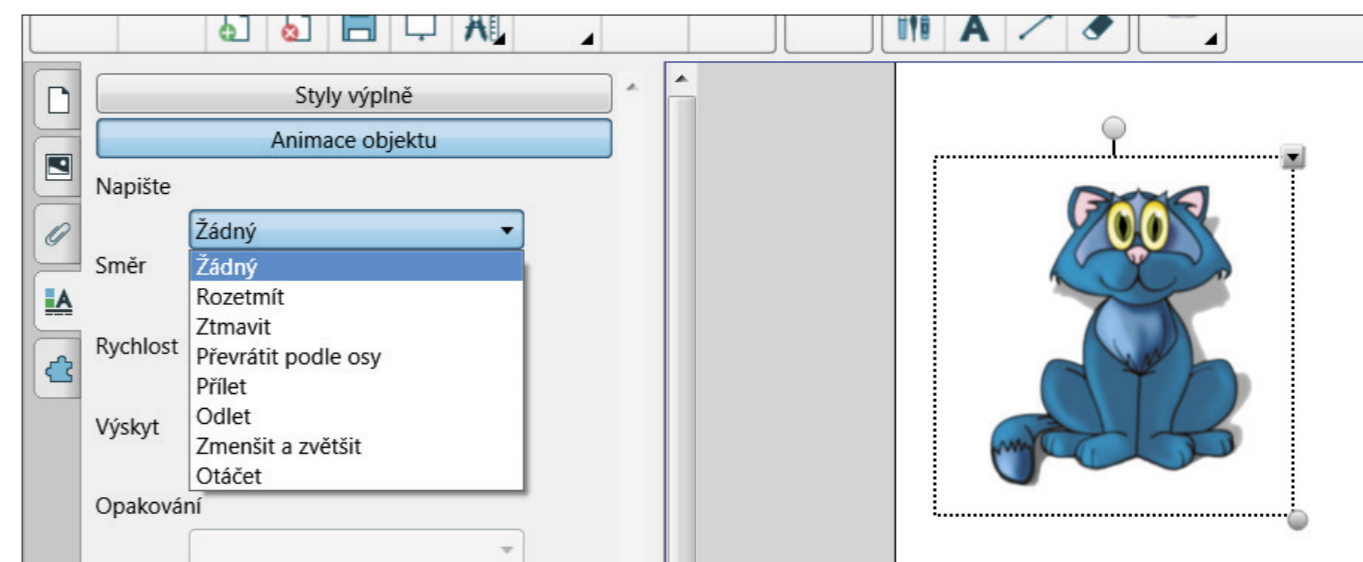
Obrázek 3: Práce s objekty

Pokud chceme jeden objekt využít vícekrát, můžeme nastavit **Nekonečný klonovač**, což je snad jediná funkce, kterou jiné programy nemají. Poté již stačí objekt uchopit, přesunout ho na zvolené místo a akci je možné provést kolikrát bude třeba. Tuto výbornou funkci využijeme např. při zařazování chybějících slov a písmen v textu v elektronických pracovních listech apod.

Další funkcí, aplikovatelnou na objekty, je možnost jejich **zamčení**. Při tvorbě vlastních materiálů si tak můžeme uzamknout pozici některých objektů, se kterými nemá být při vlastní výuce hýbáno. Opět vybereme šipku u obrázku, možnost Zamykání a poté Uzamknout pozici, pokud pozici chceme změnit, opět ho odemkne.

Pokud chceme dva objekty **spojit** v jeden (např. pro lepší manipulaci), označíme je oba a vybereme možnost Seskupení a dále Seskupit. Objekty pak můžeme posunovat společně jako jeden. Pokud jeden objekt překrývá druhý, můžeme zadní přesunout dopředu nebo naopak funkcí Pořadí, stejně tak je možné objekty přesunovat o úroveň dopředu či dozadu v případě, že jsou v několika vrstvách. Tyto funkce jsou opět jistě již velmi dobře známy z grafických editorů typu Malování.

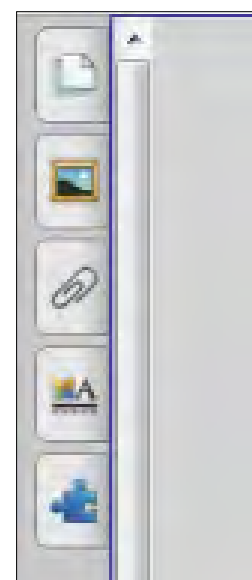
Ke každému objektu je také možné **přidat odkaz nebo zvuk**. Díky připojení odkazu je možné se následným kliknutím na objekt nebo na malou ikonu glóbu dostat na místo, na které je odkazováno (stránky na internetu, stránka v téže aplikaci, soubor v počítači či nějakou přílohu). V dolní části je možné zvolit, zda se dostat na odkazovanou stránku kliknutím jen na ikonu či na celý objekt a pokud zvolíme připojení zvuku (zvuková ukázka hudebního nástroje či zvířete, celá písnička apod.), nabízí se možnosti vložení zvuku z počítače či nahrání vlastního. Opět je možné zvolit, zda bude zvuk spuštěn pouze kliknutím na ikonu či na celý objekt.



Obrázek 4: Animace objektu

Součástí programu SMART Notebook je také rozsáhlá galerie obrázků, u kterých se nemusíte bát autorských práv a dalších grafických užitečných prvků, jako jsou různá pozadí, mapy, grafy, vlajky, portréty osobností apod.

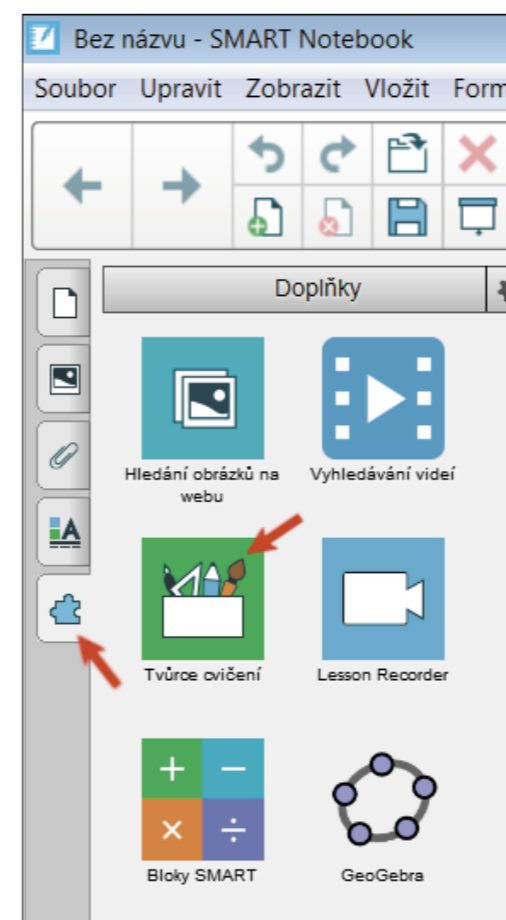
Vlastnosti objektů a jejich animace



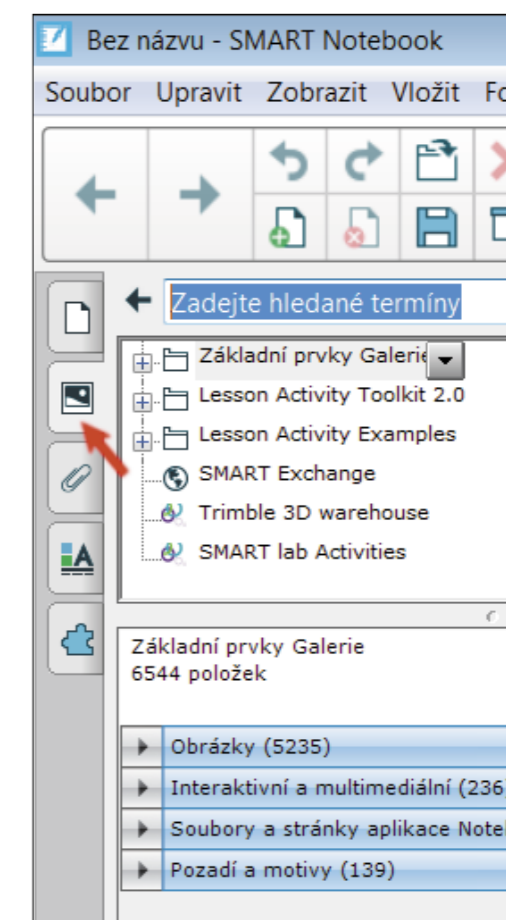
V softwaru SMART Notebook najdeme také **postranní panel záložek**, kde první ikona slouží k prohlížení stránek, druhá k vkládání obrázků z galerie a záložka s motivem kancelářské sponky slouží k přikládání příloh.

Další kartu můžeme využít pro **úpravu objektů, jejich barev, obrysů či animací**, po jejím zvolení se v postranním panelu objeví možnosti Styly výplně a Animace objektu (jak již bylo zmíněno výše, totéž lze provést i v místní nabídce objektu a položkou Vlastnosti). V možnosti Styly výplně je možné např. nastavit i průhlednost obrázku, kterou využijeme, chceme-li dát obrázek do pozadí stránky.

V záložce **Animace objektu** (viz obr. 4) je možné nastavit jeho plynulé zmizení nebo zobrazení, přílet a odlet, postupné otočení a převrácení, animované zvětšení či zmenšení a s tím související další možnosti animace, např. směr, rychlost, počet opakování animace apod. Tyto efekty využijeme např. ke skrytí objektu po kliknutí na objekt při správné odpovědi, nebo pouze jako estetický doplněk.



Obrázek 5: Tvůrce cvičení



Obrázek 6: Galerie obrázků

Tvůrce cvičení

Zajímavou pomůckou pro tvorbu vlastních žákovských aktivit je tzv. tvůrce cvičení. Získáme ho kliknutím na **poslední ikonu levého postranního panelu** a s jeho pomocí lze vytvářet jednoduchá cvičení, kdy určité prvky stránky mohou přijímat nebo odmítat další objekty na stránce zobrazené. Jinak řečeno – dílčí objekty je možné do nějakého jiného objektu přesunovat (např. správné odpovědi) a ty jsou poté přijaty (v cílovém objektu zůstanou) nebo odmítnuty a vráceny pak zpět na původní místo.

Lze takto vytvořit doplňování výsledků k příkladům, zařazovat prvky do celků se společnými vlastnostmi, vše s automatickou kontrolou správnosti přiřazení. Jako objekty cvičení lze využít různé nakreslené útvary, připravené obrázky nebo text.

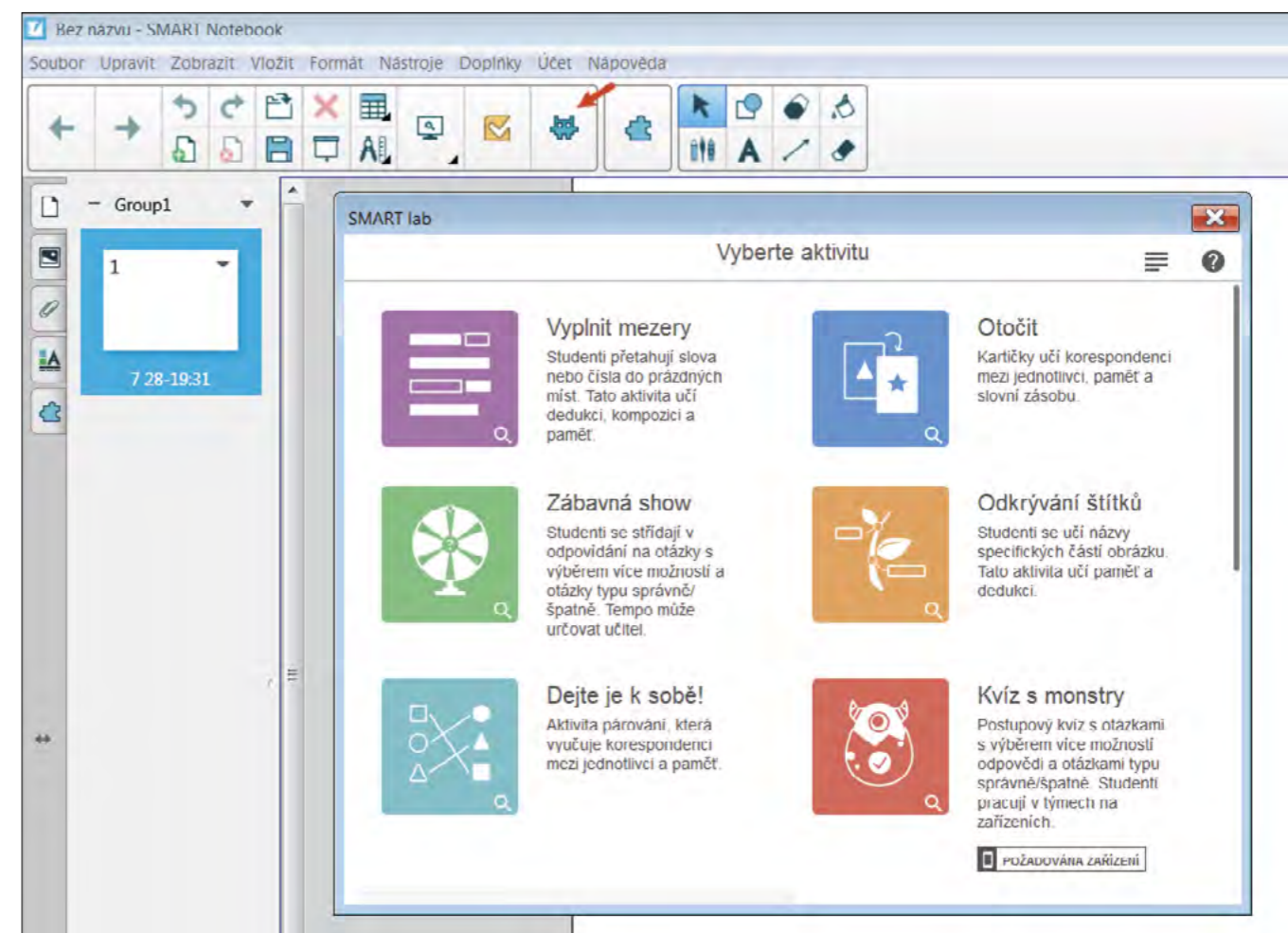
*Tip na závěr kapitoly: Veškeré zmíněné funkce a operace si lze prohlédnout v sadě mnoha připravených žákovských aktivit na <https://www.petrpexa.cz/smart/ukazky.notebook> a především si je prakticky vyzkoušet s využitím prvních dvou videí *Interaktivní výuka 1* a *Interaktivní výuka 2* na <https://www.petrpexa.cz/webinare.php>.*

Tvorba aktivit s využitím doplňku SMART Lab

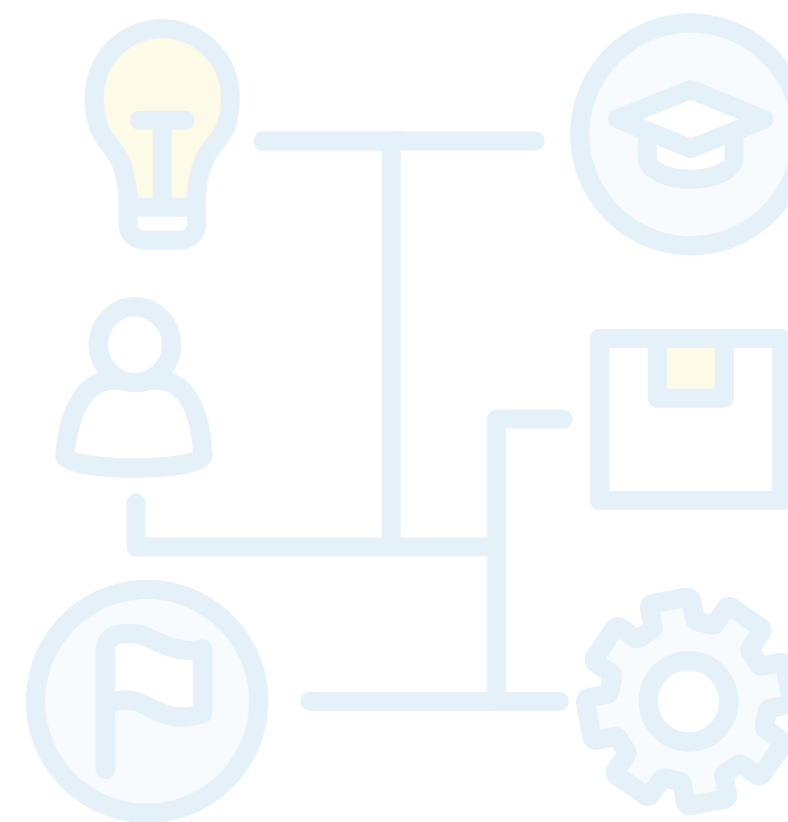
Jak již bylo zmíněno, druhou možností tvorby žákovských aktivit je využít doplněk SMART Lab, který je ve SMART Notebooku od verze 17. Nabízí celkem deset vyučovací šablon aktivit plus možnost testování či hlasování, z nichž většinu je možné rozeslat i do žákovských mobilních zařízení a použít je např. při online výuce, tedy „na dálku“.

Aktivitu jsem pro přehlednost rozdělil do **tří skupin** podle toho, jak s nimi lze ve třídě při vyučování pracovat. **Do první skupiny** patří aktivity, které jsou určeny pouze pro práci na interaktivní tabuli, **druhou skupinu** tvoří aktivity, které je možné použít na interaktivní tabuli a žáci ve třídě se k ní mohou zároveň přihlásit ze žákovských zařízení (tablety, mobilní telefony) a **třetí skupina** jsou aktivity vyžadující žákovská zařízení, bez nichž aktivity použít nelze. Přestože je u každé šablony připravený intuitivní průvodce, pomocí kterého není problém aktivitu vytvořit a použít, každá bude na následujících řádcích nejdříve popsána z pohledu žáka a potom se budeme věnovat postupu tvorby dané aktivity.

Pokud chceme vytvářet některou z aktivit ze souboru výukových šablon SMART Lab, použijeme **ikonu na panelu nástrojů** (viz obr. 7) a otevře se nové okno, které nám nabízí výběr jednotlivých šablon aktivit, uložených v cloudu SMART (na webu výrobce, nelze tedy tento doplněk provozovat bez připojení k internetu). Každá šablona má svoji jednobarevnou ikonu, název a stručný popis.



Obrázek 7: Šablony aktivit SMART Lab



Aktivity pro práci pouze na interaktivní tabuli

Dvě aktivity jsou určeny pouze pro práci na interaktivní tabuli ve třídě. Jsou to **Zábavná show** a **Zrychlení** a obě mají formu kvízové soutěže. Ve vyučování je můžeme dobře použít na začátku vyučovací hodiny pro zopakování učiva a motivaci, nebo na konci hodiny při shrnutí učiva.

Zábavná show – popis aktivity

Aktivita Zábavná show je určena pro **dva žáky** či **dvě družstva žáků**, kteří hrají proti sobě na interaktivní tabuli. Každému žákovi či družstvu je přiřazena animovaná postavička, která jej reprezentuje a žáci pak vybírají otázku roztočením „kola štěstí“. Animovaný moderátor pokládá otázku a navrhuje možné odpovědi, ze kterých si žáci vybírají a za správnou odpověď získají body. Pokud odpoví špatně, pokračuje ve hře druhé družstvo. Odpovídá na stejnou otázku, která je však již hodnocena menším počtem bodů, to se opakuje až do vybrání správné odpovědi. V průběhu hry mohou žáci vytočit také **prémii**, díky které získají za správnou odpověď na položenou otázku dvojnásobný počet bodů, nebo bonus ve formě možnosti odstranění některé špatné odpovědi. Na závěr je soutěž vyhodnocena moderátorem a nabídnuta kontrola, kde si můžeme prohlédnout všechny otázky a správné odpovědi, čímž znovu zopakujeme a upevníme učivo, které jsme do kvízu zvolili.

Zábavná show – postup tvorby aktivity

Po vybrání aktivity se otevře nové okno, ve kterém vybíráme obsah, tedy zadáváme otázky soutěžního kvízu. Otázky mohou být dvojího typu. Otázky s více možnými odpověďmi a otázky s odpověďmi – pravda, nepravda. Potom zadáváme dvě, tři nebo čtyři možné odpovědi, z nichž vždy jednu označíme jako správnou. Ve stejném okně pokračujeme obdobně přidáváním otázek. Po zadání potřebného počtu otázek můžeme v dalším okně zkontrolovat obsah. Vidíme zde seznam všech námi definovaných otázek a správných odpovědí. Po dokončení se nám okamžitě spustí naše Zábavná show.

Zrychlení – popis aktivity

Aktivita Zrychlení je určena **pro jednoho až čtyři žáky**, kteří pracují na interaktivní tabuli. Žáci závodí na dráze, rychlost jejich autíčka závisí na správnosti odpovědí na otázky, které se během závodu objevují na interaktivní tabuli. Na začátku aktivity určíme počet hráčů, pak si každý žák zvolí svého závodníka, který jej bude zastupovat v závodě. Během závodu se v každém kole objeví dvě otázky s možnými odpověďmi a žáci odpovídají kliknutím na správnou odpověď. Pokud odpoví žák správně, jeho autíčko zrychlí. Na konci závodu je zobrazen vítěz, pak celkové pořadí s počtem správných odpovědí a čas, který žák na odpovědi potřeboval. Na závěr je možné zobrazit jednotlivé otázky se správnými odpověďmi, i to, jak který žák odpovídal, což je velmi dobré jako zpětná vazba jak pro učitele, tak pro žáka.

Zrychlení – postup tvorby aktivity

Po zvolení aktivity Zrychlení si opět vybíráme otázky pro žáky do našeho závodu. Nejdříve vždy typ otázky. I zde můžeme vybírat z otázek dvojího typu, mohou mít dvě nebo více možných odpovědí, otázka může mít až 150 znaků. Potom zadáváme dvě, tři nebo čtyři možné odpovědi a z nichž vždy jednu označíme jako správnou. Ve stejném okně pokračujeme obdobně přidáváním otázek. Po zadání potřebného počtu otázek můžeme zkontrolovat obsah v dalším okně, kde vidíme seznam všech námi definovaných otázek a správných odpovědí. V nejnovější verzi programu si v tomto okně můžeme ještě vybrat, zda chceme žákům na zodpovězení otázek určit časový limit a pokud ano, tak zde nastavíme počet sekund.

Aktivity pro práci na interaktivní tabuli i žákovských zařízeních

Druhý typ aktivit tvoří aktivity určené primárně **pro práci na interaktivní tabuli**, ty si ale také může každý žák jednoduše vyhledat na internetu a pracovat na svém zařízení. Takovou „mobilní“ aktivitu poznáme tak, že po jejím spuštění se na pravém okraji obrazovky objeví ikona mobilního telefonu a hvězdy. Po kliknutí na ikonu mobilního telefonu se otevřou dvě nová okna – v prvním žáci uvidí adresu webové stránky, kde aktivitu najdou (hellosmart.com) a ID virtuální třídy, do které se mají přihlásit (vytvoření virtuální třídy vyučujícím si ukážeme později). Vyučující na svém zařízení sleduje počet připojených žáků a je vyzván ke spuštění aktivity, čímž ji zpřístupní žákům a sleduje průběh aktivity u jednotlivých žáků. Ikona hvězdy nám umožní vybrat herní prvky – soutěž, náhodu a čas.

Vyplnit mezery – popis aktivity

Aktivita Vyplnit mezery je určena pro **doplňování chybějících slov či písmen** (např. i/y) do textu. Žákovi se na tabuli zobrazí text s vynechanými místy a jeho úkolem je přetahovat slova nebo čísla umístěná pod textem do prázdných míst i s automatickou kontrolou správnosti přiřazení.

Vyplnit mezery – postup tvorby aktivity

Po vybrání aktivity se otevře nové okno pro přidání obsahu. Do prázdného prostoru napíšeme nebo vložíme text, se kterým mají žáci pracovat. Text může obsahovat až 300 znaků. Pak definujeme prázdná místa klikáním na jednotlivá slova, která budou žáci doplňovat. Pokud chceme vybrat jen část slova, nebo naopak chceme nechat žáky doplňovat více slov, můžeme použitím úchytu pro změnu velikosti vytvořit větší nebo menší prázdné místo. Celkem můžeme vytvořit deset prázdných míst. Chceme-li prázdné místo odstranit, znovu na něj klikneme. Pokud máme text připravený, můžeme ještě vybrat, zda má být správné doplnění zkontrolováno při vyzvání, ihned, či nemá být kontrolováno vůbec. Pokračujeme dalším oknem, kde vybíráme vzhled aktivity resp. grafické téma. Zvolíme si, jak bude aktivita na monitoru vypadat, v této aktivitě máme na výběr čtyři různé grafické šablony. V pravém horním rohu jsou ještě umístěny ikony pro volbu vypnutí zvuku, opakování aktivity a editaci aktivity.

Otočit – popis aktivity

V této aktivitě **žák klikáním otáčí kartičky**, na kterých mohou být obrázky nebo texty. Je to aktivita, která může být použita při vyvozování, upevňování i opakování učiva. Lze trénovat paměť i slovní zásobu a použít kartičky pro různé hry a soutěže jako při klasickém pexesu.

Otočit – postup tvorby aktivity

V novém okně se objeví dva sloupce, do kterých můžeme psát text či vkládat obrázky. V prvním sloupci je text nebo obrázek, který bude na kartičce lícem nahoru. Ve druhém sloupci je obsah, který bude lícem dolů. Po vytvoření obsahu všech kartiček pokračujeme dalším oknem, ve kterém si vybereme téma grafické úpravy kartiček. Zde máme na výběr šest motivů, po dokončení se objeví plocha s kartičkami a ikonami pro možnost připojení mobilních žákovských zařízení.

Odkrývání štítků – popis aktivity

Tato aktivita nám **umožňuje definovat popis jednotlivých částí různých obrázků, schémat či nákresů**. Žáci se učí názvy specifických částí obrázku. Aktivita trénuje paměť a učí dedukci. Na obrázku jsou vytvořeny popisky s otazníky, po kliknutí na otazník se objeví správný popis zvolené části obrázku. Je možné i popisky skrýt a ponechat pouze obrázek bez popisku a následně pak popisky odkrýt.

Odkrývání štítků – postup tvorby aktivity

Při tvorbě aktivity nejdříve vybíráme obrázek, jehož části budeme popisovat. Může to být obrázek z galerie programu Smart Notebook, našeho počítače či z internetu. Pak klikáním vybíráme části obrázku, které budou popisovány. Vybereme místo, klikneme a do otevřeného okénka napíšeme popisující text. Můžeme tak vyrobit až deset popisků u jednoho obrázku, lze měnit i styl popisku. Po dokončení se ukáže obrázek, ve kterém jsou popisky nahrazeny otazníky.

Dejte je k sobě – popis aktivity

V této aktivitě vytváří žáci **dvojice odpovídajících si slov, čísel, či obrázků**. Na ploše jsou dvě skupiny objektů a žák vybere objekt z první skupiny a úkolem je správně přiřadit objekt z druhé skupiny. Pokud vytvoří žák správnou dvojici, objekty se seskupí a zařadí. Když je dvojice vybrána špatně, vrátí se objekty na původní místo, u některých proběhne i pěkná animace.

Dejte je k sobě – postup tvorby aktivity

Na začátku je možné zvolit názvy dvou kategorií, ze kterých bude žák vybírat objekty. Pak zapisujeme texty, nebo volíme obrázky do jednotlivých skupin. Vždy dva objekty, které patří k sobě, budou na jednom řádku, celkem můžeme přidat deset dvojic objektů. Na stejné stránce zvolíme, zda se bude správnost páru kontrolovat ihned, nebo až při vyzvání. V dalším okně pak vybíráme grafické téma, tedy jak bude aktivita pro žáky vypadat. Lze vybírat ze šesti možností, které jsou velmi pěkně graficky zpracované a při kontrole správnosti jsou k vidění pěkné animace.

Seřazení – popis aktivity

V aktivitě žáci **seřazují položky v předem určeném pořadí** a učí se tak srovnávat, dedukovat a uspořádat. Na začátku jsou objekty náhodně umístěné na ploše, žák umísťuje jednotlivé objekty do prázdných rámečků v určitém pořadí. Také v této aktivitě můžeme vidět pěkné jednoduché animace při správném a nesprávném řazení.

Seřazení – postup tvorby aktivity

Nejdříve opět přidáváme obsah, tedy volíme texty či obrázky, se kterými budou žáci pracovat. Zadáváme je ve správném pořadí, tedy jak je žáci mají seřadit. Můžeme zadat až deset objektů. Pak zvolíme, zda pořadí kontrolovat ihned, při vyzvání či nekontrolovat vůbec. Dále můžeme určit, co bude napsáno na prvním a posledním štítku, čili odkud kam budou žáci seřazovat. V dalším okně vybereme jedno z osmi grafických témat, takže stejnou aktivitu můžeme zopakovat v různých variantách a nebude to pro děti jednotvárné.

Super řazení – popis aktivity

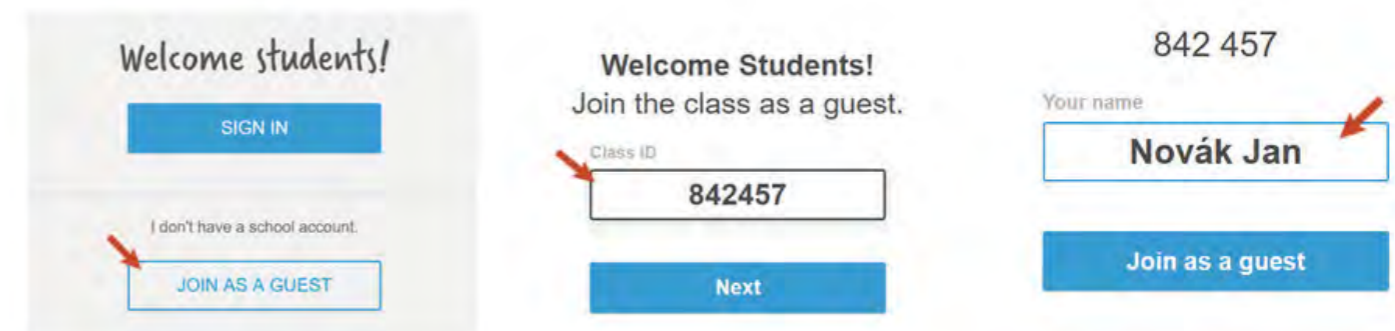
Při této aktivitě žáci třídí položky do dvou kategorií. Učí se klasifikaci a sdružování do skupin. Na začátku jsou položky náhodně umístěné na ploše. Názvy dvou kategorií pak říkají, kam má žák jednotlivé položky podle určeného kritéria přemístit. Aktivita je velmi jednoduchá jak na tvorbu, tak na vysvětlení pravidel, které je v podstatě intuitivní, ale grafika a animace jsou tak kouzelné, že ji budou chtít žáci hrát znovu a znovu.

Super řazení – postup tvorby aktivity

Na začátku přidáváme obsah. Napíšeme názvy dvou kategorií. Pak do každé kategorie zvolíme až deset položek. V dalším okně zvolíme jedno z deseti grafických témat a dokončíme aktivitu.

Aktivity pro práci na žákovských zařízeních

Do této kategorie patří aktivity **Kvíz s monstry, Response a Zapojte se!** Jsou to aktivity, které vyučující promítá na interaktivní tabuli, ale **žáci se do ní musí přihlásit přes své mobilní či stolní zařízení**. Mohou to být mobilní telefony, tablety, notebooky či klasické osobní počítače. Tyto aktivity lze úspěšně využít i při online výuce. Vyučující pak na svém osobním zařízení aktivitu spustí a sleduje postup realizace aktivity třeba z domova. Jak již bylo zmíněno výše, žák se prostřednictvím svého zařízení připojí na webovou stránku <http://hellosmart.com/> a vyplní ID virtuální třídy, které žákům vyučující předem sdělí. Při dalších spuštěních žák již pouze vybere ID z nabídky, pak zadá své jméno a může začít pracovat.



Obrázek 8: Připojení do virtuální třídy na hellosmart.com



Kvíz s monstry – popis aktivity

Po spuštění této aktivity uvidí žáci pokyn k návštěvě webové stránky **hellosmart.com** a ID třídy opět sdělí vyučující. Jak se žáci postupně přidávají, objevují se jejich jména na zařízení vyučujícího, který rozdělí žáky do týmů a spustí **kvíz**. Na displeji se objeví malé objekty představující **monstra** jednotlivých týmů. Monstra se zvětšují podle toho, zda žáci v jednotlivých týmech odpovídají správně na otázky kvízu. Po dokončení kvízu se objeví název vítězného týmu a jednoduchá animace pro vítěze. Vyučující pak může spustit kontrolu, kde uvidí žáci výsledkovou listinu a třídní revizi se správnými odpověďmi.

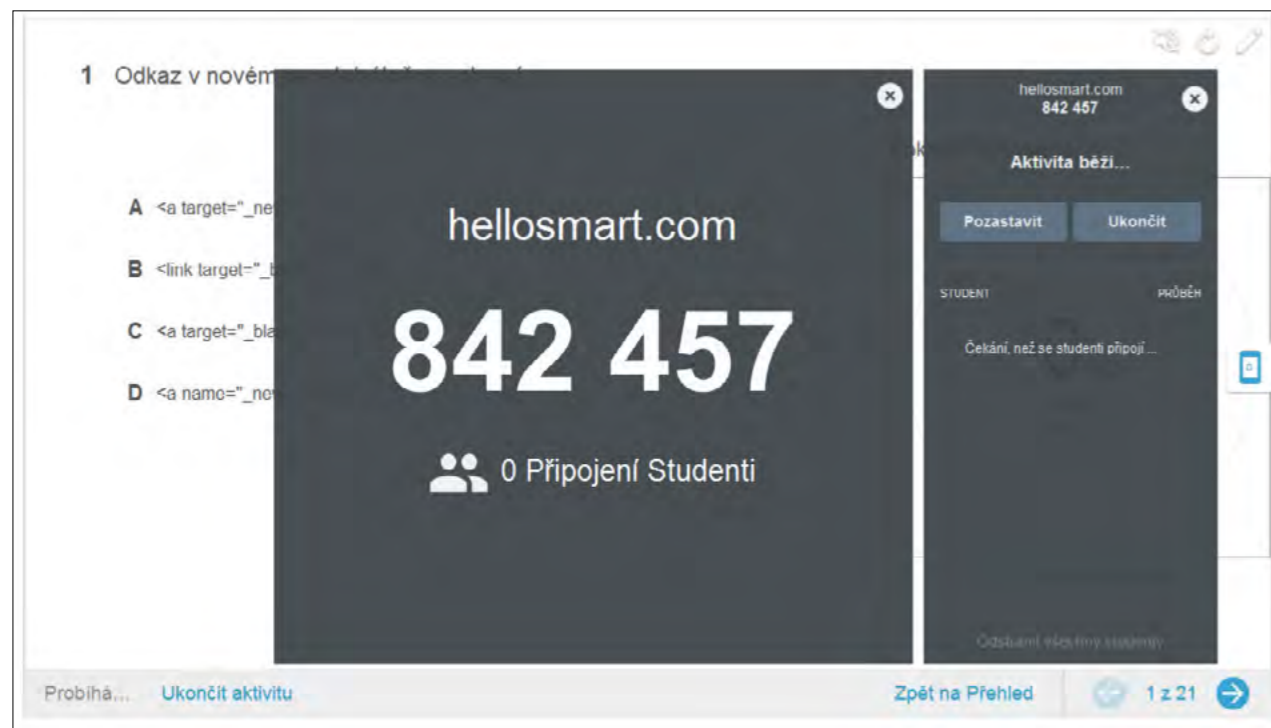
Na žákovském zařízení se při návštěvě webové stránky objeví výzva k vyčkání na rozdělení do týmů, pak uvidí žáci své monstrem a jméno. Potom už se objevují jednotlivé otázky kvízu a žáci odpovídají. Při správné odpovědi monstrem roste, při nesprávné se otázka vrátí znovu do kvízu, již bez zvolené špatné odpovědi. Po správném zodpovězení všech otázek je kvíz ukončen a žákovi se na zařízení zobrazí jeho hodnocení.

Kvíz s monstry – postup tvorby aktivity

Při tvorbě této aktivity nejdříve zvolíme, zda bude žák vybírat ze dvou či více možných odpovědí. Otázku napíšeme do rámečku. Text otázky může obsahovat až 150 znaků. Pak pod otázku zapisujeme možné odpovědi a jednu označíme jako správnou. Postupně přidáváme další otázky, až vytvoříme celý kvíz. V dalším okně si zkontrolujeme obsah, to znamená, že uvidíme všechny otázky a správné odpovědi najednou. Můžeme zde také zvolit časový limit, který bude mít žák na zodpovězení jedné otázky.

Response – popis aktivity

Tato jedna z nejlepších připravených šablon žákovské aktivity slouží k testování žáků, tvorbě kvízů, dotazníků a různých hlasovacích formulářů. Je to obdobná aplikace jako **Kahoot** nebo **MS Forms** (obě jsou dostupné i přes MS Teams), které mnozí učitelé k těmto účelům využívali v době online výuky (zmiňme ještě např. Google Forms, Quizlet a TEDEd).



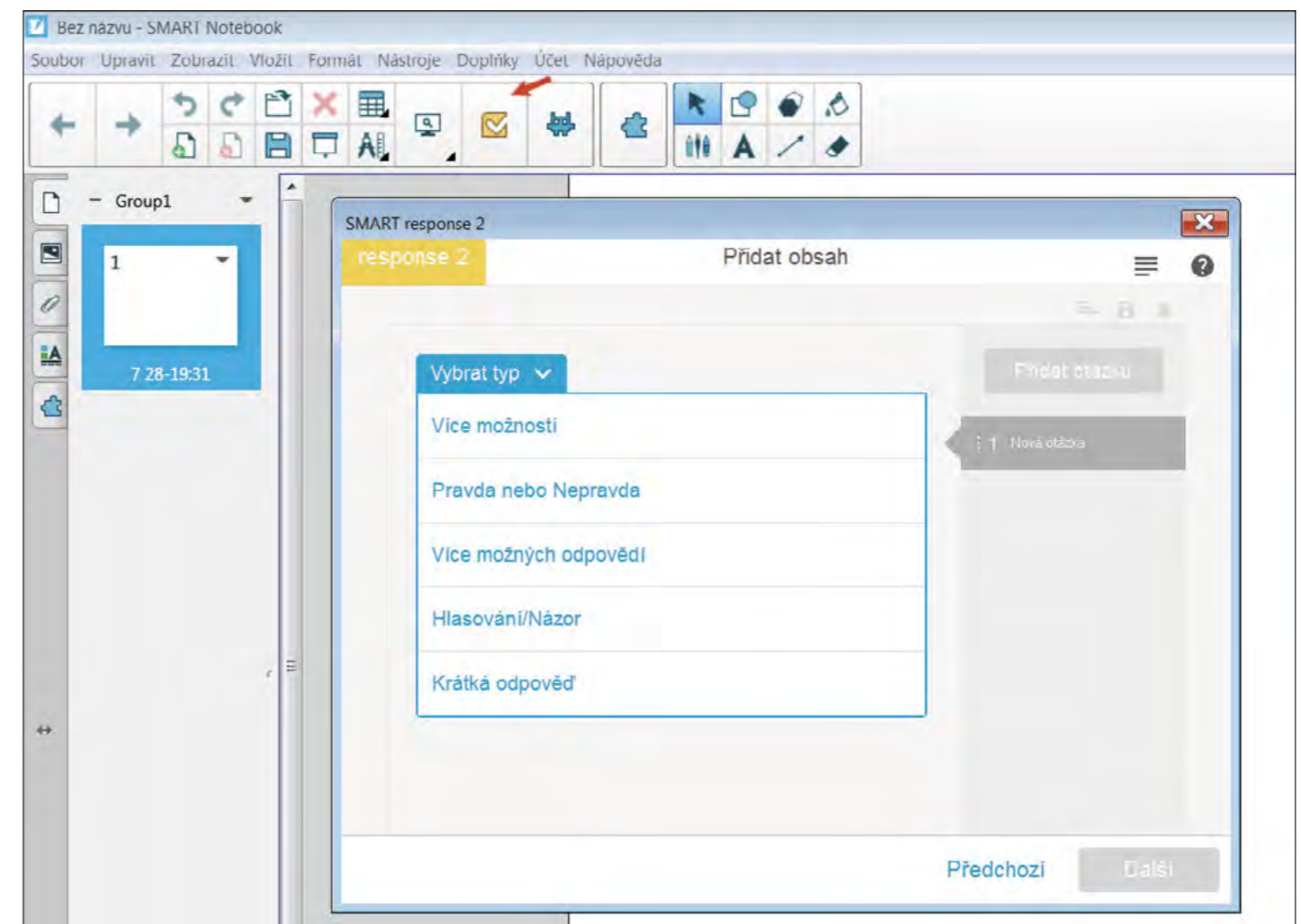
Obrázek 9: SMART Response – průběh testu

Při spuštění testu se zobrazí adresa webové stránky a ID třídy, kam se má žák přihlásit. Postupně se ukazuje počet i jména připojených žáků. Po přihlášení všech žáků vyučující test spustí. Na tabuli je možné sledovat měnící se graf, který zaznamenává procento již zodpovězených otázek. Také je možné kontrolovat, kteří žáci již odpověděli, a kteří ještě s odpověďmi otálí. Po zodpovězení všech otázek vyučující test ukončí a zobrazí se mu výsledky, které je možné i vyexportovat do přehledné tabulky ve formátu souboru MS Excel i s různými užitečnými statistickými daty (řazení, součty, průměry apod.).

Na žákovském zařízení se po přihlášení na webové stránce **hellosmart.com**, vybrání ID třídy a zadání svého jména (obr. 8) objeví první otázka hlasování. Po jejím zodpovězení žák přejde šipkou v horní části obrazovky na druhou otázku atd., až zodpoví všechny otázky. V průběhu aktivity se také může k jednotlivým otázkám šipkami vracet a opravovat je až do chvíle odeslání testu. Vyučující pak ukončí hlasování a žák si může zobrazit výsledky testu i se správným řešením.

Response – postup tvorby aktivity

Při tvorbě této aktivity opět nejdříve přidáváme obsah a začínáme volbou typu úlohy. Škála je zde široká. Můžeme vybrat otázku s nabídkou dvou či více možných odpovědí, otázku s několika správnými odpověďmi, hlasování či dokonce doplnění vlastní odpovědi. Pak do prázdného obdélníku zadáme otázku a pod něj nabídku možných odpovědí. Otázky i odpovědi mohou obsahovat text i obrázek a tam, kde je to potřeba, označíme jednu či více správných odpovědí. Postupně přidáváme další otázky až do vytvoření celého hlasování. Nakonec můžeme hlasování dát název a připsat pokyny pro žáky.



Obrázek 10: SMART Response – tvorba testu

Zapojte se – popis aktivity

Po spuštění aktivity se zobrazí jednotlivé kategorie, pokud byly zvoleny, a výzva pro vyučujícího pro spuštění aktivity. Učující může žákům otevřít okno s názvem stránky, ID třídy jim opět sdělí a sám uvidí počet a jména připojených studentů. Po spuštění okno zavře, aby bylo vidět objevující se příspěvky jednotlivých žáků. Po přidání všech příspěvků od všech žáků je možné s nimi dále pracovat, přesouvat mezi kategoriemi, seřazovat či vyhodit do koše.

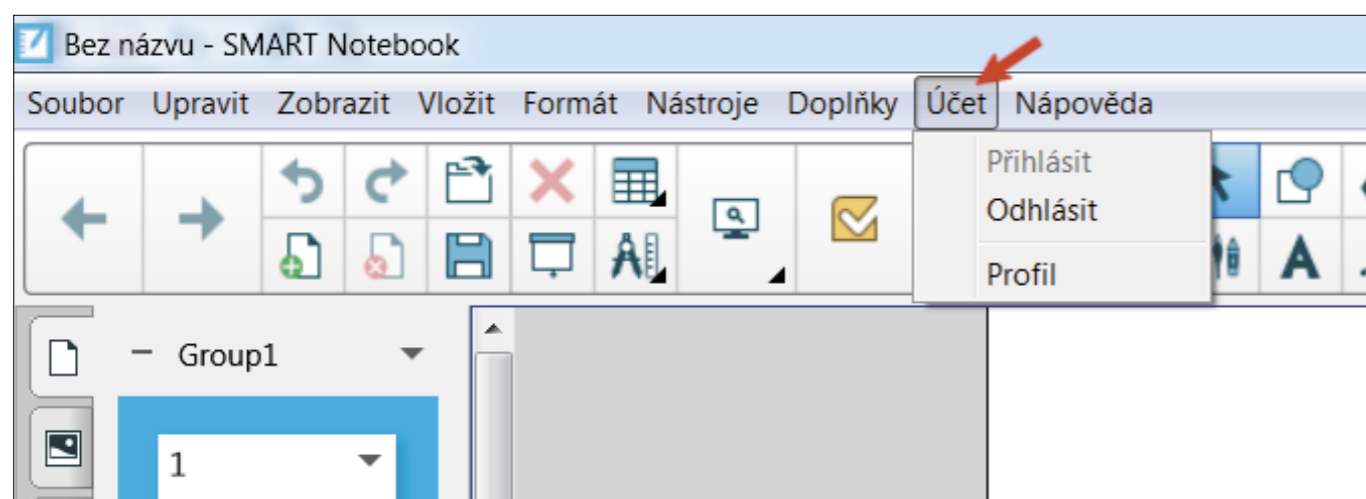
Na žákovském zařízení se po přihlášení objeví prázdný obdélník, do kterého žák napíše či vloží svůj příspěvek. Pak vybere kategorii a odešle. Obdobně přidává další příspěvky až do maximálního možného počtu.

Zapojte se – postup tvorby aktivity

Při přidávání obsahu nejdříve vybereme, jak se budou příspěvky žáků do aktivity umisťovat. Zda to bude náhodně, nebo si budou žáci vybírat z kategorií nabídnutých vyučujícím. Při výběru kategorií definujeme jednotlivé kategorie, pak zvolíme formu příspěvků žáků, zda to budou texty či obrázky, maximální počet příspěvků od jednoho žáka a zda mají být zobrazena jména studentů. A pak již můžeme žákům říci: „Zapojte se!“

Zprovoznění virtuální třídy učitelem

Pro využití aktivit, určených pro žákovská zařízení a také online formu výuky, je třeba mít jako vyučující zaregistrovaný SMART účet. Registraci lze snadno provést pomocí již existujícího Google nebo Office 365 účtu, které v současné době většina učitelů již jistě má, a proto registrace zabere pouze pár okamžiků. Registraci provedeme prvním přihlášením v nabídce Účet programu SMART Notebook a získáme tím ID třídy, které pak sdělujeme žákům před jejich zapojením do plnění aktivity. Po ukončení není nutné se odhlášovat, přihlášení je platné i po uzavření okna s programem a nemusíme si tak pro příště pamatovat heslo (platí samozřejmě pouze pro vlastní zařízení, ve třídě se z bezpečnostních důvodů určitě odhlásíme).



Obrázek 11: Registrace uživatelského účtu

Následně otevřeme ve SMART Notebook svůj pracovní soubor s aktivitami (každý má příponu.notebook), přejdeme na stránku obsahující SMART Lab nebo SMART Response aktivitu a spustíme ji. Jak již bylo zmíněno, po ukončení aktivity je možné procházet výsledky a vyexportovat je do souboru MS Excel. Výsledky i se správným řešením se zobrazí i žákům na jejich zařízení, takže reflexe resp. zpětná vazba je zajištěna dokonale.

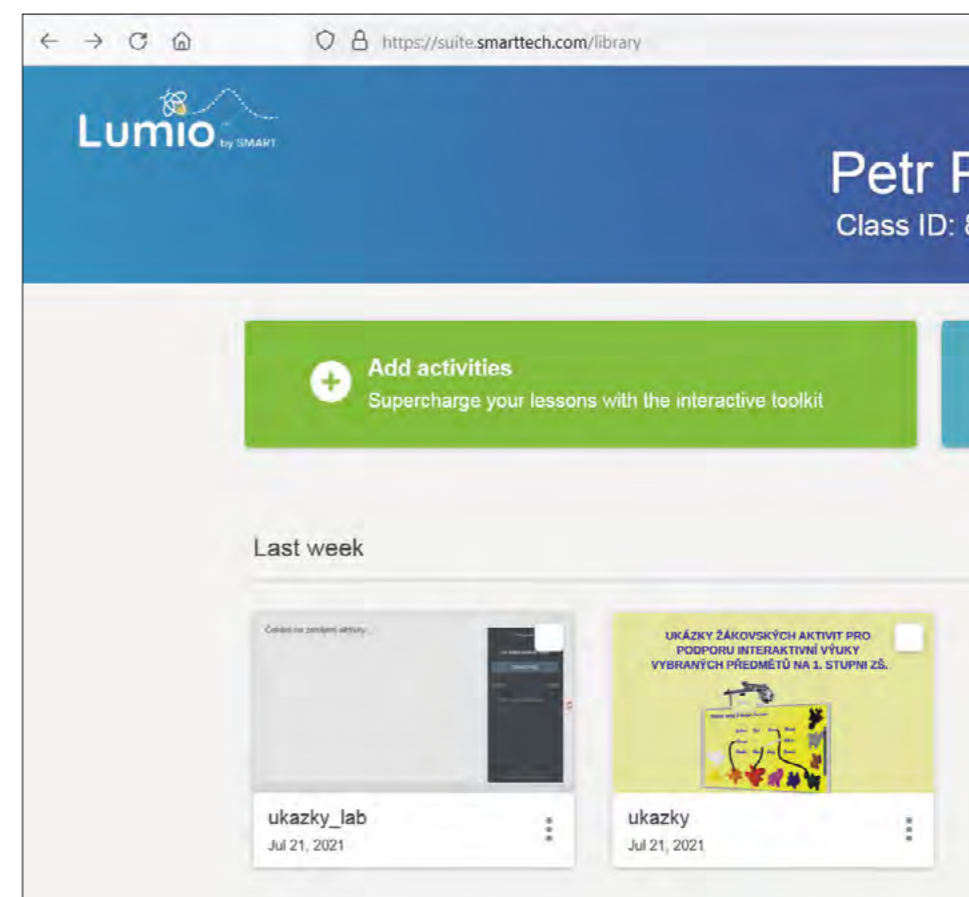
Jak se žáci připojí k aktivitě?

Pro všechny aktivity SMART Lab i SMART Response funguje online připojení stejně. **K připojení je tedy třeba vlastní zařízení žáka** (smartphone, tablet, PC) a připojené k internetu. V jakémkoliv prohlížeči webových stránek (vyzkoušeno s Google Chrome, Opera, Mozilla Firefox) pak stačí přejít na stránku **hellosmart.com**, vybrat virtuální třídu jejím ID kódem a kliknout na „Join as a guest“ (Připojit se jako host), opět viz obr. 8.

Tip na závěr kapitoly: Veškeré zmíněné SMART Lab a Response aktivity si lze prohlédnout v sadě připravených žákovských aktivit na https://www.petrpexa.cz/smart/ukazky_lab.notebook a především si je prakticky vyzkoušet s využitím třetího videa Interaktivní výuka 3 na <https://www.petrpexa.cz/webinare.php>.

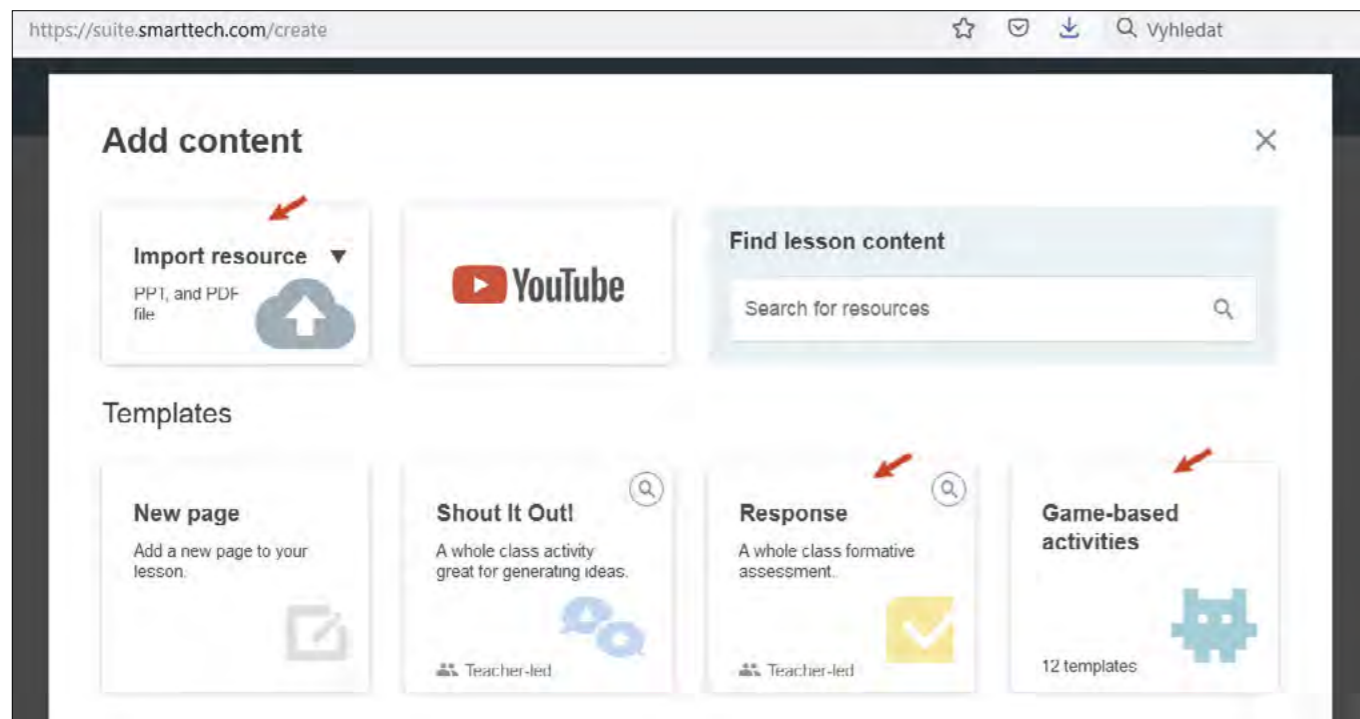
SMART Learning Suite Online/Lumio

Nechcete-li, nebo z nějakého důvodu (např. při uzavření škol) nemůžete své aktivity **provozovat přímo ve třídě přes interaktivní tabuli a lokálně nainstalovaný program SMART Notebook** na učitelském počítači, je pro tyto situace k dispozici i jeho online verze. Několik let existující pod názvem SMART Learning Suite Online. Začátkem letošního léta 2021 došlo ale k přejmenování této aplikace na krátký název Lumio a najdeme ji na <https://www.smarttech.com/lumio/>. Prohlížeč vytvořených aktivit je aktuálně i součástí MS Teams pod názvem Lumio by SMART.



Obrázek 12: Lumio SMART online

Do aplikace se přihlašujeme stejně jako do SMART Lab či Response, tedy přes účet Google nebo Office 365 a pak můžeme přímo v internetovém prostředí vytvářet nové materiály (resp. SMART Lab a Response aktivity), jak je popsáno v přechozím textu, nebo nahrávat již materiály vytvořené v klasické desktopové verzi SMART Notebooku (zelený box Add activities, viz obr. 12). Určitou nevýhodou může být fakt, že tato online verze aplikace Lumio zatím není v češtině, ale snad to takový problém není, vše je zcela intuitivní. Na obr. 13 pak vidíte základní tři funkce aplikace Lumio – import již existujících materiálů a vpravo dole známé ikony doplňků SMART Lab a Response, které zde můžeme nejen používat s importovanými aktivitami, ale i přímo aktivity vytvářet bez nainstalovaného Smart Notebooku s doplňky Lab či Response. Pokud momentálně (nebo vůbec) nemáme desktopovou verzi Smart Notebooku k dispozici.



Obrázek 13: Lumio SMART - šablony aktivit online

Užitečné webové odkazy k tématu interaktivní výuky

Portály s elektronickými výukovými materiály:

- ✓ <https://www.rvp.cz>
- ✓ <https://www.dumy.cz>
- ✓ <https://www.veskole.cz>

SMART Notebook ke stažení:

- ✓ <https://www.smarttech.com/products/education-software/>

Videozáznamy webinářů autora textu s tematikou interaktivní výuky:

- ✓ <https://www.petrpexa.cz/webinare.php>

Ukázky mnoha hotových výukových aplikací resp. žákovských aktivit:

- ✓ <https://www.petrpexa.cz/smart/ukazky.notebook>
- ✓ https://www.petrpexa.cz/smart/ukazky_lab.notebook

E-learningový kurz autora textu s tématem interaktivní výuky:

- ✓ <https://moodle.pf.jcu.cz/course/view.php?id=603>
- ✓ přístup pro hosty, přístupový klíč: dvpp

Tvorba a použití žákovských aktivit v online prostředí:

- ✓ <https://www.smarttech.com/lumio/>
- ✓ přihlášení přes vlastní Google nebo Office 365 účet



Doporučené pomůcky

- ✓ počítač s nainstalovaným programem SMART Notebook v aktuální verzi s doplňky SMART Lab a Response (ke stažení z <https://www.smarttech.com/products/education-software/>)
- ✓ vlastní mobilní zařízení s přístupem k internetu



14

WORKSHOP CODEY ROCKY

Základní instrukce

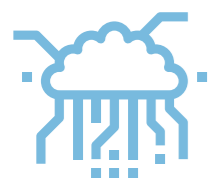
Tento kurz je doporučován pro žáky od třetí do šesté třídy základní školy nebo nižší ročníky osmiletých gymnázií. Autor má zkušenosti s výukou této látky od třetí třídy, ale je přesvědčen, že je možné toto téma učit i v nižších ročnících po menší transformaci kurzu.

Časová dotace tohoto kurzu nemůže být zcela jednoznačná. Závísí na tom, zda se studenti již seznámili s grafickým programováním a zda již mají nějaké zkušenosti s ovládáním programovatelných robotů. Učitel by měl nejprve prostudovat všechny materiály dané k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.

Níže naleznete doporučený průběh a časovou dataci. Kurz lze rozdělit do pěti částí:

1. Blokové programování – první seznámení s jazykem Scratch
 - V případě, že už žáci umí tento programovací jazyk používat a znají základní bloky, může se tahle část přeskočit.
 - Pro zopakování nebo nové vysvětlení postačí jedna vyučovací hodina (45 minut).
2. První jízda s robotem – vysvětlení nejen pohybových bloků, ale také čím program musíme startovat
 - První ukázce jízdy a určování trasy bohatě stačí opět jedna vyučovací hodina. Další lekce navazují na předešlé znalosti a pokaždé se něco přidá.
3. Náladový robot – práce s LED displejem, který dokáže zobrazovat nejen robotovu náladu
 - Pro třetí a čtvrtou část bude stačit jedna vyučovací hodina (45 minut).
4. Rozpoznávání barvy – základní podmínkové bloky a co robot Codey Rocky má dělat, když rozpozná danou barvu
 - Pro třetí a čtvrtou část bude stačit jedna vyučovací hodina (45 minut).
5. Předávání zprávy – bezdrátová komunikace mezi dvěma nebo více roboty, kteří si dokáží předávat zprávu
 - Tato část je vrcholem kurzu a také už je potřeba, aby žáci měli znalosti z předchozích částí. Doporučeno je věnovat téhle části celou vyučovací hodinu (45 minut).

Minimální doba, za kterou lze tuto výukovou oblast absolvovat, je tedy dvě vyučovací hodiny, ale ideální jsou tři až čtyři (135–180 minut).



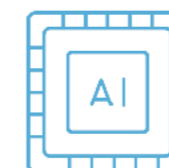
Cílem workshopu je představit moderního programovatelného robota Codey Rocky primárně určeného pro výuku blokového programování na základní škole. Tento robot kombinuje snadno použitelný robotický hardware nejen s blokovým programovacím softwarem. Nejjednodušší možností ovládání robota je přes mobilní aplikaci, která umožňuje jednoduché ovládání jako přes joystick a s přednastavenými módy pohybu. V našem kurzu se zaměříme na programovací jazyk Scratch, který je nejrozšířenějším programovacím jazykem pro malé programátory. Začneme od úplných základů, ale postupně se přeneseme až k možnosti komunikace dvou a více robotů.



Kódování



Počítání



**Umělá
inteligence
(AI)**



**Internet
věcí
(IoT)**



STEM

Teoretická část k dané problematice

Makeblock Codey Rocky kombinuje snadno použitelný robotický hardware s několika programovacími jazyky. Nejjednodušším způsobem ovládní robota je využití mobilní aplikace, která s několika přednastavenými módy pohybu umožňuje intuitivní ovládní robota přes joystick. Díky blokovému programování (Scratch) mohou robota programovat i naprostí začátečníci. Codey Rocky navíc podporuje také **AI (umělou inteligenci)** a **IoT (internet věcí)**, díky nimž si studenti mohou rozšiřovat základní STEM dovednosti.



Obrázek 1: horní část robota Codey, spodní část robota Rocky

Vlastnosti robota Codey Rocky

K dispozici je snímač zvuku, světla a barev, 6osý gyroskop, IR přijímač/vysílač, mikrofon a LED displej vyjadřující jeho pocity. Robota lze programovat v programovacím prostředí **mBlock** pomocí jazyků **Scratch** nebo **Python**. Prostředí Makeblock je k dispozici jak pro tablet nebo smartphone tak i pro počítač. Bezdrátová komunikace je u robota samozřejmostí. Pomocí Bluetooth se mobilní nebo počítačová aplikace dokáže spojit s robotem a lze na dálku robota ovládat nebo odesílat vytvořené programy. Díky vestavěné Wi-Fi se robot dokáže připojit k internetu a může si stahovat potřebná doplňková data, jako jsou třeba údaje o času a počasí.

Výhody robota Codey Rocky

- ✓ Díky LED obrazovce můžete volně vytvářet zvukové a světelné efekty
- ✓ Kompatibilní s bloky Makeblock Neuron a s LEGO stavebnicemi
- ✓ Bluetooth Dongle pro bezdrátové nahrávání
- ✓ Základní robot pro orientaci v AI a IoT technologiích

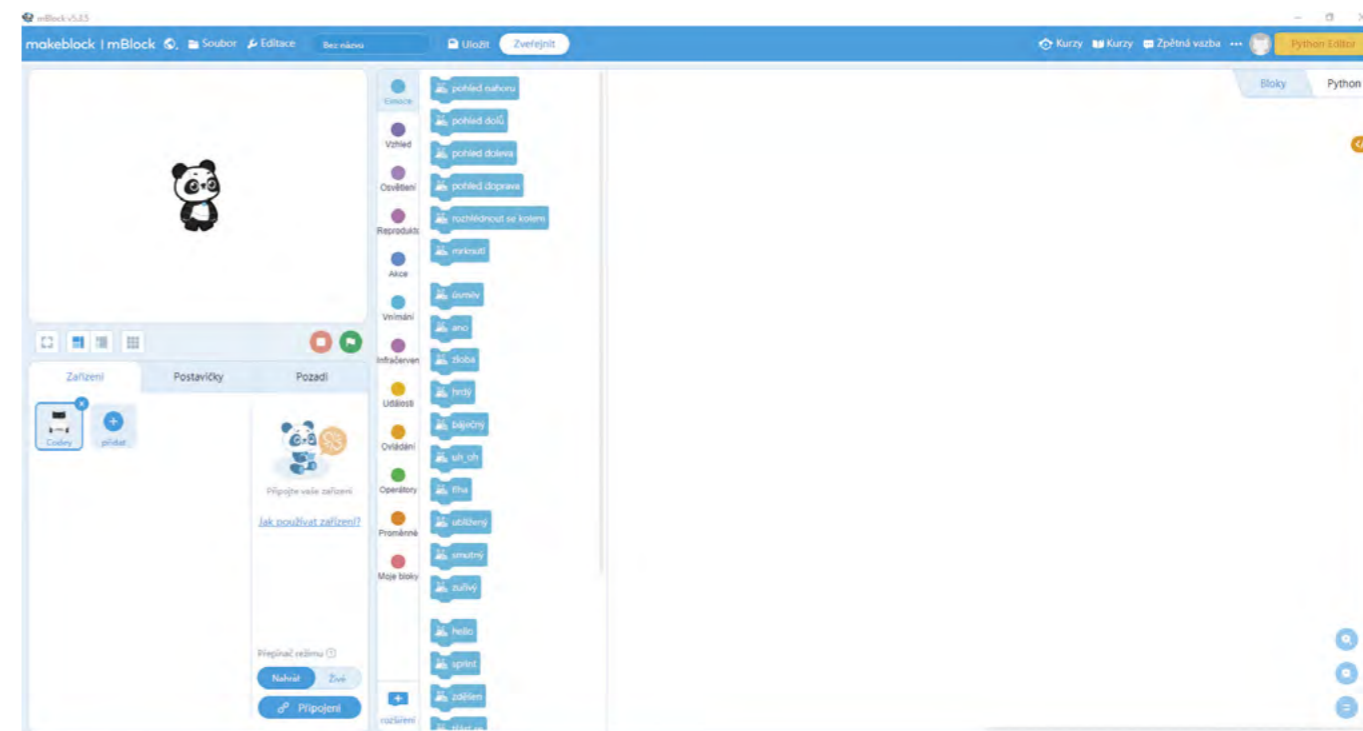
Získané dovednosti

- ✓ Znalost informačních a komunikačních technologií
- ✓ Spolupráce
- ✓ Logické myšlení
- ✓ Výpočetní myšlení

Metodická a didaktická část

Workshop lze podle potřeby rozdělit na 4 nebo na 5 částí. Jde o to, zda účastníci, již mají alespoň základní povědomí o blokovém programování. Každá lekce je o kus těžší a pokaždé se přidá jiná část blokového programování.

Blokové programování



Obrázek 2: Programovací prostředí mBlock

První část nás provede začátky blokového programování neboli jazykem Scratch. V případě, že už znáte základní bloky, můžete tuhle část klidně přeskočit. Codey Rocky má své vlastní programovací prostředí s názvem **mBlock**. <https://mblock.makeblock.com/en-us/>

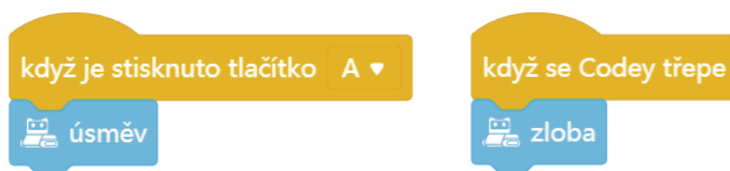
Cílem této části je účastníka:

- ✓ Seznámit s grafickým programovacím jazykem Scratch
- ✓ Vysvětlit jednotlivé kategorie bloků
- ✓ Vyzkoušet si základní bloky

Teď si ukážeme jednotlivé kategorie, které jsou velmi podobné právě grafickému programovacímu jazyku s názvem Scratch. Vysvětlíme si základy a také jak se dané bloky používají.

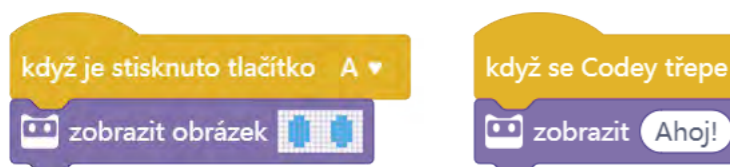
Emoce

Tato kategorie bloků slouží k programování, jak už název napovídá, emocím robota. Dokážeme přednastavenými programy určit nejen obličej, který robot ukáže na displeji, ale také pohyb a zvuk, který k dané emoci patří.



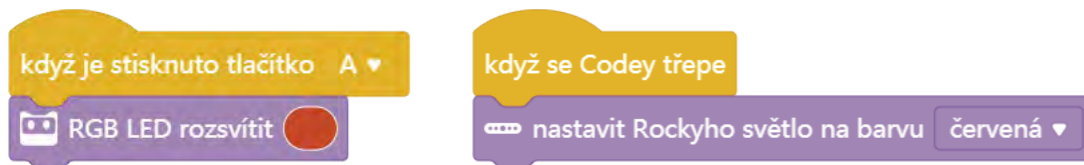
Vzhled

Zde najdete potřebné bloky, když potřebujete naprogramovat specifický výraz, obrázek nebo dokonce text, aby běžel na displeji Codey Rocky.



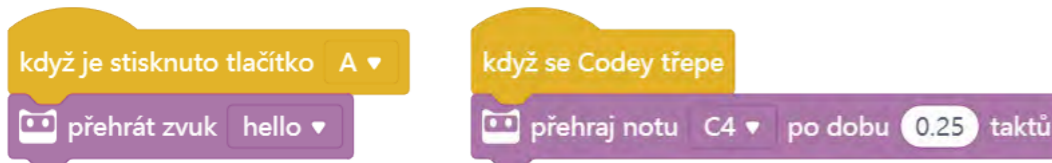
Osvětlení

Jak už bylo v popisu robota napsáno, tak na robotovi najdete jeden LED indikátor, který je také programovatelný. V této kategorii najdete pro to potřebné bloky. Dále má základna Rocky otočný programovatelný senzor barvy, ale má tam taky jednu programovatelnou RGB diodu, kterou lze také ovládat pomocí těchto bloků.



Reproduktor

Podle názvu této kategorie je patrné, že níže zmíněnými bloky můžeme ovládat zabudovaný reproduktor. Naleznete zde přednastavené bloky, které snadno zapojíte do vašeho programu.



Akce

Tato kategorie bloků bude asi pro vás nejdůležitější, protože díky těmto blokům se konečně váš robot bude moct hýbat. Později v kapitole „Prvně jezdíme“ si tyto bloky rozebereme podrobněji a ukážeme si první jízdu s robotem Codey Rocky.



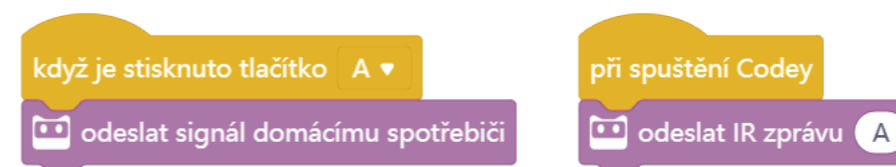
Vnímání

Bloky v této kategorii sami o sobě jen hlídají stavy senzorů. Proto je například zařazujeme k podmínkám nebo k operátorům, kde po výpočtu nebo podmínce splní své úlohy a podle příkazu se vykoná další část programu.



Infračervený

V této kategorii najdete bloky, kterými lze programovat infračervené moduly, které jsou na robotovi. Jeden slouží pro příjem signálu a druhý pro vysílání signálu.



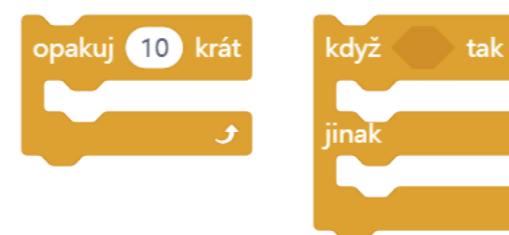
Události

Důležité je každý program začínat blokem právě z této kategorie. Bez tohoto bloku vám program fungovat nebude.



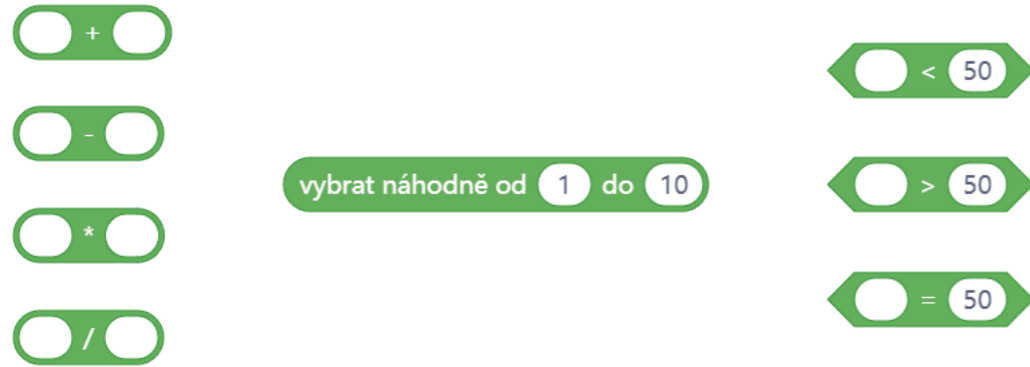
Ovládání

Oranžová barva těchto bloků je právě pro bloky, které v programu udávají určitou podmínku. Tyto bloky můžeme rozdělit na různé situace, které podmiňují. Pro příklad je zde blok, který dokáže opakovat dané bloky, které se nacházejí uvnitř podmínky. Druhá ukázka je podmínky (pro příklad z jazyka Python – if else), která za nějakého předpokladu vykoná první příkaz, když není podmínka splněna, tak se splní příkaz, který je ve slotu jinak.



Operátory

V této kategorii bloků se nachází základní matematické bloky, které v hlavních programech nebo podmínkách pomáhají určovat hodnoty sledovaných veličin. Najdete zde například: Aritmetické a logické operátory, základní matematické funkce nebo třeba náhodná čísla.



Proměnné a Moje bloky

Grafický programovací jazyk Scratch také nabízí, že jakmile máte svoji proměnnou, kterou potřebujete hlídat jinak než třeba operátory, tak v této kategoriích si vytvoříte svůj vlastní blok, který můžete následně používat do vašich programů.

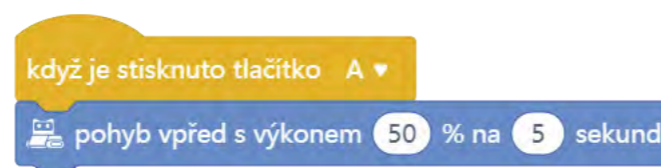
První jízda

Zde si představíme základní pohybové bloky a ukážeme si možnosti, která představují. Na konci této části již budeme schopni s robotem jezdit tam, kam jsme mu zadali v programu. Nejprve si ukážeme základní pohyb dopředu a dozadu.

Cílem této části je:

- ✓ Představit si základní pohybové bloky
- ✓ Vytvořit si první jednoduché pohybové programy a otestovat je
- ✓ Ukázat nahrávání programů do robota

Jak už v předešlé kapitole bylo napsáno, tak každý program nám musí začínat blokem ze žluté kategorie Události. My v naší ukázce budeme používat blok „když je stisknuto tlačítko (A)“. Přesuneme daný blok do programovacího pole. Dále v kategorii Akce najdeme blok s názvem „pohyb vpřed s výkonem x % na x sekund“. Tento blok si také přesuneme do programovacího pole a když najedeme poblíž žlutého bloku, tak se nám automaticky spojí.

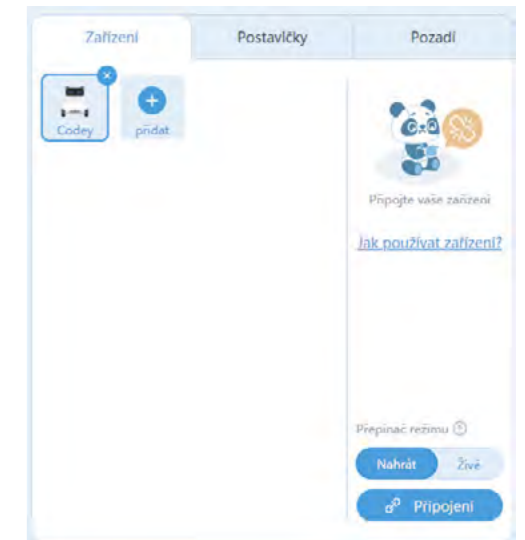


Obrázek 3: Pro ukázkou použijeme výkon 50 % a 5 sekund pro pohyb vpřed.

Jakmile máme připravený první program, připojíme robota rohu pomocí kabelu microUSB a v programu mBlock, v levém dolním rohu klikneme na Nahrát a poté na Připojení. Tím se nám robot připojí k programu mBlock a můžeme nahrávat program.

Aby nám robot nejezdil jen dopředu, zkusíme si ještě jízdu vzad. Je to jednoduché, jen vyměníme modrý blok za pohyb vzad s výkonem 50 % na 5 sekund.

Určitě si vyzkoušejte i další pohyby, jako je například zatáčení.



Obrázek 4: Ukázka programu mBlock

Náladový robot

V této další kapitole si ukážeme, jak lze pracovat s robotem, aby střídal nálady, které si vybereme ze seznamu bloků nebo mu třeba vytvoříme své vlastní.

Cílem této části je:

- ✓ Seznámit s možnostmi displeje a možnostmi mobilní/počítačové aplikace
- ✓ Vytvořit sadu obličejů pro robota
- ✓ Pomocí programu zpracovat různé nálady

Máme dvě možnosti, jak se můžeme na tento úkol podívat. Ta první je, že vývojáři mBlock již vytvořili velkou zásobu bloků emocí. Z těch základních si můžeme představit například úsměv, zloba, spánek nebo třeba závrať. Když chceme tyto bloky použít opět musíme před modré bloky použít nějaký start programu.



V druhém případě si můžeme vytvořit vlastní programy, které budou představovat buď jiné anebo již vytvořené emoce, které se v této kategorii nachází.

Sice je tato kapitola kratší na vysvětlování, ale o to bych dal dětem mnohem více času si s robotem pohrát, aby si mohly projít všechny nabízené emoce. Ti zdatnější si určitě můžou vyzkoušet naprogramovat svůj vlastní program, který může představovat třeba více emocí za sebou nebo nemusí využívat tyto modré bloky vůbec, ale můžou si pomocí základních programovacích bloků vytvořit vlastní.

Rozpoznávání barvy

Touto částí se dostáváme o krok výše a jelikož se v sadě s roboty nachází i barevné kartičky tak si ukážeme, jak je lze také používat.

Cílem této části je:

- ✓ Seznámit se světelným senzorem na základně Rocky
- ✓ Vysvětlení podmínkových bloků
- ✓ Vyzkoušet si základní jednoduché programy pro rozpoznání barvy

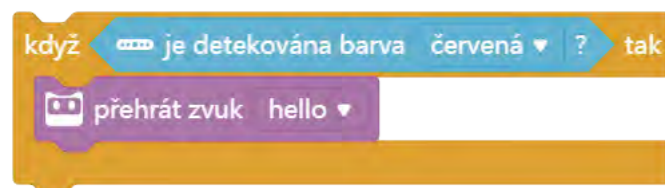
V této kapitole již zapojíme do programu také podmínky a cykly. Co tyto dva pojmy znamenají najdete v kapitole o programovacím jazyku Scratch. Podrobně ukázaný a celý program najdete v pracovním listě č. 3.

Světelný senzor se nachází na základně Rocky, kde se signál přenáší do mini PC Codey přes magnetické kontakty. Tento senzor spojuje více funkcí dohromady. Dokáže totiž nejen rozpoznávat barvy, ale také detekovat překážky nebo sledovat čáru.

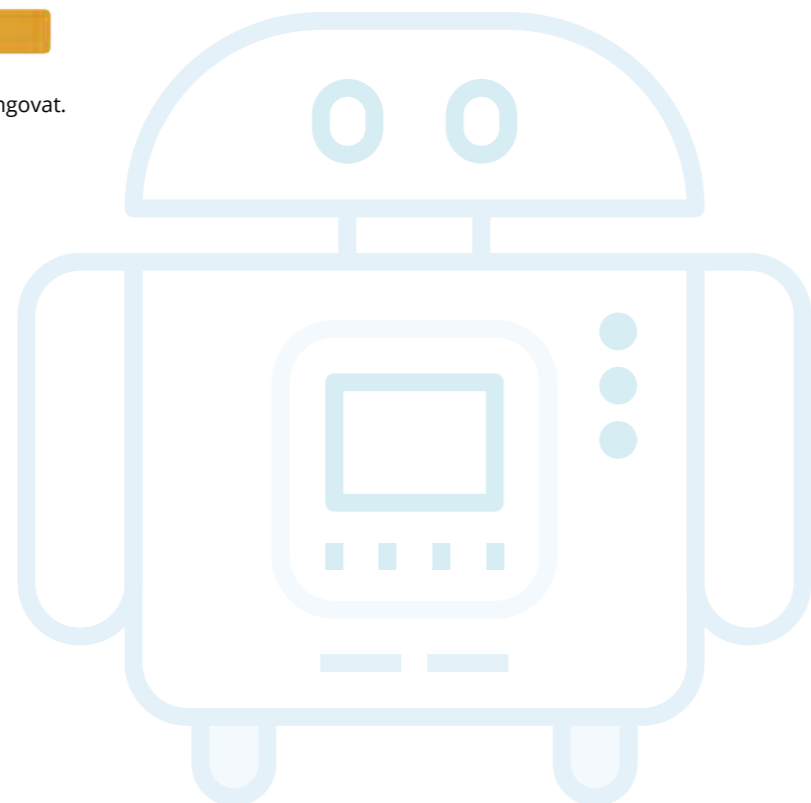


Obrázek 5: Světelný senzor

Hlavními bloky, které budou v tomto programu, jsou z kategorie *Vnímání* a *Ovládání*. Pro práci se světelným senzorem budeme používat blok „je detekována barva...?“, „Jakmile ho vhodně umístíme do podmínky „když... tak“, tak blok dokáže reagovat a následně vykonávat to, co je uvnitř podmínky.



Obrázek 6: Ukázka podmínky, jak může rozpoznávání barvy fungovat.



Předávání zprávy

Díky infračervenému přijímači a vysílači dokáží roboti mezi sebou komunikovat tak, že například jeden vyšle zprávu a druhý ji přijme a vykoná ji. V našem příkladu si ukážeme, jak dva roboti si dokáží předávat zprávu a zobrazovat různé hodnoty na displeji.

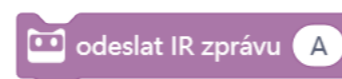
Cílem této části je:

- ✓ Vysvětlit si předávání zprávy
- ✓ S pracovním listem si kapitolu otestovat

My si teď vyzkoušíme „přelévat vodu“ mezi Codey. Úkolem bude, aby jeden Codey vyslal zprávu o přelítí a na druhém Codey se na displeji bude zobrazovat, že se hladina zvyšuje. Poslouží nám pro příklad tyto bloky. Nejprve si ukážeme bloky pro prvního robota a poté pro druhého.



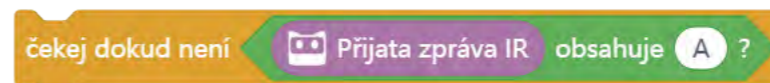
Nejprve v programu vytvoříme podmínku (čekej dokud není). Tím nám program zajistí, že dokud nebude robot nakloněn, tak se nic nestane. Modrý blok uvnitř podmínky je z kategorie bloků *Vnímání*.



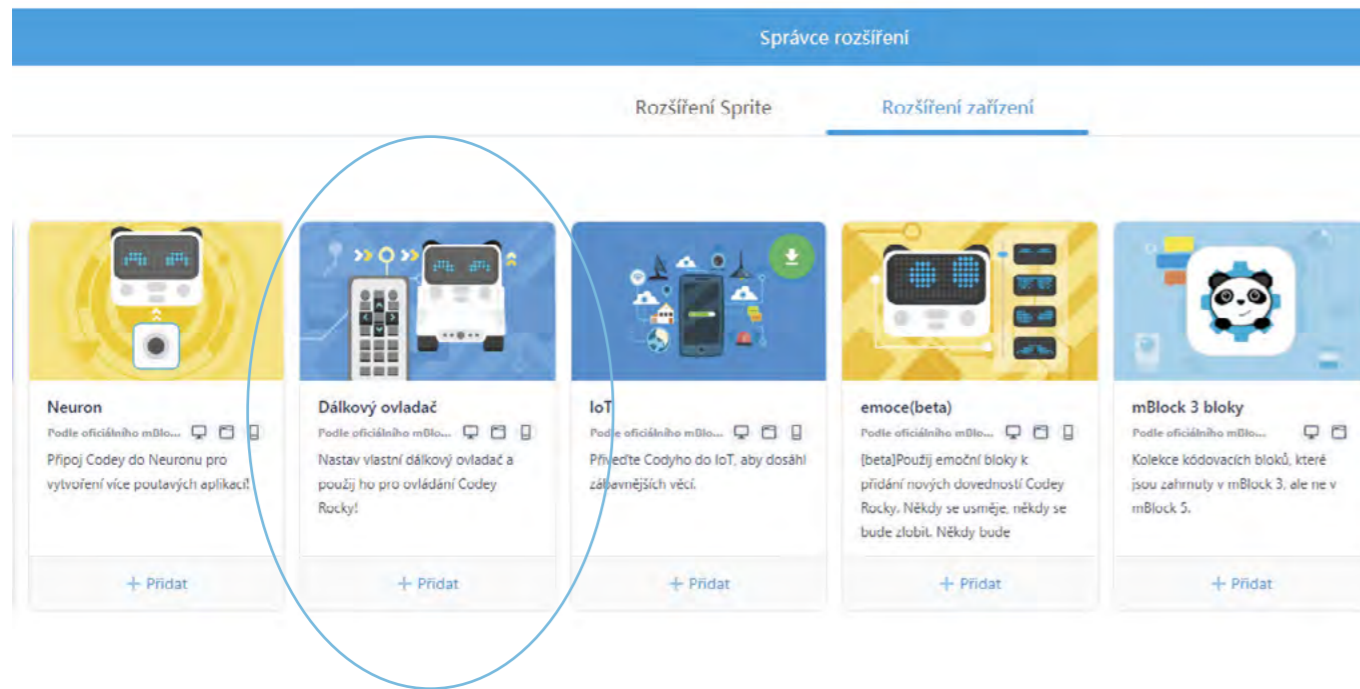
Poté musíme do programu přidat blok, který zajistí, že v okamžiku, kdy je na bloku řada, vyšle zprávu, že může druhý robot pokračovat se svým programem.



Pro druhého robota použijeme podobné bloky, ale v tuhle chvíli bude program čekat, než dostane signál od prvního robota. S tím nám pomůže blok z kategorie *Operátory* a opět musí být umístěn do podmínky.



Pro zajímavost lze infračervené moduly využít třeba pro ovládání domácích spotřebičů nebo lze robota ovládat dálkovým ovladačem. Stačí si je přidat do nabídky bloků kategorie *IR Remote / Dálkový ovladač*.



Doporučené pomůcky

Při workshopech se osvědčilo, když na jednoho robota byli maximálně dva žáci. Společně si pomáhají, doplňují se a v případě, že zrovna nerozumí jeden z nich programu, tak může druhý pomoci a vysvětlit.

Při zkoušení jízdy robota je dobré, když si žáci postaví sami dráhu, kterou mohou následně projíždět a upravovat jejich programy tak, aby robot zbytečně nenarážel do překážek.

Pracovní list 1 První jízda

když je stisknuto tlačítko A ▾

pohyb vpřed s výkonem 50 % na 1 sekund

když se Codey třepe

otočit vlevo s výkonem 50 % na 1 sekund

když je stisknuto tlačítko A ▾

pohyb vpřed s výkonem 50 % na 5 sekund

pohyb vzad s výkonem 50 % na 5 sekund

otočit vlevo s výkonem 50 % na 2 sekund

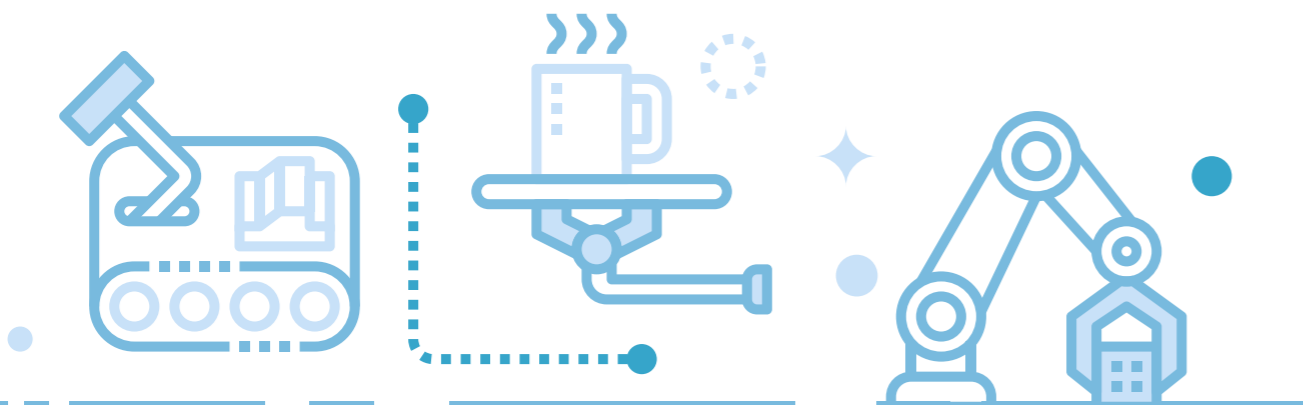
otočit vpravo s výkonem 50 % na 2 sekund

když je stisknuto tlačítko A ▾

pohyb vpřed s výkonem 50 % na 5 sekund

otočit vlevo 180 stupňů dokud není hotovo

pohyb vpřed s výkonem 50 % na 5 sekund



Pracovní list 2

Náladový robot

když je stisknuto tlačítko A ▾

- prehrát zvuk přepnout ▾
- zobrazit obrázek

když je stisknuto tlačítko C ▾

- prehrát zvuk přepnout ▾
- zobrazit obrázek

Pracovní list 3

Rozpoznání barvy

když je stisknuto tlačítko A ▾

- zobrazit obrázek

opakuj stále

- když je detekována barva zelená ▾ ? tak
 - pohyb vpřed ▾ s výkonem 30 %
- když je detekována barva modrá ▾ ? tak
 - otočit vlevo ↶ 90 stupňů dokud není hotovo
- když je detekována barva žlutá ▾ ? tak
 - otočit vpravo ↷ 90 stupňů dokud není hotovo
- když je detekována barva červená ▾ ? tak
 - zastavit

když je stisknuto tlačítko C ▾

- zastavit
- vypnout obrazovku
- zastavit vše ▾

Pracovní list 4

Předávání zprávy

když je stisknuto tlačítko A ▾

zobrazit obrázek

čekej dokud není je Codey nakloněn uši dolů ▾ ?

odeslat IR zprávu A

zobrazit obrázek po dobu 0.3 s

zobrazit obrázek po dobu 0.3 s

zobrazit obrázek po dobu 0.3 s

zobrazit obrázek po dobu 0.3 s

zobrazit obrázek

když je stisknuto tlačítko M ▾

čekej dokud není Přijata zpráva IR obsahuje A ?

počkat 0.2 sekund

zobrazit obrázek po dobu 0.3 s

zobrazit obrázek po dobu 0.3 s

zobrazit obrázek po dobu 0.3 s

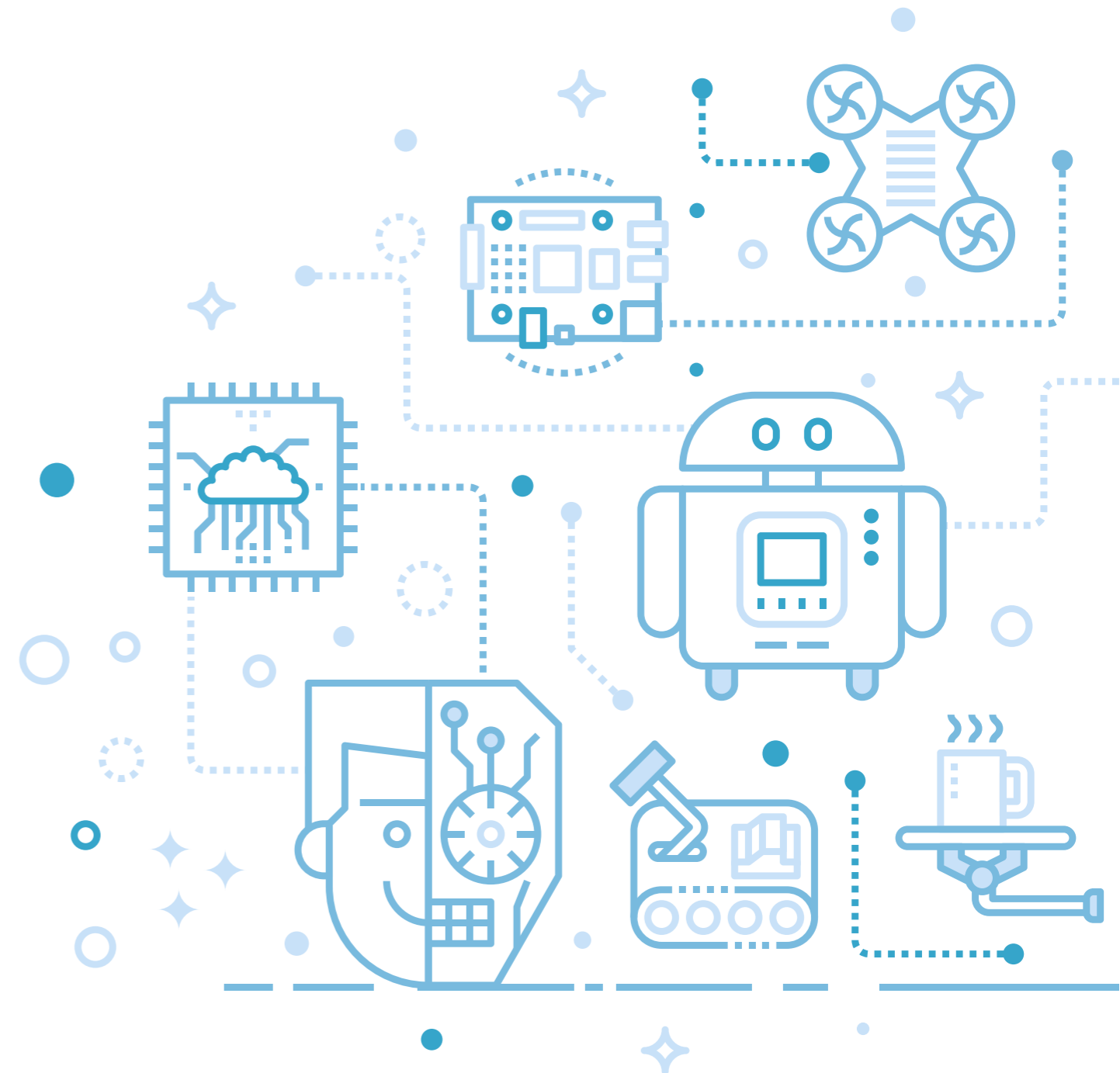
zobrazit obrázek po dobu 0.3 s

zobrazit obrázek

když je stisknuto tlačítko C ▾

vypnout obrazovku

zastavit vše ▾



15 KYBERNETICKÁ BEZPEČNOST PRO ZŠ

Základní instrukce

Tento kurz je doporučován pro žáky druhého stupně základní školy a nižší ročníky osmiletých gymnázií. V přeneseném smyslu lze problematiku aplikovat také na žáky prvního stupně, kteří disponují chytrými zařízeními připojenými bez kontroly k internetu (smartphone, tablet, notebook apod.) Kurz je koncipován jako základní a je vstupenkou do uvědomělého chování v online prostředí, což je základní kámen kybernetické bezpečnosti.

Časová dotace tohoto kurzu je stanovena až na 7 hodin. Rozdělení 3+3(4) hodiny na teoretický a legislativní základ a získání praktických dovedností kybernetické bezpečnosti. Učitel by měl nejprve prostudovat všechny materiály dané k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.



Cílem workshopu je na praktických ukázkách ukázat důsledky nezodpovědného chování v online prostoru. Workshop bude v první, teoretické části cílen na osvětu základních návyků chování na internetu. V části praktické pak bude osvětleno, jakým způsobem lze vysledovat informace o online chování jednotlivých uživatelů, jak lze získané informace propojit a jak je lze využít. V závěru bude na reálných případech ukázáno, kam může takovéto nezodpovědné chování vést.

Doporučený průběh a časové dotace podrobněji. Kurz lze rozčlenit do následujících částí:

1. Teoretický a legislativní úvod – především vymezení mantinelů a pojmu kybernetická bezpečnost, zákon o kybernetické bezpečnosti, co lze a nelze na veřejné síti provádět.
 - a. Doporučuji jednu vyučovací hodinu.
2. Sdílení informací, digitální stopa a fotografická metadata – objasnění pojmů, jak vzniká digitální stopa, jak se dá vysledovat, co může prozradit a k čemu to může vést.
 - a. Doporučuji také jednu až dvě vyučovací hodiny. Záleží na tom, zdali žáci jsou s tímto pojmem a problematikou k němu se vztahující obeznámeni či nikoliv.
3. Praktické ukázky důsledků nezodpovědného chování v online prostoru na konkrétních příkladech včetně jejich podrobné analýzy a diskuse. Co daný jedinec udělal špatně, k čemu to vedlo a jak tomu předejít. Sdílení vybraných odkazů s žáky nad rámec pracovních listů.
 - a. Doporučuji dvě vyučovací hodiny.
4. Plnění zadaných úkolů dle pracovních listů. Připraveny jsou dva, práce s fotografickými metadaty a důsledky kyberšikany - viz metodická a didaktická část.
 - a. Pro každý list doporučuji jednu vyučovací hodinu.

Pro co největší dopad praktické části je více než vhodné zjistit, jací žáci budou v rámci výuky tohoto tématu vzdělávání a kolik veřejně dostupných informací lze o jednotlivých osobách (manuálně či pomocí robotické scraperu) dohledat. Z nich doporučuji vybrat jedince, o kterém lze dohledat ve veřejných informacích co nejvíce (jehož digitální stopa je největší) a na jeho příkladě demonstrovat slabiny, které mohou vést k případnému zneužití.

Autoři:

Ing. Rudolf Vohnout, Ph.D., Přírodovědecká fakulta, Jihočeská univerzita,

Ing. Petr Břehovský, Přírodovědecká fakulta, Jihočeská univerzita

Editor: doc. RNDr. Ing. Jana Kalová, Ph.D.

Teoretická část k dané problematice

Kybernetická bezpečnost

Nejprve je nutné si definovat pojem kyberprostor. Pro drtivou většinu laické veřejnosti KYBERPROSTOR = INTERNET = WEB, ale kybernetickým prostorem jsou vymezeny všechny počítačové systémy, služby (včetně aplikací a protokolů), uživatelé a data, která se v online světě nacházejí (a lze je tak využít k získání dílčích informací nutných pro definování digitální stopy).

Kybernetická bezpečnost nemá ucelenou definici. Zásadní je si uvědomit, že vždy představuje nějaký soubor opatření vedoucích k ochraně před kriminálním či neautorizovaným užitím elektronických dat. Jinými slovy, přijmutí takových opatření, která vedou k dosažení tohoto stavu. Tato opatření mohou být jak bezpečnostní, tak technické či organizační (administrativní) povahy.

*Souhrn právních, organizačních, technických a vzdělávacích prostředků, které směřují k zajištění **ochrany počítačových systémů a dalších prvků ICT, aplikací, dat a uživatelů***

+

schopnost počítačových systémů a využívaných služeb reagovat na kybernetické hrozby či útoky a jejich následky, jakož i plánování obnovy funkčnosti počítačových systémů a služeb s nimi spojených.

Dále je důležité se seznámit s kybernetickou hrozbou. Jedná se o možnost škodlivého pokusu o poškození nebo narušení počítačové sítě nebo systému. Kybernetickou hrozbou lze také definovat jako akt směřující ke změně informace, aplikací či systému samotného. Kybernetická hrozba může způsobit:

- Únik informace – stav, kdy dojde k vyrazení chráněné informace neautorizovanému subjektu.
- Narušení integrity – poškození, změna, či vymazání dat.
- Potlačení služby – úmyslné bránění v přístupu k informacím, aplikacím, či systému.
- Nelegitimní použití je užití informací neautorizovaným subjektem či neoprávněným způsobem.

„Ten, kdo se ve jménu bezpečnosti vzdává svobody, nezaslouží si ani svobodu, ani bezpečnost.“

Benjamin Franklin

Základní legislativa

- Zákon č. 181/2014 Sb., o kybernetické bezpečnosti a o změně souvisejících zákonů (zákon o kybernetické bezpečnosti).
- vyhláška č. 82/2018 Sb., o bezpečnostních opatřeních, kybernetických bezpečnostních incidentech, reaktivních opatřeních, náležitostech podání v oblasti kybernetické bezpečnosti a likvidaci dat (vyhláška o kybernetické bezpečnosti)
- Zákon č. 110/2019 Sb. Zákon o zpracování osobních údajů (aka GDPR)
- Zákon č. 40/2009 Sb., trestní zákoník, ve znění pozdějších předpisů

„Každý má právo na ochranu před NEOPRÁVNĚNÝM shromažďováním, zveřejňováním nebo jiným zneužíváním údajů o své osobě.“

Listina základních práv a svobod v čl. 10 odst. 2 a 3

Sdílení informací

Fyzické osoby samy a dobrovolně o sobě zveřejňují stále větší množství dat (fotografie, videa aj.), přičemž k distribuci těchto dat typicky využívají služby informační společnosti.

Nejvíce jsou osobní údaje zveřejňovány v rámci sociálních sítí, které z podstaty své funkce takovéto zveřejňování předpokládají a zakotvují ve smluvních podmínkách pravidla, na základě kterých je s takovými daty zacházeno.

Je nutné si uvědomit, že v dnešní době je většina fotografií pořizována chytrými telefony, které mají v sobě automaticky aktivovanou funkci geolokace [1]. Každá fotografie, která je zveřejněna v originální podobě si s sebou tuto informaci ve formě metadat nese [2]! Pokud tedy zveřejníte (bez potřebného nastavení zabezpečení sdílení [3]) na sociální síti, že Váš otec si právě koupil nové Porsche, je automaticky známo, kde jej také parkuje.

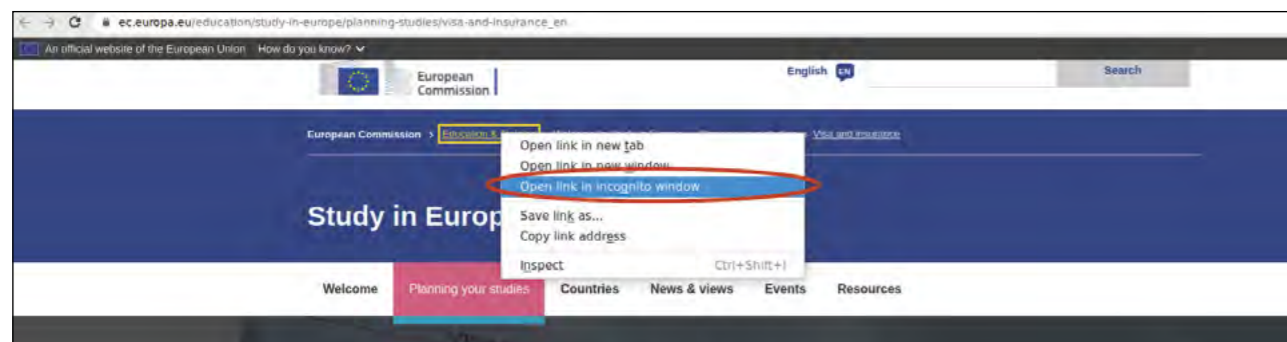
Nezapomeňte na dvě pomůcky:

- Napomáhání trestné činnosti není zákonem dovoleno.
- Neznalost zákona neomlouvá.

Digitální stopa

Veškerá aktivita v online prostoru zanechává digitální stopu [1]. Čím více času a aktivit v online světě děláme, tím je naše digitální stopa větší. [2] Digitální stopa (kromě příkladů uvedených v části „Příklady z praxe“) je poté využívána jako vstupní data pro algoritmy, které mají za úkol predikovat naše budoucí chování na internetu. To se využívá například u cílené reklamy, která poté slouží k co nejpřesnějším nabídkám na míru daného jedince. [3]

Pokud chcete svoji digitální stopu alespoň omezit (nikoliv eliminovat), používejte anonymní surfování. Nabízí jej každý prohlížeč.



Obrázek 1: Anonymní okno

Fotografická metadata

`rvohnout@rvohnout-Latitude-E7440:~/ exif IMG_20210722_132304594.jpg`

Na další straně je uveden výpis tzv. EXIF informací, což jsou metadata (data o datech) vybrané fotografie. Jsou to informace, které fotoaparát automaticky zapisuje při pořízení fotografie a jsou její nedílnou součástí. EXIF k fotografii tak například přiřadí:

- informace o modelu fotoaparátu, kterým jste snímek pořídili, a o jeho výrobci,
- datum a čas pořízení snímku,
- údaje o rozlišení fotografie,
- GPS souřadnice,
- náhled fotografie.

Na další straně je uveden pouze příklad těch údajů, které by mohly být potenciálně využity v oblasti kybernetické bezpečnosti. Informací je daleko více (viz obrázek 1 či [4]). Každý žák by také měl vědět, jak se dá u jeho chytrého telefonu pořizování těchto informací automaticky vypnout (pro jednoduchost uvádím pouze odkaz na nejrozšířenější platformu Android, resp. Google program „Photos“). [5]

```
EXIF tags in 'IMG_20210722_132304594.jpg' ('Motorola' byte order):
```

Tag	Value
Manufacturer	Motorola
Model	Moto G(7) Plus
Orientation	Right-top
X-Resolution	72
Y-Resolution	72
Resolution Unit	Inch
Software	lake_reteu_n-user 10 0PWS30.61-21-18-7 fac4a release-keys
Date and Time	2021:07:22 13:23:05
YCbCr Positioning	Centred
Compression	JPEG compression
Orientation	Right-top
X-Resolution	72
Y-Resolution	72
Resolution Unit	Inch
Exposure Time	1/50 sec.
F-Number	f/1.7
ISO Speed Ratings	139
Exif Version	Exif Version 2.2
Date and Time (Original)	2021:07:22 13:23:05
Date and Time (Digitized)	2021:07:22 13:23:05
Components Configuration	Y Cb Cr -
Shutter Speed	5,64 EV (1/49 sec.)
Aperture	1,53 EV (f/1.7)
Brightness	1,95 EV (13,24 cd/m²)
Exposure Bias	0,00 EV
Metering Mode	Centre-weighted average
Flash	Flash did not fire, compulsory flash mode
Focal Length	4,3 mm
Maker Note	1115 bytes undefined data
Sub-second Time	257539
Sub-second Time (Original)	257539
Sub-second Time (Digitized)	257539
FlashPixVersion	FlashPix Version 1.0
Colour Space	sRGB
Pixel X Dimension	4608
Pixel Y Dimension	3456
Sensing Method	One-chip colour area sensor
Scene Type	Directly photographed
Exposure Mode	Auto exposure
White Balance	Auto white balance
Digital Zoom Ratio	1,00
Scene Capture Type	Standard
GPS Tag Version	2.2.0.0
North or South Latitude	N
Latitude	48, 58, 39,0503
East or West Longitude	E
Longitude	14, 27, 0,5975
Altitude Reference	Sea level
Altitude	425,216
GPS Time (Atomic Clock)	11:23:04,00
Geodetic Survey Data	WGS-84
Name of GPS Process	12 bytes undefined data
GPS Date	2021:07:22
Interoperability Index	R98
Interoperability Version	0100

EXIF data contains a thumbnail (13929 bytes).

Obrázek 2: EXIF informace

Doporučené odkazy a materiály pro hlubší porozumění problematice

[1] <https://www.mall.tv/martyisdead/digitalni-stopa>

[2] <https://blog.avast.com/cs/what-is-your-digital-footprint-avast>

[3] <https://www.youtube.com/watch?v=gcimRZF8g3Y>

[4] <https://www.milujemefotografii.cz/jak-rozumet-exifu-co-jsou-metadata>

[5] <https://support.google.com/photos/answer/6153599>

Příklady z praxe

Nedávej na Facebook nic, co bys neřekl své babičce, se říkávalo... Realističtější rada pro dnešní dny je – nedávejte na sociální sítě nic, co by neměl vidět Váš současný nebo budoucí zaměstnavatel, partner, či rodič.

Zásadní je si uvědomit, že to, co veřejně vystavíte na internet, je již navždy zveřejněno. Nic jako smazání dat neexistuje. [1]. V době, kdy jsme na škole, bohužel většinou neřešíme, že fotografie, které vystavujeme, mohou za 10 let být důvodem pro např. nepřijetí na vysoce společensky hodnocenou pozici.

Vystavení fotografie bez vědomí toho, kdo je na ní zachycen, je porušením nařízení GDPR (v ČR implementováno do legislativního prostoru jako Zákon č. 110/2019 Sb.). Vystavení takovéto fotografie může ovšem vést k vzednutí vlny nenávistných reakcí, protože je dokázáno, že lidé se v online prostoru chovají jinak než v realitě. To může způsobit dotyčnému psychickou újmu, která v krajním případě může vést až k vyhledání psychologa, psychiatra či pokusu o sebevraždu. Uvědomte žáky, že si nesou (bez ohledu na věk) vždy následky takového jednání [2].

Relevantní odkazy

[1] <https://web.archive.org/>

[2] <https://www.mall.tv/martyisdead/pribeh-simony-a-kiany-jedna-fotka-na-instagramu-a-spousta-nenavisti>

<https://www.e-bezpeci.cz/index.php/rizikove-jevy-spojene-s-online-komunikaci/kybergrooming/33-112>

<https://o2chytraskola.cz/clanek/25/kybergrooming/4333>

Pro ty, co si chtějí rozšířit obzory:

<https://www.mall.tv/martyisdead/rizika-socialnich-siti>

<https://martyisdead.mall.tv/vyuka/>

<https://kyberbezpecnost.csirt.cz/cs/kyberbezpecnost/pro-uzivatele/>

<https://knihy.nic.cz/files/edice/cybersecurity.pdf>

<https://edu.ceskatelevize.cz/video/119-phishing>

<https://www.csfd.cz/film/812238-socialni-dilema/>

<https://www.documentaryarea.tv/player.php?title=The%20Social%20Dilemma>

<https://o2chytraskola.cz/video/8/nejsem-lovna-film-v-siti>

<https://o2chytraskola.cz/video/1/husta-lida>

https://o.seznam.cz/wp-content/uploads/na_hory_metodika_Seznam_cz.pdf

<https://www.e-bezpeci.cz/index.php/vzdelavani/tiskoviny-pro-ucitele-a-rodice>

http://oldwww.upol.cz/fileadmin/user_upload/PdF/veda-vyzkum-zahr/2016/seminare/Zak_jako_obet_kybergroomingu.pdf

<https://www.mesto-uh.cz/kybergrooming>

Metodická a didaktická část

Na úvod je vhodné uvést, že pro tento kurz je mnohem vhodnější, pokud žáci obdrží pracovní listy v elektronické podobě ve formátu PDF, než vytištěné. Odkazy byly sice zkráceny pomocí nástroje „bit.ly“, ale i přesto přepisováním odkazů do internetového prohlížeče může dojít k překlepům a tudíž zbytečnému zdržování.

Jak bylo vysvětleno v první kapitole, kurz je rozdělen do dvou částí. Postupně se zastavíme u obou a uvedeme, co v nich učít.

Fotografie není pouhý obrázek

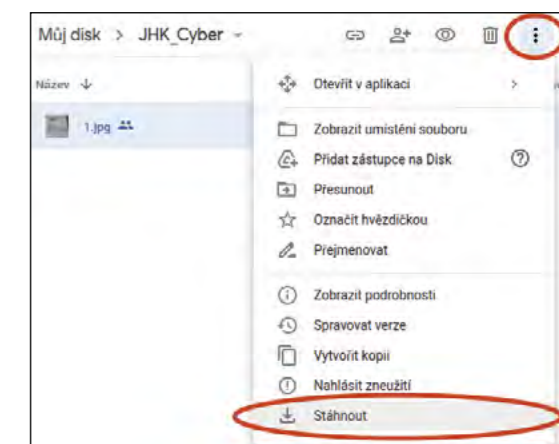
Účelem této části je zjistit, co jsou a jak vypadají metadata z fotografie a jaké všechny informace poskytují.

V této části je třeba zvládnout následující úkoly:

1. Stáhnout si vybranou fotografii z datového úložiště.

Jako úložiště slouží Google Drive s veřejně dostupnou složkou „JHK_Cyber“. Zde je umístěno celkem 12 fotografií (pojmenovaných „1.jpg“ – „12.jpg“). Každá z nich byla pečlivě vybrána tak, aby obsahovala různá metadata, která se k dané fotografii vztahují. Tato metadata jsou původní a nebyla pro účely kurzu nijak účelově modifikována.

Každou z fotografií je nutné nejprve stáhnout. Pro tento účel rozdejte žákům USB flashky, které dostanete během workshopu.



Obrázek 3: Google Drive

Vybranou fotografii je nejprve nutné označit a až poté stáhnout, tak jak mají žáci uvedeno v příslušném pracovním listu. Fotografie (resp. soubor) ať žáci nepřejmenovávají.

2. Seznámit se s fotografií, včetně metadat prostředím online nástroje.

Pro tento účel slouží nástroj <https://www.verexif.com/en>

Ten umožňuje jednak zobrazit důležitá metadata z fotografie, ale také je kompletně odstranit. Nástroj také kromě metadat zobrazuje náhled fotografie (vpravo nahoře) a (Google) mapu, na které je znázorněna lokalita, kde byla fotografie pořízena.

V této fázi žáky vyzvete, ať Vám řeknou, kde a kdy byla fotografie pořízena.

EXIF DATA	
Camera make :	motorola
Camera model :	moto g(7) plus
Date/Time :	2021/07/22 13:23:05
Resolution :	4608 x 3456
Orientation :	rotate 90
Flash used :	No
Focal length :	4.3mm
Exposure time :	0.020 s (1/50)
Aperture :	f/1.7
ISO equiv. :	139
Whitebalance :	Auto
Metering Mode :	center weight
GPS Latitude :	N 48° 58' 39.0503"
GPS Longitude :	E 14° 27' 0.5975"
GPS Altitude :	425.22m
JPEG Quality :	95

Obrázek 4: EXIF Data z VerEXIF

3. Použit stejný online nástroj k editaci (kompletnímu vymazání) metadat.

Pro tento účel slouží tlačítko „Remove Exif“

Remove Exif

4. Nahrát fotografie na USB flashku poskytnutou vyučujícím.

Aplikace poté nabídne takto pozměněnou fotografii uložit znovu pod názvem „foto_no_exif.jpg“. Požádejte žáky ať ji nahrají na zapůjčenou USB flashku a vrátí Vám ji.

Součástí tohoto pracovního listu je také diskuse ve třídě na téma, jaké údaje z metadat fotografií mohou vést k páchání trestné činnosti, proč je záhodno tyto informace před publikováním online vymazat/editovat a k čemu by potenciálně mohly tyto informace v rukou neoprávněné osoby vést.

Kyberšikana

Tato aktivita je rozdělena na dvě části:

- *Před začátkem se ujistěte, že je funkční audio výstup na počítačích, že studenti mají funkční sluchátka.*
- Účelem první části je nejprve objasnit pojem kyberšikana a to takovým způsobem, aby se žákům dostal pod kůži a věděli přesně, co reprezentuje, a že se jedná o společenský problém chování v online světě.

Za pomoci instruktážního videa budou žáci písemně odpovídat na zadané otázky z videa a až budou mít hotovo, je nutné s nimi vést na toto téma diskusi (její délka, intenzita a míra detailu je plně ve Vaší kompetenci).

Účelem druhé části je žákům na reálném příkladu demonstrovat ničivé důsledky kyberšikany. Na videu také uvidí, že lidé se v online světě chovají jinak než ve světě skutečném, kde si začnou uvědomovat důsledky svého chování teprve poté, když jsou konfrontováni s realitou.

Stejně jako v předchozí části, budou žáci písemně odpovídat na zadané otázky z videa a až budou mít hotovo je (zde extrémně) nutné s nimi vést na toto téma diskusi (její délka, intenzita a míra detailu je plně ve Vaší kompetenci).

Doporučené pomůcky

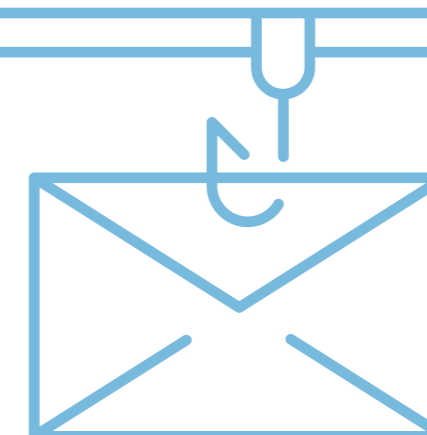
- Každý žák svůj počítač s (rychlým) připojením k internetu umožňujícím paralelně streamovat videa.
- Na PC s operačním systémem Windows bude funkční prohlížeč internetových stránek.
- Sluchátka a správně nastavený zvukový výstup.
- 20 ks USB flashek.

Pro správné zvládnutí úkolů je vhodné mít alespoň základní znalost anglického jazyka, protože pro pracovní list „Fotografie není pouhý obrázek“ jsou webové stránky v anglickém jazyce. I přesto, že je vše v pracovním listu objasněno, lze tímto předejít případným dotazům. Vysvětlete dětem, že znalost anglického jazyka je pro dnešní svět zásadní.

Pracovní listy

Přílohou tohoto materiálu jsou pracovní listy. Tyto pracovní listy jsou k dispozici v editovatelné elektronické formě, aby si je každý učitel mohl upravit, např. dle toho, co má již s žáky probráno.

Pracovní listy jsou pro základní pokrytí problematiky dva. V pracovních listech počítám s tím, že škola má k dispozici pomůcky výše uvedené.



Pracovní list 1

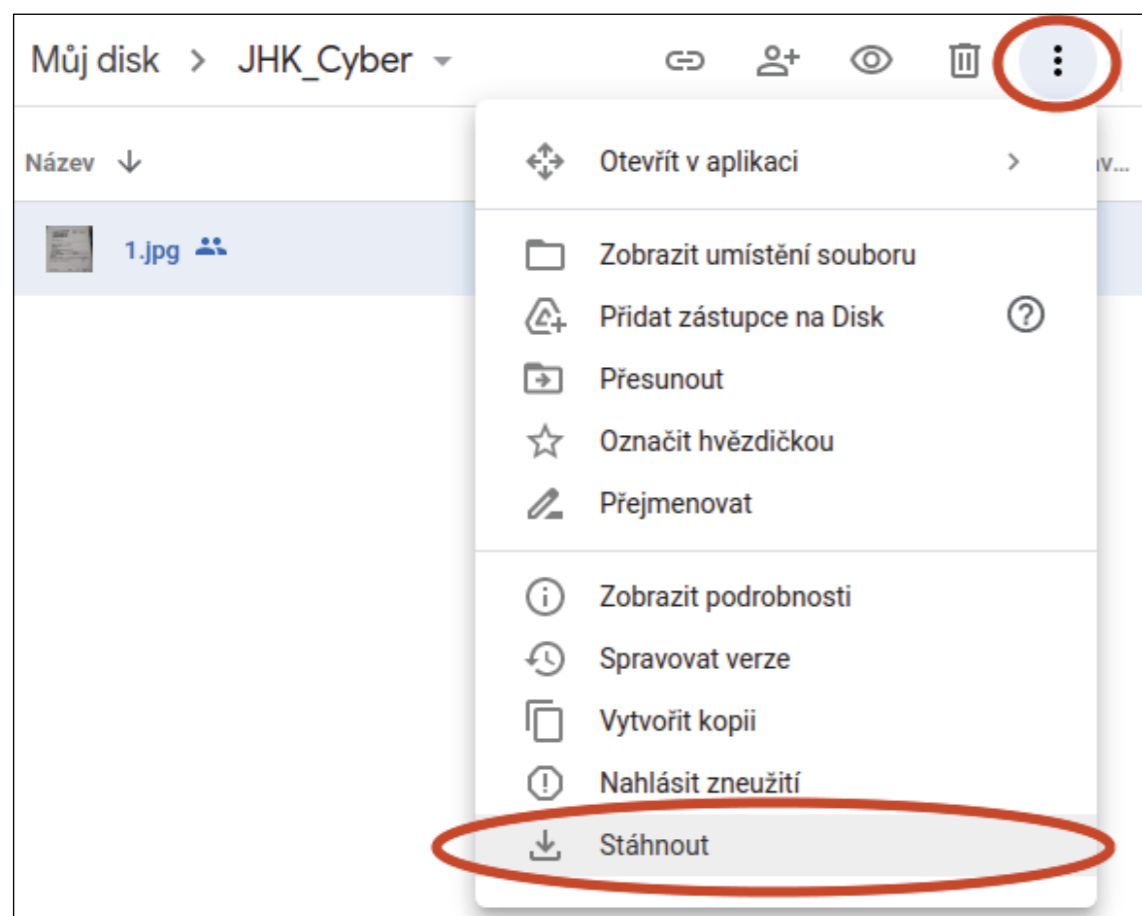
Fotografie není pouhý obrázek

Co budeme potřebovat

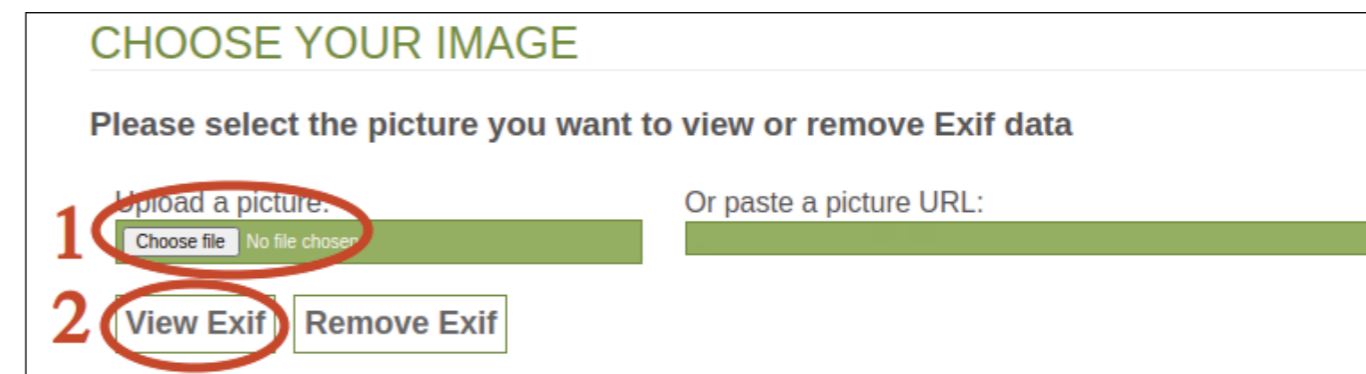
- Počítač s přístupem na internet a prohlížečem webových stránek.
- Základní znalost anglického jazyka.
- USB flashku.

A jdeme na to

1. Na nastartované počítači otevřete prohlížeč webových stránek a zkopírujte následující adresu do adresního řádku (případně pokud máte tento pracovní list jako soubor na počítači ve formátu PDF, přímo na odkaz níže klikněte):
<https://bit.ly/3iIDnQf>
2. Zobrazí se seznam (souborů) fotografií pojmenovaných „1.jpg“ – „12.jpg“. Vyberte a označte si libovolnou fotografii a stáhněte ji na USB flashku dodanou vyučujícím (soubor nepřejmenovávejte).



3. Otevřete si nové okno v prohlížeči (nebo jej spusťte znovu, pokud jste prohlížeč zavřeli) a do adresního řádku napište následující webovou adresu (nebo jako v předchozím případě - pokud máte tento pracovní list jako soubor na počítači ve formátu PDF přímo na odkaz níže klikněte)
<https://www.verexif.com/en>
4. Vyberte soubor nahraný na USB flashce a klikněte na tlačítko **View Exif**



5. Zobrazí se Vám kromě náhledu nahrané fotografie také tzv. EXIF data, což jsou informace o fotografii (tzv. Metadata = data o datech).
 - a. Vhodným způsobem sdělte učiteli, jaká z těchto dat se podle vás dají využít k páčání trestné činnosti a proč. Zaměřte se na mapu.
6. Nyní přistoupíme k výmazu EXIF informací. Z fotografie se tak stane pouhý obrázek, bez dodatečných informací. Klikněte na tlačítko **Remove Exif** a soubor uložte opět na USB flashku.
7. Přesvědčte se, že nově nahraný soubor (obrázek) již neobsahuje žádné dodatečné informace (to můžete udělat více způsoby – volba je vás).
8. USB flashku s oběma soubory odevzdejte učiteli.

Vyzkoušejte si výše uvedené na některé z vašich fotografií – např. z vašeho mobilního telefonu a to tak, že přímo z mobilního telefonu půjdete na výše uvedenou stránku **<https://www.verexif.com/en>** a nahrajete vámi vybranou fotku (nemusíte mít strach, webová stránka fotografie neukládá). Přesvědčte se, že pozice na mapě opravdu souhlasí s tím, kde byla fotografie původně pořízena.

Na závěr se se svým vyučujícím pobavte, jaká metadata mohou podle Vás být potenciálně nebezpečná a mohou sloužit k páčání například Kyberšikany.

Co jste se naučili

Že fotografie není pouhý obrázek. Co všechno dodatečné informace (metadata, EXIF data) obsahují a jak tyto informace vymazat před publikováním fotografie na internet či sociální sítě.

Pracovní list 2

Kyberšikana

Co budeme potřebovat

- Počítač s přístupem na internet a prohlížečem webových stránek.
- Funkční audio výstup vyvedený na sluchátka.
- USB flashku.

A jdeme na to

První část

1. Na nastartovaném počítači otevřete prohlížeč webových stránek a zkopírujte následující adresu do adresního řádku (případně pokud máte tento pracovní list jako soubor na počítači ve formátu PDF, přímo na odkaz níže klikněte):
<https://bit.ly/3ijEy1L>
2. Nasadte si sluchátka a podívejte se na video.
 - a. Ujistěte se, že ve sluchátkách slyšíte zvuk. Pokud ne a nevíte si rady, zavolejte na pomoc učitele.
3. Zodpovězte následující otázky vztahující se k tomu, co jste právě viděli. Nebojte se video si pustit opakovaně.

Jak byste vlastními slovy definovali pojem „kyberšikana“?

Jak se liší od šikany klasické a proč může být šikana v online prostoru ještě horší?

Pokud se k již probíhající kyberšikaně „nachomýtnete“, jak je správné se zachovat? Měli byste se k ní přidat?

Pokud jste obětí kyberšikany, jak byste měli postupovat, aby Vám někdo druhý uvěřil a měli jste proti agresorům důkazy? Co je pravidlo „tři N“?

Je správné kyberšikanu nahlásit? Co je Sexting?

Druhá část

4. Znovu otevřete prohlížeč webových stránek a zkopírujte následující adresu do adresního řádku (případně pokud máte tento pracovní list jako soubor na počítači ve formátu PDF, přímo na odkaz níže klikněte):
<https://bit.ly/3yNkiC1>
5. Nasadte si sluchátka a podívejte se na video.
Nezapomeňte kliknout na ikonu u videa :
🔊 Zapnout zvuk
 - a. Ujistěte se, že ve sluchátkách slyšíte zvuk. Pokud ne a nevíte si rady, zavolejte na pomoc učitele.
6. Zodpovězte následující otázky vztahující se k tomu, co jste právě viděli. Nebojte se video si pustit opakovaně.

Co udělala Simona a Kiana podle tebe špatně? Kde nastala zásadní chyba?

.....

.....

.....

.....

Jak se zachovali jejich spolužáci? Jaký byl rozdíl v jejich chování v online prostoru (na sociálních sítích) a ve skutečnosti?

.....

.....

.....

.....

K čemu u Kiany vedlo chování jejích spolužáků v online světě? Jakou pomoc musela vyhledat?

.....

.....

.....

.....

Na Facebooku si vždy nastavte, aby pokud Vás někdo označí na fotografii, jste tuto akci museli schválit. Pokud ji neschválíte, fotografie se neobjeví na Vašem profilu.

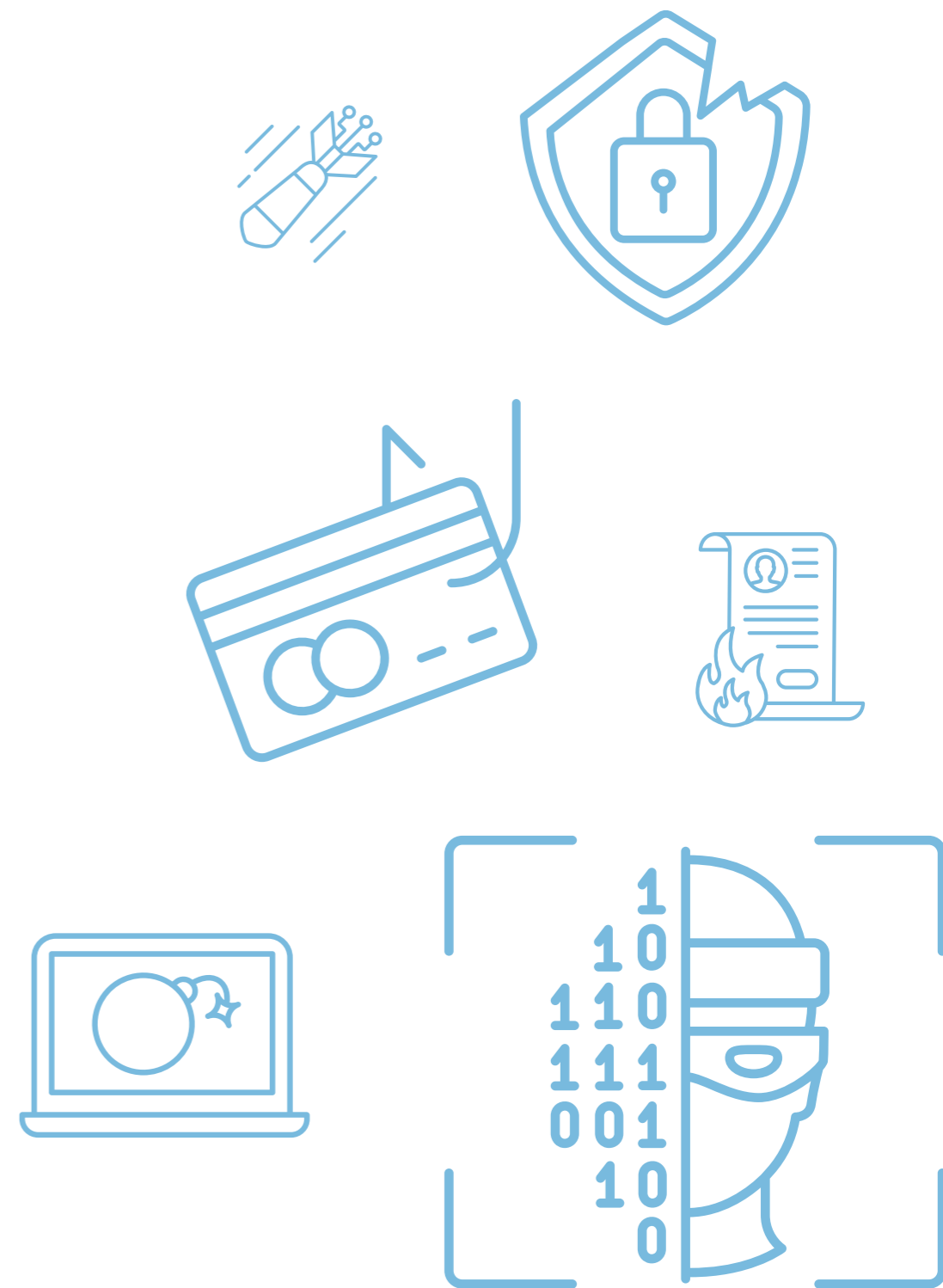
<https://www.facebook.com/help/226296694047060>

Dále si povolte, kdo může co přidávat do Vašich příspěvků, a že takováto označení vždy podléhají Vašemu schválení

<https://www.facebook.com/help/247746261926036>

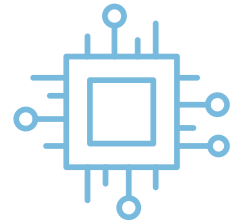
Co jste se naučili

Definovat pojem kyberšikana a především mu porozumět. Základy chování na sociálních sítích a etiku při publikování a sdílení fotografií. Důsledky kyberšikany a nepřirozené chování dětí v online prostoru.



**METODIKA ZÁKLADY
PROGRAMOVÁNÍ**

BLOKOVÉ PROGRAMOVÁNÍ PRO ZŠ



Cílem workshopu Metodika Základy programování/blokové programování je naučit posluchače "programátorsky" myslet. Začínáme základy algoritmizace, formulací problému, vytvořením algoritmu, nakreslením vývojového diagramu a konečně sestavením a odladěním programu. V každé části jsou vždy části teoretická a praktická, kde jsou jak příklady vyřešené učitelem, tak příklady pro posluchače k procvičení. Nakonec je zařazen vždy vzorový příklad, který by posluchači měli vyřešit samostatně nebo ve dvojici. Programovací jazyk zde není podstatný, pro svoji názornost a jednoduchost byl v praktických příkladech zvolen jazyk Python v.3.

Základní instrukce

Tento kurz je doporučován pro žáky základních škol. K tomuto dokumentu je připraven materiál v prostředí Jupyterhub a/nebo Jupyter Notebook, kde si jednotlivé úlohy budou moci posluchači vyzkoušet. Toto prostředí bude posluchačům k dispozici odkudkoliv s internetovým připojením. Pokud budou chtít posluchači použít vlastní počítač, předpokládá učitel předinstalovaný základní programovací jazyk Python ve verzi 3.3 nebo vyšší.

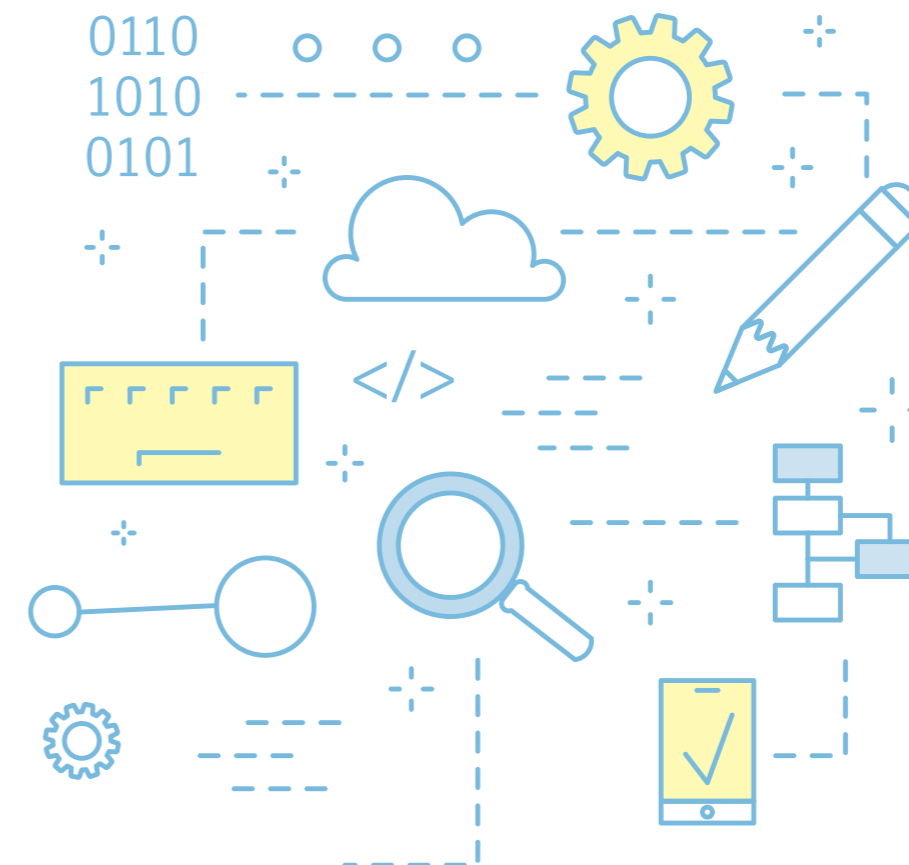
Znalost základů programování jakéhokoliv programovacího jazyka je žádoucí, nikoliv však nutná.

Časová dotace tohoto kurzu nemůže být zcela jednoznačná. Závisí na tom, zda se studenti již seznámili s programováním a zda již mají obecně nějaké zkušenosti s programováním, případně s Pythonem. Učitel by měl nejprve prostudovat všechny materiály dané k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.

Metodika používá jako zdroj informací

- Programátorskou cvičebnici Algoritmy v příkladech od Radka Pelánka
- Programátorské kuchařky MatfyzPress 2011
- Výuka algoritmizace na ZŠ – Bc. Jana Bromová
- <http://www.ivt.mzf.cz/algoritmizace-a-programovani/uvod-do-algoritmu>

Základní znalost programování není nutná, ale je žádoucí. Programovací jazyk není určen, v podstatě je možné použít jakýkoliv, v příkladech této metodiky byl použit jazyk Python pro jeho názornost.



Blokové programování

Blokové programování vychází z blokové struktury programu. To je v IT označení pro zdrojový kód programu rozčleněný do samostatných bloků, které ho rozdělují na souvislé logické části nebo samostatné funkční části u funkcí a procedur. Blokovaná struktura se hojně využívá hlavně v moderních technikách programování, které se označují také jako strukturované jazyky.

Poznámky

- Ačkoliv je v příkladech použit pro názornost jazyk Python, není toto výuka Pythonu.
- V učebnici, která byla zvolena jako základ najdete i obtížné kategorie, které v této metodice zásadně neuvádím.
- Pro žáky volte takové příklady, které složitostí odpovídají jejich možnostem a znalostem a které stihnou s přehledem udělat v určeném čase.

Začínáme

Kategorie obtížnosti

Kategorie obtížnosti jsou subjektivní.

V roli experta jste vůči žákům vy, vždy uvažujte s tímto poměrem

Nápad	Začátečník	Expert	Kódování	Začátečník	Expert
1	Vcelku jasné	Zřejmé	1	20 minut	5 minut
2	Trocha rozmýšlení	Rutina	2	1 hodina	15 minut
3	Těžké	Trocha rozmýšlení	3	Půl dne	1 hodina

*„Mysl není nádoba, kterou je potřeba naplnit,
ale oheň, který je potřeba vznítit.“*

Plutarchos

Algoritmy

Základní pojmy

Algoritmizace

Algoritmizace je přesný postup, který se používá při tvorbě programu pro počítač, jehož prostřednictvím lze řešit nějaký konkrétní problém.

Algoritmizaci lze rozdělit do několika kroků:

- Formulace problému
- Analýza úlohy
- Vytvoření algoritmu
- Sestavení programu
- Odladění programu

Formulace problému

V této etapě je třeba přesně formulovat požadavky, určit výchozí hodnoty, požadované výsledky, jejich formu a přesnost řešení. Tvůrce algoritmu musí dokonale rozumět řešenému problému, jinak nemůže algoritmus sestavit – v praxi programátoři spolupracují s odborníky z oblastí, pro které mají vytvořit algoritmus.

Analýza úlohy

Při analýze úlohy si ověříme, zda je úloha řešitelná a uděláme první návrh řešení. Definujeme způsob vstupu a tvar vstupních hodnot. Zjistíme, zda je úloha řešitelná a za jakých podmínek a zda má případně více řešení. Pokud má úloha více postupů, jak ji řešit (což je u složitějších úloh běžné), zvolíme jedno řešení, případně si zdůvodníme, proč právě toto řešení bylo zvoleno. Dále pak již postupujeme podle něj.

Vytvoření algoritmu úlohy

Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy. Pojem algoritmu se nejčastěji objevuje při programování, kdy se jím myslí teoretický princip řešení problému (oproti přesnému zápisu v konkrétním programovacím jazyce). Obecně se ale algoritmus může objevit v jakémkoli jiném vědeckém odvětví.

Vývojový diagram (flowchart)

Vývojový diagram je druh diagramu, který slouží ke grafickému znázornění jednotlivých kroků algoritmu, pracovního postupu nebo nějakého procesu. Vývojový diagram obsahuje obrazce různého tvaru (obdélníky, kosočtverce, aj.), navzájem propojené pomocí šipek. Obrazce reprezentují jednotlivé kroky, šipky tok řízení. Vývojové diagramy standardně nezobrazují tok dat, ten je zobrazován pomocí data flow diagramů. Vývojové diagramy jsou často využívány v informatice během programování pro analýzu, návrh, dokumentaci nebo řízení procesu.

Tvorbě vývojových diagramů bude věnována jedna z dalších kapitolek.

Sestavení programu

Na základě algoritmu řešené úlohy sestavíme program (zdrojový text) v konkrétním programovacím jazyce. Ze zdrojového textu se pomocí překladače do strojového kódu vytvoří spustitelný program (případně interpretem se přeloží a spustí jednotlivé příkazy programu). Lze říci, že dobře provedená analýza úlohy a algoritmizace je velmi důležitá pro řešení daného problému a je základním předpokladem sestavení programu pro počítač.

Tvorbě programů v jazyce Python bude věnována jedna z dalších kapitol.

Pro naši vzorovou ukázkou **Ciferace** se použije tzv. rekursivní volání funkce, proto si jej znázorníme již nyní

Rekursivní volání funkce

Není to nic složitého, funkce prostě zavolá sama sebe. Ciferace je krásná ukáзка rekursivní funkce. Podmínkou je, že součet všech čísel v čísle musí být jednočíslný. Pokud není, dosadíme za číslo součet a zavoláme funkci znovu.

Odladění programu

Naprogramováním činnost nekončí. Program musí být tzv. "odblbený". Odladěním chceme odstranit chyby z programu. Programátor musí spolu s testerem odchytil a ošetřit všechny možné, i velmi nepravděpodobné možnosti. Celá kapitola v programování je tzv. error handling tj. ošetřování možných chyb. Příkladem ošetření možné chyby je například zabránění dělení nulou, které by skončilo chybou.

Další odladování se týká např. toho, aby program běžel co nejrychleji a aby neměl velké nároky na paměť.

Nejčastější chyby jsou chyby v zápise, tzv. syntaktické – ty odhalí překladač a dělají je i zkušení programátoři. Horší jsou logické chyby, které vyplývají z nesprávně navrženého algoritmu – projeví se nesprávnou činností programu nebo špatnými výsledky – při odstraňování těchto chyb může pomoci ladící program (debugger) umožňující sledování aktuálního stavu proměnných a krokování. Teprve po odstranění všech druhů chyb můžeme program použít k praktickému řešení úloh. Důležité je následné testování. V týmu řešitelů jsou tzv. testeři.

V případě složitějších nebo komerčních úloh je tester role, která je oddělena od vývojáře, aby nevznikala profesní slepota. Testování a metodika testování je opět celá věda.

Algoritmizace v praxi

Proces psaní příkazů v programovacím jazyce se nazývá **programování**.

- **Fáze analýzy** – Co bude program umět definuje **analytik**, ten pak dává zadání pro **programátora (vývojáře)**.
- **Fáze programování** – Vývojář podle popisu vytvoří program (přepíše řešení do programovacího jazyka). Zápis programu v programovacím jazyce se nazývá **zdrojový kód**. Poté je spuštěn program, který dokáže přeložit tento zdrojový kód do jazyka počítače, a vznikne tak **spustitelný program**. První otestování programu se nazývá jeho **ladění (debugging)**.
- **Fáze testování** – Tento program dostane **tester**, který se snaží najít v programu chyby nebo nesprávné chování neodpovídající zadání. Pokud objeví chybu, pak ji předá zpět programátorovi k dořešení.
- **Fáze akceptace**, zaškolení obsluhy programu.
- **Fáze změn, údržba a podpora** – Program je předán uživatelům. Dochází k požadavkům na úpravy nebo přidání funkcí.

UML

Unified Modeling Language (UML) je grafický modelovací jazyk pro specifikaci, vizualizaci a dokumentaci informačních systémů a aplikací. UML umožňuje popsat systém pomocí slov a obrázků. Můžeme jím modelovat různé systémy.

Výčet diagramů standardu UML 2.0:

Strukturní diagramy:

- Diagram tříd (Class Diagram)
- Diagram objektů (Object Diagram)
- Diagram komponent (Component Diagram)
- Diagram složených struktur (Composite Structure Diagram)
- Diagram nasazení (Deployment Diagram)
- Diagram balíčků (Package Diagram)

Diagramy chování:

- Diagram případů užití (Use Case Diagram)
- Diagram aktivit (Activity Diagram)
- Stavový diagram (State Machine Diagram)

Diagramy interakce:

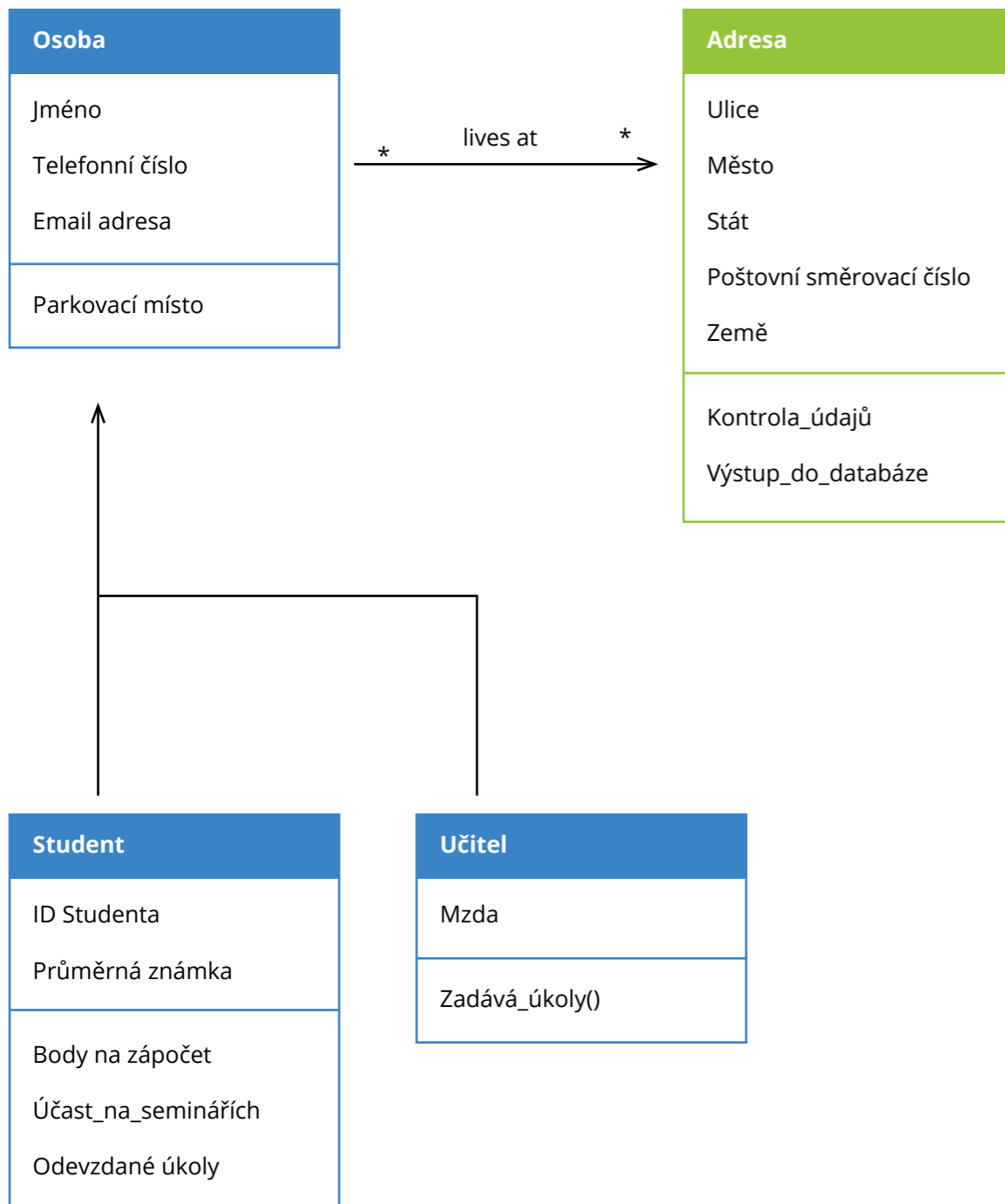
- Sekvenční diagram (Sequence Diagram)
- Diagram komunikace (Communication Diagram)
- Diagram přehledu interakcí (Interaction Overview Diagram)
- Diagram časování (Timing Diagram)

Často slyšíme pojem **diagramy tříd** (class diagrams) nebo **use-case diagramy**. UML je jazyk, který byl standardizován Object Management Group (OMG), mezinárodní asociací pro tzv. "open standards for object-oriented applications (<http://www.omg.org>)."

Základy jazyka UML a schopnost se jazykem UML vyjádřit se naučí prakticky každý. Vyjadřováním prostřednictvím UML máme jistotu, že nám ostatní porozumí.

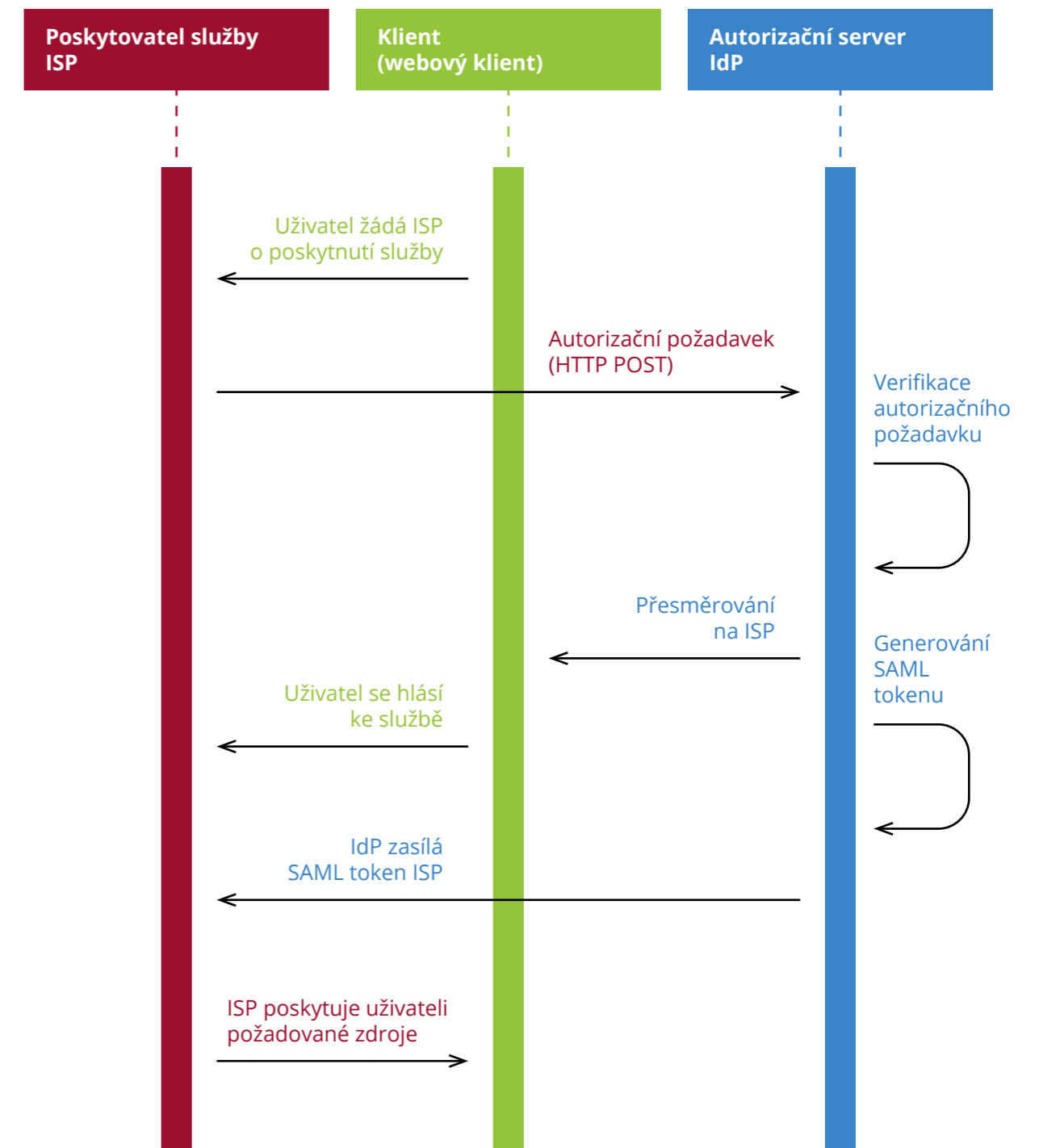
Aktuálně se používá UML verze 2.0, ("OMG Unified Modeling Language: Superstructure, Version 2.0, Revised Final Adopted Specification, October 2004", from <http://www.omg.org>).

Class diagram



UML Class diagram

Sekvenční diagram

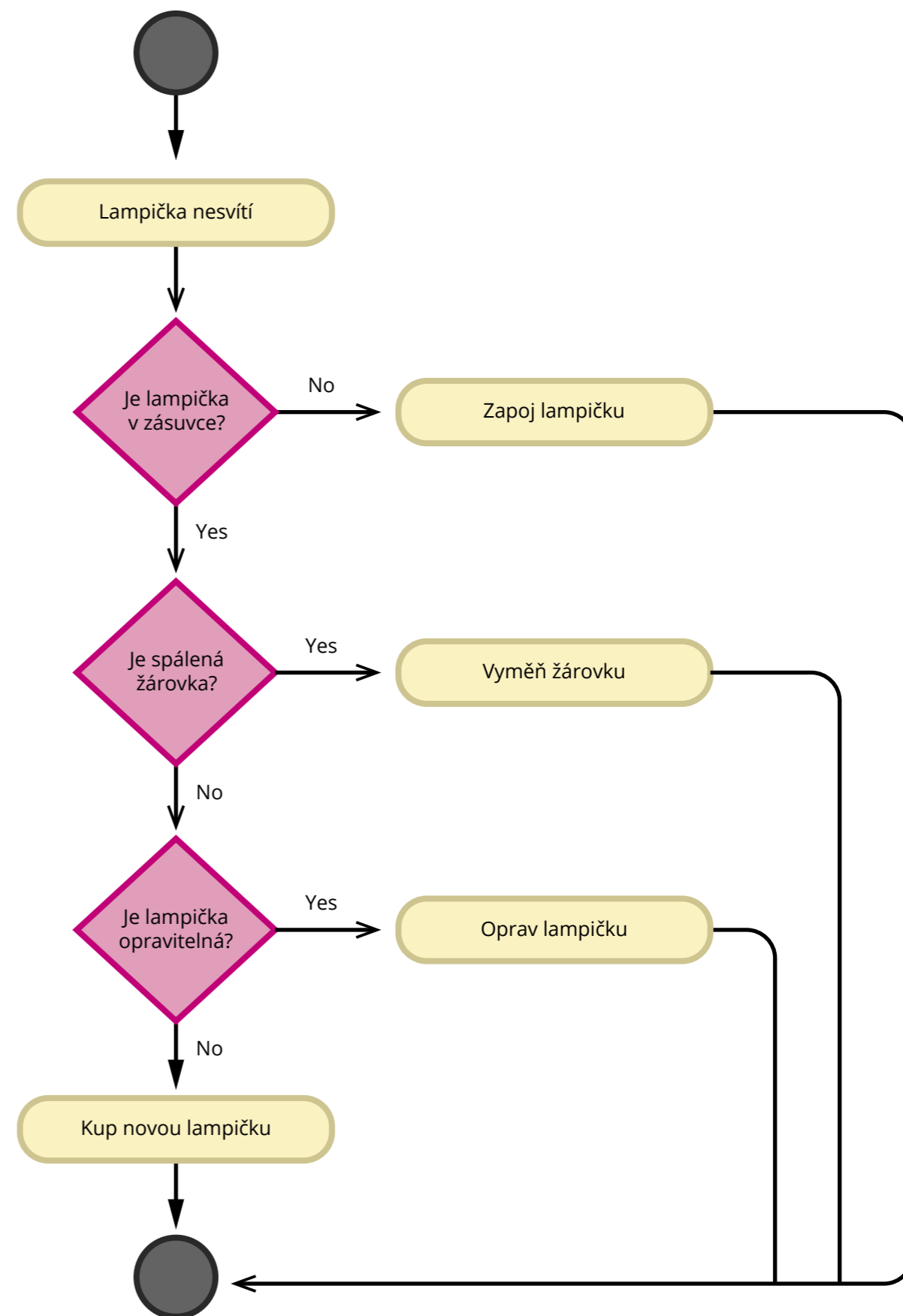
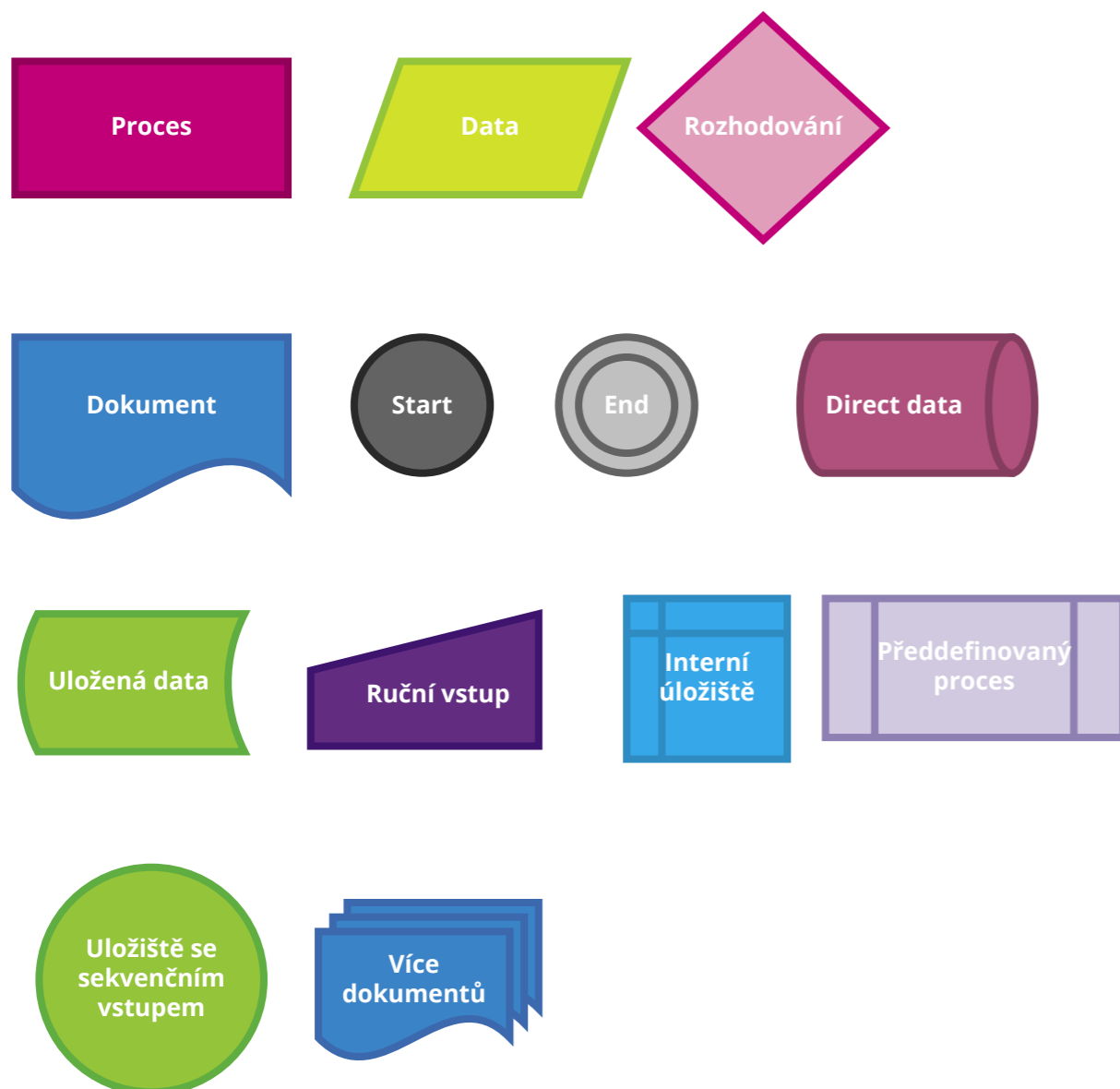


UML Sekvenční diagram - SAML protokol

Flowchart diagram

Protože flowchart diagram (neboli vývojový diagram) se přímo týká programování, budeme u něj o něco podrobnější. Nejužívanější symboly vývojového diagramu jsou na následujícím obrázku.

Více zde



Příklad vývojového diagramu je na následujícím obrázku. Zkusíme si však i některé vlastní.

Algoritmus

Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy. Příkladem je kuchařský recept.

Vlastnosti algoritmů

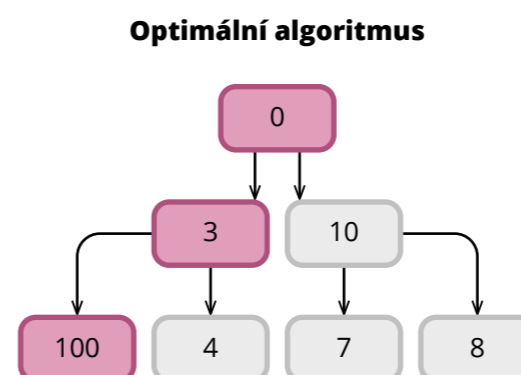
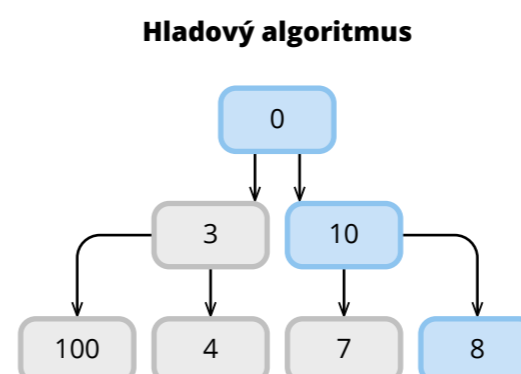
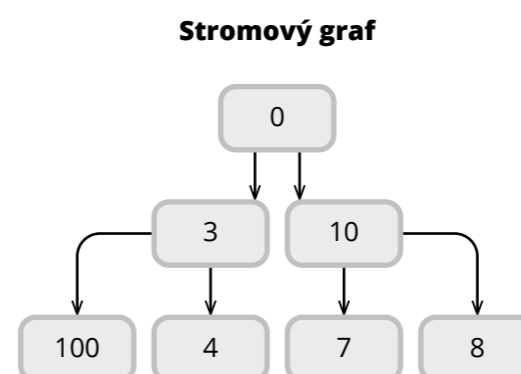
- **Elementárnost** – Algoritmus se skládá z konečného počtu jednoduchých (elementárních) kroků.
- **Konečnost** – Každý algoritmus musí skončit v konečném počtu kroků.
- **Obecnost** – Algoritmus neřeší jeden konkrétní problém (2/3), ale obecnou třídu obdobných problémů (a/b).
- **Determinovanost** – Algoritmus je determinovaný, pokud za stejných podmínek (pro stejné vstupy) poskytuje stejný výstup. Tohle je problematické - existují např. pravděpodobnostní algoritmy, kde do hry vstupuje náhoda.
- **Determinismus** – Každý krok algoritmu musí být jednoznačně a přesně definován; v každé situaci musí být naprosto zřejmé, co a jak se má provést, jak má provádění algoritmu pokračovat. Vyjádření výpočetní metody v programovacím jazyce se nazývá program.
- **Výstup** – Algoritmus vede od zpracování hodnot k výstupu.

Metody návrhu algoritmů

Algoritmů je samozřejmě mnoho. V učebnicích se dočteme například o těchto:

- Třídění
- Hladový algoritmus
- Halda
- Grafy
- Dijkstrův algoritmus
- Minimální kostra
- Rozděl a panuj
- Dynamické programování
- Vyhledávací stromy
- Hešování
- Řetězce a vyhledávání v textu
- Rovinné grafy
- Eulerovské tahy
- Toky v sítích
- Intervalové stromy

Pro praktickou ukázkou si vybereme zjednodušený hladový algoritmus



Hladový algoritmus

Hladový algoritmus (anglicky greedy search) Takovými algoritmy rozumíme ty, které hledají řešení celé úlohy po jednotlivých krocích a splňují následující dvě podmínky:

- V každém kroku zvolí lokálně nejlepší řešení.
- Provedené rozhodnutí již nikdy neodvolává (tedy „nebacktrackuje“).

Lokálně nejlepší řešení je takové, které v aktuálním kroku vybere tu možnost, která nám na tomto místě nejvíce pomůže (bez jakéhokoliv ohledu na globální stav). Může to být třeba nejvyšší hodnota, nebo nejkratší cesta v grafu.

Pokud ale od hladového algoritmu chceme, aby nám našel **globálně nejlepší řešení**, musí naše úloha splnit předpoklad, že si výběrem lokálně nejlepšího řešení nezhoršíme to globální. Tento předpoklad se nedá formulovat obecně a je nutné se nad ním zamyslet zvlášť u každé úlohy.

Hladový algoritmus si představíme v Praktické části.

Základy Pythonu

Python je pravděpodobně nejsnazší současný programovací jazyk, s nímž se dobře pracuje, má obrovské množství knihoven téměř na cokoliv a velkou základnu, která vždy ráda pomůže. Kód v Pythonu se dobře píše i čte, proto jsem ho pro názorné ukázky algoritmizace a programování zvolila i já.

Python je tzv. "cross-platform", což znamená, že běží na různých operačních systémech. Není potřeba nic kompilovat, což si za daň bere, že je o poznání pomalejší než např. jazyk C.

Více zde



Budeme používat Python verze 3.3 a vyšší. Jak to zjistíme?

Prostě se optáme operačního systému, jaký Python je na něm nainstalován.

Jupyter Notebook

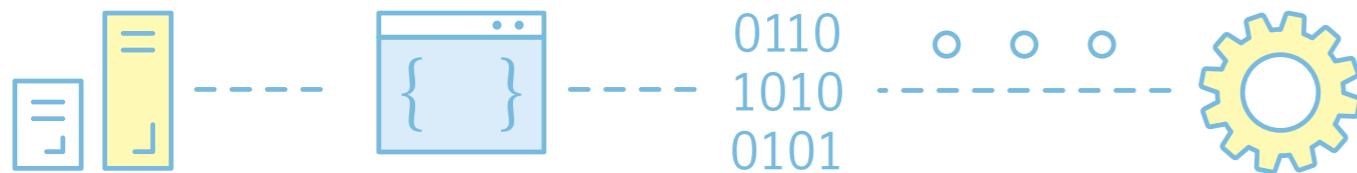
Příklady zde jsou uváděné v tzv. Jupyteru.

Poznámka: Pokud bude učitel ve výuce používat **Jupyterhub** nebo **Jupyter Notebook**, měl by jej v kurzu představit. V tomto materiálu však, vzhledem k rozsahu, uveden není.

```
!python -V
Python 3.9.1
```

Pokud nemáte Python nainstalován, zde je postup pro Linux Ubuntu.

```
sudo apt-get install python3
sudo apt-get install idle3
sudo apt-get install python3-pip
```



Jednoduché datové typy a vestavěné funkce v Pythonu

Vestavěné funkce

Python obsahuje řadu vestavěných funkcí, které nám mohou usnadnit život. Vidíte je v tabulce abecedně seřazené. **Podrobnější popis naleznete v QR kódu.**



abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	import()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

Jaké funkce a metody můžeme od třídy chtít, se dozvíme pomocí klíčového slova `dir()`. Zapamatujte si ho.

```
dir(„Marta“)

['capitalize',
 'casefold',
 'center',
 'count',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'format_map',
 'index',
 'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
 'lstrip',
 'maketrans',
 'partition',
 'removeprefix',
 'removesuffix',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

```
seasons = ['Spring', 'Summer', 'Fall', 'Winter']
dict(enumerate(seasons))
{0: 'Spring', 1: 'Summer', 2: 'Fall', 3: 'Winter'}compass.get_field_strength()
- intenzita magnetického pole
compass.heading() - azimut ve stupních, nutná inicializace pomocí compass.calibrate()
```

Základní datové typy

Každý programovací jazyk umí prezentovat položky dat. Python poskytuje více předdefinovaných datových typů, ale prozatím se budeme zabývat pouze několika z nich. Python představuje celá čísla (kladná a záporná celá čísla) pomocí typu `int` a řetězce (sekvence znaků Unicode) pomocí typu `str`. Zde je několik příkladů celých čísel a řetězců:

```
type(-973)
int
type(210624583337114377340864637790190801098222508621955072)
int
```

```
type('Tomáš Garrigue Masaryk')
str
type('αβγ ÷© + ■ ¶☉☄$')
str
x='αβγ ÷© + ■ ¶☉☄$'
len(x)
15
```

Příklad použití klíčového slova `dir` nám řekne, co můžeme od datového typu nebo funkce požadovat.

```
dir(x)
[...]
'capitalize',
'casefold',
'center',
'count',
'encode',
'endswith',
'expandtabs',
'find',
... ]
x.count(' ')
4
```

V textu jsou 4 mezery

Základní datové typy v Pythonu

Typ	Popis	Příklady hodnot
int	celá čísla	1, 42, -5, 200
float	reálná čísla (přesněji čísla v plovoucí desetinné čárce, přičemž Python používá desetinnou tečku, nikoliv čárku)	2.5, 3.25, -12.37832
bool	pravdivostní hodnoty	True, False
str	řetězce	"prase", "pes"

Základní datové typy Pythonu najdeme **v QR kódu**.



Složené datové typy

V Pythonu existuje pět základních složených datových struktur:

- bytearray
- bytes
- list
- str
- tuple

List

V Pythonu je list hlavní datovou strukturou používanou k ukládání sekvence prvků. Sekvence datových prvků uložených v seznamu nemusí být stejného typu.

Chcete-li vytvořit list, musí být datové prvky uzavřeny v [] a musí být odděleny čárkou. Následující kód například vytvoří dohromady čtyři datové prvky, které jsou různých typů.

```
muj_list = ["Rumcajs", 33, "Cipisek", True]
muj_list
['Rumcajs', 33, 'Cipisek', True]
type(muj_list)
list
```

List může obsahovat jiné listy i další jednoduché nebo složené datové typy.

```
muj_list = ["Rumcajs", 33, "Cipisek", True, [1, 2, 'Žlutoučký kůň úpěl ďábelské ódy']]
```

Slicing

s(-15)	s(-14)	s(-13)	s(-12)	s(-11)	s(-10)	s(-9)	s(-8)	s(-7)	s(-6)	s(-5)	s(-4)	s(-3)	s(-2)	s(-1)
T	y	r	i	o	n		L	a	n	i	s	t	e	r
s(0)	s(1)	s(2)	s(3)	s(4)	s(5)	s(6)	s(7)	s(8)	s(9)	s(10)	s(11)	s(12)	s(13)	s(14)

```
muj_list=list('Tyrion Lanister')

muj_list[:5] # od začátku 5 znaků
['T', 'y', 'r', 'i', 'o']

muj_list[:-1] # od konce všechny znaky
['r', 'e', 't', 's', 'i', 'n', 'a', 'L', ' ', 'n', 'o', 'i', 'r', 'y', 'T']

muj_list[:-3] # od konce každý třetí znak
['r', 's', 'a', 'n', 'r']

barvy=['Červená', 'Zelená', 'Modrá', 'Žlutá']
barvy[:2]
['Červená', 'Zelená']

barvy[-2]
'Modrá'

barvy[:2]
['Červená', 'Modrá']

barvy[:-2]
['Žlutá', 'Zelená']
```

```
s="I have a cat "
```

```
for i in s:
    print(s)
    s=s.lstrip(i)
```

```
I have a cat
```

```

have a cat
have a cat
ave a cat
ve a cat
e a cat
 a cat
a cat
 cat
cat
at
t

```

Nesting

List podporuje vnořování (nesting), jednotlivé datové typy je možné uvnitř listu vnořovat.

```

a = [1,2,[100,200,300],['Marta',['Rudolf']],6]
max(a[2])
300

a[3][1][0]
'Rudolf'

```



Iterace

```

barvy
['Červená', 'Zelená', 'Modrá', 'Žlutá']

```

```

for barva in barvy:
    print(f'{barva} je v našem seznamu')

```

Červená je v našem seznamu

Zelená je v našem seznamu

Modrá je v našem seznamu

Žlutá je v našem seznamu

Lambda

Lambda je zvláštní typ funkce v Pythonu. Často se používá ve výpočtech ve složených datových typech. V příkladech si ukážeme časté použití lambda funkce.

Filtrování dat

```

list(filter(lambda x: x > 100, [-5, 200, 300, -10, 10, 1000]))
[200, 300, 1000]

```

```

list(filter(lambda x: len(x)>5, barvy))
['Červená', 'Zelená']

```

Transformace dat

Pro transformaci dat využijeme ještě další funkci `map()`.

```

list(map(lambda x: x ** 2, [11, 22, 33, 44,55]))

```

```

[121, 484, 1089, 1936, 3025]

```

```

list(map(lambda x: f'{x} je delší než 5' if len(x)>5 else f'{x} je menší než 5',
barvy))

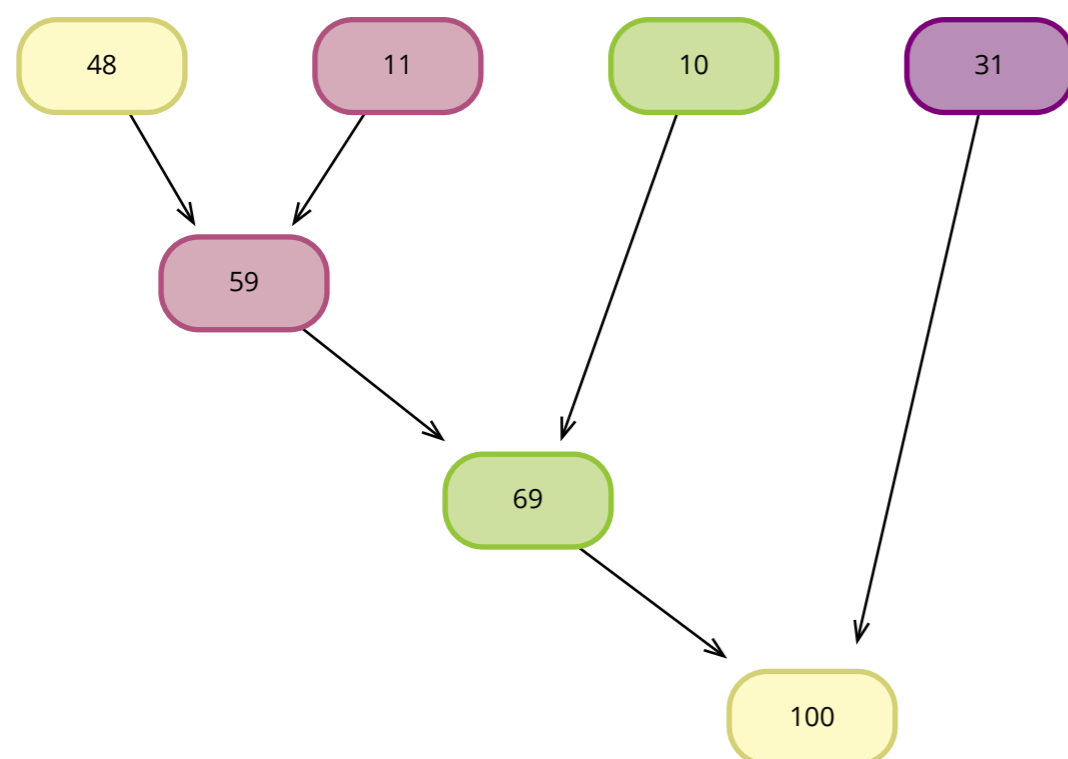
```

```

['Červená je delší než 5',
'Zelená je delší než 5',
'Modrá je menší než 5',
'Žlutá je menší než 5']

```

Reduce - Agregace dat



Lambda - redukce

```

from functools import reduce #pro použití funkce reduce se musí importovat knihovna
functools nebo její část
čísla=[48,11,10,31]
reduce(lambda x, y:x+y, čísla)
100
  
```

List comprehension

Netýká se pouze datové struktury list ale i dalších. Tato konstrukce bývá v Pythonu hojně využívána.

```

[i for i in barvy if len(i)>5]
['Červená', 'Zelená']

import math
čísla=[2,6,8,10,7,5]
[math.sqrt(x) for x in čísla]
[1.4142135623730951,
 2.449489742783178,
 2.8284271247461903,
 3.1622776601683795,
 2.6457513110645907,
 2.23606797749979]
  
```

List metody

Syntax	Description
L.append(x)	Přidává položku na konec listu L
L.count(x)	Vrátí počet výskytů položky x v listu L
L.extend(m)	L += m přidává všechny položky na konec listu L
L.index(x, start, end)	Vrací index pozice hledané položky, pokud je jich víc, tak první zleva; pokud položku nenajde odpoví ValueError exception
L.insert(i, x)	Vkládá položku x do listu L na pozici int i
L.pop()	Vrací hodnotu položky a zároveň odstraňuje výskyt položky v listu L. Pokud je položek více, odstraní první zprava
L.pop(i)	Vrací hodnotu položky a zároveň odstraňuje výskyt položky v listu L na pozici int i v listu L
L.remove(x)	Odstraní výskyt položky x v listu L, pokud položku x nenajde odpoví ValueError exception
L.reverse()	Obrátí pořadí položek v listu L
L.sort(...)	Setřídí list L


```
barvy.append(['Oranžová', "Duhová"])
barvy.extend(['Fialová', "Purpurová"])
```

```
barvy
```

```
['Červená',
 'Zelená',
 'Modrá',
 'Žlutá',
 ['Oranžová', 'Duhová'],
 'Fialová',
 'Purpurová']
```

Range

```
for i in range(2,25,3): # rozsah od 2(včetně) do 25(ne včetně) krok jsou 3
    print(i, end=",")
2,5,8,11,14,17,20,23,
```

Tuples

Datová struktura, která je immutable - tedy pouze pro čtení. Oproti listu má velmi málo možností - count a index. Tuply jde sčítat.

```
t= (1,2,3,'Marta')
```

```
t += ("Vohnoutová",)
```

```
type(t)
```

```
tuple
```

```
dir(t)
```

```
[...
 'count',
 'index']
```

Dictionary (slovník)

Slovník je datová struktura, která je tvořena vždy klíčem (key) a hodnotou (value). Klíč musí být v *dictionary* jedinečný. V Pythonu se dictionary využívá velmi často. Uzavírá se mezi složené závorky {}

```
barvy_semaforu = {
    "Stop": "Červená",
    "Připrav se": "Oranžová",
    "Jed": "Zelená"
}
```

```
barvy_semaforu.get("Stop")
'Červená'
```

```
barvy_semaforu['Stop']
'Červená'
```

```
barvy_semaforu['Barva co není v semaforu'] = 'Růžová' # přidám do slovníku další položku
```

```
barvy_semaforu
{'Stop': 'Červená',
 'Připrav se': 'Oranžová',
 'Jed': 'Zelená',
 'Barva co není v semaforu': 'Růžová'}
```

```
barvy_semaforu.keys()
```

```
dict_keys(['Stop', 'Připrav se', 'Jed', 'Barva co není v semaforu'])
```

```
barvy_semaforu.values()
```

```
dict_values(['Červená', 'Oranžová', 'Zelená', 'Růžová'])
```

```
barvy_semaforu.items()
```

```
dict_items([('Stop', 'Červená'), ('Připrav se', 'Oranžová'), ('Jed', 'Zelená'),
 ('Barva co není v semaforu', 'Růžová')])
```

Set (sada)

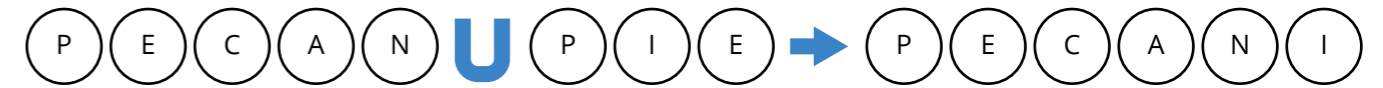
Typ set je datový typ, který podporuje některé množinové operace a je iterovatelný. Stejně jako dictionary se set uvozuje složenými závorkami {}. Set neřeší pořadí položek.

```
zelené = {"lupení", "tráva", "volno na semaforu"}
červené = {"rudá růže", "meloun", "stop na semaforu"}
žluté = {"lupení", "tráva", "pampelišky", "sluníčko"}
```

```
type(zelené)
set

dir(zelené)
[...
'add',
'clear',
'copy',
'difference',
'difference_update',
'discard',
'intersection',
'intersection_update',
'isdisjoint',
'issubset',
'issuperset',
'pop',
'remove',
'symmetric_difference',
'symmetric_difference_update',
'union',
'update']
```

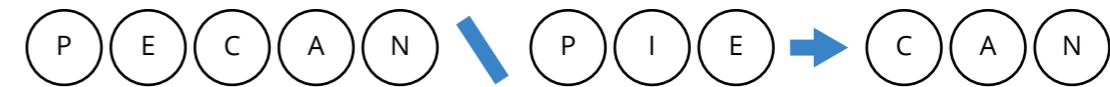
Set - množinové operace



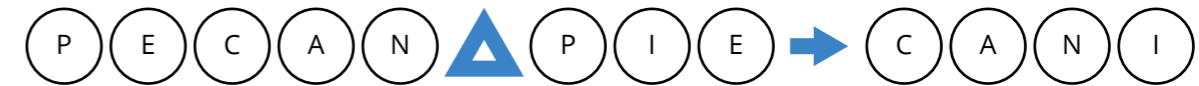
$\text{set}(\text{"PECAN"}) \cup \text{set}(\text{"PIE"}) \rightarrow \{\text{'A'}, \text{'C'}, \text{'E'}, \text{'I'}, \text{'N'}, \text{'P'}\}$ # sjednocení



$\text{set}(\text{"PECAN"}) \cap \text{set}(\text{"PIE"}) \rightarrow \{\text{'P'}, \text{'E'}\}$ # průnik



$\text{set}(\text{"PECAN"}) - \text{set}(\text{"PIE"}) \rightarrow \{\text{'C'}, \text{'A'}, \text{'N'}\}$ # rozdíl



$\text{set}(\text{"PECAN"}) \Delta \text{set}(\text{"PIE"}) \rightarrow \{\text{'C'}, \text{'A'}, \text{'N'}, \text{'I'}\}$ # symetrický rozdíl

```
set("PECAN") | set("PIE") # sjednocení
{'A', 'C', 'E', 'I', 'N', 'P'}
```

```
zelené | žluté
```

```
{'lupení', 'pampelišky', 'sluníčko', 'tráva', 'volno na semaforu'}
```

Aritmetické operace

1. Aritmetické operace
2. Srovnávací operace
3. Přiřazovací operace
4. Logické operace
5. Bitové operace
6. Membership operace
7. Identitní operace

Aritmetické operace

1. sčítání +
2. odčítání -
3. násobení *
4. dělení /
5. umocňování **

```
8**3 # umocňování
512
```

Srovnávací operace

```
a, b=2,6
a <= b, a!= b, a >= b, a > b
(True, True, False, False)
```

Přiřazovací operace

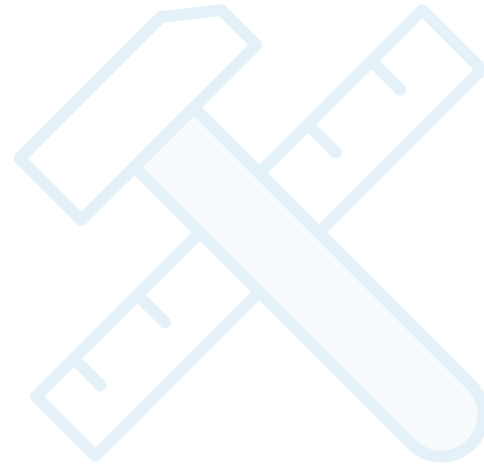
```
a=5
b=a
```

Logické operace

1. and
2. or
3. not

```
a, b=2,6
a <= b and a!= b and a >= b and a > b
```

Více zde



False

Membership operace

```
"Zelená" in barvy
True
```

Identitní operace

```
a=[12.10,7]
b=[12.10,7]
a == b #Srovnávací operace
True
a is b #Identitní operace
False
```

Flow control

Zpracovávají cykly a podmínky.

1. if-elif-else
2. while
3. for

while

```
while True:
    num = input("Vložte celé kladné číslo nebo nulu pro konec: ")
    try:
        num = int(num)
    except ValueError:
        num = -1
    if num < 0:
        print ("Prosím pouze celé kladné číslo.")
        continue
    elif num == 0:
        print ("Končím program..")
        break
    num_cube = (num ** 3)
```

```
print ("Krychle o straně {0} má objem {1}.".format (num, num_cube))
print ("Nashledanou...")
```

Vložte celé kladné číslo nebo nulu pro konec: a

Prosím pouze celé kladné číslo.

Vložte celé kladné číslo nebo nulu pro konec: -5

Prosím pouze celé kladné číslo.

Vložte celé kladné číslo nebo nulu pro konec: 88

Krychle o straně 88 má objem 681472.

Vložte celé kladné číslo nebo nulu pro konec: 0

Končím program..

Konec programu.

break continue pass

```
print("break")
for i in range(0,10):
    if i == 6:
        print('here',end=',')
        break
    print(i, end=',')
print('\n')
print("continue")
for i in range(0,10):
    if i == 6:
        print("here",end=',')
        continue
    print(i, end=',')
print('\n')
```

```
print("pass")
for i in range(0,10):
    if i == 6:
        print("here",end=',')
        pass
    print(i, end=',')
print('\n')
break
0,1,2,3,4,5,here,
```

```
continue
0,1,2,3,4,5,here,7,8,9,
pass
0,1,2,3,4,5,here,6,7,8,9,
```

If-elif-else

```
t=(100000,999,9999)
for lines in t:
    if lines < 1000:
        print(lines," small")
    elif lines < 10000:
        print(lines," medium")
    else:
        print(lines," large")
100000 large
999 small
9999 medium
```

Vytváření a volání funkcí

def jméno_funkce(argumenty):

tělo funkce

```
def vyber_sudá_čísla(seznam_čísel):
    return [i for i in seznam_čísel if i%2 == 0]

print(vyber_sudá_čísla([4,7,88,21,15,4,72,62,112,332,331]))
[4, 88, 4, 72, 62, 112, 332]

def vyber_dělitelná_čísla(seznam_čísel, dělitel):
    return [i for i in seznam_čísel if i%dělitel == 0]

print(vyber_dělitelná_čísla([4,9,88,21,15,4,72,62,112,332,333,331,12],3))
[9, 21, 15, 72, 333, 12]
```


Praktická část

Praktická část předpokládá u každého příkladu postupovat v krocích, co jsme se naučili:

1. Formulace problému
2. Analýza úlohy
3. Vytvoření algoritmu úlohy - vývojový diagram
4. Sestavení programu a jeho odladění

Jako příklad si vezměme jednoduchou úlohu, často používanou ve hře geocaching – ciferaci.

Ciferace - 1.úloha prováděná učitelem

Ciferace - formulace problému

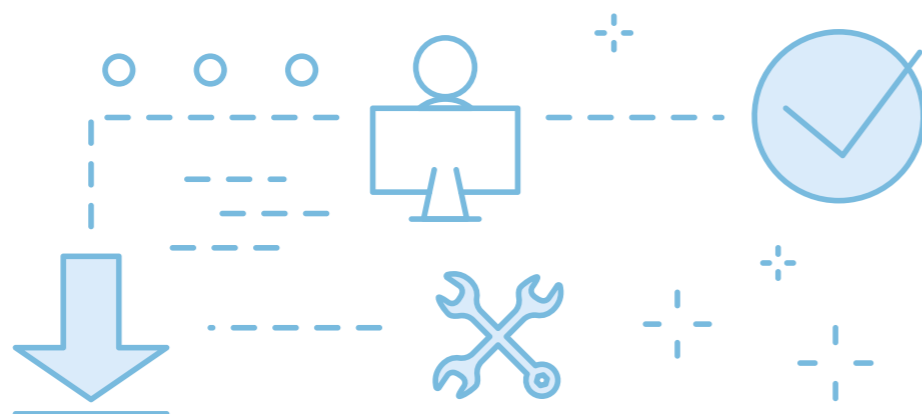
Při ciferaci sečteme všechny číslice z daného čísla. Pokud výsledný součet není jednociferný, součet opakujeme do té doby, až dostaneme jednociferné číslo. Toto jednociferné číslo je výsledkem ciferace.

Ciferace - analýza úlohy

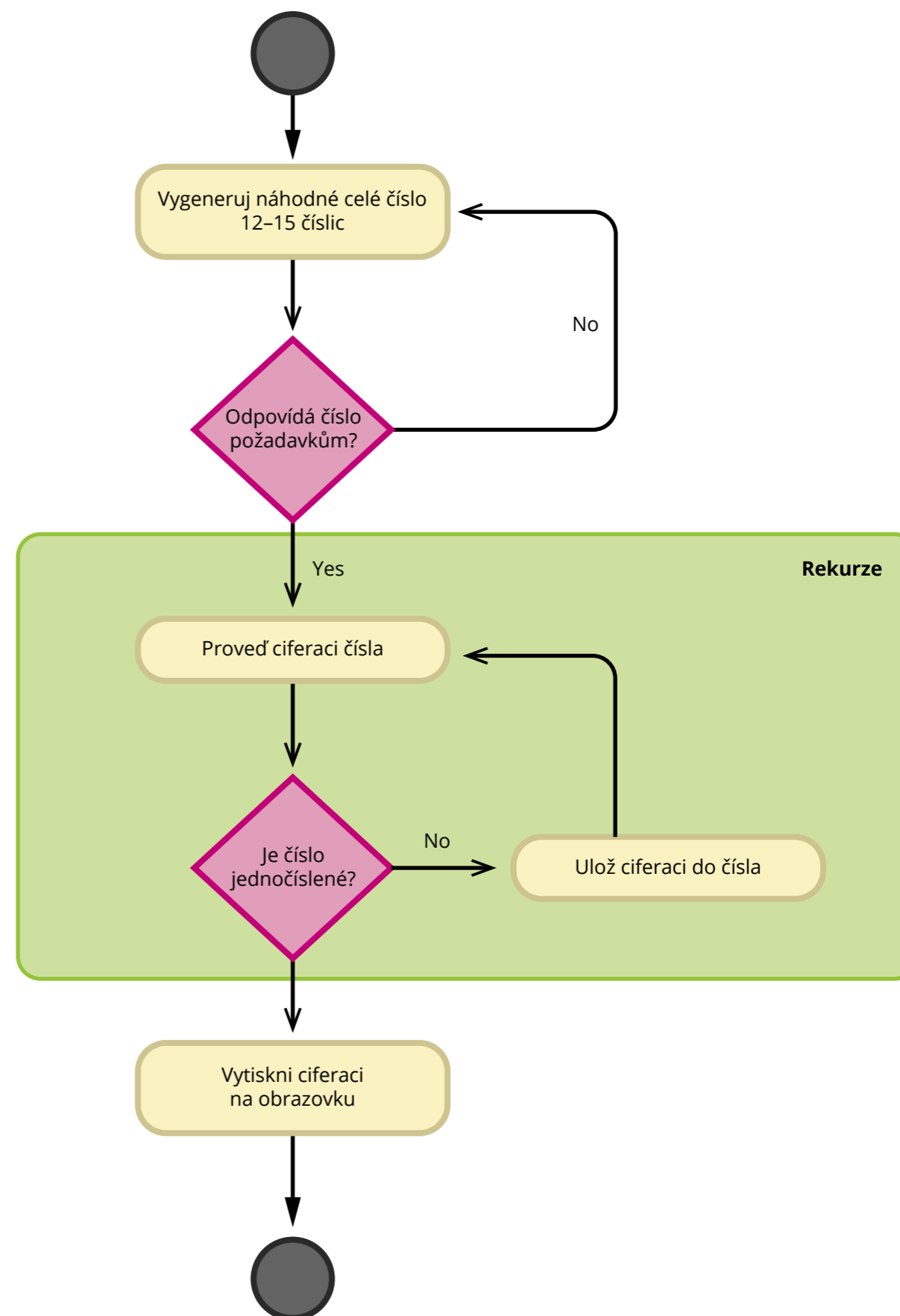
Zadání čísla – náhodné číslo o délce 15 až 20 číslic vybere počítač.

Zkontrolujeme, zda číslo odpovídá našim požadavkům:

- Číslo je celé kladné číslo
- Číslo obsahuje 15 až 20 číslic



Ciferace - vytvoření algoritmu úlohy - vývojový diagram



Cyklus

Rozdělte číslo na x číslic

Sečtěte x číslic

Zbyla vám jediná číslice?

– Pokud ano, vytiskněte ji na obrazovku a program končí.

– Pokud ne, uložte součet do čísla a vraťte se na Cyklus.

Ciferace - sestavení programu

```
#ciferace
```

```
#Autor: Marta Vohnoutová
```

```
import random as rd

def ciferace(číslo):
    c=sum([int(i) for i in str(číslo)])
    if len(str(c))==1:
        return c
    else:
        return ciferace(c)

while True:
    číslo = rd.randint(1e12,1e16)
    if isinstance(číslo, int) and číslo in range(int(1e12), int(1e16)):
        break

číslo = 248877668265883584
print(f'Pro číslo {číslo} je ciferace {ciferace(číslo)}')
Pro číslo 248877668265883584 je ciferace 6
```

Koně a pštrosi – 2. úloha prováděná učitelem

Koně a pštrosi – formulace problému

Vytvořte Python program, který:

1. Spočte podle zadaného počtu hlav a nohou, kolik po dvorku běhá koní a kolik pštrosů.
2. Pokud úloha nebude mít řešení, kdy například počet nohou bude lichý nebo počet nohou bude vůči počtu hlav více než čtyřikrát větší, program napíše, že úloha nemá řešení.

Koně a pštrosi – analýza úlohy

Vytvoříme funkci nazvanou *koně_a_pštrosi*, která bude jako atributy předávat počty hlav a počty nohou.

Zkontrolujeme, zda počet nohou není liché číslo a zda počet hlav *4 není menší než počet nohou. Pro takové případy napíšeme, že úloha nemá řešení.

Cyklus

Začneme s předpokladem, že po dvoře pobíhají pouze pštrosi.

Postupně budeme odečítat pštrosy a přidávat koně.

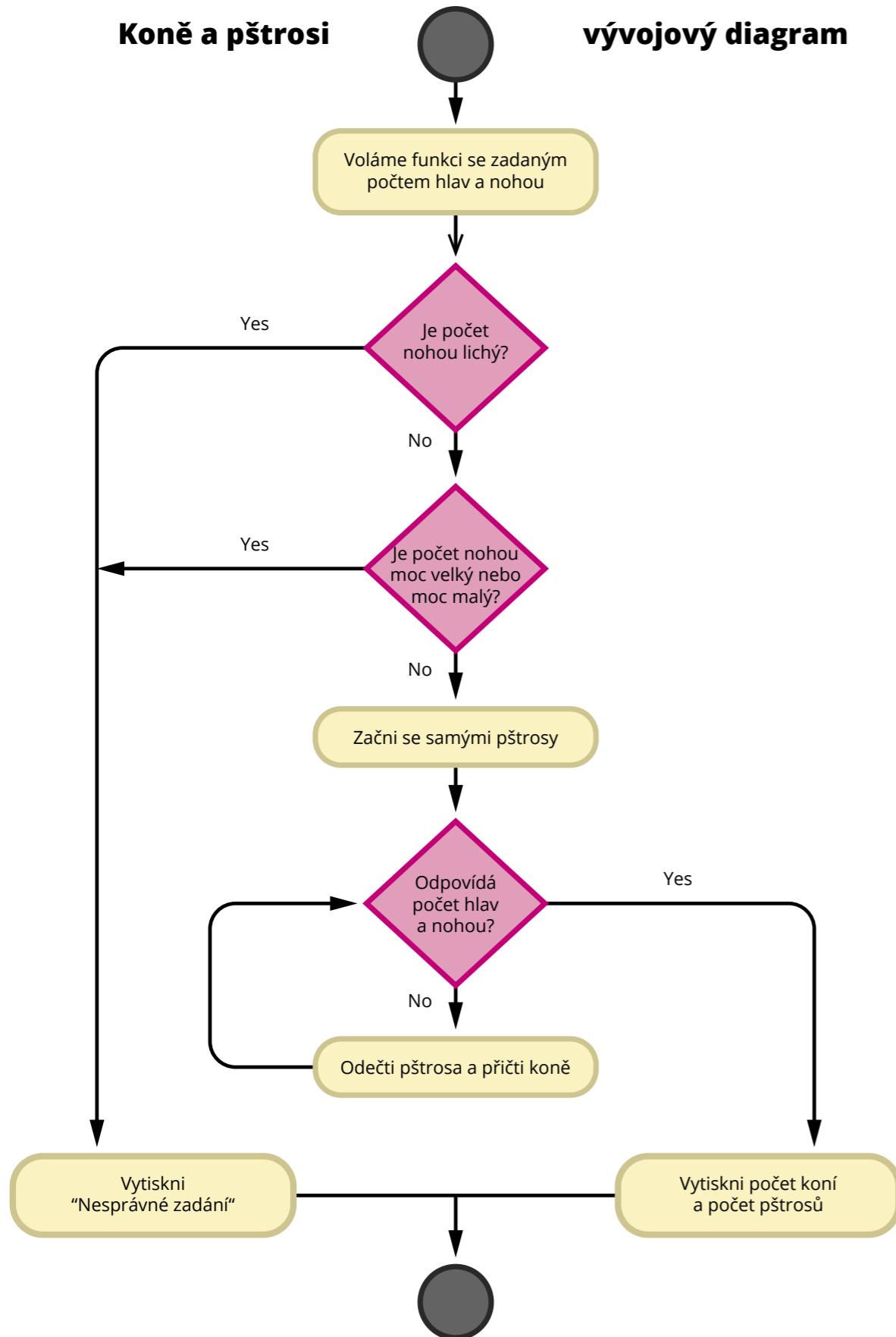
Pokud bude odpovídat počet nohou v zadání počtu nohou našich koní a pštrosů na dvorku, vrátíme řešení a ukončíme program.

Pokud nedojdeme k výsledku, napíšeme, že úloha nemá řešení.

Koně a pštrosi - vytvoření algoritmu úlohy

Koně a pštrosi - sestavení programu

Koně a pštrosi vývojový diagram

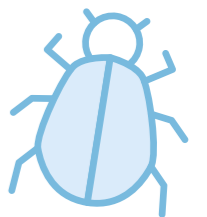
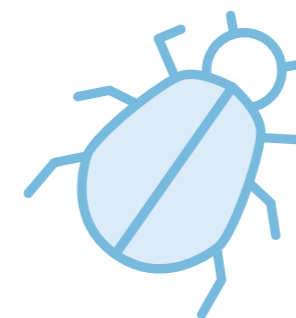
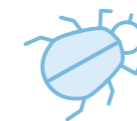


```
## koně a pštrosi
```

```
## Marta Vohnoutová
```

```
def koně_a_pštrosi(hlavy, nohy):
    if hlavy * 2 > nohy or hlavy * 4 < nohy or nohy%2 !=0:
        return 'Úloha nemá řešení'
    koně=0
    pštrosi=hlavy
    while True:
        if (pštrosi * 2) + (koně * 4) < nohy:
            pštrosi -=1
            koně +=1
        if (pštrosi+koně == hlavy) and ((pštrosi * 2) + (koně * 4) == nohy):
            break
    return f"Koní je {koně} a pštrosů je {pštrosi}"

print(koně_a_pštrosi(55,112))
Koní je 1 a pštrosů je 54
```



Úlohy pro procvičování žáky

Úlohy pro procvičování žáky nemají přiložené řešení, ale při školení učitelů je možné řešení učitelům předat. U každé úlohy je označena její obtížnost, která je samozřejmě relativní. Protože se jedná především o výuku algoritmizace a programového myšlení, tak, ačkoliv úlohy nejsou nijak složité, je vyžadováno provedení všech kroků:

1. Formulace problému
2. Analýza úlohy
3. Vytvoření algoritmu úlohy - vývojový diagram
4. Sestavení programu a jeho odladění

Úloha č.1 – snadná – Matematické hádanky

Napište Python program, který vyřeší následující matematické hádanky:

1. Najděte správná celá čísla A, B, C, D, E, F, pro která platí $ABCDEF \times 3 = BCDEFA$, kde ABCDEF je celé kladné číslo, které se vytvoří spojením jednotlivých číslic A, B, C, D, E, F. Správné možnosti vytiskněte jako *list* a v něm *tuply* takto: [(A1, B1, C1, D1, E1, F1), (A2, B2, C2, D2, E2, F2)...].
Příklad výstupu: [(1,4,2,8,5,7), (2,8,5,7,1,4)]
2. Najděte celé kladné číslice označené L, O, G, I, C, pro které bude platit $(L+O+G+I+C)3 = LOGIC$, kde LOGIC je celé kladné číslo vytvořené sdružením (concatenating) číslic L, O, G, I, C. Výsledek vytiskněte na obrazovku jako *list* obsahující *tuply* takto: [(L1, O1, G1, I1, C1), (L2, O2, G2, I2, C2)...].
3. Najděte celé kladné číslice označené C, O, W, E, D, pro které bude platit $COW \times COW = DEDCOW$, kde COW vznikne sdružením jednotlivých číslic C, O, W. Výsledek vytiskněte na obrazovku jako *list* obsahující *tuply* takto: [(C1, O1, W1, E1, D1), (C2, O2, W2, E2, D2)...].

Důležitá poznámka: Číslice mohou být v rozsahu 0...9 a v řešení se nesmí opakovat. Každá proměnná obsahuje jinou číslici.

Nápověda: Použijte permutace k vytvoření možných kombinací čísel. Permutace je podle definice uspořádaná n-tice obsahující každý prvek právě jednou. Permutace vám zpřístupní import.



```
from itertools import permutations as pm
```

```
# Marta Vohnoutová - úkol1 - Matematické hádanky
```

```
from itertools import permutations as pm
```

```
def ukol1_1():
```

```
    vystup=[]
```

```
    r=range(0,10)
```

```
    x=list(pm(r,6))
```

```
    for i in x:
```

```
        if (i[0]*100000+i[1]*10000+i[2]*1000+i[3]*100+i[4]*10+i[5]) *3 == (i[1]*100000+i[2]*10000+i[3]*1000+i[4]*100+i[5]*10+i[0]):
```

```
            vystup.append(i)
```

```
    print('Ukol 1.1',vystup)
```

```
def ukol1_2():
```

```
    vystup=[]
```

```
    r=range(0,10)
```

```
    x=list(pm(r,5))
```

```
    for i in x:
```

```
        if (i[0]+i[1]+i[2]+i[3]+i[4])**3 == (i[0]*10000+i[1]*1000+i[2]*100+i[3]*10+i[4]):
```

```
            vystup.append(i)
```

```
    print('Ukol 1.2',vystup)
```

```
def ukol1_3():
```

```
    vystup=[]
```

```
    r=range(0,10)
```

```
    x=list(pm(r,5))
```

```
    for i in x:
```

```
        if (i[0]*100+i[1]*10+i[2])*(i[0]*100+i[1]*10+i[2]) == (i[4]*100000+i[3]*10000+i[4]*10000+i[0]*100+i[1]*10+i[2]):
```

```
            vystup.append(i)
```

```
    print('Ukol 1.3', vystup)
```

```
ukol1_1()
```

```
ukol1_2()
```

```
ukol1_3()
```

```
Ukol 1.1 [(1, 4, 2, 8, 5, 7), (2, 8, 5, 7, 1, 4)]
```

```
Ukol 1.2 [(0, 4, 9, 1, 3), (0, 5, 8, 3, 2), (1, 9, 6, 8, 3)]
```

```
Ukol 1.3 [(3, 7, 6, 4, 1)]
```


Úloha č.2 – snadná – Cyklus

Pro 10 náhodných čísel v rozsahu od 19 do 999 a pro číslo 27 vytvořte cyklus, který bude provádět následující kroky:

1. Jestliže je číslo n sudé, proveďte operaci $n/2$
2. Jestliže je číslo n liché, proveďte operaci $3n + 1$
3. Celý cyklus opakujte, dokud nebude číslo $n == 1$ (n je rovno 1)
4. Pro každé číslo počítejte počet kroků než dosáhne číslo n hodnoty 1.

Vytiskněte všechny kroky pro každé číslo n a vytiskněte číslo s největším počtem kroků.

Nápověda: Náhodné číslo v daném rozsahu dostanete takto:

```
from random import randint as rd
n=rd(19,999)
```

```
# Marta Vohnoutová - úkol2 - Cyklus
```

```
from random import randint as rd
```

```
def ukol2():
    vystup={}
    for _ in range(0,10):
        vystup[rd(19,999)]=[]

    vystup[27]=[]

    for k, v in vystup.items():
        n=k
        while True:
            vystup[k].append(n)
            if n==1:
                break
            if n%2 == 0: # sude
                n=n/2
            elif n%2!= 0: # liche
                n=3*n+1
        return vystup
```

```
vystup=ukol2()
#print(vystup)
cislo=0
steps=0
for k, v in vystup.items():
    if len(v)>steps:
        steps=len(v)
        cislo=k
print(f'Pro číslo {cislo} je maximum {steps} kroků k jedničce')
```

Pro číslo 27 je maximum 112 kroků k jedničce



Úloha č.3 – snadné – Počítání samohlásek

Napište program v Pythonu, který spočte počet samohlásek ('a', 'e', 'i', 'o', 'u', 'y') v každém slově.

Zjednodušení: Slova jsou oddělena vždy mezerami, písmena jsou všechna malá a bez diakritiky. Např. věta „učíme se python kazdy den“ bude mít výstup:

Výstup: Počet samohlásek je 3 1 2 2 1

```
# Marta Vohnoutová - úkol2 - Cyklus

def ukol3(veta):
    vowels = ('a', 'e', 'i', 'o', 'u', 'y')
    slova = veta.split()
    for i in slova:
        print(i, sum([i.count(j) for j in vowels]), end=',')

ukol3('ucime se python kazdy den')
ucime 3,se 1,python 2,kazdy 2,den 1,
```

Úloha č.4 – snadné – Veliká písmena

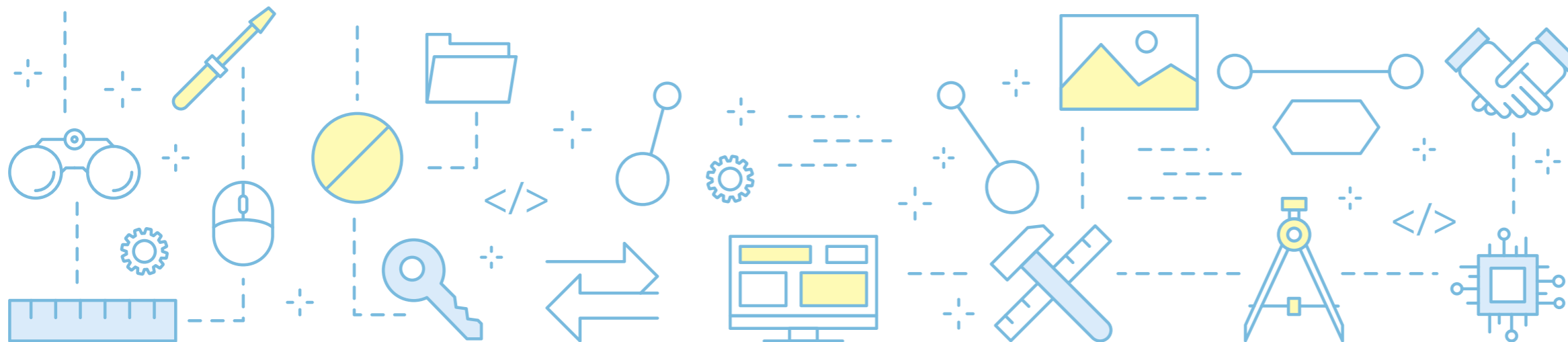
Z obrazovky zadejte nebo náhodně vygenerujte celé kladné číslo o několika číslicích. Na obrazovku pak vytiskněte číslo složené z velikých číslic. Abyste se netrápili se zadáním velikých číslic, tady je. Kdo si troufá, může hvězdičky zaměnit za konkrétní číslici.

Nápověda: Definování velkých číslic

```
Zero = [" *** ",
        " *  * ",
        "*   *",
        "*   *",
        "*   *",
        " *  * ",
        " *** "]

One = [" * ", " *** ", " * ", " * ", " * ", " * ", " *** "]
Two = [" *** ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Three = [" *** ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Four = [" *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Five = [" *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Six = [" *** ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Seven = [" *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Eight = [" *** ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]
Nine = [" *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * ", " *  * "]

Digits = [Zero, One, Two, Three, Four, Five, Six, Seven, Eight, Nine]
```



Grafy

Nejprve se musí nainstalovat knihovna Turtle. Pokud používáte Jupyter Notebook nebo Jupyterhub, jako my zde, nainstalujte si jednodušší verzi Turtle jako součást knihovny mobilechelonian. Pak se prostředí natáhne.

```
!pip install mobilechelonian
```

Nainstalovanou knihovnu musíme nainportovat.

```
from mobilechelonian import Turtle
```

Turtle teorie

Pro další inspiraci přidávám následující užitečné odkazy, ze kterých čerpáme i my.

- Turtle graphics příklady
- Turtle graphics příklady v Pythonu
- Radek Pelánek Želví grafika
- Dokumentaci k Turtle pak najdete zde

Turtle (želva) se hodí pro děti, je snadná a efektní. Želví grafiku můžeme programovat mnoha způsoby, my se budeme držet Pythonu v.3.

```
t = Turtle()
dir(t) # dir nám opět může ukázat, jaké možnosti turtle má
```

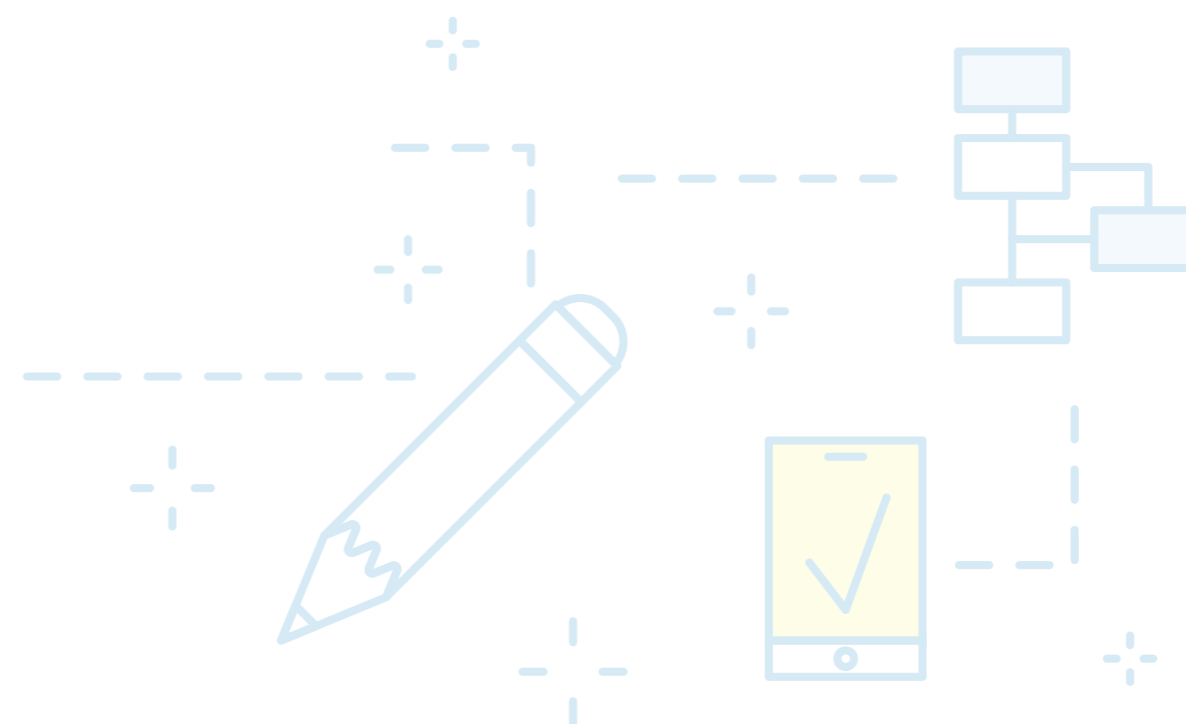
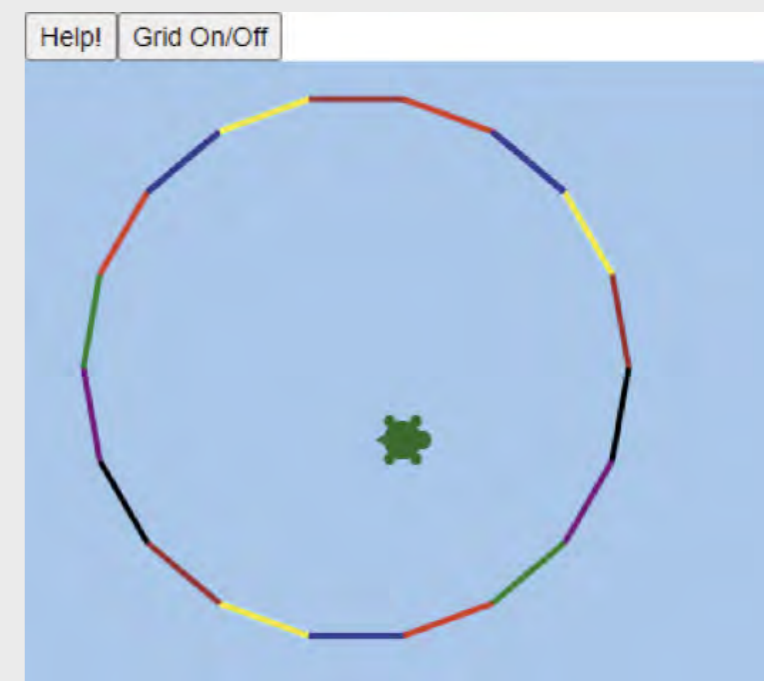
```
t = Turtle()
t.speed(9)
t.penup()
t.setposition(0,0)
t.home()
```

```
t = Turtle()
t.speed(5)
colours=["red","blue","yellow","brown","black","purple","green"]

t.penup(); t.left(90); t.forward(200);t.right(90);t.pendown()

for i in range (0,18):
    t.pencolor(colours[i%7])
    t.right(20)
    t.forward(50)

t.penup()
t.right(180)
t.home()
```



Úlohy pro procvičování žáky

Úlohy pro procvičování žáky nemají přiložené řešení, ale při školení učitelů je možné řešení učitelům předat. U každé úlohy je označena její obtížnost, která je samozřejmě relativní. Protože se jedná především o výuku algoritmizace a programového myšlení, tak, ačkoliv úlohy nejsou nijak složité, je vyžadováno provedení všech kroků:

Úloha č.5 Turtle – snadná – Napiš písmeno

Napiš funkci, která bude mít jako parametr požadované velké písmeno a barvu. Postupně můžeme do funkce přidávat další písmena.

Marta Vohnoutová

```
def napiš_písmeno(písmeno,barva):
```

```
    t = Turtle()
```

```
    t.speed(5)
```

```
    t.home()
```

```
    t.left(90)
```

```
    t.pencolor(barva)
```

```
    if písmeno == "M":
```

```
        t.forward(100)
```

```
        t.right(150)
```

```
        t.forward(70)
```

```
        t.left(120)
```

```
        t.forward(70)
```

```
        t.right(150)
```

```
        t.forward(100)
```

```
    if písmeno == "A":
```

```
        t.right(25)
```

```
        t.forward(120)
```

```
        t.right(125)
```

```
        t.forward(120)
```

```
        t.penup();t.left(180);t.forward(55);t.left(60)
```

```
        t.pendown();t.forward(60)
```

```
    t.penup()
```

```
    t.home()
```



```
napiš_písmeno('A','cyan')
```

Úloha č. 6 Turtle – snadná – Nakresli tvar podle zadání.

Pokud budeš umět, můžeš i každou stranu.

Zadání:

a)

- Délka = 2
- Opakuj 100x.
 - dopředu délka
 - doprava 89
 - přidávej k délce 2

Marta Vohnoutová

```
t = Turtle()
```

```
t.speed(5)
```

```
t.home()
```

```
délka = 2
```

```
colours=["red","blue","yellow","green","black","darkbrown"]
```

```
#colours=["purple","purple","purple","purple","purple","purple"]
```

```
j=0
```

```
for i in range(100):
```

```
    t.pencolor(colours[j])
```

```
    j +=1
```

```
    if j==4:
```

```
        j=0
```

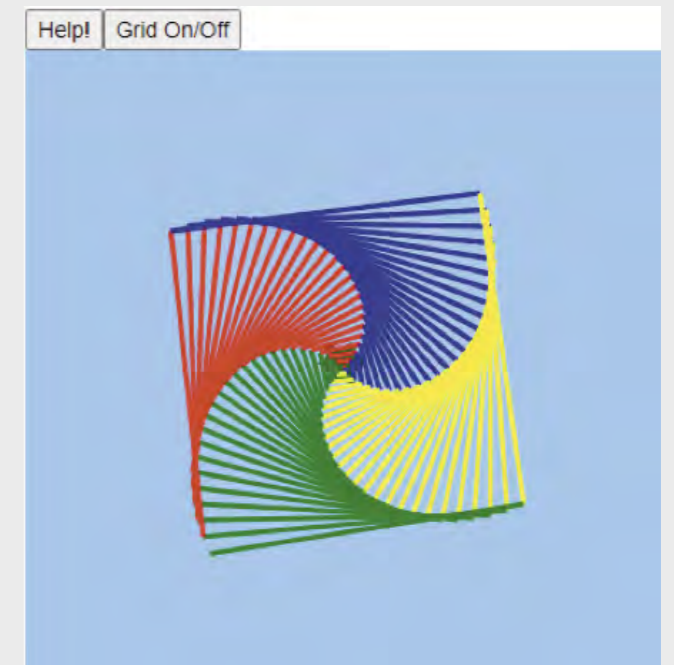
```
    t.forward(délka)
```

```
    t.right(89)
```

```
    délka += 2
```

```
t.penup()
```

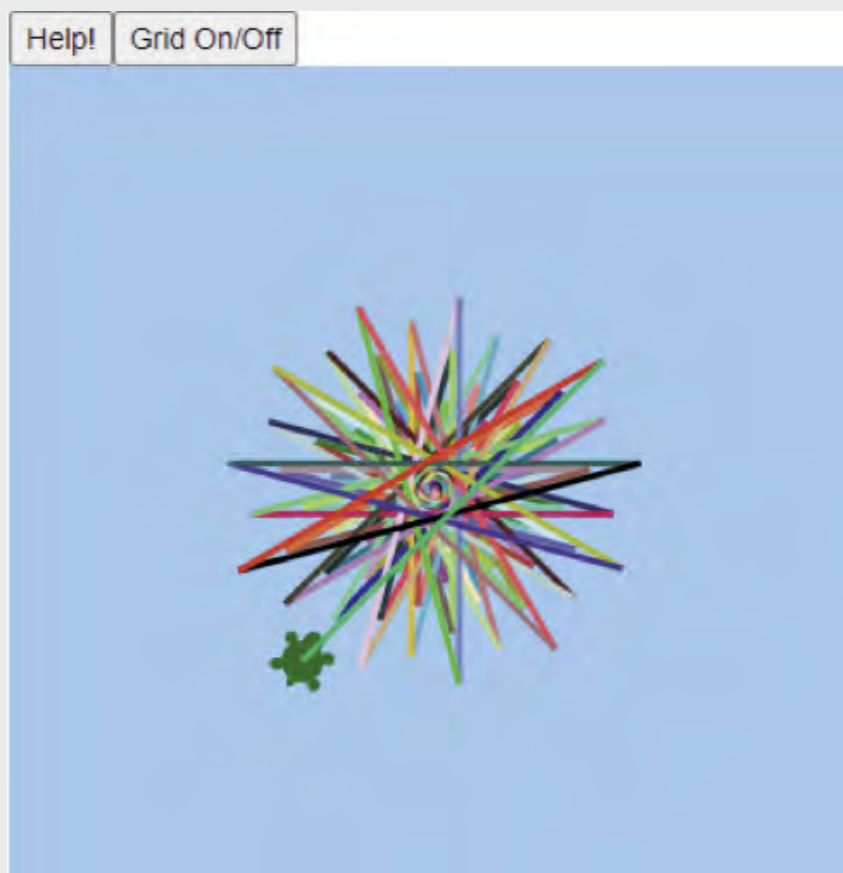
```
t.home()
```



b)

- opakování $n=40$ $\alpha=165$
- střídejte náhodně barvy
- náhodné číslo z rozsahu se generuje pomocí knihovny *random*
- *from random import randint as rd*

```
n=40
α=165
from random import randint as rd
t = Turtle()
t.speed(5)
t.penup()
t.home()
t.pendown()
délka = 2
for n in range(1,101):
    color = '#' + hex(rd(0,256)) [2:] + hex(rd(0,256)) [2:] + hex(rd(0,256)) [2:]
    t.pencolor(color)
    t.forward(délka)
    t.right(α)
    délka += 2
```



Generování bludiště

Na závěr na ukázkou příklad, generování bludiště. Bohužel knihovna Turtle je trochu omezena a nenašla jsem např. příkaz na vyplnění tvaru barvou.

Neděste se – bludiště je zde pro zajímavost na závěr.

```
# maze labyrinth 2 creating
#

from random import shuffle, randrange
from collections import OrderedDict
w=8
h=10

def make_maze(w = 16, h = 8):
    q=''
    vis = [[0] * w + [1] for _ in range(h)] + [[1] * (w + 1)]
    ver = [["| "] * w + ['|'] for _ in range(h)] + [[]]
    hor = [["+ -"] * w + ['+'] for _ in range(h + 1)]

    def walk(x, y):
        vis[y][x] = 1

        d = [(x - 1, y), (x, y + 1), (x + 1, y), (x, y - 1)]
        shuffle(d)
        for (xx, yy) in d:
            if vis[yy][xx]: continue
            if xx == x: hor[max(y, yy)][x] = "+ "
            if yy == y: ver[y][max(x, xx)] = "| "
            walk(xx, yy)

    walk(randrange(w), randrange(h))
    for (a, b) in zip(hor, ver):
        q += (''.join(a + ['\n'] + b + ['\n']))

    print(q)

lab = {}
row = 0
col = 0
```

```

for i in q:
    if i == " ":
        lab.update({(row,col):True}) # True = path is open
    elif i not in ('\n',' '):
        lab.update({(row,col):False}) # False = path is closed, it is wall

    if i == '\n':
        row += 1
        col = 0
    else:
        col += 1

sorted_lab=OrderedDict(sorted(lab.items()))
sorted_lab[(1,0)]=True
sorted_lab[(2*h-1,2*w)]=True
return sorted_lab

```

```

s_lab = make_maze(w,h)
#print(s_lab)

```

```

'''
# and draw
'''

```

```
import csv
```

```

def draw_lab(s_lab,w,h):
    smart = Turtle()
    smart.speed(50000)

```

```

x=30
y=30
smart.penup()
smart.setposition(x,y)

```

```

for rc,v in s_lab.items():
    # print(rc[0],rc[1],v)
    if v:
        smart.pencolor('lightblue')

```

```

else:
    smart.pencolor('black')

#smart.begin_fill()
smart.pendown()
for k in range(4):
    smart.forward(14)
    smart.right(90)

# smart.end_fill()
x = smart.posX+15
smart.penup()
smart.setposition(x,y)

```

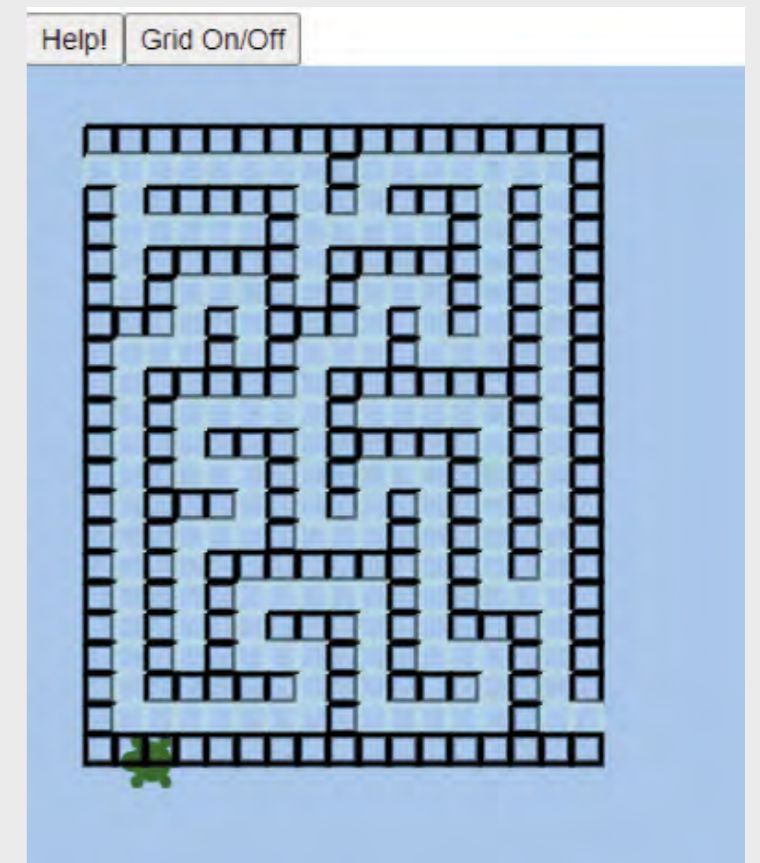
```

if w == rc[1]:
    y = smart.posY +15
    x=30
    smart.penup()
    smart.setposition(x,y)

```

```
# turtle.done()
```

```
draw_lab(s_lab,2*w,2*h)
```



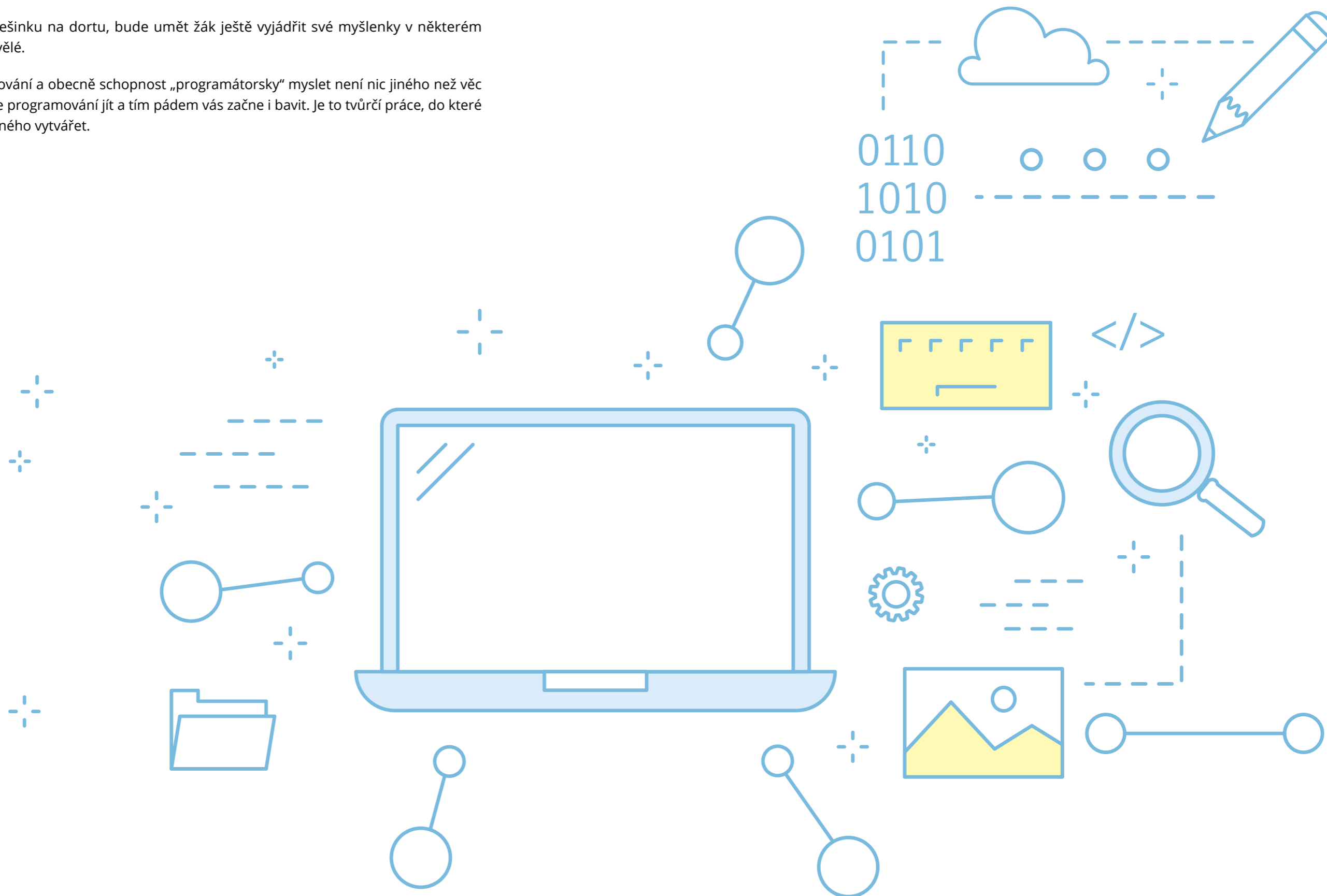
Co jste se naučili

Schopnost „programátorsky“ myslet patří dnes k důležitým dovednostem. Pokud tento kurz přispěje k tomu, aby učitelé i žáci uměli formulovat problém, analyzovat ho, zakreslit jednotlivé kroky řešení a rozhodování v programu a předvídání základních druhů chyb, které by při běhu mohly nastat, pak jsem spokojená.

Pokud k tomu, jako pověstnou třešinku na dortu, bude umět žák ještě vyjádřit své myšlenky v některém programovacím jazyce, bude to skvělé.

Stejně jako hra na piano, programování a obecně schopnost „programátorsky“ myslet není nic jiného než věc tréninku. Po určité době vám začne programování jít a tím pádem vás začne i bavit. Je to tvůrčí práce, do které můžete dát něco svého a něco cenného vytvářet.

Držím palce.



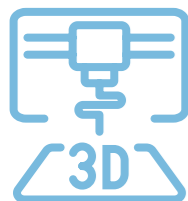


3D MODELOVÁNÍ A 3D TISK PRO ZŠ

Základní instrukce

Tento kurz je doporučen žákům druhého stupně základní školy nebo nižším ročníkům gymnázií. Autor má zkušenosti s výukou této látky od šesté třídy, nicméně do jisté míry a v jiném programu by se tento předmět mohl vyučovat i v nižších ročnících. Tento kurz lze vyučovat také v rámci všech ročníků středních škol a gymnázií. Jediný rozdíl bude v programu a hloubce učiva. Jedná se o kurz, kde se budeme zabývat postupem tvorby 3D modelů a jejich tiskem pomocí technologie FDM.

Časová dotace tohoto kurzu není striktně dána. Může záviset na úrovni všech účastníků kurzu a jejich zkušeností s 3D modelováním a 3D tiskem. Vyučující by se měl řídit časovou dotací na dané škole. Doporučená časová dotace jsou dvě spojené hodiny týdně, jak v případě výuky, tak v případě kroužku. Tisknutí pomocí 3D tiskárny je doporučeno při výuce nebo přes pracovní dny, kdy je možná kontrola o přestávkách.



Cílem kurzu je představit možnosti využití 3D modelování a 3D tisku pomocí technologie Fused Deposition Modeling (nanášení roztavené plastové struny) ve výuce na základních školách. Je zde představen celý proces od náčrtku, přes 3D modelování, slicování modelu, až po samotný tisk a možnosti postprocesingu. Na začátku kurzu jsou účastníci seznámeni s bezpečností práce a základy zacházení s 3D tiskárnou. Součástí jsou čtyři pracovní listy, na kterých jsou postupně představeny základní prvky tvorby modelů pro 3D tisk.

Zde předkládám doporučený průběh a časovou dotaci. Kurz lze rozčlenit do čtyř částí:

1. Úvod

- a. Bezpečnost práce (osobní + zařízení) - v rámci tohoto kurzu by měli žáci/studenti dodržovat základní bezpečnost práce v IT nebo jiné učebně, která je dána řádem učebny. Dále budou seznámeni s bezpečností práce a užívání 3D tiskárny a příslušenství.
- b. Základní informace o tiskárně a první tisk - seznámení s tiskárnou, ukázka tisku (tisk poté necháme běžet i během výuky)
- c. Od modelu k tisku – ještě než začneme s žáky/studenty vytvářet modely, musíme vysvětlit základní postup této tvorby. Popis jednotlivých dílčích kroků:
 - 2D náčrt
 - Vytažení do 3D modelu
 - Slicování modelu pro tisk
 - Tisk a jeho příprava
 - Následné zpracování

2. 3D modelování

- a. 2D – slouží k načrtnutí, kótování a zavazbení základního nákresu 2D zobrazení budoucího modelu.
- b. 3D – slouží k převedení dvojrozměrného nákresu do trojrozměrné podoby, kde je možné získaný model dále upravovat.

3. 3D tisk

- a. Slicing – úprava a příprava již hotového modelu na samotný tisk.
- b. Vlastnosti 3D tiskárny (FDM)
- c. Práce s 3D tiskárnou
- d. Orientace součásti pro tisk
- e. Druhy filamentu

Autoři:

Mgr. Tomáš Sosna, Pedagogická fakulta, Jihočeská univerzita v Českých Budějovicích

Mgr. Jakub Geyer, Přírodovědecká fakulta, Jihočeská univerzita v Českých Budějovicích

Editor: doc. RNDr. Ing. Jana Kalová, Ph.D.

Minimální doba, za kterou lze tento výukový balík absolvovat, je tedy čtyři vyučovací hodiny – první dvouhodinová: části 1–2, druhá dvouhodinová: část 2–3. Doporučený počet je 10 hodin (2, 2, 2, 2, 2) včetně představení projektů. Je třeba počítat s možností, že za běhu bude nutné přidat či ubrat hodiny.

Pro části 1 a úvod do problematiky částí 2 a 3 je vhodnou výukovou metodou frontální výuka spojená se samostatnou prací žáků/studentů v hodině. Učitel by měl vždy část látky vysvětlit a ukázat, poté nechat žáky/studenty, aby si vše vyzkoušeli a stihali pracovat všichni najednou.

V závěrečných hodinách pak lze s úspěchem použít projektovou výuku, kdy žáci buď dostanou přidělené téma, nebo si jej zcela sami zvolí na základě dosažené úrovně konstruování. Tyto projekty by si měli žáci sami navrhnout, rozměřit, vymodelovat, připravit k tisku a vytisknout.

Pro výuku je doporučen CAD program Solidworks, který je sice licencovaný, nicméně pro základní školu nejvhodnější z hlediska možnosti tvorby 3D modelů, uživatelského rozhraní, udržitelnosti a další návaznosti na středních a vysokých školách. V případě dobré jazykové vybavenosti žáků (učitele) lze použít podobný parametrický free CAD program OnShape, který běží ve webovém rozhraní, nicméně bohužel není v češtině. Každý žák potřebuje svůj počítač s programem nebo v případě programu Onshape připojení k internetu. Doporučuji na každou školu alespoň jednu 3D tiskárnu, v ideálním případě jednu tiskárnu na maximálně 10 žáků. Alespoň učitelův počítač by měl obsahovat program Slicer, který upravuje a připravuje tisk hotového 3D modelu.

Pro 3D tisk je nezbytné další vybavení a materiál, především filament (pro začátečníky doporučuji jako nejvhodnější materiál PLA a PET) a přípravky na přípravu tiskové podložky (odmašťovač, lepidlo, apod.). Dále je vhodné základní nářadí (špachtle, šroubováky, klíče, brusný papír, apod.). Pro spojování částí modelů je možné zajistit základní spojovací materiál případně 3D pero (které je možné rovněž použít pro kreativní projekty).

Doporučená literatura:

Průša, J. (2014). *Základy 3D tisku*. (<https://www.prusa3d.cz/kniha-zaklady-3d-tisku-josefa-prusi/>).

Vláčilová, H., Vilímková, M., & Hencl, L. (2006). *SolidWorks*. Brno: Computer Press.

Jako další zdroj je vždy důležité seznámit se s manuálem ke konkrétní 3D tiskárně.



Teoretická část k dané problematice

Pro 3D tisk na základní škole je doporučena tiskárna typu FDM (FFF). Jedná se o tiskárnu, která automaticky taví plast a vrství jej do požadovaného tvaru. Můžeme si to představit jako automatickou „tavnou pistoli“, která stále nanáší jednu vrstvu vedle druhé a na ně další, dokud není fyzický model hotov. Takovou tiskárnou je například Prusa I3 MK3S+, která disponuje jednoduchým manuálním ovládním a přívětivým uživatelským rozhraním. V rámci kurzu se tento druh tiskárny naučí bez problémů ovládat každý žák či pedagog.

CAD

CAD v překladu znamená počítačem podporované kreslení nebo rýsování. Je to program, který používáme po celou dobu konstrukce (tvorba součástí, výkresu, animace...). Původně se s CAD systémem počítalo jako s programem pro navrhování integrovaných spojů v počítačích, až později se začal používat ve strojírenství a architektuře, kde se jednalo o navrhování součástí a staveb. Ještě později se CAD začíná používat v geodézii, kartografii a geografických informačních systémech (vazba na databáze). Objevení CAD technologií kvalitně posunulo metodiku konstruování.

Asi největší předností počítačového návrhu je jeho možnost návaznosti na další technologické činnosti. Jako příklad lze uvést některé komplikované tvary při výrobě automobilů.

Výběr softwaru

V dnešní době existuje na trhu velké množství různých programů pro 3D modelování. Tyto programy lze rozdělit podle oblastí využití (strojírenství, elektrotechnika, architektura aj.), dostupnosti (free verze, licencované verze), podle způsobu postupu při modelování (parametrické, neparametrické) a spoustu dalších dělení. Parametrický CAD znamená CAD program, který vytváří model postupně, obvykle od 2D náčrtu po 3D model s využitím vztahů/omezení (constraints), oproti tomu neparametrický CAD (direct modeling) vytváří 3D modely přímo bez vztahů a bez závislé historie kroků.

Licence/free verze

Pro většinu škol je nejdůležitějším faktorem dostupnost neboli cena. Sice není potřeba žáky učit hned v programu, který nám umožní vymodelovat vše, na co si vzpomeneme, nicméně pakliže máme kvalitní software k dispozici a žáci se v něm naučí orientovat a pracovat (stačí základy), mají do budoucna dobrou přípravu a zkušenost, protože na středních/vysokých školách se pracuje především s těmito kvalitními programy a je výhodné, když žáci mohou kontinuálně navázat na své zkušenosti ze základní školy. V opačném případě dochází dost často k přeučování již získaných zkušeností a nelogických postupů.

Aktuálně lze získat licencovaný program na ZŠ poměrně levně, respektive školy mají možnost čerpat peníze na takové programy přímo z ministerstva v rámci šablon nebo dalších projektů.

3D tisk

3D tisk je proces, při kterém se z digitální předlohy (3D model) vytváří fyzický model. Jedná se o aditivní proces výroby (Additive Manufacturing) pomocí automatizované 3D tiskárny. Tisk se provádí po vrstvách.

Použití

3D tisk nachází uplatnění všude tam, kde je zapotřebí výroba prototypů, malovýroba, personalizovaná výroba, výroba jinak nesehnatelných součástí (již se nevyrábí), apod.

Velkou výhodou 3D tisku je snadné sdílení modelů. Pokud tak někdo na jednom konci světa vytvoří model součástky (včetně ověření možnosti jeho tisku), tento model lze snadno předat či sdílet komukoliv na světě, kdo má přístup k odpovídající 3D tiskárně.

Oblasti užití

- Výroba zboží či příprava odlitků zboží (šablony)
- Letectví, kosmonautika, automobilový průmysl
- Zdravotnictví (např. přesné náhrady kostí)
- Stavebnictví
- Umění

Druhy 3D tisku

Mezi nejrozšířenější formy 3D tisku patří FDM (FFF), SLA (vč. DLP) a SLS (vč. DMLS). Pro použití ve výuce ZŠ je nejvhodnější, nejjednodušší a zároveň ekonomicky nejlevnější variantou 3D tisk FDM.

FDM (FFF)

Fused Deposition Modeling (Fused Filament Fabrication) patří mezi nejrozšířenější formy 3D tisku a zároveň je nejvhodnější a ekonomicky nejlevnější varianta pro použití ve výuce na ZŠ.

- Princip (elektronicky řízené) „tavné pistole“. Tisk probíhá kladením jednotlivých linek roztaveného plastu, které postupně tvoří vrstvy modelu.
- Nelze tisknout „do vzduchu“, často je tak nutné využít tištěných podpor či jinak tento problém řešit (viz. kapitola orientace modelu pro tisk).
- Materiál = filament (plastová struna podobná té do křovinořezu).
- Některé tiskárny umožňují tisk z více filamentů, resp. jejich střídání. Toho lze využít pro tisk rozpustných podpor, vícebarevného tisku či kombinaci např. flexibilních a pevných materiálů.

Druhy filamentu

Základní a nejpoužívanější filamenty, které jsou vhodné i pro základní a střední školy, jsou filamenty z plastu. Podle jejich vlastností, teploty tisku atp. je můžeme rozdělit na následující:

ABS

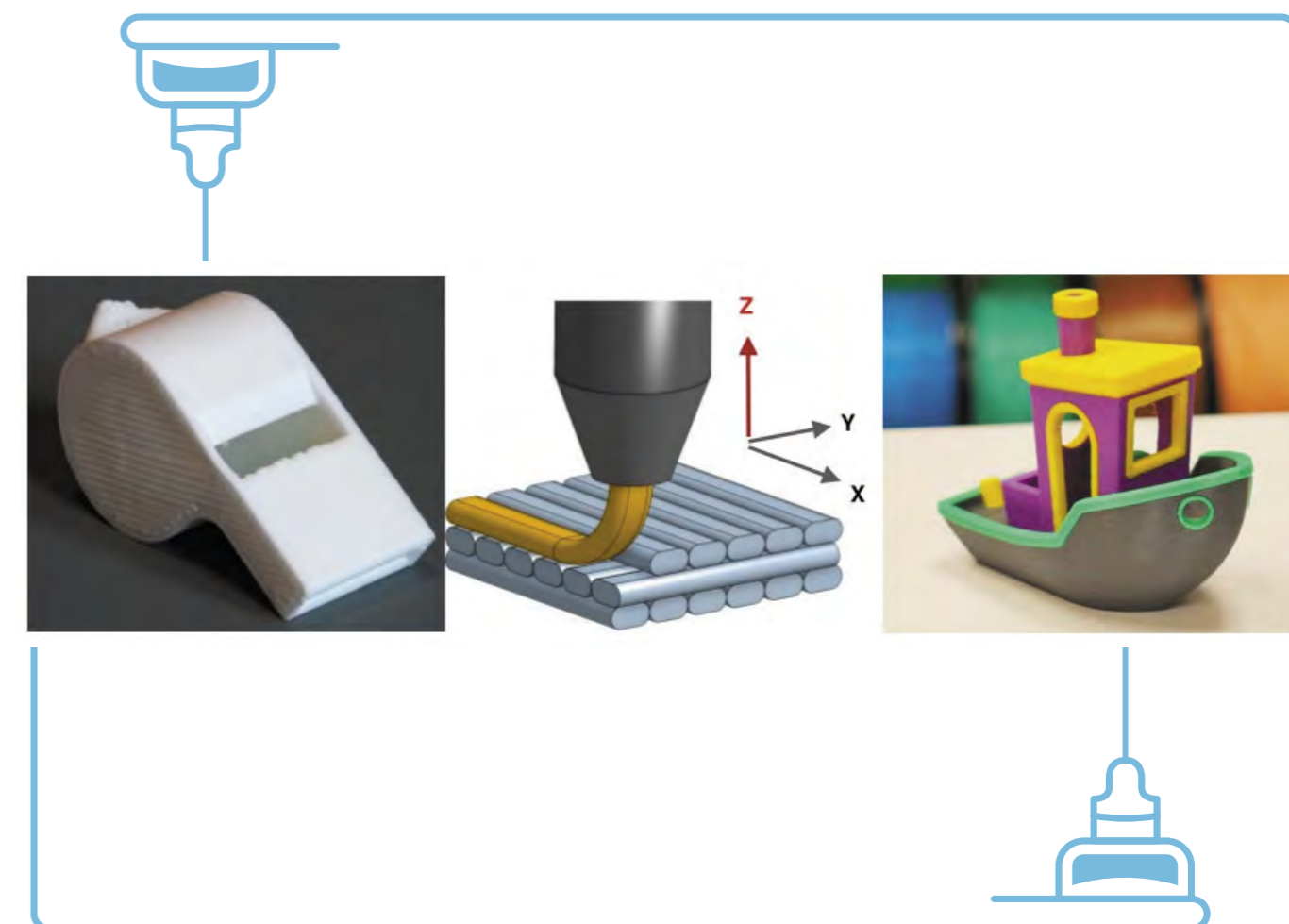
Tento druh filamentu je poměrně odolný vůči mechanickým poškozením, dále je velmi tuhý, houževnatý a odolný proti extrémním teplotám. Jednou z jeho dobrých vlastností pro školství je jeho zdravotní nezávadnost (dají se z něj vyrábět jídelní misky, hrníčky atp.). Kromě těchto vlastností je tento filament i snadno opracovatelný. Teplota tisku je 220–240 °C.

PLA

Asi nejrozšířenější filament u nás, který je biologicky plně odbouratelný. Vyrábí se ze zpracované cukrové třtiny, bramborového škrobu či kaučuku. Z vytištěného modelu se hůře odstraňují případně podpěry. Teplota tisku je 185–235 °C.

PET

Je to filament, který propojuje nejlepší vlastnosti filamentů ABS a PLA. Zároveň má velmi malou teplotní deformaci a je odolný. Velmi výhodný pro FDM technologie tisku. Teplota tisku je 220–260 °C.



Příklady z praxe

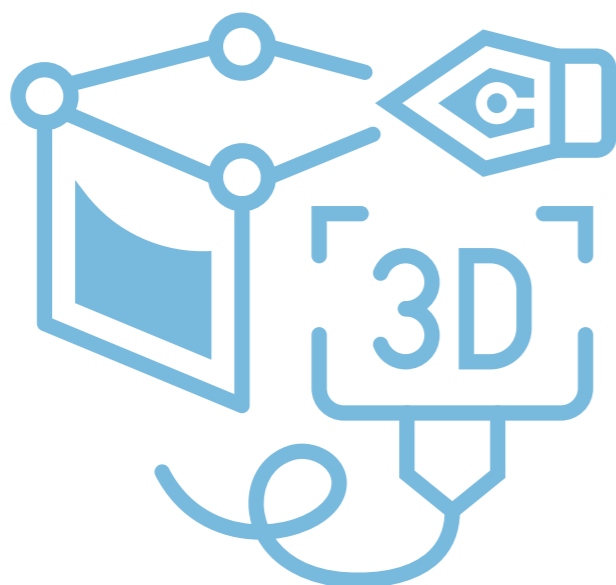
Autora při vytváření tohoto materiálu inspirovala příhoda z praxe, která se stala v rámci výuky kroužku na ZŠ.

Dva žáci již měli zkušenosti s 3D modelováním, nicméně v neparametrických programech, což způsobovalo velké problémy nejen při přeorientování na parametrický CAD, ale také v možnostech modelování. Oba chlapci byli do modelování velmi nadšení, ale jak sami přiznali, byli limitováni možnostmi neparametrického Tinkercadu.

Díky parametrickému SolidWorksu si i přes počáteční problémy, spojené s přeorientováním, tento program oblíbili a prakticky již zvládají modelovat i poměrně složité konstrukce, které jsou vhodné spíše pro vyšší ročníky středních škol.

Absolvováním této výuky žáci/studenti získají nebo si prohloubí základní návyky v rámci 3D modelování. Dále se dozví informace ze světa 3D tisku, možností volby programů a základů technické dokumentace. Zde záleží pouze na učiteli, který musí odhadnout úroveň svých žáků a jak hluboko si může dovolit ponořit se do dané problematiky.

V neposlední řadě lze tímto úkolem rozvíjet technické znalosti žáků – již zmiňované základy technické dokumentace, prostorovou představivost, technickou představivost aj. S úspěchem lze na závěr zadat žákům projekt, kde si každý dle svých schopností a představivosti může vytvořit vlastní model, který si i sám vytiskne (naučí se obsluhovat 3D tiskárnu). Tyto vytisknuté modely mohou sloužit i pro propagaci školy v rámci dnu otevřených dveří aj.



Metodická a didaktická část

Úvod

V rámci první hodiny by žáci na začátku měli být především seznámeni s bezpečností práce, aby se předešlo možnému zranění žáků nebo poškození tiskárny a příslušenství. Dále by se žáci měli ve zkratce seznámit s tím, co je 3D tiskárna a provést první krátký tisk (předem připravený GCODE). Následně je již možné se pustit do základů modelování.

Bezpečnost práce

V rámci kurzu by měli žáci/studenti dodržovat základní bezpečnost práce v IT nebo jiné učebně, která je dána řádem učebny. Dále by měli být hned v úvodu seznámeni s bezpečností práce a správným užíváním 3D tiskárny a příslušenství. Žáky je zejména potřeba upozornit na:

- Práci s elektrickým zařízením (nebezpečí při políhnutí, ventilační otvory zdroje a elektroniky, atd.).
- Nářadí s ostrými hranami (nože, špachtle, apod.). **Zejména zranění ostrou špachtlí při sundávání po tisku patří mezi nejčastější úrazy související s 3D tiskem.**
- Části zařízení s vysokou teplotou (tryska, vyhřívaná podložka, krokové motory)
- Mechanické pohyblivé části – hrozí přiskřípnutí.
- Chemikálie (aceton, IPA, líc, lepidla, atd.).

Při práci s 3D tiskárnou by měli žáci vždy dodržovat: následující zásady:

- Se zařízením pracovat pouze na pokyn učitele a vždy dodržovat jeho pokyny. Pokud budou mít žáci s tiskem jakýkoliv problém, měli by se ihned obrátit na vyučujícího.
- Nekonzumovat jídlo a nemanipulovat s tekutinami v blízkosti tiskáren.
- Pozor na statický náboj (zejména displeje některých tiskáren jsou citlivé na statický náboj – může dojít k chvilkovému rozrušení displeje nebo v extrémním případě i k poškození).
- Vyvarovat se poškození tiskového povrchu (tryskou, špachtlí, výměna tiskového plátu, apod.).
- Nevypínat tiskárnu, dokud tryska nevychladne pod 50 °C. Při závažnější chybě tisku mají uživatelé 3D tiskárny tendenci tiskárnu vypnout, to však vede k vypnutí ventilátoru a nárůstu teploty nad tryskou, což může vést k jejímu uspání či poškození.
- V učebně dostatečně větrejte a zbytečně neinhaluje zbytkové látky při tisku (ani v případě bezpečných materiálů). Pozor ale na průvan, který by mohl ztížit tiskové podmínky.
- Nenechávejte nikdy tisk bez dozoru.

Žáci by se měli rovněž seznámit s manuálem a bezpečnostními pokyny ke konkrétní tiskárně.

Seznámení s tiskárnou a první tisk

Žáci/studenti by si měli ve skupině vyzkoušet kontrolu tiskárny a přípravu pro první tisk. Na začátku je potřeba je seznámit se základním fungováním tiskárny (osy, tryska, tisková plocha/podložka) a práci s nimi. Rovněž je potřeba se seznámit se základním ovládáním (ovládací prvky, vypínač, reset tlačítko – jeli k dispozici).

Pokud je k dispozici více tiskáren, ideálně by měl učitel vysvětlit a ukázat postupně očištění/odmaštění tiskové plochy, odejmutí a usazení tiskového plátu (je-li jím tiskárna vybavena), přehřev a zavedení filamentu. Tento postup mohou studenti pod dozorem replikovat na dalších tiskárnách. Důležité je umožnit studentům opakovat dílčí kroky a vždy počkat na dokončení, aby se zamezilo případným chybám.

Následně je možné spustit první tisk (z předem připraveného GCODE). Vhodné jsou například vzorové modely od PrusaResearch (dostupné s instalací Prusa software nebo na <https://www.prusa3d.cz/3d-modely-pro-tisk/>). Mohou to být například píšťalka či Marvin.

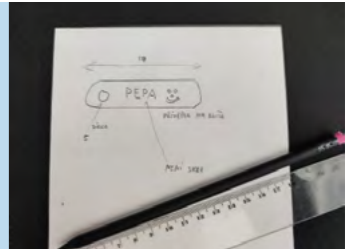


Od modelu k tisku (jak postupovat)

Ještě, než začneme s žáky/studenty vytvářet modely, musíme vysvětlit základní postup této tvorby a následných kroků pro jejich vytištění. Dílčí kroky je možno shrnout do několika bodů:

2D Náčrt

Náčrt na papír, který pomůže ucelit představu o finálním produktu. Později může být využit jako základ pro 2D skicu v CAD.



Představa

Již pro náčrt je vhodné mít k dispozici pravítko, úhloměr, apod. pro snazší odhad velikostí.

Tvorba 3D modelu

Pomocí 2D náčrtů a jejich „vytažením“ do 3D postupně vytváříme model.



STL/OBJ

Export modelu a načtení do sliceru.

Slicing

Příprava modelu pro tisk na konkrétní tiskárně a z konkrétního materiálu. Nastavení parametrů tisku (rychlost, perimetry, výplň, atd.)



GCODE

Přenos do tiskárny.

Tisk

Příprava materiálu, tiskové plochy, tisk.



Tištěný díl

U modelu jsme zapomněli zaoblit hrany, můžeme je tak např. zabrousit.

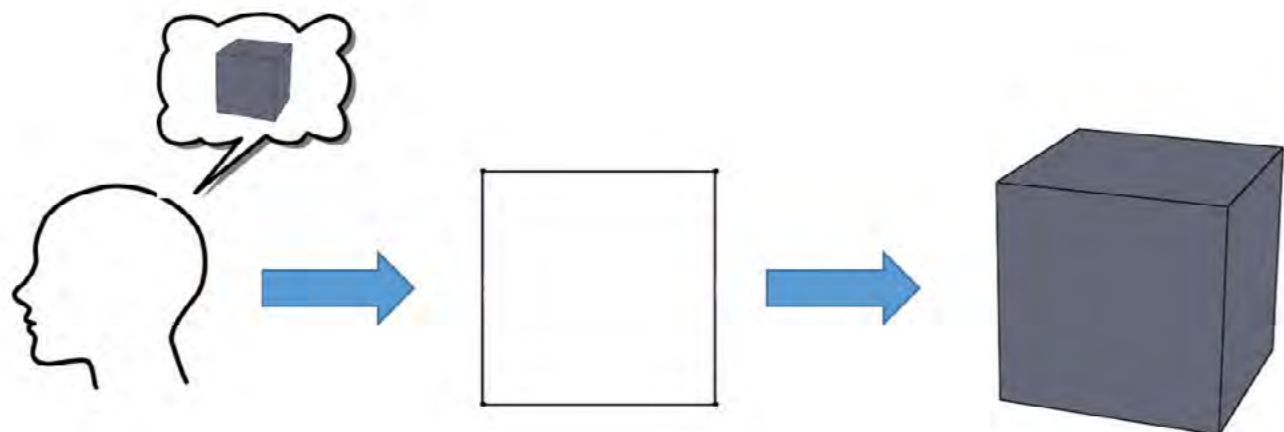
Postprocesing (následné zpracování)

Očištění, lepení, broušení, spojování, atd.



3D modelování

Je to proces tvorby výsledného modelu (od myšlenky, přes náčrt až po hotový model), který má několik fází. Tyto fáze mají jasnou posloupnost, kterou nelze měnit.



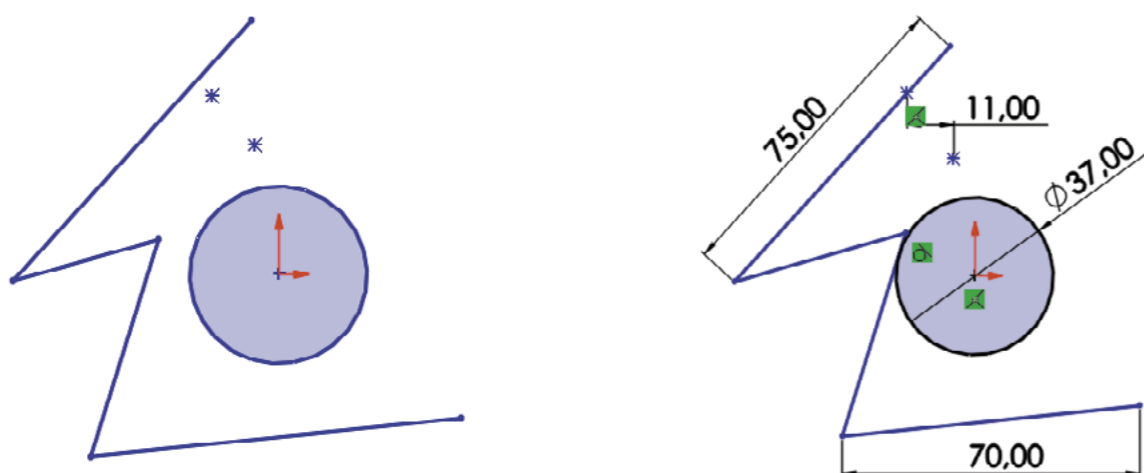
2D náčrt

Slouží k načrtnutí, kótování a zavazbení základního nákresu 2D zobrazení budoucího modelu.

V první fázi bychom měli seznámit žáky/studenty s prostředím programu, ve kterém budeme pracovat. Žáci /studenti by si měli osvojit vlastnosti a funkce prostředí. V parametrických CAD programech vytváříme napřed 2D náčrt. Ten se obvykle vytváří na skicu.

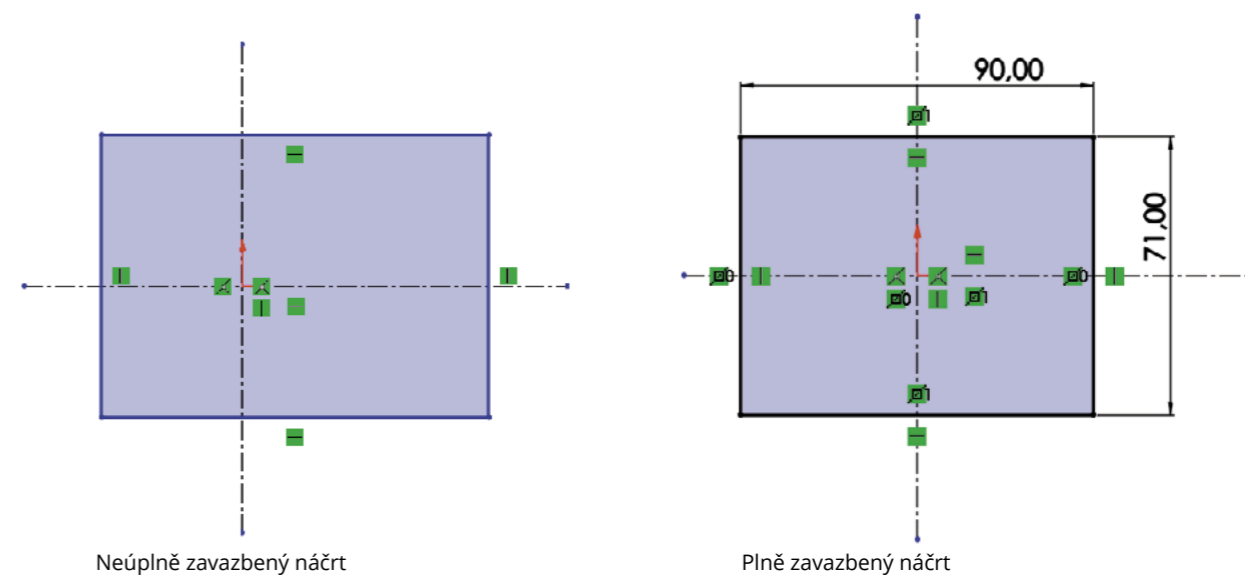
Skica

Skica je "papír", na který lze zakreslit, pomocí jednotlivých nástrojů, různé čáry, tvary, body atp. Tyto kresby můžeme zakótovat a zavazbit, případně pomocí některého nástroje zkopírovat, zrcadlit, ořezat, spojit atp. Po dokončení náčrtu skicu jednoduše uzavřeme (můžeme se do ní kdykoli během práce vrátit).



Plně určený náčrt

Po nakreslení náčrtu je vždy dobré tento náčrt tzv. plně určit. To můžeme učinit pomocí vazeb (tečná, sjednoceno, kolmá, ...) nebo kót (rozměrů). Zabráníme tak možné deformaci náčrtu a zároveň si ověříme, že daný náčrt má všechny potřebné parametry k převodu na 3D model. V rámci kótování a vazbení můžeme velmi dobře využívat osy, které nám práci ulehčí.



3D model

Slouží k převedení dvojrozměrného nákresu do trojrozměrné podoby, kde je možné získaný model dále upravovat. Po dokončení 2D náčrtu uzavřeme skicu a můžeme využít některou z funkcí na vytvoření 3D modelu. Funkci vybíráme podle vhodnosti k našemu náčrtu. Máme několik základních funkcí:

- Vysunutí
- Rotace
- Spojení profilů
- Tažení po křivce

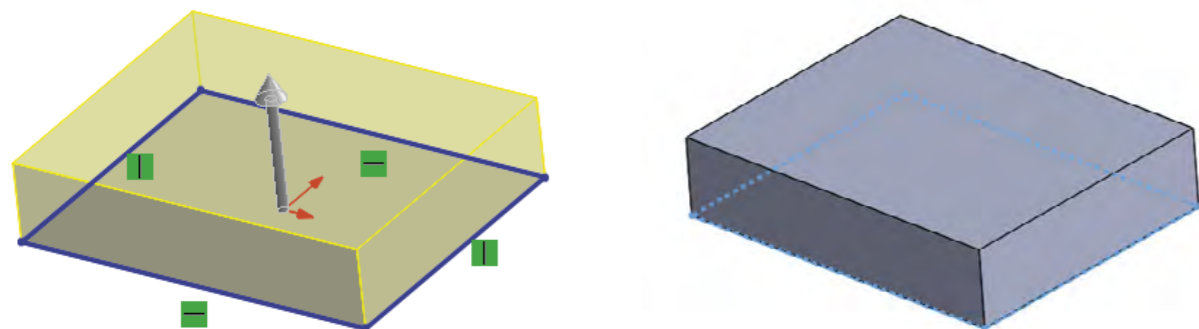
Dále zde můžeme již hotový model upravovat pomocí dalších nástrojů k tomu určených, nicméně je téměř vždy lepší udělat všechny úpravy, pokud je to možné, již v 2D náčrtu.

Vysunutí

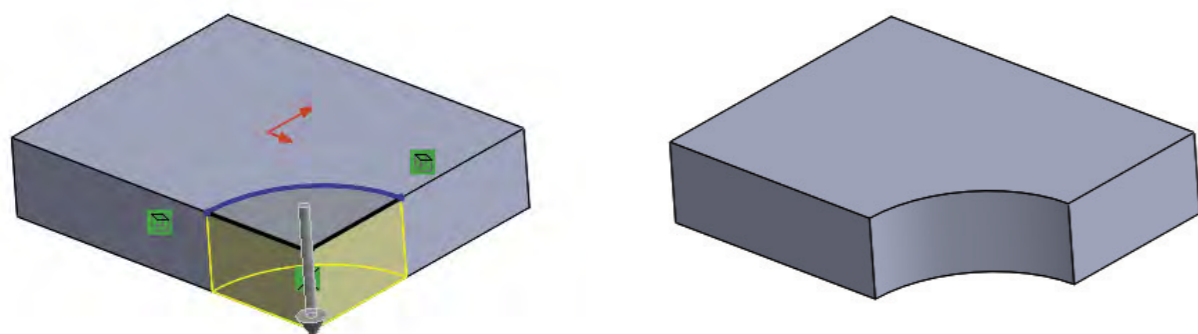
Tato funkce vysouvá námi vybraný 2D náčrt, uzavřenou část tohoto náčrtu nebo jiný uzavřený tvar. Lze zvolit směr, tvar a délku vysunutí. Jedná se o nejjednodušší a též nejpoužívanější funkci. Má dvě možnosti použití, a to Přidání vysunutím a Odebrání vysunutím.

Přidání vysunutím

Vysouvá materiál dle zadané délky a tvaru daným směrem a tím vytváří z 2D náčrtu prostorový 3D model.

**Odebrání vysunutím**

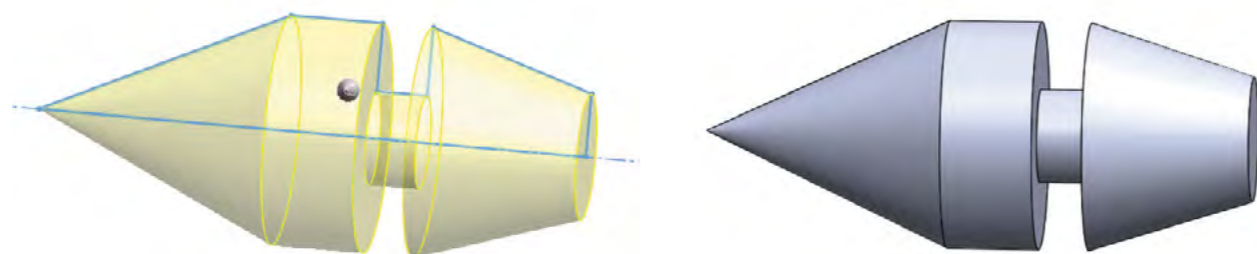
Vysouvá (odebírá) materiál dle zadané délky a tvaru daným směrem a tím vytváří do již hotového 3D modelu různé otvory, či ho jinak tvaruje.

**Rotace**

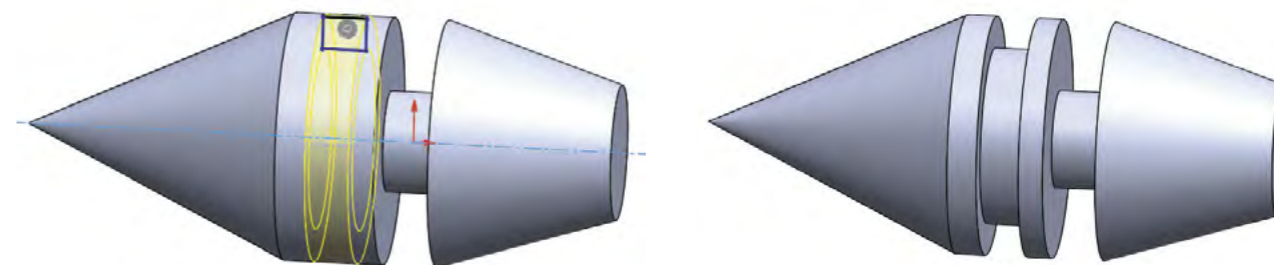
Tato funkce potřebuje ke správnému fungování jednu z os, kolem které rotuje námi nakreslený 2D náčrt a tím ho převádí na 3D model. Je zde zapotřebí prostorové představivosti, neboť na první pohled nemusíme mít 2D náčrt správně, či se nám může jen zdát, že ho nemáme správně. Jedná se o druhou nejčastěji používanou funkci. I v tomto případě máme dvě možnosti použití - Přidání rotací a Odebrání rotací.

Přidání rotací

Pomocí osy rotujeme nakreslený náčrt a ten nám vytvoří model. Náčrt musí být uzavřený k ose a zároveň poslední část kopíruje osu, jinak nám program bude hlásit chybu. Můžeme určit úhel rotace - zda se součást rotuje kolem osy dokola či pouze pod námi zadaným úhlem.

**Odebrání rotací**

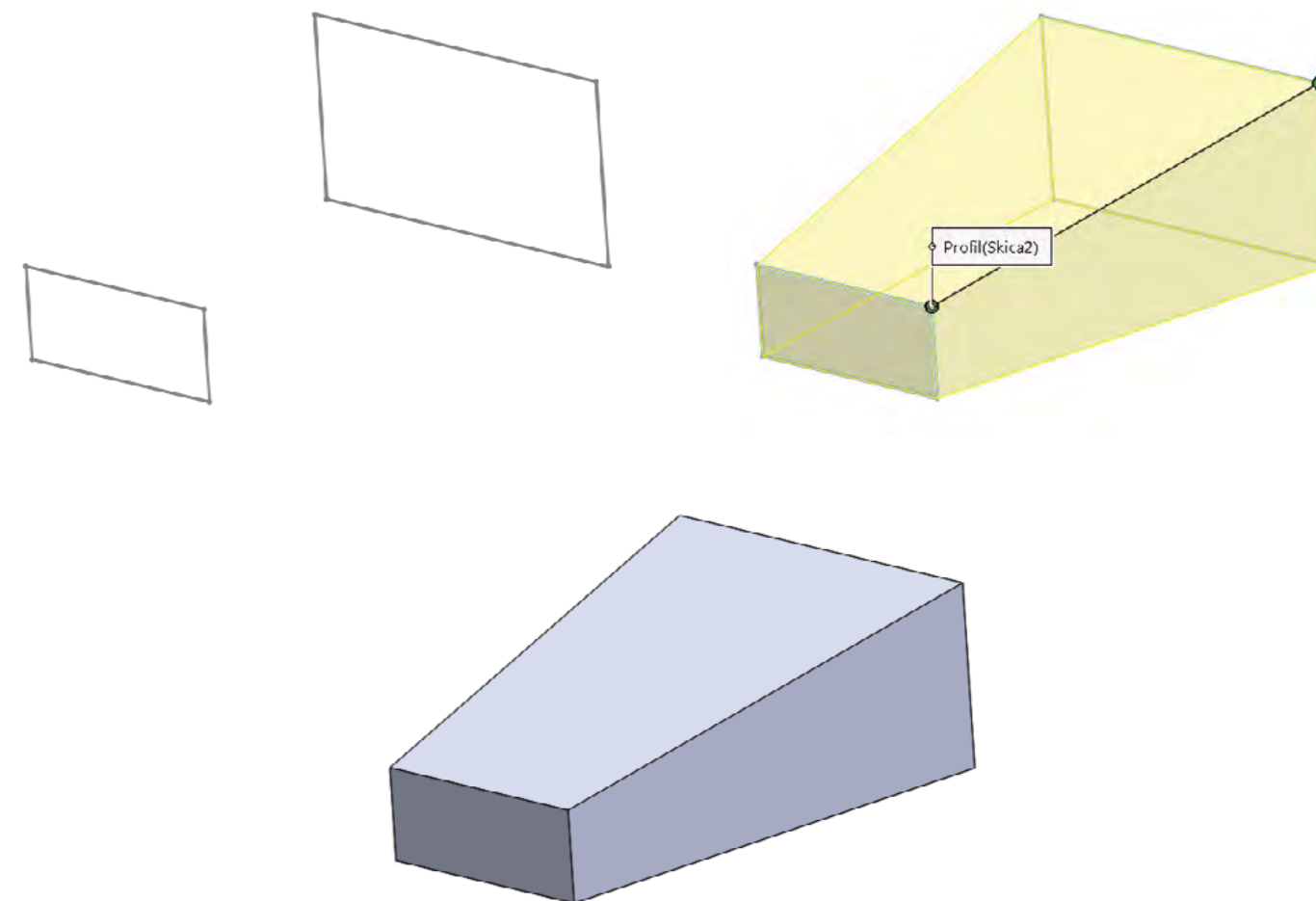
Pomocí osy rotujeme nově vytvořený náčrt "ve" vytvořeném modelu. Zde nemusíme řešit uzavření náčrtu s osou, nicméně náčrt musí být uzavřený. Většinou se tato funkce využívá pro úpravy již hotových polotovary modelu.

**Spojení profilů**

I zde se jedná o pokročilejší funkci. Jak napovídá název, jedná se o spojení dvou již vytvořených profilů (náčrtů). Tyto náčrty leží každá na jiné skici a jsou od sebe obvykle vzdáleny, opět si zde můžeme pomoci Rovinou. Máme zde možnost přidávat nebo odebrat.

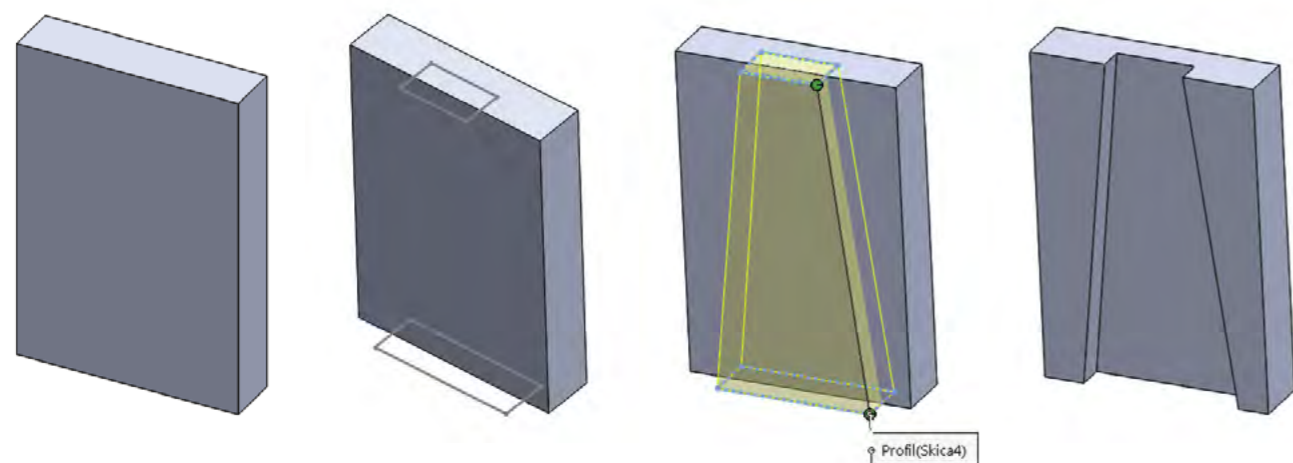
Přidání spojením profilů

Jedná se o vytváření 3D modelu z dvou nebo více náčrtů. Je doporučeno jednotlivé náčrty umísťovat od sebe ve správných a promyšlených rozměrech, aby nedošlo k chybě (spojení by se někde protínalo).



Odebrání spojením profilů

Podobně jako u odebrání tažením, i zde můžeme odebrat část materiálu již hotového modelu. Opět si musíme udělat dva nebo více náčrtů na různé roviny (využijeme nástroj Rovina). Po dokončení a uzavření všech skic použijeme funkci Odebrání spojením profilů a máme hotovo.

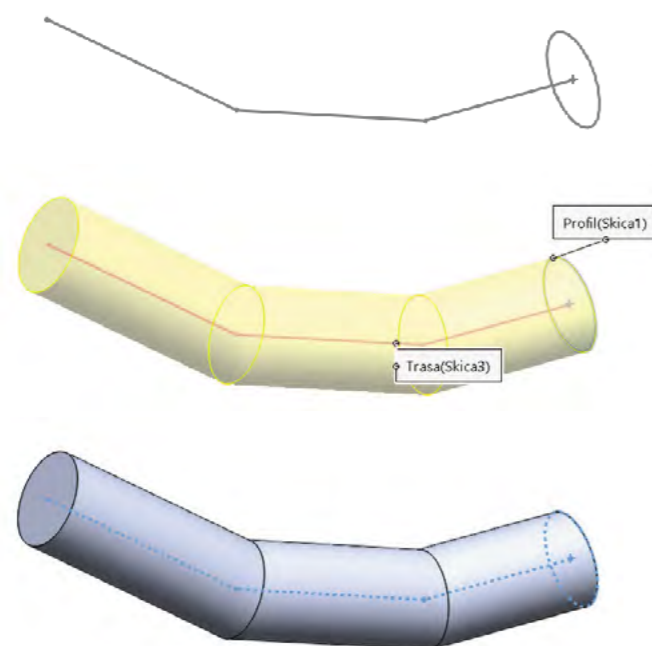


Tažení po křivce

Tato funkce již patří k těm pokročilejším. Jak již napovídá název, dochází při použití této funkce k „tažení“ určitého tvaru (například kružnice) po dané křivce (přímka, oblouk atd.) čímž vzniká požadovaný model (potrubí, drátu atp.). Pro použití této funkce potřebujeme dvě roviny, ve kterých na skici nakreslíme tvar a křivku. I zde můžeme přidávat a odebrat.

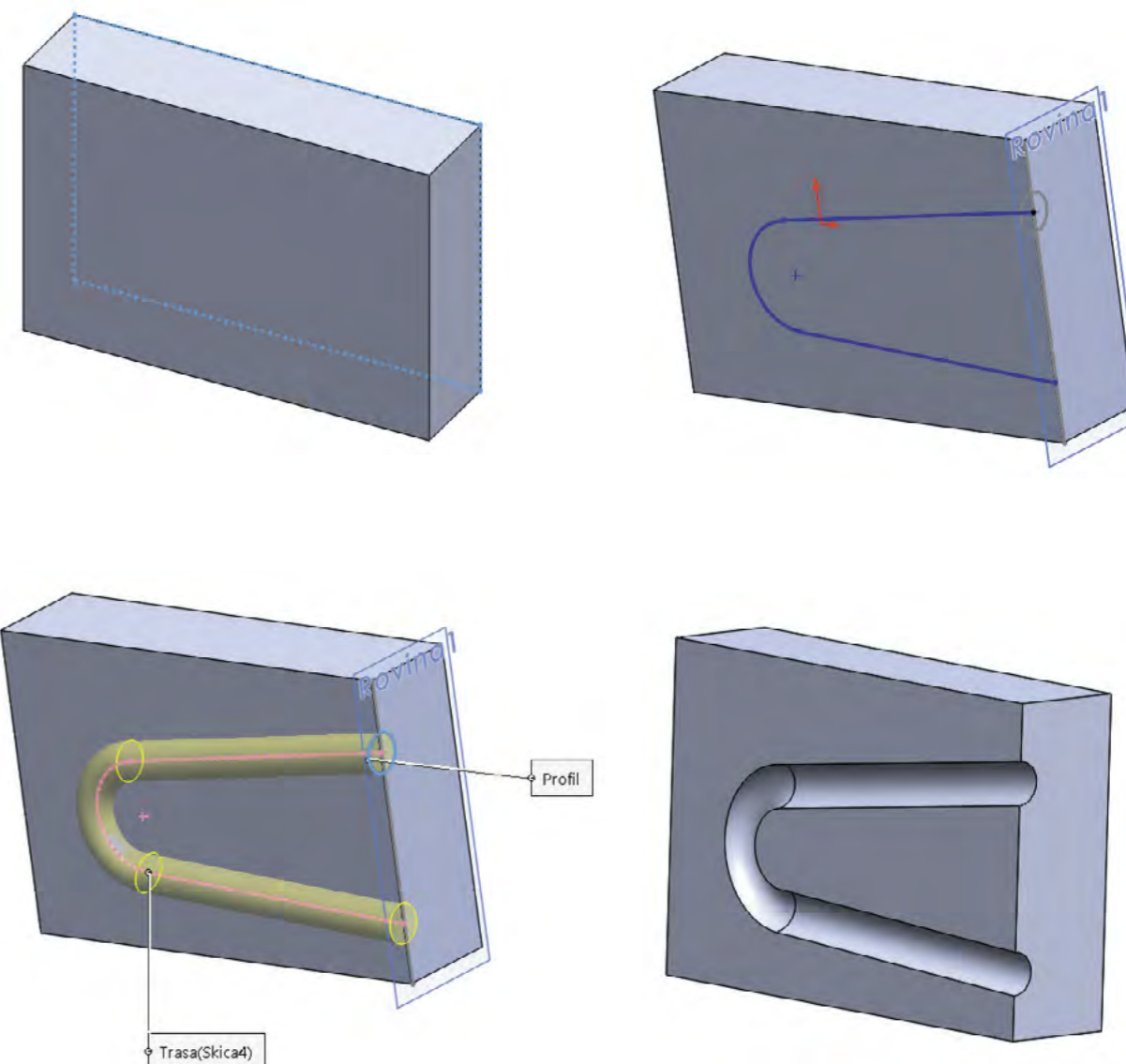
Přidání tažením po křivce

Pro jakýkoliv náčrt obvykle používáme přední rovinu, ne jinak tomu bude i v případě tvaru (kružnice), po dokončení a uzavření skici si zvolíme rovinu kolmou na přední rovinu, a to pravou rovinu. Na této rovině si otevřeme skicu a nakreslíme pomocí přímek křivku. Po ukončení a uzavření skici použijeme funkci Přidání tažením po křivce, kdy volíme, co je tvar a co je křivka.



Odebrání tažením po křivce

Používá se především, pokud chceme z již hotového modelu částečně odebrat určitý složitější tvar. Opět budeme potřebovat dvě na sebe kolmé roviny se skicami, navíc budeme potřebovat pravou rovinu posunutou na okraj původního modelu, což uděláme přes příkaz Rovina a určíme vzdálenost a pak na ní otevřeme skicu. Na této skice vytvoříme tvar, který budeme táhnout. Poté si vytvoříme skicu na již hotové ploše modelu a nakreslíme křivku. Po ukončení i druhé skici stačí jen použít funkci Odebrat tažením po křivce a zvolit, co bude tvar a co křivka.



Zákonitosti tvorby v 3D programu

Každý konstruktér má svůj léty prověřený způsob konstruování v 3D programech, proto je těžké popsat naprosto obecný a všude aplikovatelný postup, kterým vždy dojdeme kýženého výsledku. Přesto jsou některé zákonitosti, které se vyplatí dodržovat a ulehčit si tak práci. Tyto zákonitosti lze uplatnit v průběhu celé tvorby finálního modelu, nicméně pro přehlednost si je rozdělíme do třech základních kategorií:

1. Obecné
2. 2D náčrt
3. 3D model

Obecné

Zásady, které se vyplatí dodržovat před zahájením samotného modelování, případně jsou dosti obecné a dají se realizovat v kterékoli fázi modelování.

- Rozmyslet si výsledný model a postup, který k němu povede.
- Promyslet, změřit případně propočítat všechny rozměry.
- Správně vybrat rovinu, na které chceme vytvářet náčrt

2D náčrt

Zásady, které se vyplatí dodržovat při tvorbě 2D náčrtu a jeho ukončení.

- Základní náčrt se snažím kreslit co nejpodobnější výslednému plně určenému náčrtu.
- V případě, že mi nejde kreslit nějaký tvar, můžu si pomoci vazbami.
- Nepoužívat všude jen kóty (je to možná pohodlnější) - práce navíc.
- Vždy končit plně určeným náčrtem.
- Po dokončení náčrtu vždy uzavírat skicu a až poté používat nástroje pro 3D model.

3D model

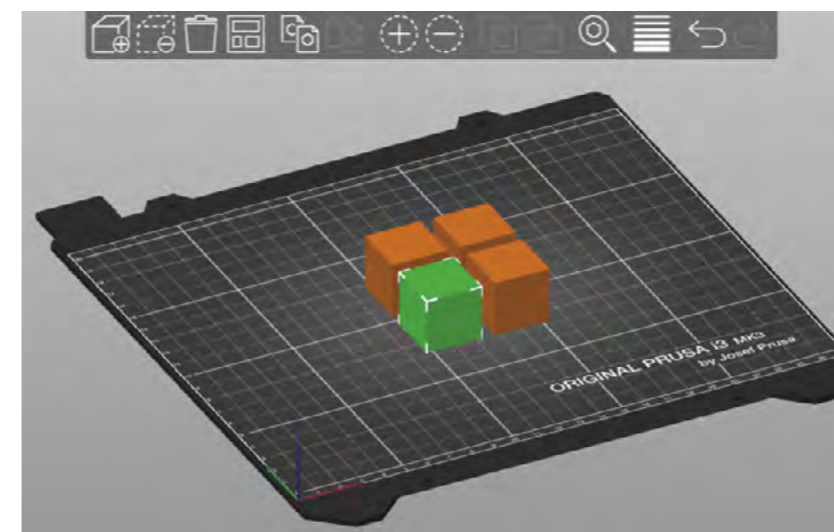
- Volit správné funkce (vysunutí, rotace...) pro převod 2D náčrtu na 3D model.
- Při práci s funkcemi dbát na správné označování náčrtu (jinak nemusí být výsledek správný nebo může nastat chyba).
- Dokončovací úpravy (zaoblení, zkosení atp.) se vždy dělají až před uložením a ukončením modelu.

3D tisk

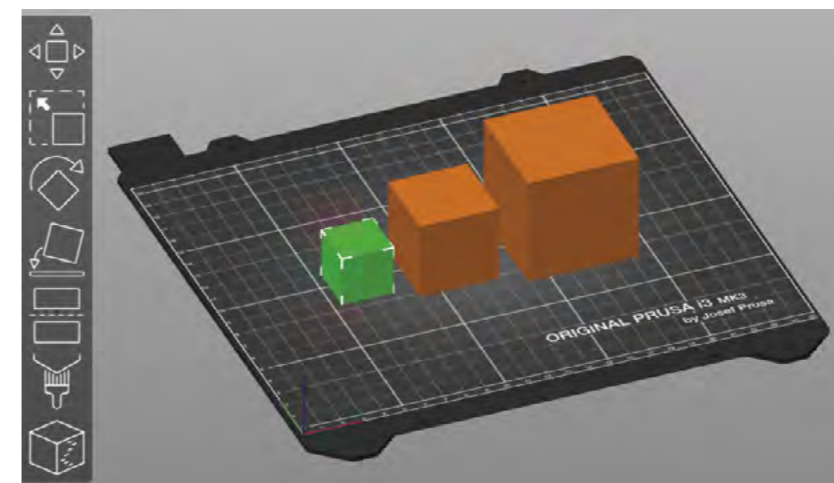
Slicováním je nazýván proces převodu 3D objektu na strojový kód (pokyny, kterým rozumí 3D tiskárna – u FDM nejčastěji GCODE). Slicer (obecný název pro software provádějící slicování) převede model na jednotlivé tahy 3D tiskárny a vše uloží do GCODE, který současně obsahuje veškerá další nastavení (teploty, rychlost, výška vrstvy, apod.).

Volba vhodného sliceru je obvykle dána podporou konkrétní tiskárny (tvorba vlastního profilu tiskárny pro slicer je značně náročná). Pro rozšířenější tiskárny však často existují profily (konfigurace) pro různé slicery. Mezi nejznámější patří Slic3r, Cura, Simplify3D). Některé tiskárny mají slicer jako cloudovou službu a vyžadují tak připojení k internetu. Níže uvedené příklady byly vytvořeny v PrusaSliceru (odvozená verze Slic3ru), který podporuje celou řadu 3D tiskáren i celou řadu funkcí. Výhodou pro výuku je možnost přepnutí mezi režimy jednoduchý/pokročilý/expert a podpora mnoha jazyků včetně češtiny.

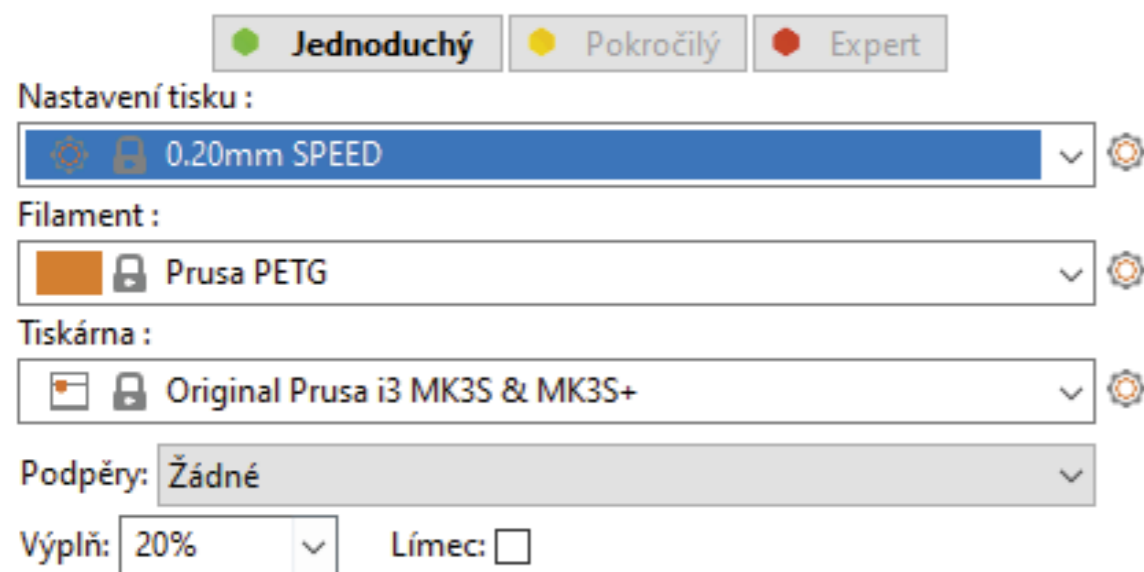
Nejdříve je potřeba do sliceru načíst modely – objekty pro tisk (obvykle soubory ve formátu STL nebo OBJ). Objekt můžeme tisknout i vícekrát (více instancí), nebo můžeme tisknout více různých objektů. Zde nám může velmi pomoci možnost automatického uspořádání objektů na tiskové ploše.



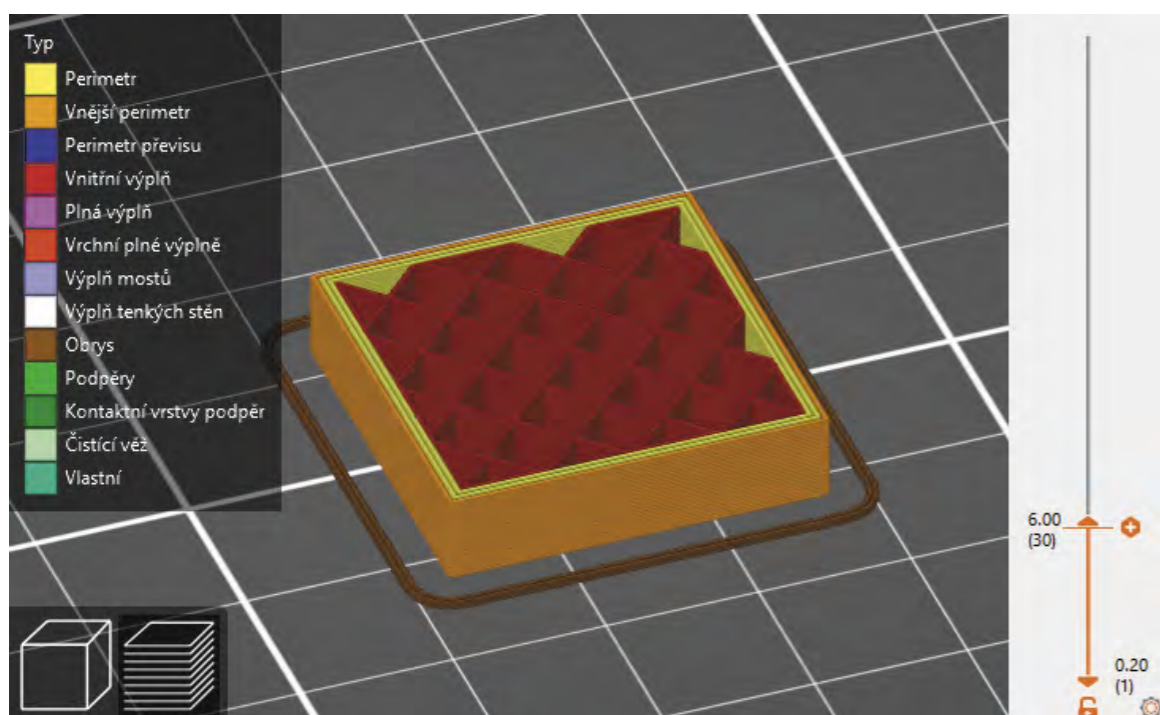
Objekty můžeme různě přemísťovat, měnit velikost, otáčet apod. Důležitá je zejména orientace tištěného dílu, především pak to, kterou plochou hrana dosedá na tiskovou plochu (první vrstva tisku). Více o orientaci v kapitole *Orientace objektu pro tisk*.



Základním nastavením pro každý tisk je volba tiskárny a filamentu. Dále je potřeba zvolit minimálně základní profil pro tiskové nastavení (obsahující především údaje o výšce vrstvy, počtu perimetrů a výplni).



V režimu náhledu lze pomocí posuvníku procházek vrstvy, což je výhodné jak pro náhled dovnitř objektu (výplň), tak pro kontrolu možných potenciačních problémů (především tisku „do vzduchu“). Kliknutím na malé „+“ na posuvníku můžeme také přidat kód pro výměnu filamentu (např. pro změnu barvy) či pozastavení tisku.



Přehled nejdůležitějších nastavení

Nastavení tiskárny	
– Průměr trysky	<i>Nutno změnit v případě výměny trysky. Průměr trysky má však vliv na další vlastnosti při tisku, ideální je tak použít nějaký z již připravených profilů.</i>
Nastavení filamentu	
– Filament	
– Teplota	<i>Nastavení teploty by mělo vždy odpovídat doporučením výrobce konkrétního filamentu. Někdy je však nutno trochu experimentovat. Např. vyšší teploty vedou obvykle k pevnějšímu spojení vrstev, ale naopak při tisku převisů a mostů může dojít k větším defektům.</i>
– Násobič extruze	<i>Ovlivňuje množství vytlačovaného filamentu. Někdy může být žádoucí toto množství zvýšit (zejména pokud má filament menší průměr než je obvyklý).</i>
– Chlazení	<i>Nastavení chlazení (tiskového ventilátoru) je důležité pro správný tisk většiny filamentů. Pomáhá předcházet nežádoucím efektům při tisku. U některých filamentů s větší tepelnou roztažností však může způsobovat deformace a odlepování již v průběhu tisku.</i>
Nastavení tisku	
– Vrstvy a perimetry	
– Perimetry	<i>Počet obvodových vrstev zásadně ovlivňuje pevnost výsledného dílu. Pro výslednou pevnost je obvykle důležitější počet perimetrů než hustota výplně.</i>
– Detekovat perimetry přemostění (pokročilý režim)	<i>Mění parametry pro tisk mostů a převisů (obvykle zlepšuje jejich tisk – zmírňuje možné defekty).</i>
– Vyhnout se přejíždění perimetrů (expertní režim)	<i>Zamezí pohybu trysky mimo tištěné objekty (přejíždění), mimo nezbytných. Používá se především k zamezení vzniku strunek (tenkých vláček mezi částmi objektu).</i>
– Výplň	<i>Lze nastavit různé vzory výplně pro obsah a pro horní a spodní vrstvu a její hustotu. Nastavení výplně zásadně ovlivňuje mechanické vlastnosti výsledného tištěného dílu. Výplně slouží také k podpoře horních vrstev a zásadně ovlivňuje délku tisku, vhodné nastavení je tak velmi důležité.</i>
– Obrys a límeč	
– Šířka límce	<i>Límeč slouží k rozšíření základny objektu u první vrstvy. Napomáhá dobrému přilnutí objektu k podložce zejména u objektů s malou základnou (kontaktní plochou) a předchází odlepování rohů. Obvykle se používá nastavení 2–5 mm.</i>
– Podpěry	
– Generovat podpěry	<i>Zapnutí podpěr</i>
– Automaticky generované podpěry	<i>Slicer automaticky přidá podpěry tam, kde to považuje za nutné (v pokročilých režimech možno upřesnit).</i>
– Pouze na tiskové podložce	<i>Budou použity pouze podpěry, které začínají na tiskové podložce, nikoliv na samotném modelu.</i>

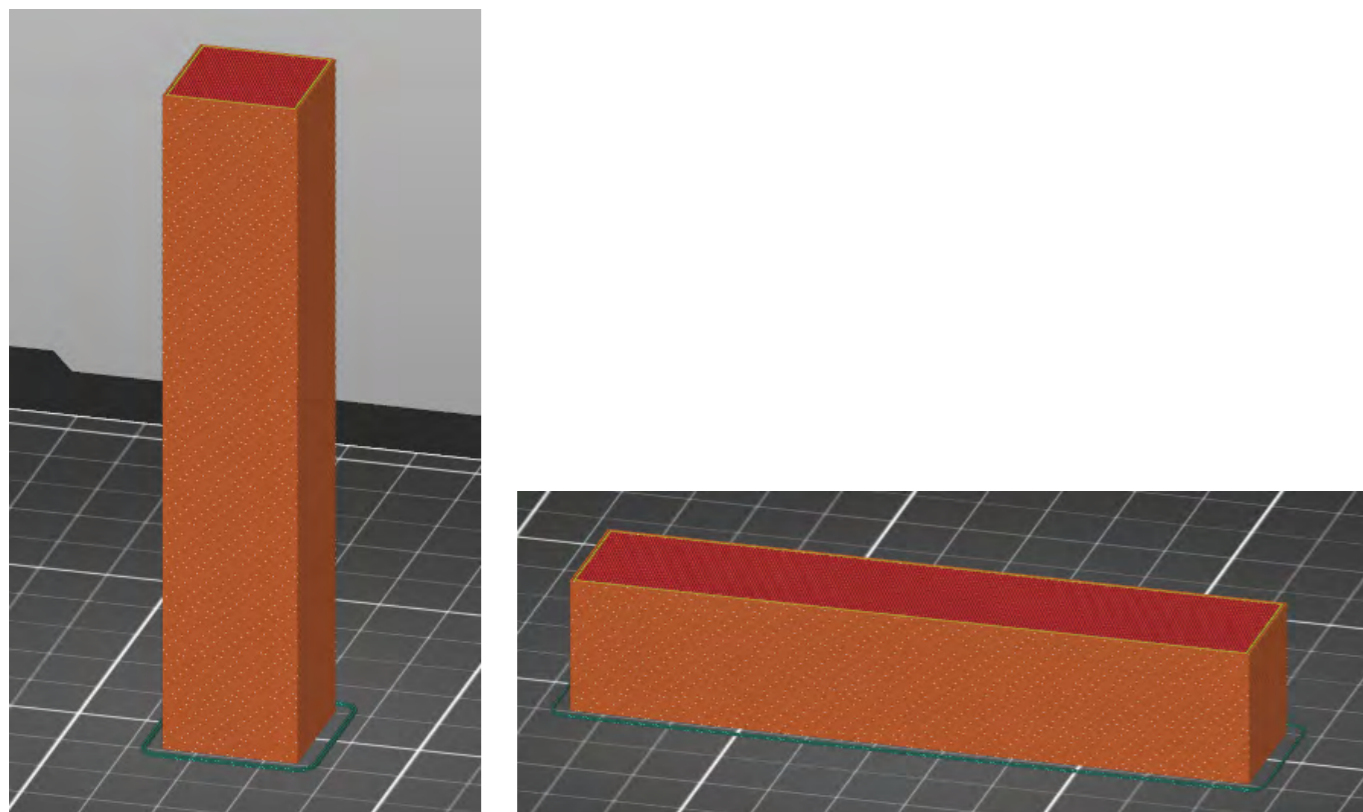
PrusaSlicer obsahuje celou řadu pokročilých funkcí („malované“ podpěry, variabilní výšku vrstvy, vícemateriálový tisk apod.). Přestože řada z těchto funkcí je velmi užitečná, jejich zvládnutí je časově náročné a vyžaduje velmi dobrou znalost základních nastavení pro slicování a často zkušeností s tiskem. Jejich zahrnutí do výuky je tak čistě na zvážení konkrétním vyučujícím s ohledem na náročnost a časovou dotaci.

Orientace objektu pro tisk

Orientace objektu pro 3D tisk zásadně ovlivňuje vlastnosti výsledného objektu. Asi nejzásadnější vlastností je, zda jsou pro tisk nutné podpory. Orientace objektu ovlivňuje především tyto faktory:

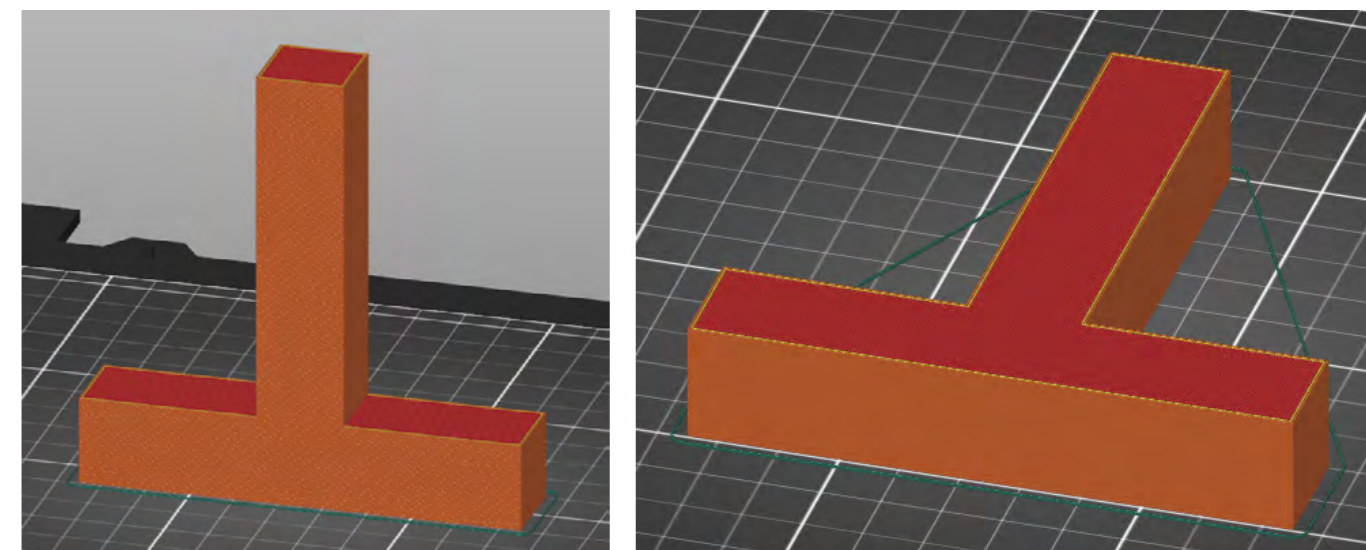
- Nejslabší vazby v objektu bývají mezi vrstvami. Pokud objekt „praskne“ v důsledku mechanického namáhání, stane se tak obvykle mezi vrstvami.
- Převisy je možné obvykle bezpečně tisknout do 45° (nebo o něco více při nižší výšce vrstvy v poměru k průměru tryska). Mosty lze tisknout na kratší vzdálenosti (v závislosti na druhu filamentu).
- Kontaktní plocha s podložkou (1. vrstva) je zásadní pro úspěšnosti tisku.

Jak tisknou následující model kvádrů?



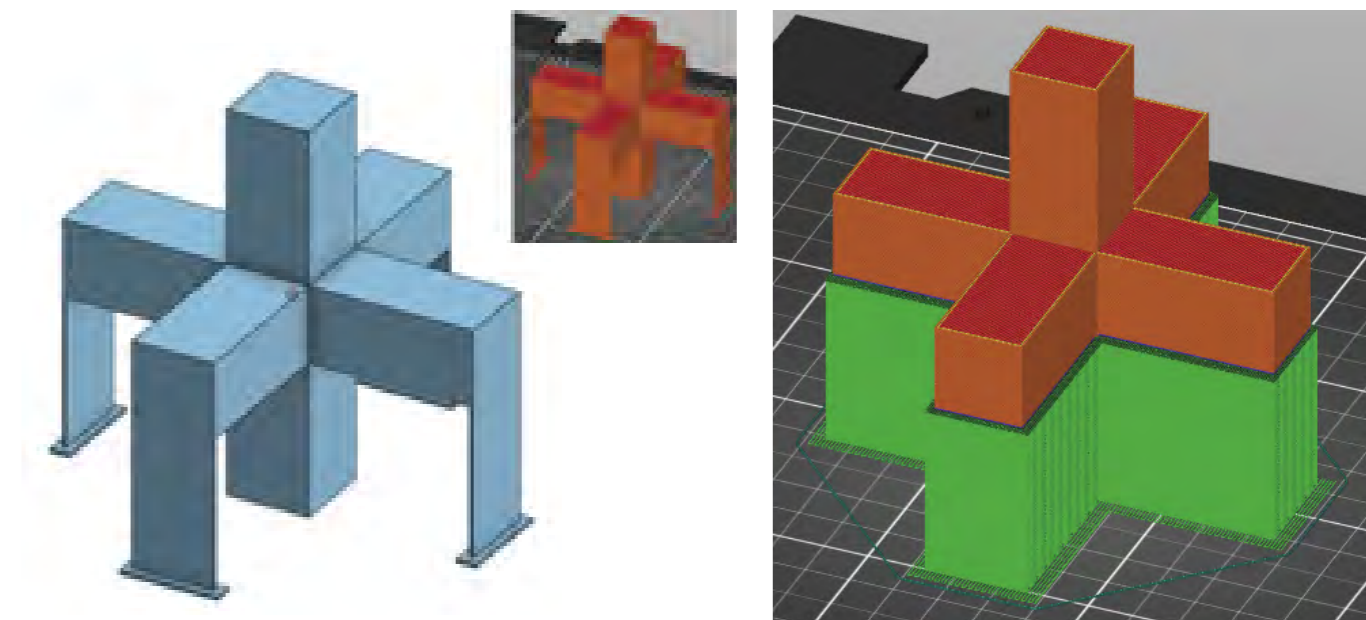
Vzhledem k jednoduchosti jeho tvaru je možné vytisknout kvádr položený na kteroukoli z jeho stran. Nicméně pokud bude tištěn „na výšku“, jeho kontaktní plocha s podložkou bude poměrně malá (hrozí odlepení především ke konci tisku) a současně bude poměrně snadné tento model kvádrů zlomit (vzhledem k orientaci vrstev). Vhodnější je orientace na jednu ze stran s větší plochou.

Jak tisknou následující model ve tvaru „T“?



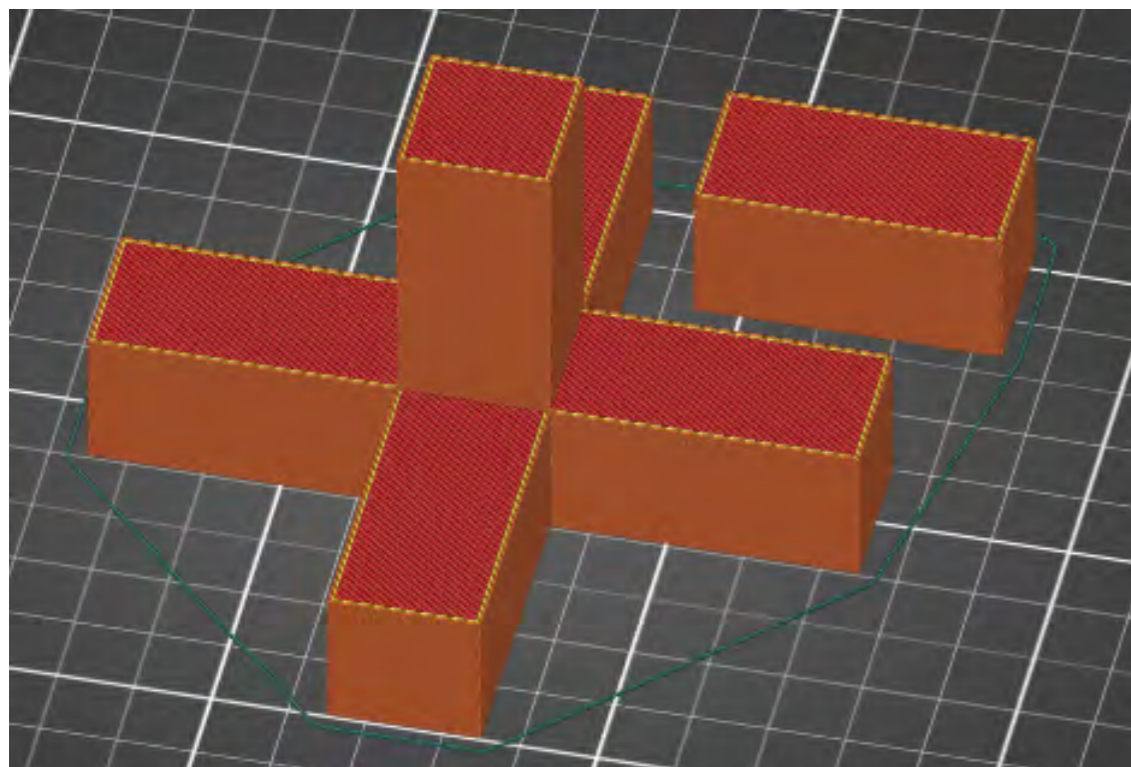
Jedná se o podobnou situaci jako u předchozího příkladu. Přestože základna je v tomto případě při orientaci „na výšku“ větší, s ohledem na pevnost výsledného dílu je vhodnější orientace „na ležato“.

Jak tisknou následující model?



Objekt nelze orientovat tak, aby bylo možné ho vytisknout bez nutnosti podpor. Vzhledem k jednoduchým rovným plochám lze však použít jak automaticky generované podpory ve sliceru, tak si vymodelovat podpory vlastní (které šetří filament a snadněji se oddělují). Využito je zde přitom faktu, že mosty tisknout lze (na rozdíl od 90° převisů).

Alternativně je také možné model rozdělit a vytisknout na 2 části, které se následně slepí či sešroubují (při patřičné úpravě modelu). Na možnost rozdělit model pro tisk na více částí se často zapomíná.



Základní údržba 3D tiskárny

Jako každý stroj i 3D tiskárna se musí udržovat, aby fungovala správně, tisk probíhal kvalitně a zároveň se prodloužila délka její životnosti.

Tiskárna

Tiskárna by neměla být uskladněna a používána v prašném prostředí z důvodů špinění podložky, případně materiálu v průběhu tisku a větráku motůrku. Po prvním spuštění je potřeba tiskárnu zkalibrovat, stejně tak ji musíme znovu zkalibrovat po např. nějakém větším stěhování, převážení atp. Samotnou kalibraci nás provází krok za krokem 3D tiskárna. Můžeme promazávat řemeny, které posouvají tryskou.

Podložka

Je zakázáno na podložku sahat prsty. Mastí se a tisk nemusí k podložce přiléhat. Podložku před tiskem vyčistíme od mastnoty a nečistot (zbytky předešlého tisku, nečistoty atp.) čistíme pomocí například technického lihu, poté máme jistotu, že tisk bude dobře přiléhat k podložce.

Doporučené pomůcky

V první řadě je potřeba, aby každý žák měl svůj počítač s funkčním programem SolidWorks, případně počítač s internetem, kde se bude moci spustit v prohlížeči program OnShape. Je vhodné mít 3D tiskárnu ve stejné třídě, kde výuka/kroužek probíhá.

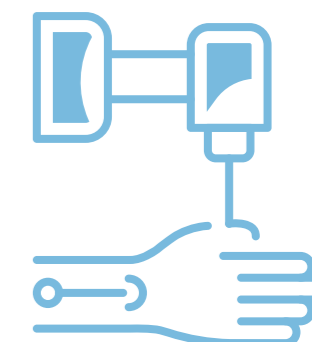
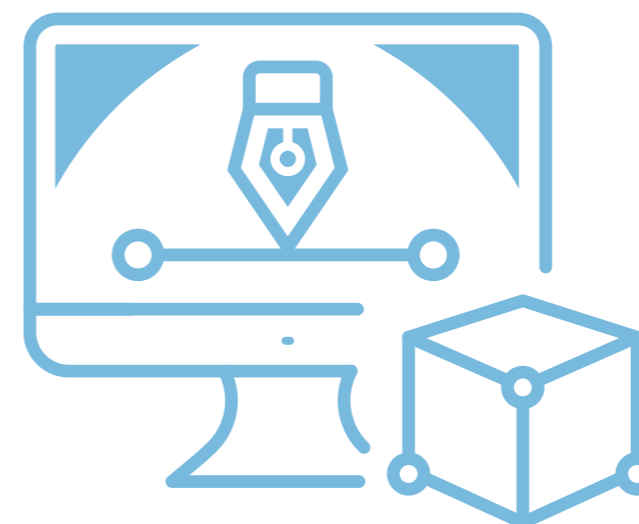
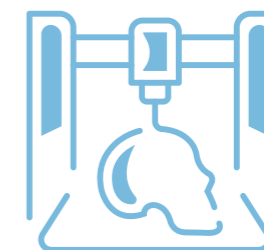
V ideálním případě by žáci měli mít stejnou verzi programu jako má učitel/lektor.

V rámci představivosti a použití rozměrů je vhodné mít k dispozici některé z měřidel (skládací metr, posuvné měřítko, plastové pravítko aj.).

Pracovní list

Přílohou tohoto materiálu jsou pracovní listy. Tyto pracovní listy jsou k dispozici v editovatelné elektronické formě, aby si je každý učitel mohl upravit, např. pokud bude chtít změnit rozměry, či něco přidat.

Pracovní listy jsou čtyři. V pracovních listech počítám s tím, že škola má k dispozici licenci na SolidWorks, případně využívá program OnShape v internetovém prohlížeči. Pokud má k dispozici pouze některý z free neparametrických programů, je nutné tyto listy upravit, zejména postup, použité funkce a nástroje. Rozměry a posloupnost úkolů lze ponechat.



Pracovní list 1

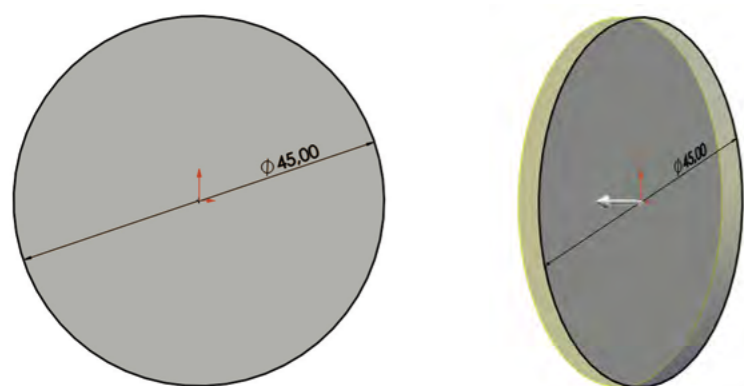
Jednoduchý přívěšek emotikon

Co budeme potřebovat

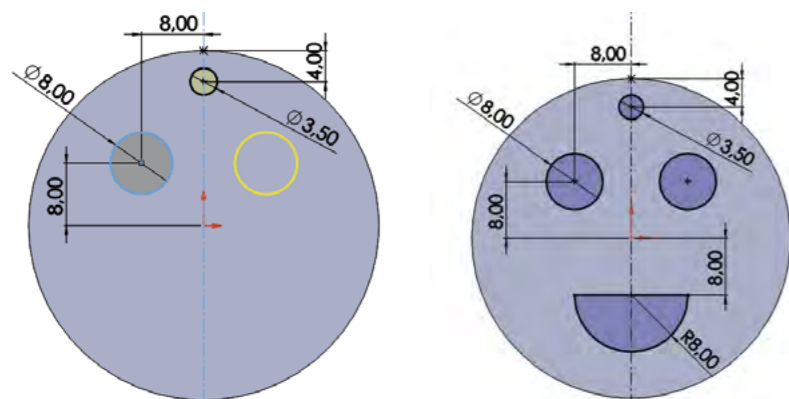
- Počítač s nainstalovaným CAD programem SolidWorks a programem PrusaSlicer.
- Případně počítač s internetem - OnShape.
- 3D tiskárnu Prusa I3 MK3S+.

Ajdeme na to

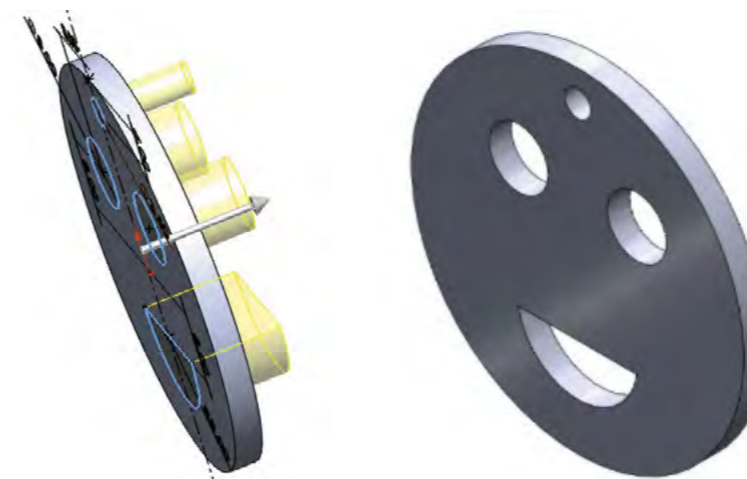
Po spuštění programu a otevření skici si ze středového bodu nakreslíme kružnici. Této kružnici pomocí nástroje Inteligentní kóta dáme rozměr 45 mm, čímž získáme kompletní 2D základní náčrt. Tento náčrt si pomocí funkce Přidat vysunutím vysuneme o 3 mm a tím vytvoříme 3D model. Na vzniklé ploše (kolečku) si otevřeme novou skicu a rozdělíme plochu přesně ve středu vertikální osou.



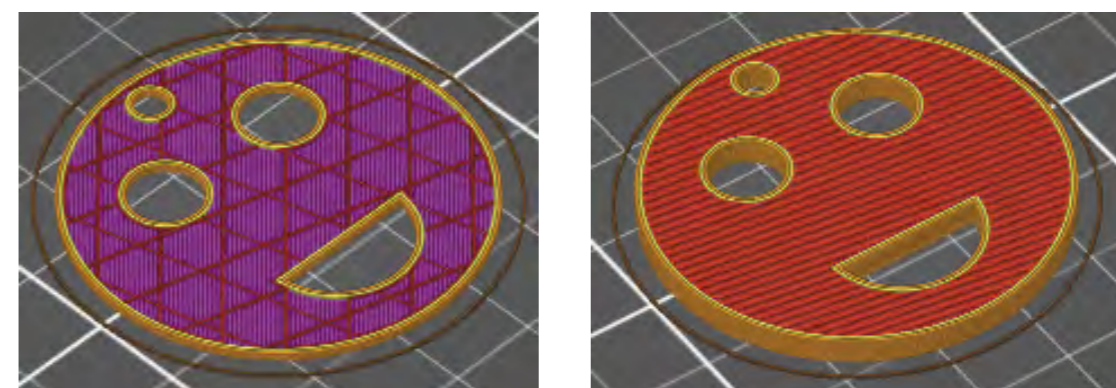
V horní části osy vytvoříme kružnici a dáme jí rozměr 3,5 mm, poté na pravé straně od osy narýsuje kružnici a dáme jí rozměr 8 mm. Tuto kružnici ozrcadlíme na druhou stranu. Zrcadlení provedeme tak, že pomocí levého tlačítka myši a klávesy CTRL vybereme jak kružnici, tak osu (zrcadlení). Pokud je vše správně, tak ještě před potvrzením se objeví žlutý náhled kružnice. Dále vytvoříme oblouk, který spojíme přímkou - dostaneme půlkruh.



V poslední fázi modelování ukončíme skicu a vybereme všechny tři kružnice i půlkruh a pomocí funkce Odebrat vysunutím odebereme Skrz vše. Finální výrobek uložíme ve formátu STL, který nám zaručuje možnost slicování a 3D tisku.



V rámci slicování si výrobek virtuálně položíme na tiskový plán 3D tiskárny a můžeme upravit strukturu a výplň tisku (jednoduché výrobky se z pravidla netisknou na maximální výplň). Při takto jednoduchých modelech bývá zvykem vnitřek modelu tisknout žebrovaně. Dále získáme z programu informace o délce tisku. Po úspěšném slicování nastává samotný tisk - dodržujte bezpečnostní požadavky na 3D tiskárny.



Poznámky a doporučení

- Na začátku kreslení vždy volíme přední rovinu, model se nám poté lépe zobrazuje v dalších pohledech.
- Kromě kót existují i vazby, nebojte se je používat.
- Před dokončením 3D náčrtu (uzavřením skici) mějte vždy náčrt plně určený - všechny kreslené čáry a objekty jsou černé (určují je rozměry nebo vazby).
- Zvolený postup je nejrychlejší a do jisté míry nejjednodušší - lze ovšem použít i jiné postupy.

Co jste se naučili?

Vytvořili jsme skicu, na které se vytváří 2D náčrt a pomocí základních nástrojů samotný 2D náčrt. Pomocí funkcí jsme vytvořili z 3D náčrtu 3D model jednoduchého přívěšku - emotikon.

Pracovní list 2

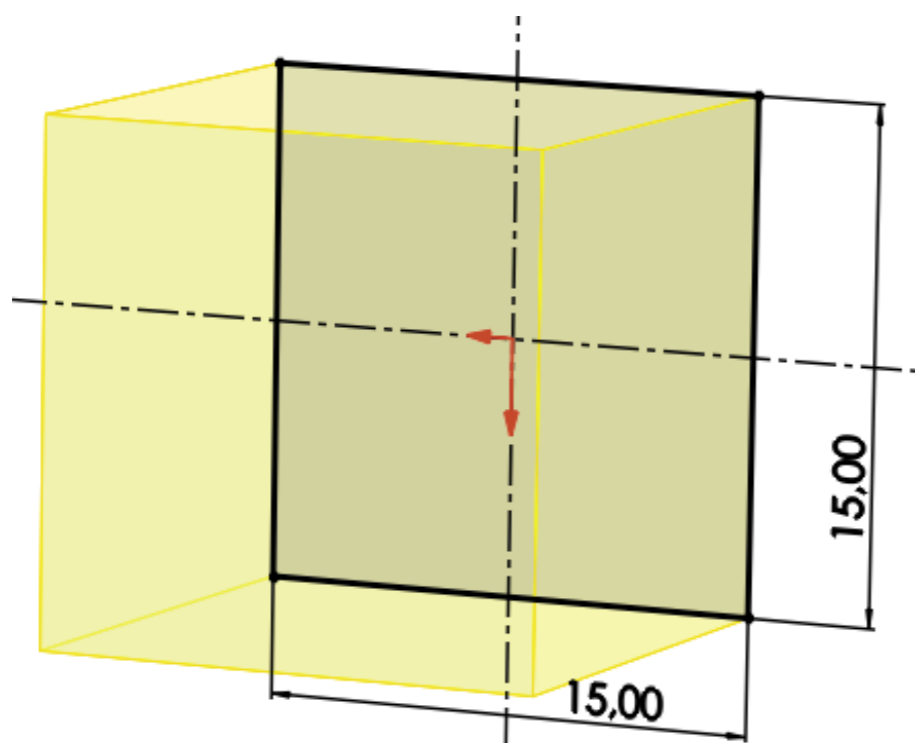
Hrací kostka

Co budeme potřebovat

- Počítač s nainstalovaným CAD programem SolidWorks a programem PrusaSlicer.
- Případně počítač s internetem - OnShape.
- 3D tiskárnu Prusa I3 MK3S+.

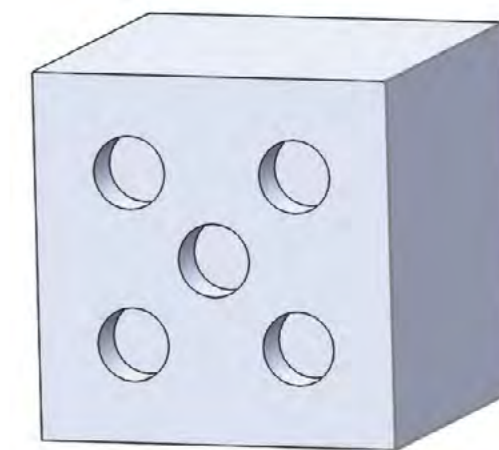
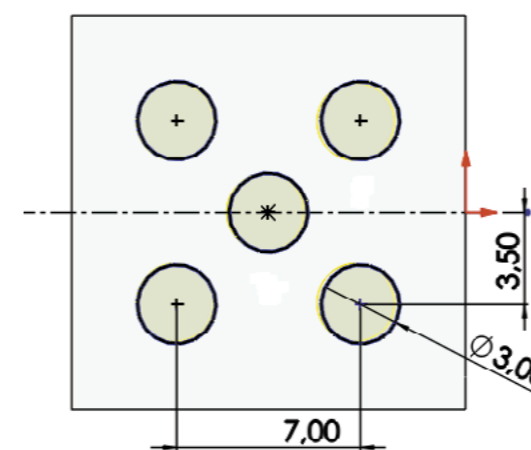
Ajdeme na to

Po spuštění programu a otevření skici si připravíme osy x a y. Poté narýsuje čtverec o délce strany 15 mm, tento čtverec zavazbíme k osám. Vysuneme nakreslený čtverec do 3D podoby, tloušťka vysunutí bude 15 mm (aby vznikla krychle).

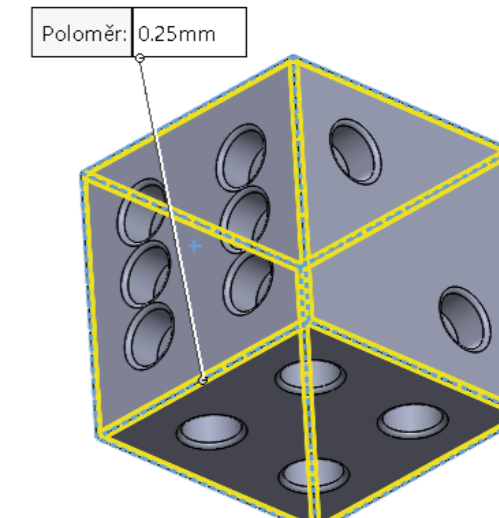
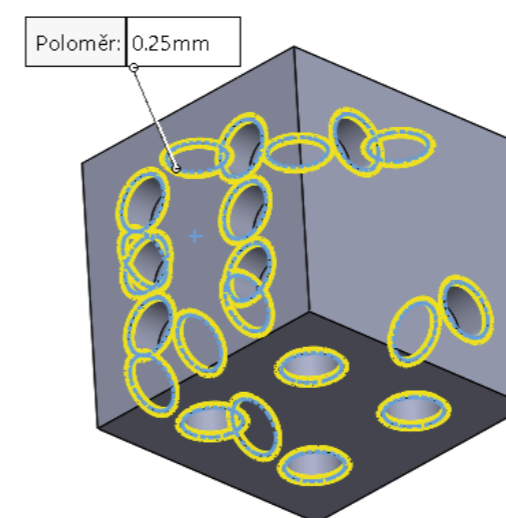


Zvolíme jednu stranu krychle a otevřeme na ni skicu. Na této skice nakreslíme kružnici o průměru 3 mm, jejíž střed bude vzdálený od obou os 3,5 mm. Pomocí nástroje Lineární pole si kružnici „nakopírujeme“ tak, abychom měli čtyři kružnice (viz obrázek). Vzdálenost mezi kružnicemi je 7 mm. Do průsečíku os nakreslíme stejnou kružnici.

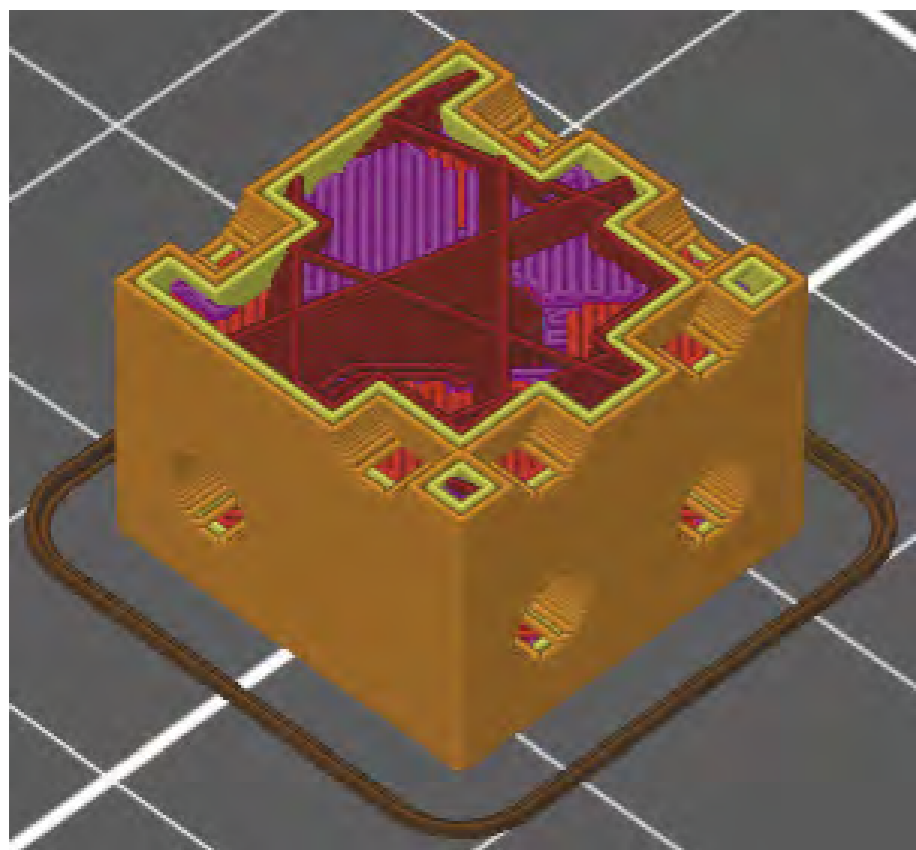
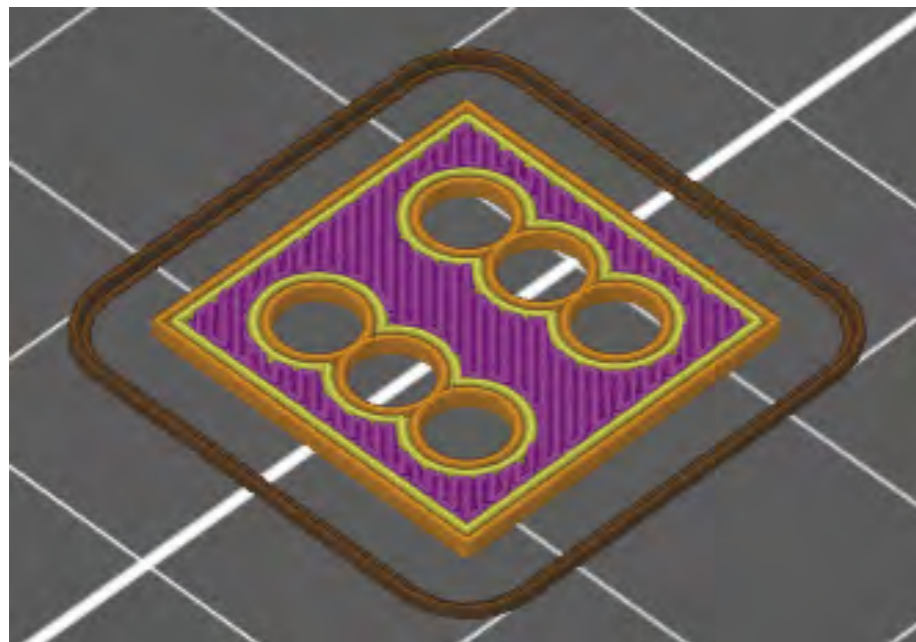
Pomocí funkce Vysunutí odebereme materiál ze všech kružnic, hloubka odebrání je 2 mm, a získáme první stranu hrací kostky. Obdobným způsobem lze vytvořit i ostatní strany hrací kostky.



Po dokončení všech stran použijeme dokončovací nástroj Zaoblit/Zkosit a zaoblíme hrany hrací kostky dle uvážení od 0,25 mm do 1 mm, čímž docílíme snadnější otáčení kostky při hodu. Dále zaoblíme všechny otvory. Po dokončení uložíme model ve formátu STL, který nám zaručuje možnost slicování a 3D tisku.



V rámci slicování si výrobek **virtuálně** položíme na tiskový plán 3D tiskárny a můžeme upravit strukturu a výplň tisku (jednoduché výrobky se z pravidla netisknou na maximální výplň). Při takto jednoduchých modelech bývá zvykem vnitřek modelu tisknout žebrovaně. Dále získáme z programu informace o délce tisku. Po úspěšném slicování nastává samotný tisk – dodržujte bezpečnostní požadavky na 3D tiskárny.



Poznámky a doporučení

- Na začátku kreslení vždy volíme přední rovinu, model se nám poté lépe zobrazuje v dalších pohledech.
- Kromě kót existují i vazby, nebojte se je používat.
- Před dokončením 3D náčrtu (uzavřením skici) mějte vždy náčrt plně určený – všechny kreslené čáry a objekty jsou černé (určují je rozměry nebo vazby).
- Zvolený postup je nejjednodušší a do jisté míry nejrychlejší – lze ovšem použít i jiné postupy.
- Je dobré dopředu připomenout, jak jsou číslice na hrací kostce umístěny.

Co jste se naučili?

Procvičili jsme funkci Vysunutí a zároveň vytvořili jednoduchou hrací kostku pro kteroukoli stolní hru. Nově jsme si ukázali, jak po dokončení model ještě upravit pomocí nástrojů Zaoblit/Zkosit a tím docílit jeho realističtější a funkčnější podoby.



Pracovní list 3

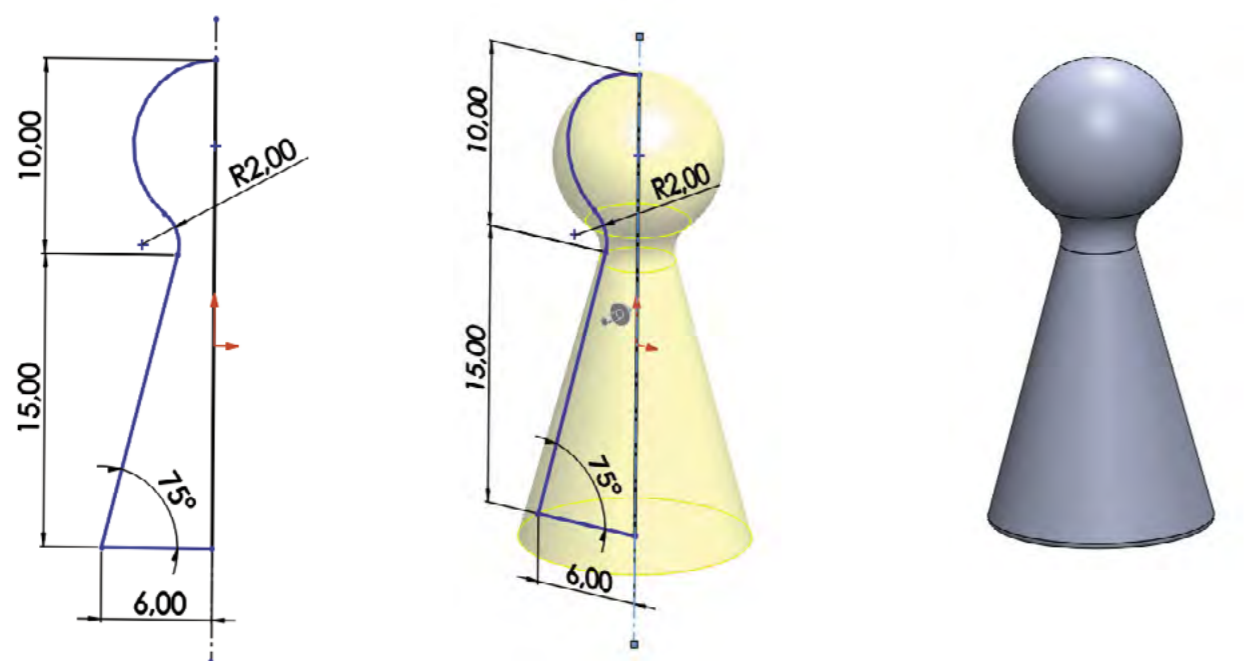
Hrací figurka

Co budeme potřebovat

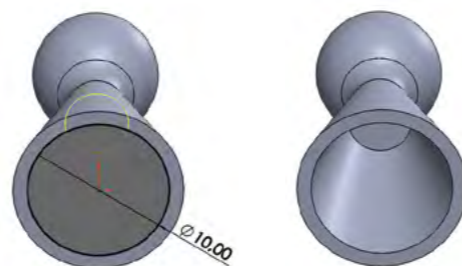
- Počítač s nainstalovaným CAD programem SolidWorks a programem PrusaSlicer.
- Případně počítač s internetem - OnShape.
- 3D tiskárnu Prusa I3 MK3S+.

Ajdeme na to

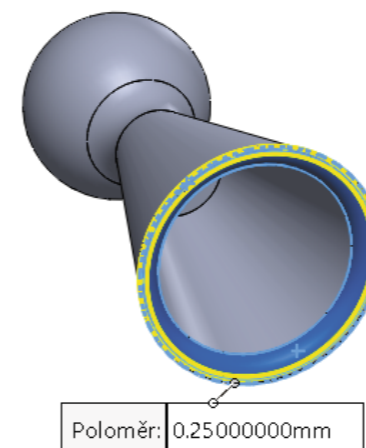
Po spuštění programu a otevření skici si připravíme vertikální osu, která bude procházet středem. Poté nakreslíme pomocí jednoduchých tvarů polovinu hrací figurky, polovina prochází středem (vertikální osou). Pomocí vazeb a kót získáme plně určený náčrt. Celková výška hrací figurky je 25 mm. Po dokončení náčrtu a pomocí funkce Přidání rotací otortujeme náčrt hrací figurky kolem vertikální osy a vznikne nám celá hrací figurka.



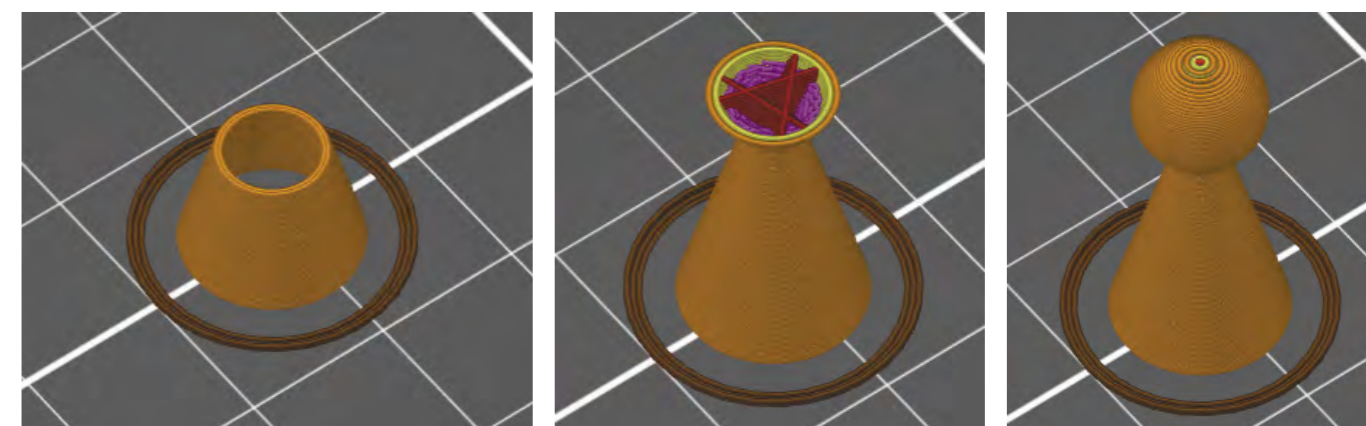
Pro úsporu materiálu i času při tisku na 3D tiskárně si zespodu na podstavu figurky nakreslíme kružnici o průměru 10 mm. Kružnici odebereme pomocí vysunutí kuželovitě, hloubka odebrání bude 12 mm.



Na závěr pomocí nástroje Zaoblit/Zkosit zaoblíme spodní a vnitřní hrany figurky o 0,25 mm. Hrací figurku poté uložíme ve formátu STL, který nám zaručuje možnost slicování a 3D tisku.



V rámci slicování si výrobek **virtuálně** položíme na tiskový plán 3D tiskárny a můžeme upravit strukturu a výplň tisku (jednoduché výrobky se z pravidla netisknou na maximální výplň). Dále získáme z programu informace o délce tisku. Po úspěšném slicování nastává samotný tisk - dodržujte bezpečnostní požadavky na 3D tiskárny.



Poznámky a doporučení

- Na začátku kreslení vždy volíme přední rovinu, model se nám poté lépe zobrazuje v dalších pohledech.
- Kromě kót existují i vazby, nebojte se je používat.
- Před dokončením 3D náčrtu (uzavřením skici) mějte vždy náčrt plně určený - všechny kreslené čáry a objekty jsou černé (určují je rozměry nebo vazby).
- Zvolený postup je nejrychlejší a dá se říci nejjednodušší - lze ovšem použít i jiné postupy.
- Figurka se dá pro lepší rozlišení doplnit např. o obličej, růžky, ocásek atp.

Co jste se naučili?

Naučili jsme se používat funkci Rotace a zároveň jsme vytvořili vlastní hrací figurku, kterou lze použít v kterékoli stolní hře. Zároveň jsme si ukázali, jak můžeme v případě potřeby upravit funkci Vysunutí - do kuželu atp.

Pracovní list 4

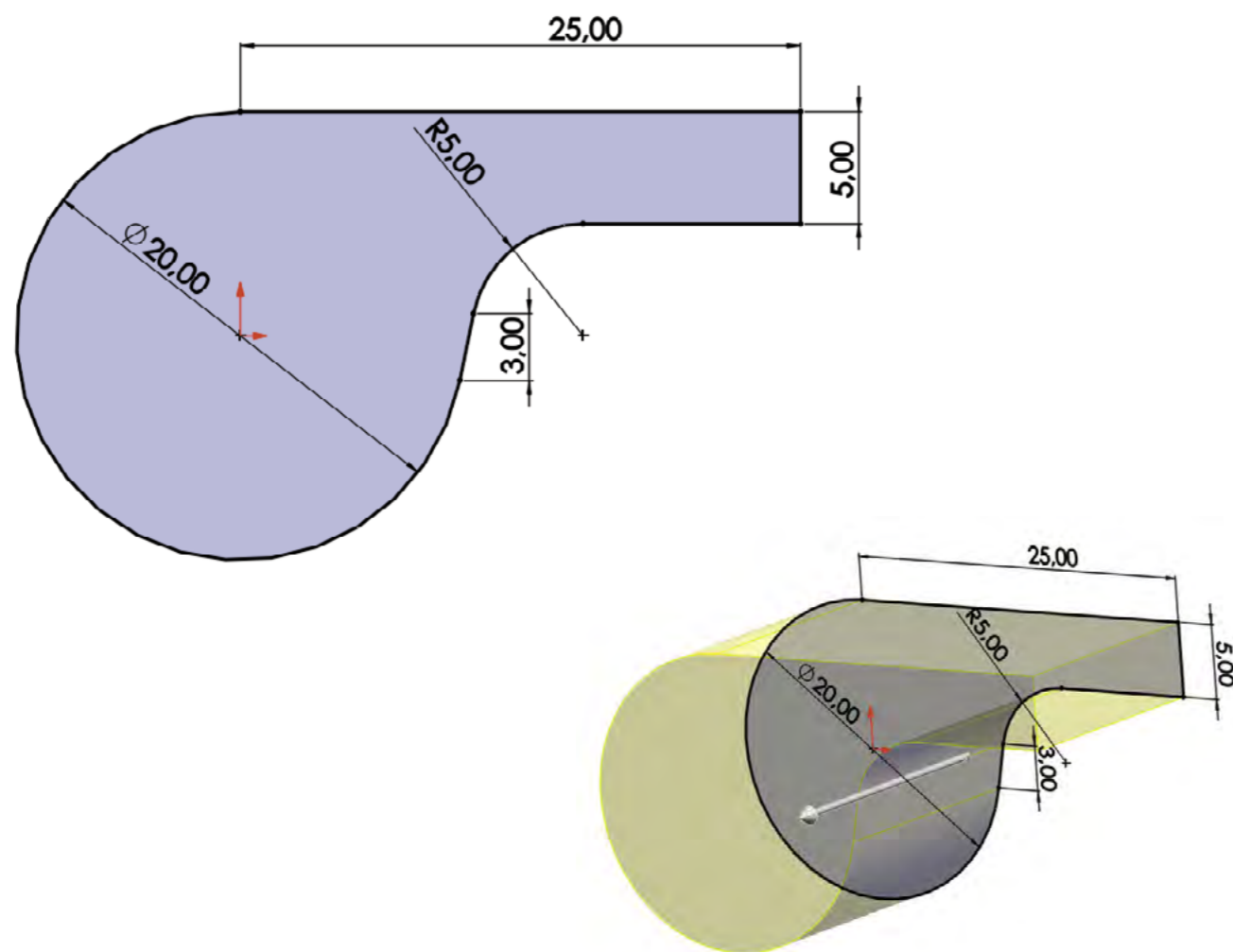
Píšťalka

Co budeme potřebovat

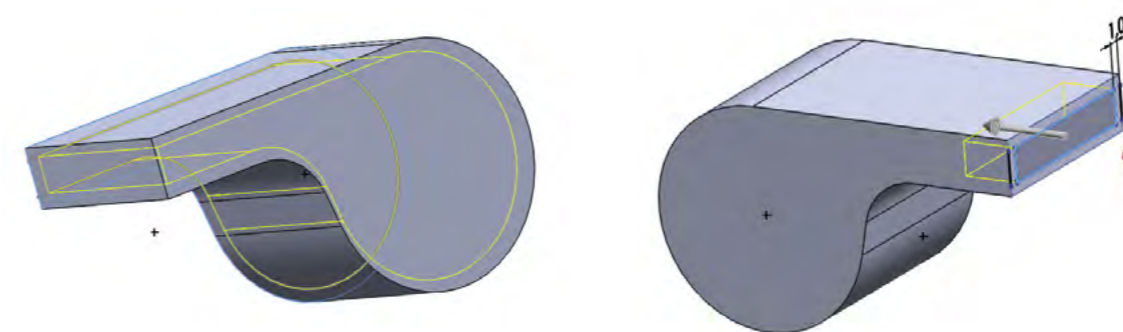
- Počítač s nainstalovaným CAD programem SolidWorks a programem PrusaSlicer.
- Případně počítač s internetem - OnShape.
- 3D tiskárnu Prusa I3 MK3S+.

Ajdeme na to

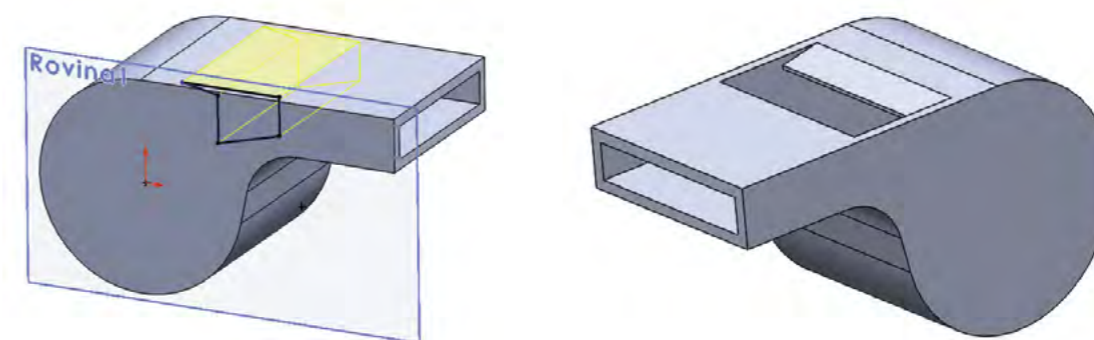
Po spuštění programu a otevření skici si ve středu nakreslíme kružnici o průměru 20 mm. Poté pomocí čar dokreslíme, zavazbíme a okótujeme i zbytek náčrtu píšťalky viz obrázek. Takto vzniklý plně určený náčrt pomocí funkce Vysunutí vysuneme o 17 mm.



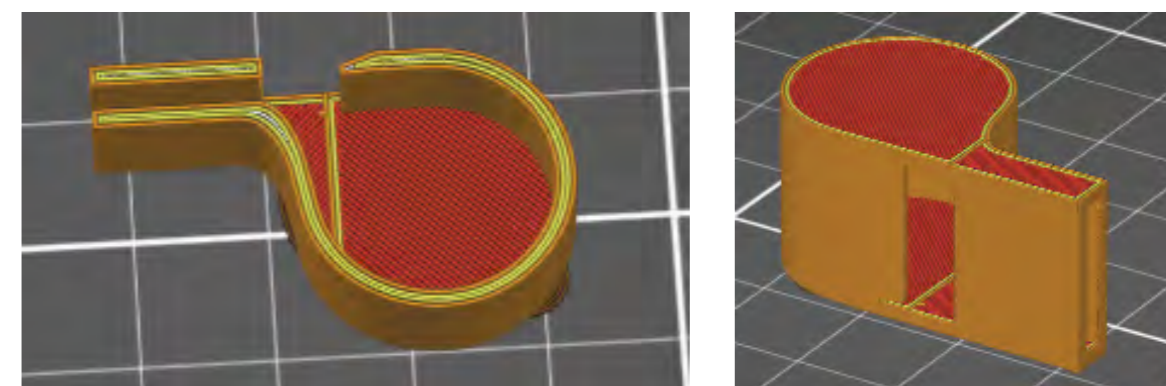
Máme základní model, ze kterého vznikne píšťalka. Pomocí Roviny (je odsazena 1 mm od plochy směrem dovnitř) si uděláme na přední stranu skicu a pomocí nástroje Odsadit entity odsadíme celý náčrt o 1 mm do středu (zmenšíme o 1 mm). Ukončíme skicu a na takto připraveném novém náčrtu vytvoříme skořepinu (využijeme funkci Skořepina), čímž získáme dutý model. Poté si na boční části modelu vytvoříme skicu a načrtne otvor pro vstup vzduchu do píšťalky (1 mm od každé strany). Nakonec pomocí odebrání vysunutím otvor vytvoříme.



Poslední částí je druhý otvor shora píšťalky. Pomocí nástroje Rovina si umístíme rovinu pro skicu (je umístěna 16 mm od Přední roviny). Otevřeme skicu a načrtne náčrt druhého otvoru (viz obrázek). Tento náčrt odebereme vysunutím o 15 mm. Hotový model píšťalky poté uložíme ve formátu STL, který nám zaručuje možnost slicování a 3D tisku.



V rámci slicování si výrobek **virtuálně** položíme na tiskový plán 3D tiskárny a můžeme upravit strukturu a výplň tisku (jednoduché výrobky se z pravidla netisknou na maximální výplň). Dále získáme z programu informace o délce tisku. Po úspěšném slicování nastává samotný tisk - dodržujte bezpečnostní požadavky na 3D tiskárny.

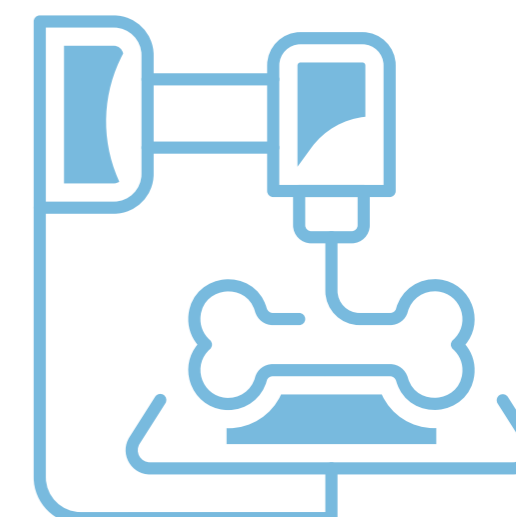
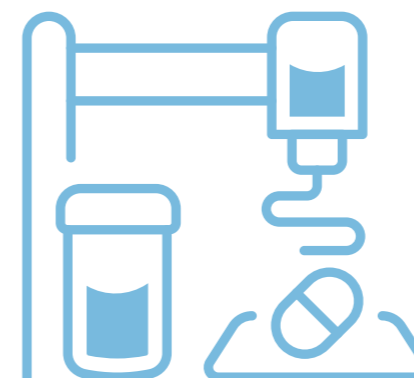
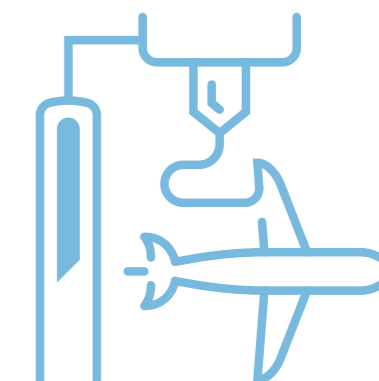
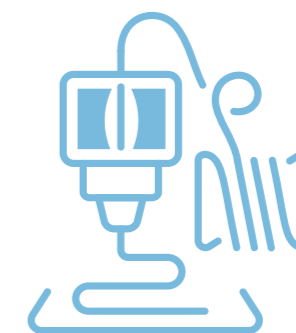


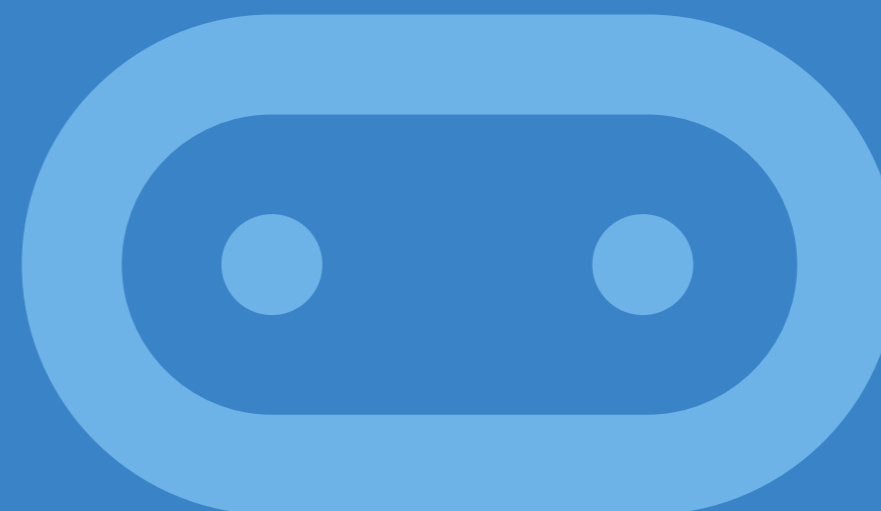
Poznámky a doporučení

- Tento výrobek je mnohem složitější než předchozí, a proto je dobré nechat jej až ke konci předmětu/kroužku/kurzu.
- Na začátku kreslení vždy volíme přední rovinu, model se nám poté lépe zobrazuje v dalších pohledech.
- Kromě kót existují i vazby, nebojte se je používat.
- Před dokončením 3D náčrtu (uzavřením skici) mějte vždy náčrt plně určený – všechny kreslené čáry a objekty jsou černé (určují je rozměry nebo vazby).
- Zvolený postup je nejrychlejší a dá se říci nejjednodušší – lze ovšem použít i jiné postupy.

Co jste se naučili?

Naučili jsme se používat pokročilejší funkci, a to funkci *Skořepina*, která nám pomáhá s vytvářením dutých modelů. Procvičili jsme si používání rovin a funkci *Vysunutí* (přidání i odebrání). Výsledkem je praktický výrobek – *píšťalka*, který potěší každé dítě.





PRACOVNÍ POSTUPY PRO STŘEDNÍ ŠKOLY

PROGRAMOVÁNÍ MICRO:BIT PRO SŠ

Základní instrukce

Tento kurz je doporučován pro žáky prvního nebo druhého ročníku středních škol anebo vyšší ročníky osmiletých gymnázií. Je doporučen zejména pro studenty technických škol a gymnázií. Zejména u technických škol je dobré počítat ještě s nějakou složitější platformou, např. Arduinem.

Časová dotace tohoto kurzu nemůže být zcela jednoznačná. Závisí na tom, zda se studenti již seznámili s programováním a zda již mají nějaké zkušenosti s Micro:bitem. Učitel by měl nejprve prostudovat všechny materiály připravené k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.



Cílem workshopu je představit vývojovou destičku BBC Micro:bit primárně určenou pro výuku programování na základní i střední škole. V našem kurzu se zaměříme na programovací jazyk MicroPython jako podmnožinu jazyka Python. Začneme od úplných základů, ale postupně se přeneseme až k možnosti komunikace dvou Micro:bitů a ukážeme si práci ve dvojici anebo i ve větší skupině. Finálním produktem by mělo být vytvoření zabezpečovacího zařízení tvořeného dvěma Micro:bity, z nichž jeden má funkci sensoru a druhý hlásiče.

Doporučený průběh a časovou dotaci naleznete níže. Kurz lze rozčlenit do čtyř částí:

- 1. Práce s výstupy** – výstup na displej a přehrávání zvuku.
 - a. Pokud již proběhl nějaký kurz Micro:bitu, je možné tuto část přeskočit nebo minimalizovat.
 - b. Pokud žáci již měli nějaké jiné grafické programování např. Scratch, pak postačí jedna vyučovací hodina (45 minut)
 - c. Pokud žáci ještě neměli programování a neznají Micro:bit, pak doporučuji dvě vyučovací hodiny.
- 2. Práce se vstupy** – stisk tlačítka, detekce pohybu, magnetického pole, zvuku, světla, měření teploty.
 - a. Pokud žáci znají Micro:bit a programovali, postačí jedna vyučovací hodina. Pozor, pokud znají Microbit V1, u nového typu jsou některé věci navíc.
 - b. Ve všech ostatních případech doporučuji dvě vyučovací hodiny. Je nutné probrat podmíněné příkazy.
- 3. Komunikace mezi dvěma Micro:bity** – posílání zpráv mezi dvěma Micro:bity pomocí vestavěného radio modulu. Bezpečnost této komunikace.
 - a. Pokud s tímto žáci již pracovali, je možné tuto pasáž zcela přeskočit.
 - b. Jinak doporučuji jednu vyučovací hodinu.
- 4. Samotná tvorba zabezpečovacího zařízení** je už pro všechny stejná. Doporučuji nejméně dvě vyučovací hodiny. V první hodině učitel zvolí jednu z možností zabezpečení a společně s žáky jej zkonstruuje. Ve druhé a případně dalších hodinách žáci samostatně pracují na svých projektech. Je zde možnost přidání jedné nebo více hodin, kdy žáci budou své projekty představovat ostatním.

Minimální doba, za kterou lze tento výukový balík absolvovat, jsou tedy dvě vyučovací hodiny – první hodina: části 1 až 3, druhá hodina: část 4. Doporučený počet je osm hodin (2, 2, 1, 3) včetně představení projektů. Je třeba počítat s možností, že za běhu bude nutné přidat či ubrat hodiny.

Pro části 1 až 3 a první hodinu části 4 je vhodnou metodou výuky výklad spojený se samostatnou prací žáků v hodině. Učitel by měl vždy část látky vysvětlit a pak nechat žáky, aby si vše vyzkoušeli a přišli případně na další možnosti.

Je třeba počítat rovněž s tím, že od části 3 dále by žáci měli pracovat v ideálně dvoučlenných týmech, pokud je málo Micro:bitů tak vícečlenných tak, aby v každém týmu byly nejméně dva Micro:bity.

V závěrečných hodinách pak lze s úspěchem použít projektovou výuku, kdy žáci buď dostanou přidělené téma nebo si jej zcela sami zvolí. V poslední hodině by každý tým měl prezentovat své funkční zabezpečovací zařízení a předvést jeho přednosti a slabiny.

Pro výuku je doporučen Micro:bit verze 2 (Micro:bit V2). Nejlépe pro každého žáka jeden. Doporučuji vybavit se navíc USB kabely a držáky baterií pro běh Micro:bitu mimo počítač. To lze pořídit v kompletu za cca 550 Kč. Nakoupit lze například zde. Současně doporučuji vybavit se náhradními AAA bateriemi.

Máme-li Micro:bit V1, je nutné řešit audio výstup tak, jak je to popsáno v učebnici níže. V tom případě budete potřebovat dva kabely s krokodýlky pro každého a nějaký zdroj zvuku, např. sluchátka.

Pro výuku použijeme programovací jazyk MicroPython, který je podmnožinou jazyka Python. Pro Editor, ve kterém budeme psát programy, máme dvě možnosti:

1. **Thonny** – podle mě lepší volba. Umožňuje program pouze vzdáleně spustit na Micro:bitu bez ukládání na Micro:bit. Ukládání na micro:bit je ale možné též. To se hodí např. při testování zvuku – po stisku reset zvuk skončí. Na druhou stranu není zde kontrola syntaxe.
2. **Mu** – Umožňuje kontrolu syntaxe, ale program musí být před spuštěním na micro:bit uložen.

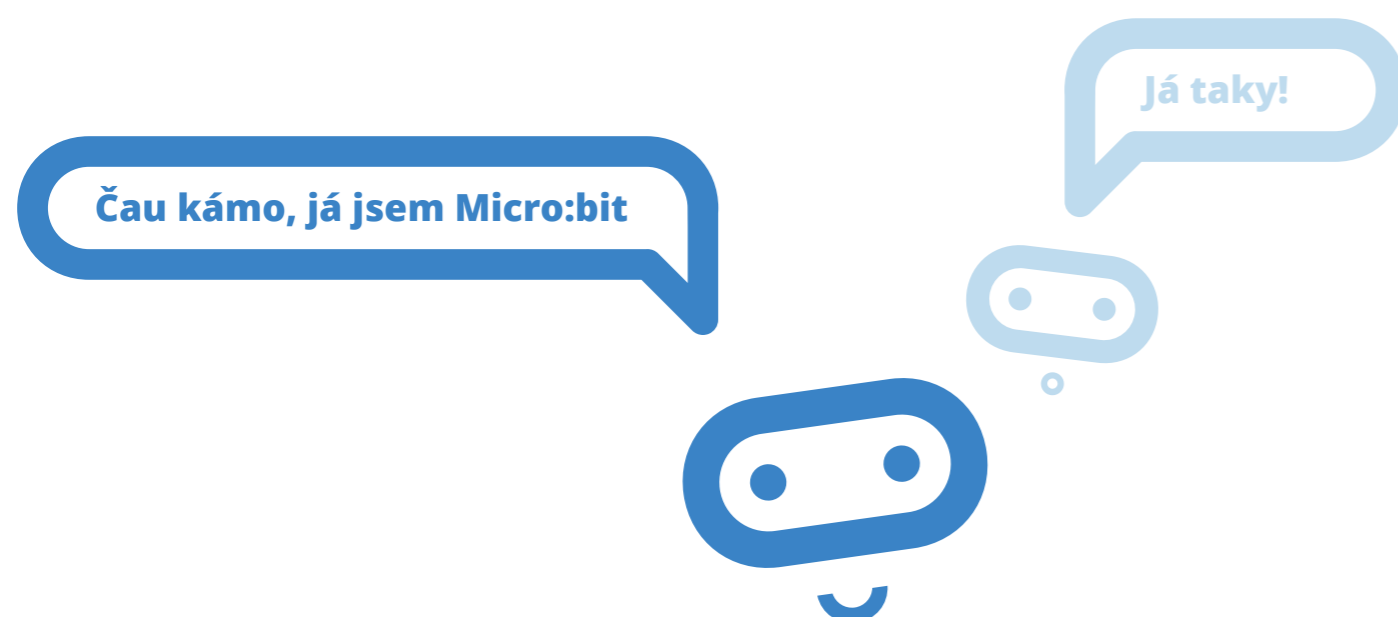
Z hlediska popsaných programů je zcela jedno, který z editorů si zvolíte.

Co se týče podkladů pro výuku, máme několik možností.

Česká učebnice vypracovaná v rámci projektu **iMyšlení** s názvem **Robotika pro střední školy: programujeme Micro:bit pomocí Pythonu** – to by měla být základní literatura pro tento projekt.

O MicroPythonu na micro:bitu jsou následující stránky: **BBC micro:bit MicroPython documentation**.

Mnoho dalších informací a návodů; včetně mnoha zajímavých projektů, lze rovněž najít na adrese **microbiti.cz**, anebo původní webové stránky projektu **BBC Micro:bit**.



Teoretická část k dané problematice

Micro:bit je kapesní počítač, který vás uvede do světa software a hardware a jejich vzájemné spolupráce. Obsahuje LED displej, tlačítka, sensory a mnoho programovatelných funkcí. Nová verze micro:bitu obsahuje i mikrofon a buzzer. (<http://www.microbit.org>)

Vlastnosti Micro:bitu (V2 znamená pouze verze V2):

- ✓ Matice 5x5 LED diod
- ✓ Dvě programovatelná tlačítka
- ✓ Jedno dotekové tlačítko (V2)
- ✓ Akcelerátor, umožňující reagovat na pohyb
- ✓ Čidlo světla, magnetického pole a teploměr
- ✓ Mikrofon (V2)
- ✓ Buzzer (primitivní reproduktor) (V2)
- ✓ Možnost vzájemné komunikace pomocí radia
- ✓ Bluetooth
- ✓ Konektory pro připojení dalších zařízení, z nich tři pro pohodlné připojení pomocí kabelů s krokodýlky
- ✓ Možnost snadného připojení k rozšiřujícím zařízením pomocí pásu konektorů

Existují dvě verze micro:bitu. Starší verze prodáváná od roku 2016, nyní označovaná jako V1 a novější od října 2020 označovaná jako V2.



Obrázek 1: čelní strana micro:bitu, vlevo V1, vpravo V2

Rozhodně si nyní pořizujte V2, je lépe vybavená a má větší paměť pro programy při de facto stejné pořizovací ceně. Její výhodou je integrovaný buzzer a mikrofon, takže pro práci s tímto textem nepotřebujete nic jiného než dva micro:bity.

Micro:bit lze programovat jak pomocí grafického programování, tak pomocí klasických textových jazyků.

Grafické programování – takové programování, kdy program netvoříme psaním kódu na klávesnici, ale posouváním grafických bloků po obrazovce. Mezi tyto nástroje patří například Scratch, MakeBlock nebo Open Roberta Lab.

Microsoft MakeCode – grafický programovací jazyk vyvinutý firmou Microsoft pro programování Micro:bitu.



Obrázek 2: Zadní strana micro:bitu, vlevo V1, vpravo V2

V případě micro:bitu lze použít následující „klasické“ textové jazyky JavaScript a MicroPython.

- ✓ JavaScript můžeme použít v online editoru na stránkách microbit.org, pokud se do něj přepneme v programovém prostředí grafického programovacího jazyka MakeCode.
- ✓ MicroPython můžeme opět programovat v online prostředí domovské stránky micro:bitu anebo pomocí již zmíněných editorů Mu a Thonny.



Příklady z praxe

Autora k vytvoření tohoto materiálu přivedla příhoda z praxe, která se stala, když prvním rokem ověřoval svou učebnici na základní škole v rámci testování.

Dva bratři vlastníci dva micro:bity si doma vytvořili primitivní zabezpečovací zařízení. Jeden microbit ukryli na verandě domu a naprogramovali jej, aby reagoval na změnu světla posláním signálu druhému micro:bitu umístěnému v jejich pokoji. Ten měli připojený k reproduktoru, aby je na tuto skutečnost upozornil. Když přišel domů někdo z rodičů, byli o tom včas informováni a mohli odložit tablety, knihy atd. a vzít si do ruky učebnice.

Ačkoliv z pedagogického hlediska s tím samozřejmě nesouhlasím, z didaktického hlediska se jedná o zajímavý případ spojený se spontánní týmovou spoluprací a využitím znalostí získaných v kroužku.

Dalo by se tedy shrnout, že absolvováním tohoto kurzu žáci získají nebo si prohloubí základní programátorské návyky. Dále se dozví informace ze světa hardware a integrace software a hardware. Zde záleží pouze na učiteli, který musí odhadnout úroveň svých žáků a jak hluboko si může dovést ponořit se do dané problematiky.

Dále lze na této problematice dobře vysvětlit možné bezpečnostní problémy bezdrátové komunikace – např. odposlouchávání, man in the middle attack atd.

V neposlední řadě lze tímto úkolem rozvíjet práci ve skupině – pro správnou funkci naprosté většiny případů jsou nutné dva micro:bity. S úspěchem lze na závěr zadat žákům projekt ve skupině, který budou pak muset před ostatními představit a obhájit.



Metodická a didaktická část

Jak bylo vysvětleno v první kapitole, kurz je rozdělen do čtyř částí. Postupně se zastavíme u všech čtyř a řekneme si co v nich učít.

Práce s výstupy

V této části je třeba zvládnout následující úkoly:

- ✓ Seznámit se s micro:bitem
- ✓ Seznámit se s prostředím editoru Mu nebo Thonny
- ✓ Naučit se nahrát vytvořený kód do micro:bitu
- ✓ Práce s displejem – zobrazení ikonky a textu
- ✓ Práce se zvukem

Na úvod rozdává učitel žákům micro:bity a v krátkosti jim je představí. Upozorníte žáky na skutečnost, že micro:bit má na sobě popisky jednotlivých součástí. Vysvětlíte význam jednotlivých tlačítek – A, B programovatelná tlačítka, Reset slouží k opětovnému spuštění programu od začátku anebo k vypnutí micro:bitu při podržení 3 sekundy.

Nyní si studenti spustí zvolený editor. Seznamte je s prostředím a základními principy používání zvoleného editoru. V dalším průběhu budu předpokládat použití mnou doporučeného editoru Thonny.

Zkuste vytvořit jednoduchý program a nahrát jej do micro:bitu. Ukažte žákům rozdíl mezi možnostmi Uložit (program se uloží na disk) a Spustit (program se spustí rovnou do micro:bitu). Vysvětlíte, že po připojení je micro:bit přístupný podobně jako flash disk a uložený program do něj lze nahrát i tak, že jej na tento disk přetáhneme. Ukažte žákům, že během nahrávání bliká žlutá dioda na zadní straně. Také jim ukažte, že ukončený program mohou opět spustit stiskem tlačítka Reset.



```
from microbit import *
display.scroll("Ahoj svete")
```

U dalších programů v této části nebudou již většinou vkládány zdrojové kódy. Naleznete je v příložených pracovních listech.

Nechte žáky upravit program tak, že přidáte blok pro nekonečné opakování. Zkuste přidat časovou prodlevu. Vyzkoušejte zobrazení připravených obrázků, opět s vhodnými časovými prodlevami. Vyzkoušejte případně i cyklus pro pevný počet opakování. Vysvětlíte žákům rozdíly mezi cykly while a for.

Vyzkoušejte si přehrávání zvuku. Pokud máte micro:bit V1, musíte prvně připojit výstupní zařízení. Vyzkoušejte bloky „hraj tón“, „hraj melodii“ a „play sound“ (pouze V2). Můžete nechat žáky složit vlastní melodii, ale v tom případě doporučuji vybavit se klapkami na uši a třídu zvukově izolovat.

Práce se vstupy

- ✓ Kromě pomůcek z minulé kapitoly doporučuji jeden nebo více silných magnetů na třídu.

Nejjednodušším typem vstupu jsou dvě tlačítka značená A a B. Jejich nejjednodušší použití je pomocí bloku „Po stisknutí tlačítka A(B)“ v kategorii vstup. Naprogramujte akci (výstup na displej, zahrání tónu“ po stisku zvoleného tlačítka). Přiřaďte aktivitu k oběma tlačítkům. Např. takto:



```
from microbit import *
while True:
    if button_a.is_pressed():
        display.show(Image.HAPPY)
    if button_b.is_pressed():
        display.show(Image.SAD)
    sleep(100)
    display.clear()
```

Vyzkoušejte si nyní další možnosti vstupů.

- ✓ microphone.sound_level() – test na úroveň zvuku u micro:bitu V2
- ✓ pin_logo.is_touched() – stisknuto logo na micro:bitu V2
- ✓ accelerometer.was_gesture("shake") – zatřesení micro:bitem
- ✓ temperature() – teplota
- ✓ display.read_light_level() – úroveň osvětlení
- ✓ compass.get_field_strength() – intenzita magnetického pole
- ✓ compass.heading() – azimut ve stupních, nutná inicializace pomocí compass.calibrate()

Napište program, který po stisku tlačítka A zobrazí teplotu, po stisku tlačítka B intenzitu světla a současném stisku obou tlačítek zobrazí intenzitu magnetického pole. Experimentujte se zhasínáním a rozsvícením a osvětlením a dále s přiblížením magnetu. Napište program, který po stisku tlačítka A zobrazí teplotu, po stisku tlačítka B intenzitu světla a současném stisku obou tlačítek zobrazí intenzitu magnetického pole. Experimentujte se zhasínáním a rozsvícením a osvětlením a dále s přiblížením magnetu.

Můžete na neznalcích vyzkoušet magii. Přiblížíte-li ruku zatátou v pěst k micro:bitu, zobrazí se smajlík. Neříkejte, že máte v té pěstě silný magnet a vyzvěte je, ať to zkusí po vás. Můžete úkol vylepšit tím, že řeknete, že obrázek zobrazí jen tomu, kdo má magický potenciál. Slovo magický pak můžete nahradit slovem magnetický a sledujte kdy si toho dotyčný všimne. Můžete vyzkoušet ještě další magický program, při zatřesení vám dá micro:bit náhodnou odpověď ano/ne. Můžete se ho tak dotazovat na mám/nemám.

Můžete vyzkoušet ještě další magický program, při zatřesení vám dá micro:bit náhodnou odpověď ano/ne. Můžete se ho tak dotazovat na otázky na které je odpověď ano/ne. Můžete přidat i možnosti nevím, nelze rozhodnout atd.

Komunikace mezi dvěma micro:bity

- ✓ Pro splnění této hodiny musí již být vytvořeny nejméně dvoučlenné skupiny vybavené dvěma micro:bity.

Na úvod naprogramujte takovýto jednoduchý programek na obou micro:bitech ve skupině. >>

Je důležité, aby měly různé skupiny různou skupinu rádia (zde 23), jinak se budou vzájemně rušit. Skupinu volíme z intervalu <0,83>. (Nechte nejdříve žáky, aby se přesvědčili, co se stane, když to neudělají. Bude se vám pak dobře vykládat bezpečnost bezdrátového spojení.)

Poté co si žáci rozdělí skupiny rádia, je nechte zkusit si vzájemně posílat signál. Je-li to možné (a bezpečné), vypusťte je mimo učebnu, ať si otestují dosah micro:bitů ve volném prostoru i přes zdi. Bude se to hodit při tvorbě zabezpečovacího zařízení.

Naeditujte složitější případ, kdy se posílá jiné číslo při stisku A a jiné při stisku B. Máte-li V2, tak třetí se pošle při stisku loga. Přijímající micro:bit pak podle přijatého čísla rozhodne o své činnosti.

Žáci mohou kód upravit tak, že význam při poslání signálu z jednoho micro:bitu může být např. „Půjdeme na hřiště?“, „Půjdeme se učit“, „Máme úkoly“ a odpovědi zpět např. „Ano“, „Ne“ a „Nevím“. Ponechte žákům, ať si sami vyberou a řešení otestují. Micro:bity lze nyní označit jako tázající a odpovídající.

Vysvětlete žákům nebezpečí této komunikace. Pokud útočník zná použitou skupinu a význam kódů může konverzaci odposlouchávat. Navíc může i do komunikace vstoupit a způsobit zmatení účastníků.

```
from microbit import *
import radio
kanal = 23
radio.on()
radio.config(channel = kanal)
display.clear()
while True:
    if button_a.is_pressed():
        radio.send("Zprava")
    zprava = radio.receive()
    if (zprava):
        display.scroll(zprava)
        zprava = ""
    sleep(100)
radio.off()
```

Tvorba zabezpečovacího zařízení

Vytvoříme jednoduché zabezpečovací zařízení, které bude reagovat na intenzitu světla.

Vyzkoušejte si nejprve na jednoduchém programu rozdíl v hodnotě intenzity světla mezi zhasnutím a rozsvícením (zavřenými a otevřenými žaluziemi). >>

Potřebujeme dva micro:bity:

- ✓ Vysílač – hlídá intenzitu světla a při rozsvícení (zhasnutí) vyšle signál. Kód by mohl vypadat např. takto: >>

```
from microbit import *
while True:
    display.scroll(display.read_light_level())
```

```
from microbit import *
import radio
kanal = 23
radio.on()
radio.config(channel = kanal)
while True:
    display.scroll(display.read_light_level())
    if (display.read_light_level()>127):
        radio.send("Poplach")
```

- ✓ Přijímač – dostane info, že došlo k incidentu a vyhlásí poplach. Kód například takto: >>

Nyní zadejte žákům projekt na vytvoření zabezpečovacího zařízení pomocí dvou micro:bitů. Napovězte jim, co mohou použít – stisk tlačítka (naopak i uvolnění tlačítka), změna intenzity světla, teploty, magnetického pole, audio signál, pohyb.

```
from microbit import *
import music
import radio
kanal = 23
radio.on()
radio.config(channel = kanal)
display.clear()
while True:
    zprava = radio.receive()
    if (zprava):
        display.scroll(zprava)
        zprava = ""
        music.play(music.NYAN)
    sleep(100)
radio.off()
```

Doporučené pomůcky

V první řadě potřebujeme dostatek micro:bitů. Ideální stav je, když má každý žák svůj micro:bit. Ideálnější je pak, když má svůj osobní micro:bit, pokud si to může škola anebo rodiče žáků dovolit.

Ke každému micro:bitu potřebujeme připojovací USB kabel a doporučuji i přenosný držák na baterie. Ideální je koupit vše dohromady jako výhodný balíček, jak je popsáno v úvodu textu.

Dále potřebujeme počítač, který má nainstalovaný editor Mu nebo Thonny, anebo asi nejlépe oba. Určitě doporučuji druhou možnost, nebudeme tolik ovlivněni vrtochy internetového připojení.

Pro vysvětlení reakce na magnetické pole potřebujeme také silnější magnet (např. „vypůjčený“ z nástěnky).

Pokud máme starší micro:bit verze jedna, pak ještě potřebujeme externí repráčky anebo sluchátka.

Pracovní list

Přílohou tohoto materiálu jsou pracovní listy. Tyto pracovní listy jsou k dispozici v editovatelné elektronické formě, aby si je každý učitel mohl upravit, např. dle toho, co má již s žáky probráno.

Pracovní listy jsou čtyři, pro každé z výše uvedených témat jeden. V pracovních listech počítám s tím, že škola má k dispozici Micro:bit V2. Pokud má k dispozici pouze starší verzi V1, je nutné tyto listy upravit. Rozdíly jsou zejména v tom, že u V1 je nutno pro přehrání zvuku připojit externí zařízení a dále V1 nemá mikrofon a dotekové tlačítko, takže tyto dvě zařízení nelze použít jako čidlo pro vyhlášení poplachu.

Pracovní list 1 Práce s výstupy micro:bitu

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem Thonny nebo Mu
- ✓ Micro:bit V2
- ✓ Propojovací USB kabel

Ajdeme na to

Vezměte si svůj micro:bit a pořádně si jej prohlédněte. Na přední straně máte pole 5x5 LED diod, které umí svítit různou intenzitou červené barvy. Po jejich stranách máte dvě programovatelná tlačítka označení A a B. Nad diodami se ještě nachází programovatelné dotekové tlačítko a vpravo nahoře nad LED je malá tečka, což je vlastně mikrofon. Otočme nyní micro:bit a podívejme se na zadní stranu. Nahoře máme dva porty microUSB pro připojení k PC a konektor pro připojení baterie packu. Mezi nimi je tlačítko RESET pro opětovné spuštění programu od začátku. Všimněte si, že další části jsou zde popsány. Pro úplnost ještě dodejme, že z obou stran dole vidíme piny pro připojení dalších periférií. Celý micro:bit je například možné zasunout do nějakého externího zařízení.

Spusťte editor Thonny anebo Mu a seznamte se s ním.

Zkuste vytvořit jednoduchý program a nahrát jej do micro:bitu. Zkuste si vysvětlit mezi možnostmi spustit a uložit jako u Thonny. Všimněte si, že během nahrávání bliká žlutá dioda na zadní straně. Vyzkoušejte si i to, že program lze spustit opakovaně pomocí tlačítka RESET (anebo odpojením a připojením micro:bitu ke zdroji energie).

Zkuste přidat blok while True (pozor u True musí být velké písmeno) a přidat časovou prodlevu. Vyzkoušejte zobrazení připravených obrázků, opět s vhodnými časovými prodlevami.

Upravíme program, tak aby text zobrazil třikrát:

Pozor - pro tři zobrazení opravdu musí rozsah být 1 až 4. Je to vlastně polozavřený (zleva) interval.

```
from microbit import *
display.scroll("Ahoj svete")
```

```
from microbit import *
while True:
    display.scroll("Ahoj svete")
    sleep(100)
```

```
from microbit import *
for i in range(1, 4):
    display.scroll("Ahoj svete")
    sleep(1000)
display.clear()
```

Vyzkoušejte si přehrávání zvuku:



```
from microbit import *
import music
display.show(Image.HAPPY)
while True:
    music.play(music.POWER_UP)
    sleep(1000)
```

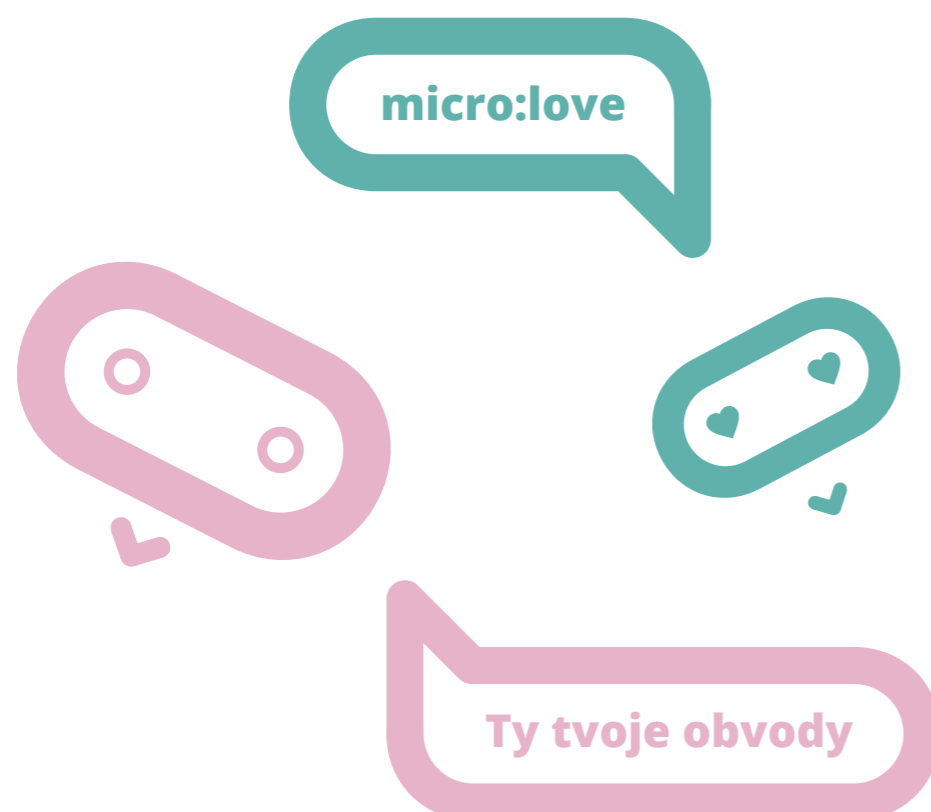
Až vás začnou bolet uši, stiskněte tlačítko reset na micro:bitu (pouze u Thonny). Můžete nechat micro:bit i promluvit:



```
from microbit import *
import speech
speech.say("Attention please", speed=100)
```

Co jste se naučili

Zobrazovat obrázky a text na displeji micro:bitu. Víte, jak část kódu spustit jednou, opakovaně, několikrát. Dále umíte na micro:bitu přehrát tón a nechat jej promluvit.



Pracovní list 2 Práce se vstupy micro:bitu

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem Thonny nebo Mu
- ✓ Micro:bit V2
- ✓ Propojovací USB kabel

A jdeme na to

Nejjednodušším typem vstupu jsou dvě tlačítka značená A a B. Micro:bit V2 navíc dotykové logo. Použití dobře demonstruje následující příklad:



```
from microbit import *
import music
while True:
    if button_a.is_pressed():
        display.show(Image.HAPPY)
    elif button_b.is_pressed():
        display.show(Image.SAD)
    elif pin_logo.is_touched():
        music.play(music.DADADADUM)
    sleep(100)
    display.clear()
```

Jedná se o podmíněný příkaz. Pokud platí první podmínka, provede se a skočí se na konec bloku, a tak dále. Část jinak (else) zde chybí.

Nyní vyzkoušejte měřit další veličiny a zobrazit jejich hodnotu: intenzitu světla, magnetického pole a teplotu. Můžete vyzkoušet i azimut. Nezapomeňte, že před použitím se musí micro:bit zkalibrovat (micro:bit napíše výzvu, pak se objeví uprostřed displeje bod a natáčením micro:bitu musíme celý displej zaplnit).

Použijte následující příkazy:

- ✓ microphone.sound_level() – test na zvuk u micro:bitu V 2
- ✓ pin_logo.is_touched() – stisknuto logo namicro:bitu V 2
- ✓ accelerometer.was_gesture("shake") – zatřesení
- ✓ temperature() – teplota
- ✓ display.read_light_level() – úroveň osvětlení
- ✓ compass.get_field_strength() – intenzita magnetického pole
- ✓ compass.heading() – azimut ve stupních, nutná inicializace pomocí compass.calibrate()

Napište program, který po stisku tlačítka A zobrazí teplotu, po stisku tlačítka B intenzitu světla a současném stisku obou tlačítek zobrazí intenzitu magnetického pole. Experimentujte se zhasínáním a rozsvícením a osvětlením a dále s přiblížením magnetu.

Zkuste podobně experimentovat s úrovní hlasitosti v okolí.

Můžete na neznalcích vyzkoušet magii: ➤

Přiblížíte-li ruku zaťatou v pěst k micro:bitu, zobrazí se smajlík. Neříkejte, že máte v té pěstí silný magnet a vyzvěte je, ať to zkusí po vás. Můžete úkol vylepšit, tím že řeknete, že obrázek zobrazí jen tomu, kdo má magický potenciál. Slovo magický pak můžete nahradit slovem magnetický a sledujte kdy si toho dotýčný všimne.

Můžete vyzkoušet ještě další magický program, při zatřesení vám dá micro:bit náhodnou odpověď ano/ne. Můžete se ho tak dotazovat na mám/nemám.

```
from microbit import *
hodnota = 5000
compass.calibrate()
pocatek = compass.get_field_strength()
while True:
    sleep(100)
    sila = compass.get_field_strength()
    if abs(sila - pocatek) > hodnota:
        display.show(Image.HAPPY)
        sleep(3000)
        display.clear()
```

Co jste se naučili

Programovat vstupní události, např. stisk tlačítka nebo zatřesení microbitem. Naučili jste se odečítat fyzikální veličiny z reálného světa – teplotu, světlo, hluk a magnetické pole. Vše toto později využijete v tvorbě zabezpečovacích zařízení.

Pracovní list 3 Komunikace mezi dvěma micro:bity

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem MakeCode anebo s přístupem na internet
- ✓ 2 Micro:bity nejlépe V2
- ✓ Propojovací USB kabel
- ✓ Doporučuji pro oba Micro:bity použít Baterie Box – pro testování spojení v „terénu“

A jdeme na to

V této a následující hodině musí být vytvořeny nejméně dvoučlenné skupiny vybavené dvěma micro:bity.

Na úvod naprogramujte takovýto jednoduchý prográmk na obou micro:bitech ve skupině. ➤

Je důležité, aby měly různé skupiny různou skupinu rádia (zde 23) z intervalu <0,83>, jinak si vzájemně polezete do zelí. Signál by měly obdržet všechny micro:bity se stejnou skupinou.

Zkoušejte si posílat vzájemně signál. Připojte micro:bity na baterie pack a naopak je odpojte z USB kabelu a vezměte je na procházku ven a vyzkoušejte dosah signálu.

Pro testování si také můžete nahrát na jeden micro:bit, který ponecháte připojený u počítače následující program a jít „na procházku“ jen s jedním micro:bitem, na kterém ponecháte předchozí program a zjistit tak dosah. Nebo upravte program na micro:bitu, aby posílal automaticky zprávy po dvou vteřinách a vyzkoumáte tak dosah micro:bitu z učebny.

```
from microbit import *
import radio
kanal = 23
radio.on()
radio.config(channel = kanal)
display.clear()
while True:
    if button_a.is_pressed():
        radio.send("Zprava")
        zprava = radio.receive()
    if (zprava):
        display.scroll(zprava)
        zprava = ""
    sleep(100)
radio.off()
```


Pozor - pokud připojíte Micro:bit k počítači, vždy nejdříve odpojte Battery Pack, aby nedošlo k poškození Micro:bitu vyšším napětím.

Naeditujte složitější případ, kdy se posílá jiné číslo při stisku A a jiné při stisku B. Přijímající micro:bit pak podle přijatého čísla rozhodne o své činnosti (např., kterou zprávu zobrazí).

Máte-li V2, upravte příklad tak, že třetí číslo se pošle při stisku loga.

Upravte kód tak, že význam při poslání signálu z jednoho micro:bitu může být např. „Půjdem na hřiště?“, „Půjdeme se učit?“, „Máme úkoly?“ a odpovědi zpět např. „Ano“, „Ne“ a „Nevím“. V takovémto případě už bude nutné mít v micro:bitu různý kód.

Uvědomte si nebezpečí této komunikace. Pokud útočník zná použitou skupinu a význam kódů, může konverzaci odposlouchávat. Navíc může i do komunikace vstoupit a způsobit zmatení účastníků.

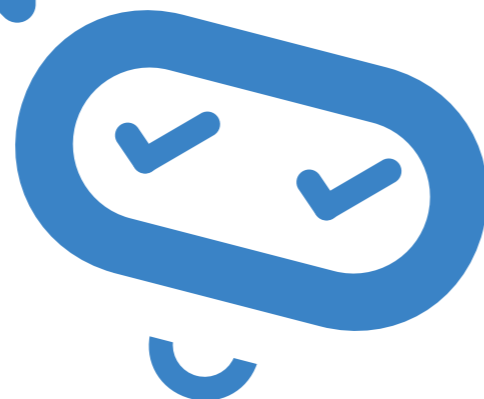
Co jste se naučili

Jak komunikují dva micro:bity.

Poslat zprávu z jednoho micro:bitu na druhý a dešifrovat její význam.

Rovněž jste se dozvěděli něco o bezpečnosti bezdrátové komunikace.

Easy! Nebo ne?



Pracovní list 4

Tvorba zabezpečovacího zařízení

Co budeme potřebovat

- ✓ Počítač s nainstalovaným editorem Thonny nebo Mu
- ✓ 2 Micro:bity nejlépe V2
- ✓ Propojovací USB kabel
- ✓ Doporučuji pro oba Micro:bity použít Batery Box – možnost umístění i mimo počítač

A jdeme na to

Vytvoříme jednoduché zabezpečovací zařízení, které bude reagovat na intenzitu světla. Budeme opět pracovat ve dvojicích.

Vyzkoušejte si nejprve na jednoduchém programu rozdíl v hodnotě intenzity světla mezi zhasnutím a rozsvícením (zavřenými a otevřenými žaluziemi).



```
from microbit import *
while True:
    display.scroll(display.read_light_level())
```

Stanovte si hodnotu, která pro vás bude znamenat dělicí hodnotu mezi světlem a tmou pro vyhlášení poplachu.

Nyní již vytvoříme jednoduché zabezpečovací zařízení. Hodnotu 128 můžete nahradit odpozorovanou hodnotou z předchozího programu. Tentokrát budeme nahrávat rozdílné programy na každý Micro:bit.

- ✓ Vysílač – hlídá intenzitu světla a při rozsvícení (zhasnutí) vyšle signál – poplach. Kód by mohl vypadat např. takto:



```
from microbit import *
import radio
kanal = 23
radio.on()
radio.config(channel = kanal)
while True:
    display.scroll(display.read_light_level())
    if (display.read_light_level()>127) :
        radio.send("Poplach")
```

- ✓ Přijímač – dostane info, že došlo k incidentu a vyhlásí poplach. Kód by mohl vypadat například takto:



```
from microbit import *
import music
import radio
kanal = 23
radio.on()
radio.config(channel = kanal)
display.clear()
while True:
    zprava = radio.receive()
    if (zprava):
        display.scroll(zprava)
        zprava = ""
        music.play(music.NYAN)
        sleep(100)
radio.off()
```

Zkuste upravit kód přijímače, tak aby opakoval varování, dokud není např. stisknuto tlačítko. (potvrzení poplachu)

Nyní vytvořte libovolné zabezpečovací zařízení pomocí dvou micro:bitů.

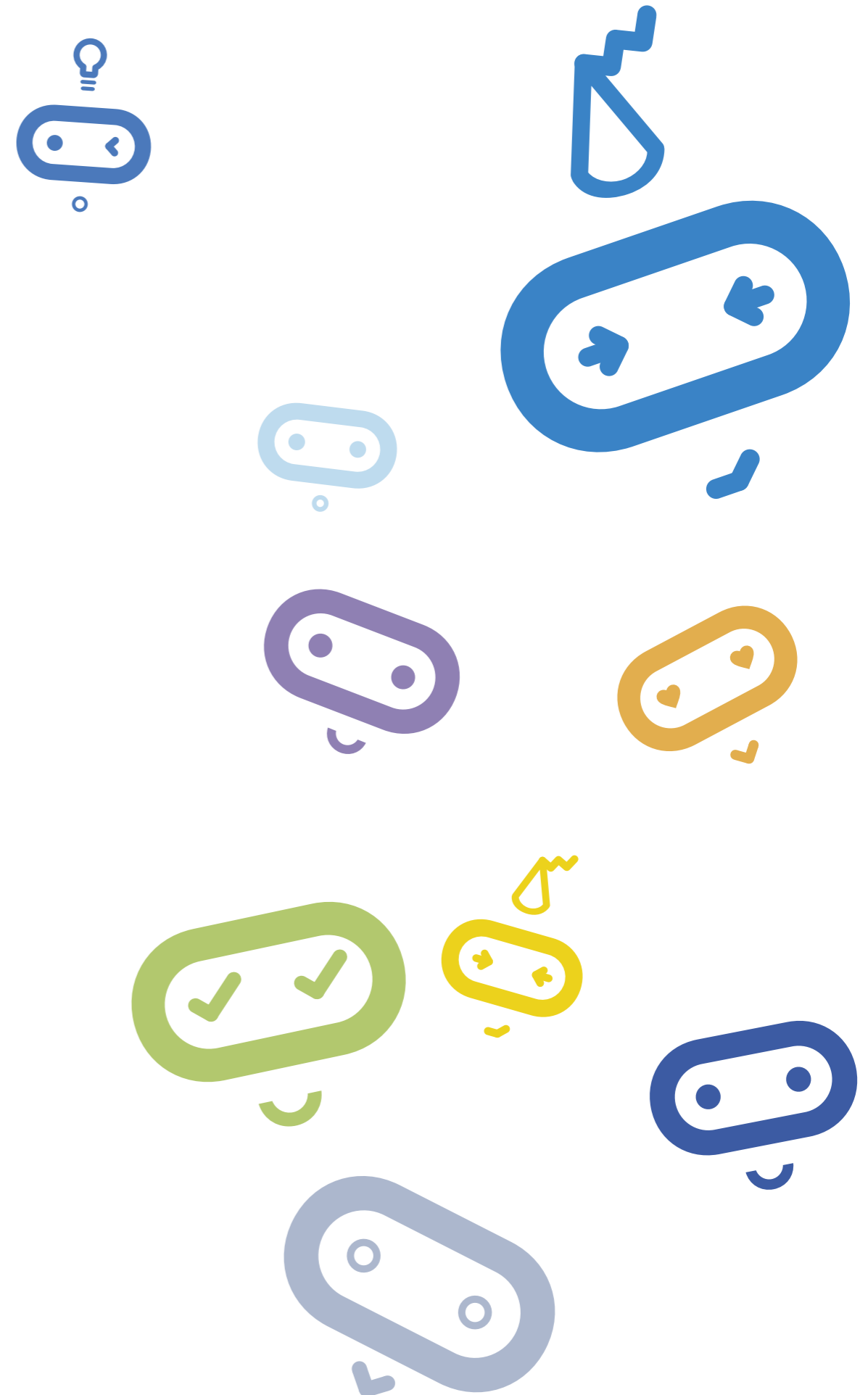
Můžete vyzkoušet různé typy událostí:

- ✓ stisk tlačítka (naopak i uvolnění tlačítka)
- ✓ změna intenzity světla
- ✓ změna teploty
- ✓ změna intenzity magnetického pole
- ✓ audio signál (zvuk)
- ✓ pohyb micro:bitu (zatřesení)

Co jste se naučili

Znalosti získané v tomto kurzu jsou vlastně znalostmi ze světa robotiky. Umíte nyní sledovat své okolí a zjišťovat jeho změny – zde se jedná o analogii s lidskými smysly. Na jednu stranu sice nemáte smysly „chuť a zrak“, ale máte zde naopak smysl intenzita magnetického pole. Na tyto události nyní umíte reagovat zprávami na displej či zvukem. Umíte rovněž komunikovat s jiným micro:bitem na dálku.

Je třeba si uvědomit, že ačkoliv tento kurz vás mnoho naučil, záleží pouze na vás, jak se získanými dovednostmi naložíte a zda je dále budete prohlubovat pomocí dalších kurzů či další literatury.



2022

TVORBA DIGITÁLNÍHO OBSAHU PRO SŠ

Základní instrukce

Tento kurz je určen zejména pro žáky prvních a druhých ročníků středních škol nebo pro vyšší ročníky osmiletých gymnázií. Autor kurzu se řadu let zabývá 2D a 3D CAD počítačovou grafikou a 3D technologiemi. U samotného využití kurzu vždy záleží na přístupu, invenci a zkušenostech lektora. Pak je možné jej využít též v závěrečných ročních základních škol.

Časovou dotaci kurzu nelze jednoznačně určit. Pro absolvování kurzu se u žáků nepředpokládají žádné předchozí znalosti a zkušenosti s 3D modelováním a 3D technologiemi. Výslednou dotaci lektor stanoví po prostudování materiálů ke kurzu.

Příklad rozdělení kurzu a doporučená časová dotace:

1. Pojem „Digitální obsah“ a pojem „3D“, využití cloudových technologií.

- Vzhledem k veliké šířce samotného pojmu „Digitální obsah“ je třeba nejdříve toto téma uchopit obecně a vysvětlit, co vše se pod tímto pojmem může skrývat a co naopak už digitálním obsahem není. Zároveň je pak nutno uvést, čeho konkrétně se kurz týká, to znamená vysvětlit využití 3D prostoru ve výpočetní technice.
- Vysvětlení pojmu „cloudové technologie“ a jejich využití v oblasti počítačových aplikací – obecný popis a konkrétní příklad v oboru 3D modelování s aplikací **TinkerCAD**.
- Zde by měla stačit jedna, maximálně dvě vyučovací hodiny (45–90 minut).

2. Vytvoření osobního účtu aplikace TinkerCAD, seznámení se systémem.

- Předpokládá se práce na osobním počítači či notebooku s připojením k internetu. Prvním krokem bude vytvoření účtu, pod kterým se budou žáci k systému přihlašovat
- Základní seznámení se systémem TinkerCAD – vzhledem k tomu, že systém je lokalizovaný a velmi intuitivní, naučí se s ním žáci pracovat poměrně rychle.
- Opět zde budou stačit 2 vyučovací hodiny (90 minut)

3. Vytvoření jednoduchého modelu podle instrukcí lektora + model na „volné téma“.

- Zde je třeba přihlídnout k možnostem (počet žáků a počítačových pracovišť, plánovaný čas apod.), protože na této části je možné strávit poměrně dost času. Rozhodně lze doporučit v této části nespěchat. Práce v systému žáky většinou baví, ne každý však má potřebnou 3D představivost a tak je třeba postup přizpůsobit podle těch pomalejších.
- Doporučená dotace jsou alespoň tři vyučovací hodiny (135 minut).

4. Příprava modelu pro 3D tisk, využití aplikace 3D skeneru.

- Vzhledem k tomu, že ne každá škola má dnes k dispozici 3D tiskárny, je tato část kurzu volitelná (bude zařazena jako součást workshopu). Stejně tak je na volbě lektora, zda zařadí použití aplikace 3D skeneru všemi žáky, či zda sám předvede ukázkou a žákům zadá práci na doma.
- Samotný proces 3D tisku je poměrně zdoluhavý, doporučuji se žáky vždy modely do tiskárny nahrát, spustit tisk a mezitím se věnovat jiné činnosti, např. práci s 3D skenerem. Protože se u 3D tisku občas vyskytnou technické problémy (např. s tiskovým materiálem) je dobré nechat si pro tuto činnost časovou rezervu. Doporučuji tedy minimálně dvě vyučovací hodiny (90 minut).



Cílem kurzu je využití cloudových nástrojů a mobilních technologií pro práci s 3D grafikou. Úroveň je pro první a druhé ročníky SŠ. V současné době jsou volně k dispozici cloudové (on-line) nástroje pro vytváření a editaci 3D objektů. Mnoho 3D modelů pak poskytli jejich autoři na on-line databázích i ke stažení a k jejich dalšímu volnému využití. Naučíme se tak pracovat s modelovací cloudovou aplikací. Budeme vytvářet nové a také upravovat již hotové 3D modely. Modelář umožňuje export pro následný 3D tisk, což bude také ilustrativně předvedeno. Výkonnější mobilní telefony nebo tablety lze, společně s vhodnou aplikací, použít jako 3D skener. I toto je způsob, jak vytvořit 3D model, který opět můžeme následně upravit a případně vytisknout. Výstupem kurzu by tak měl být jednoduchý 3D model vytvořený pomocí PC na cloudovém 3D modeláři a vysvětlení možností s případnou ukázkou práce s 3D tiskárnou a 3D skenerem.

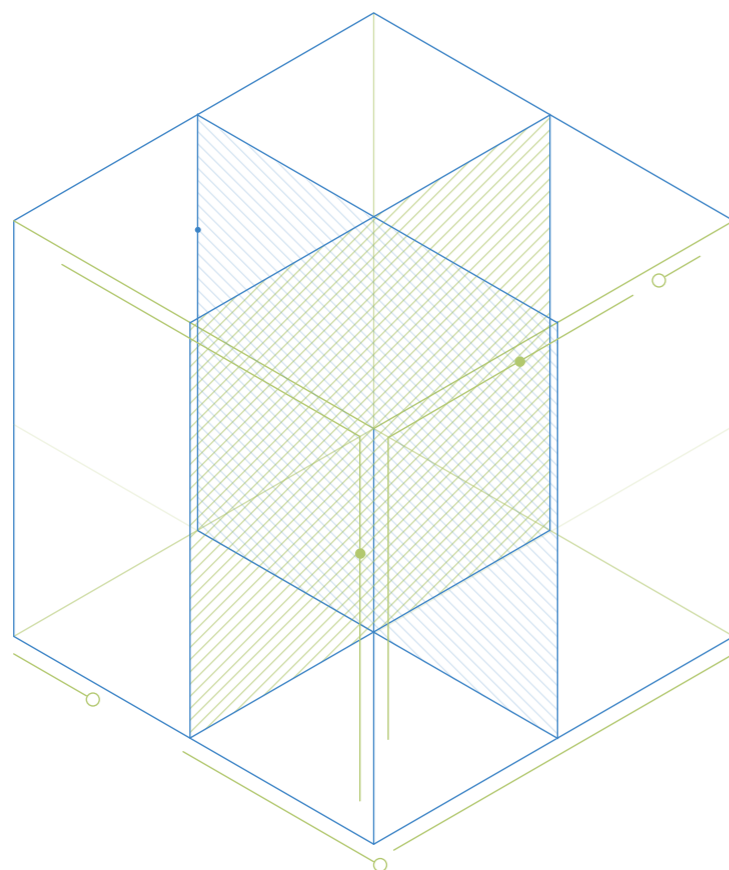
Minimální doba absolvování kurzu je osm vyučovacích hodin. Výsledná doba se může v průběhu kurzu upravit podle potřeby.

Pro část 1 je použita metoda výkladu s ukázkami, části 2 a 3 budou realizovány jako praktické cvičení v počítačové učebně. Podle možností lektora, počtu žáků a také podle dostupného vybavení mohou praktická cvičení probíhat individuálně či ve dvojicích. Pro část 4 se pak předpokládá buď využití učebny s 3D tiskárnami (pokud takovou učebnou škola disponuje), případně lze pracovat i s jednou či dvěma tiskárnami přímo v počítačové učebně.

Pro samotný kurz je doporučeno, aby každý žák měl k dispozici své počítačové pracoviště. Není třeba žádný zvláštní výpočetní výkon, stačí počítač s běžnou grafikou, nezbytné je připojení k internetu.

Pro využití 3D tisku je samozřejmě nutností 3D tiskárna – záleží na možnostech školy, optimální se jeví využít zároveň alespoň 4 tiskárny. Mobilní aplikace 3D skeneru vyžadují poměrně velký výpočetní výkon mobilního telefonu a tak je na zvážení lektora, zda se pokusit zapojit všechny nebo jen některé žáky (negativně to může působit na žáky, kteří z jakéhokoliv důvodu nevládnou vyhovující mobilní telefon).

Tento kurz rozhodně nemá být manuálem pro výuku 3D modelování, chce pouze naznačit možnosti vytváření digitálního obsahu s využitím cloudových technologií. Jako podkladů je třeba využít internetové zdroje (kterých je naštěstí dostatek), žádná vhodná publikace bohužel v současné době není k dispozici.



Teoretická část k dané problematice

Digitální obsah

Digitální obsah je nesmírně široký pojem. Než se pustíme do práce, je třeba jej alespoň nějak uchopit (neexistuje žádná obecná definice pojmu Digitální obsah).

Pro naše potřeby bychom mohli definovat tvorbu digitálního obsahu jako vytvoření a uložení nějakého datového souboru pomocí výpočetní techniky (počítač, notebook, mobilní zařízení, digitální fotoaparát, skener apod.). Tento datový soubor pak může být uložen lokálně přímo na zařízení, kterým jsme obsah vytvořili, nebo můžeme využít řadu jiných úložišť (přenosné, síťové, cloudové atd.).

Uvedme alespoň několik příkladů, co je vytvořený digitální obsah: prostý textový soubor, digitální fotografie, e-mailová zpráva, webové stránky, zvukový či videosoubor,...

Ale také co není tvorbou digitálního obsahu: prohlížení webových stránek, poslouchání zvukového souboru či sledování videosouboru, hraní počítačových her,...

Již z tohoto krátkého výčtu jste si jistě udělali představu, jak širokým pojmem je tvorba digitálního obsahu a že tedy samozřejmě není možné toto téma směstnat na plochu jednoho krátkého kurzu.

Pojem 3D

Při výuce 3D grafiky a 3D technologií vždy u žáků začínám s dotazem: „Co rozumíte pod pojmem 3D“ nebo co to znamená, když je něco tzv. „3D“ a „2D“ (předpokládají se tedy alespoň základní znalosti z geometrie těles). Je nutné, aby před tím, než začneme používat 3D modelář, žáci chápali rozdíl mezi prací ve dvourozměrném a třírozměrném prostoru.

Základní znalosti o 3D by měli mít žáci již ze ZŠ, nicméně není bohužel výjimkou, že problémy s představivostí ve 3D mají i studenti technických středních škol.

Cloudové technologie

Opět se jedná o velmi široký pojem. Většina uživatelů si dnes „cloud“ ztotožňuje s ukládáním dat na nějaké vzdálené úložiště. To je ovšem pouze jeden z mnoha způsobů využití cloudových technologií. Samotné téma je opět mimo rozsah tohoto kurzu.

Pro naše potřeby budeme využívat tzv. cloudové aplikace, tzn., že takovou aplikaci spouštíme jako klienti na nějakém vzdáleném zařízení (serveru), na které máme zřízený přístup – zpravidla musíme mít vytvořený uživatelský účet. Jak již bylo výše uvedeno, budeme pracovat s aplikací, která umožňuje vytváření 3D modelů.

Jako velmi efektivní se pro výuku jeví využití jednoduché (nikoliv však primitivní) aplikace TinkerCAD pocházející z dílny společnosti Autodesk.

Příklady z praxe

Autor při tvorbě tohoto materiálu vycházel z praktické výuky odborných předmětů a z práce v laboratoři na střední odborné škole. Zde se jednak přímo vyučuje 3D modelování, ale navíc při práci v laboratoři s technikou různého druhu (HW, síťová technika, měřicí přístroje atd.) často dochází k situaci, kdy je třeba něco opravit či pořídit pomůcku, držáček, krabičku apod. V takových případech se velmi osvědčila právě možnost takový předmět vymodelovat a posléze si jej podle potřeby vytisknout na 3D tiskárně.

Při dnešní ceně a stále se rozšiřující vybavenosti škol i firem 3D technologiemi je takovýto postup již dnes poměrně běžný i v odborné praxi a jeví se jako velmi efektivní právě v případech potřeby různých pomůcek v počtech jednotek kusů nebo takových pomůcek, které nelze koupit.

Pokud se tedy žákům správně vysvětlí toto využití, budou již takovou praxi považovat za běžnou a zvyknou si na její využití v dalším životě – při dalším studiu, v zaměstnání či v podnikatelské činnosti.

Metodická a didaktická část

TinkerCAD

Zkratkou CAD (Computer Aided Design) označujeme v podstatě projektování s využitím počítačů. Mezi nejznámější CADovské aplikace patří AutoCAD (rovněž z dílny AutoDesku), který se u nás objevil již kolem roku 1985 v rámci projektu „2000 pracovišť Automatizace inženýrských prací“. V dnešní době je rodina produktů velmi široká a mezi základní aplikace pro 3D modelování patří právě TinkerCAD (dále jej budeme označovat jako 3D modelář).

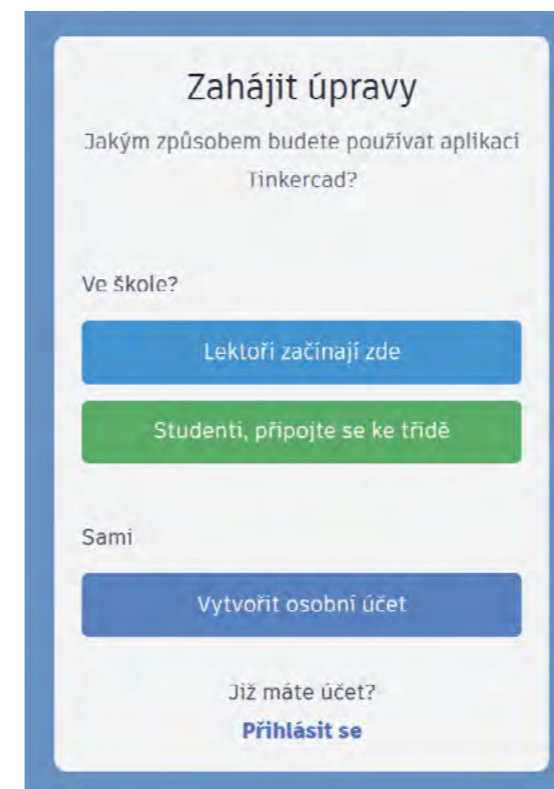


Obr. 1 – Logo systému TinkerCAD

Jak již bylo řečeno, tento kurz není manuálem pro použití TinkerCADu, takže přistoupíme rovnou k samotnému modelování. Ještě před tím je však potřeba se k systému přihlásit a vytvořit účet. Přístup k systému je na odkazu: www.tinkercad.com

Vytvoření účtu a přihlášení

Po volbě „Přidejte se“ máme několik možností. Pro náš kurz jsou předurčeny školní účty (Lektor, Studenti), ale můžeme využít i individuální účty. Školní účty však umožňují sdružovat studentské účty do tříd a pro lektora je tak práce daleko snadnější.



Obr. 2 – Možnosti přihlášení do systému TinkerCAD

Předpokládá se, že si lektor proces přihlášení před samotným kurzem důkladně vyzkouší, tzn. projde si proces vytvoření lektorského účtu a žákovských účtů a vytvoření třídy s několika žákovskými účty.

Pracovní prostředí systému TinkerCAD

Samotné pracovní prostředí je navrženo velmi přehledně a je lokalizováno do češtiny. Po přihlášení se ocitneme v Dashboardu, odkud máme možnost otevřít své, již dříve vytvořené projekty nebo vytvořit projekt nový. Lektoři mohou vstupovat do jednotlivých tříd, z Dashboardu je možno vstoupit také do Galerie, což je poměrně rozsáhlá databáze projektů vytvořených jednotlivými uživateli.

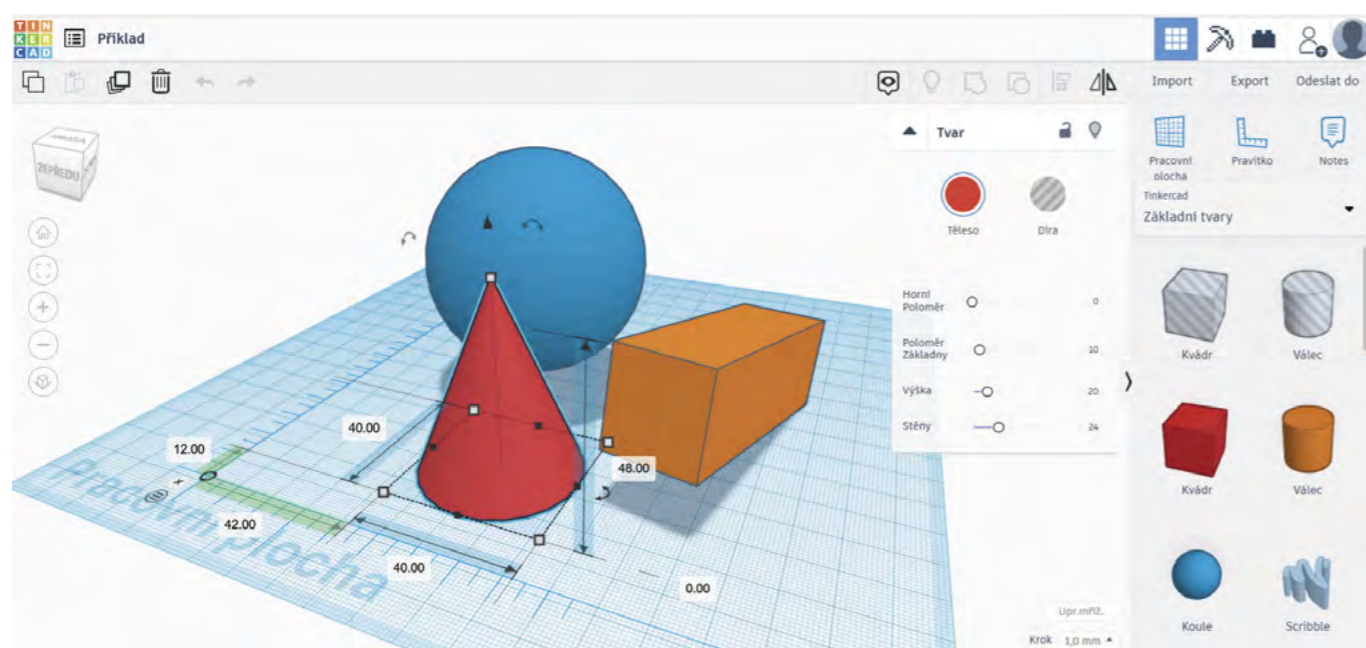
Do Dashboardu se můžeme kdykoliv vrátit klepnutím na ikonu TinkerCADu vlevo nahoře. Jakékoliv změny na modelu se po návratu do Dashboardu automaticky ukládají. Stejně jako aplikace, tak i jednotlivé modely jsou uloženy cloudově, takže k nim máme přístup z jakéhokoliv zařízení (tedy např. i z mobilního telefonu), které je připojeno k internetu.

Po volbě „Vytvořit nový projekt“ se pak již ocitneme na pracovní ploše samotného modeláře. Systém našemu projektu automaticky přiřadí název, ten je však lépe upravit podle našich potřeb. Projekt se také automaticky uloží a z Dashboardu jej pak můžeme kdykoliv znovu otevřít, případně smazat.

Některé důležité položky aneb co je dobré vědět:

- **Ovládání pohledu na model (pracovní plochu):**
 - kolečkem myši provádíme +Zoom nebo -Zoom
 - panoramování - pohybem myši při stisknutém kolečku
 - naklánění a otáčení - pohybem myši při stisknutém pravém tlačítku
 - řízení pohledu můžeme provádět i pomocí ikon v levé části obrazovky
- **V pravé části obrazovky je panel Tvary, ze kterého můžeme:**
 - vkládat na pracovní plochu základní tělesa, tvary, znaky, text apod.
 - měnit aktuální pracovní rovinu podle plochy na konkrétním tělese
 - přidat na pracovní plochu pravítko a stanovit si relativní počátek souřadnicových os
 - přidávat pracovní poznámky k jednotlivým tělesům a tvarům
- **Kliknutím na objekt dojde k jeho zvýraznění a u objektu pak můžeme:**
 - měnit jeho rozměry (tažením, posuvníkem nebo číselným zadáním rozměru)
 - zvedat jej v ose z
 - natáčet jej v jednotlivých rovinách
 - měnit jeho barvu
 - transformovat jej na „díru“
 - řídit jeho viditelnost
 - uzamknout jej před nechtěným posunutím
 - zrcadlit
- **Více objektů lze vybrat myší při zároveň stisknuté klávese Shift. Pak můžeme:**
 - objekty sdružovat do skupin
 - skupiny zpět rozdělit na jednotlivé objekty
 - provádět průniky objektů (pokud je jedním z objektů „díra“, pak průnikem vyvrtáme do objektu otvor)

Řadu dalších možností jistě zjistíte sami při práci na konkrétním modelu.



Obr. 3 – Pracovní plocha systému TinkerCAD

Import a export

Pro import hotových modelů TinkerCAD podporuje formáty STL, SVG a OBJ. Takto lze na pracovní plochu vložit dříve vytvořený model, ať už náš vlastní nebo stažený ze sdílené databáze.

U exportu je možné kromě již zmíněných formátů využít navíc ještě formát GLB. Export použijeme při 3D tisku nebo pro řezání laserovou rezačkou. Pokud vlastníme přímo podporovanou tiskárnu, můžeme model rovnou poslat do 3D tiskárny. V opačném případě musíme do tiskárny nahrát náš vyexportovaný model. Jako nepsaný standard se zde používá nejčastěji formát STL.

Samotný 3D tisk je natolik rozsáhlé téma, že by vystačil na samostatný kurz, my budeme předpokládat, že lektor již s 3D tiskem, resp. s 3D tiskárnou má určité zkušenosti. Případně může alespoň základní dovednosti získat na workshopu navazujícím na tento kurz.

3D skener

Specializovaný 3D skener je zařízení velice drahé a používá se např. při tvorbě 3D objektů v projektování nebo při měření ve strojírenství. Další možnost využití je např. při vytváření objektů pro Virtuální realitu (VR).

Nicméně, již nějakou dobu lze pro skenování jednoduchých objektů do prostředí 3D modelářů využít aplikace na mobilních telefonech nebo na tabletech. Tyto aplikace využívají integrovaný fotoaparát, se kterým je potřeba daný objekt obejít a vyfotografovat řadu snímků. Z nich pak aplikace složí a vytvoří 3D model skenovaného objektu. Pokud aplikace podporuje export do některého z výše uvedených formátů, můžeme pak naskenovaný objekt naimportovat do 3D modeláře a zde jej dále upravit nebo rovnou vytisknout na 3D tiskárně.

Aplikací dnes existuje celá řada pro různé operační systémy mobilních zařízení. Některé jsou volně k použití, za jiné je nutno zaplatit. Nejedná se ale o nijak závratné částky.

Ne každá aplikace však poskytuje použitelné výsledky. Vzhledem k tomu, že se tento obor poslední dobou poměrně bouřlivě rozvíjí, lze očekávat i rychlý vývoj těchto aplikací.

Doporučuji vyzkoušet několik aplikací a pro ukázkou předvést takovou, která vám bude vyhovovat.

Doporučené pomůcky

Pro práci s 3D modelářem

je nejlépe využít školní počítačovou učebnu s běžnými osobními počítači připojenými k internetu + 3D tiskárna. Vhodná, i když ne nezbytná, je také prezentační technika – projektor + plátno.

Pro předvedení práce s 3D skenerem – volitelná část

Předpokládá se, že bude předvádět pouze lektor – je třeba mobilní zařízení (telefon či tablet) s vysokým výpočetním výkonem a kvalitním fotoaparát. Není nezbytný přístup zařízení k internetu, záleží na použité aplikaci.

Přehled aktuálně použitelných aplikací pro mobilní telefony a tablety je k dispozici např. zde (v době vytváření tohoto kurzu – léto 2021):

<https://www.aniwaa.com/buyers-guide/3d-scanners/best-3d-scanning-apps-smartphones/>

Některé z nich jsou pouze pro platformu iOS, některé můžeme využít i na zařízeních s OS Android.



Pracovní list Práce v systému TinkerCAD – Přívěšek

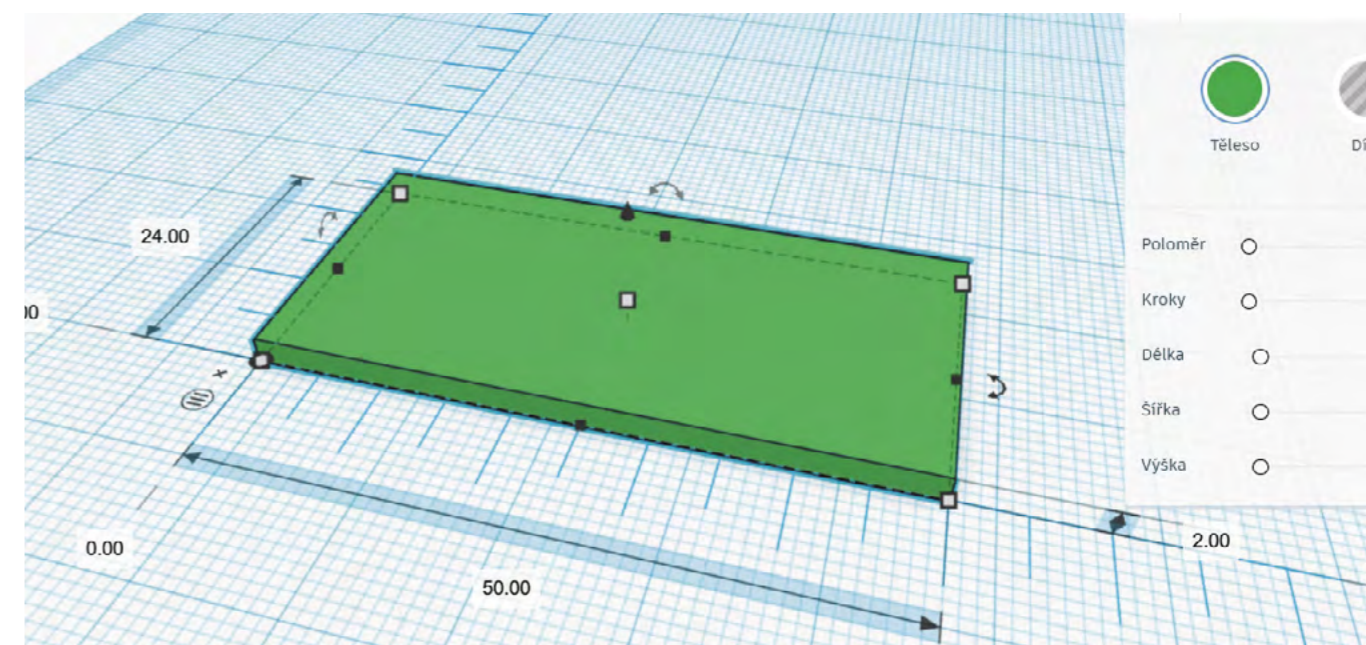
Co budeme potřebovat

- Běžný osobní počítač či notebook
- Přístup na internet
- 3D tiskárnu

A jdeme na to

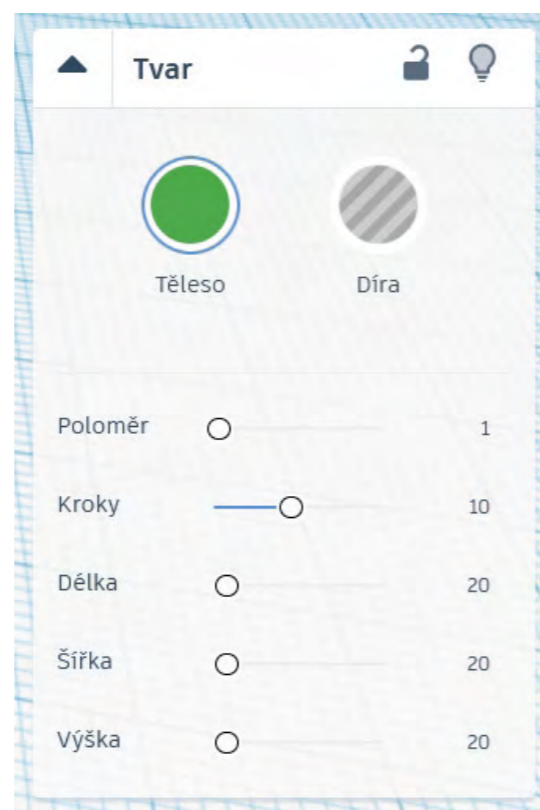
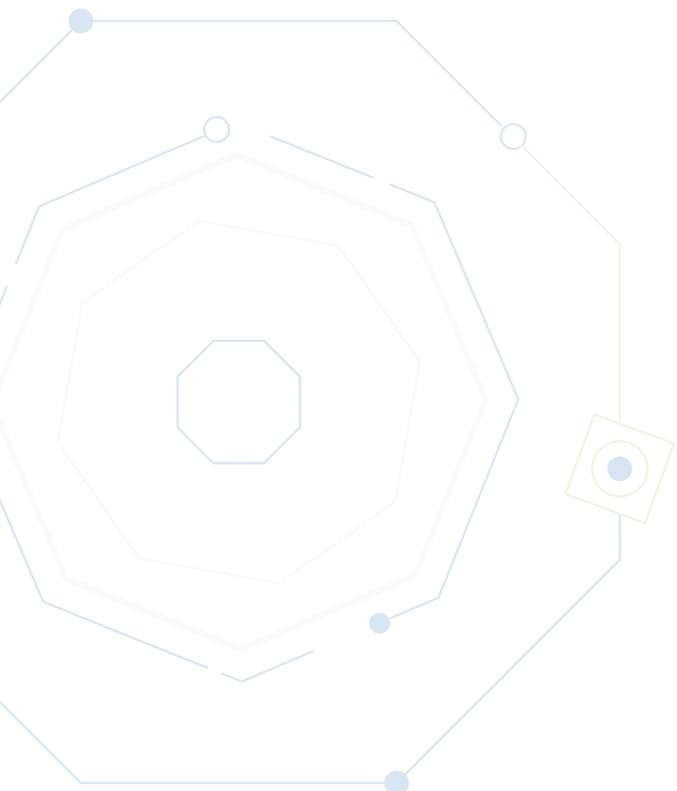
Předpokládá se, že již máte vytvořený účet, umíte se přihlásit a pohybovat se v prostředí 3D modeláře TinkerCAD. Základní dovednosti při práci si teď vyzkoušíme na příkladu vymodelování jednoduchého přívěšku na klíče. Jedná se v podstatě o podložku s očkem, na které je umístěn jednoduchý 3D text. Postup je následující:

1. Na pracovní plochu umístíme pravítko a do počátku souřadného systému (0, 0) vložíme tvar „kvádr“. Jeho rozměry nastavíme na $x = 50$, $y = 24$, $z = 2$. Záměrně neuvádíme jednotky, protože rozměry jsou relativní a význam mají až při tisku na 3D tiskárně.



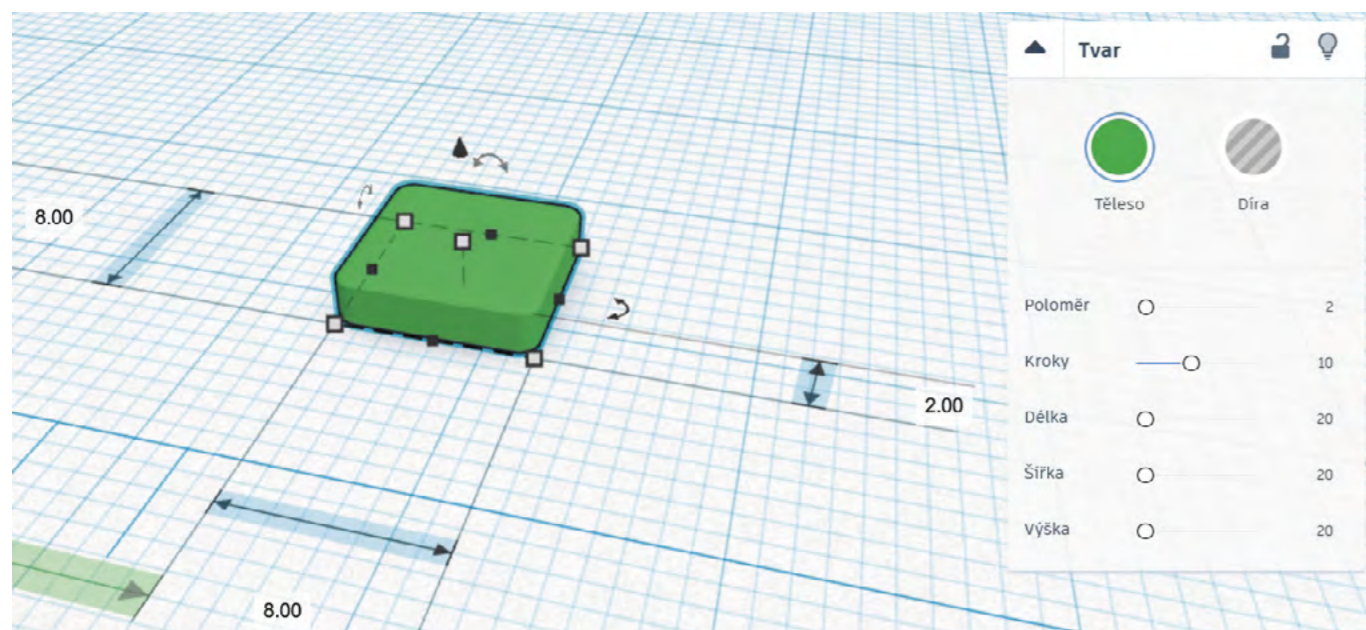
Obr. 4 – Základní tvar přívěšku

- Poloměr zaoblení rohů nastavíme na hodnotu 1 a počet kroků zaoblení na 10.



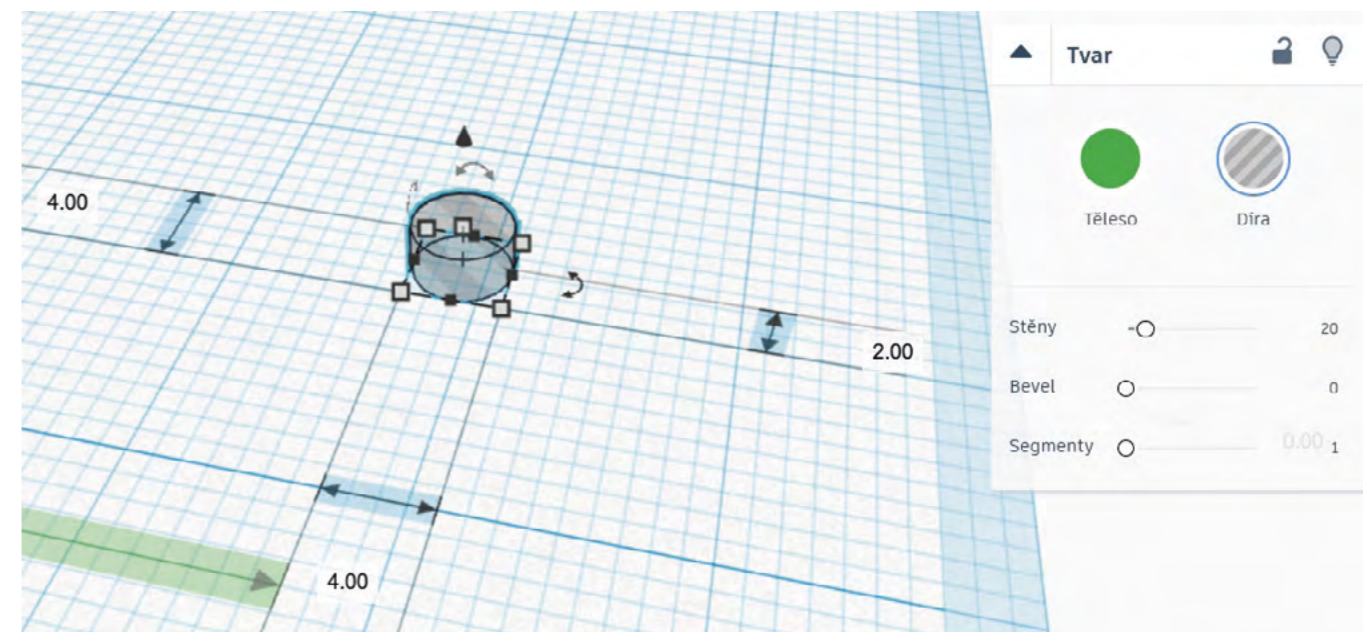
Obr. 5 – Nastavení poloměru zaoblení rohů

- Podobným způsobem jako podložku vytvoříme i základ pro očko. Zde bude poloměr 2 se stejným počtem kroků, tedy 10. Rozměry vidíte na obrázku.



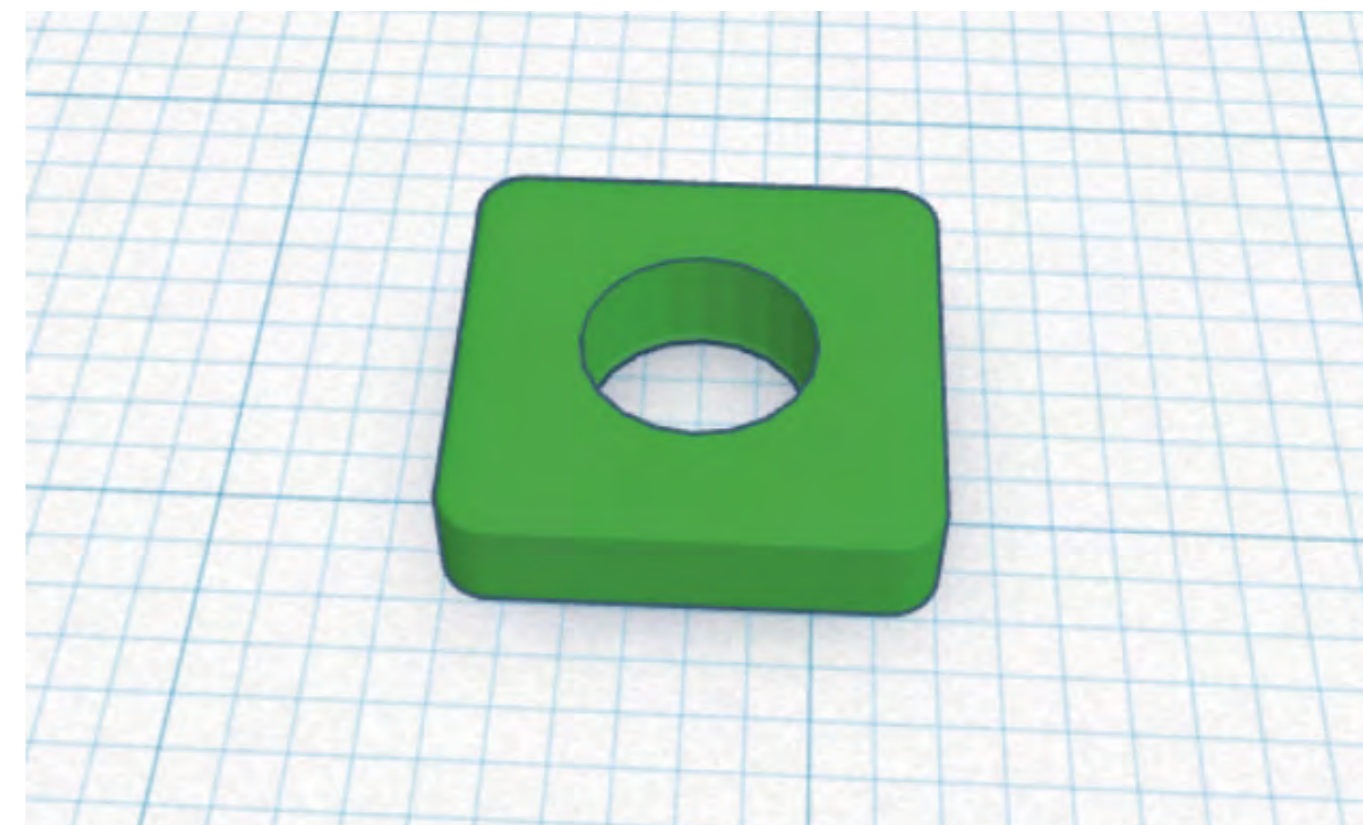
Obr. 6 – Základ pro očko

- Nyní je třeba vytvořit díru pro kroužek na klíče. Zde je třeba již uvažovat tak, že vlastní „vyvrtání“ díry provedeme jako průnik válce a podložky, přičemž válec je třeba před průnikem změnit z typu „Těleso“ na typ „Díra“. Díra má průměr 4.



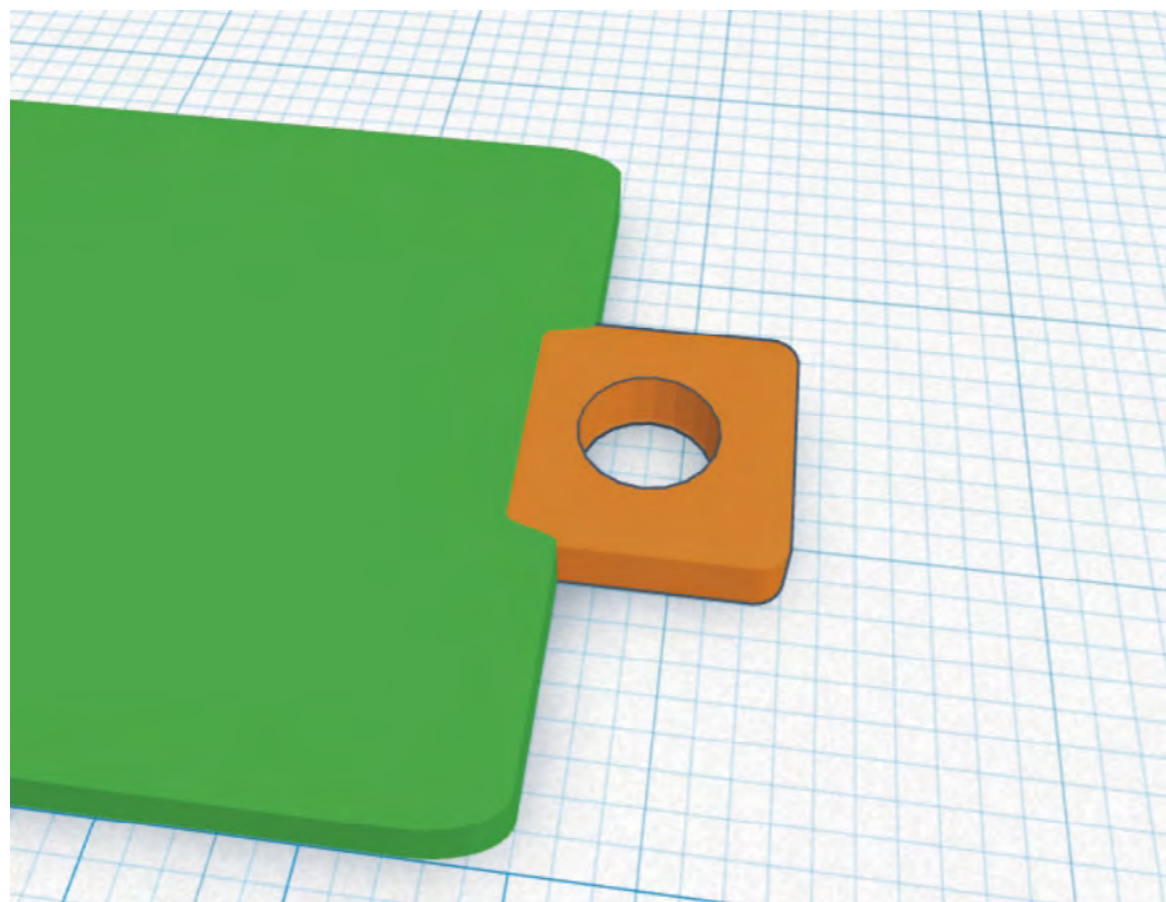
Obr. 7 – Díra pro kroužek na klíče

- Díru umístíme doprostřed základu pro očko. Pomocí klávesy Ctrl + kliknutím vybereme oba tvary a jejich sjednocením dojde k vytvoření díry.



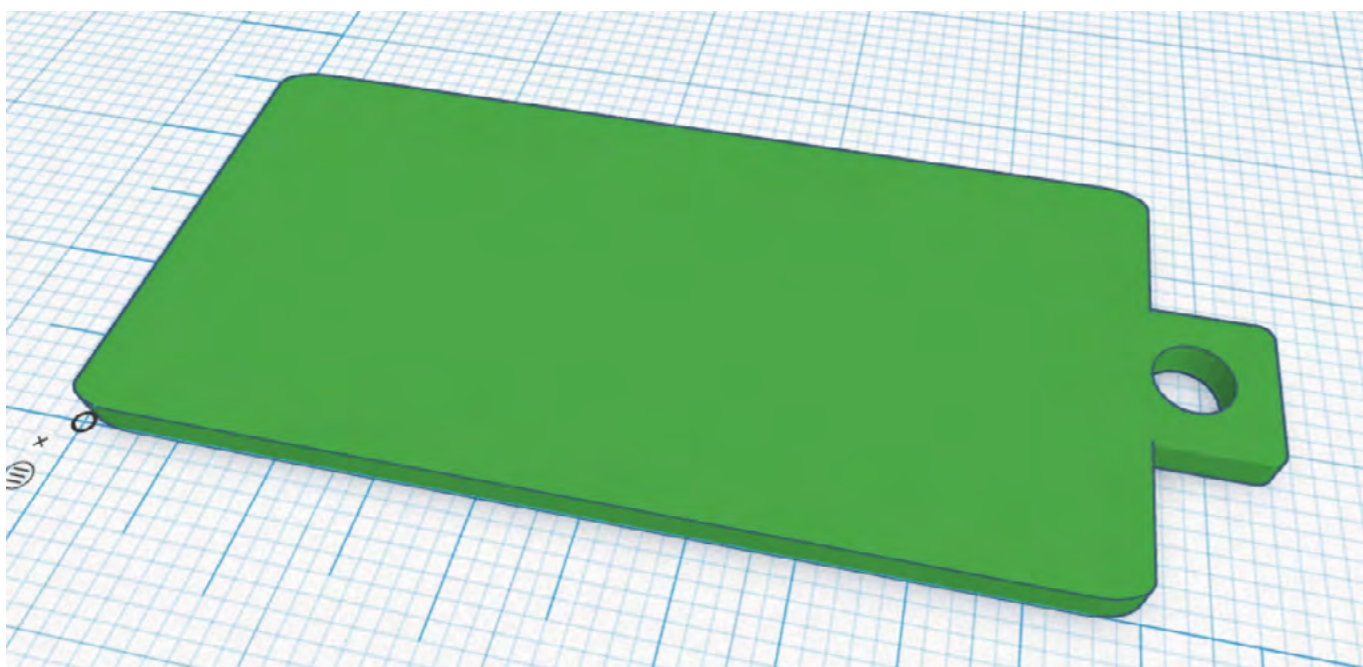
Obr. 8 – Průnik díry a základu pro očko

6. Očko poté přesuneme do příslušné pozice k podložce (raději s menším přesahem dovnitř podložky).
Pro přehlednost je očko vyznačeno oranžově, ale na výsledných barvách nezáleží, protože přívěšek bude stejně mít barvu materiálu z 3D tiskárny.



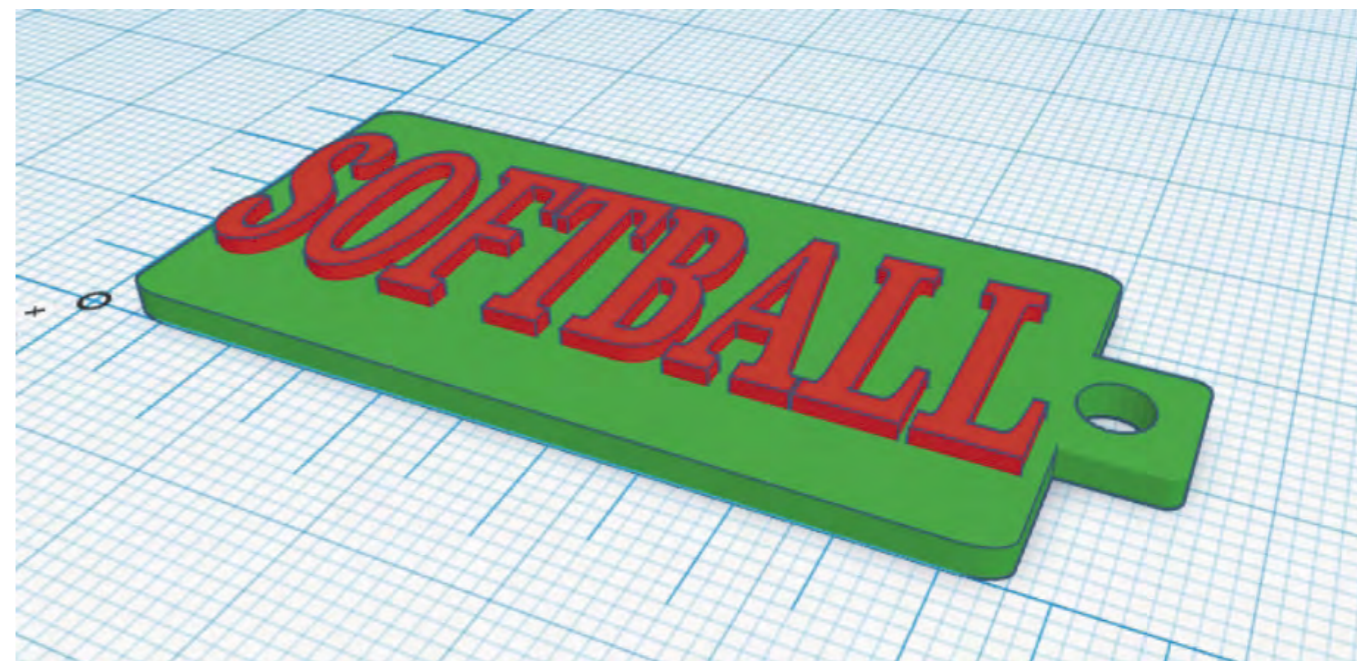
Obr. 9 – Vzájemná pozice oka a podložky

7. Sjednocením obou těles vznikne podložka s očkem.



Obr. 10 – Podložka s očkem

8. Zbývá jen vytvoření 3D textu a jeho umístění na podložku. Není třeba, aby text příliš vystupoval z podložky. Nevypadá to příliš dobře a také tisk bude probíhat kratší dobu. Tím máme model připraven pro 3D tisk, který provede učitel. Z prostředí TinkerCADu lze na některé tiskárny tisknout přímo, případně bude nutné ještě před tiskem provést export modelu do formátu STL a tento soubor pak přenést do ovladače tiskárny.



Obr. 11 – Výsledný model přívěšku na klíče

Barvy použité při modelování slouží pouze pro přehlednost při práci, výsledná barva vytisknutého modelu závisí na barvě použitého tiskového materiálu.

Co jsme se naučili?

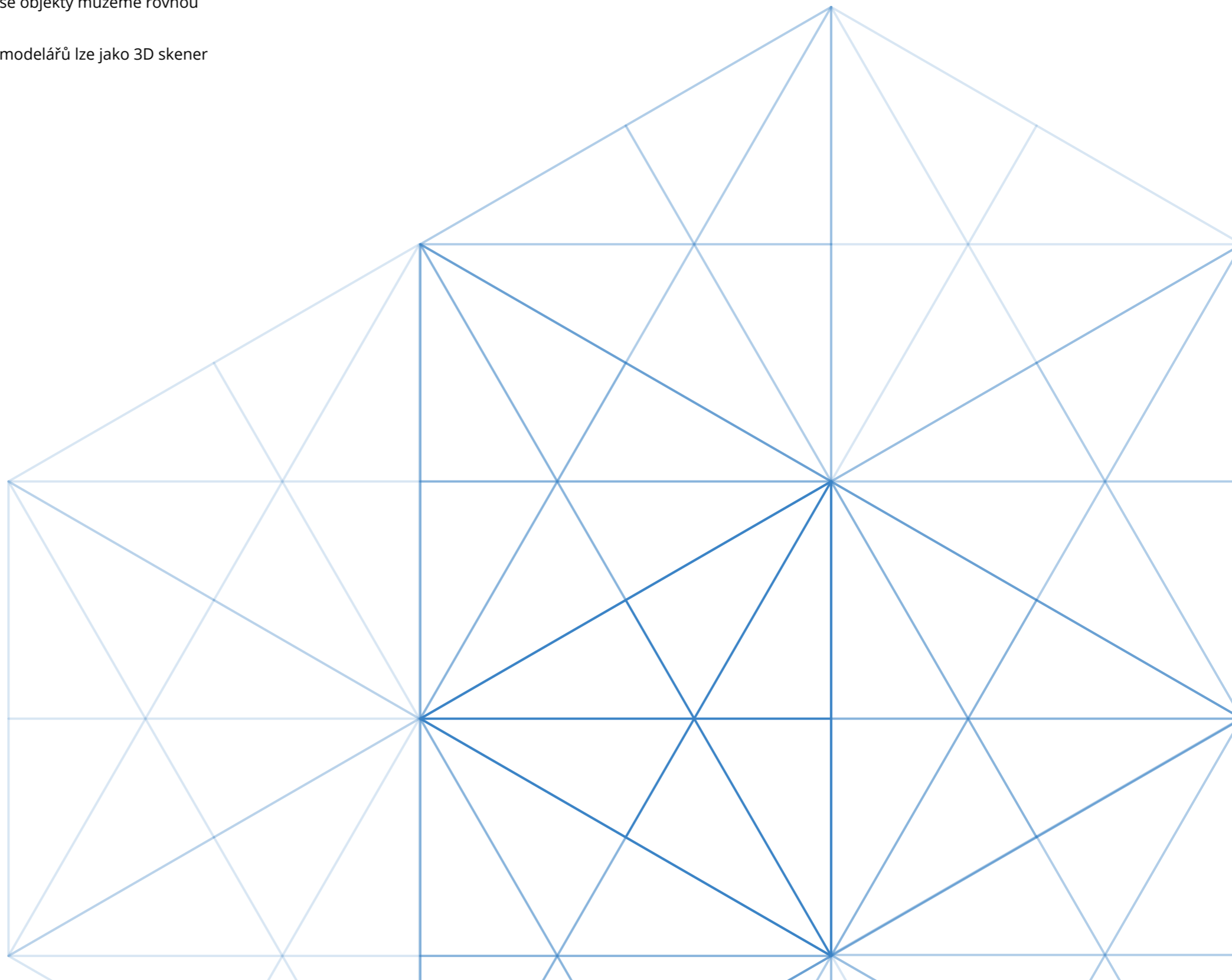
- Víme, jak se přihlásit do 3D modeláře TinkerCAD.
- Zvládáme pohyb v pracovním prostředí modeláře.
- Umíme pracovat se základními tvary a také je umíme editovat.
- Umíme vytvořit jednoduchý model a připravit ho pro tisk na 3D tiskárně.

Co umíme po absolvování celého kurzu?

- Víme, že pojem Digitální obsah je velmi široký a že každý z nás denně takový obsah pořizuje, aniž si to pokaždé uvědomuje.
- V tomto kurzu jsme pojem zúžili do oblasti 3D a umíme pomocí cloudového 3D modeláře vytvářet a editovat jednoduché 3D objekty.
- Z těchto objektů umíme připravit výstup vhodný pro 3D tisk, případně naše objekty můžeme rovnou na 3D tiskárně vytisknout.
- Dozvěděli jsme se, že pro snímání jednoduchých reálných objektů do 3D modelářů lze jako 3D skener využít i některé aplikace na mobilních zařízeních.

Použité zdroje

Obrázky v kurzu byly pořízeny jako screenshoty z aplikace TinkerCAD.



23 KYBERNETICKÁ BEZPEČNOST PRO SŠ

Základní instrukce

Tento kurz je doporučován pro studenty středních škol a vyšší ročníky víceletých gymnázií. Pokud je na ZŠ vyučována informatika intenzivně (např. již od 3. třídy), může být kurz doporučen také pro vyšší ročníky (8.+9. třída) druhého stupně ZŠ. Předpokládá se ne úplně základní orientace v online světě a běžné využívání chytrých zařízení připojených bez kontroly k internetu (smartphone, tablet, notebook apod.) Kurz je koncipován jako pokročilá problematika uvědomělého chování v online prostředí, kde bohužel díky množství informací se obezřetnost a bezpečnost dostává na druhou kolej.

Časová dotace tohoto kurzu je stanovena na 6 hodin. Rozdělení 4+2 hodiny na teoretický a legislativní základ a získání praktických dovedností v oblasti detekce podvodného obsahu. Učitel by měl nejprve prostudovat všechny materiály dané k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.

Doporučený průběh a časové dotace podrobněji. Kurz lze rozčlenit do následujících částí:

1. Teoretický a legislativní úvod – především vymezení mantinelů a pojmu kybernetická bezpečnost, zákon o kybernetické bezpečnosti a kyberkriminalita. S tím spojená identita na internetu a taktiky vylákání informací.
 - a. Doporučuji dvě vyučovací hodiny.
2. Praktické ukázky podvodných elektronických zpráv a proč na internetu platí více než kde jinde rčení „dvakrát měř, jednou řež“. Na konkrétních příkladech ukážeme, jak lze podvody identifikovat a to včetně jejich analýzy a diskuse.
 - a. Doporučuji dvě vyučovací hodiny.
3. Plnění zadaných úkolů dle již předzpracovaných kvízů či online testů. Případně dle pracovních listů vypracovaných vyučujícím.
 - a. Pro každý list či kvíz doporučuji jednu vyučovací hodinu.

Pro co největší dopad praktické části je více než vhodné zjistit, jací studenti budou v rámci výuky tohoto tématu vzdělávání a kolik veřejně dostupných informací lze o jednotlivých osobách dohledat. Pro zaslání testovacích phishingových útoků je nutné znát funkční emailové adresy studentů.



Cílem workshopu je na praktických příkladech demonstrovat důsledky nezodpovědného chování v online prostoru. Workshop bude v první, teoretické části cílen na osvětu návyků chování na internetu a to primárně v oblasti včasné detekce závadného obsahu. V části praktické pak bude konkretizováno, jakým způsobem lze závadný obsah identifikovat a jaká jsou pravidla pro práci s ním. V závěrečné části budou rozebrány důsledky nevěnování dostatečné obezřetnosti chování na internetu.

Autoři:

Ing. Rudolf Vohnout, Ph.D., Přírodovědecká fakulta, Jihočeská univerzita,

Ing. Petr Břehovský, Přírodovědecká fakulta, Jihočeská univerzita

Editor: doc. RNDr. Ing. Jana Kalová, Ph.D.

Teoretická část k dané problematice

Kybernetická bezpečnost

Kybernetická bezpečnost nemá ucelenou definici. Zásadní je si uvědomit, že vždy představuje nějaký soubor opatření vedoucí k ochraně před kriminálním či neautorizovaným užitím elektronických dat. Jinými slovy přijmutí takových opatření, která vedou k dosažení tohoto stavu. Tato opatření mohou být jak bezpečnostní, tak technické či organizační (administrativní) povahy.

*Souhrn právních, organizačních, technických a vzdělávacích prostředků, které směřují k zajištění **ochrany počítačových systémů a dalších prvků ICT, aplikací, dat a uživatelů***

+

schopnost počítačových systémů a využívaných služeb reagovat na kybernetické hrozby či útoky a jejich následky, jakož i plánování obnovy funkčnosti počítačových systémů a služeb s nimi spojených.

Dále je důležité se seznámit s kybernetickou hrozbou. Jedná se o možnost škodlivého pokusu o narušení počítačové sítě nebo systému. Kybernetickou hrozbou lze také definovat jako akt směřující ke změně informace, aplikací či systému samotného. Kybernetická hrozba může způsobit:

- Únik informace – stav, kdy dojde k vyrazení chráněné informace neautorizovanému subjektu.
- Narušení integrity – poškození, změna, či vymazání dat.
- Potlačení služby – úmyslné bránění v přístupu k informacím, aplikacím, či systému.
- Nelegitimní použití – užití informací neautorizovaným subjektem či neoprávněným způsobem.

„Ten, kdo se ve jménu bezpečnosti vzdává svobody, nezaslouží si ani svobodu, ani bezpečnost.“

Benjamin Franklin

Kybernetická hrozba a kriminalita

Kolouch a spol. (2018) kybernetickou hrozbou definuje jako akt směřující ke změně informace, aplikací či systému samotného.

Kyberkriminalitu lze definovat jako jednání namířené proti počítačovému systému, počítačové síti, **datům či uživatelům** nebo jako jednání, při němž je počítačový systém použit jako nástroj pro spáchání trestného činu.

V tomto kurzu se budeme výhradně soustředit na **Phishing**, resp. **Spearing**. Jedná se o podskupinu tzv. **Sociálního inženýrství**. To představuje techniky pro manipulování osob, což není nic jiného než vhodné aplikování psychologických metod. Typickými příznaky nátlaku jsou

- Vydávání se za autoritu – (kyber)kriminálník se snaží vzbudit dojem, že s Vámi hovoří z nadřazené pozice v projednávané záležitosti (nadřízený, policista, exekutor apod.).
- Hrozba ztráty (příležitosti) – je důvod, který Vás má přimět jednat, abyste neutrpěli ztrátu (neodtáhli Vám automobil, nezavřeli Vás do vězení apod.) nebo abyste nepřišli o příležitost si přilepšit (nevyhrajete dovolenou na Bahamách, nepovýší Vás apod.).
- Časová tíseň – (kyber)kriminálník Vás nesmí nechat přemýšlet, abyste ho neodhalili, proto Vás bude nutit reagovat rychle, pudově (do hodiny Vám zablokujeme účet, hned zítra Vás vyhodí z práce apod.).
- Utajování – (kyber)kriminálník si je vědom, že i když Vás zmanipuloval, můžete se o jeho požadavku zmínit kolegovi, který není zmanipulován a může tedy podvod snadněji prohlédnout. Často se Vás bude snažit přesvědčit, že utajování je pro Vás výhodné (nechcete, aby Vám kolegové záviděli, jde o citlivé informace, jde o státní zájem apod.)

Pokud se v jakékoliv komunikaci setkáte s výše uvedenými příznaky, velmi pravděpodobně se Vás někdo snaží podvést.

- Zdroj hrozby *způsobený člověkem*.
- Formu zavinění: *úmyslná*.

Motivace takového činu lze rozdělit:

- Za účelem získání finančního prospěchu.
- Za účelem získání konkurenční převahy.
- Za účelem dokázání svých schopností.
- Za účelem odplaty.
- Z důvodu neplnění povinností.

Základní legislativa

- Zákon č. 181/2014 Sb., o kybernetické bezpečnosti a o změně souvisejících zákonů (zákon o kybernetické bezpečnosti).
 - vyhláška č. 82/2018 Sb., o bezpečnostních opatřeních, kybernetických bezpečnostních incidentech, reaktivních opatřeních, náležitostech podání v oblasti kybernetické bezpečnosti a likvidaci dat (vyhláška o kybernetické bezpečnosti)
- Zákon č. 110/2019 Sb. Zákon o zpracování osobních údajů (aka GDPR)
- Zákon č. 40/2009 Sb., trestní zákoník, ve znění pozdějších předpisů

Doménový základ

Doručené zprávy často obsahují odkazy na webové stránky a před jejich návštěvou je dobré si rychlým pohledem zkontrolovat, zda nejsou podezřelé. Na příkladovém odkazu (URL) si nejprve ukážeme důležité části každého odkazu.

https://connect.jcu.cz/pokus/?launcher=false

https:// – definuje, jakým protokolem se stránkou náš prohlížeč komunikuje

connect.jcu.cz – doménové jméno serveru, tj. kam se připojujeme

/pokus/ – detailnější specifikace toho, co od serveru chceme získat

?launcher=false – parametry, kterými upřesňujeme, co od serveru chceme

*Kdybychom měli výše uvedené URL převést do srozumitelné věty, zněla by asi takto: Prohlížeči můj webový, prosím tě, zajdi na **web connect.jcu.cz** a dones mi data z lokality **pokus**, a když se budou ptát jestli chci **launcher**, řekni jim, že **ne**. Jo a ten server umí nářečí **https**, tak se přepni do této řeči.*

Doménové jméno je identifikátor, který se skládá z jednotlivých částí (tzv. label) oddělených tečkami. Čím více je label *vpravo*, tím je *obecnější* → geograficky si to lze představit například jako umístění ulice ve městě, státě, kontinentu, planetě. Také záleží na pořadí.

branisovska.cb.cz.europe.earth.space

Každá změna v doménovém jméně znamená, že jde o úplně jiný identifikátor, například

branisovska.plzen.cz.europe.earth.space – ulice je v jiném městě

branisovska.cb.de.europe.earth.space – ulice je v jiném státě

branisovska.cb.cz.usa.venus.space – ulice je na jiném kontinentu

branisovska.cb.cz.europe.venus.space – ulice je na jiné planetě

cb.branisovska.cz.europe.earth.space – jde o ulici CB ve městě branisovska



Co je a jak rozeznat phishing

Volně jej lze definovat jako činnost, při které se útočník záměrně snaží vylákat osobní údaje (většinou přístupové, či finanční) za účelem jejich zneužití. Toho dosahuje buď vydáváním se za někoho jiného (píše ti tvůj učitel, že máš pětku) či zneužitím důvěřivosti předstíráním organizace (píše ti tvá škola, že máš ředitelskou důtku). V drtivé většině se jako medium používá elektronická pošta (email).

Jinými slovy se jedná o sociální inženýrství aplikované do zpráv elektronické pošty s cílem získat přihlašovací údaje (jméno a heslo) se označuje pojmem phishing.

Existují také specializované formy Phishingu:

- **Spearing** (či spear phishing) – je sofistikovaný útok využívající interní informace o službách, lidech či majetku dané organizace k provedení cíleného phishingového útoku. Bude podrobně rozebrán v příkladech z praxe.
- **Pharming** – technicky pokročilý útok manipulací s DNS serverem, kdy útočník získá kontrolu nad překlady doménových jmen na IP adresy. Přesměrování probíhá na servery kontrolované útočníkem.

Emailová adresa

Překontrolujte doménu a jméno. Generické adresy jsou podezřelé.

Hlavička

Kompletní hlavička emailu popisuje, přes jaké servery byl email doručován. Neobvyklé servery mohou znamenat phishing.

Oslovení

Jak Vás obvykle odesílatel oslovuje? Liší se nějak oslovení od obvyklého?

Rozloučení

Neliší se rozloučení od obvyklého?

Odkazy

Postavte myš na odkaz aniž byste klikali a podívejte se, zda není podezřelý.

Přílohy

Přílohy od podezřelých odesílatelů jsou většinou škodlivé.

Forma

Je dopis správně česky?

Snaží se Vás pisatel dostat do časové tísně?

Příliš dobré, nebo naopak příliš špatné zprávy.

Čas odeslání

Útočníci často pracují v jiném časovém pásmu. Pokud je tedy čas odeslání v hlubokých nočních hodinách, může to být podezřelé.

Kyberkriminálníci (až na velmi vzácné případy) nemůžou použít doménové jméno serveru, za který se vydávají, takže volí obvykle některou z následujících strategií.

1. případ:

Vůbec se nesnaží skrývat a doufají, že na URL se nikdo nedívá; pak se lze setkat například s takovými adresami:

`http://1.1.1.2/login.php` – použití IP adresy v odkazu

`https://super.domain.eu?login=ap@jcu.cz` – zcela náhodná doména

2. případ:

Předpokládají, že uživatel neví, jak se správná doména jmenuje, a zkusí něco, co vypadá podobně:

`https://jcu.eprihlasi.cz/login` – jiné pořadí jednotlivých částí domény

3. případ:

Pozornější uživatele zkusí ošálit použitím textu, který odpovídá doménovému jménu, v jiné části URL:

`http://shortdomain.cz/login.jcu.cz` – doména je ve specifikaci cesty

`http://www.ucj.cz?param=login.jcu.cz` – doména je v parametru

4. případ:

Použijí doménové jméno, které je podobné, a doufají, že to uživatel přehlédne:

`https://login.jcuu.cz` – doména se velmi podobá

`https://login-jcu.cz` – vložená pomlčka zcela mění význam

`https://login.jcu.cc` – změna v jednom písmenu

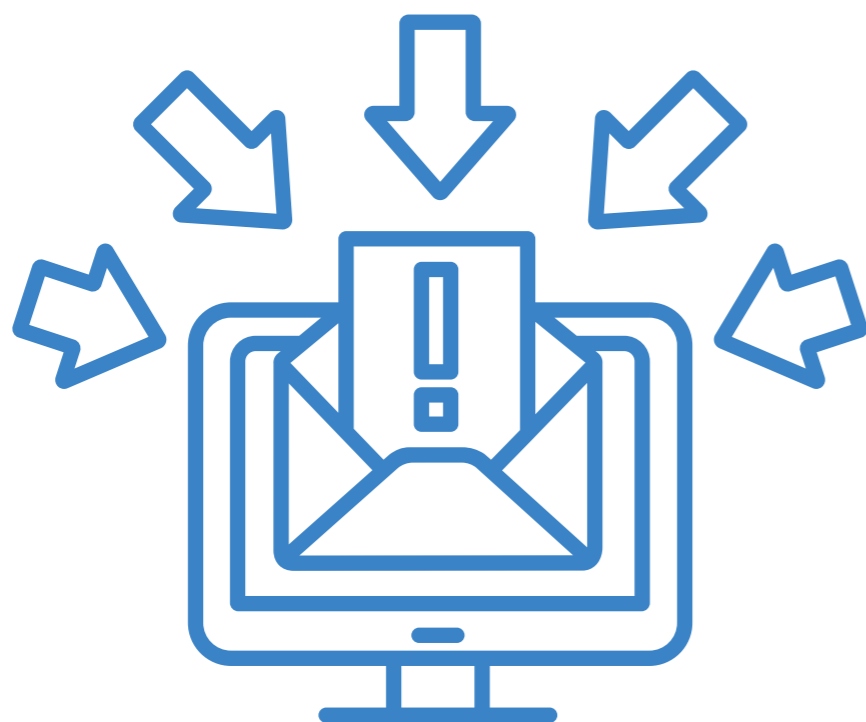
Pokud už stránku navštívíme, může nám pomoci také skutečnost, zda stránka používá zabezpečené spojení (https). Zde je třeba provést kontrolu certifikátu. Pokud se jedná o „Let's Encrypt“ je třeba se mít na pozoru.

Existují také specializované iniciativy, které se problematikou phishingu zabývají. Neznámější je asi „Anti Phishing Working Group“

<https://apwg.org/about-us/>

Doporučené odkazy a materiály pro hlubší porozumění problematiky

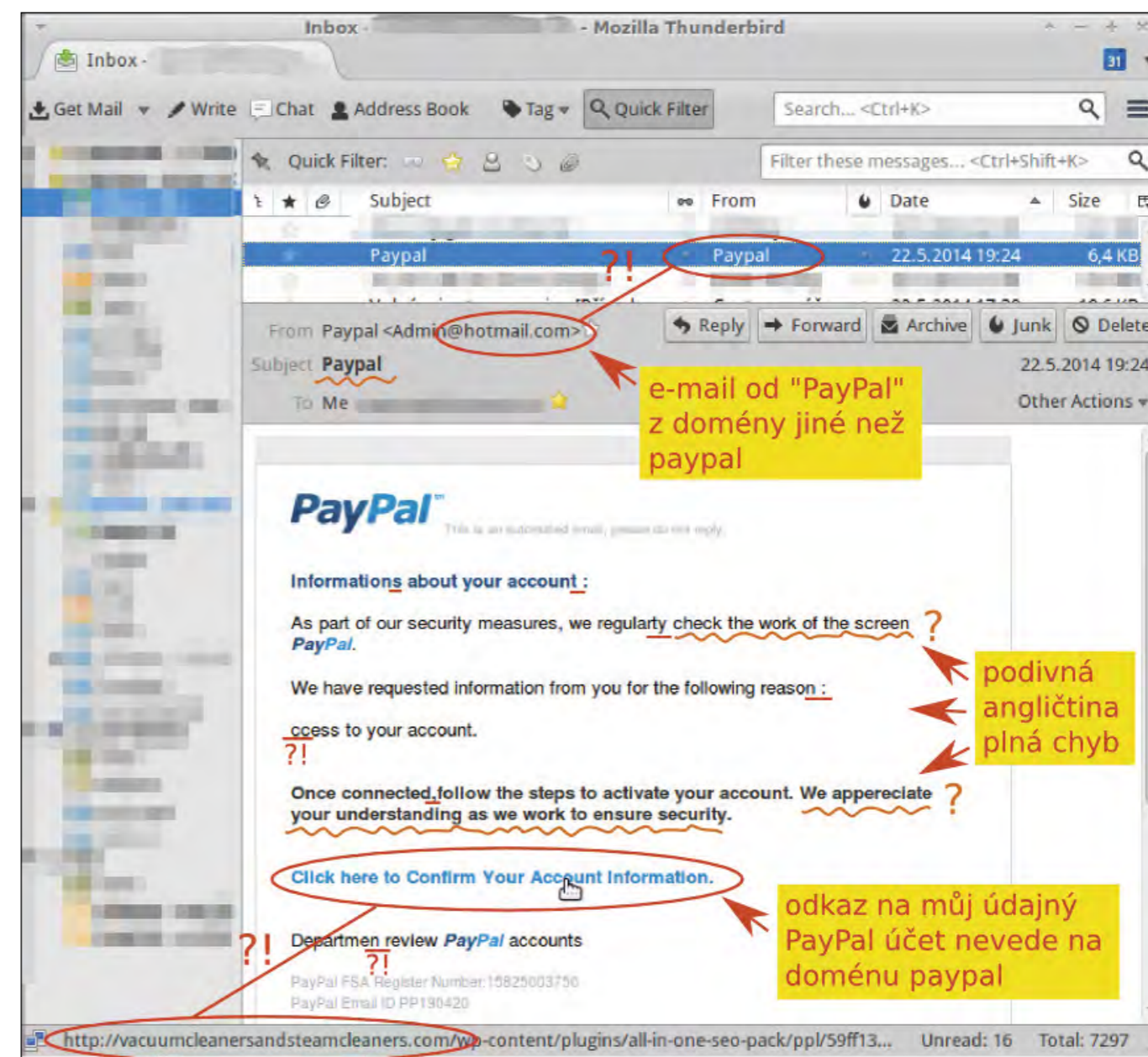
- [1] <https://www.eset.com/cz/blog/prevence/phishing-stoji-za-tretinou-pruniku-jak-poznat-skodlive-e-maily/>
- [2] <https://edu.ceskatelevize.cz/video/119-phishing>
- [3] <https://kyberbezpecnost.csirt.cz/cs/kyberbezpecnost/pro-uzivatele/phishing-jak-jej-vcas-rozpoznat-a-venaletet/>



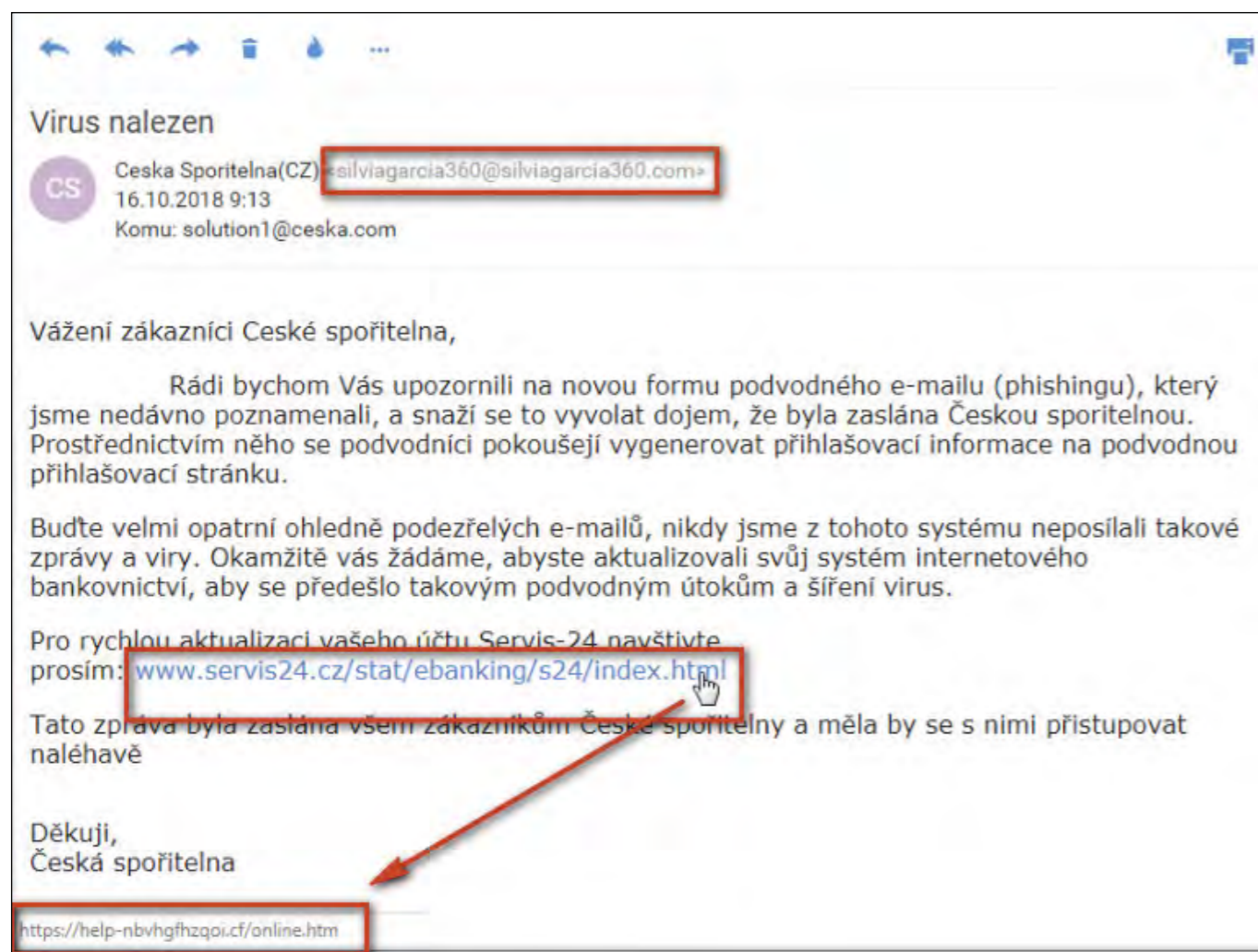
Příklady z praxe

Níže uvádíme konkrétní příklady z praxe podvržených emailů včetně podrobného rozboru, jak je lze identifikovat.

Jak si můžete povšimnout, nejzásadnějším vodítkem je odkaz na nevalidní doménu. U těch více amatérských je také adresa odeslání, ta, které nepatří doména dané instituce (níže tedy Paypal a Česká spořitelna).



Obrázek 1: Paypal Phishing

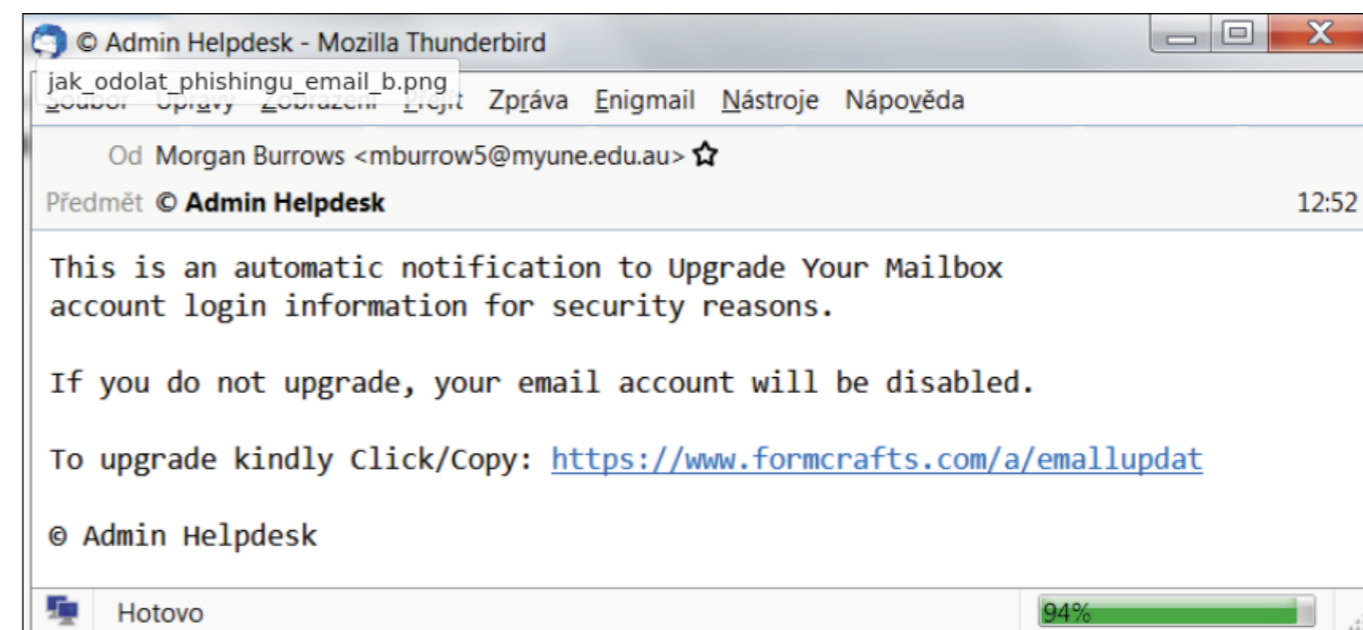


Obrázek 2: Česká spořitelna Phishing

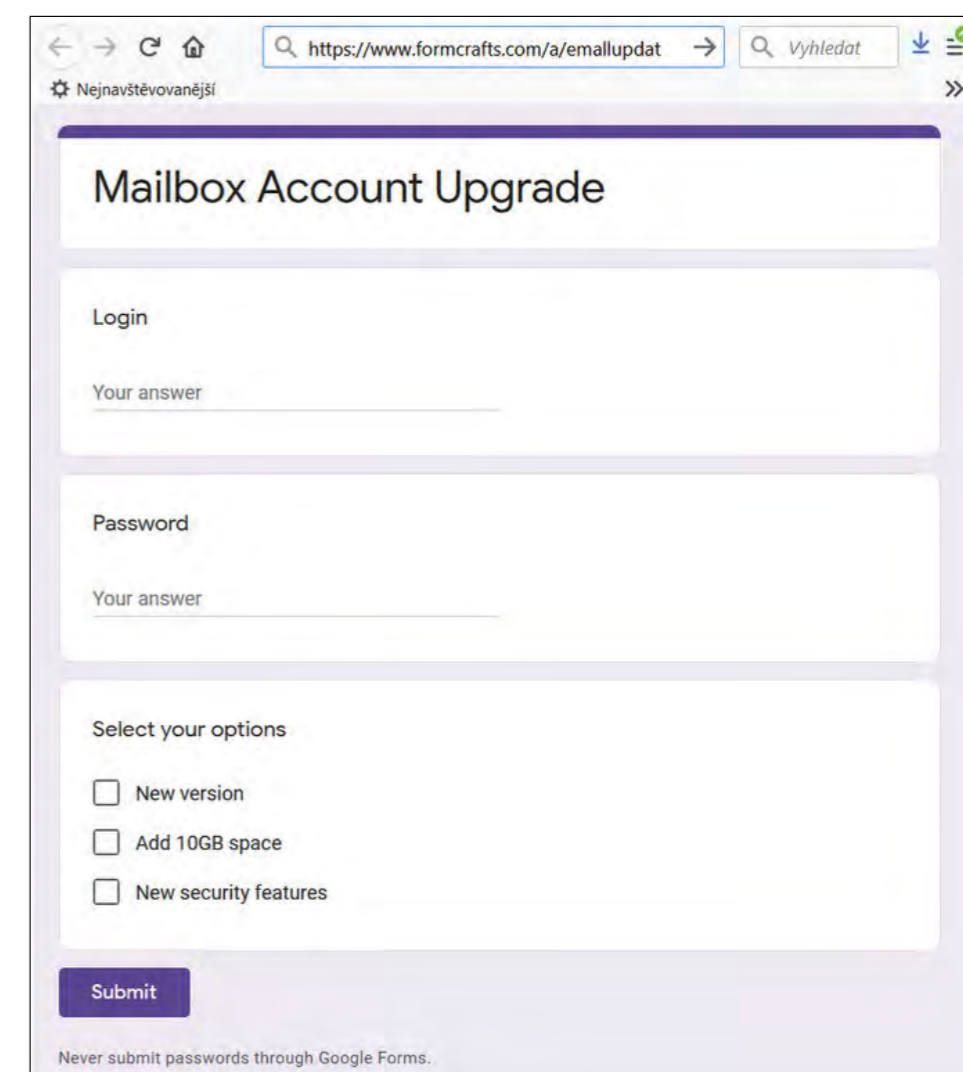
Podvodné zprávy se závadnou přílohou (typicky malware) již nejsou tak časté. V drtivé většině případů je totiž odhalí poštovní brány a antivirová řešení.

Co ovšem časté je, jsou typicky odkazy na formuláře, které vypadají jako skutečné (typicky Google nebo Microsoft Forms) a jen tak mimoděk po Vás také chtějí jméno a heslo.

Začíná být stále více běžnou praxí, že si firmy (ale také školy) testují interně své zaměstnance za účelem zjištění jejich schopnost identifikovat podvržené emailové zprávy. Tomuto kroku ale většinou předchází (většinou povinně) školení. Jeho absolvování bývá stvrzeno podpisem. Ne vždy ovšem takové školení přinese kýžený efekt (zaměstnanci neposlouchají, nudí se apod.). Proto se potom přistupuje právě k testování v reálném prostředí.



Obrázek 3: Google Forms Phishing



Níže je uveden jeden takový příklad. Pod ním bude rozebráno, jak lze identifikovat, že se jedná o podvrženou zprávu.



Obrázek 4: CESNET Spearfishing

Obrázek 3 představuje už celkem sofistikovaný spearfishing. Ve společnosti CESNET totiž:

- Mají systém bezúročných zaměstnaneckých půjček.
- Existuje důležitý interní systém Řešitel.
- Komerční banka je hlavní bankou společnosti.

Bližší analýza:

- Text neobsahuje žádné pravopisné chyby ani překlepy.
- Doména „resitel-cesnet.cz“ skutečně existovala – je však založena pouze za účelem tohoto spearfishingového útoku. Předpokládá se, že každý uživatel internetu ví, že jednotlivé úrovně domén se odlišují tečkami.
- O žádnou půjčku jste nežádal.

Účelem je ze zaměstnanců vylákat jejich přístupové údaje (i když v tom případě pouze pro testovací účely) – pokud je zaměstnanec zadá správně, zobrazí se mu stránka s informací, že se stal obětí podvodné emailové zprávy, ale naštěstí pouze testovací a nemusí se tak obávat následků.

U phishingu a spearfingu je vždy nutné věnovat analýze takových zpráv čas, pokud se Vám něco na první pohled nepozdává.

Relevantní odkazy

[1] https://support.zcu.cz/index.php/Phishing_-_p%C5%99%C3%ADklady

Pro ty, co si chtějí rozšířit obzory:

<https://knihy.nic.cz/files/edice/cybersecurity.pdf>

<https://support.zcu.cz/index.php/Phishing>

<https://www.csfd.cz/film/812238-socialni-dilema/>

<https://www.documentaryarea.tv/player.php?title=The%20Social%20Dilemma>



Metodická a didaktická část

Na úvod je vhodné uvést, že pro tento kurz nejsou připravovány žádné pracovní listy. Je totiž žádoucí, aby si je připravil sám vyučující na základě mnoha odkazů v tomto materiálu. Dále je mnohem flexibilnější, pokud studenti obdrží pracovní listy v elektronické podobě ve formátu PDF než vytištěné. Přepisováním případných odkazů do internetového prohlížeče může dojít k překlepům a tudíž zbytečnému zdržování.



Phishingové a kyberbezpečnostní testy

1) Nejedná se přímo o pracovní listy, ale berte tuto sekci jako námět k jejich vytvoření. Zkuste, ať si studenti zkusí udělat online test zdali dokáží podvodné zprávy sami rozpoznat. Poté konzultujte s nimi jejich úspěšnost a nechte je v případě neúspěchu mezi sebou probírat jednotlivé příklady.

[1] <https://phishingquiz.withgoogle.com/>

[2] <https://www.sonicwall.com/phishing-iq-test/>

2) Jako ještě sofistikovanější doporučujeme personalizovaný phishingový test z

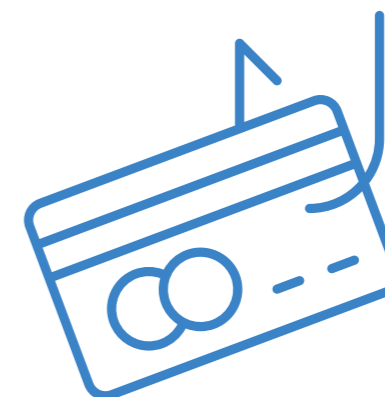
<https://phishing-iq-test.com/>

3) Pro pokročilé procvičení kybernetické bezpečnosti velmi doporučujeme absolvovat a do pracovních listů zahrnout komplexní online test, který podporuje Česká bankovní asociace a Policie České republiky.

<https://kybertest.cz/>

Velmi doporučujeme zařadit do výuky (a také jako inspiraci pro tvorbu pracovních listů) materiály z <https://o2chytraskola.cz/>

Jedná se o kvalitně zpracované materiály na témata od základního chování v online světě až po sofistikované Cyber(security+crime) materiály. Nespornou výhodou jsou již existující testy ke každému tématu chování.

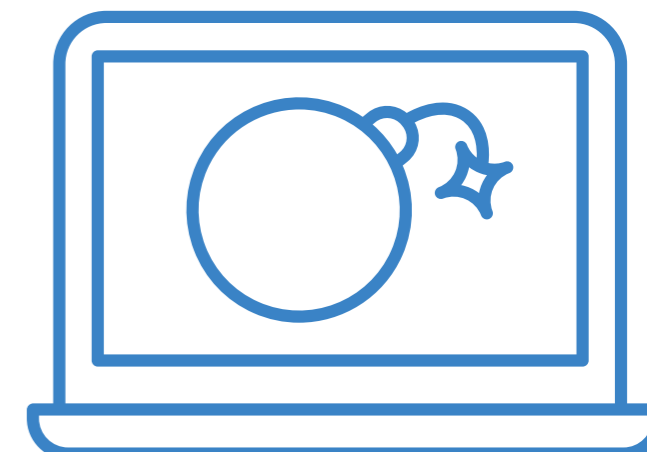


Doporučené pomůcky

- Každý žák svůj počítač s (rychlým) připojením k internetu umožňujícím paralelně streamovat videa.
- Na PC s operačním systémem Windows bude funkční prohlížeč internetových stránek.
- Sluchátka a správně nastavený zvukový výstup.
- PDF verze pracovních listů s interaktivními formuláři.
- Volitelně (pro ukládání PDF pracovních listů) 20 ks USB flashek.

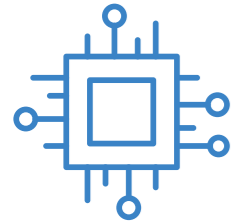
Studentům řekněte, že je více než vhodné si dělat poznámky.

Pro správné zvládnutí úkolů je žádoucí mít středně pokročilou znalost anglického jazyka. Odkazy na online phishingové testy a další materiály jsou často v angličtině. I přesto, že předpokládáme, že dojde v úvodu hodiny k objasnění zadání a v pracovním listu bude také objasněno, lze tímto předejít případným dotazům. Vysvětlete studentům, že znalost anglického jazyka je pro dnešní svět (a informatiku) zásadní.



**METODIKA ZÁKLADY
PROGRAMOVÁNÍ**

BLOKOVÉ PROGRAMOVÁNÍ PRO SŠ



Cílem workshopu Metodika Základy programování/blokové programování je naučit posluchače "programátorsky" myslet. Začínáme základy algoritmizace, formulací problému, vytvořením algoritmu, nakreslením vývojového diagramu a konečně sestavením a odladěním programu. V každé části jsou vždy části teoretická a praktická, kde jsou jak příklady vyřešené učitelem, tak příklady pro posluchače k procvičení. Nakonec je zařazen vždy vzorový příklad, který by posluchači měli vyřešit samostatně nebo ve dvojici. Programovací jazyk zde není podstatný, pro svoji názornost a jednoduchost byl v praktických příkladech zvolen jazyk Python v.3.

Základní instrukce

Tento kurz je doporučován pro žáky středních škol. K tomu dokumentu je připraven materiál v prostředí Jupyterhub a/nebo Jupyter Notebook, kde si jednotlivé úlohy budou moci posluchači vyzkoušet. Toto prostředí bude posluchačům k dispozici odkudkoliv s internetovým připojením. Pokud budou chtít posluchači použít vlastní počítač, předpokládá učitel předinstalovaný základní programovací jazyk Python ve verzi 3.3 nebo vyšší.

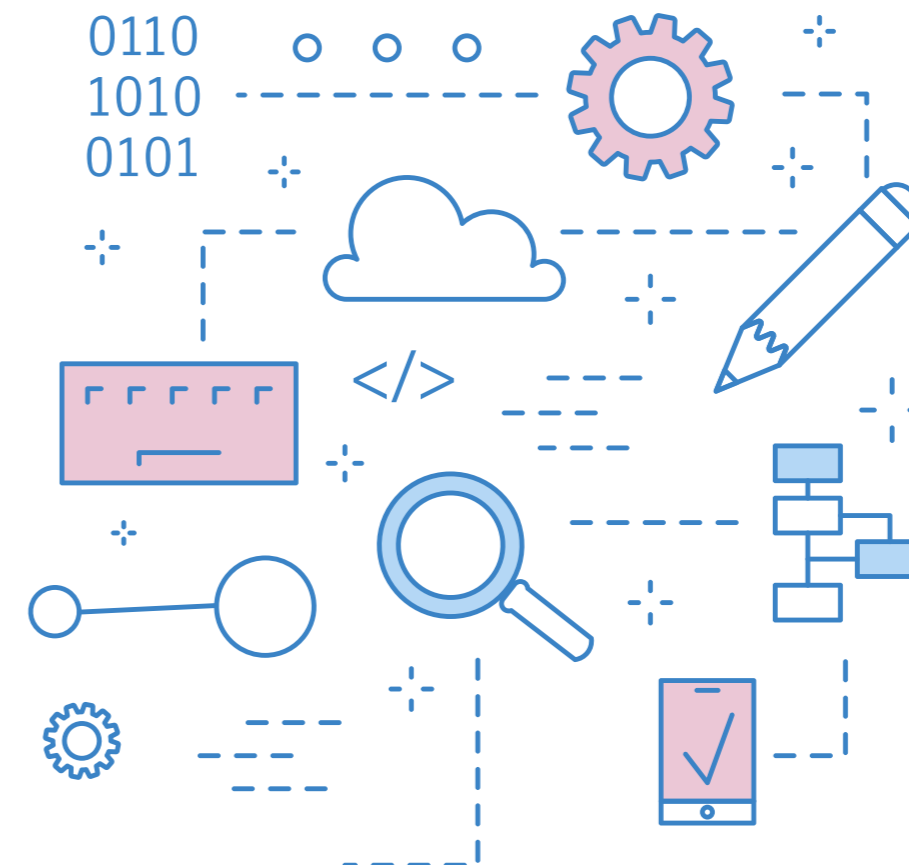
Znalost základů programování jakéhokoliv programovacího jazyka je žádoucí, nikoliv však nutná.

Časová dotace tohoto kurzu nemůže být zcela jednoznačná. Závisí na tom, zda se studenti již seznámili s programováním a zda již mají obecně nějaké zkušenosti s programováním, případně s Pythonem. Učitel by měl nejprve prostudovat všechny materiály dané k tomuto kurzu a pak se sám rozhodnout, kolik času kurzu věnovat.

Metodika používá jako zdroj informací

- Programátorskou cvičebnici Algoritmy v příkladech od Radka Pelánka
- Programátorské kuchařky MatfyzPress 2011
- Výuka algoritmizace na ZŠ – Bc. Jana Bromová
- <http://www.ivt.mzf.cz/algoritmizace-a-programovani/uvod-do-algoritmu>

Základní znalost programování není nutná, ale je žádoucí. Programovací jazyk není určen, v podstatě je možné použít jakýkoliv, v příkladech této metodiky byl použit jazyk Python pro jeho názornost.



Blokové programování

Blokové programování vychází z blokové struktury programu. To je v IT označení pro zdrojový kód programu rozčleněný do samostatných bloků, které ho rozdělují na souvislé logické části nebo samostatné funkční části u funkcí a procedur. Blokovaná struktura se hojně využívá hlavně v moderních technikách programování, které se označují také jako strukturované jazyky.

Poznámky

- Ačkoliv je v příkladech použit pro názornost jazyk Python, není toto výuka Pythonu.
- V učebnici, která byla zvolena jako základ najdete i obtížné kategorie, které v této metodice zásadně neuvádím.
- Pro žáky volte takové příklady, které složitostí odpovídají jejich možnostem a znalostem a které stihnou s přehledem udělat v určeném čase.

Začínáme

Kategorie obtížnosti

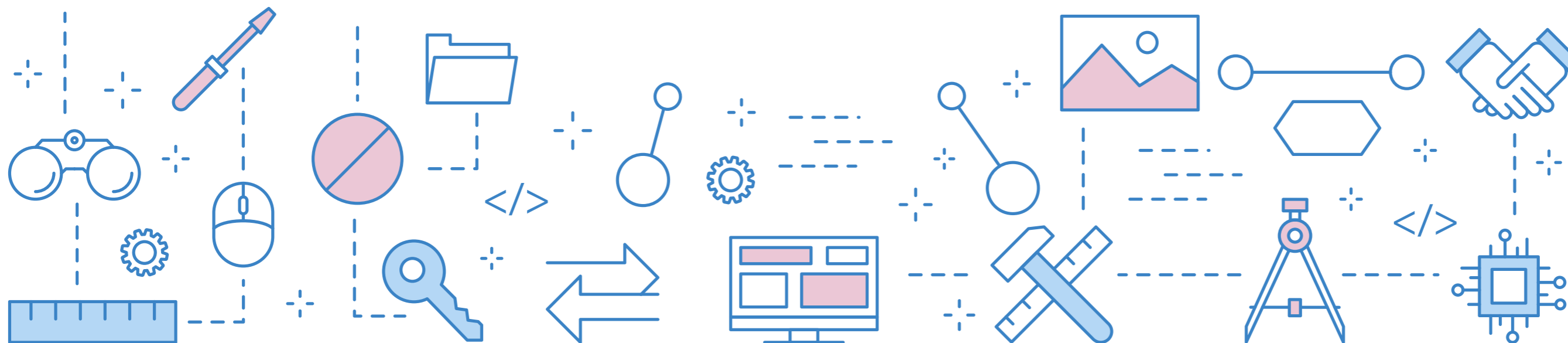
Kategorie obtížnosti jsou subjektivní.

V roli experta jste vůči žákům vy, vždy uvažujte s tímto poměrem

Nápad	Začátečník	Expert	Kódování	Začátečník	Expert
1	Vcelku jasné	Zřejmé	1	20 minut	5 minut
2	Trocha rozmýšlení	Rutina	2	1 hodina	Rutina
3	Těžké	Trocha rozmýšlení	3	Půl dne	1 hodina

„Mysl není nádoba, kterou je potřeba naplnit, ale oheň, který je potřeba vznítit.“

Plutarchos



Algoritmy

Základní pojmy

Algoritmizace

Algoritmizace je přesný postup, který se používá při tvorbě programu pro počítač, jehož prostřednictvím lze řešit nějaký konkrétní problém.

Algoritmizaci lze rozdělit do několika kroků:

- Formulace problému
- Analýza úlohy
- Vytvoření algoritmu
- Sestavení programu
- Odladění programu

Formulace problému

V této etapě je třeba přesně formulovat požadavky, určit výchozí hodnoty, požadované výsledky, jejich formu a přesnost řešení. Tvůrce algoritmu musí dokonale rozumět řešenému problému, jinak nemůže algoritmus sestavit – v praxi programátoři spolupracují s odborníky z oblastí, pro které mají vytvořit algoritmus.

Analýza úlohy

Při analýze úlohy si ověříme, zda je úloha řešitelná a uděláme první návrh řešení. Definujeme způsob vstupu a tvar vstupních hodnot. Zjistíme, zda je úloha řešitelná a za jakých podmínek a zda má případně více řešení. Pokud má úloha více postupů, jak ji řešit (což je u složitějších úloh běžné), zvolíme jedno řešení, případně si zdůvodníme, proč právě toto řešení bylo zvoleno. Dále pak již postupujeme podle něj.

Vytvoření algoritmu úlohy

Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy. Pojem algoritmu se nejčastěji objevuje při programování, kdy se jím myslí teoretický princip řešení problému (oproti přesnému zápisu v konkrétním programovacím jazyce). Obecně se ale algoritmus může objevit v jakémkoli jiném vědeckém odvětví.

Vývojový diagram (flowchart)

Vývojový diagram je druh diagramu, který slouží ke grafickému znázornění jednotlivých kroků algoritmu, pracovního postupu nebo nějakého procesu. Vývojový diagram obsahuje obrazce různého tvaru (obdélníky, kosočtverce, aj.), navzájem propojené pomocí šipek. Obrazce reprezentují jednotlivé kroky, šipky tok řízení. Vývojové diagramy standardně nezobrazují tok dat, ten je zobrazován pomocí data flow diagramů. Vývojové diagramy jsou často využívány v informatice během programování pro analýzu, návrh, dokumentaci nebo řízení procesu.

Tvorbě vývojových diagramů bude věnována jedna z dalších kapitol.

Sestavení programu

Na základě algoritmu řešené úlohy sestavíme program (zdrojový text) v konkrétním programovacím jazyce. Ze zdrojového textu se pomocí překladače do strojového kódu vytvoří spustitelný program (případně interpretem se přeloží a spustí jednotlivé příkazy programu). Lze říci, že dobře provedená analýza úlohy a algoritmizace je velmi důležitá pro řešení daného problému a je základním předpokladem sestavení programu pro počítač.

Tvorbě programů v jazyce Python bude věnována jedna z dalších kapitol.

Pro naši vzorovou ukázkou **Ciferace** se použije tzv. rekursivní volání funkce, proto si jej znázorníme již nyní

Rekursivní volání funkce

Není to nic složitějšího, funkce prostě zavolá sama sebe. Ciferace je krásná ukáзка rekursivní funkce. Podmínkou je, že součet všech čísel v čísle musí být jednočíslný. Pokud není, dosadíme za číslo součet a zavoláme funkci znovu.

Odladění programu

Naprogramováním činnost nekončí. Program musí být tzv. "odlbený". Odladěním chceme odstranit chyby z programu. Programátor musí spolu s testerem odchytil a ošetřit všechny možné, i velmi nepravděpodobné možnosti. Celá kapitola v programování je tzv. error handling tj. ošetřování možných chyb. Příkladem ošetření možné chyby je například zabránění dělení nulou, které by skončilo chybou.

Další odladování se týká např. toho, aby program běžel co nejrychleji a aby neměl velké nároky na paměť.

Nejčastější chyby jsou chyby v zápise, tzv. syntaktické – ty odhalí překladač a dělají je i zkušení programátoři. Horší jsou logické chyby, které vyplývají z nesprávně navrženého algoritmu – projeví se nesprávnou činností programu nebo špatnými výsledky – při odstraňování těchto chyb může pomoci ladící program (debugger) umožňující sledování aktuálního stavu proměnných a krokování. Teprve po odstranění všech druhů chyb můžeme program použít k praktickému řešení úloh. Důležité je následné testování. V týmu řešitelů jsou tzv. testeři.

V případě složitějších nebo komerčních úloh je tester role, která je oddělena od vývojáře, aby nevznikala profesní slepota. Testování a metodika testování je opět celá věda.

Algoritmizace v praxi

Proces psaní příkazů v programovacím jazyce se nazývá **programování**.

- **Fáze analýzy** – Co bude program umět definuje **analytik**, ten pak dává zadání pro **programátora (vývojáře)**.
- **Fáze programování** – Vývojář podle popisu vytvoří program (přepíše řešení do programovacího jazyka). Zápis programu v programovacím jazyce se nazývá **zdrojový kód**. Poté je spuštěn program, který dokáže přeložit tento zdrojový kód do jazyka počítače, a vznikne tak **spustitelný program**. Prvotní otestování programu se nazývá jeho **ladění (debugging)**.
- **Fáze testování** – Tento program dostane **tester**, který se snaží najít v programu chyby nebo nesprávné chování neodpovídající zadání. Pokud objeví chybu, pak ji předá zpět programátorovi k dořešení.
- **Fáze akceptace**, zaškolení obsluhy programu.
- **Fáze změn, údržba a podpora** – Program je předán uživatelům. Dochází k požadavkům na úpravy nebo přidání funkcí.

UML

Unified Modeling Language (UML) je grafický modelovací jazyk pro specifikaci, vizualizaci a dokumentaci informačních systémů a aplikací. UML umožňuje popsat systém pomocí slov a obrázků. Můžeme jím modelovat různé systémy.

Výčet diagramů standardu UML 2.0:

Strukturní diagramy:

- Diagram tříd (Class Diagram)
- Diagram objektů (Object Diagram)
- Diagram komponent (Component Diagram)
- Diagram složených struktur (Composite Structure Diagram)
- Diagram nasazení (Deployment Diagram)
- Diagram balíčků (Package Diagram)

Diagramy chování:

- Diagram případů užití (Use Case Diagram)
- Diagram aktivit (Activity Diagram)
- Stavový diagram (State Machine Diagram)

Diagramy interakce:

- Sekvenční diagram (Sequence Diagram)
- Diagram komunikace (Communication Diagram)
- Diagram přehledu interakcí (Interaction Overview Diagram)
- Diagram časování (Timing Diagram)

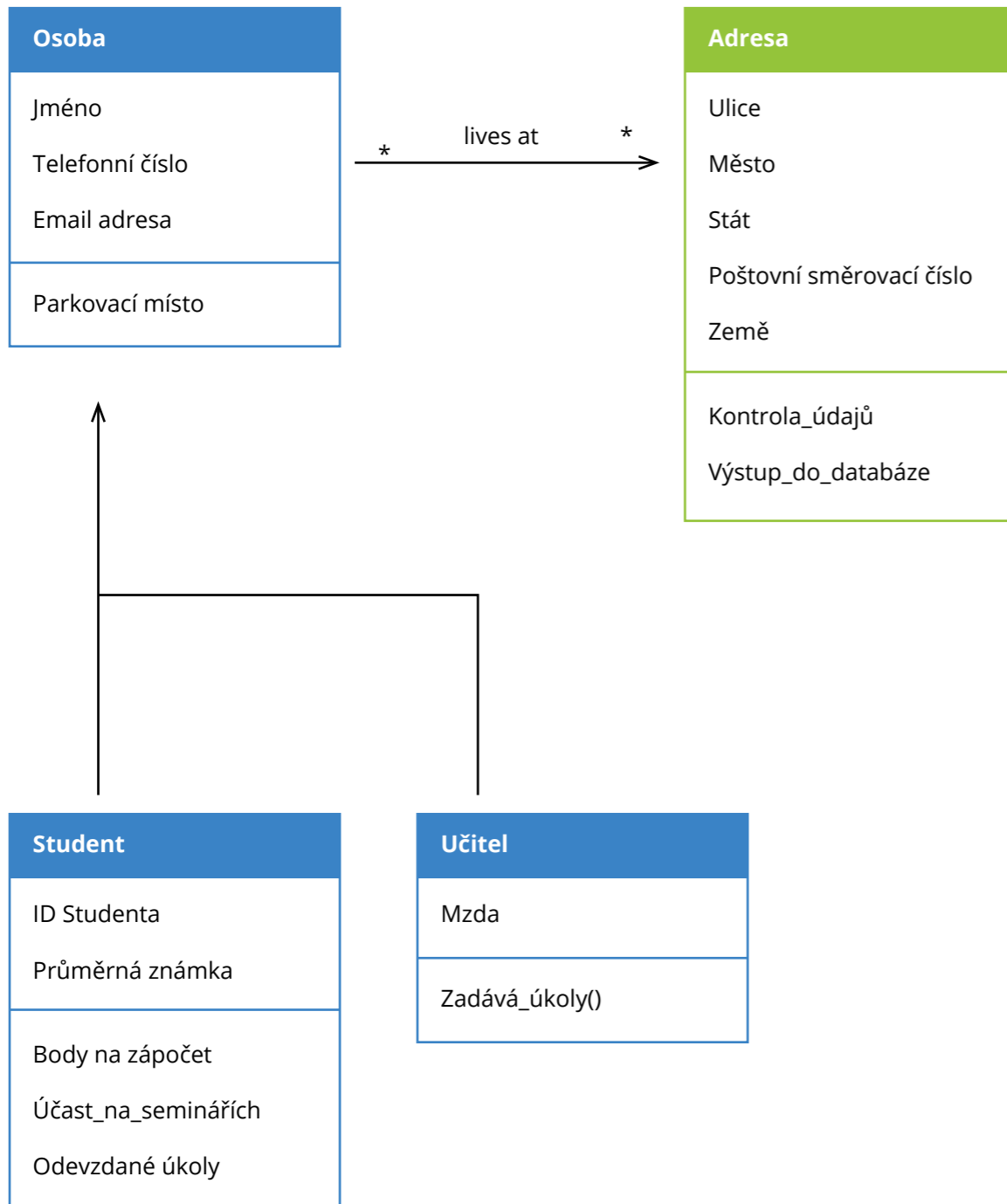
Často slyšíme pojem **diagramy tříd** (class diagrams) nebo **use-case diagramy**. UML je jazyk, který byl standardizován Object Management Group (OMG), mezinárodní asociací pro tzv. "open standards for object-oriented applications (<http://www.omg.org>)."

Základy jazyka UML a schopnost se jazykem UML vyjádřit se naučí prakticky každý. Vyjadřováním prostřednictvím UML máme jistotu, že nám ostatní porozumí.

Aktuálně se používá UML verze 2.0, ("OMG Unified Modeling Language: Superstructure, Version 2.0, Revised Final Adopted Specification, October 2004", from <http://www.omg.org>).

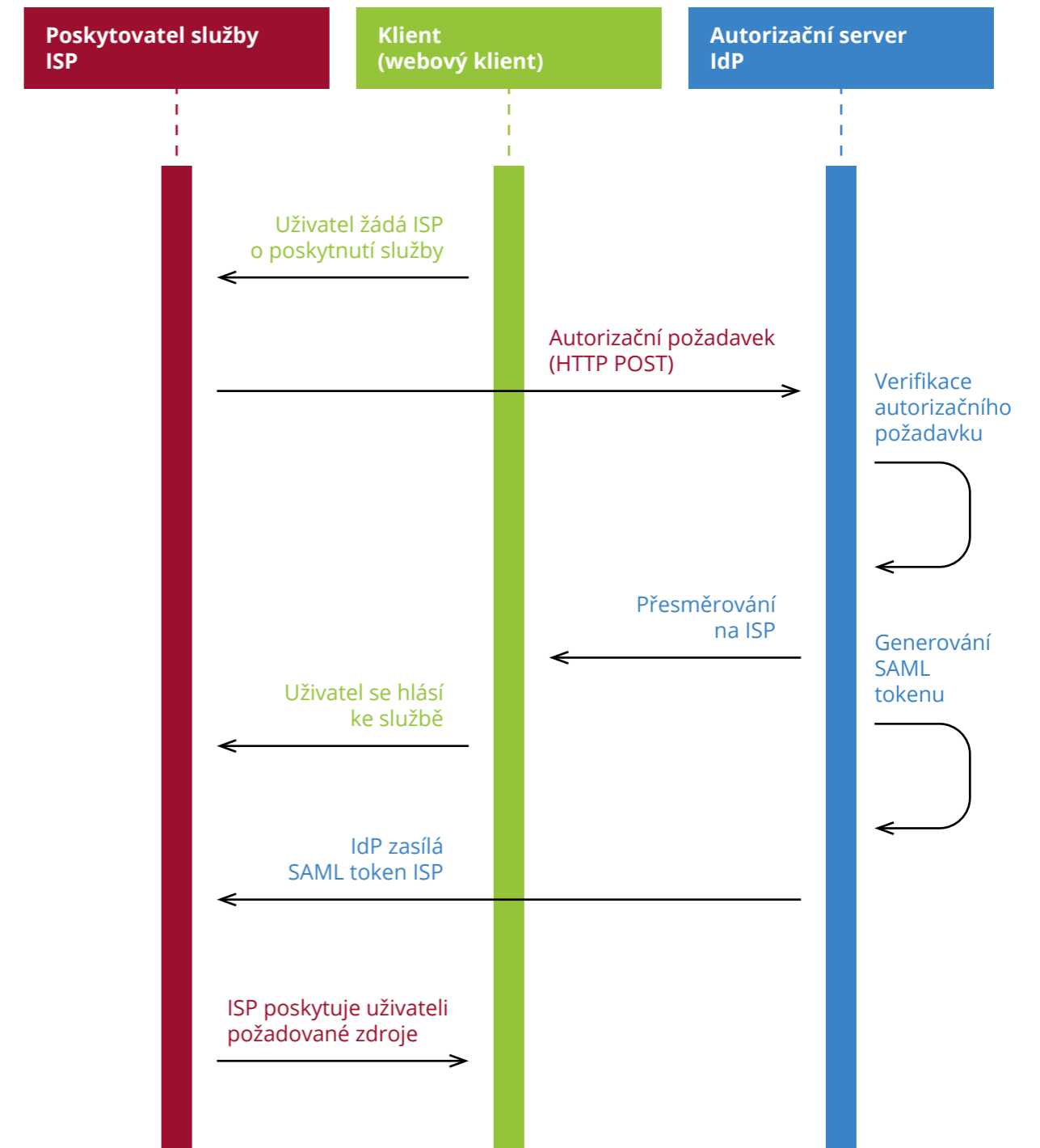


Class diagram



UML Class diagram

Sekvenční diagram

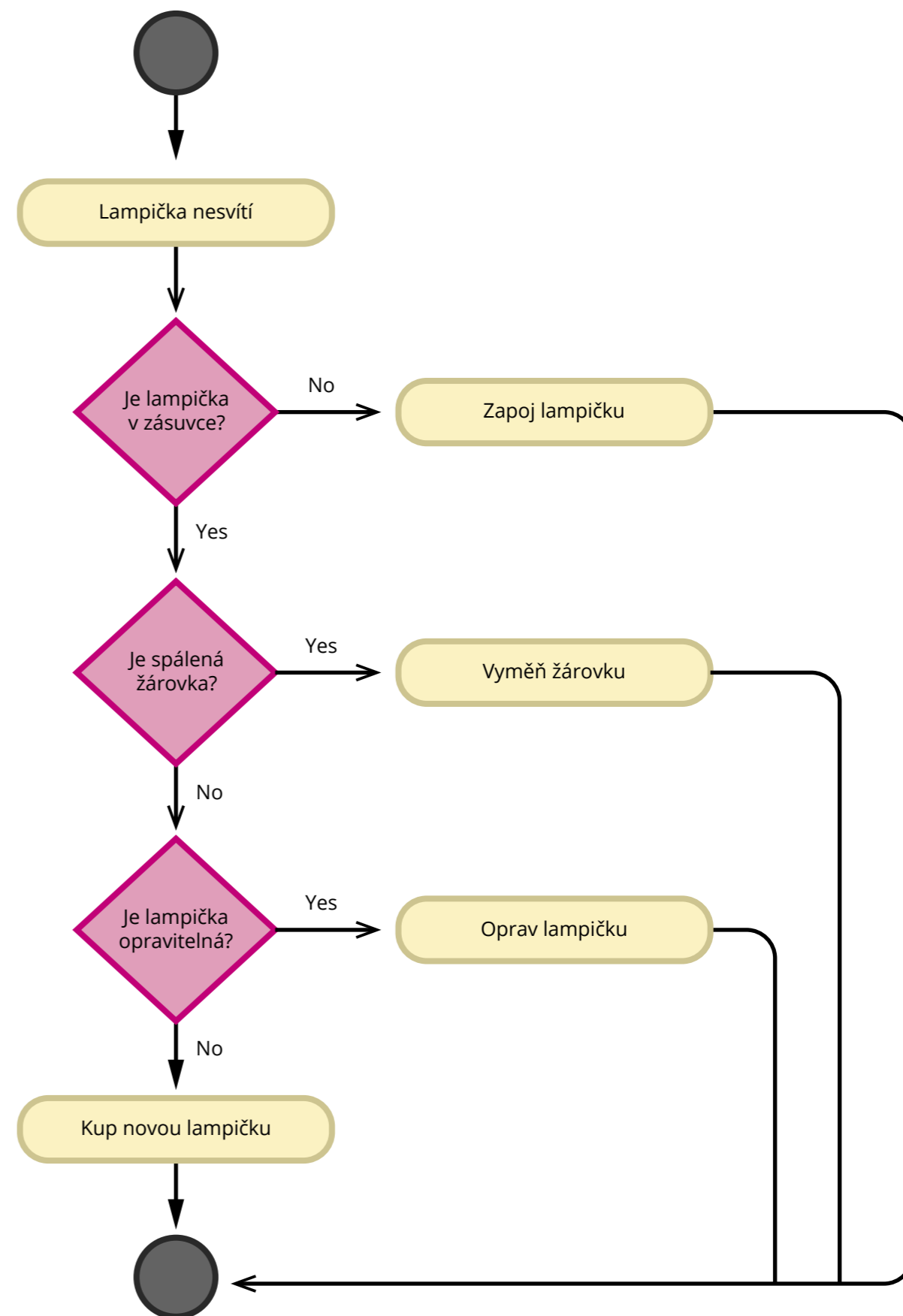
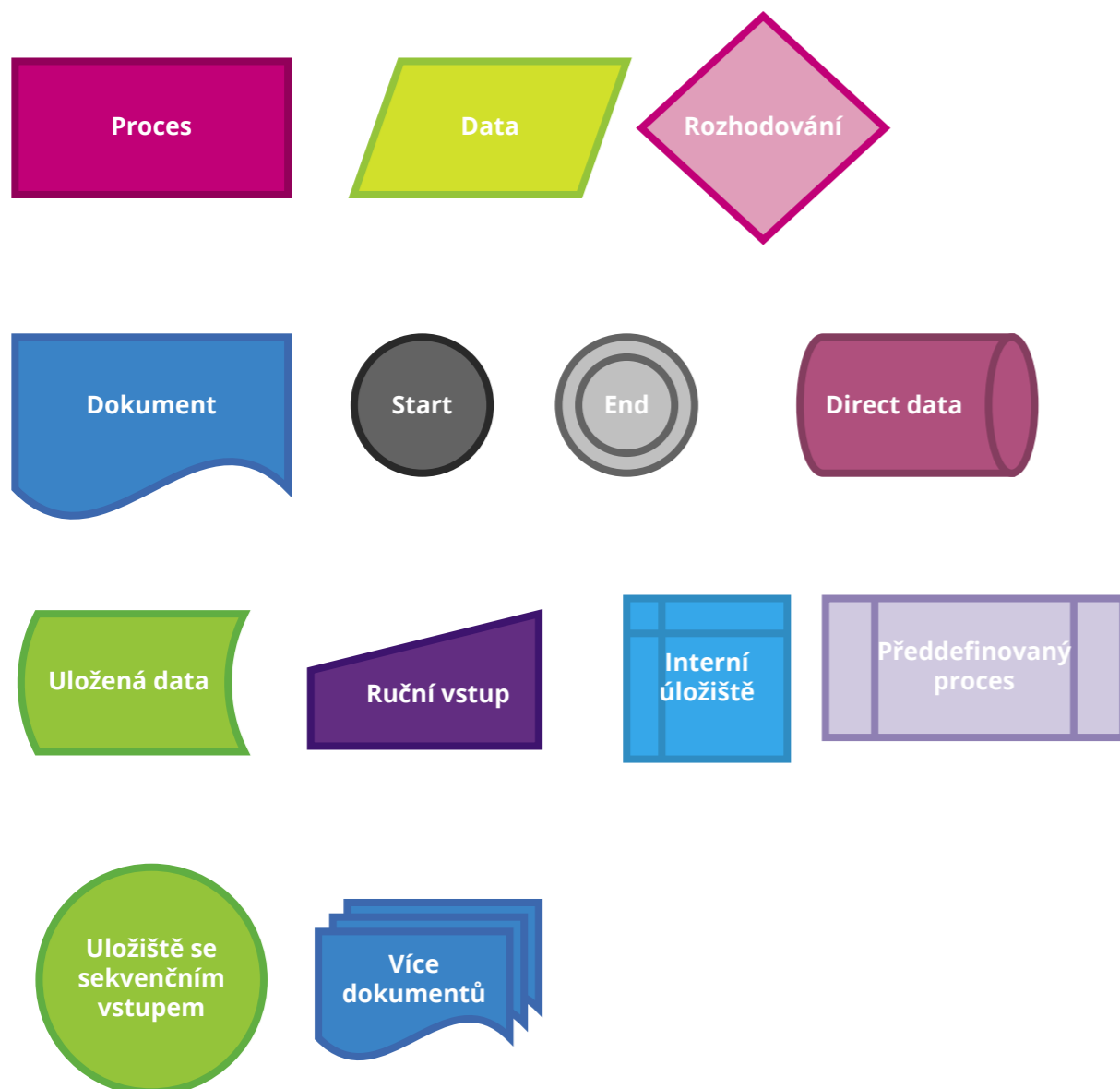


UML Sekvenční diagram - SAML protokol

Flowchart diagram

Protože flowchart diagram (neboli vývojový diagram) se přímo týká programování, budeme u něj o něco podrobnější. Nejužívanější symboly vývojového diagramu jsou na následujícím obrázku.

Více zde



Příklad vývojového diagramu je na následujícím obrázku. Zkusíme si však i některé vlastní

Algoritmus

Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy. Příkladem je kuchařský recept.

Vlastnosti algoritmů

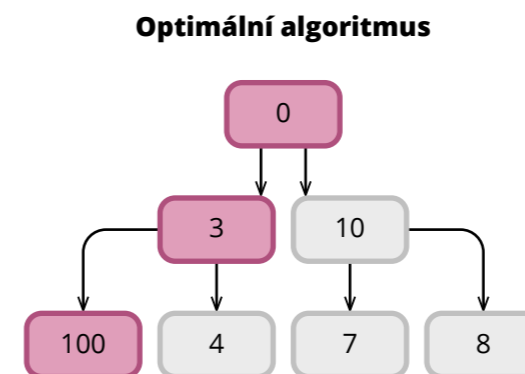
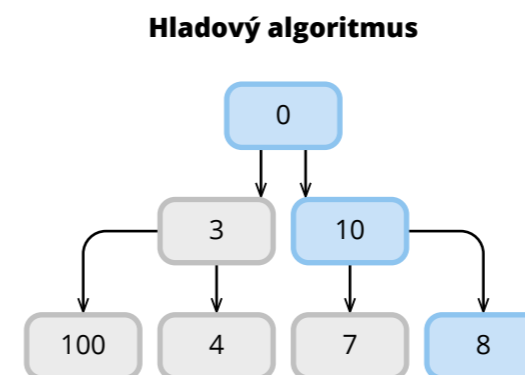
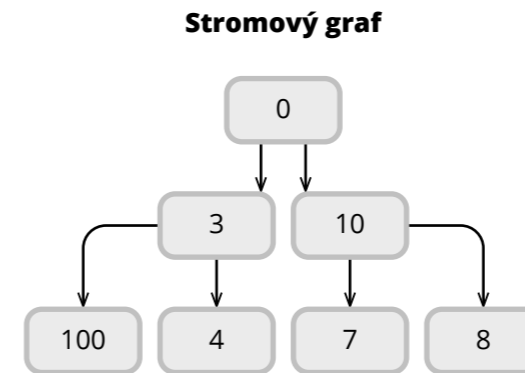
- **Elementárnost** – Algoritmus se skládá z konečného počtu jednoduchých (elementárních) kroků.
- **Konečnost** – Každý algoritmus musí skončit v konečném počtu kroků.
- **Obecnost** – Algoritmus neřeší jeden konkrétní problém (2/3), ale obecnou třídu obdobných problémů (a/b).
- **Determinovanost** – Algoritmus je determinovaný, pokud za stejných podmínek (pro stejné vstupy) poskytuje stejný výstup. Tohle je problematické - existují např. pravděpodobnostní algoritmy, kde do hry vstupuje náhoda.
- **Determinismus** – Každý krok algoritmu musí být jednoznačně a přesně definován; v každé situaci musí být naprosto zřejmé, co a jak se má provést, jak má provádění algoritmu pokračovat. Vyjádření výpočetní metody v programovacím jazyce se nazývá program.
- **Výstup** – Algoritmus vede od zpracování hodnot k výstupu.

Metody návrhu algoritmů

Algoritmů je samozřejmě mnoho. V učebnicích se dočteme například o těchto:

- Třídění
- Hladový algoritmus
- Halda
- Grafy
- Dijkstrův algoritmus
- Minimální kostra
- Rozděl a panuj
- Dynamické programování
- Vyhledávací stromy
- Hešování
- Řetězce a vyhledávání v textu
- Rovinné grafy
- Eulerovské tahy
- Toky v sítích
- Intervalové stromy

Pro praktickou ukázkou si vybereme zjednodušený hladový algoritmus



Hladový algoritmus

Hladový algoritmus (anglicky greedy search) Takovými algoritmy rozumíme ty, které hledají řešení celé úlohy po jednotlivých krocích a splňují následující dvě podmínky:

- V každém kroku zvolí lokálně nejlepší řešení.
- Provedené rozhodnutí již nikdy neodvolává (tedy „nebacktrackuje“).

Lokálně nejlepší řešení je takové, které v aktuálním kroku vybere tu možnost, která nám na tomto místě nejvíce pomůže (bez jakéhokoliv ohledu na globální stav). Může to být třeba nejvyšší hodnota, nebo nejkratší cesta v grafu.

Pokud ale od hladového algoritmu chceme, aby nám našel **globálně nejlepší řešení**, musí naše úloha splnit předpoklad, že si výběrem lokálně nejlepšího řešení nezhoršíme to globální. Tento předpoklad se nedá formulovat obecně a je nutné se nad ním zamyslet zvlášť u každé úlohy.

Hladový algoritmus si představíme v Praktické části.

Základy Pythonu

Python je pravděpodobně nejsnazší současný programovací jazyk, s nímž se dobře pracuje, má obrovské množství knihoven téměř na cokoli a velkou základnu, která vždy ráda pomůže. Kód v Pythonu se dobře píše i čte, proto jsem ho pro názorné ukázky algoritmizace a programování zvolila i já.

Python je tzv. "cross-platform", což znamená, že běží na různých operačních systémech. Není potřeba nic kompilovat, což si za daň bere, že je o poznání pomalejší než např. jazyk C.

Více zde



Budeme používat Python verze 3.3 a vyšší. Jak to zjistíme?

Prostě se optáme operačního systému, jaký Python je na něm nainstalován.

Jupyter Notebook

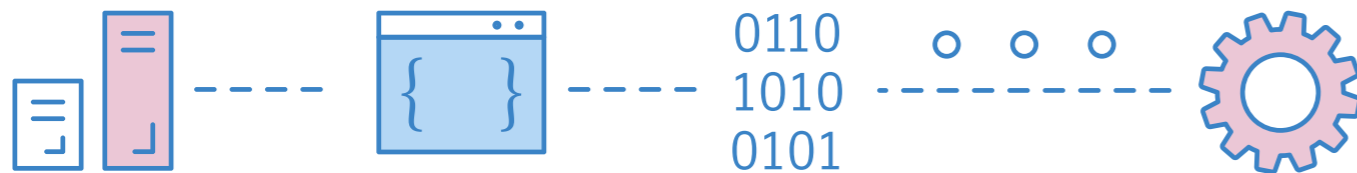
Příklady zde jsou uváděné v tzv. Jupyteru.

Poznámka: Pokud bude učitel ve výuce používat **Jupyterhub** nebo **Jupyter Notebook**, měl by jej v kurzu představit. V tomto materiálu však, vzhledem k rozsahu, uveden není.

```
!python -V
Python 3.9.1
```

Pokud nemáte Python nainstalován, zde je postup pro Linux Ubuntu.

```
sudo apt-get install python3
sudo apt-get install idle3
sudo apt-get install python3-pip
```



Jednoduché datové typy a vestavěné funkce v Pythonu

Vestavěné funkce

Python obsahuje řadu vestavěných funkcí, které nám mohou usnadnit život. Vidíte je v tabulce abecedně seřazené. **Podrobnější popis naleznete v QR kódu.**



abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	import()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

Jaké funkce a metody můžeme od třídy chtít, se dozvíme pomocí klíčového slova `dir()`. Zapamatujte si ho.

```
dir(„Marta“)

['capitalize',
 'casefold',
 'center',
 'count',
 'encode',
 'endswith',
 'expandtabs',
 'find',
 'format',
 'format_map',
 'index',
 'isalnum',
 'isalpha',
 'isascii',
 'isdecimal',
 'isdigit',
 'isidentifier',
 'islower',
 'isnumeric',
 'isprintable',
 'isspace',
 'istitle',
 'isupper',
 'join',
 'ljust',
 'lower',
 'lstrip',
 'maketrans',
 'partition',
 'removeprefix',
 'removesuffix',
 'replace',
 'rfind',
 'rindex',
 'rjust',
 'rpartition',
 'rsplit',
 'rstrip',
 'split',
 'splitlines',
 'startswith',
 'strip',
 'swapcase',
 'title',
 'translate',
 'upper',
 'zfill']
```

```
seasons = ['Spring', 'Summer', 'Fall', 'Winter']
dict(enumerate(seasons))
{0: 'Spring', 1: 'Summer', 2: 'Fall', 3: 'Winter'}compass.get_field_strength()
- intenzita magnetického pole
compass.heading() - azimut ve stupních, nutná inicializace pomocí compass.calibrate()
```

Základní datové typy

Každý programovací jazyk umí prezentovat položky dat. Python poskytuje více předdefinovaných datových typů, ale prozatím se budeme zabývat pouze několika z nich. Python představuje celá čísla (kladná a záporná celá čísla) pomocí typu `int` a řetězce (sekvence znaků Unicode) pomocí typu `str`. Zde je několik příkladů celých čísel a řetězců:

```
type(-973)
int
type(210624583337114377340864637790190801098222508621955072)
int
```

```
type('Tomáš Garrigue Masaryk')
str
type('αβγ ÷© + ■ ¶☉☄$')
str
x='αβγ ÷© + ■ ¶☉☄$'
len(x)
15
```

Příklad použití klíčového slova `dir` nám řekne, co můžeme od datového typu nebo funkce požadovat.

```
dir(x)
[...]
'capitalize',
'casefold',
'center',
'count',
'encode',
'endswith',
'expandtabs',
'find',
... ]
x.count(' ')
4
```

V textu jsou 4 mezery

Základní datové typy v Pythonu

Typ	Popis	Příklady hodnot
int	celá čísla	1, 42, -5, 200
float	reálná čísla (přesněji čísla v plovoucí desetinné čárce, přičemž Python používá desetinnou tečku, nikoliv čárku)	2.5, 3.25, -12.37832
bool	pravdivostní hodnoty	True, False
str	řetězce	"prase", "pes"

Základní datové typy Pythonu najdeme **v QR kódu**.



Složené datové typy

V Pythonu existuje pět základních složených datových struktur:

- bytearray
- bytes
- list
- str
- tuple

List

V Pythonu je list hlavní datovou strukturou používanou k ukládání sekvence prvků. Sekvence datových prvků uložených v seznamu nemusí být stejného typu.

Chcete-li vytvořit list, musí být datové prvky uzavřeny v [] a musí být odděleny čárkou. Následující kód například vytvoří dohromady čtyři datové prvky, které jsou různých typů.

```
muj_list = ["Rumcajs", 33, "Cipisek", True]
muj_list
['Rumcajs', 33, 'Cipisek', True]
type(muj_list)
list
```

List může obsahovat jiné listy i další jednoduché nebo složené datové typy.

```
muj_list = ["Rumcajs", 33, "Cipisek", True, [1, 2, 'Žlutoučký kůň úpěl ďábelské ódy']]
```

Slicing

s(-15)	s(-14)	s(-13)	s(-12)	s(-11)	s(-10)	s(-9)	s(-8)	s(-7)	s(-6)	s(-5)	s(-4)	s(-3)	s(-2)	s(-1)
T	y	r	i	o	n		L	a	n	i	s	t	e	r
s(0)	s(1)	s(2)	s(3)	s(4)	s(5)	s(6)	s(7)	s(8)	s(9)	s(10)	s(11)	s(12)	s(13)	s(14)

```
muj_list=list('Tyrion Lanister')

muj_list[:5] # od začátku 5 znaků
['T', 'y', 'r', 'i', 'o']

muj_list[:-1] # od konce všechny znaky
['r', 'e', 't', 's', 'i', 'n', 'a', 'L', ' ', 'n', 'o', 'i', 'r', 'y', 'T']

muj_list[:-3] # od konce každý třetí znak
['r', 's', 'a', 'n', 'r']

barvy=['Červená', 'Zelená', 'Modrá', 'Žlutá']
barvy[:2]
['Červená', 'Zelená']

barvy[-2]
'Modrá'

barvy[:2]
['Červená', 'Modrá']

barvy[:-2]
['Žlutá', 'Zelená']
```

```
s="I have a cat "
```

```
for i in s:
    print(s)
    s=s.lstrip(i)
```

```
I have a cat
```

```

have a cat
have a cat
ave a cat
ve a cat
e a cat
 a cat
a cat
 cat
cat
at
t

```

Nesting

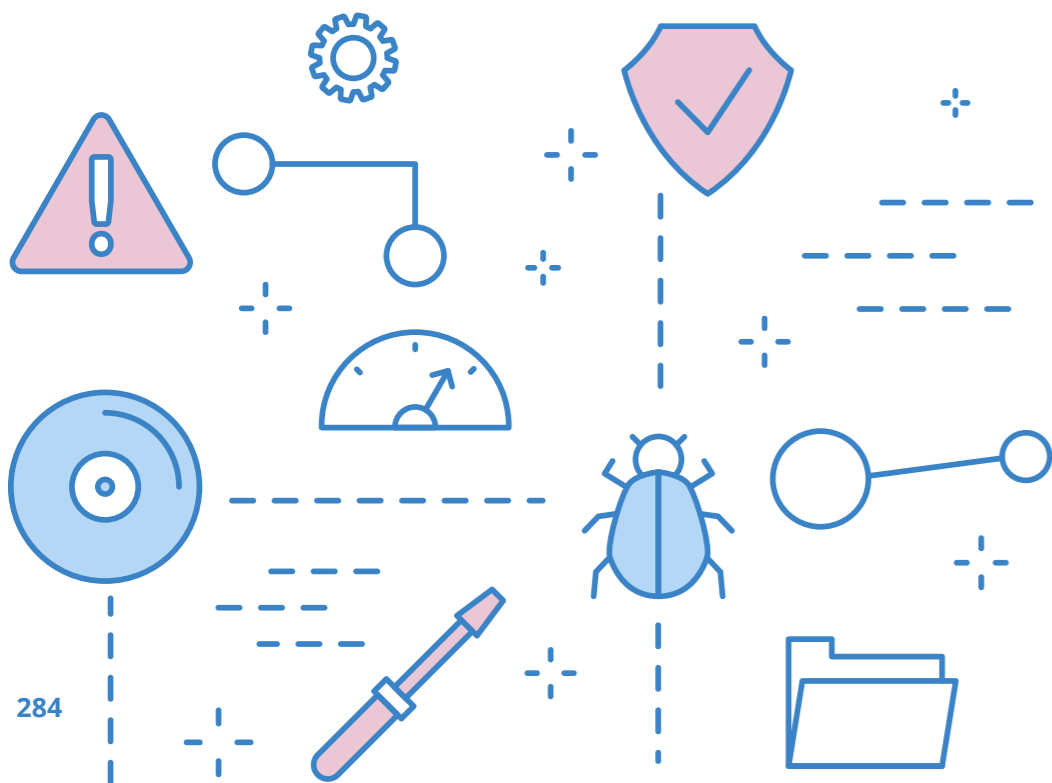
List podporuje vnořování (nesting), jednotlivé datové typy je možné uvnitř listu vnořovat.

```

a = [1,2,[100,200,300],['Marta',['Rudolf']],6]
max(a[2])
300

a[3][1][0]
'Rudolf'

```



Iterace

```

barvy
['Červená', 'Zelená', 'Modrá', 'Žlutá']

```

```

for barva in barvy:
    print(f'{barva} je v našem seznamu')

```

Červená je v našem seznamu

Zelená je v našem seznamu

Modrá je v našem seznamu

Žlutá je v našem seznamu

Lambda

Lambda je zvláštní typ funkce v Pythonu. Často se používá ve výpočtech ve složených datových typech. V příkladech si ukážeme časté použití lambda funkce.

Filtrování dat

```

list(filter(lambda x: x > 100, [-5, 200, 300, -10, 10, 1000]))
[200, 300, 1000]

```

```

list(filter(lambda x: len(x)>5, barvy))
['Červená', 'Zelená']

```

Transformace dat

Pro transformaci dat využijeme ještě další funkci `map()`.

```

list(map(lambda x: x ** 2, [11, 22, 33, 44,55]))

```

```

[121, 484, 1089, 1936, 3025]

```

```

list(map(lambda x: f'{x} je delší než 5' if len(x)>5 else f'{x} je menší než 5',
barvy))

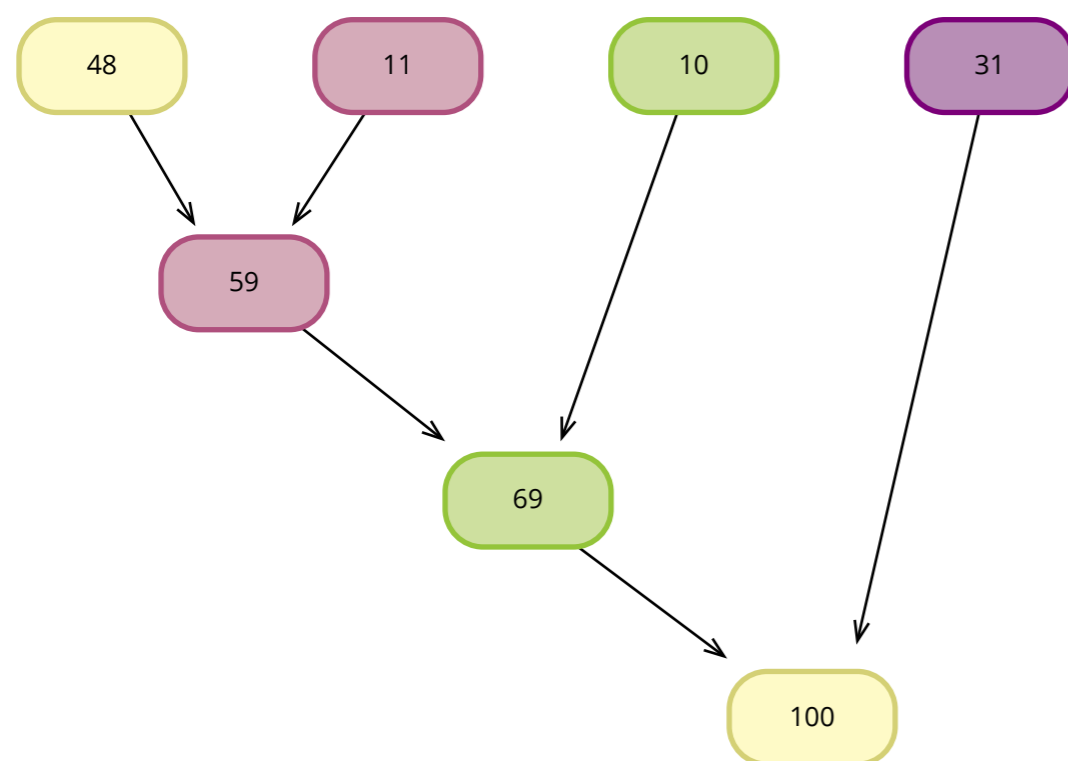
```

```

['Červená je delší než 5',
'Zelená je delší než 5',
'Modrá je menší než 5',
'Žlutá je menší než 5']

```


Reduce - Agregace dat



Lambda - redukce

```

from functools import reduce #pro použití funkce reduce se musí importovat knihovna
functools nebo její část
čísla=[48,11,10,31]
reduce(lambda x, y:x+y, čísla)
100
  
```

List comprehension

Netýká se pouze datové struktury list ale i dalších. Tato konstrukce bývá v Pythonu hojně využívána.

```

[i for i in barvy if len(i)>5]
['Červená', 'Zelená']
  
```

```

import math
čísla=[2,6,8,10,7,5]
[math.sqrt(x) for x in čísla]
[1.4142135623730951,
2.449489742783178,
2.8284271247461903,
3.1622776601683795,
2.6457513110645907,
2.23606797749979]
  
```

List metody

Syntax	Description
L.append(x)	Přidává položku na konec listu L
L.count(x)	Vrátí počet výskytů položky x v listu L
L.extend(m)	L += m přidává všechny položky na konec listu L
L.index(x, start, end)	Vrací index pozice hledané položky, pokud je jich víc, tak první zleva; pokud položku nenajde odpoví ValueError exception
L.insert(i, x)	Vkládá položku x do listu L na pozici int i
L.pop()	Vrací hodnotu položky a zároveň odstraňuje výskyt položky v listu L. Pokud je položek více, odstraní první zprava
L.pop(i)	Vrací hodnotu položky a zároveň odstraňuje výskyt položky v listu L na pozici int i v listu L
L.remove(x)	Odstraní výskyt položky x v listu L, pokud položku x nenajde odpoví ValueError exception
L.reverse()	Obrátí pořadí položek v listu L
L.sort(...)	Setřídí list L

```
barvy.append(['Oranžová', "Duhová"])
barvy.extend(['Fialová', "Purpurová"])
```

```
barvy
```

```
['Červená',
 'Zelená',
 'Modrá',
 'Žlutá',
 ['Oranžová', 'Duhová'],
 'Fialová',
 'Purpurová']
```

Range

```
for i in range(2,25,3): # rozsah od 2(včetně) do 25(ne včetně) krok jsou 3
    print(i, end=",")
2,5,8,11,14,17,20,23,
```

Tuples

Datová struktura, která je immutable - tedy pouze pro čtení. Oproti listu má velmi málo možností - count a index. Tuply jde sčítat.

```
t= (1,2,3,'Marta')
```

```
t += ("Vohnoutová",)
```

```
type(t)
```

```
tuple
```

```
dir(t)
```

```
[...
 'count',
 'index']
```

Dictionary (slovník)

Slovník je datová struktura, která je tvořena vždy klíčem (key) a hodnotou (value). Klíč musí být v dictionary jedinečný. V Pythonu se dictionary využívá velmi často. Uzavírá se mezi složené závorky {}

```
barvy_semaforu = {
    "Stop": "Červená",
    "Připrav se": "Oranžová",
    "Jed": "Zelená"
}
```

```
barvy_semaforu.get("Stop")
'Červená'
```

```
barvy_semaforu['Stop']
'Červená'
```

```
barvy_semaforu['Barva co není v semaforu'] = 'Růžová' # přidám do slovníku další položku
```

```
barvy_semaforu
{'Stop': 'Červená',
 'Připrav se': 'Oranžová',
 'Jed': 'Zelená',
 'Barva co není v semaforu': 'Růžová'}
```

```
barvy_semaforu.keys()
```

```
dict_keys(['Stop', 'Připrav se', 'Jed', 'Barva co není v semaforu'])
```

```
barvy_semaforu.values()
```

```
dict_values(['Červená', 'Oranžová', 'Zelená', 'Růžová'])
```

```
barvy_semaforu.items()
```

```
dict_items([('Stop', 'Červená'), ('Připrav se', 'Oranžová'), ('Jed', 'Zelená'),
 ('Barva co není v semaforu', 'Růžová')])
```

Set (sada)

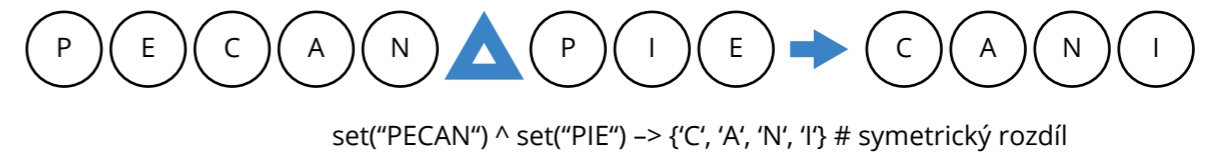
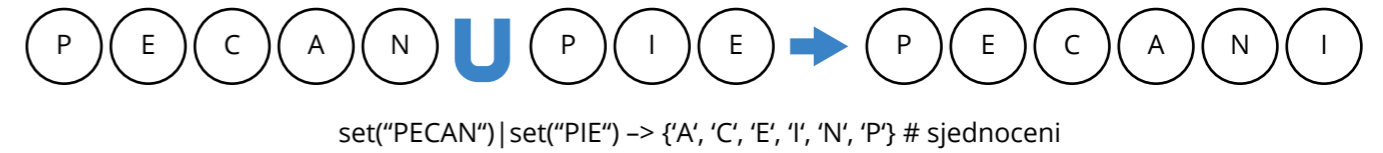
Typ set je datový typ, který podporuje některé množinové operace a je iterovatelný. Stejně jako dictionary se set uvozuje složenými závorkami {}. Set neřeší pořadí položek.

```
zelené = {"lupení", "tráva", "volno na semaforu"}
červené = {"rudá růže", "meloun", "stop na semaforu"}
žluté = {"lupení", "tráva", "pampelišky", "sluníčko"}
```

```
type(zelené)
set

dir(zelené)
[...
'add',
'clear',
'copy',
'difference',
'difference_update',
'discard',
'intersection',
'intersection_update',
'isdisjoint',
'issubset',
'issuperset',
'pop',
'remove',
'symmetric_difference',
'symmetric_difference_update',
'union',
'update']
```

Set - množinové operace



```
set("PECAN") | set("PIE") # sjednocení
{'A', 'C', 'E', 'I', 'N', 'P'}
```

```
zelené | žluté
```

```
{'lupení', 'pampelišky', 'sluníčko', 'tráva', 'volno na semaforu'}
```

Aritmetické operace

1. Aritmetické operace
2. Srovnávací operace
3. Přiřazovací operace
4. Logické operace
5. Bitové operace
6. Membership operace
7. Identitní operace

Aritmetické operace

1. sčítání +
2. odčítání -
3. násobení *
4. dělení /
5. umocňování **

```
8**3 # umocňování
512
```

Srovnávací operace

```
a, b=2,6
a <= b, a!= b, a >= b, a > b
(True, True, False, False)
```

Přiřazovací operace

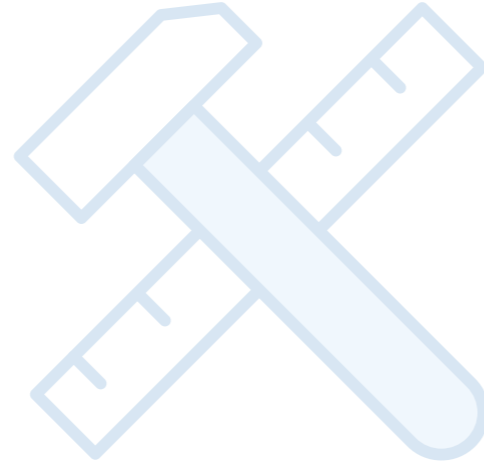
```
a=5
b=a
```

Logické operace

1. and
2. or
3. not

```
a, b=2,6
a <= b and a!= b and a >= b and a > b
```

Více zde



False

Membership operace

```
"Zelená" in barvy
True
```

Identitní operace

```
a=[12.10,7]
b=[12.10,7]
a == b #Srovnávací operace
True
a is b #Identitní operace
False
```

Flow control

Zpracovávají cykly a podmínky.

1. if-elif-else
2. while
3. for

while

```
while True:
    num = input("Vložte celé kladné číslo nebo nulu pro konec: ")
    try:
        num = int(num)
    except ValueError:
        num = -1
    if num < 0:
        print ("Prosím pouze celé kladné číslo.")
        continue
    elif num == 0:
        print ("Končím program..")
        break
    num_cube = (num ** 3)
```



```
print ("Krychle o straně {0} má objem {1}.".format (num, num_cube))
print ("Nashledanou...")
```

Vložte celé kladné číslo nebo nulu pro konec: a

Prosím pouze celé kladné číslo.

Vložte celé kladné číslo nebo nulu pro konec: -5

Prosím pouze celé kladné číslo.

Vložte celé kladné číslo nebo nulu pro konec: 88

Krychle o straně 88 má objem 681472.

Vložte celé kladné číslo nebo nulu pro konec: 0

Končím program..

Konec programu.

break continue pass

```
print("break")
for i in range(0,10):
    if i == 6:
        print('here',end=',')
        break
    print(i, end=',')
print('\n')
print("continue")
for i in range(0,10):
    if i == 6:
        print("here",end=',')
        continue
    print(i, end=',')
print('\n')
```

```
print("pass")
for i in range(0,10):
    if i == 6:
        print("here",end=',')
        pass
    print(i, end=',')
print('\n')
break
0,1,2,3,4,5,here,
```

```
continue
0,1,2,3,4,5,here,7,8,9,
pass
0,1,2,3,4,5,here,6,7,8,9,
```

If-elif-else

```
t=(100000,999,9999)
for lines in t:
    if lines < 1000:
        print(lines," small")
    elif lines < 10000:
        print(lines," medium")
    else:
        print(lines," large")
100000 large
999 small
9999 medium
```

Vytváření a volání funkcí

def jméno_funkce(argumenty):

tělo funkce

```
def vyber_sudá_čísla(seznam_čísel):
    return [i for i in seznam_čísel if i%2 == 0]

print(vyber_sudá_čísla([4,7,88,21,15,4,72,62,112,332,331]))
[4, 88, 4, 72, 62, 112, 332]

def vyber_dělitelná_čísla(seznam_čísel, dělitel):
    return [i for i in seznam_čísel if i%dělitel == 0]

print(vyber_dělitelná_čísla([4,9,88,21,15,4,72,62,112,332,333,331,12],3))
[9, 21, 15, 72, 333, 12]
```

Praktická část

Praktická část předpokládá u každého příkladu postupovat v krocích, co jsme se naučili:

1. Formulace problému
2. Analýza úlohy
3. Vytvoření algoritmu úlohy - vývojový diagram
4. Sestavení programu a jeho odladění

Jako příklad si vezměme jednoduchou úlohu, často používanou ve hře geocaching – ciferaci.

Ciferace - 1.úloha prováděná učitelem

Ciferace - formulace problému

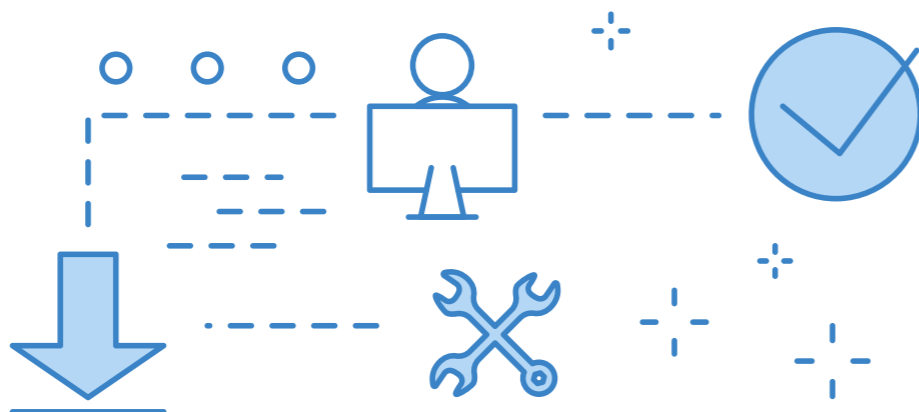
Při ciferaci sečteme všechny číslice z daného čísla. Pokud výsledný součet není jednociferný, součet opakujeme do té doby, až dostaneme jednociferné číslo. Toto jednociferné číslo je výsledkem ciferace.

Ciferace - analýza úlohy

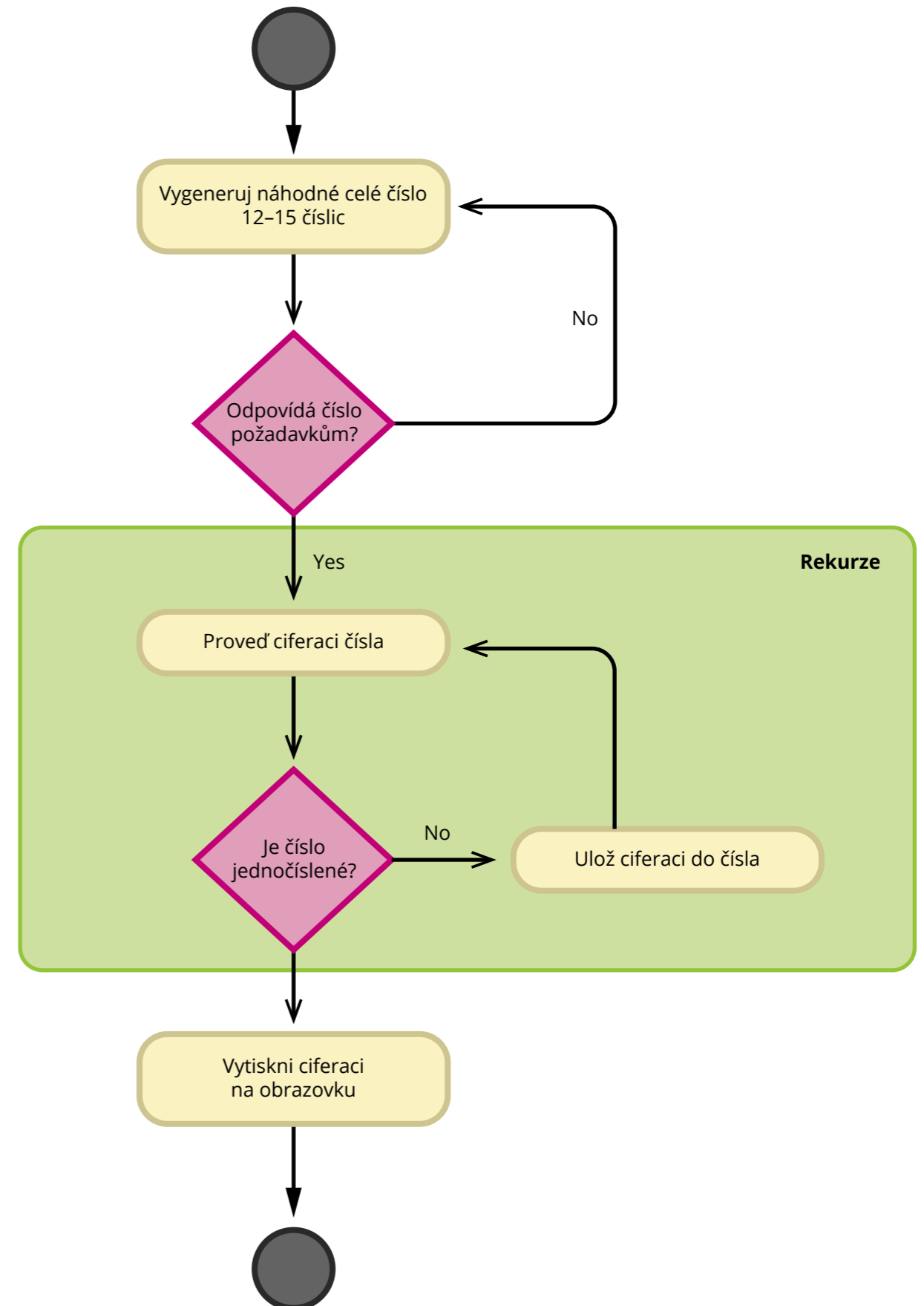
Zadání čísla – náhodné číslo o délce 15 až 20 číslic vybere počítač.

Zkontrolujeme, zda číslo odpovídá našim požadavkům:

- Číslo je celé kladné číslo
- Číslo obsahuje 15 až 20 číslic



Ciferace - vytvoření algoritmu úlohy - vývojový diagram



Cyklus

Rozdělte číslo na x číslic

Sečtěte x číslic

Zbyla vám jediná číslice?

– Pokud ano, vytiskněte ji na obrazovku a program končí.

– Pokud ne, uložte součet do čísla a vraťte se na Cyklus.

Ciferace - sestavení programu

#ciferace

#Autor: Marta Vohnoutová

```
import random as rd

def ciferace(číslo):
    c=sum([int(i) for i in str(číslo)])
    if len(str(c))==1:
        return c
    else:
        return ciferace(c)

while True:
    číslo = rd.randint(1e12,1e16)
    if isinstance(číslo, int) and číslo in range(int(1e12), int(1e16)):
        break

číslo = 248877668265883584
print(f'Pro číslo {číslo} je ciferace {ciferace(číslo)}')
Pro číslo 248877668265883584 je ciferace 6
```

Najděte slova podle zadaného vzoru - 2.úloha prováděná učitelem

Slova podle vzoru - formulace problému

Vytvořte Python program, který:

1. Stáhněte si slovník dictionary.txt, který najdete v **QR kódu**.
2. Ve slovníku dictionary.txt najděte všechna slova odpovídající zadanému vzoru:



Zadaný vzor je 0. 1. 2. 3. 2. 4. 5. 6. 7 Slova odpovídající zadanému vzoru musí splňovat - příklad pro vzor '0. 0. 1. 2. 3' ['AARON', 'LLOYD', 'OOZED']

Slova podle vzoru - analýza úlohy

Zadání vzoru ve tvaru - list, uvnitř čísla oddělená čárkami např.[0,1, 1, 2, 3,1]

Zkontrolujeme, zda vzor odpovídá našim požadavkům:

- list
- uvnitř čísla oddělená čárkami

Stáhni a otevři soubor dictionary.txt

Cyklus

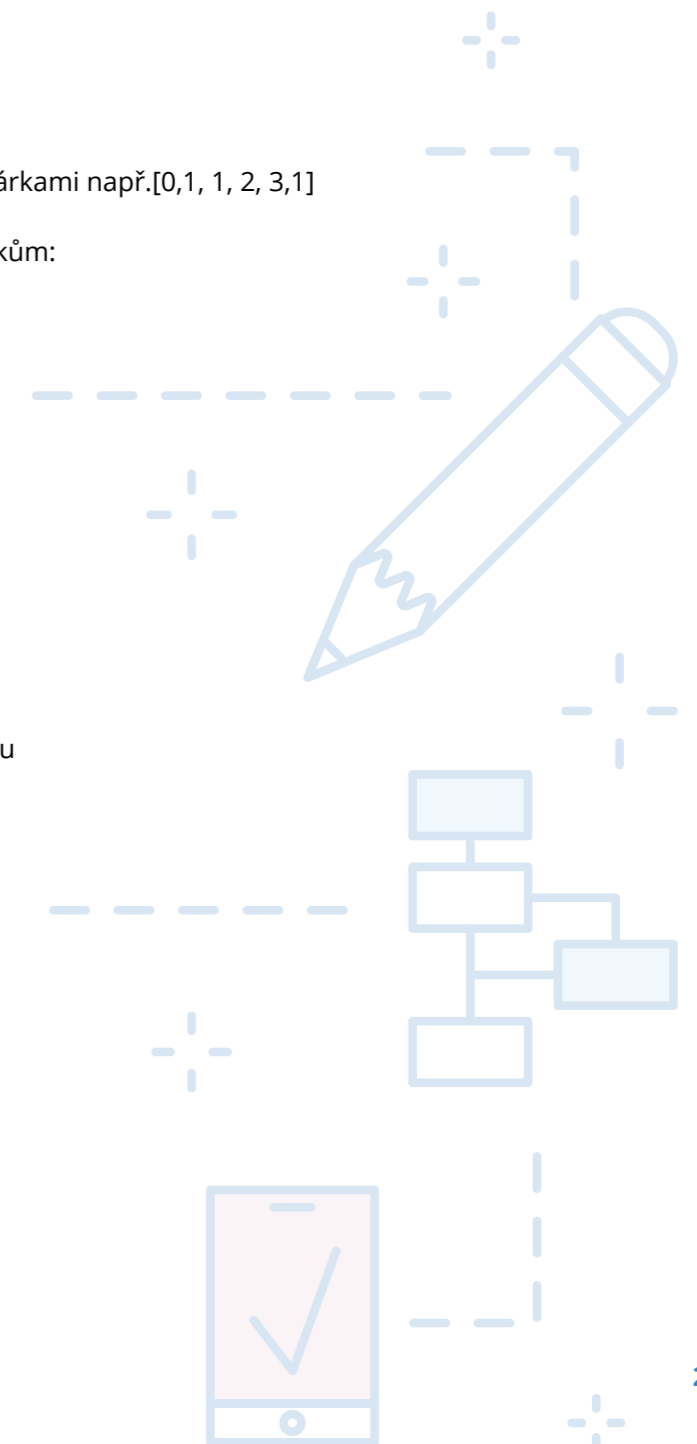
Ze souboru čti slovo po slovu

Zavolej funkci, která zjistí, zda slovo odpovídá vzoru

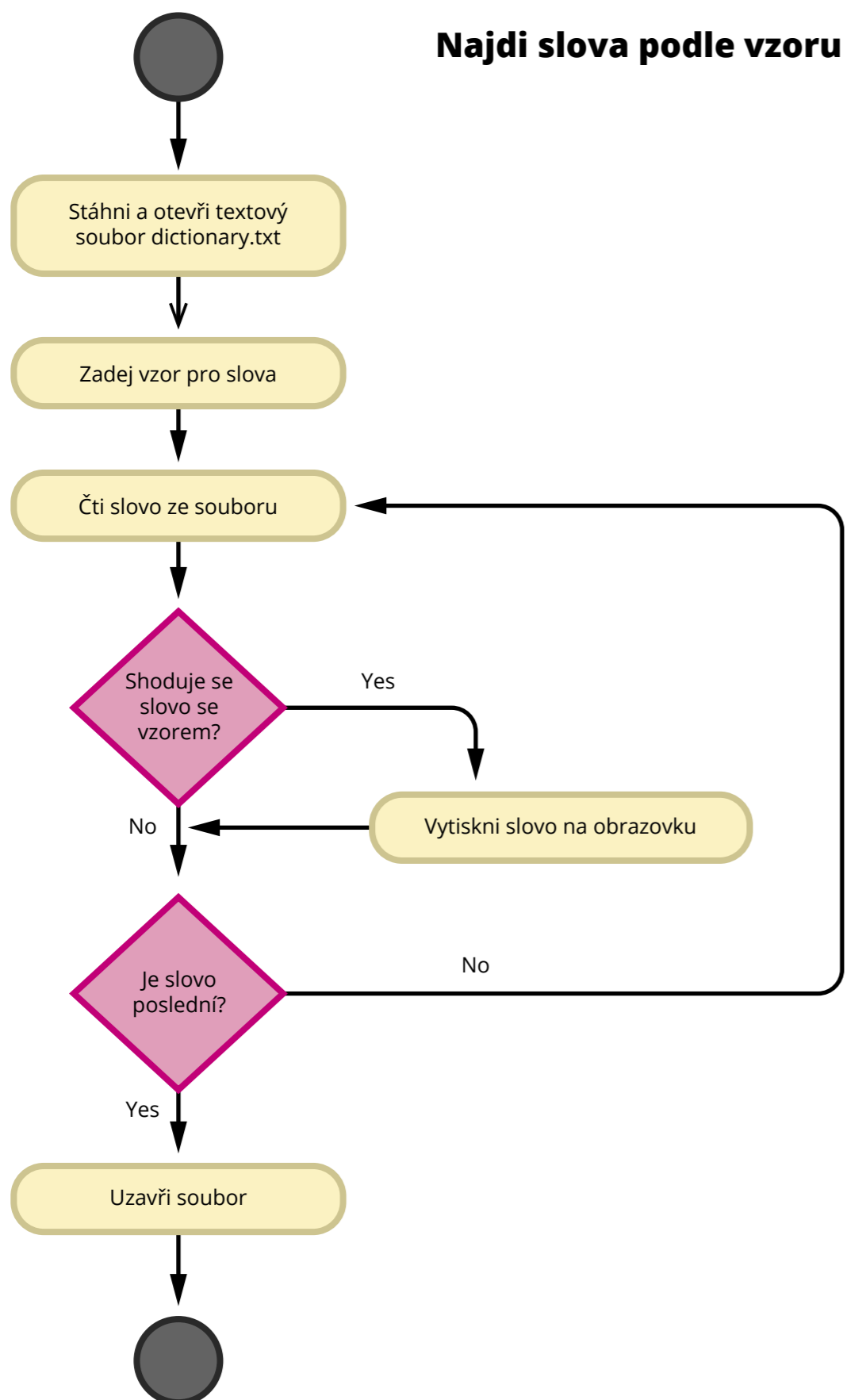
- Pokud ano, vytiskněte slovo na obrazovku

Je slovo poslední?

- Pokud ano, uzavřete soubor a program končí
- Pokud ne, přečtěte další slovo



Slova podle vzoru - vývojový diagram



Slova podle vzoru - sestavení programu

```

# Marta make patterns find words
import urllib

urllib.request.urlretrieve("https://raw.githubusercontent.com/MartaVohnoutovaBukovec/IOS-655-Python-a-Bash/master/dictionary.txt", "/home/marta/Zakladni_skoly_2021/dictionary.txt")

f = open("/home/marta/Zakladni_skoly_2021/dictionary.txt", "r")
pattern_to_find=[0,1, 1, 2, 3,1]

def make_pattern(word):

    w=""
    pattern=[]
    for i in range(len(word)):
        if w.find(word[i]) == -1:
            w += word[i]
            pattern.append(w.find(word[i]))

    return pattern == pattern_to_find

print(pattern_to_find)
for word in f:
    if make_pattern(word.rstrip('\n')):
        print(word, end=' ')

f.close()

[0, 1, 1, 2, 3, 1]
ADDEND
ASSETS
ATTEST
BEETLE
EGGING
ESSAYS
FEEBLE
ISSUES
NEEDLE
SEETHE
  
```


Úlohy pro procvičování žáky

Úlohy pro procvičování žáky nemají přiložené řešení, ale při školení učitelů je možné řešení učitelům předat. U každé úlohy je označena její obtížnost, která je samozřejmě relativní. Protože se jedná především o výuku algoritmizace a programového myšlení, tak, ačkoliv úlohy nejsou nijak složité, je vyžadováno provedení všech kroků:

1. Formulace problému
2. Analýza úlohy
3. Vytvoření algoritmu úlohy - vývojový diagram
4. Sestavení programu a jeho odladění

Úloha č.1 – snadná – Matematické hádanky

Napište Python program, který vyřeší následující matematické hádanky:

1. Najděte správná celá čísla A, B, C, D, E, F, pro která platí $ABCDEF \times 3 = BCDEFA$, kde ABCDEF je celé kladné číslo, které se vytvoří spojením jednotlivých číslic A, B, C, D, E, F. Správné možnosti vytiskněte jako *list* a v něm *tuply* takto: [(A1, B1, C1, D1, E1, F1), (A2, B2, C2, D2, E2, F2)...].
Příklad výstupu: [(1,4,2,8,5,7), (2,8,5,7,1,4)]
2. Najděte celé kladné číslice označené L, O, G, I, C, pro které bude platit $(L+O+G+I+C)3 = LOGIC$, kde LOGIC je celé kladné číslo vytvořené sdružením (concatenating) číslic L, O, G, I, C. Výsledek vytiskněte na obrazovku jako *list* obsahující *tuply* takto: [(L1, O1, G1, I1, C1), (L2, O2, G2, I2, C2)...].
3. Najděte celé kladné číslice označené C, O, W, E, D, pro které bude platit $COW \times COW = DEDCOW$, kde COW vznikne sdružením jednotlivých číslic C, O, W. Výsledek vytiskněte na obrazovku jako *list* obsahující *tuply* takto: [(C1, O1, W1, E1, D1), (C2, O2, W2, E2, D2)...].

Důležitá poznámka: Číslice mohou být v rozsahu 0...9 a v řešení se nesmí opakovat.

Nápověda: Použijte permutace k vytvoření možných kombinací čísel. Permutace vám zpřístupní import

```
from itertools import permutations as pm
```

```
# Marta Vohnoutová - ukoll - Matematické hádanky
```

```
from itertools import permutations as pm
```

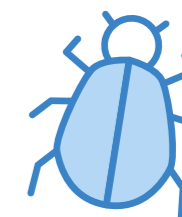
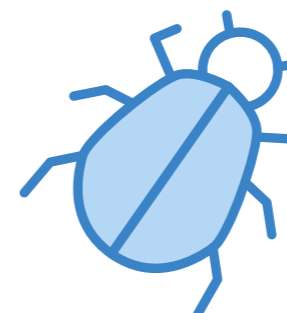
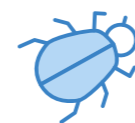
```
def ukoll_1():
    vystup=[]
    r=range(0,10)
    x=list(pm(r,6))
```

```
for i in x:
    if (i[0]*100000+i[1]*10000+i[2]*1000+i[3]*100+i[4]*10+i[5]) *3 == (i[1]*100000+i[2]*10000+i[3]*1000+i[4]*100+i[5]*10+i[0]):
        vystup.append(i)
print('Ukol 1.1',vystup)
```

```
def ukoll_2():
    vystup=[]
    r=range(0,10)
    x=list(pm(r,5))
    for i in x:
        if (i[0]+i[1]+i[2]+i[3]+i[4])**3 == (i[0]*10000+i[1]*1000+i[2]*100+i[3]*10+i[4]):
            vystup.append(i)
    print('Ukol 1.2',vystup)
```

```
def ukoll_3():
    vystup=[]
    r=range(0,10)
    x=list(pm(r,5))
    for i in x:
        if (i[0]*100+i[1]*10+i[2])*(i[0]*100+i[1]*10+i[2]) == (i[4]*100000+i[3]*10000+i[4]*10000+i[0]*100+i[1]*10+i[2]):
            vystup.append(i)
    print('Ukol 1.3', vystup)
```

```
ukoll_1()
ukoll_2()
ukoll_3()
Ukol 1.1 [(1, 4, 2, 8, 5, 7), (2, 8, 5, 7, 1, 4)]
Ukol 1.2 [(0, 4, 9, 1, 3), (0, 5, 8, 3, 2), (1, 9, 6, 8, 3)]
Ukol 1.3 [(3, 7, 6, 4, 1)]
```



Úloha č.2 – snadná – Cyklus

Pro 10 náhodných čísel v rozsahu od 19 do 999 a pro číslo 27 vytvořte cyklus, který bude provádět následující kroky:

1. Jestliže je číslo n sudé, proveďte operaci $n/2$
2. Jestliže je číslo n liché, proveďte operaci $3n + 1$
3. celý cyklus opakujte, dokud nebude číslo $n == 1$
4. Pro každé číslo počítejte počet kroků než dosáhne číslo n hodnoty 1.

Vytiskněte všechny kroky pro každé číslo n a vytiskněte číslo s největším počtem kroků.

Nápověda: Náhodné číslo v daném rozsahu dostanete takto:

```
from random import randint as rd
n=rd(19,999)
```

```
# Marta Vohnoutová - úkol2 - Cyklus
```

```
from random import randint as rd
```

```
def ukol2():
    vystup={}
    for _ in range(0,10):
        vystup[rd(19,999)]=[]

    vystup[27]=[]

    for k, v in vystup.items():
        n=k
        while True:
            vystup[k].append(n)
            if n==1:
                break
            if n%2 == 0: # sude
                n=n/2
            elif n%2!= 0: # liche
                n=3*n+1
        return vystup
```

```
vystup=ukol2()
#print(vystup)
cislo=0
steps=0
for k, v in vystup.items():
    if len(v)>steps:
        steps=len(v)
        cislo=k
print(f'Pro číslo {cislo} je maximum {steps} kroků k jedničce')
```

```
Pro číslo 27 je maximum 112 kroků k jedničce
```

Úloha č.3 – snadné – Počítání samohlásek

Napište program v Pythonu, který spočte počet samohlásek ('a', 'e', 'i', 'o', 'u','y') v každém slově.

Zjednodušení: Slova jsou oddělena vždy mezerami, písmena jsou všechna malá a bez diakritiky. Např. věta „ucíme se python kazdy den“ bude mít výstup:

Výstup: Počet samohlásek je 1 1 1 3 2 2

```
# Marta Vohnoutová - úkol2 - Cyklus
```

```
def ukol3(veta):
    vowels = ('a', 'e', 'i', 'o', 'u','y')
    slova = veta.split()
    for i in slova:
        print(i, sum([i.count(j) for j in vowels]), end=',')
```

```
ukol3('ucime se python kazdy den')
ucime 3,se 1,python 2,kazdy 2,den 1,
```

Úloha č.4 – snadné – Veliká písmena

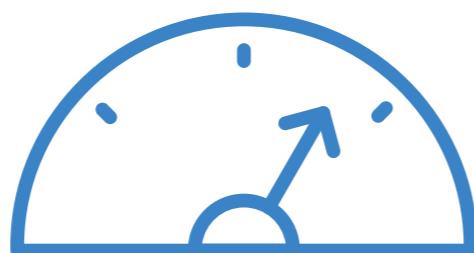
Z obrazovky zadejte nebo náhodně vygenerujte celé kladné číslo o několika číslicích. Na obrazovku pak vytiskněte číslo složené z velikých číslic. Abyste se netrápili s zadáním velikých číslic, tady je. Kdo si troufá, může hvězdičky zaměnit za konkrétní číslici.

Nápověda: Definování velkých číslic

```
Zero = [" *** ",
        " *  * ",
        "*   *",
        "*   *",
        "*   *",
        " *  * ",
        " *** "]

One = [" * ", "*** ", " * ", " * ", " * ", " * ", "****"]
Two = [" *** ", "*  *", "** *", " * ", " * ", " * ", "*****"]
Three = [" *** ", "*  *", " * ", " ** ", " * ", " *  *", " *** "]
Four = ["  * ", " ** ", " * * ", " ** ", "*****", "  * ", " * "]
Five = ["*****", " * ", " *  ", " *** ", "  * ", " *  *", " *** "]
Six = [" *** ", " *  ", " *  ", "*****", " *  *", " *  *", " *** "]
Seven = ["*****", "  *", "  * ", " * ", " *  ", " *  ", " *  "]
Eight = [" *** ", "*  *", " *  *", " *** ", " *  *", " *  *", " *** "]
Nine = [" *****", " *  *", " *  *", " *****", "  *", "  *", "  *"]

Digits = [Zero, One, Two, Three, Four, Five, Six, Seven, Eight, Nine]
```



Grafy

Nahlédneme malinko do jednoho z nástrojů na vytváření grafů – jmenuje se **networkx**. Pokud ho chcete používat, musíte jej nejprve nainstalovat.

Více zde



Instalace se provádí např. pomocí instalačního programu pip nebo conda.

```
pip install networkx
```

Dále se musí do prostředí Pythonu naimportovat. Pro náš příklad použijeme následující importy. Pokud v kurzu použijete připravené virtuální stroje, bude prostředí již připraveno, jinak si je musíte doinstalovat sami.

Importy

```
import networkx as nx
import matplotlib.pyplot as plt
from networkx.drawing.nx_pydot import graphviz_layout
import operator
```

```
dir(nx.Graph()) # metody networkx
[...
 'add_edge',
 'add_edges_from',
 'add_node',
 'add_nodes_from',
 'add_weighted_edges_from',
 'adj',
 'adjacency',
 'adjlist_inner_dict_factory',
 'adjlist_outer_dict_factory',
 'clear',
 'clear_edges',
 'copy',
 'degree',
 'edge_attr_dict_factory',
 'edge_subgraph',
 'edges',
 'get_edge_data',
 'graph',
```

```
'graph_attr_dict_factory',
'has_edge',
'has_node',
'is_directed',
'is_multigraph',
'name',
'nbunch_iter',
'neighbors',
'node_attr_dict_factory',
'node_dict_factory',
'nodes',
'number_of_edges',
'number_of_nodes',
'order',
'remove_edge',
'remove_edges_from',
'remove_node',
'remove_nodes_from',
'size',
'subgraph',
'to_directed',
'to_directed_class',
'to_undirected',
'to_undirected_class',
'update']
```

```
muj_graf = nx.DiGraph() # vytváříme naši instanci grafu
```

```
muj_graf.add_nodes_from(['a1', 'b1a1', 'b2a1', 'c1b1a1', 'c2b1a1', 'c3b2a1', 'c4b2a1']) #
přidáme nody
```

```
print(f"Tento můj graf má nyní {muj_graf.number_of_nodes()} nódů.")
```

```
Tento můj graf má nyní 7 nódů.
```

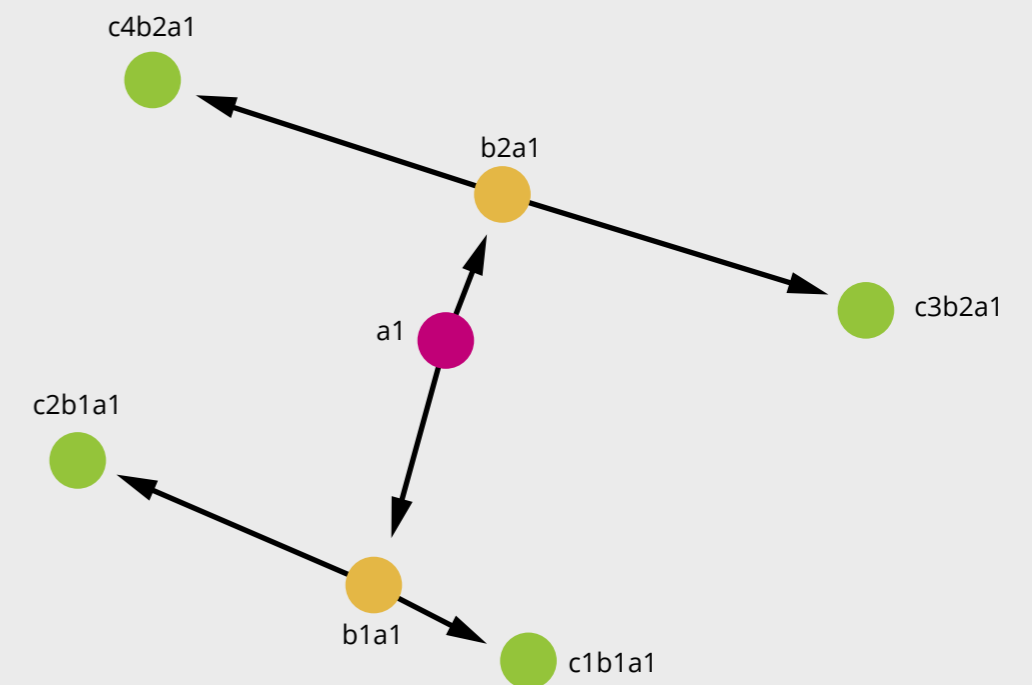
```
# přidáme strany grafu a váhy
```

```
muj_graf.add_edge("a1", "b1a1", weight=3.0)
muj_graf.add_edge("a1", "b2a1", weight=10.0)
muj_graf.add_edge("b1a1", "c1b1a1", weight=100.0)
```

```
muj_graf.add_edge("b1a1", "c2b1a1", weight=4.0)
muj_graf.add_edge("b2a1", "c3b2a1", weight=7.0)
muj_graf.add_edge("b2a1", "c4b2a1", weight=8.0)
```

```
color_list = ["violet", "darkorange", "darkorange",
             "limegreen", "limegreen", "limegreen", "limegreen"]
```

```
nx.draw_networkx(muj_graf, node_color=color_list, with_labels=True) # nakreslíme si
naš graf
```



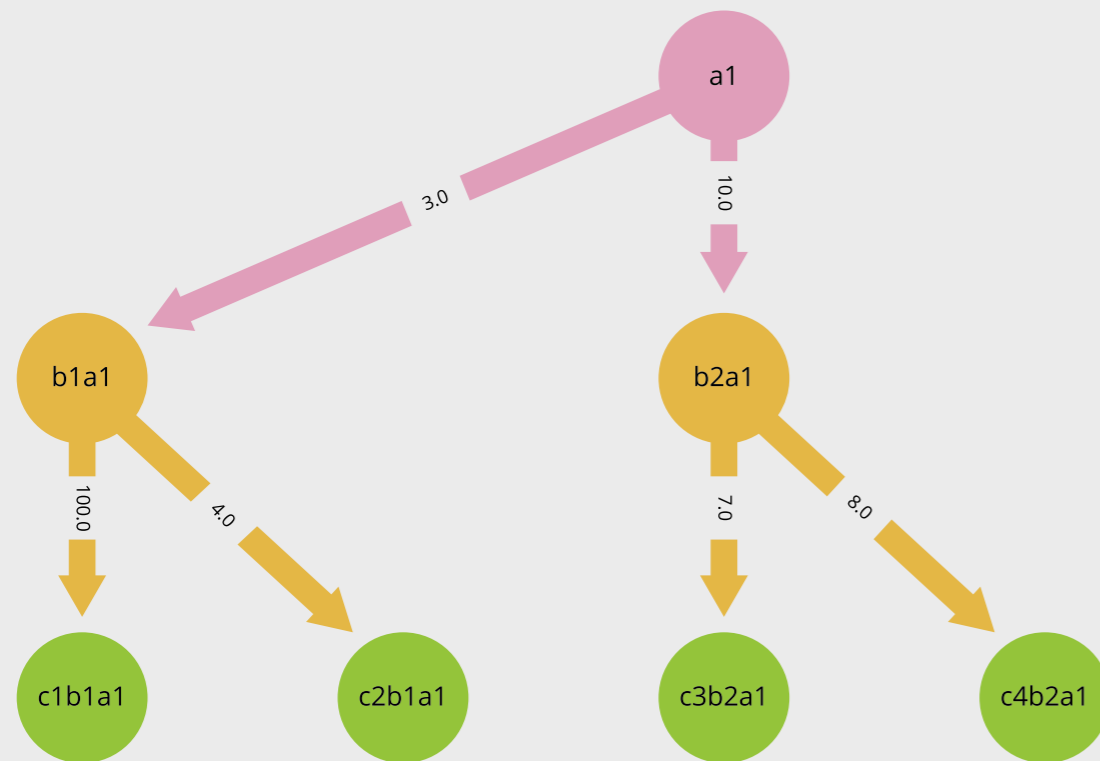
A trochu hezčí graf našeho stromu

```
edge_color_list = ["violet", "violet",
                  "darkorange", "darkorange", "darkorange", "darkorange"]
```

```
pos = graphviz_layout(muj_graf, prog="dot")
```

```
nx.draw(muj_graf, pos, node_color=color_list, edge_color = edge_color_list, width=5,
node_size=2000, with_labels=True)
```

```
labels = nx.get_edge_attributes(muj_graf, 'weight')
nx.draw_networkx_edge_labels(muj_graf, pos, edge_labels=labels)
plt.show()
```

```

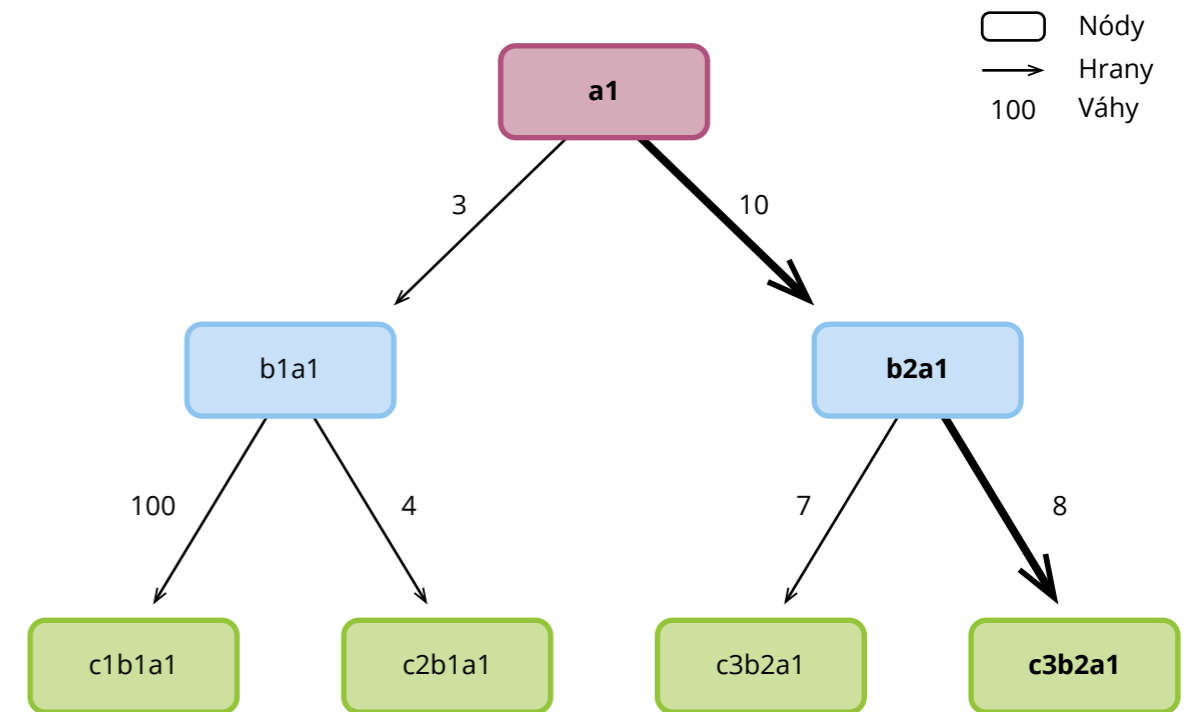
nodes = list(můj_graf.nodes)
nodes
['a1', 'b1a1', 'b2a1', 'c1b1a1', 'c2b1a1', 'c3b2a1', 'c4b2a1']
edges=list(můj_graf.edges)
edges
[('a1', 'b1a1'),
 ('a1', 'b2a1'),
 ('b1a1', 'c1b1a1'),
 ('b1a1', 'c2b1a1'),
 ('b2a1', 'c3b2a1'),
 ('b2a1', 'c4b2a1')]
můj_graf.edges[('a1', 'b1a1')]
{'weight': 3.0}
list(můj_graf.neighbors('c4b2a1'))
[]

můj_graf['a1']['b1a1']['weight']
3.0
  
```

Hladový algoritmus ve stromu

Příklad řešení hladového algoritmu ve stromu.

Nejprve sestavíme jednoduchý strom, co je na obrázku.



Strom s váhami hladový algoritmus

Formulace problému

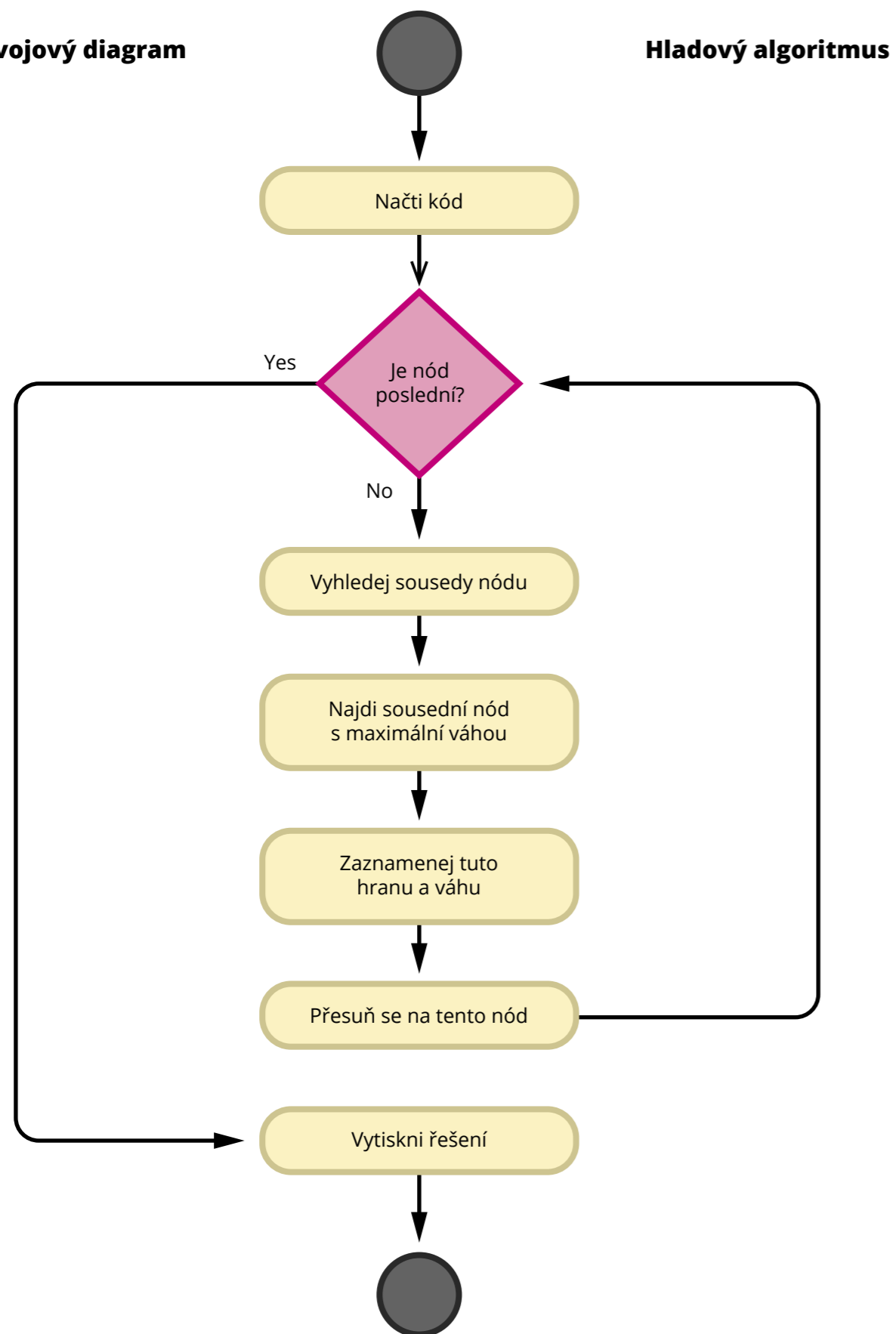
Hladový algoritmus si v každé části cesty vybírá nejziskovější trasu. To sice ve finále nemusí být nejvýhodnější, ale takto funguje.

Analýza úlohy

Začínáme u kořene stromu a pokračujeme k nejbližšímu sousedovi s nejlepší vahou. Končíme, když nód nemá žádné další sousedy.

Vytvoření algoritmu úlohy - vývojový diagram

Vývojový diagram



Sestavení programu a jeho odladění

```

import networkx as nx
import operator

můj_graf = nx.DiGraph()
můj_graf.add_nodes_from(['a1', 'b1a1', 'b2a1', 'c1b1a1', 'c2b1a1', 'c3b2a1', 'c4b2a1'])

můj_graf.add_edge("a1", "b1a1", weight=3.0)
můj_graf.add_edge("a1", "b2a1", weight=10.0)
můj_graf.add_edge("b1a1", "c1b1a1", weight=100.0)
můj_graf.add_edge("b1a1", "c2b1a1", weight=4.0)
můj_graf.add_edge("b2a1", "c3b2a1", weight=7.0)
můj_graf.add_edge("b2a1", "c4b2a1", weight=8.0)

node=list(můj_graf.nodes)[0]
while True:
    neighbors=list(můj_graf.neighbors(node))
    if len(neighbors) == 0:
        break
    max_weight=0
    branch={}
    for neighbor in neighbors:
        branch[(node, neighbor)] = můj_graf[node][neighbor]['weight']
    print(max(branch.items(), key=operator.itemgetter(1)))
    node = max(branch.items(), key=operator.itemgetter(1))[0][1]

(('a1', 'b2a1'), 10.0)
(('b2a1', 'c4b2a1'), 8.0)
  
```

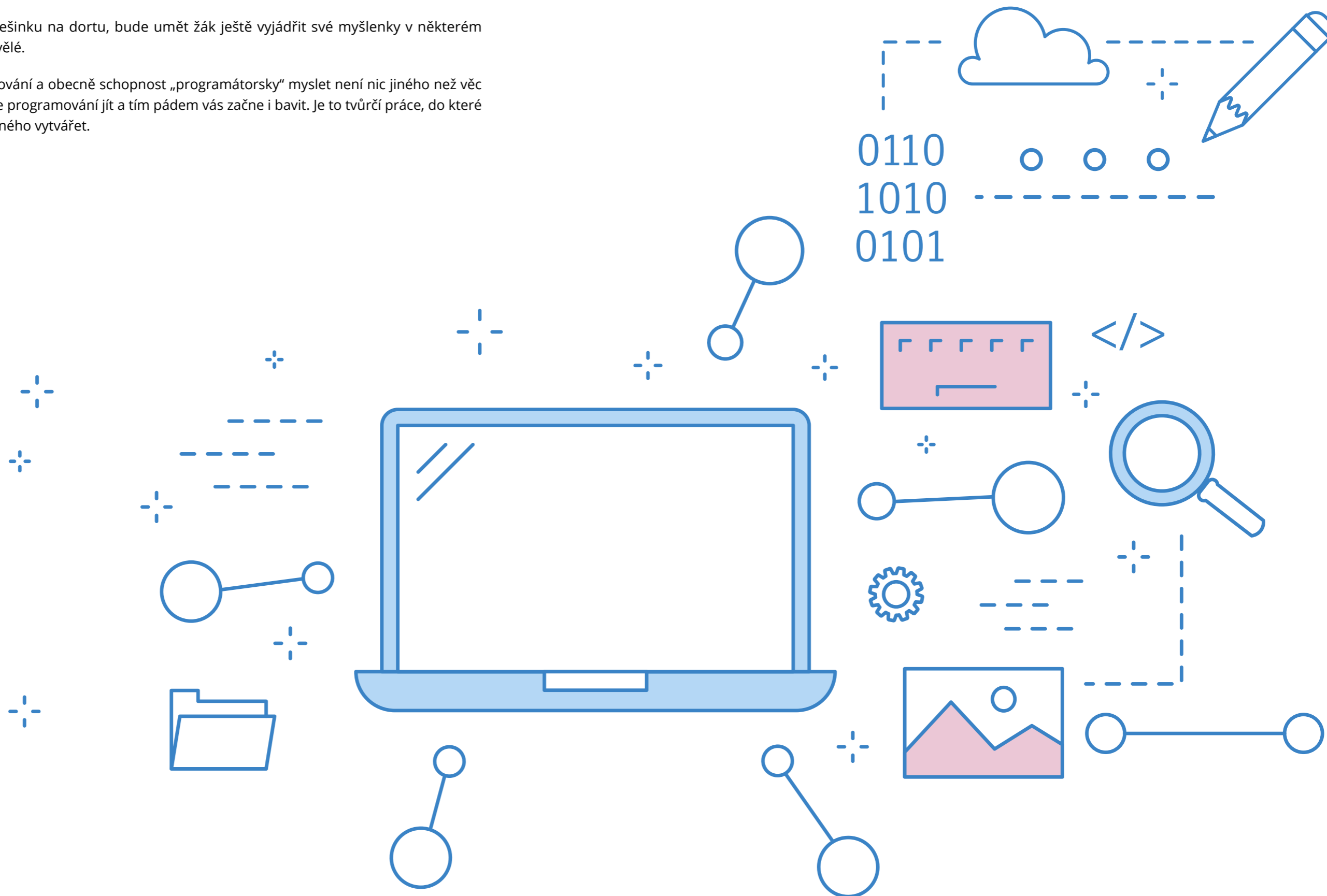
Co jste se naučili

Schopnost „programátorsky“ myslet patří dnes k důležitým dovednostem. Pokud tento kurz přispěje k tomu, aby učitelé i žáci uměli formulovat problém, analyzovat ho, zakreslit jednotlivé kroky řešení a rozhodování v programu a předvídání základních druhů chyb, které by při běhu mohly nastat, pak jsem spokojená.

Pokud k tomu, jako pověstnou třešinku na dortu, bude umět žák ještě vyjádřit své myšlenky v některém programovacím jazyce, bude to skvělé.

Stejně jako hra na piano, programování a obecně schopnost „programátorsky“ myslet není nic jiného než věc tréninku. Po určité době vám začne programování jít a tím pádem vás začne i bavit. Je to tvůrčí práce, do které můžete dát něco svého a něco cenného vytvářet.

Držím palce.



25 INTERNET VĚCÍ (IOT) PRO SŠ

Základní instrukce

Tento kurz je doporučen i pro studenty středních škol, kteří již mají nějaké základní znalosti v oblasti algoritmizace a programování. Ideální je pak znalost základu jazyka Python, nicméně i znalost základů jakéhokoliv jiného programovacího jazyka je postačující (jazyk Python bude využíván pouze na základní úrovni, která se příliš neliší od jiných programovacích jazyků)

Časová dotace tohoto kurzu významně závisí na dosavadních znalostech studentů v oblasti algoritmizace a programování. Je na konkrétním pedagogovi, aby na základě znalosti svých studentů rozhodl o délce kurzu.

Celé téma lze rozdělit do několika částí, přičemž některé se částečně prolínají a navazují na sebe. Celkově je postupováno od teoretického úvodu přes zprovoznění ESP 32 až k řešení praktických úloh.

1. Seznámení s problematikou internetu věcí obecně, definice základních pojmů a popis hlavních principů a účelu celého konceptu. Toto téma je volitelné a jedná se spíše o motivační úvod do dané problematiky. Iz tohoto důvodu není v rámci tohoto materiálu zpracováno ve formě pracovního listu.
 - a. Pokud toto téma nebylo dosud vyučováno, doporučuji mu věnovat jednu vyučovací hodinu.
 - b. V případě, že studenti již danou oblast a její principy znají, je možné toto téma vynechat.
2. Seznámení s ESP32, jeho možnostmi a programováním. Příprava a zprovoznění zařízení a jeho programování v jazyce Micropython. Komunikace pomocí sériové linky.
 - a. Vzhledem k tomu, že tato oblast bude patrně pro všechny studenty zcela nová, je nutné počítat minimálně s dvěma vyučovacími hodinami pro zvládnutí přípravy zařízení k činnosti a další hodinou pro ukázkou práce s Micropythonem.
3. Komunikační možnosti ESP32 – využití vestavěné Wi-Fi v režimu stanice. Vytvoření jednoduchého webového serveru běžícího na ESP32.
 - a. Toto téma je poněkud rozsáhlejší a zcela zásadní pro další oblasti. Je možné pouze odhadnout potřebný čas pro jeho zvládnutí (podle předchozích znalostí studentů). Minimální rozsah tématu tak lze odhadnout na dvě vyučovací hodiny.
4. Práce se vstupy – možnosti snímání hodnot externím čidlem a jejich publikace na webu.
 - a. Problematice doporučuji věnovat minimálně jednu až dvě vyučovací hodiny.
5. Práce s výstupy – ovládnutí RGB LED modulu, ukázková spolupráce vstupního a výstupního modulu, vytvoření funkčního IoT řešení s plným využitím možností ESP32.
 - a. Problematice doporučuji věnovat minimálně jednu až dvě vyučovací hodiny.

V rámci bodů 4 a 5 bude zahrnuto i spojení znalostí z předchozích oblastí do finálního celku reprezentujícího funkční aplikaci měřící teplotu v místnosti a reagující na ní rozsvícením kontrolní diody a informující uživatele prostřednictvím webu.

Na základě výše uvedeného lze odhadovat, že minimální doba pro realizaci celého kurzu činí cca 8–10 vyučovacích hodin (pro studenty zběhlejší v oblasti algoritmizace se programování a alespoň minimální znalostí jazyka Python). Tento údaj je však nutno považovat pouze za orientační.



Cílem tohoto kurzu je představit studentům problematiku internetu věcí (IoT). Tento pojem je relativně nový, ale do budoucna má značný potenciál. Proto je potřeba s ním studenty seznámit. V rámci kurzu se seznámíme s jedním zařízením používaným v oblasti IoT, konkrétně se jedná o vývojovou desku s procesorem ESP32. Pro experimenty s ní použijeme programovací jazyk Micropython a ukážeme si příklady konkrétního použití ESP32 pro Internet věcí.

Pro první část je vhodnou formou výuky výklad. Ten by měl být zastoupen i v ostatních částech, převažovat by u nich však postupně měla samostatná práce studentů. Výklad by tak měl být v posledních částech kurzu zaměřen především na zadání úlohy a případně vzorovou ukázkou řešení.

Vzhledem k možným rozdílům ve znalostech jednotlivých studentů lze také doporučit práci ve dvoučlenných týmech, a to především od části 2 dále.

U částí 4 a 5 je pak v druhé polovině již možné přistoupit k projektové výuce, kdy by studentské týmy měly (například i formou soutěže) vytvořit vlastní řešení IoT, které by následně prezentovaly.

Celý předkládaný kurz je rovněž nutno vybavit nebo podpořit příslušným technickým vybavením. V tomto případě se jedná především o zařízení ESP32 v počtu minimálně jeden kus na dva studenty. Zde je však nutné počítat i s rozdílnými úrovněmi znalostí studentů a je možné, že někteří studenti budou schopni kurz absolvovat samostatně a budou toto řešení preferovat. V takovém případě by měli mít k dispozici samostatnou sadu ESP32 a je tedy doporučeno mít tyto desky k dispozici pro cca 75 % studentů.

Ke každému ESP32 je nutno počítat s USB kabelem (shodný s kabely pro mobilní telefony s koncovkou microUSB). Pro ukázkou práce se vstupy a výstupy je dále nutné mít příslušná vstupní i výstupní zařízení. Doporučeno je pracovat s RGB LED diodou jako výstupem a čidly pro teplotu, tlak či osvětlení na vstupu (po jednom kusu pro každé ESP). Pro práci s komunikačními možnostmi ESP32 je pak velmi vhodné a preferované používat společnou Wi-Fi síť odemčenou pro přístup z neautorizovaných zařízení (bez kontroly MAC adresy, přístupové heslo je možné použít). Pro bližší kontakt studentů s danou technologií je vhodné rovněž počítat s využitím jejich mobilních telefonů. Pro práci se vstupy a výstupy je rovněž nutné mít k dispozici i nepájivé kontaktní pole (tzv. breadboard) s alespoň minimální sadou propojovacích kabelů.

V neposlední řadě je nutné upozornit na to, že zařízení se ve fázi programování a testování připojuje ke stolnímu počítači či notebooku ideálně s operačním systémem Linux nebo Windows. Po naprogramování pak může pracovat i samostatně. Pro ukázkou samostatného provozu ESP32 bez připojení k počítači je vhodné využít napájení prostřednictvím powerbanky pro mobilní telefony.

V oblasti softwaru je pak situace velmi jednoduchá. Využíván bude zejména specializovaný modul v Pythonu od firmy Espressif (výrobce ESP32) umožňující komunikaci ESP32 a rovněž software Micropython (<https://micropython.org/>), který je volně dostupný. Jako operační systém pro výuku je možné využít MS Windows, ev. Linux (jednodušší instalace podpory). Obecná podpora pak spočívá v instalaci jazyka Python v.3x a vývojového prostředí Thonny (<https://thonny.org/>).

Pro instalaci Micropythonu lze využít např. český návod „ESP32 Návod: podpora Micropythonu“ (https://chiptron.cz/articles.php?article_id=204), ev. postupu na webu <https://blog.vyoralek.cz/iot/esp8266-a-ESP32-zaloha-a-nahrani-noveho-firmware-pomoci-esptool/>.



Teoretická část k dané problematice

Oblast internetu věcí je v posledních několika letech stále aktuálnější. O co se však jedná? Jak již název naznačuje, základem je zde využití internetu, a tedy možnost komunikace mezi jednotlivými zařízeními. Internet je pro takovou komunikaci ideální platformou a umožňuje využít na něm existující nástroje a protokoly. Mezi nimi vyniká protokol TCP a na vyšší úrovni pak především protokol HTTP používaný při provozu webu.

Základním smyslem internetu věcí je zajistit schopnost jednotlivých koncových zařízení sbírat informace o svém provozu a svém okolí, tyto informace dále komunikovat a na základě jejich vyhodnocení a komunikace s ostatními zařízeními pak řídit jejich činnost.

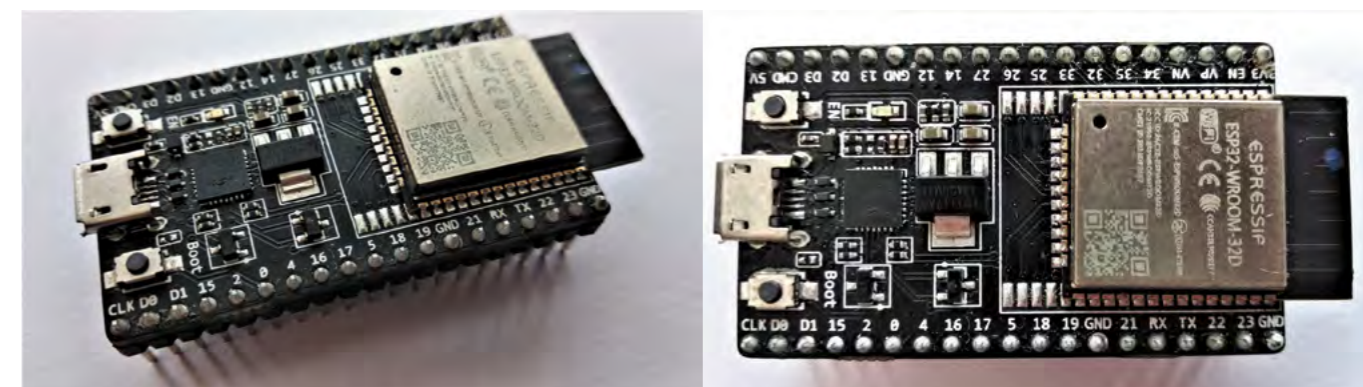
V rámci IoT je klíčová volba koncového zařízení, které by mělo umožnit výše zmíněnou komunikaci, a to jak směrem k přímo připojeným komponentám (čidla, akční prvky), tak do Internetu. Typickými představiteli jsou zařízení typu SoC (System on Chip). Mezi ně pak patří systémy typu Arduino vynikající zejména širokou škálou možných periférií, a to jak vstupních (čidla všeho druhu), tak výstupních (ovladače motoru, relé atd.). Zajištění vnější konektivity a komunikace u těchto zařízení je však poněkud problematictější, protože žádnou jednotnou komunikační platformou a přístupem k internetu samy o sobě nedisponují. Podstatně vyšší kategorii tvoří v tomto směru zařízení typu Raspberry Pi, které již běží na standardním operačním systému typu Linux. I u něj je možné využít periférie používané například s Arduinem.

Společným rysem těchto zařízení je pak jejich využití ve formě vloženého nebo vestavěného prvku (angl. embedded system) v jiném koncovém systému, ve kterém dosud mikroprocesorová technika byla využívána málo nebo vůbec. Typicky může jít například o domácí spotřebiče.

Ideálním řešením by však bylo zařízení někde na půl cesty mezi výše zmíněnými, tedy s velmi dobrou konektivitou k jednotlivým čidlům a výkonným prvkům, ale i s možností internetové konektivity. Takovým zařízením je pak ESP32.

SoC systém ESP32 disponuje bezdrátovým připojením Wi-Fi v pásmu 2,4 GHz, Bluetooth připojením (které je však v jazyce Micropython obtížně využitelné) a připojením pomocí USB rozhraní (takzvanou sériovou linkou). Procesor ESP32 je dvoujádrový. Kromě toho má zařízení řadu digitálních a analogových vstupů a rovněž digitálních výstupů s případnou regulací šířky pulzu (Pulse Width Modulation – PWM) používanou při řízení motorů či nastavení jasů LED osvětlení.

ESP32 lze programovat různými způsoby. V našem kurzu využijeme programovací jazyk Micropython a jeho konkrétní implementaci na ESP32.



Obrázek 1: Vývojová deska ESP32

Příklady z praxe

V závěru kurzu se zaměříme na ukázkou praktického využití principu IoT. Jak již však bylo řečeno, nejprve musíme vytvořit komunikační prostředí mezi ESP32 a Internetem, v našem případě webovým prohlížečem. V takto vytvořené platformě je pak již možné ukazovat využití vstupu a výstupu ESP32.

My si vyzkoušíme sběr informací o teplotě a tlaku v dané místnosti pomocí čidla a předávání těchto informací uživateli připojenému k internetu. Druhým příkladem bude ukázkou využití získané informace pro ovládání výstupního zařízení, v našem případě RGB LED diody. Ukázky jsou ryze praktické a ukazují, jakým způsobem je možno využít obdobnou technologii například pro řízení domácností a sběr informací nutných k jejímu řízení.

Absolvováním tohoto kurzu studenti získají nebo si prohloubí základní programátorské návyky. Dále se dozví informace ze světa hardware a integrace software a hardware. Zde záleží pouze na učiteli, který musí odhadnout úroveň svých žáků a jak hluboko si může dovolit ponořit se do dané problematiky.

Příklady zároveň studentům ukáží, že i na úrovni jejich znalostí lze s touto technologií experimentovat a prakticky využít ve vlastních projektech vytvářených jak pro domácí užití, tak v rámci školní výuky.

Metodická a didaktická část

Jak bylo vysvětleno v první kapitole, kurz je rozdělen do několika částí. Nyní se pozastavíme u každé z nich.

Seznámení s ESP32

V této části rozdělá učitel studentům zařízení ESP32 a USB kabel pro jeho připojení k počítači. Žáci poté své zařízení připojí ke svému PC nebo notebooku a provedou instalaci nutných komponent, aby bylo možné se zařízením ESP32 pracovat v jazyce Micropython (popsáno v pracovním listu). Funkčnost řešení studenti následně ověří otestováním krátkého programu v jazyce Python provedeného na zařízení ESP32. Rolí učitele je v tomto případě zejména vysvětlování a kontrola činnosti jednotlivých studentů a vysvětlení jednotlivých kroků, které jsou prováděny.

Internetové připojení ESP32

Tento bod navazuje na předchozí a studenti v něm implementují krátký program, který umožní připojení ESP32 do Wi-Fi sítě školy. Dalším krokem je pak vytvoření velmi jednoduchého webového serveru, na který bude možné se připojit z libovolného zařízení ve školní síti (detailed opět popsány v pracovním listu). Role učitele spočívá v tomto případě opět zejména ve vysvětlení jednotlivých programových kroků, které je nutné provést pro zajištění výše popsané funkcionality. Tento bod předpokládá alespoň základní znalosti studentů v oblasti webových technologií (základní znalost html, která však může být nahrazena širším výkladem učitele o této problematice).

ESP32 jako senzor

Cílem tohoto bodu je využití výstupů bodů předchozích a rozšíření funkcionality jednoduchého webového serveru tak, aby jeho prostřednictvím bylo možné získávat údaje z teplotního čidla, které bude k ESP32 externě připojeno (detailed opět v pracovním listu). Tyto údaje pak budou využity dvěma různými cestami – pro řízení výstupu přímo na ESP32 (v našem případě RGB LED) a také pro informování uživatele přes webový server.

Náplní bodu tak bude jednak vytvoření patřičného hardwarového zapojení a rovněž rozšíření funkcionality pro indikaci teploty a rozšíření webového serveru o prezentaci teploty. Role učitele bude zaměřena na kontrolu hardwarového zapojení a vysvětlení základních programových kroků, které bylo potřeba provést pro dosažení stanovených cílů. I tento bod vyžaduje alespoň základní znalosti jazyka html, které bude v případě nutnosti potřeba nahradit širším výkladem učitele v této problematice.

Doporučené pomůcky

Doporučenou pomůckou pro celý kurz je níže popsaná sada komponent, která tvoří základ pro všechny úkoly. Těchto sad musí být k dispozici odpovídající množství (jedna sada pro 1 – 2 studenty). Sadu tvoří:

- Vývojová deska ESP32
- Micro USB kabel
- Nepájivé kontaktní pole
- Modul RGB LED diody
- Tlakové a teplotní čidlo BMP180
- Propojovací kablíky (cca 10)

Zbytek vybavení je již pouze softwarový a vyžaduje mít k dispozici pro každé ESP32 1 PC nebo notebook s přístupem k internetu a v daném prostoru celkově dostupnou otevřenou Wi-Fi, aby bylo možné na ni ESP32 napojit.

Pracovní list

Přílohou tohoto materiálu jsou pracovní listy. Tyto pracovní listy jsou k dispozici v editovatelné elektronické formě, aby si je každý učitel mohl upravit, např. dle toho, co má již s žáky probráno.

Pracovní listy jsou čtyři, dva jsou zaměřeny na vytvoření předpokladů pro praktické užití ESP32, další dva pak na řešení praktických úkolů. V pracovních listech je předpokládána dostupnost všech komponent popsaných výše.

Pracovní list 1

Seznámení s ESP32

Co budeme potřebovat

- Počítač s nainstalovaným OS MS Windows s přístupem na internet, plnými přístupovými právy pro instalaci SW a dostatkem volné kapacity na pevném disku
- ESP32
- Propojovací USB kabel s microUSB koncovkou

A jdeme na to

Nejprve se seznámte s vývojovou deskou zařízení ESP32. Deska obsahuje samotný čip ESP32 a související doplňkové obvody a ovládací prvky. Těmi jsou dvě tlačítka po obou stranách microUSB konektoru. Pokud se na ESP32 díváte tak, že vidíte USB konektor, pak tlačítko vlevo je tlačítko resetu a tlačítko vpravo umožňuje přejít do bootovacího režimu, kdy je možné nahrávat software. Za čipem ESP32 také najdete integrovanou (na plošném spoji) anténu pro Wi-Fi a Bluetooth a po obou stranách dvě řady tzv. pinů umožňujících připojení dalších zařízení, případně další komunikaci.

Prvním krokem, který musíme provést, je instalace programovacího jazyka Python 3 a souvisejícího nástroje pro správu balíčků a rozšíření pip. Instalaci Pythonu 3 provedete stažením aktuální verze ze stránky www.python.org a následnou samotnou instalací, která se nijak neliší od jiných programů instalovaných na platformě Windows. Při instalaci zadejte, že cesta k Pythonu se má zařadit do proměnné PATH Windows, a že pro program Python bude vytvořena ikona na pracovní ploše.

Další kroky již budou probíhat v samotném Pythonu, kde pomocí programu pip3 nainstalujeme rozšíření pro ESP32.

```
pip3 install esptool
```

Dalším krokem je nutné již připojit ESP32 pomocí kabelu USB k počítači. Ten by měl stáhnout potřebné ovladače a začít komunikovat s ESP32 pomocí sériového portu. Právě označení sériového rozhraní je zásadní pro další činnost, neboť právě na tomto portu musíme s ESP32 komunikovat. Číslo nebo označení aktuálních sériových portů je možné zjistit pomocí příkazu „mode“ zadaného na příkazový řádek Windows (po spuštění příkazu CMD).

```
mode
```

Port ESP32 bude označen jako COM s navazujícím číslem (např. COM3). V dnešní době bývá tento sériový port jediný, který bývá připojen.

Dalším krokem je ověření komunikace s ESP32 pomocí skriptu esptool.py, který má značné možnosti a používá se pro všechnu komunikaci s ESP32. Zásadním parametrem uvedeného příkazu je označení portu, které jsme zjistili v předchozím kroku. Výstup zadaného příkazu nám pak identifikuje konkrétní zařízení ESP32 a uvede jeho základní charakteristiku. Pokud se tak nestane, je vhodné v průběhu komunikace stisknout na ESP32 tlačítko

bootu (toto řešení problémů s inicializací komunikace je použitelné i v následujících bodech naší úlohy).

```
esptool.py --port COMx flash_id
```

Nyní tedy máme ESP32 připojeno k počítači a jsme schopni s ním komunikovat. Dalším krokem je instalace jazyka Micropython na ESP32, přičemž nejprve je nutné provést výmaz paměti ESP32. To provedeme příkazem:

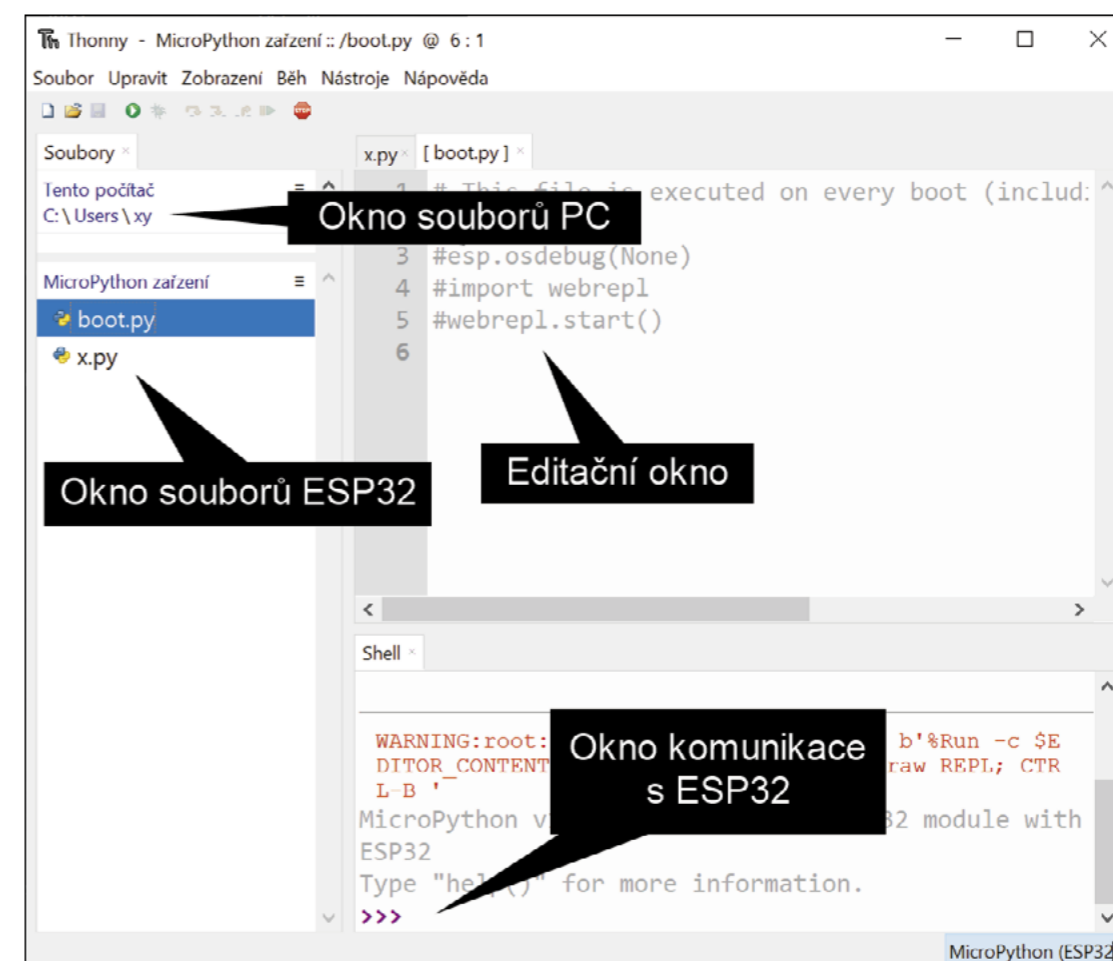
```
esptool.py --chip esp32 --port COMx erase_flash
```

Nyní k instalaci Micropythonu. Její první krok spočívá v získání vhodné verze tohoto jazyka ze stránek www.micropython.org. Při výběru aktuální verze nezapomeňte zvolit správný typ čipu, tedy ESP32. Soubor s koncovkou „bin“ uložte do počítače. (V době přípravy tohoto materiálu byla aktuální verze <https://micropython.org/resources/firmware/ESP32-20210418-v1.15.bin>.)

Nyní již převedeme získanou verzi Micropythonu přímo do paměti ESP32. To provedeme přiloženým příkazem, kde uvedete jako poslední parametr aktuální název a umístění vaší verze Micropythonu, kterou jste v předchozím kroku stáhli. Proces instalace jazyka chvíli trvá, buďte proto trpěliví.

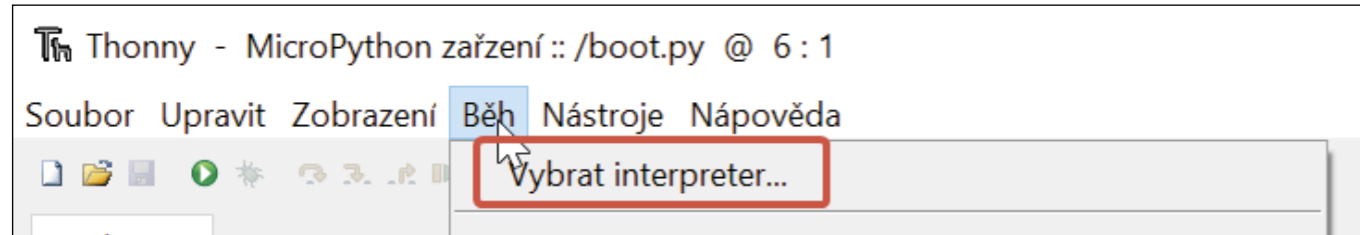
```
esptool.py --chip esp32 --port /dev/ttyUSB0 --baud 460800 write_flash -z 0x1000 /cesta/xxxx.bin
```

Nyní tedy máme ESP32 kompletně připraveno pro použití s jazykem Micropython. Posledním krokem je tedy ještě instalace vývojového prostředí na stolní počítač. Tímto prostředím bude jednoduchý programový editor Thonny, který najdete na adrese www.thonny.org a jeho instalaci provedete stejným způsobem jako instalaci jazyka Python.



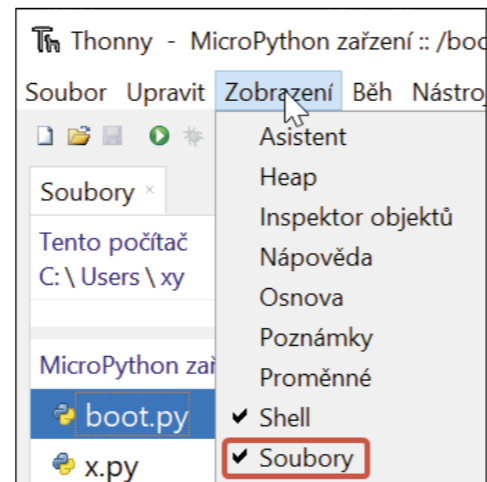
Obrázek 2: Editor Thonny (české lokalizace je experimentální)

Prostředí Thonny je nutné upravit pro použití ESP32, kde v nabídce Běh (Run) zvolíte výběr interpreteru – z dostupných možností zvolíte „MicroPython (ESP32)“.

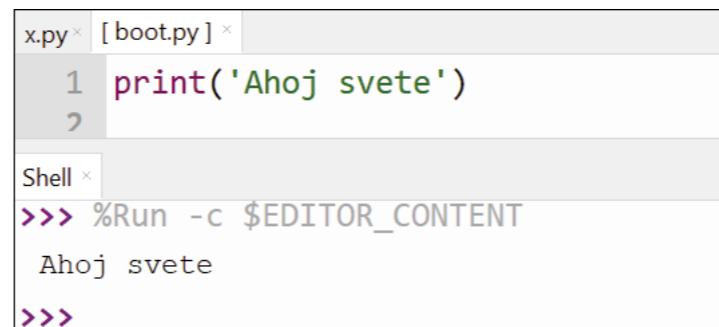


Obrázek 3: Výběr správného interpreteru (následuje volba „MicroPython (ESP32)“)

Vhodné je také z nabídky zobrazení zaškrtnout volbu soubory, kde v levé části uvidíte dvě sekce z nichž jedna bude váš stolní počítač a druhá sekce bude zařízení ESP32. V této sekci bude jediný soubor s názvem boot.py, který ESP32 spouští při každém startu.



Obrázek 4: Výběr zobrazení panelu souborů



Obrázek 5: Ověření funkčnosti komunikace s ESP32

Nyní již můžeme do hlavního editoru zapsat jednoduchý příkaz „print(“Hello World““), který následně necháme vykonat pomocí spouštěcí šipky na mikrokontroleru (ESP32). Ve spodní části okna by se měl vypsát příslušný text.

Co jste se naučili

V této části jsme se naučili připojit zařízení ESP32 k počítači a celou sestavu připravit takovým způsobem, aby bylo možné ESP32 programovat prostřednictvím jazyka MicroPython. V dalších krocích pak budeme s takto vytvořenou sestavou dále pracovat a zaměříme se již na funkcionalitu programovanou pro ESP32.

Pracovní list 2

ESP32 na Internetu

Co budeme potřebovat

- Počítač s nainstalovanými programy z Pracovního listu 1 a s přístupem na internet
- Funkční Wi-Fi síť umožňující připojení dalších zařízení (bez filtrace MAC adres, heslo není překážkou, funkční server DHCP pro přidělení IP adresy)
- ESP32 a propojovací USB kabel
- Libovolná powerbanka pro testování samostatného provozu ESP32

A jdeme na to

Připojte k počítači zařízení ESP32. Otevřete si vývojové prostředí Thonny a jednoduchým příkazem ověřte, že jste schopni s ESP32 komunikovat (viz Pracovní list 1). V této části zajistíme, aby ESP32 komunikovalo s Internetem, a to na úrovni webových stránek.

V prvním kroku si ověříme, jaké Wi-Fi sítě jsou v daném prostoru k dispozici. Příslušný programový kód máte uvedený. Pokud chcete, aby byl spouštěn ihned po zapnutí ESP32, pište jej do souboru s názvem „boot.py“.

Jen připomeňme, že ESP32 pracuje pouze ve frekvenčním pásmu 2,4 GHz. Mezi zobrazenými sítěmi by měla být rovněž ta, kterou ve škole používáte. Programu, který zde máte zobrazený (Obrázek 6), se říká Wi-Fi scanner a umožňuje zjistit, jaké sítě jsou v daném okolí k dispozici.

```
import network

station = network.WLAN(network.STA_IF)

station.active(True)

wifis = station.scan()

print(wifis)
```

```
>>> %Run -c $EDITOR_CONTENT
[(b'...', b'\x00rcFO\xe6', 6, -43, 4, False), (b'...', b'\xd5G\x07\xc6\x80', 3, -56, 3, False), (b'...', b'\x00rcFR/', 8, -64, 4, False), (b'...', b'\xcc\xb2Ua\xa7<', 7, -71, 4, False), (b'...', b'\x00rc\xfa:', 12, -72, 3, False), (b'...', b'\xe4\xbe\xed\x0e\x0e\x83', 4, -74, 4, False), (b'...', b'\xd8G2\xc5\xc2\xfa', 3, -77, 3, False), (b'...', b'\xd1T\xf6\x88\xf0', 9, -78, 4, False), (b'...', b'\xc8\x00\x90\xebT', 1, -84, 3, False), (b'...', b'\xdc\xcfW\xe8', 12, -89, 4, False), (b'...', b'\x90\D\x87\r", 6, -90, 4, False), (b'...', b'\xc8\xd3\xa34\xeb\n', 7, -90, 4, False), (b'...', b'\x04\x8d8FJ\xf3', 3, -91, 4, False), (b'...', b'8C)e(W', 11, -93, 4, False), (b'...', b'\xa8^EH\x91\xa4', 1, -94, 3, False)]
```

Obrázek 6: Výpis Wi-Fi sítí v okolí – výstup (jména sítí (SSID) jsou rozmazána)

Druhým krokem je připojení k Wi-Fi, kterou jste si zvolili pomocí jejího názvu (tzv. SSID). Síť může být otevřená nebo vyžadovat ověření vaší totožnosti pomocí hesla. Tato možnost je běžnější a pokud tomu tak je, obraťte se na svého vyučujícího, aby vám heslo sdělil. Pokud do Thonny přepíšete kód uvedený níže a spustíte jej, připojíte se k vámi uvedené síti Wi-Fi. Toto bude potvrzeno níže uvedeným výpisem od ESP32, který vám zobrazí aktuální nastavení vašeho Wi-Fi. Z uvedených čtyř IP adres konfigurace sítě je zásadní zejména ta první, která udává, jaká IP adresa byla vašemu ESP32 přidělena. V případě, že výpis těchto adres nevidíte, zkuste restart ESP32 tlačítkem RESET nebo odpojte ESP32 od počítače a následně opět připojte a celý proces opakujte.

```
ssid = 'sitxx'

passwd = 'xxxx'

import network

import time

sta_if = network.WLAN(network.STA_IF)

if not sta_if.isconnected():

    print('connecting to network...')

    sta_if.active(True)

    sta_if.connect(ssid, passwd)

    while not sta_if.isconnected():

        sleep(1)

print('network config:', sta_if.ifconfig())
```

Dalším krokem je zprovoznění jednoduchého webového serveru na ESP32. Využijeme předchozí kód pro připojení k Wi-Fi síti a doplníme jej dle Obrázku 8. Ověření funkcionality celého řešení provedete velmi snadno. Prostřednictvím webového prohlížeče na stolním počítači se připojte na IP adresu, kterou jste zjistili v předchozím bodu. Pokud vidíte odpověď vašeho webového serveru na ESP32, vše jste provedli správně.

```
try:

    import usocket as socket

except:

    import socket

html = """<html><head></head><body><h2>Vase ESP32 vas vita!</h2></body></html>"""
```

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

s.bind(('', 80))

s.listen(5)

print('starting server')

while True:

    conn, addr = s.accept()

    print('Connection request from %s' % str(addr))

    response = html

    conn.send('HTTP/1.1 200 OK\n')

    conn.send('Content-Type: text/html\n')

    conn.send('Connection: close\n\n')

    conn.sendall(response)

    conn.close()

    print('Connection closed')
```

```
>>> %Run -c $EDITOR_CONTENT
network config: ('192.168.43.194', '255.255.255.0', '192.168.43.1', '192.168.43.1')
starting server
Connection request from ('192.168.43.116', 49490)
Connection closed
```

Obrázek 8: Jednoduchý web server – výstup

Uvedený program pracuje v nekonečné smyčce, činnost lze ukončit vypnutím ESP32.

Pokud jste kód vložili do souboru „boot.py“, je možné ESP32 odpojit od PC a připojit např. k powerbance a provozovat jej i takto samostatně.

Co jste se naučili

Při plnění tohoto vcelku náročného pracovního listu jste se naučili využít nastavené komunikace se zařízením ESP32, abyste jej mohli programovat a měnit jeho chování. Základním chováním je však komunikace ESP32 a my jsme si ukázali, jak tuto komunikaci zajistit prostřednictvím sítě Wi-Fi tak, aby se vaše ESP32 chovalo jako webový server a vy jste mohli v dalších krocích právě tento webový server obohacovat o další funkcionality.

Pracovní list 3

ESP32 jako senzor

Co budeme potřebovat

- Počítač s nainstalovanými programy z Pracovního listu 1 a 2 a s přístupem na internet
- Funkční Wi-Fi síť umožňující připojení dalších zařízení (bez filtrace MAC adres, heslo není překážkou, funkční server DHCP pro přidělení IP adresy)
- ESP32 a propojovací USB kabel
- Čidlo využívající čip BMP180

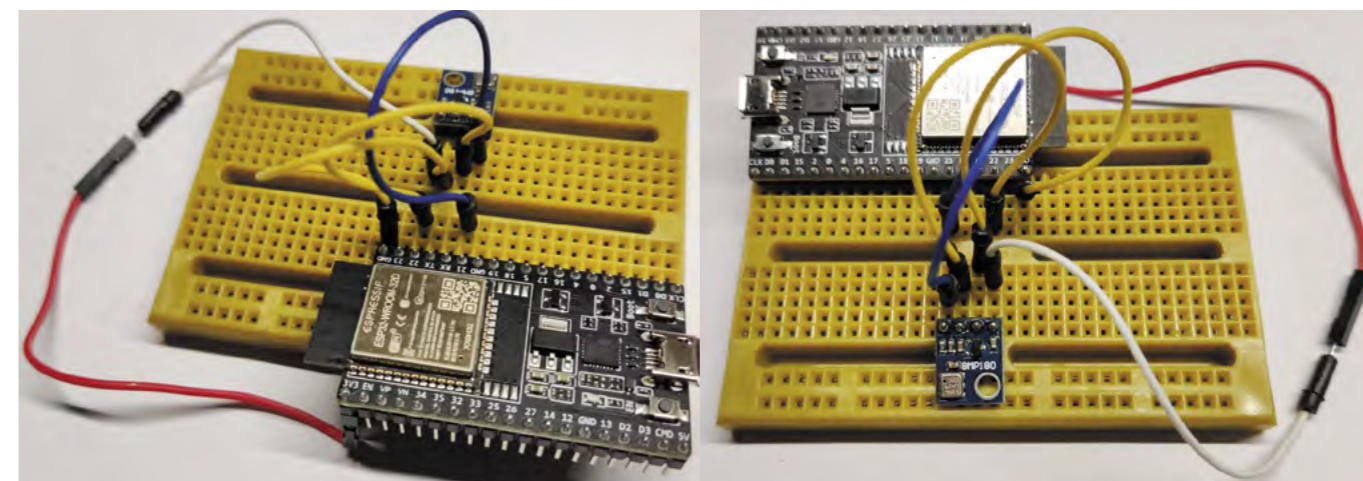
A jdeme na to

Připojte k počítači zařízení ESP32. Otevřete si vývojové prostředí Thonny a jednoduchým příkazem ověřte, že jste schopni s ESP32 komunikovat (viz Pracovní list 1). V této části zajistíme, aby ESP32 snímalo informace z externího čidla teploty a tlaku a tyto informace publikovalo přes webový server.



Obrázek 9: Modul čidla BMP180

V prvním kroku zprovozníme externí čidlo. K tomu budete potřebovat nepájivé kontaktní pole se zapojením dle obrázku. Čidlo BMP180 je digitální a s ESP32 komunikuje pomocí tzv. I2C sběrnice. Aby tato komunikace pracovala, je potřeba do ESP32 nahrát obslužnou knihovnu pro BMP180, kterou najdete na adrese <https://github.com/micropython-IMU/micropython-bmp180>. Soubor bmp180.py nahrajte pomocí Thonny do ESP32.



Obrázek 10: Zapojení ESP32 a čidla BMP180

Nyní již můžeme přistoupit k programování ESP32. Kód pro čtení hodnot z čidla následuje. Informace z externího čidla budou aktualizovány po 5 sekundách.

```
import time

from bmp180 import BMP180

from machine import SoftI2C, Pin

bus = SoftI2C(scl=Pin(22), sda=Pin(21), freq=100000)

bmp180 = BMP180(bus)

bmp180.oversample_sett = 2

bmp180.baseline = 101325

while True:

    temp = bmp180.temperature

    press = bmp180.pressure

    altitude = bmp180.altitude

    print("Temperature: ", temp)

    print("Pressure: ", press)

    print("Altitude: ", altitude)

    time.sleep(5)
```

Pro zobrazování hodnot přes webový server je potřeba uvedený nekonečný cyklus zrušit a upravit kód webserveru. Ten nyní bude prohlížeč uživatele informovat o tom, že stránka se bude obnovovat každých 5 sekund a při každém obnovení se aktualizují hodnoty z čidla.

Celý kód upraveného webserveru včetně čtení z čidla následuje (kód předpokládá funkční připojení ESP32 k wifi, viz Pracovní list 2):

```
try:
    import usocket as socket
except:
    import socket

import time

from bmp180 import BMP180
from machine import SoftI2C, Pin

bus = SoftI2C(scl=Pin(22), sda=Pin(21), freq=100000)
bmp180 = BMP180(bus)
bmp180.oversample_sett = 2
bmp180.baseline = 101325

def temp():
    temp = bmp180.temperature
    return str(temp)

def web_page():
    html = """
<html>
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta http-equiv="refresh" content="2" />
</head>
<body>
    <h2>Vase ESP32 vas vita</h2>
    <p>Aktualni teplota je (C): """+temp()+"</p>
</body>
</html>
"""
```

```
return html

def server():
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind(('', 80))
    s.listen(5)

    while True:
        try:
            conn, addr = s.accept()
            conn.settimeout(3.0)
            print('Received request from %s' % str(addr))
            request = conn.recv(1024)
            conn.settimeout(None)
            request = str(request)
            print('GET Request Content = %s' % request)
            response = web_page()
            conn.send('HTTP/1.1 200 OK\n')
            conn.send('Content-Type: text/html\n')
            conn.send('Connection: close\n\n')
            conn.sendall(response)
            conn.close()
        except OSError as e:
            conn.close()
            print('Connection closed')

server()
```

Jak je vidět, z čidla se zde využívá pouze informace o teplotě, doplnění dalších informací můžete zkusit samostatně.

Co jste se naučili

V tomto Pracovním listu jsme navázali na Pracovní list 2 a ukázali jsme si, jak připojit externí digitální čidlo k ESP32 a jak z něj získávat hodnoty teploty, tlaku a informaci o nadmořské výšce. Také jsme se naučili, jak informaci o teplotě publikovat pomocí webového serveru.

Pracovní list 4

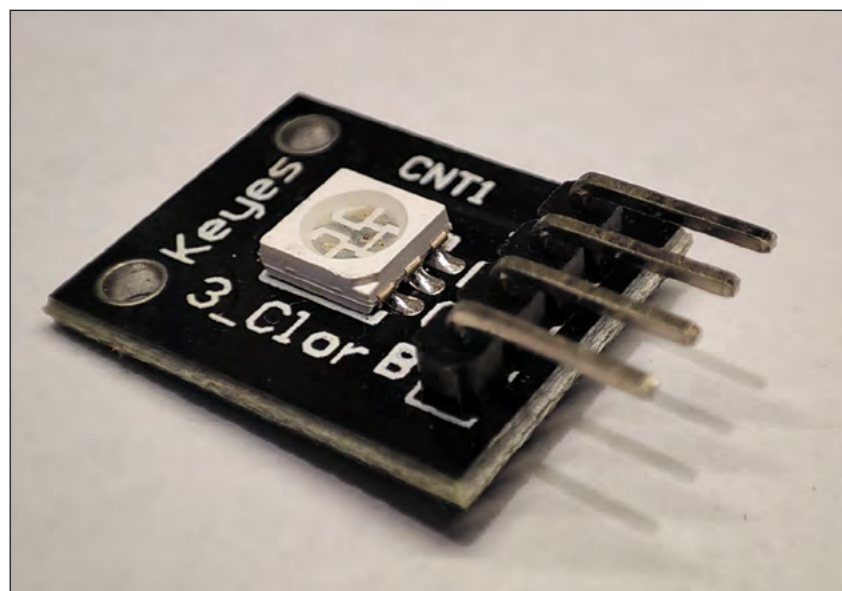
ESP32 jako IoT zařízení

Co budeme potřebovat

- Počítač s nainstalovanými programy z Pracovního listu 1 a 2 a 3 s přístupem na internet
- Funkční Wi-Fi síť umožňující připojení dalších zařízení (bez filtrace MAC adres, heslo není překážkou, funkční server DHCP pro přidělení IP adresy)
- ESP32 a propojovací USB kabel
- Čidlo využívající čip BMP180, RGB LED modul

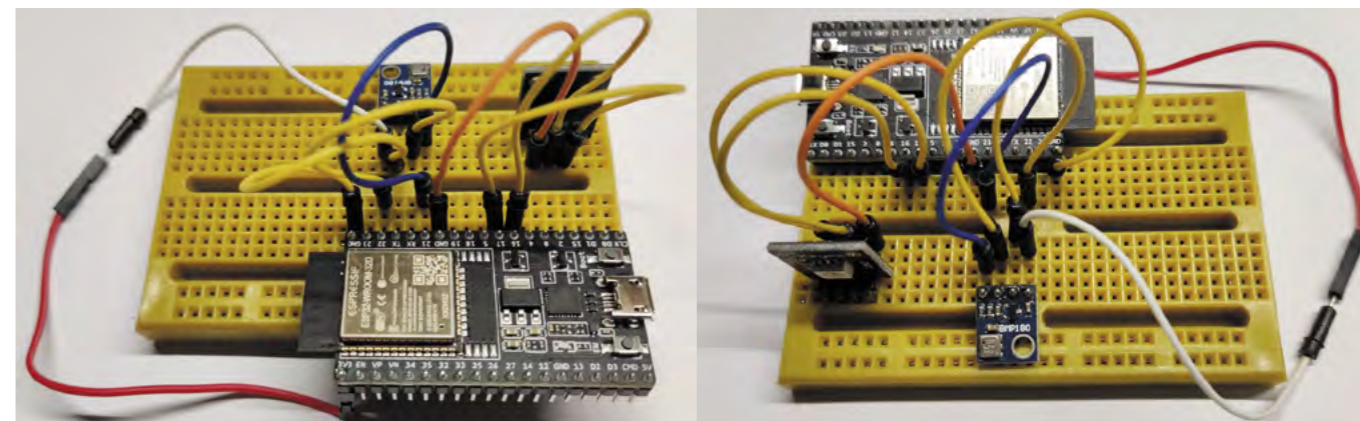
A jdeme na to

Připojte k počítači zařízení ESP32. Otevřete si vývojové prostředí Thonny a jednoduchým příkazem ověřte, že jste schopni s ESP32 komunikovat (viz Pracovní list 1). V této části nejprve zprovozníme RGB LED modul umožňující indikaci stavu zařízení či naměřených hodnot. V druhé části se pak podíváme, jak zkombinovat činnosti RGB modulu s výstupy Pracovního listu 3.



Obrázek 11: Modul RGB LED

Nejprve tedy zprovoznění RGB LED modulu. Pro to budete potřebovat nepájivé kontaktní pole se zapojením dle Obrázku 12. RGB LED modul je externí modul, na kterém je umístěna RGB dioda se společnou anodou nebo katodou.



Obrázek 12: Zapojení modulů BMP180 a RGB LED

Dále uvedený kód pak umožňuje ovládat jednotlivé barevné složky RGB.

```
import time

from machine import Pin

led_red = Pin(16, Pin.OUT)
led_green = Pin(17, Pin.OUT)
led_blue = Pin(18, Pin.OUT)

led_red.value(0)
led_green.value(0)
led_blue.value(0)

while True:

    led_red.value(1.0)

    time.sleep(0.5)

    led_red.value(0)

    time.sleep(0.5)

    led_green.value(1.0)

    time.sleep(0.5)

    led_green.value(0)

    time.sleep(0.5)

    led_blue.value(1.0)
```



```

time.sleep(0.5)

led_blue.value(0)

time.sleep(0.5)

```

Druhým cílem je využít informaci o teplotě z externího modulu dle Pracovního listu 3 pro řízení barvy RGB diody. Kompletní kód tohoto úkolu následuje (je zřejmé, že je nutné doplnit i hardwarové zapojení dle Pracovního listu 3). Rozhodující teplotou je zde 30 °C, pod touto teplotou svítí LED červeně (teplota je příliš nízká), nad ní pak zeleně (teplota je dostatečná). Měření teploty probíhá každých 5 sekund.

```

import time

from machine import Pin

led_red = Pin(16, Pin.OUT)
led_green = Pin(17, Pin.OUT)
led_blue = Pin(18, Pin.OUT)

led_red.value(0)
led_green.value(0)
led_blue.value(0)

from bmp180 import BMP180
from machine import SoftI2C, Pin

bus = SoftI2C(scl = Pin(22), sda = Pin(21), freq = 100000)

bmp180 = BMP180(bus)

bmp180.oversample_sett = 2
bmp180.baseline = 101325

while True:

    temp = bmp180.temperature

    if temp < 30:

        led_red.value(1)

        led_green.value(0)

```

```

else:

    led_green.value(1)

    led_red.value(0)

time.sleep(5)

```

Naším finálním projektem bude spojení funkcionalit z Pracovního listu 3 a dosud uvedených v tomto pracovním listu. Zde ale narazíme na určitý problém. Chceme totiž pravidelně měřit teplotu a reagovat na ni dvěma odlišnými postupy, a to současně. Tento stav jasně vede na paralelní zpracování informace o teplotě, což se dá zajistit zprovozněním více tzv. programových vláken. Kompletní kód tohoto již pokročilého úkolu následuje (včetně částí zajišťujících připojení k síti Wi-Fi).

```

import time

from machine import Pin

led_red = Pin(16, Pin.OUT)
led_green = Pin(17, Pin.OUT)
led_red.value(0)
led_green.value(0)

from bmp180 import BMP180
from machine import SoftI2C, Pin

bus = SoftI2C(scl=Pin(22), sda=Pin(21), freq=100000)

bmp180 = BMP180(bus)

bmp180.oversample_sett = 2
bmp180.baseline = 101325

import _thread

act_temp = '0'

def update():

    global act_temp

```

```

while True:

    temp = bmp180.temperature

    if temp<30:

        led_red.value(1)

        led_green.value(0)

    else:

        led_green.value(1)

        led_red.value(0)

    time.sleep(5)

    act_temp=str(temp)

_thread.start_new_thread(update, ())

import network

def connect(ssid,passwd):

    import network

    sta_if = network.WLAN(network.STA_IF)

    if not sta_if.isconnected():

        print('connecting to network...')

        sta_if.active(True)

        sta_if.connect(ssid, passwd)

        while not sta_if.isconnected():

            pass

        print('network config:', sta_if.ifconfig())

connect('sitSSID','sitPASSWD')

try:

    import usocket as socket

except:

```

```

import socket

def web_page(temp):

    html = """

<html>

<head>

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <meta http-equiv="refresh" content="5" />

</head>

<body>

    <h2>Vase ESP32 vas vita</h2>

    <p>Aktualni teplota je (C): """+temp+"""/>

</body>

</html>

"""

    return html

def server():

    global act_temp

    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    s.bind(('', 80))

    s.listen(5)

    while True:

        try:

            conn, addr = s.accept()

            conn.settimeout(3.0)

            print('Received request from %s' % str(addr))

            request = conn.recv(1024)

            conn.settimeout(None)

            request = str(request)

```

```

print('GET Request Content = %s' % request)

response = web_page(act_temp)

conn.send('HTTP/1.1 200 OK\n')

conn.send('Content-Type: text/html\n')

conn.send('Connection: close\n\n')

conn.sendall(response)

conn.close()

except OSError as e:

    conn.close()

    print('Connection closed')

server()

```

Klíčová je zde funkce „update“, která zjišťuje informaci o teplotě z čidla každých 5 sekund a také na tuto teplotu reaguje nastavením správné barvy RGB diody (místo které je možné si představit například ovládání topení v rodinném domě). V každém cyklu je také aktualizována globální proměnná „act_temp“. Funkce „update“ je spouštěna v samostatném pracovním programovém vláknu, což zajišťuje nezávislé fungování.

V hlavním vláknu našeho ESP32 pak běží webový server, který po přijetí požadavku od webového prohlížeče vrátí uživateli aktuální teplotu převzatou právě z proměnné „act_temp“. Stránka s údajem o teplotě je aktualizována opět každých 5 sekund.

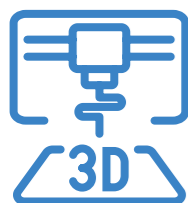
Co jste se naučili

V Pracovním listu 4 jsme navázali na Pracovní listy 2 a 3 a ukázali jsme si, jak ovládat externí RGB LED modul. V druhé části jsme pak zajistili, že teplota zjištěná externím teplotním čidlem řídí RGB diodu. V poslední části jsme pak k již získané funkcionalitě doplnili publikaci hodnoty teploty přes webový server s tím, že pro správnou funkci bylo nutné použít vláknové programování ESP32.



2026

3D MODELOVÁNÍ A 3D TISK PRO SŠ



Cílem kurzu je představit možnosti využití 3D modelování a 3D tisku technologií Fused Filament Fabrication (nanášení roztavené plastové struny) ve výuce na středních školách. Je zde představen celý proces od náčrtku, přes 3D modelování, slicing, až po samotný tisk a možnosti postprocesingu. Na začátku kurzu jsou účastníci seznámeni s bezpečností práce a základy zacházení s 3D tiskárnou. Součástí jsou také vzorové úlohy včetně řešení, které je možné použít ve výuce.

Základní instrukce

Tento kurz je doporučen žákům středních škol a gymnázií (bez ohledu na zaměření). Autor má zkušenosti s výukou této látky pro studenty vysoké školy, firemní zaměstnance a širší veřejnost. Druhý autor má zkušenosti s výukou této látky již od šesté třídy ZŠ a v rámci kroužku. Tento kurz lze vyučovat v rámci všech ročníků středních škol a gymnázií. V rámci kurzu jsou představeny postupy a metodiky tvorby 3D modelů a možnosti jejich tisku pomocí technologie FFF (FDM).

Časová dotace tohoto kurzu není striktně dána. Může záviset na úrovni všech účastníků kurzu a jejich zkušeností s 3D modelováním, respektive 3D tiskem. Vyučující by se měl řídit časovou dotací na dané škole. Doporučená časová dotace jsou dvě spojené hodiny týdně, jak v případě výuky, tak v případě kroužku. Tisknutí pomocí 3D tiskárny je doporučeno při výuce nebo přes pracovní dny, kdy je možná kontrola o přestávkách či vzdálený dohled s možností zásahu.

Níže naleznete doporučený průběh a časovou dotaci. Kurz lze rozčlenit do tří částí:

1. Úvod

- a. Bezpečnost práce (osobní + zařízení) - v rámci tohoto kurzu by měli žáci/studenti dodržovat základní bezpečnost práce v IT nebo jiné učebně, která je dána řádem učebny. Dále budou seznámeni s bezpečností práce a užívání 3D tiskárny a příslušenství.
- b. Seznámení s 3D tiskárnou a první tisk (tisk poté necháme běžet i přes výuku).
- c. Od modelu k tisku – ještě, než začneme s žáky/studenty vytvářet modely, musíme vysvětlit základní postup této tvorby. Popis jednotlivých dílčích kroků:
 - 2D náčrt
 - 3D model
 - Slicing modelu pro tisk
 - Tisk a jeho příprava
 - Následné zpracování

2. 3D modelování

- a. 2D – slouží k načrtnutí, kótování a zavazbení základního nákresu 2D zobrazení budoucího modelu.
- b. 3D – slouží k převedení dvojrozměrného nákresu do trojrozměrné podoby, kde je možné získaný model dále upravovat.

3. 3D tisk

- a. Slicing – úprava a příprava již hotového modelu na samotný tisk.
- b. Orientace objektu pro tisk
- c. Práce s 3D tiskárnou
- d. Vlastnosti 3D tiskárny (FDM)

Autoři:

Mgr. Jakub Geyer, Přírodovědecká fakulta, Jihočeská univerzita v Českých Budějovicích

Mgr. Tomáš Sosna, Pedagogická fakulta, Jihočeská univerzita v Českých Budějovicích

Editor: doc. RNDr. Ing. Jana Kalová, Ph.D.

Minimální doba, za kterou lze tento výukový balík absolvovat, je tedy čtyři vyučovací hodiny – první dvouhodinová: část 1–2, druhá dvouhodinová: část 2–3. Doporučený počet je 10 hodin (2, 2, 2, 2) včetně představení projektů. Je třeba počítat s možností, že za běhu bude nutné přidat či ubrat hodiny.

Pro část 1 a úvod do problematiky částí 2 a 3 je vhodnou výukovou metodou frontální výuka spojená se samostatnou prací žáků/studentů v hodině. Učitel by měl vždy část látky vysvětlit a ukázat, poté nechat žáky/studenty, aby si vše vyzkoušeli a stíhali pracovat všichni najednou.

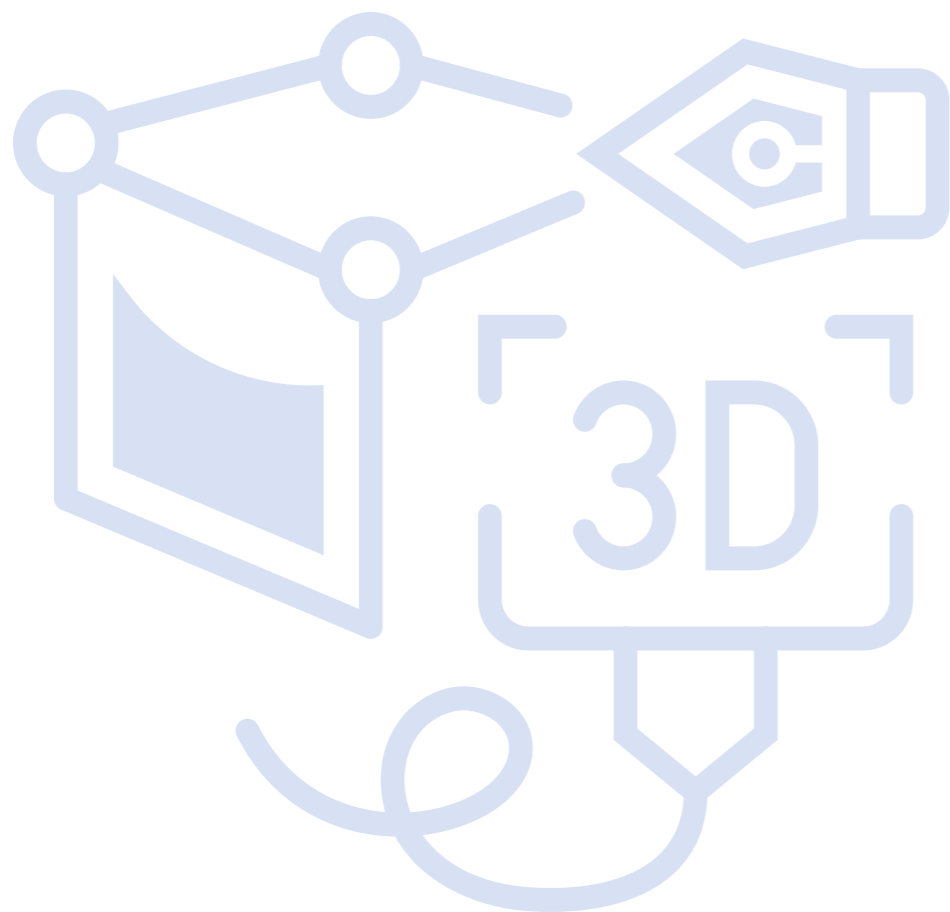
V závěrečných hodinách pak lze s úspěchem použít projektovou výuku, kdy žáci buď dostanou přidělené téma nebo si jej zcela sami zvolí na základě dosažené úrovně konstruování. Tyto projekty by si měli žáci sami navrhnout, rozměřit, vymodelovat, připravit k tisku a vytisknout.

Doporučená literatura:

Průša, J. (2014). *Základy 3D tisku*. (<https://www.prusa3d.cz/kniha-zaklady-3d-tisku-josefa-prusi/>).

Vláčilová, H., Vilímková, M., & Hencl, L. (2006). *SolidWorks*. Brno: Computer Press.

Jako další zdroj je vždy důležité seznámit se s manuálem ke konkrétní 3D tiskárně.



Teoretická část k dané problematice

CAD

CAD v překladu znamená počítačem podporované kreslení nebo rýsování. Je to program, který používáme po celou dobu konstrukce (tvorba součástí, výkresu, animace...). Původně se s CAD systémem počítalo jako s programem pro navrhování integrovaných spojů v počítačích, až později se začal používat ve strojírenství a architektuře, kde se jednalo o navrhování součástí a staveb. Ještě později se CAD začíná používat v geodézii, kartografii a geografických informačních systémech (vazba na databáze). Objevení CAD technologií kvalitně posunulo metodiku konstruování.

Asi největší předností počítačového návrhu je jeho možnost návaznosti na další technologické činnosti. Jako příklad lze uvést některé komplikované tvary při výrobě automobilů.

Výběr softwaru

V dnešní době existuje na trhu velké množství různých programů pro 3D modelování. Tyto programy lze rozdělit podle oblastí využití (strojírenství, elektrotechnika, architektura aj.), dostupnosti (free verze, licencované verze), podle způsobu postupu při modelování (parametrické, neparametrické) a spoustu dalších dělení. **Parametrický CAD** znamená CAD program, který vytváří model postupně obvykle od 2D náčrtu po 3D model s využitím vztahů/omezení (constraints), oproti tomu **neparametrický CAD** (direct modeling) vytváří 3D modely přímo bez vztahů a bez závislé historie kroků.

Pro většinu škol je nejdůležitějším faktorem dostupnost nebo chcete-li cena. Sice není potřeba žáky učit hned v programu, který nám umožní vymodelovat vše, na co si vzpomeneme, nicméně pakliže máme kvalitní software k dispozici a žáci se v něm naučí orientovat a pracovat (stačí základy), mají do budoucna dobrou přípravu a zkušenost, protože na středních/vysokých školách se pracuje především s těmito kvalitními programy. Tím žáci mohou kontinuálně navázat na své zkušenosti ze základní školy. V opačném případě dochází dost často k přeučování již získaných zkušeností a nelogických postupů.

Aktuálně lze získat licencovaný program na ZŠ/SŠ poměrně levně, respektive školy mají možnost čerpat peníze na takové programy přímo z ministerstva v rámci šablon nebo dalších projektů.

3D skenování

Alternativní možností tvorby modelů je 3D skenování. Obsluha ať už ručních či stolních skenerů je však často poměrně náročná a vyžaduje další zpracování modelu ve speciálním software. Stejně tak přímé zpracování fotografií pomocí fotogrammetrického software (např. Meshroom) je poměrně náročné a často vyžaduje podrobné nastavení stylem pokus/omyl. Kombinované zařízení 3D tiskárna + skener s jednoduchým rozhraním (např. XYZprinting da Vinci Pro 3v1) zase dosahuje bez dodatečných úprav modelu převážně velmi špatných výsledků. Pro začlenění do výuky na středních školách tak 3D skenování není v současné době příliš vhodné.

3D tisk

3D tisk je proces, při kterém se z digitální předlohy (3D model) vytváří fyzický model. Jedná se o aditivní proces výroby (Additive Manufacturing). Tisk se provádí po vrstvách.

Použití

3D tisk nachází uplatnění všude tam, kde je zapotřebí výroba prototypů, malovýroba, personalizovaná výroba, výroba jinak nesehnatelných součástí (již se nevyrábí), apod.

Velkou výhodou 3D tisku je snadné sdílení modelů. Pokud tak někdo na jednom konci světa vytvoří model součástky (včetně ověření možnosti jeho tisku), tento model lze snadno předat či sdílet komukoliv na světě, kdo má přístup k odpovídající 3D tiskárně.

Oblasti užití

- Výroba zboží či příprava odlitků zboží (šablony)
- Letectví, kosmonautika, automobilový průmysl
- Zdravotnictví (např. přesné náhrady kostí)
- Stavebnictví
- Umění

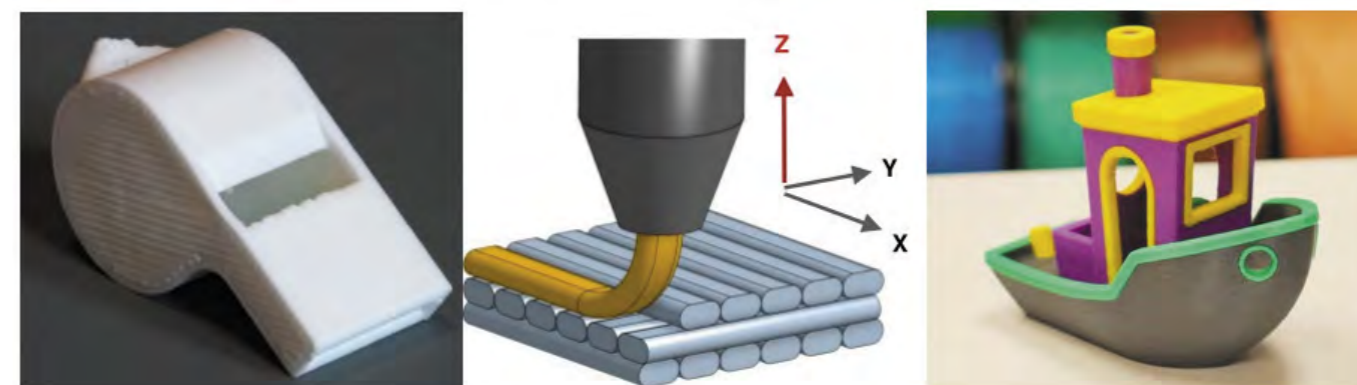
Druhy 3D tisku

Mezi nejrozšířenější formy 3D tisku patří FFF (FDM) SLA (vč. DLP) a SLS (vč. DMLS). Pro použití ve výuce ZŠ/SŠ je nejvhodnější a zároveň ekonomicky nejlevnější variantou 3D tisk FFF. V případě ostatních druhů 3D tisku proces zahrnuje práci s nebezpečnými či přímo toxickými materiály a látkami, jejich zahrnutí např. na technicky zaměřených SŠ je tak nutno pouze za dodržení maximálních bezpečnostních opatření.

FDM (FFF)

Fused Filament Fabrication (Fused Deposition Modeling)

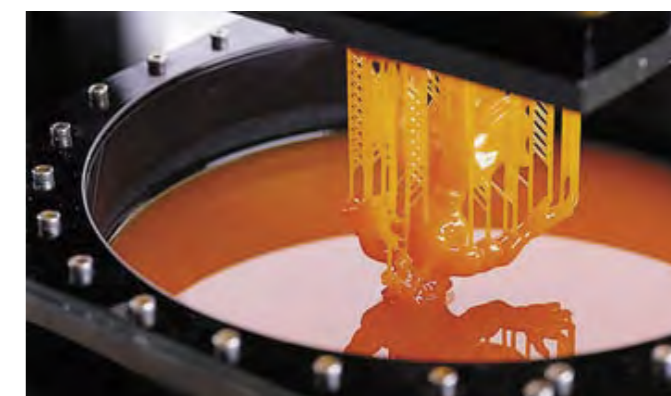
- Princip (elektronicky řízené) „tavné pistole“. Tisk probíhá kladením jednotlivých linek roztaveného plastu, které postupně tvoří vrstvy modelu.
- Nelze tisknout „do vzduchu“, často je tak nutné využít tištěných podpor či jinak tento problém řešit (viz. kapitola *Orientace modelu pro tisk*).
- Materiál = filament (plastová struna podobná té do křovinořezu).
- Některé tiskárny umožňují tisk z více filamentů, resp. jejich střídání. Toho lze využít pro tisk rozpustných podpor, vícebarevného tisku či kombinaci např. flexibilních a pevných materiálů.



SLA / DLP

Stereolithography / Digital Light Processing

- Vytváření objektů pomocí postupného vytvrzování tekuté pryskyřice pomocí působení světla specifické vlnové délky, nejčastěji UV záření či laser. Tisk probíhá postupným osvětlováním jednotlivých vrstev.
- Nelze tisknout do vzduchu, díky možnosti velmi jemného, detailního tisku (proti FFF) jsou však podpory obvykle tvořeny tenkou „stromovou strukturou“. Rovněž dutiny mohou představovat problém (podtlak při tisku, nutnost vypouštěcích otvorů).
- Po dokončení tisku je model nutno dokonale umýt (obvykle v isopropyl-alkoholu) a následně dovytvdřit.
- Materiál = resin (pryskyřice).



SLS / DMLS

Selective Laser Sintering / Direct Metal Laser Sintering

- Tisk probíhá spékáním prachových plastových (SLS) nebo kovových (DMLS) částic laserem po vrstvách.
- Nejsou nutné podpory (přirozená podpora) a rovněž je možný tisk provázaných modelů oddělených nespečeným prachem. Po vytištění je však nutné očištění („vyhrabání“) jednotlivých výtisků.



Další (EBM, Solidscape, c3Dp/3DCP)

Selective Laser Sintering / Direc Metal Laser Sintering

- Existuje celá řada dalších specifických druhů 3D tisku, většina z nich však nemá obecné uplatnění v ZŠ/SŠ vzdělávání. Pro průmyslové střední školy může být nicméně zajímavá např. existence 3DCP
 - 3D tisk domů obvykle z betonu (princip podobný FFF).

Druhy filamentu

Mezi nejběžnější používané filamenty (tiskové struny pro FFF) patří:

PLA (Polylactic Acid)

Asi nejrozšířenější filament u nás, který je biologicky plně odbouratelný. Vyrábí se z cukrové třtiny, bramborového škrobu či kaučuku. Vyniká prakticky nulovou teplotní roztažností, možností tisku velmi nízkých vrstev (až 0,05 mm) -> vysoké detaily a dobrými mechanickými vlastnostmi. Teplota tisku je obvykle 185–230 °C. Hlavními nevýhodami je nízká teplotní odolnost a nízká odolnost vůči chemikáliím i UV záření.

PET (Polyethylentereftalát)

Je to filament, který propojuje nejlepší vlastnosti filamentů ABS a PLA (snadný tisk a dobrá odolnost jak mechanická, tak teplotní, chemická i UV). Zároveň má velmi malou teplotní deformaci. Velmi výhodný pro FDM technologie tisku a nezávadný (tzv. food-safe). Teplota tisku je obv. 220–250 °C.

ABS (Akrylonitrilbutadienstyren)

Velmi rozšířený druh plastu a první, který se pro 3D tisk FFF začal používat. Jedná se o mechanicky odolný a přitom dobře opracovatelný materiál s vysokou teplotní odolností. Nevýhodou je poměrně velká teplotní roztažnost a zbytkové zplodiny při zahřívání na vysokou teplotu (malá koncentrace, ale doporučeno při tisku dostatečně větrat). Obvyklá teplota pro tisk je 230–260 °C.

Seznam dalších rozšířených druhů filamentů:

- | | |
|--|--|
| – CPE (Co-Polyester) | – ASA (Acrylonitrile styrene acrylate) |
| – PC (Polykarbonát) | – PMMA (Polymethylmethakrylát) |
| – PP (Polypropylen) | – HIPS (High Impact Polystyrene) |
| – Nylon/PA (Nylon (Polyadmidy)) | – PVA (Polyvinylalkohol) |
| – TPE/TPU (Thermoplastic elastomers/
polyurethanes) | – BVOH (Butendiol-Vinylalkohol) |

Filamenty mohou být doplněny o různá aditiva (např. různé třpytky, dřevěné či dokonce kovové částice, karbonové vlákno, apod.). Pozor, některé filamenty jsou abrazivní a mohou tak vyžadovat speciální trysku!

Pro využití ve školním prostředí jsou nejvhodnější filamenty z PET (případně CPE) a PLA, ze kterých lze snadno tisknout na všech FFF tiskárnách a jsou zdravotně nezávadné. Tisk z ABS je možný za dostatečného větrání (což lze doporučit obecně, pouze pozor na případný průvan, který může způsobit komplikace při tisku u otevřených tiskáren). Další materiály jsou na tisk již náročnější s ohledem na tiskové parametry, vlastnosti a přípravu tiskové podložky. Zejména tisk z flexibilních filamentů (Nylon, TPE, apod.) může být problematický, především u tiskáren s bowdenovým extruderem (viz. kapitola Vlastnosti 3D tiskáren FFF). Pokud je škola vybavena multimateriálovou tiskárnou, je vhodné disponovat také filamenty rozpustnými ve vodě pro tisk vodou rozpustných podpor (PVA/BVOH).

Příklady z praxe

Jinak, než 3D tiskem to někdy nejde

Mezi velmi časté použití 3D tiskárny patří personalizovaná výroba, tedy výroba něčeho, co je přizpůsobeno na míru přání autora. Studenti velmi často mají mnoho nápadů na něco speciálního, co by si rádi vytvořili nebo přizpůsobili svým potřebám. Jednoho z kurzů se například zúčastnil také student na kolečkovém křesle, který si vymodeloval a vytiskl držák na nápoje přesně na svůj vozík.

Častým případem pro využití 3D tisku je také tisk součástek/náhradních dílů, které již nejsou jinak k dostání, nebo jejich zakázková výroba by byla příliš drahá. Jeden ze studentů kurzu například vymodeloval náhradní držák struhadel do mixéru pro maminku, která by jinak musela tento mixér již vyhodit.

Příklad z výuky na ZŠ

Dva žáci již měli zkušenosti s 3D modelováním, nicméně v neparametrických programech, což způsobovalo velké problémy nejen při přeorientování na parametrický CAD, ale také v možnostech modelování. Oba chlapci byli do modelování velmi nadšení, ale jak sami přiznali, byli limitováni možnostmi neparametrického Tinkercadu.

Díky parametrickému SolidWorksu si i přes počáteční problémy, spojené s přeorientováním, tento program oblíbili a prakticky již zvládají modelovat i poměrně složité konstrukce, které jsou vhodné spíše pro vyšší ročníky středních škol.

Absolvováním této výuky žáci/studenti získají nebo si prohloubí základní návyky v rámci 3D modelování. Dále se dozví informace ze světa 3D tisku, možností volby programů a základů technické dokumentace. Zde záleží pouze na učiteli, který musí odhadnout úroveň svých žáků a jak hluboko si může dovolit ponořit se do dané problematiky.

V neposlední řadě lze tímto úkolem rozvíjet technické znalosti žáků – již zmiňované základy technické dokumentace, prostorovou představivost, technickou představivost aj. S úspěchem lze na závěr zadat žákům projekt, kde si každý dle svých schopností a představivosti může vytvořit vlastní model, který si i sami vytisknou (naučí se obsluhovat 3D tiskárnu). Tyto vytisknuté modely mohou sloužit i pro propagaci školy v rámci dnu otevřených dveří aj.



Metodická a didaktická část

Úvod – první hodina

V rámci první hodiny by žáci na začátku měli být především seznámeni s bezpečností práce, aby se předešlo možnému zranění žáků nebo poškození tiskárny a příslušenství. Dále by se žáci měli ve zkratce seznámit s tím, co je 3D tiskárna a provést první krátký tisk (předem připravený GCODE). Následně je již možné se pustit do základů modelování.

Bezpečnost práce

V rámci kurzu by měli žáci/studenti dodržovat základní bezpečnost práce v IT nebo jiné učebně, která je dána řádem učebny. Dále by měli být hned v úvodu seznámeni s bezpečností práce a správným užívání 3D tiskárny a příslušenství. Žáky je zejména potřeba upozornit na:

- Práci s elektrickým zařízením (nebezpečí při polížení, ventilační otvory zdroje a elektroniky, atd.).
- Nářadí s ostrými hranami (nože, špachtle, apod.). **Zejména zranění ostrou špachtlí při sundávání po tisku patří mezi nejčastější úrazy související s 3D tiskem.**
- Části zařízení s vysokou teplotou (tryska, vyhřívaná podložka, krokové motory)
- Mechanické pohyblivé části – hrozí přiskřípnutí.
- Chemikálie (aceton, IPA, líh, lepidla, atd.).

Při práci s 3D tiskárnou by měli žáci vždy dodržovat:

- **Se zařízením pracovat pouze na pokyn učitele** a vždy dodržovat jeho pokyny. Pokud budou mít žáci s tiskem jakýkoliv problém, měli by se ihned obrátit na vyučujícího.
- Nekonzumovat jídlo a nemanipulovat s tekutinami v blízkosti tiskáren.
- Pozor na statický náboj (zejména displeje některých tiskáren jsou citlivé na statický náboj – může dojít k chvilkovému rozrušení displeje nebo v extrémním případě i k poškození).
- Vyvarovat se poškození tiskového povrchu (tryskou, špachtlí, výměna tiskového plátu, apod.).
- **Nevypínat tiskárnu, dokud tryska nevychladne pod 50°C.** Při závažnější chybě tisku mají uživatelé 3D tiskárny tendenci tiskárnu vypnout, to však vede k vypnutí ventilátoru a nárůstu teploty nad tryskou, což může vést k jejímu uspání či poškození.
- V učebně dostatečně větrejte a zbytečně neinhalyte zbytkové látky při tisku (ani v případě bezpečných materiálů). Pozor ale na průvan, který by mohl ztížit tiskové podmínky.
- Nenechávejte nikdy tisk bez dozoru.

Žáci by se měli rovněž seznámit s manuálem a bezpečnostními pokyny ke konkrétní tiskárně.

Seznámení s tiskárnou a první tisk

Žáci/studenti by si měli ve skupině vyzkoušet kontrolu tiskárny a přípravu pro první tisk. Na začátku je potřeba se seznámit se základním fungováním tiskárny (osy, tryska, tisková plocha/podložka) a práci s nimi. Rovněž je potřeba se seznámit se základním ovládním (ovládací prvky, vypínač, reset tlačítko – jeli k dispozici).

Pokud je k dispozici více tiskáren, ideálně by měl učitel vysvětlit a ukázat postupně očištění/odmaštění tiskové plochy, odejmutí a usazení tiskového plátu (je-li jím tiskárna vybavena), přehřev a zavedení filamentu. Tento postup mohou studenti pod dozorem replikovat na dalších tiskárnách. Důležité je umožnit studentům opakovat dílčí kroky a vždy počkat na dokončení, aby se zamezilo případným chybám.

Následně je možné spustit první tisk (z předem připraveného GCODE). Vhodné jsou například vzorové modely od PrusaResearch (dostupné s instalací Prusa software nebo na <https://www.prusa3d.cz/3d-modely-pro-tisk/>). S ohledem na čas jsou vhodné např. píšťalka, Marvin či otvírák na lahve.

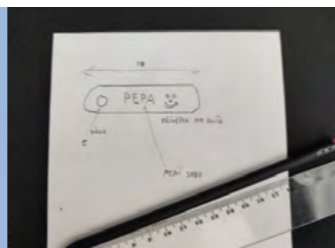


Od modelu k tisku (jak postupovat)

Ještě, než začneme s žáky/studenty vytvářet modely, musíme vysvětlit základní postup této tvorby a následných kroků pro jejich vytištění. Dílčí kroky je možno shrnout do několika bodů:

2D Náčrt

Náčrt na papír, který pomůže ucelit představu o finálním produktu. Později může být využit jako základ pro 2D skicu v CAD.



Představa

Již pro náčrt je vhodné mít k dispozici pravítko, úhloměr, apod. pro snazší odhad velikostí.

Tvorba 3D modelu

Pomocí 2D náčrtů a jejich „vytažením“ do 3D postupně vytváříme model.



STL/OBJ

Export modelu a načtení do sliceru.

Slicing

Příprava modelu pro tisk na konkrétní tiskárně a z konkrétního materiálu. Nastavení parametrů tisku (rychlost, perimetry, výplň, atd.)



GCODE

Přenos do tiskárny.

Tisk

Příprava materiálu, tiskové plochy, tisk.



Tištěný díl

U modelu jsme zapomněli zaoblit hrany, můžeme je tak např. zabrousit.

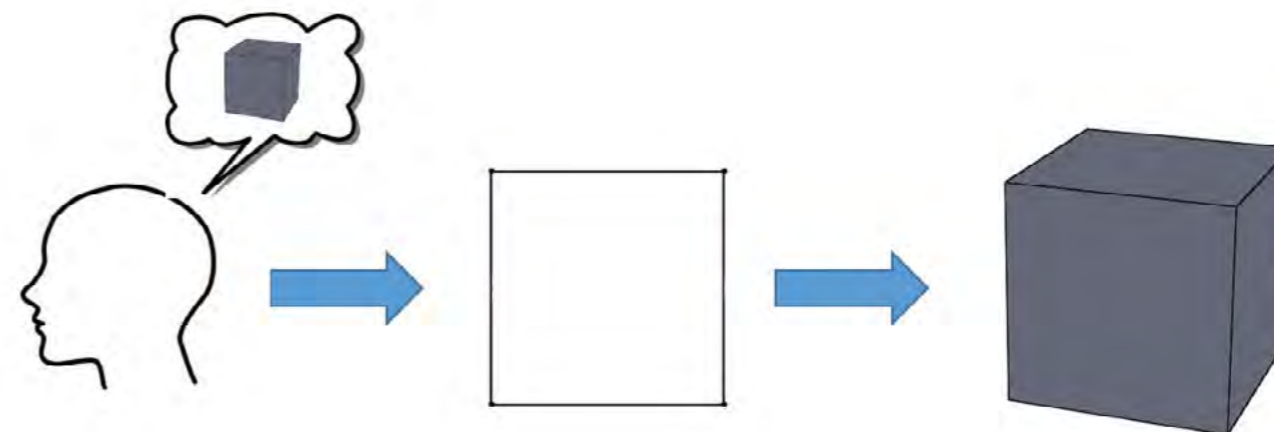
Postprocesing (následné zpracování)

Očištění, lepení, broušení, spojování, atd.



3D modelování

Je to proces tvorby výsledného modelu (od myšlenky, přes náčrt až po hotový model), který má několik fází. Tyto fáze mají jasnou posloupnost, kterou nelze měnit.



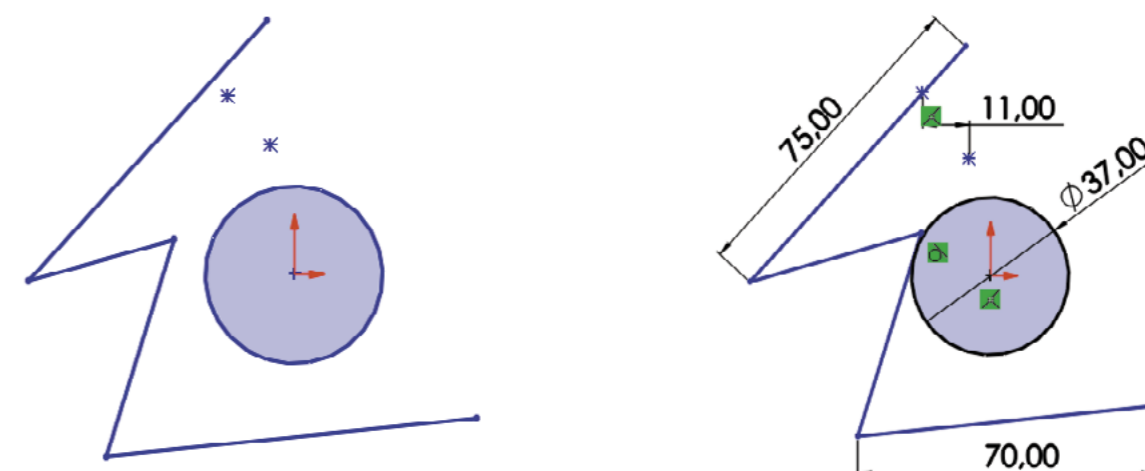
2D náčrt

Slouží k načrtnutí, kótování a zavazbení základního nákresu 2D zobrazení budoucího modelu.

V první fázi bychom měli seznámit žáky/studenty s prostředím programu, ve kterém budeme pracovat. Žáci /studenti by si měli osvojit vlastnosti a funkce prostředí. V parametrických CAD programech vytváříme napřed 2D náčrt. Ten se obvykle vytváří na skicu.

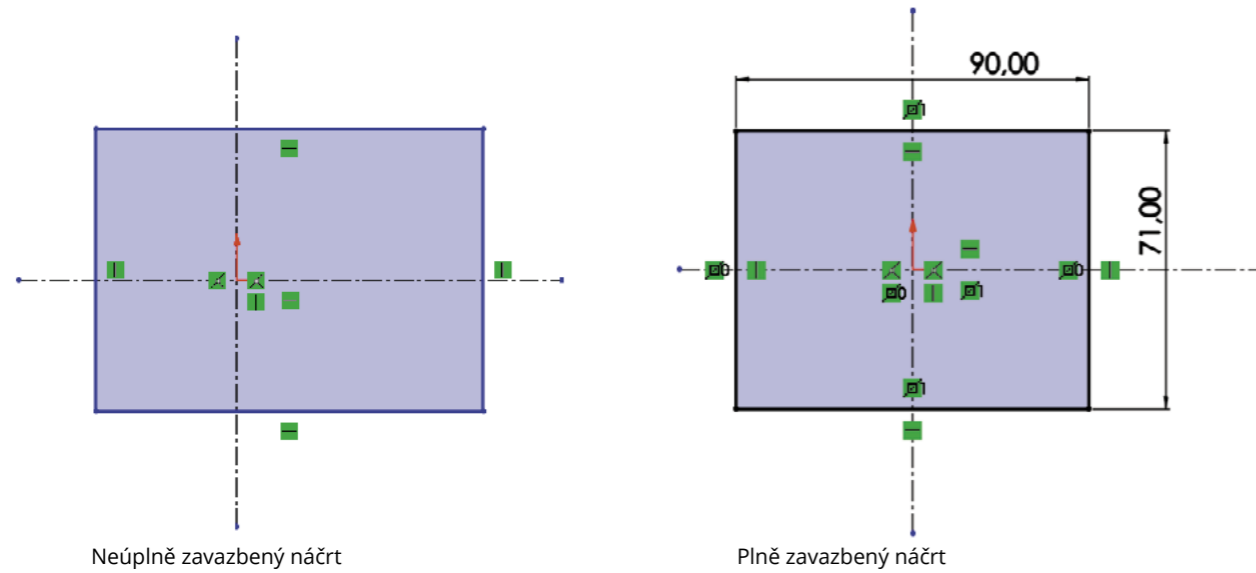
Skica

Skica je "papír", na který lze zakreslit, pomocí jednotlivých nástrojů, různé čáry, tvary, body atp. Tyto kresby můžeme zakótovat a zavazbit, případně pomocí některého nástroje zkopírovat, zrcadlit, ořezat, spojit atp. Po dokončení náčrtu skicu jednoduše uzavřeme (můžeme se do ní kdykoli během práce vrátit).



Plně určený náčrt

Po nakreslení náčrtu je vždy dobré tento náčrt tzv. plně určit. To můžeme učinit pomocí vazeb (tečná, sjednoceno, kolmá, ...) nebo kót (rozměrů). Zabráníme tak možné deformaci náčrtu a zároveň si ověříme, že daný náčrt má všechny potřebné parametry k převodu na 3D model. V rámci kótování a vazbení můžeme velmi dobře využívat osy, které nám práci ulehčí.

**3D model**

Slouží k převedení dvojrozměrného nákresu do trojrozměrné podoby, kde je možné získaný model dále upravovat. Po dokončení 2D náčrtu uzavřeme skicu a můžeme využít některou z funkcí na vytvoření 3D modelu. Funkci vybíráme podle vhodnosti k našemu náčrtu. Máme několik základních funkcí:

- Vysunutí
- Rotace
- Spojení profilů
- Tažení po křivce

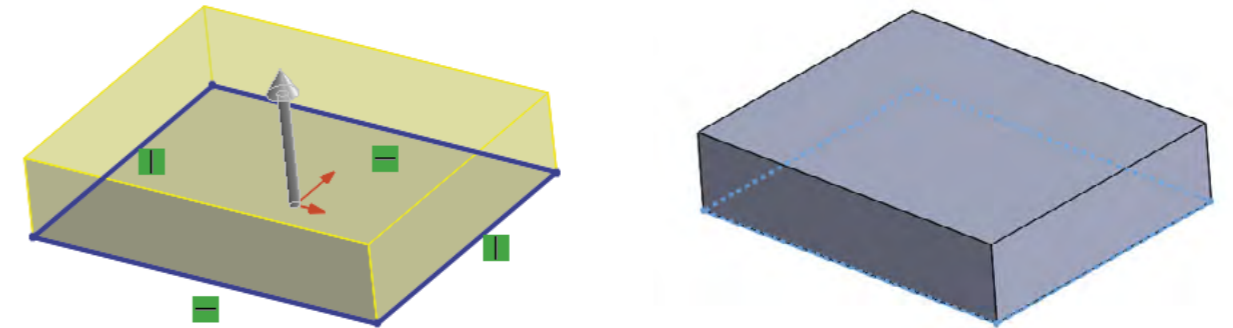
Dále zde můžeme již hotový model upravovat pomocí dalších nástrojů k tomu určených, nicméně je téměř vždy lepší udělat všechny úpravy, pokud je to možné, již v 2D náčrtu.

Vysunutí

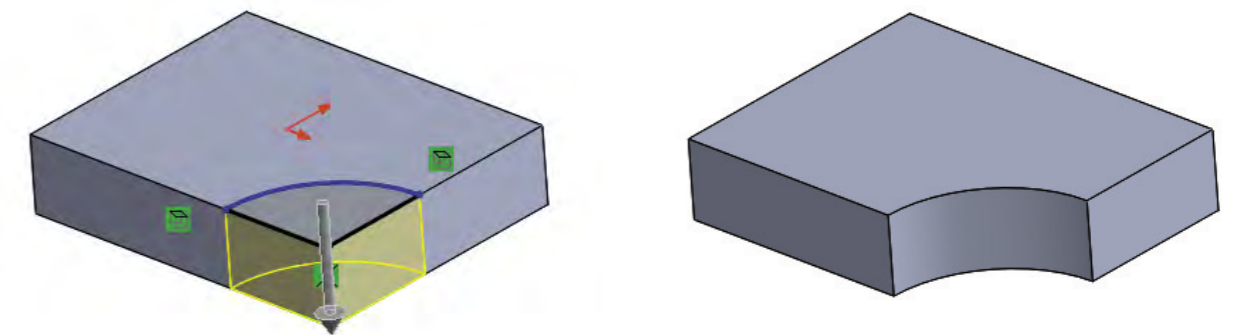
Tato funkce vysouvá námi vybraný 2D náčrt, uzavřenou část tohoto náčrtu nebo jiný uzavřený tvar. Lze zvolit směr, tvar a délku vysunutí. Jedná se o nejjednodušší a též nejpoužívanější funkci. Má dvě možnosti použití, a to Přidání vysunutím a Odebrání vysunutím.

Přidání vysunutím

Vysouvá materiál dle zadané délky a tvaru daným směrem a tím vytváří z 2D náčrtu prostorový 3D model.

**Odebrání vysunutím**

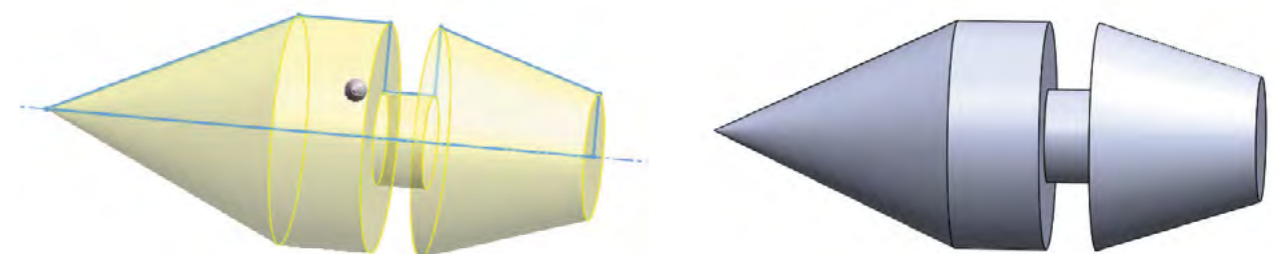
Vysouvá (odebírá) materiál dle zadané délky a tvaru daným směrem a tím vytváří do již hotového 3D modelu různé otvory, či ho jinak tvaruje.

**Rotace**

Tato funkce potřebuje ke správnému fungování jednu z os, kolem které rotuje námi nakreslený 2D náčrt a tím ho převádí na 3D model. Je zde zapotřebí prostorové představivosti, neboť na první pohled nemusíme mít 2D náčrt správně, či se nám může jen zdát, že ho nemáme správně. Jedná se o druhou nejčastěji používanou funkci. I v tomto případě máme dvě možnosti použití - Přidání rotací a Odebrání rotací.

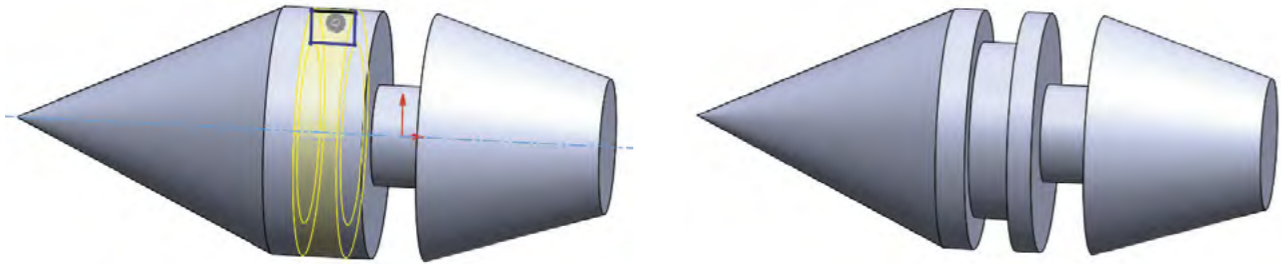
Přidání rotací

Pomocí osy rotujeme nakreslený náčrt a ten nám vytvoří model. Náčrt musí být uzavřený k ose a zároveň poslední část kopíruje osu, jinak nám program bude hlásit chybu. Můžeme určit úhel rotace - zda se součást rotuje kolem osy dokola či pouze pod námi zadaným úhlem.



Odebrání rotací

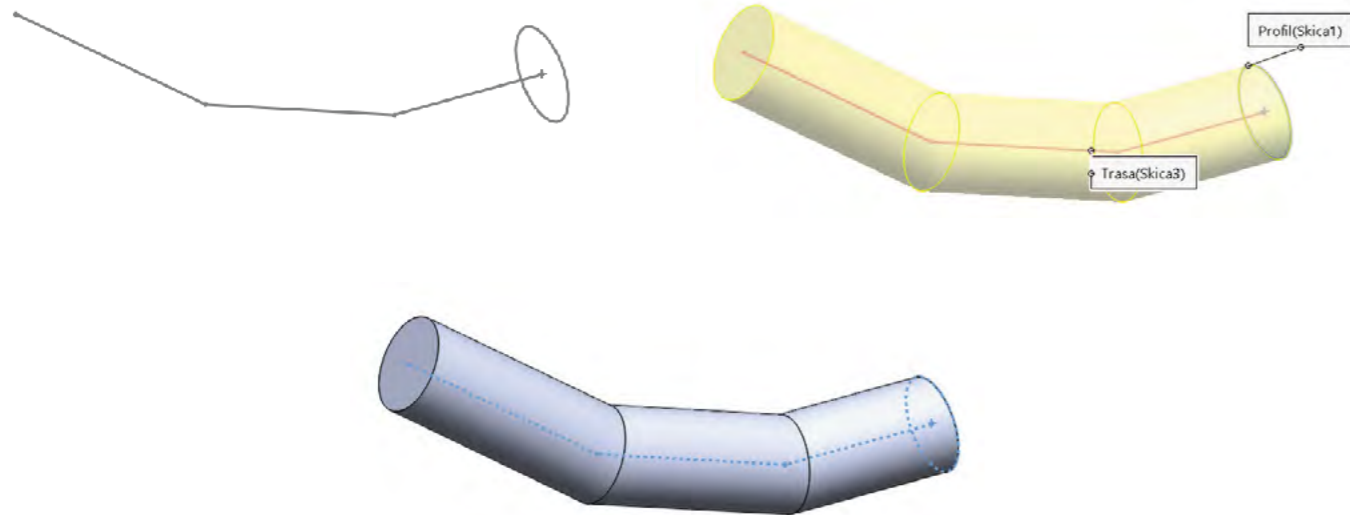
Pomocí osy rotujeme nově vytvořený náčrt "ve" vytvořeném modelu. Zde nemusíme řešit uzavření náčrtu s osou, nicméně náčrt musí být uzavřený. Většinou se tato funkce využívá pro úpravy již hotových polotovarů modelu.

**Tažení po křivce**

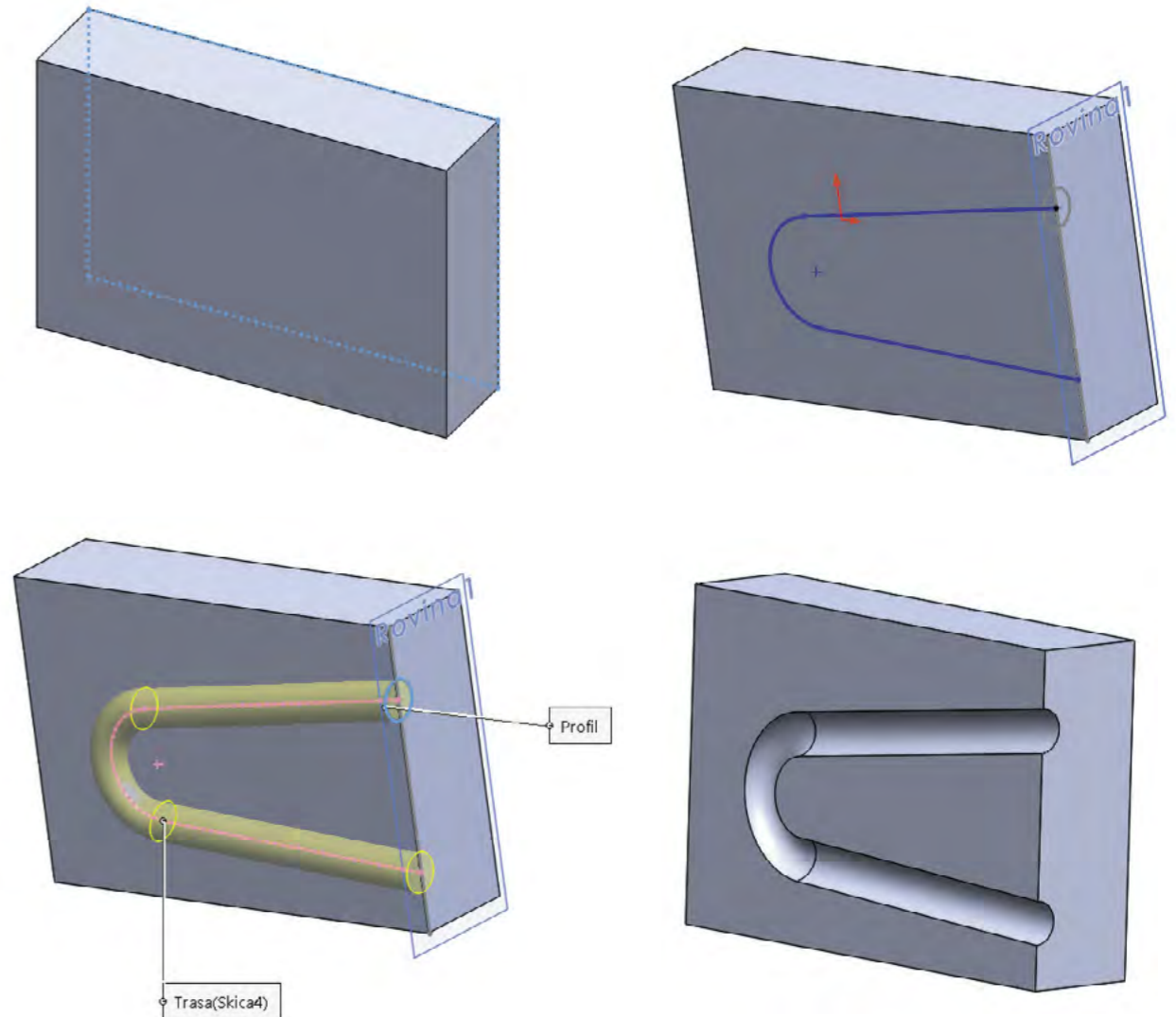
Tato funkce již patří k těm pokročilejším. Jak již napovídá název, dochází při použití této funkce k "tažení" určitého tvaru (například kružnice) po dané křivce (přímka, oblouk atd.) čímž vzniká požadovaný model (potrubí, drátu atp.). Pro použití této funkce potřebujeme dvě roviny, ve kterých na skici nakreslíme tvar a křivku. I zde můžeme přidávat a odebírat.

Přidání tažením po křivce

Pro jakýkoliv náčrt obvykle používáme přední rovinu, ne jinak tomu bude i v případě tvaru (kružnice), po dokončení a uzavření skici si zvolíme rovinu kolmou na přední rovinu, a to pravou rovinu. Na této rovině si otevřeme skicu a nakreslíme pomocí přímek křivku. Po ukončení a uzavření skici použijeme funkci Přidání tažením po křivce, kdy volíme, co je tvar a co je křivka.

**Odebrání tažením po křivce**

Používá se především, pokud chceme z již hotového modelu částečně odebrat určitý složitější tvar. Opět budeme potřebovat dvě na sebe kolmé roviny se skicami, navíc budeme potřebovat pravou rovinu posunout na okraj původního modelu, což uděláme přes příkaz Rovina a určíme vzdálenost a pak na ní otevřeme skicu. Na této skice vytvoříme tvar, který budeme táhnout. Poté si vytvoříme skicu na již hotové ploše modelu a nakreslíme křivku. Po ukončení i druhé skici stačí jen použít funkci Odebrat tažením po křivce a zvolit, co bude tvar a co křivka.

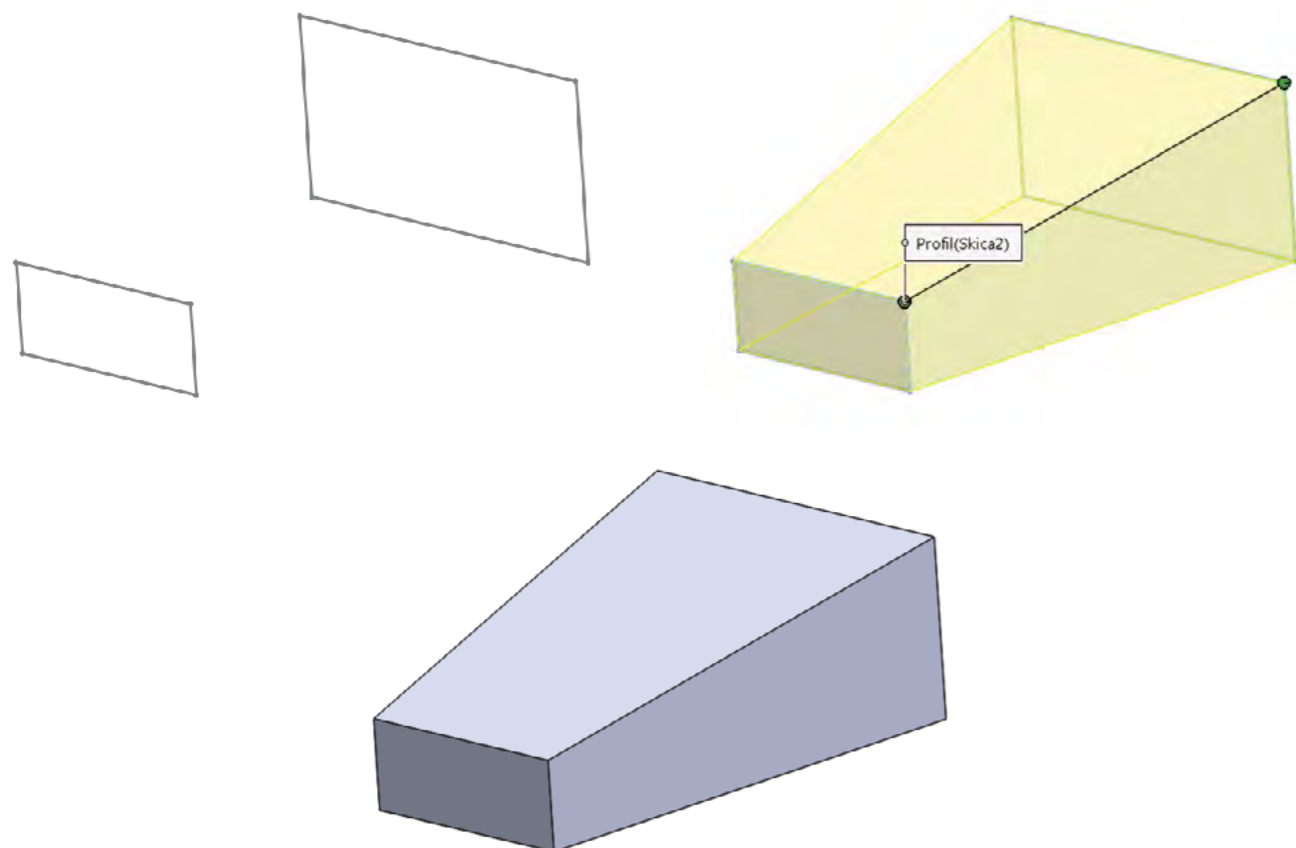


Spojení profilů

I zde se jedná o pokročilejší funkci. Jak napovídá název, jedná se o spojení dvou již vytvořených profilů (náčrtů). Tyto náčrty leží každá na jiné skice a jsou od sebe obvykle vzdáleny, opět si zde můžeme pomoci Rovinou. Máme zde možnost přidávat nebo odebírat.

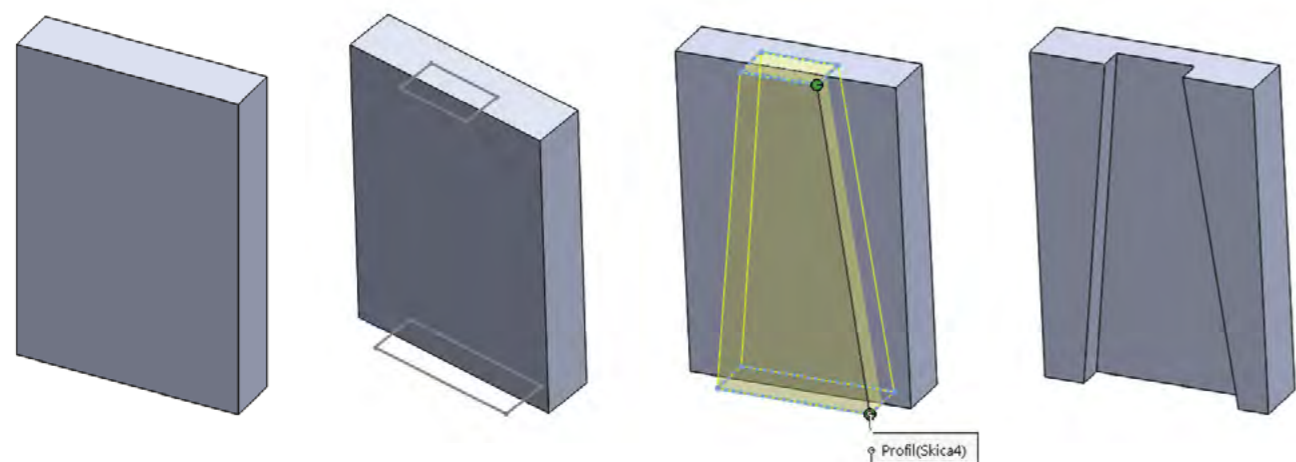
Přidání spojením profilů

Jedná se o vytváření 3D modelu z dvou nebo více náčrtů. Je doporučeno jednotlivé náčrty umísťovat od sebe ve správných a promyšlených rozměrech, aby nedošlo k chybě (spojení by se někde protínalo).



Odebrání spojením profilů

Podobně jako u odebrání tažením, i zde můžeme odebírat část materiálu již hotového modelu. Opět si musíme udělat dva nebo více náčrtů na různé roviny (využijeme nástroj Rovina). Po dokončení a uzavření všech skic použijeme funkci Odebrání spojením profilů a máme hotovo.



Zákonitosti tvorby v 3D programu

Každý konstruktér má svůj léty prověřený způsob konstruování v 3D programech, proto je těžké popsat naprosto obecný a všude aplikovatelný postup, kterým vždy dojdeme kýženého výsledku. Přesto jsou některé zákonitosti, které se vyplatí dodržovat a ulehčit si tak práci. Tyto zákonitosti lze uplatnit v průběhu celé tvorby finálního modelu, nicméně pro přehlednost si je rozdělíme do třech základních kategorií:

1. Obecné
2. 2D náčrt
3. 3D model

Obecné

Zásady, které se vyplatí dodržovat před zahájením samotného modelování, případně jsou dosti obecné a dají se realizovat v kterékoli fázi modelování.

- Rozmyslet si výsledný model a postup, který k němu povede. Často pomůže hrubý náčrt.
- Promyslet, změřit případně propočítat všechny rozměry.
- Zamyslet se nad vztahy výsledného objektu k jeho okolí (např. uchycení apod.).
- Vhodně zvolit orientaci, rozdělení na díly a postup při modelování.
- Bude výsledný model možné vytisknout (v závislosti na použité technologii)?
- Jaké budou požadavky na mechanické vlastnosti a materiál pro výsledný produkt?
- Model je vhodné průběžně verzovat a jednotlivé kroky popisovat (pokud to software umožňuje).

2D náčrt

Zásady, které se vyplatí dodržovat při tvorbě 2D náčrtku:

- Základní náčrt se snažím kreslit co nejpodobnější výslednému plně určenému náčrtu.
- Vždy končit plně určeným náčrtem.
- Pokud existuje přirozený vztah, upřednostňovat zavazbení před dodatečnými kótami (např. pokud má být kružnice ve středu obdélníku, jeho střed by měl být určen zavazbením na středy stran obdélníku, nikoliv kótami o hodnotě poloviny strany). Často lze vhodně použít pomocné tvary.
- Jednu skicu je možné využít pro více 3D operací.

3D model

Zásady, které se vyplatí dodržovat při tvorbě 3D modelu:

- Volit správné funkce (vysunutí, rotace...) pro převod 2D náčrtu na 3D model.
- Při práci s funkcemi dbát na správný výběr ze skici.
- Pokud existuje přirozený vztah, upřednostňovat tento před použitím fixních hodnot (např. pokud má být vysouvaná část vždy o určitou část menší než vedlejší dříve vysunutá část, mělo by být použito vysunutí k hraně + offset).
- Nebát se pokročilých funkcí (např. duplikování součástí či jednotlivých funkcí).
- Tolerance, zaoblení a zkosení aplikujeme ideálně až nakonec.

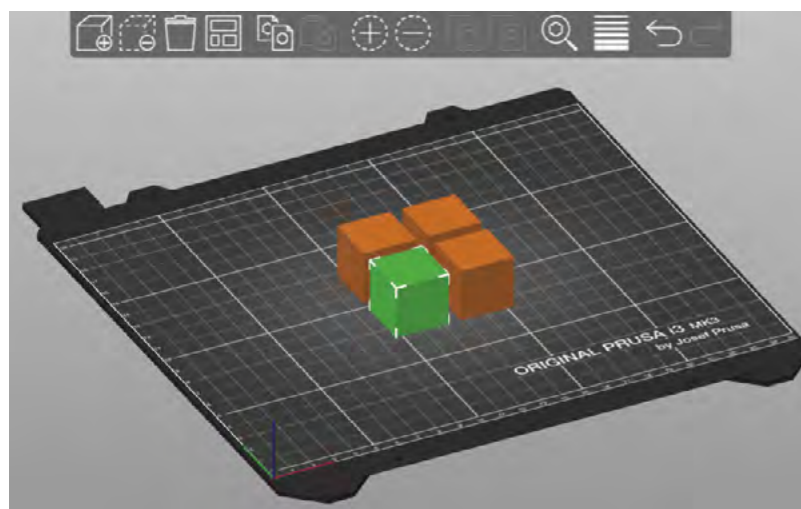
3D tisk

Slicing – úprava a příprava již hotového modelu na samotný tisk.

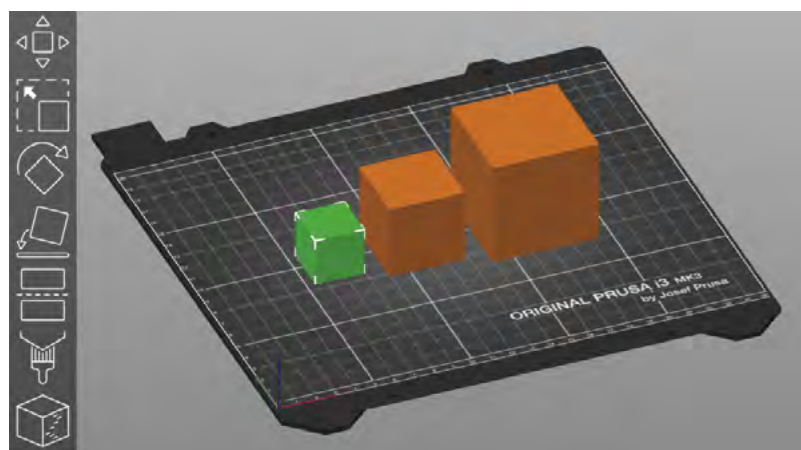
Slicem je nazýván proces převodu 3D objektu na strojový kód (pokyny, kterým rozumí 3D tiskárna – u FFF nejčastěji GCODE). Slicer (obecný název pro software provádějící slicing) převede model na jednotlivé tahy 3D tiskárny a současně GCODE obsahuje veškerá další nastavení (teploty, rychlost, výška vrstvy, apod.).

Volba vhodného sliceru je obvykle dána podporou konkrétní tiskárny (tvorba vlastního profilu tiskárny pro slicer je značně náročná). Pro rozšířenější tiskárny však často existují profily (konfigurace) pro různé slicery. Mezi nejznámější patří Slic3r, Cura, Simplify3D). Některé tiskárny mají slicer jako cloudovou službu a vyžadují tak připojení k internetu. Níže uvedené příklady byly vytvořeny v PrusaSliceru (odvozená verze Slic3ru), který podporuje celou řadu 3D tiskáren i celou řadu funkcí. Výhodou pro výuku je možnost přepnutí mezi režimy jednoduchý/pokročilý/expert a podpora mnoha jazyků včetně češtiny.

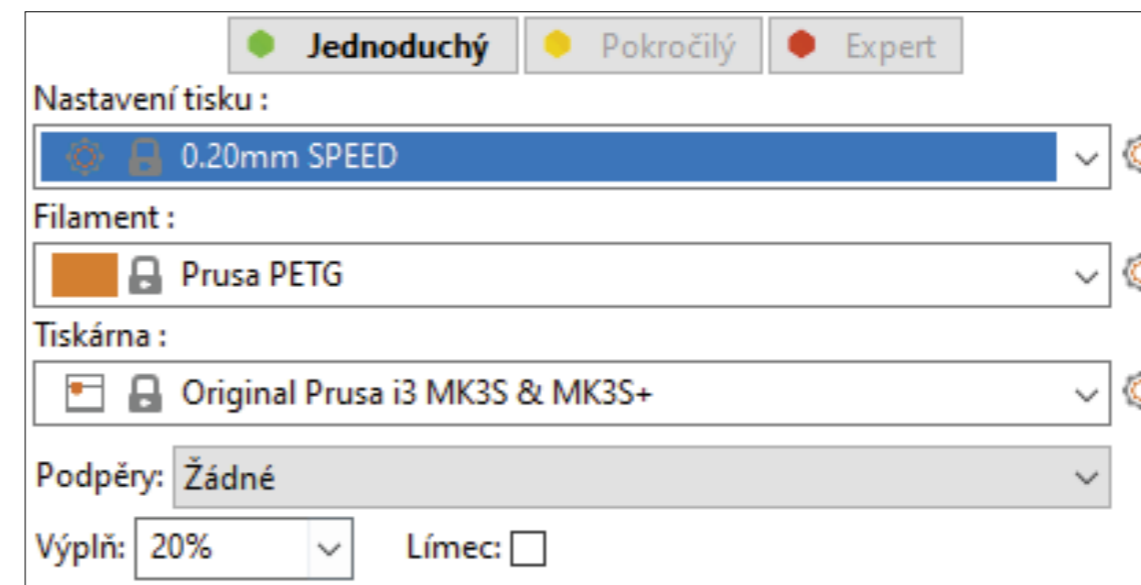
Nejdříve je potřeba do sliceru načíst modely – objekty pro tisk (obvykle soubory ve formátu STL nebo OBJ). Objekt můžeme tisknout i vícekrát (více instancí) nebo můžeme tisknout více různých objektů. Zde nám může velmi pomoci možnost automatického uspořádání objektů na tiskové ploše.



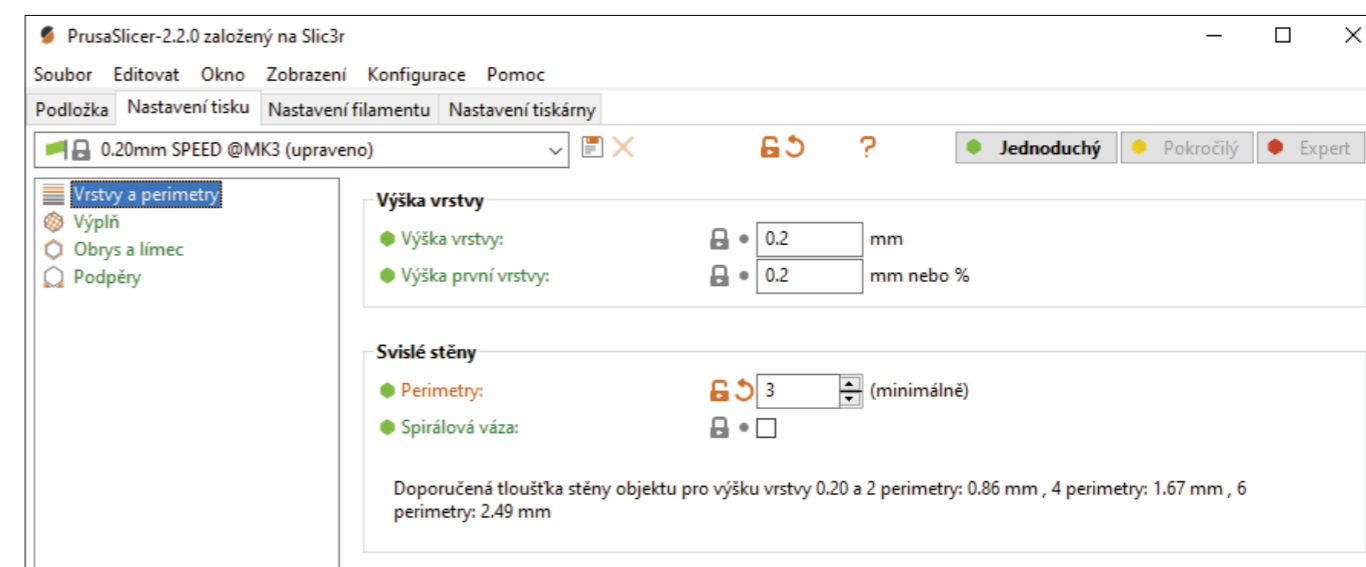
Objekty můžeme různě přemísťovat, měnit velikost, otáčet apod. Důležitá je zejména orientace tištěného dílu, především pak to, kterou plochou hrana dosedá na tiskovou plochu (první vrstva tisku). Více o orientaci v kapitole *Orientace objektu pro tisk*.



Základním nastavením pro každý tisk je volba tiskárny a filamentu. Dále je potřeba zvolit minimálně základní profil pro tiskové nastavení (obsahující především údaje o výšce vrstvy, počtu perimetrů a výplni). Volba filamentu určuje zejména teplotu trysky a tiskové podložky.



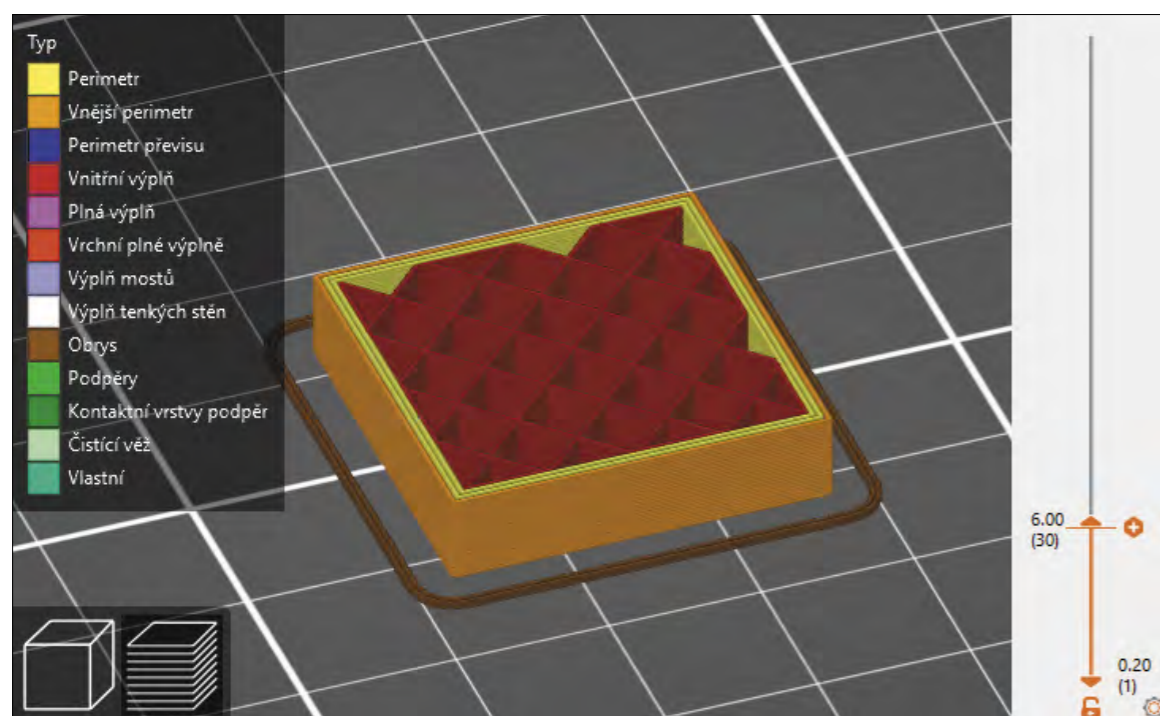
Tyto 3 typy nastavení můžeme dále přesněji konfigurovat na jednotlivých záložkách, kde máme k dispozici podrobné konfigurace jednotlivých profilů (v závislosti na režimu: Jednoduchý/Pokročilý/Expert).



Kliknutím na tlačítko slicovat proběhne generování GCODE a náhledu samotného tisku (vrstev). Mezi režimy práce s objekty a tiskovým náhledem je možné kdykoliv přepnout pomocí tlačítek „kostky“ v levém spodním rohu.



V režimu náhledu lze pomocí posuvníku procházet vrstvy, což je výhodné jak pro náhled dovnitř objektu (výplň), tak pro kontrolu možných potencionálních problémů (především tisku „do vzduchu“). Kliknutím na malé „+“ na posuvníku můžeme také přidat kód pro výměnu filamentu (např. pro změnu barvy) či pozastavení tisku.



Přehled nejdůležitějších nastavení

Nastavení tiskárny

- Průměr trysky *Nutno změnit v případě výměny trysky. Průměr trysky má však vliv na další vlastnosti při tisku, ideální je tak použít nějaký z již připravených profilů.*

Nastavení filamentu

- Filament
 - Teplota *Nastavení teploty by mělo vždy odpovídat doporučením výrobce konkrétního filamentu. Někdy je však nutno trochu experimentovat. Např. vyšší teploty vedou obvykle k pevnějšímu spojení vrstev, ale naopak při tisku převisů a mostů může dojít k větším defektům.*
 - Násobič extruze *Ovlivňuje množství vytlačovaného filamentu. Někdy může být žádoucí toto množství zvýšit (zejména pokud má filament menší průměr než je obvyklý).*
 - Chlazení *Nastavení chlazení (tiskového ventilátoru) je důležité pro správný tisk většiny filamentů. Pomáhá předcházet nežádoucím efektům při tisku. U některých filamentů s větší tepelnou roztažností však může způsobovat deformace a odlepování již v průběhu tisku.*

Nastavení tisku

- Vrstvy a perimetry
 - Perimetry *Počet obvodových vrstev zásadně ovlivňuje pevnost výsledného dílu. Pro výslednou pevnost je obvykle důležitější počet perimetrů než hustota výplně.*
 - Detekovat perimetry přemostění (pokročilý režim) *Mění parametry pro tisk mostů a převisů (obvykle zlepšuje jejich tisk – zmírňuje možné defekty).*
 - Vyhnout se přejíždění perimetrů (expertní režim) *Zamezí pohybu trysky mimo tištěné objekty (přejíždění), mimo nezbytných. Používá se především k zamezení vzniku strunek (tenkých vláček mezi částmi objektu).*
- Výplň *Lze nastavit různé vzory výplně pro obsah a pro horní a spodní vrstvu a její hustotu. Nastavení výplně zásadně ovlivňuje mechanické vlastnosti výsledného tištěného dílu. Výplně slouží také k podpoře horních vrstev a zásadně ovlivňuje délku tisku, vhodné nastavení je tak velmi důležité.*
- Obrys a límec
 - Šířka límce *Límec slouží k rozšíření základny objektu u první vrstvy. Napomáhá dobrému přilnutí objektu k podložce zejména u objektů s malou základnou (kontaktní plochou) a předchází odlepování rohů. Obvykle se používá nastavení 2–5 mm.*
- Podpěry
 - Generovat podpěry *Zapnutí podpěr*
 - Automaticky generované podpěry *Slicer automaticky přidá podpěry tam, kde to považuje za nutné (v pokročilých režimech možno upřesnit).*
 - Pouze na tiskové podložce *Budou použity pouze podpěry, které začínají na tiskové podložce, nikoliv na samotném modelu.*

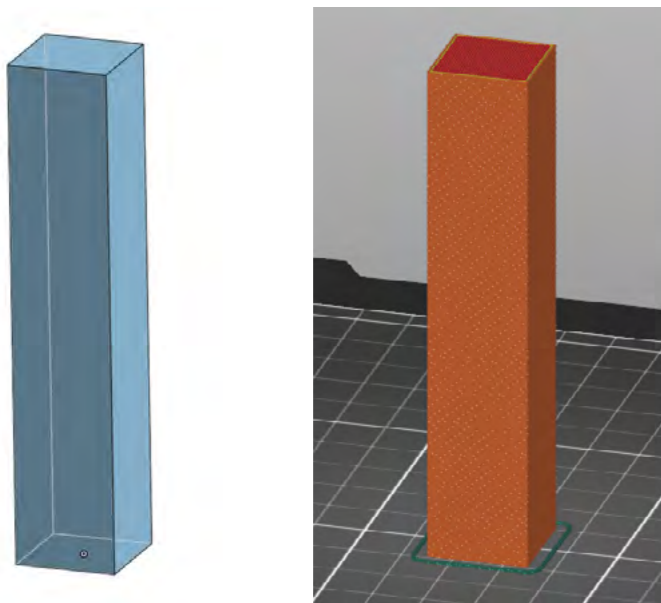
PrusaSlicer obsahuje celou řadu pokročilých funkcí (modifikátory, „malované“ podpěry, variabilní výšku vrstvy, vícemateriálový tisk, vlastní GCODE, apod.) a rovněž podporuje SLA tisk. Přestože řada z těchto funkcí je velmi užitečná, jejich zvládnutí je časově náročné a vyžaduje velmi dobrou znalost základních nastavení pro slicing a často zkušeností s tiskem.

Orientace objektu pro tisk

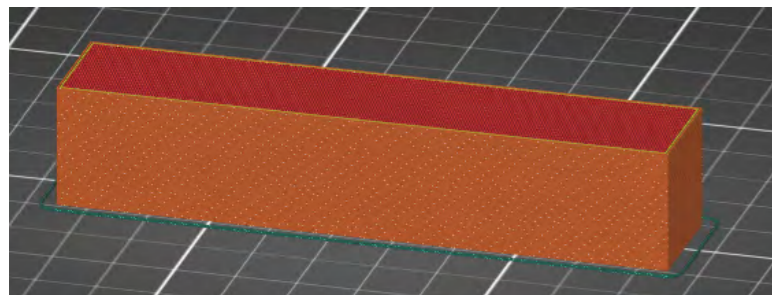
Orientace objektu pro 3D tisk zásadně ovlivňuje vlastnosti výsledného objektu. Asi nejzásadnější vlastností je, zda jsou pro tisk nutné podpory. Orientace objektu ovlivňuje především tyto faktory:

- Nejslabší vazby v objektu bývají mezi vrstvami. Pokud objekt „praskne“ v důsledku mechanického namáhání, stane se tak obvykle mezi vrstvami.
- Převisy je možné obvykle bezpečně tisknout do 45° (nebo o něco více při nižší výšce vrstvy v poměru k průměru tryska). Mosty lze tisknout na kratší vzdálenosti (v závislosti na druhu filamentu).
- Kontaktní plocha s podložkou (1. vrstva) je zásadní pro úspěšnosti tisku.

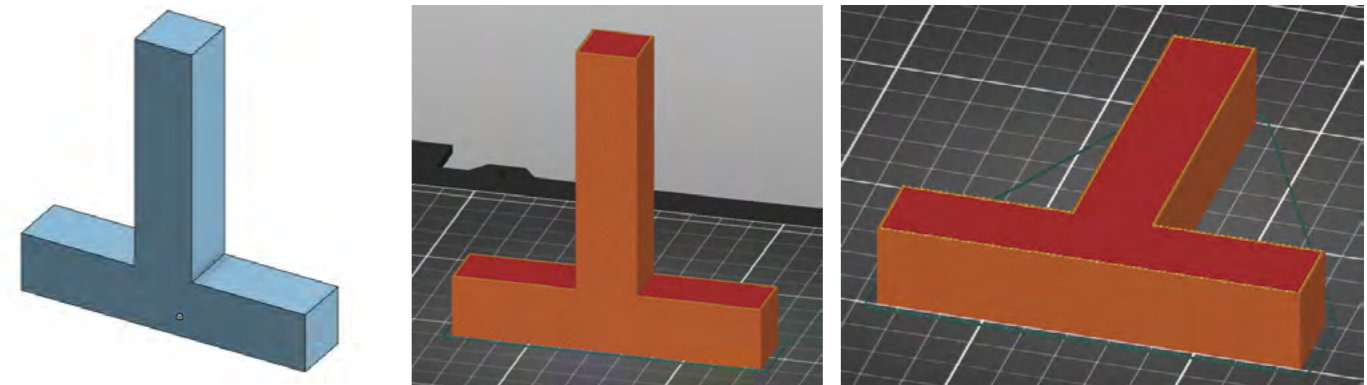
Jak tisknout kvádr na obrázku níže?



Vzhledem k jednoduchosti jeho tvaru je možné vytisknout kvádr položený na kteroukoli z jeho stran. Nicméně pokud bude tištěn „na výšku“, jeho kontaktní plocha s podložkou bude poměrně malá (hrozí odlepení především ke konci tisku) a současně bude poměrně snadné tento model kvádrů zlomit (vzhledem k orientaci vrstev). Vhodnější je orientace na jednu ze stran s větší plochou.

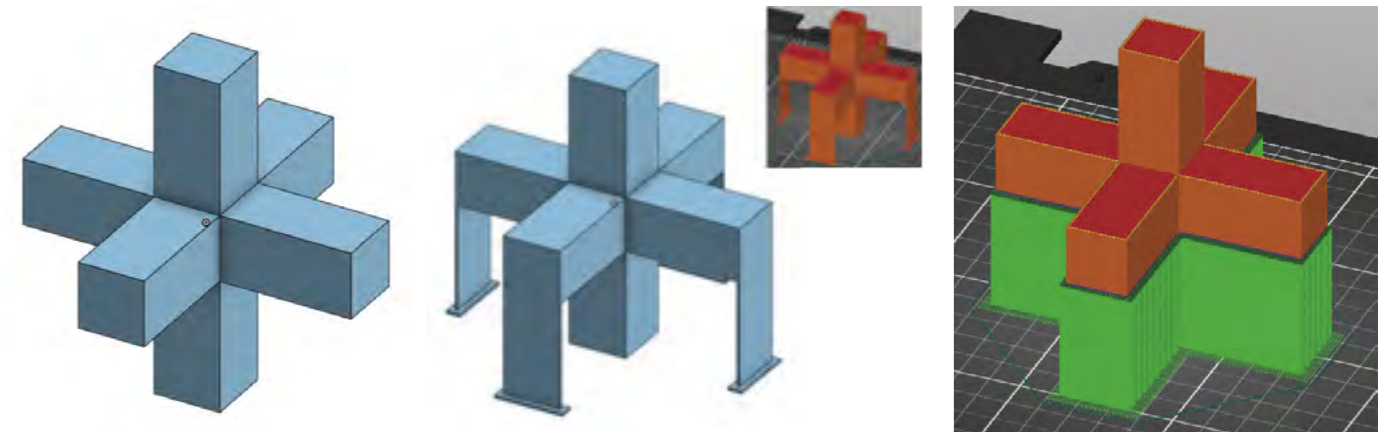


Jak tisknou objekt ve tvaru „T“ na obrázku níže?



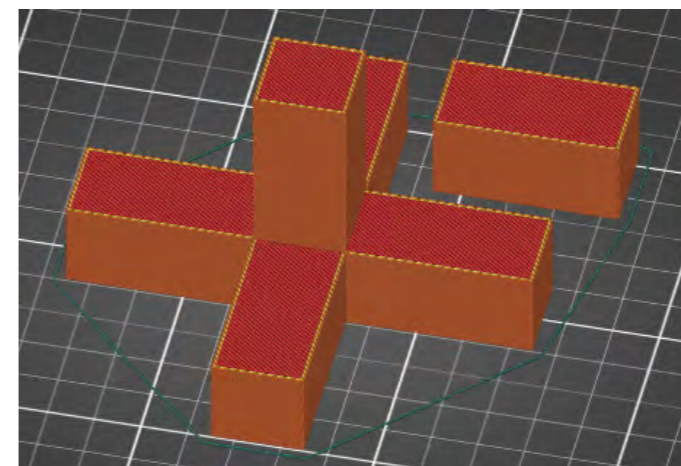
Jedná se o podobnou situaci jako u předchozího příkladu. Přestože základna je v tomto případě při orientaci „na výšku“ větší, s ohledem na pevnost výsledného dílu je vhodnější orientace „na ležato“.

Jak tisknou následující model?

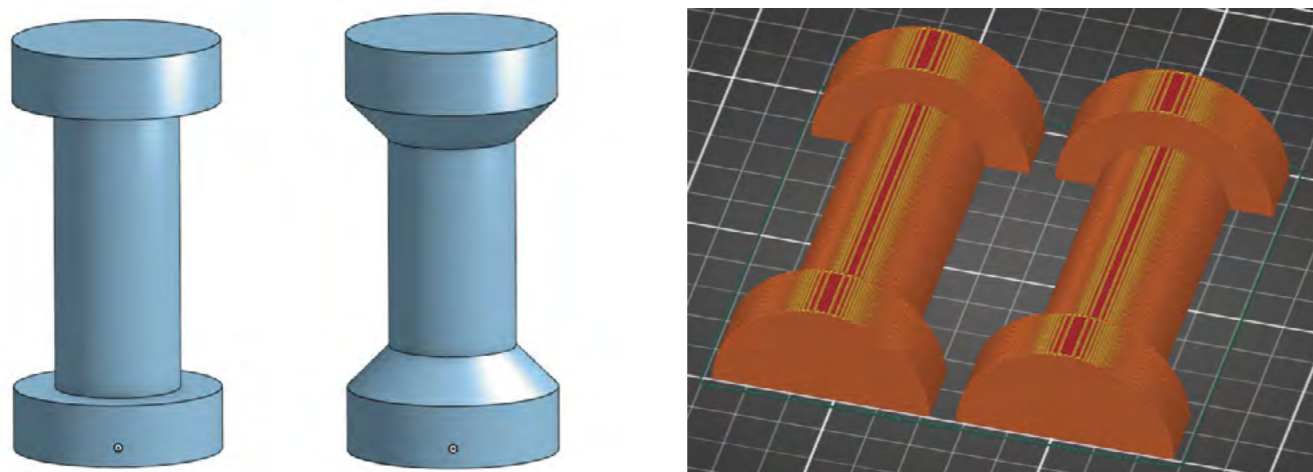


Objekt nelze orientovat tak, aby bylo možné ho vytisknout bez nutnosti podpor. Vzhledem k jednoduchým rovným plochám lze však použít jak automaticky generované podpory ve sliceru, tak si vymodelovat podpory vlastní (které šetří filament a snadněji se oddělují). Využito je zde přitom faktu, že mosty tisknout lze (na rozdíl od 90° převisů).

Alternativně je také možné model rozdělit a vytisknout na 2 části, které se následně slepí či sešroubují (při patřičné úpravě modelu). Na možnost rozdělit model pro tisk na více částí se často zapomíná.

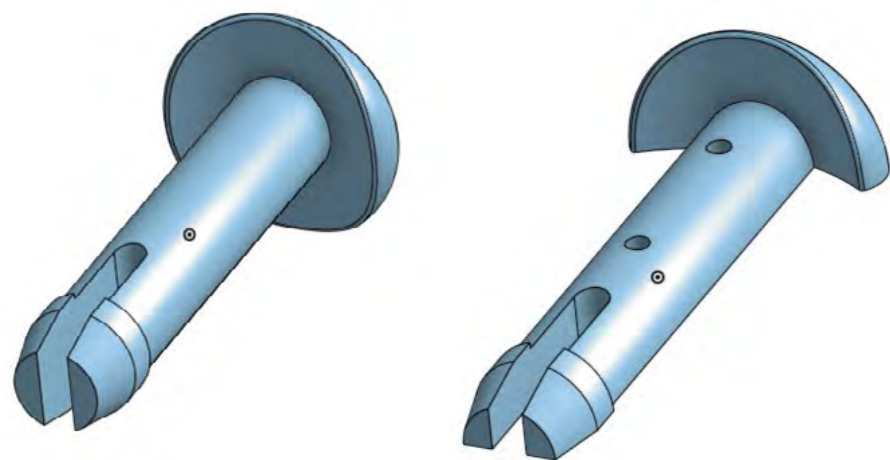


Jak tisknout objekt ve tvaru „činky“ na obrázku níže?

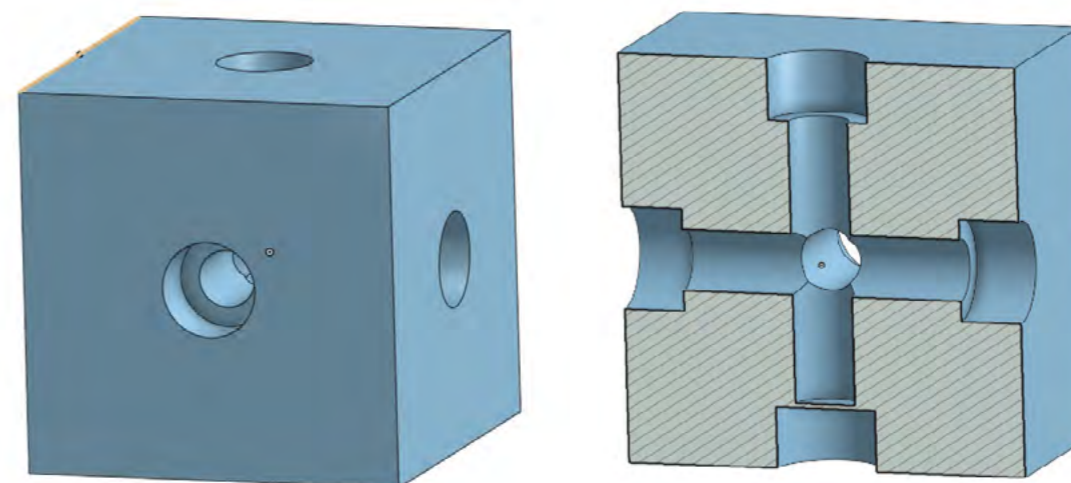


Objekt nelze orientovat tak, aby bylo možné ho vytisknout bez nutnosti podpor. Pokud to použití dílu umožňuje, jedním z řešení je přidání „náběhu“ ve sklonu 45° (z estetického hlediska přidán i do spodní části – není nutné). Opět se však může vyskytnout problém v pevnosti kvůli orientaci vrstev. I tento problém je však možné řešit například zapuštěním dlouhého šroubu skrz celý objekt (na kombinování 3D tisku s jinými běžně dostupnými součástkami se často zapomíná).

Dalším řešením je rozdělení dílu na 2 poloviny, kde každá bude tištěna zvlášť. Výhodou je zachování originálního tvaru (bez „náběhů“) a lepší mechanická pevnost. Poloviny je k sobě následně možno slepit či upravit model např. pro sešroubování (či kombinace). V případě následného lepení je vhodné tisknout na hladkou tiskovou plochu a úprava modelu v podobě pomocných/vodících součástek (např. kruhové sloty, do kterých se zasune filament či drobné kvádry tištěné zvlášť). Příklad z praxe na obr. níže – kolíček z konstrukce bazénu, tištěný na 2 části s otvory 1,8, do kterých se při slepování vlepil filament, který pomohl se snadnějším slepováním obou částí.

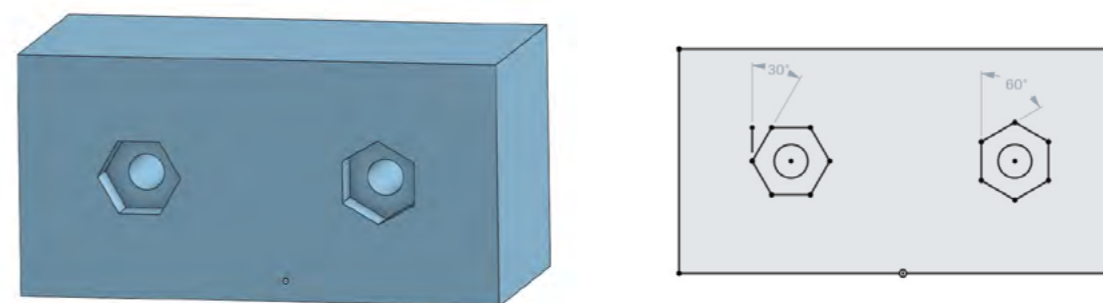


Jak tisknout kostku (se sloty na šrouby ze všech stran) na obrázku níže?

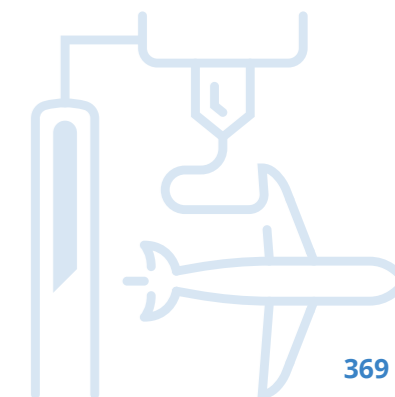


Jediným problematickým místem je ve skutečnosti slot na spodní straně, kde je 90° převis (došlo by zde k tisku kruhových perimetrů do vzduchu). Drobnou úpravou modelu je možné změnit tento převis na most tak, že zde slot uzavřeme v tloušťce 1-2 vrstev. Po vytištění se tento slot snadno opět „otevře“ za použití vrtáčku (či šroubováku nebo přímo šroubem, protože tenký most je poměrně křehký).

Jaká je vhodnější orientace slotu pro matku na kvádro níže? (při zachování orientace objektu pro tisk)



Vhodnější je varianta vlevo, protože tisk 30° převisu a krátkého mostu není problém, zatímco při tisku 60° převisu u varianty vpravo by již mohlo docházet k drobným deformacím při tisku (prověšování).



Práce s 3D tiskárnou

Příprava podložky

Dobře očištěná a odmaštěná tisková podložka je základním předpokladem pro úspěšný tisk. K odmaštění se používá nejčastěji IPA (isopropylalkohol) nebo „okena“ s obsahem alkoholu. Dále je nutno v závislosti na tiskovém materiálu a materiálu podložky (sklo / PEI / apod.) tiskovou podložku připravit. **Některé kombinace a materiálů podložky nevyžadují další přípravu (např. PLA+PEI, PET+PEI)**, v případě některých materiálů je však nutno aplikovat na podložku např. PVA lepidlo (např. Kores tyčinka), maskovací pásku, izolepu, 3D lack (sprej), apod. Vždy je nutné postupovat dle manuálu k tiskárně (dle typu tiskového povrchu) a doporučení výrobce filamentu.

První vrstva

Dobrá první vrstva je jedním z hlavních faktorů pro úspěšný tisk. Kromě dobře připravené podložky je tak velmi důležitá výška trysky pro první vrstvu. Zejména konzistence napříč tiskovou plochou je důležitá. U tiskáren vybavených nějakým typem výškové sondy je výška první vrstvy realizována ofsetem vůči měření této sondy. Některé tiskárny (např. Prusa) umožňují v průběhu tisku první vrstvy ještě softwarově doladit tuto výšku. Ideální výška může být přitom závislá také na struktuře podložky a materiálu filamentu. Tisk první vrstvy je tak nutno vždy sledovat a v případě potřeby zajistit korekci/doladění.

Základní údržba

Jako každý stroj i 3D tiskárna se musí udržovat, aby fungovala správně, tisk probíhal kvalitně a zároveň se prodloužila délka její životnosti. Tiskárna ani filamenty by neměly být uskladněny a používány v prašném či vlhkém prostředí. Po prvním spuštění je potřeba tiskárnu zkalibrovat, stejně tak ji musíme znovu zkalibrovat po např. nějakém větším stěhování, převážení, výměny tiskového plátu atp. Přesnou údržbu je nutno provádět v souladu s manuálem, obv. je vyžadováno občasné promazání ložisek, dotažení řemenů apod.

Vlastnosti 3D tiskárny (FDM)

Níže jsou uvedeny vybrané základní součásti FDM/FFF 3D tiskáren, se kterými by měli být uživatelé (studenti) seznámeni.

Extruder

Jedná se o sestavu, která se stará o vytlačování (pohyb) filamentu. Existují 2 základní typy. Přímý extruder (direct extruder) je extruder umístěný přímo nad tryskou, jeho výhodou je snazší tisk z flexibilních filamentů. Vzdálený extruder (bowden extruder) je extruder oddělený od trysky delším vodičem filamentu (obv. teflonovou hadičkou). Jeho výhodou spočívá v nižší hmotnosti tiskové hlavy (snazší pohyb při vyšších rychlostech).

Senzory

3D tiskárny jsou často vybaveny celou řadou senzorů, s kterými je dobré se seznámit, zejména v případě nutnosti jejich údržby, štelování či výměny. Mezi nejběžnější patří teplotní senzory na trysce a tiskové podložce, koncové spínače na osách (při použití krokových motorů se zpětnou vazbou nemusí být přítomny), senzor filamentu, výšková sonda.

Tryska

Tryska je místo, kde dochází k vytékání roztaveného filamentu. Trysky je možno často měnit za účelem změny průměru trysky (nejběžnější je 0,4 mm). Některé abrazivní materiály vyžadují použití tvrdých trysek, nebo trysek s integrovanou nízko-abrazivní špičkou (např. Olsson Ruby).

Krokové motory

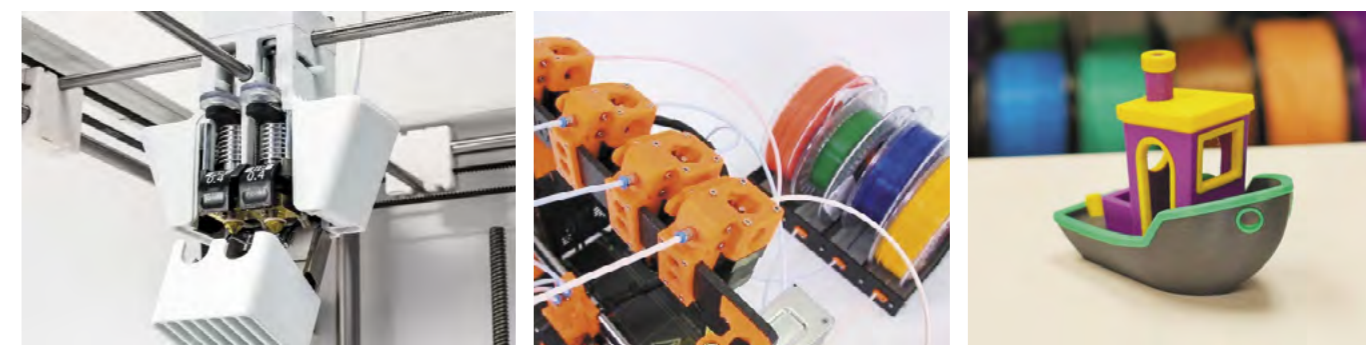
Krokové motory realizují pohyb trysky a filamentu. Jejich vlastnosti přímo ovlivňují kvalitu a rychlost tisku. Některé krokové motory umožňují také využití zpětné vazby k detekci nárazu.

Ventilátory

FFF 3D tiskárna zpravidla obsahuje minimálně 2 ventilátory. Ventilátor na chladiči nad tryskou a tzv. tiskový ventilátor, který se stará o chlazení vytlačeného filamentu (některé levnější tiskárny tento nemusí obsahovat, což ale výrazně komplikuje tisk převisů a mostů). Dále jsou často umístěny ventilátory u zdroje napájení či řídicí elektroniky.

Vícemateriálový tisk

Některé tiskárny umožňují automatický tisk z více materiálů (více trysek či automatické prohazování filamentů). V některých případech se může jednat také o externí přídavné moduly. Díky tisku z více filamentů je možné dosáhnout barevných výtisků, což však vyžaduje, aby byl model připraven na díly dle jednotlivých barev. Dále je také možné využít vícemateriálový tisk především pro tisk podpor z rozpustných materiálů (např. PVA/BVOH filamenty).



Pro tisk jednoduchých vícebarevných objektů je také možné využít ruční prohazování filamentů po vrstvách (podpora např. v PrusaSliceru) či použitím „duhových“ filamentů.



Doporučené pomůcky

Pro výuku je možné doporučit CAD software OnShape, Solidworks, nebo Fusion360. Volba vhodného programu závisí především na licenčních možnostech, zaměření a vybavení školy a jazykových dovednostech žáků/studentů.

- Software OnShape lze doporučit s ohledem na běh bez instalace (ve webovém prohlížeči), aktuálně (2021) však není k dispozici v české jazykové mutaci. Vyžaduje permanentní připojení k internetu v přiměřené rychlosti. Výhodou je možnost kolaborace na jednotlivých projektech.
- Software Solidworks je dostupný v češtině, ale nemá k dispozici bezplatnou verzi pro školy či hobby využití.
- Software Fusion360 má možnost dodatečné lokalizace do češtiny, ale proti výše uvedeným je jeho GUI mírně méně intuitivní a není tak příliš vhodný pro začátečníky.

Každý žák potřebuje svůj počítač s programovým vybavením (modelovací software, slicer, apod.). V závislosti na možnostech školy doporučuji pro výuku alespoň dvě 3D tiskárny, v ideálním případě jednu tiskárnu na cca 4–8 žáků, pro technicky zaměřené školy jednu tiskárnu na 2–4 žáky.

Pro 3D tisk je nezbytné další vybavení a materiál, především filament (pro začátečníky doporučuji jako nejvhodnější materiál PLA a PET) a přípravky na přípravu tiskové podložky (odmašťovač, lepidlo, apod.). Dále je vhodné základní nářadí (špachtle, šroubováky, klíče, brusný papír, apod.). Pro spojování částí modelů je možné zajistit základní spojovací materiál, případně 3D pero (které je možné rovněž použít pro kreativní projekty).

V rámci představitosti a použití rozměrů je vhodné mít k dispozici některé z přesných měřidel (posuvné měřítko, úhloměr apod.).

Přílohou tohoto materiálu jsou vzorové úlohy. Jejich zadání je k dispozici v editovatelné elektronické formě, aby si je každý učitel mohl upravit.

Vzorové úlohy

Řešení následujících úloh je dostupné na:

- Úloha 1: <https://cad.onshape.com/documents/d4fce5697101dc01dbb64f40/>
- Úloha 2: <https://cad.onshape.com/documents/d6782c8dbc58f7b63ca9478b/>
- Úloha 3: <https://cad.onshape.com/documents/b6f4dbf71fc38d118455dde3/>
- Úloha 4: <https://cad.onshape.com/documents/f5f8c685794fb7501affbb3e/> (2 postupy)
- Úloha 5: <https://cad.onshape.com/documents/e1f721cd5793ebd50389935b/>

Po přihlášení do OnShape libovolným účtem je možné vytvořit kopii vzorových úloh, kterou lze následně volně editovat.

Úloha 1: Přívěšek na klíče

Co budeme potřebovat

- Počítač s CAD software a slicerem
- 3D tiskárna
- Cca 5 g filamentu

Zadání

Vytvořte model přívěšku na klíče s Vaší jménem a dalším motivem (zde „smajlík“). Přívěšek by měl mít zaoblené rohy a zkosené hrany. Jméno a motiv mohou být buďto zapuštěny nebo vysunuty. V případě zapuštění je možné vzory aplikovat na obě strany přívěšku.



Poznámky a doporučení

- Aby byl nápis a motiv čitelný, je vhodné ho zapustit/vytáhnout cca 1 mm.
- Zkosení (chamfer) hran je v Z ose vhodnější než zaoblení (fillet), protože zaoblením vznikají převisy více než 45°, které se obtížně tisknou.
- Zapuštěný text je možno zkopírovat na druhou stranu pomocí kruhového vzoru (circular pattern), nikoliv pomocí zrcadlení (mirror).

Co jste se naučili?

- ✓ Osvojili jste si základní tvorbu 2D náčrtů a jejich vytažení do 3D.
- ✓ Funkce jste volili tak, aby byl výsledný model vhodný pro 3D tisk.

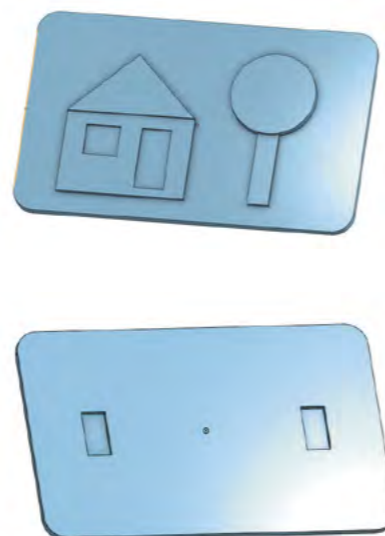
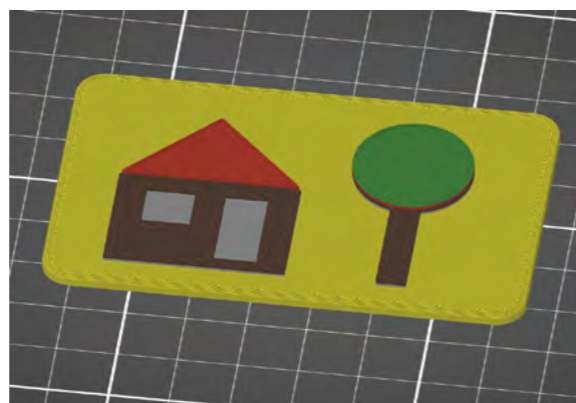
Úloha 2: Magnetek na ledničku

Co budeme potřebovat

- Počítač s CAD software a slicerem
- 3D tiskárna
- Filamenty různých barev
- Magnety nebo magnetickou pásku

Zadání

Vytvořte model „magnetky na lednici“ s motivem dle vlastní fantazie tak, aby bylo možné při 3D tisku využít techniky tisku barev po vrstvách. Tato technika funguje tak, že se vždy po několika vrstvách vymění filament. Magnety/pásku zapusťte (alespoň částečně) ze zadní strany. Nezapomeňte zkosit/zaoblit ostré hrany. Model připravte ve sliceru pro tisk (přidejte výměnu filamentu v jednotlivých vrstvách).



Poznámky a doporučení

- Pokud mají být části modelu jinou barvou, musí mít vůči základně různou výšku.
- Tisk mostů je možný, ale při větších vzdálenostech může dojít v závislosti na filamentu k drobným průvěsům. Je potřeba s tím počítat.
- Když plánujete zapustit jiný díl (magnety/páska), je vždy potřeba počítat alespoň s minimální rezervou (tolerancí).

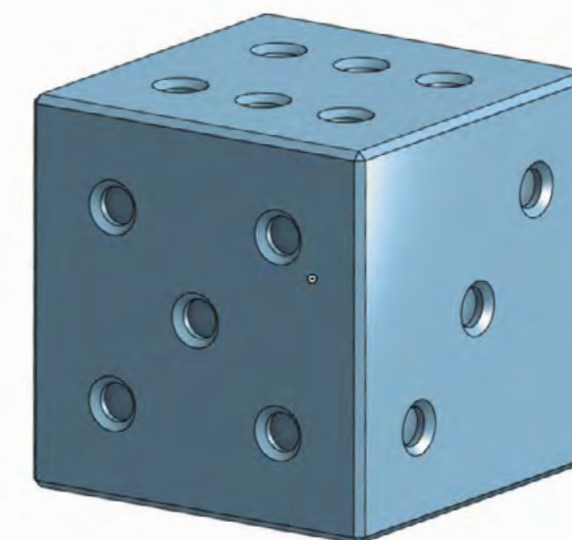
Co jste se naučili?

- ✓ Procvičili jste si praxi tvorby 2D náčrtů a jejich vytažení do 3D včetně pokročilejších závislostí (symetrie, offset).
- ✓ Připravili jste model pro vícebarevný tisk s využitím techniky výměny filamentu po vrstvách.

Úloha 3: Hrací kostka

Co budeme potřebovat

- Počítač s CAD software a slicerem
- 3D tiskárna
- Cca 10 g filamentu



Zadání

Vytvořte model šestistěnné hrací kostky. Nezapomeňte zaoblit/zkosit hrany. Čísla na jednotlivých stranách řešte pomocí zapuštěných „teček“.

Pro pokročilé: Model vypracujte jako parametrický, tedy kostku bude možno zvětšovat/zmenšovat nastavením délky hrany a samostatně nastavovat průměr a hloubku teček.

Poznámky a doporučení

- Zejména pro parametrický model je důležité používat především vazby, nikoliv zbytečně nadužívat kóty. Často jsou užitečné pomocné čáry.
- Někdy je možno využít ve 2D náčrtu „tečky“ z náčrtu na protilehlé straně kostky.
- Výhodou parametrického modelu je zde možnost nezávisle nastavovat velikost teček. Pokud by to nebylo potřeba, lze kostku pochopitelně symetricky zvětšovat např. ve sliceru.

Co jste se naučili?

- ✓ Osvojili jste si práci s různými rovinami včetně možnosti využití prvků z jiných rovin („promítnutí“).
- ✓ Procvičili jste si používání pomocných čar, definice vztahů a závislostí.
- ✓ Naučili jste se vytvářet parametrický model (pro pokročilé).

Úloha 4: Krabička

Co budeme potřebovat

- Počítač s CAD software a slicerem
- 3D tiskárna
- Cca 30 g filamentu

Zadání

Vytvořte model kulaté krabičky složené ze dvou částí (spodní a víčko). Tyto části budou držet uzavřeny pomocí tření a přirozených „drážek“ mezi vrstvami. Můžete přidat také prolisy pro snazší otevírání.

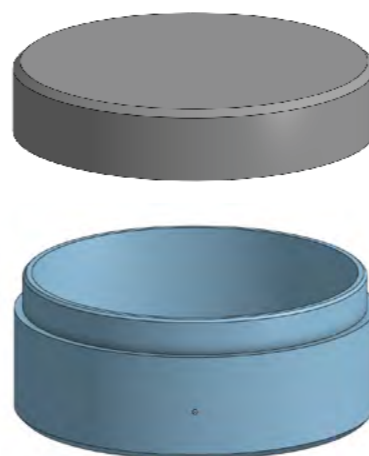
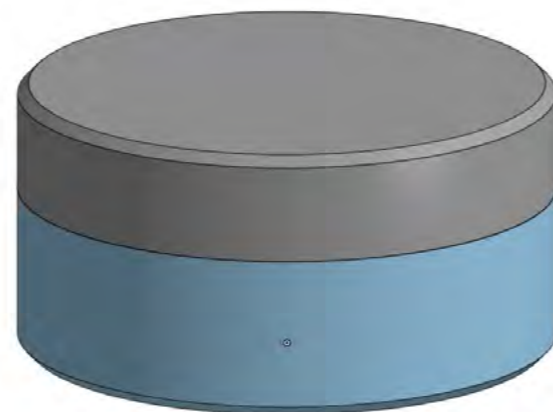
Pro pokročilé: Krabičku je možno vytvořit za použití jediného 2D náčrtku (a to hned minimálně dvěma způsoby).

Poznámky a doporučení

- Při práci s více díly je často užitečné si některé díly skrývat nebo použít funkce jako průhlednost či průřez.
- V závislosti na přesnosti tisku konkrétní 3D tiskárny může být potřeba přidat drobné tolerance (mezery), pokud krabička nepůjde zavřít.
- Tloušťka stěny krabičky by ideálně měla být v násobcích průměru tiskové trysky.

Co jste se naučili?

- ✓ Osvojili jste si práci s modelem složeným z více částí.
- ✓ Naučili jste se využívat závislosti pro „vytahování“ z 2D náčrtu do 3D modelu.



Úloha 5: Spojování dílů

Co budeme potřebovat

- Počítač s CAD software a slicerem
- 3D tiskárna
- Cca 10 g filamentu
- Spojovací materiál (šroubky, maticky)
- Posuvné měřítko

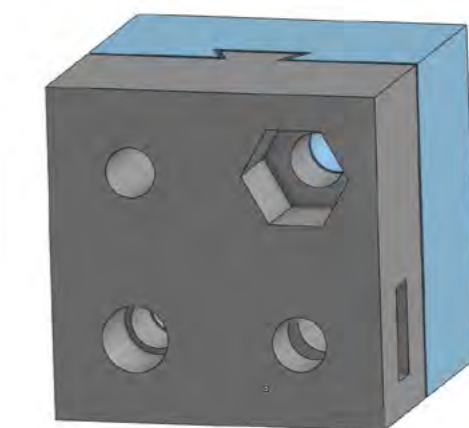
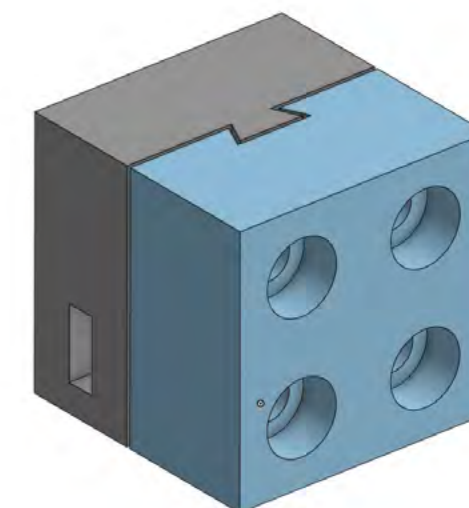
Zadání

Existují různé způsoby, jak spojit 2 tištěné díly dohromady. Vyzkoušejte si některý z níže uvedených na jednoduchém modelu krychle. Způsoby lze mezi sebou kombinovat.

- Zasunutí dílů pomocí „zámků“
- Slepění dílů (ideálně s použitím pomocných/vodících dílů či zapuštění)
- Šroubování přímo do plastu
- Zapuštění matek různého druhu (šestihránné, čtvercové, zápustné, apod.)

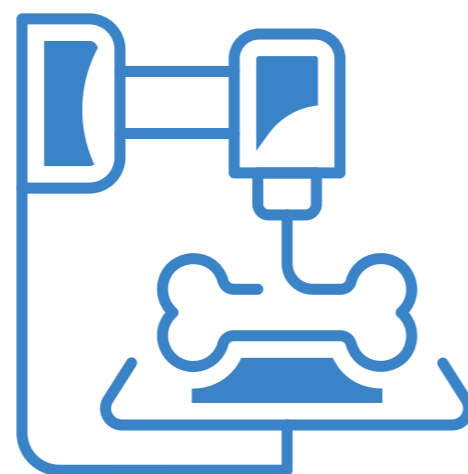
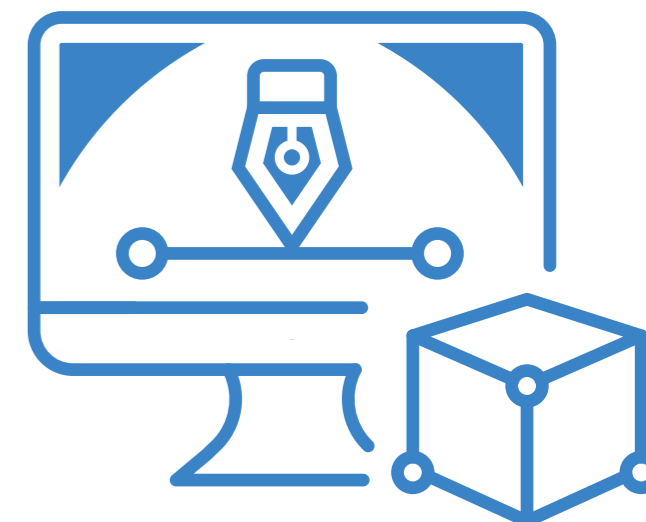
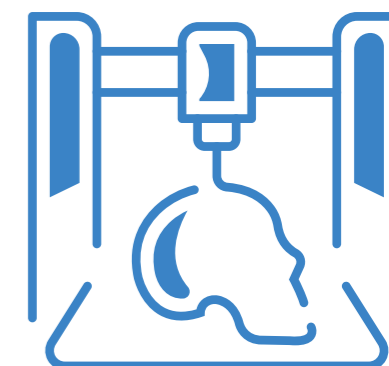
Poznámky a doporučení

- Funkce vysunutí z 2D do 3D (stejně jako další funkce) lze aplikovat na 1 či více dílů.
- Zasunutí tištěných dílů obvykle vyžaduje alespoň malou rezervu/toleranci (např. 0,2–0,3 mm).
- Šroubování přímo do plastu je možné, pokud se nepředpokládá časté rozebírání.
- Matky vyžadují malou rezervu/toleranci, která ale nesmí být příliš velká, aby se matka neprotočila.
- Pro zasunutí z boku (vůči šroubu) jsou vhodnější čtvercové matky.
- Zápustné matky lze do plastu snadněji vložit pomocí jejich nahřátí (např. pájkou).
- Vhodnější orientace šestihránné matky pro tisk je taková, kde 2 ze stran jsou orientovány horizontálně (a žádná vertikálně).



Co jste se naučili?

- ✓ Procvičili jste si tvorbu modelu tvořeného více částmi.
- ✓ Prakticky jste si vyzkoušeli různé způsoby spojování tištěných dílů.





IMPULS
PRO KARIÉRU
A PRAXI

