Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

DIPLOMOVÁ PRÁCE

Podpora rozvoje učitelských kompetencí ve výuce informatiky

2020

Bc. Tereza Nedjalková

Vedoucí práce: doc. RNDr. Miroslav Kolařík, Ph.D.

Studijní obor: Učitelství výpočetní techniky pro střední školy, prezenční forma

Bibliografické údaje

Autor:	Bc. Tereza Nedjalková
Název práce:	Podpora rozvoje učitelských kompetencí ve výuce informatiky
Typ práce:	diplomová práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2020
Studijní obor:	Učitelství výpočetní techniky pro střední školy, prezenční forma
Vedoucí práce:	doc. RNDr. Miroslav Kolařík, Ph.D.
Počet stran:	102
Přílohy:	CD
Jazyk práce:	český

Bibliograhic info

Author:	Bc. Tereza Nedjalková
Title:	Supporting the development of teaching competencies in computer science education
Thesis type:	master thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2020
Study field:	Computer Science for Education, full-time form
Supervisor:	doc. RNDr. Miroslav Kolařík, Ph.D.
Page count:	102
Supplements:	CD
Thesis language:	Czech

Anotace

Ve spolupráci s neziskovou organizací Czechitas z.s. diplomantka vytvořila metodický materiál pro podporu rozvoje učitelských kompetencí pro výuku informatiky na základních a středních školách. Výukový text svým obsahem reaguje na připravované změny v RVP a jeho hlavním cílem je zvvýšit kvalitu a atraktivitu výuky informatiky.

Synopsis

In cooperation with Czechitas z.s. the diploma student created a methodological material to support the development of teaching competencies for teaching computer science at primary schools and secondary schools. The educational text responds to the prepared changes in the FEP (Framework educational programme) and its main goal is to increase the quality and attractiveness of information education technologies.

Klíčová slova: vzdělávání; informatika; technologie; metodika; digitální dovednosti; bezpečnost; algoritmizace; programování

Keywords: education; informatics; technology; methodology; digital skills; safety; algorithmization; programming

Na tomto místě bych ráda poděkovala panu doc. RNDr. Miroslavu Kolaříkovi, Ph.D, za trpělivost a cenné rady, které mi pomohly při tvorbě této práce. Druhý obrovský dík patří všem z týmu Czechitas, keří se podíleli na realizaci myšlenky Učitelské akademie.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracovala samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Úvod

Pro svou diplomovou práci jsem zvolila téma *Podpora rozvoje učitelských kompetencí ve výuce informatiky*. Cílem práce bylo vytvořit pomocný materiál pro začínající i zkušené učitele informatiky, kteří se potýkají s připravovanými změnami v rámcovém vzdělávacím programu (dále RVP), ke kterým se pojí rozšíření výuky informatiky o několik nových témat.
Problémem je skutečnost, že mezi učiteli informatiky je velká část neaprobovaných vyučujících. Jedná se většinou o učitele jiných přírodovědných předmětů, kteří úvazek v informatice dostali z důvodu absence aprobovaného učitele na dané škole. Pro tyto učitele jsou připravované změny velkým „strašákem“, jelikož daný obor nestudovali a nemají tedy v oblasti informačních technologií (dále IT) takový přehled, aby byli schopni nová témata zajímavě a srozumitelně představit svým žákům a studentům. A právě tito učitelé mi byli inspirací pro téma mé kvalifikační práce.
Společně s kolegyněmi a kolegy z neziskové organizace Czechitas jsme se rozhodli těmto vyučujícím podat pomocnou ruku a vytvořit materiál, díky kterému by si doplnili mezery ve svých vědomostech a posílili základy svého IT vzdělání. Po konzultaci s několika odborníky nejen z řad Czechitas jsem dala dohromady seznam témat, která se objevují v připravovaném, revidovaném RVP a která by, dle mého i jejich názoru, byla pro učitele zajímavá a obohacující. Následně jsem, na základě dotazníkového šetření, získala na koncept projektu cennou zpětnou vazbu od zástupců cílové skupiny a doplnila jsem několik dalších oblastí vzdělávání, po kterých byla ze strany učitelů poptávka.

V první části práce se zaměřím na připravovanou revizi RVP v oblasti informatiky a informačních technologií. Ve stručném přehledu představím vizi Národního ústavu pro vzdělávání (dále NÚV) a průběh vytváření nového kurikula.

V druhé části se budu věnovat projektu „Učitelská akademie“, jeho jednotlivým částem a tématickým modulům, kterými jsou:

1. Úvod do světa IT
2. Metodika výuky
3. Digitální dovednosti
4. Bezpečnost IT
5. Algoritmizace
6. Programování

1 Revize RVP [1]

Rámcový vzdělávací program tvoří obecně závazný rámec pro tvorbu školních vzdělávacích programů škol všech oborů vzdělání v předškolním, základním, základním uměleckém, jazykovém a středním vzdělávání.
Rámcové vzdělávací programy jsou dokumenty, které je potřeba po jisté době podrobit revizi. Do všech RVP vymezujících předškolní, základní i střední vzdělávání je třeba promítnout změny, které se odehrávají jak ve společnosti, tak ve sféře ekonomické, politické, sociální či přírodní.
Změny probíhají mimo jiné i ve struktuře a náročnosti povolání. Je velmi pravděpodobný vznik nových a zcela odlišných profesí, pozic a s tím se pojící i zánik těch stávajících.
Revizí rozumíme cílené a dlouhodobé přezkoumávání RVP a zpracovávání návrhů pro jejich úpravy na základě přijatých pravidel a postupů. Pro přezkoumávání jsou využívány poznatky kurikulárního výzkumu a teorie, zjištění z analýz výsledků nejrůznějších zkoušek, jako státní části maturity, závěrečných zkoušek nebo přijímacích zkoušek na střední školy zakončené maturitou.
Základním cílem je připravit moderní kurikulum reagující na nové potřeby společnosti i vybrané problémy, které byly identifikovány ve stávajícím vzdělávání. Revize se tedy musí týkat především obsahu vzdělávání, který je třeba koncipovat tak, aby odpovídal současným společenským potřebám.
Úkolem je jednoznačně vymezit obsah a rozsah vzdělávání společný pro všechny. Ten by měl být základem pro individuální rozvoj každého žáka. V současné podobě obsahu RVP to tedy znamená revidovat především cílové struktury, očekávané výsledky vzdělávání, upravit, změnit, redukovat a doplnit vše nezbytné. Tyto zamýšlené změny by měly přispět k tomu, aby žáci a studenti dosahovali požadovaných výsledků učení, měli dostatečný časový prostor na získání a upevnění nabytých znalostí a rozvoj tvořivosti. Měli by být vybaveni dovednostmi a postoji potřebnými k řešení problémů, schopni přizpůsobovat se změnám a umět se vyrovnat s jejich následky.

1.1 Návrh revizí RVP v oblasti informačních a komunikačních technologií (ICT) [1]

Základní východiska a teze revizí ICT kurikula

1. **Rozsah revizí RVP pro základní, gymnazialní a odborné vzdělávání**: Je nutno revidovat více částí RVP, nejen oblast *Informační a komunikační technologie* (pro základní vzdělávání – ZV), respektive *Informatika a informační komunikační technologie* (pro gymnázia – G) nebo *Vzdělávání v informačních a komunikačních technologiích* (pro střední odborné vzdělávání – SOV).

2. **Nároky na časovou dotaci v učebním plánu**: Vědomosti a dovednosti v oblasti digitální gramotnosti a informatického myšlení je třeba rozvíjet po celou dobu školní docházky, tedy již od předškolního vzdělávání.

3. **Rozvoj digitální gramotnosti**: Je třeba na něj dbát nejen v hodinách informatiky, ale vhodnou formou zařazovat problematiku do různých předmětů.

4. **Rozvoj informatických kompetencí**: Porozumění informatice je vyžadováno ve stále více oblastech vzdělávání a pracovních oborech. Proto je nutné klást důraz na vzdělávací obsah již od počátku základní školy.

5. **Rozvoj oborových kompetencí dalších vzdělávacích oblastí**: Vývoj informačních a komunikačních technologií je velmi dynamický a jejich využití v nejrůznějších oblastech lidských činností se neustále rozšiřuje. Proto je revize všech vzdělávacích oblastí RVP nezbytná

6. **Využití digitálních technologií ve výuce a vzdělávání**: Oblast digitálních technologií přináší do vzdělávání řadu příležitostí. Schopnost využívat digitální technologie pro učení, vzdělávání sama sebe a zvyšování vlastní kvalifikace. Zařazení digitálních technologií do výukových aktivit a napojení na neformální vzdělávací aktivity žáků je proto nutnou částí rozvoje digitální gramotnosti žáků.

Digitální gramotnost [1]

Digitální gramotností rozumíme soubor digitálních kompetencí (vědomostí, dovedností, postojů, hodnot), které jedinec potřebuje k bezpečnému, sebejistému, kritickému a tvořivému využívání digitálních technologií při práci, při učení, ve volném čase i při svém zapojení do společenského života. [2]
Digitální kompetence chápeme jako průřezové klíčové kompetence, tj. Kompetence, bez kterých není možné rozvíjet u dětí a žáků plnohodnotně další klíčové kompetence [2] . Jejich základní charakteristikou je aplikace[1] – využití digitál-

[1] podle [3] je gramotnost definována jako „schopnost aplikace některých specifických dovedností“

ních technologií při různých činnostech nebo řešení problémů.
Oblasti digitálních kompetencí:

1. Člověk, společnost a digitální technologie.
2. Tvorba digitálního obsahu.
3. Informace, sdílení a komunikace v digitálním světě.

Informatické myšlení [1]

Informatické myšlení[2] je způsob uvažování, které jedinci umožňuje rozpoznávat aspekty světa a využívat informatických prostředků k porozumění a uvažování o přirozených i umělých systémech a procesech. Informaticky myslící jedinec (žák) při řešení nejrůznějších životních situací cílevědomě a systematicky volí a uplatňuje optimální postupy. K tomu využije schopnosti [5]:

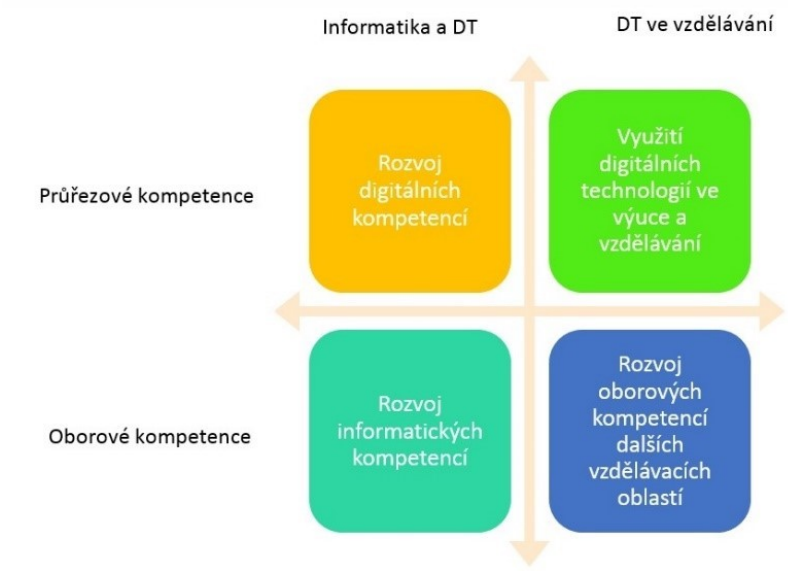
1. Rozpoznávat a formulovat problémy s ohledem na jejich řešitelnost.
2. Získávat, zaznamenávat, uspořádávat, strukturovat a předávat data a informace.
3. Rozkládat systémy a procesy na části, odhalovat jejich vztahy a strukturu, modelovat situace.
4. Vytvářet a modelovat postupy a řešení, která lze přenechat k vykonání jinému člověku nebo stroji.
5. Vytvářet formální popisy skutečných situací a pracovních postupů.
6. Testovat, analyzovat, vyhodnocovat, porovnávat a vylepšovat uvažovaná řešení.

Koncept rozvoje digitální gramotnosti a informatického myšlení dětí a žáků [1]

U žáků od počátku školní docházky je třeba rozvíjet digitální, informatické i ostatní oborové kompetence související s používáním digitálních technologií (dále DT) v systému, který obsáhne celou školní výuku, zahrnuje aktivity žáků ve škole i jejich zkušenosti z aktivit mimo školu. Pro názornost lze způsob, jak rozvíjet digitální a informatické kompetence žáků, schematicky rozdělit do čtyř oblastí, které by měly být součástí školního vyučování, viz obrázek 1 [6].

Digitální gramotnost je ve schématu znázorněna jako *Digitální kompetence* a informatické myšlení jako součást *Informatických kompetencí*. Toto rozdělení naznačuje i jejich začlenění do kurikula. Digitální kompetence budou rozvíjeny

[2]Představení konceptu informatického myšlení viz [4]



Obrázek 1: Koncept rozvoje digitálních dovedností a informatických kompetencí žáka [6]

převážně průřezově a rozvoj informatického myšlení bude převážně záležitostí vzdělávacího předmětu Informatika.
V levé horní části se nacházejí vzdělávací cíle a na ně navázané výukové aktivity, které se zaměřují na rozvoj digitálních kompetencí, přičemž tento úkol mají učitelé většiny předmětů. Podstatou je rozvíjet digitální kompetence promyšleným způsobem napříč různými předměty.
V pravé horní části se nacházejí aktivity, které podporují výuku a učení žáka obecně. Učitel i žák využívají technologie jako didaktické prostředky. Je na učiteli, jaké prostředky pro danou výuku zvolí a jak zprostředkuje konkrétní učivo pro danou skupinu žáků.
V levé spodní části se nacházejí vzdělávací cíle a s nimi související výukové aktivity zaměřené na rozvoj informatických kompetencí. Jedná se o cíle a aktivity vycházející z oboru Informatika. V pravé spodní části se nacházejí vzdělávací cíle a s nimi související výukové aktivity zaměřené na využití digitálních technologií v oboru.

1.2 Průběh revizí ICT kurikula [1]

Revize RVP v oblasti informačních a komunikačních technologií probíhají ve kmenovém úkolu Národního ústavu pro vzdělávání *Inovace ICT kurikula – úkoly plynoucí ze Strategie digitálního vzdělávání*. Práce byly zahájeny v roce 2016 na základě dokumentu *„Tvorba a revize kurikulárních dokumentů pro předškolní, základní a střední vzdělávání na národní úrovni“*, který 5. dubna 2016 schválila porada vedení MŠMT.
Práce v úkolu byly rozděleny na 4 dílčí části:

1. **Zpracování podkladů**, podnětů a doporučení k úpravám RVP ve všeobecně vzdělávací části v oblasti informatiky a informačních a komunikačních technologií a návrh koncepce rozvoje digitální gramotnosti a informatického myšlení žáků.
   V roce 2016 byl zpracován interní materiál, kde byly shrnuty podněty k revizím a popsán návrh koncepce rozvoje digitální gramotnosti a informatického myšlení žáků.

2. **Komplexní revize vzdělávací oblasti Informační a komunikační technologie** (ZV), resp. Informatika a informační a komunikační technologie (G) nebo Vzdělávání v informačních a komunikačních technologiích (SOV).
   Byl připraven návrh nového vzdělávacího obsahu této vzdělávací oblasti. Vzdělávací oblast byla rozdělena na dvě části. V první části vlastní obor Informatika obsahuje popis očekávaných výstupů v informatických tématech a v tématech na rozhraní informatiky a uživatelských dovedností. Ve druhé části jsou popsány očekávané výstupy digitálních kompetencí žáků, jež tvoří souhrnně digitální gramotnost. Předpokladem rozvoje těchto kompetencí je promyšlené a plánované využívání digitálních technologií ve výuce různých předmětů tak, aby žáci měli dostatek příležitostí učit se pod vedením učitele bezpečně a tvořivě pracovat s digitálními technologiemi.

3. **Komplexní revize vzdělávacích oblastí v RVP pro předškolní vzdělávání (PV)**, kde se identifikovaly počátky rozvoje digitální gramotnosti a informatického myšlení dětí v předškolním vzdělávání.

4. **Dílčí revize ostatních vzdělávacích oblastí**. Zde se identifikovala témata ostatních vzdělávacích oblastí, ve kterých rozvoj digitálních technologií zasáhl do jejich vzdělávacího obsahu. Tímto způsobem se pokryly základy pro rozvoj digitálních kompetencí.

2 Projekt Učitelská akademie

Při tvoření konceptu *Učitelské akademie* jsem zvolila následující postup. Nejdříve posbírat podněty, podle nich nastínit řešení, tento koncept rozebrat a tzv. opřipomínkovat a následně vybrat jen to nejlepší.
Pro realizaci myšlenky bylo potřeba sesbírat podněty od zástupců námi nastíněné cílové skupiny, aby mohl být program efektivně vymyšlen a aby obsahoval vše, co případní zájemci potřebují a ocení.
Při monitoringu současných problémů učitelů na nejrůznějších konferencích (UčitelIN, Erasmus+, IKAP, zasedání Digikoalice) vyvstaly tyto jako nejzávažnější a nejčastěji se opakující:

- Revize RVP mi „dělá život těžší“, kde mohu získat potřebné vzdělání v oboru? Programování a jeho didaktika je jen jedním z témat, ke kterému chybí ucelený materiál. Hledám materiál/kurz, kde se mohu naučit metody, jak na výuku programování.
- S učením začínám a nevím, jak se zorientovat mezi jednotlivými přístupy. Které jsou didakticky vhodné a fungují například v kontextu programování na 2. stupni ZŠ?
- Chci začít, ale mezi normálními povinnostmi nenacházím čas se důkladně a svépomocí zorientovat.
- Ztrácím čas (úvazku i konkrétní hodiny) správou techniky.
- Chybějící podpora kolegů i vedení („informatické olympiády nestojí za absenci z výuky, jelikož logické myšlení se vyučuje v matematice“, nedostatek vybavení, chybí možnost dalšího vzdělávání).
- Nedostatek zdrojů v češtině (takových, které by odpovídaly potřebám učitelů – uvádějící cíle, úroveň, čas, nástroje/vybavení).
- Nedostatečná hodinová dotace.

Na základě zjištěných problémů byl nastíněn projekt *Učitelská akademie*, který má svým obsahem reagovat právě na tyto nedostatky a snažit se co nejlépe pomoci k jejich odstranění.
Cílem projektu je vytvoření výukového programu pro učitele a následné zlepšení IT vzdělávání. Důvodem a hlavním podnětem k vzniku této práce je, mimo výše zmíněné podněty učitelů z praxe, skutečnost, že máme obecně nedostatek učitelů IT. Školy problém řeší tak, že výuku informatiky přidělují učitelům, kteří IT neznají, ale mají k němu nějaký vztah. Tedy vybraným technologiím rozumí a problematika je zajímá, ale chybí jim komplexnější znalosti a učitelské dovednosti z oblasti informačních technologií. Druhou částí cílové skupiny jsou studenti, kteří opouštějí vysokou školu a přichází do IT oboru nedostatečně připraveni/bez praxe.

2.1 Vývoj projektu

První částí projektu byl společný brainstorming mě a několika kolegů z organizace Czechitas, z mého pohledu odborníků na vzdělávání, školní i mimoškolní. Z tohoto setkání vzešel první návrh obsahu kurzu, který vypadal následovně:

1. Jak učit IT – Algoritmy a programování.
    - Algoritmy – Automatizace.
    - Platforma Code.org a její využití ve výuce.
    - Jednodušší roboti.
    - Aktivity a hry inspirovány souborem volně dostupných výukových materiálů zaměřených na seznámení s počítači – CS Unplugged (Computer Science Unplugged).
    - Rozklad problému na podproblémy.
    - Aktivity zaměřené na schopnosti řešit problémy bez využití informačních technologií.
    - Programování – vybrané dva jazyky vysvětlené důkladněji.
    - Čím začít ve výuce programování? Podmínky, cykly, proměnné, ...
2. Metodická specifika.
    - Jak si mají dělat poznámky?
    - Jak poznám, že stíhají? Že už splnili úlohu?
    - Důraz na to, aby se děti i učitelé učili vzájemně.
3. Orientace v IT světě.
    - Architektura (knihovny aj.).
    - Přehled programovacích jazyků.
4. Digitální dovednosti.
    - Instalace a správa.
    - Bezpečnost uživatelská/Hacking (ilegální průnik do počítačového systému pomocí prolomení bezpečnostní ochrany)/Oprávnění.
    - Debugging (ladění – postup pro nalézání a snižování množství chyb v programu).
    - Klávesové zkratky.
    - Audio, video, grafika.
    - WiFi zapojování.
    - Hygiena práce.

- Práce s daty a sdílení dat přes Cloud (Google Drive, Dropbox, atd.).
- Prostředky online komunikace (elektronická pošta, okamžité posílání zpráv (messaging) a sociální sítě).
- Soukromí (privacy).

5. Zpracování dat.
   - MS Excel.
   - Vizualizace dat.
6. Technologie v neinformatické výuce.
   - Poznámky ze školy (kam si je psát?).
   - GPS v zeměpise.
   - Trénink (jazyky aj.).
   - Tvorba hudby, videí, grafiky.
   - Úprava fotek.
   - Využití MS Minecraft for Education (Chemie).
   - Matematika – Geogebra.
   - Simulace stavby ve virtuálím prostředí (mosty, raketa do vesmíru, ...).

Všechna témata mají společných několik cílů, které by měly být naplněny. V prvé řadě je to odůvodnění, proč vlastně danou látku učit a proč je pro žáky důležitá. Nedílnou součástí každého tématického celku je postavení zúčastněných učitelů do pozice žáků, aby si zažili aktivity z jejich pohledu, nebo také sdílení zkušeností (dobrých i špatných) z praxe nejen v rovině lektor – účastníci, ale také mezi účastníky navzájem.

Výzkum v cílové skupině

Na základě dotazníkového šetření mezi pedagogy bylo zjištěno, že hlavní témata, kterým učitelé chtějí lépe porozumět jsou *algoritmizace, programování, robotika, umělá inteligence, využití mobilního telefonu/tabletu ve výuce, sdílení dat.* Dále projevili respondenti zájem o *přehled vhodných výukových pomůcek a vhodných online aplikací*, které by mohli ve výuce použít, ideálně pak těch bezplatných. Častým společným problémem bylo totiž získávání financí na modernizaci ICT ve škole.
Při zjišťování oblastí, které by se dotazovaní chtěli naučit lépe učit se pak mimo výše zmíněná témata objevilo *bezpečnost, zpracování dat, tvorba a úprava fotografií a videí, práce online.*
Na obecnou otázku „Co by nemělo ve studiu chybět“ se opakovaly nejčastěji odpovědi zahrnující *novinky ze světa IT, poptávka firem/vysokých škol, sdílení*

zkušeností, logické úlohy, možnost vyzkoušet si nové technologie a tipy na zařazení *projektové výuky do IT*.
Co se týče podoby případného reálného kurzu, počet hlasujících za online výuku/webináře byl srovnatelný s těmi, kteří by upřednostnili přímou výuku i s nutností dojíždět do většího města.
Za účelem sběru podnětů a důležitých rad byla navázána spolupráce s odborníky na vzdělávání z firmy Microsoft (která má vlastní vzdělávací platformu pro učitele – Microsoft 365 Education) či Jednoty školských informatiků (dále JŠI), kteří jsou skvělým ukazatelem toho, jaká je skutečná situace na školách.
Po zapracování připomínek ze strany zástupců výše zmíněných organizací a účastníků dotazníkového šetření se osnova projektu ustálila na následující podobě, od které se posléze odvíjela tvorba materiálů k jednotlivým modulům.

Cíle studia

Cílem studia je propojit učitele a odborníky, porozumět IT, naučit se lépe učit a mít k dispozici konkrétní podklady. Také je účastníkům umožněno mezi sebou diskutovat a sdílet zkušenosti.

Cílová skupina

Pro koho je studium určeno?

- Pro všechny učitele, kteří mají vystudovaný jiný obor a chtějí se rekvalifikovat na učitele informatiky.
- Pro ty učitele, kteří se necítí v oblasti informačních technologií jisti (například v důsledku toho, že informatiku studovali dávno a nestíhají sledovat rychlý vývoj tohoto oboru).
- Pro vysokoškolské studenty, kteří se připravují na vstup do praxe.

Témata jsou vhodná pro učitele základních i středních škol. U témat, kde je výrazný rozdíl cílových skupin bude studium rozděleno (např. programování – ZŠ: Scratch, SŠ: JavaScript).

Výukové moduly

1. **Orientace v IT a technologiích:** Cílem modulu je posílit rozhled učitele a jeho základní orientace ve světě IT. Během výuky si představíme, z jakých fází se skládá řešení IT projektu a co za ním všechno stojí. Podíváme se na základní pojmy, bez kterých se v informatice jen těžko obejdeme.
   Učitel potřebuje mít jistotu, kam jeho výuka zapadá. Musí vědět proč to dělá a může o tom diskutovat s dětmi. Také může získat podporu v dalších učitelích. Pochopí, jak malá část IT je programování. Je mentorem a průvodcem ve světě IT. On se stává tím, ke komu mohou mladí vzhlížet.

2. **Metodika výuky:** Cílem tohoto modulu je naučit se efektivněji učit (nejen programování, znalosti budou přenositelné i do dalších oblastí).
   Učitel potřebuje získat potřebné „know how“ k tomu, aby dokázal efektivně vysvětlovat a předával své znalosti srozumitelně a poutavou formou. Žáky je třeba zaujmout a přimět je k tomu, aby si z každé hodiny odnesli co nejvíce.

3. **Digitální dovednosti:** Cílem modulu je naučit se naplno využívat nejrůznější technologie, které máme k dispozici. Seznamujeme se se základními pojmy a postupy týkajícími se online kanceláře a jejího využití ve vzdělávání. Setkáváme se s balíkem služeb G Suite a naučíme se je naplno využívat při přípravě na výuku i v jednotlivých hodinách napříč předměty. Dále se v tématu zaměřujeme na umělou inteligenci v rukách učitele, její vývoj v posledních desetiletích i to, kam se bude v nastávající době posouvat. Důležitou součástí výukového procesu je navázat spojení s žákem. Učitel je mediátorem výuky, on je ten, který něco přináší. Dokáže pokládat otázky, které jsou pro žáka výzvou. Posiluje se důvěra žáka v učitele, hodiny se stávají interaktivnější. Ve třídě se všichni učí navzájem a sdílí mezi sebou vědomosti a znalosti.

4. **Bezpečnost v IT:** Cílem modulu je ovládnout svět informačních technologií, umět získávat informace a užívat internet bezpečným způsobem. Zároveň bude téma bezpečnosti propojeno v souladu s RVP a školními předměty. Účastníci tak získají cenné a ucelené informace o postupech v případě problémů s kyberšikanou či se dozví, jak rozpoznat falešné zprávy (fake news).

5. **Algoritmizace:** Co je to algoritmus? V průběhu tohoto bloku budeme nad problémy přemýšlet analytickým způsobem, rozkládat je na dílčí části a postupně je řešit.
   Naučíme se zacházet s několika typy programovatelných hraček (Ozobot, Lego Mindstorms) a ukážeme si, jak s jejich pomocí ozvláštnit a přiblížit téma algoritmizace (a programování) všem žákům a studentům.

6. **Programování:** V informatice je programování velké téma, ze kterého má spousta lidí úplně zbytečné obavy. Dotýkáme se jej napříč všemi moduly a samozřejmě, že se mu budeme věnovat i v samostatném bloku.
   Řekneme si, jaké programovací jazyky jsou vhodné pro výuku programování na různých typech škol, nebo jak vůbec s jeho výukou začít.
   Seznámíme se základy programování ve vybraném jazyce vhodném pro určitou věkovou skupinu (ZŠ: Scratch, SŠ: JavaScript). Vysvětlíme si, na jakých příkladech problematiku efektivně vysvětlit a jak informace předat, ne jen jak se programuje. Hlavně chceme učitelům ukázat, jak programování učit.

3 Výukové moduly

V následující části se budeme věnovat jednotlivým výukovým modulům.
Výuka je koncipována jako průnik několika výukových stylů, od frontální výuky, přes práci ve skupinách až po projektovou výuku. Ve všech částech je dostatek prostoru pro diskusi a výměnu zkušeností jak mezi lektory a účastníky, tak mezi učiteli navzájem. Dále je výuka samozřejmě doplněna o řadu samostatných cvičení a podkladů pro samostudium.

3.1 Úvod do světa IT

Osnova tohoto modulu je inspirována obsahem kurzu *Poznej svět IT*, který pro potřeby Czechitas navrhl Vladimír Schreiner. Pro potřeby *Učitelské akademie* a této diplomové práce byla však zachována pouze osnova a současný obsah byl vytvořen mnou s využitím uvedených zdrojů a literatury.

Jak již bylo v předchozí části práce uvedeno, cílem tohoto modulu je posílit rozhled učitele a jeho základní orientace ve světě IT. Vedlejším cílem je vytvořit si slovníček pojmů, kterým budou účastníci rozumět. Odnést si představu o tom, co v IT je pro každého zajímavé a kterým směrem se dále orientovat.
Forma výuky: frontální výuka, diskuse.
Potřebný materiál: odvíjí se od požadované podoby slovníčku – chceme-li papírový, pak pomůcky pro jeho výrobu. Upřednostníme-li elektronickou verzi, pak pro každého účastníka chytrý telefon/tablet/počítač.
Zdroje pro samostudium: Kariéra v IT – https://opracivit.czechitas.cz/

Proč je dokola tolik IT?

Lidé, kteří mají peníze věří, že oblast IT je vhodným místem pro jejich znásobení, jelikož IT pozice jsou dobře finančně hodnocené. Informační technologie mění naše chování a svět kolem nás: nakupování v obchodech vs. služby jako Rohlík.cz, papírový autoatlas vs. GPS navigace, nákup CD vs. služby jako Spotify, platba složenkou vs. internetové bankovnictví.

Diskuse: Jaké známe IT produkty? Jaké jsou mezi nimi rozdíly?
Zde se jedná o téma na rozpovídání a uvolnění skupiny, společné sestavení žebříčku oblíbených aplikací. Může vypadat následovně:

- Webové aplikace (Czechitas.cz).
- Mobilní aplikace (Messenger).
- Desktopové aplikace (MS Word).
- Hry (Solitaire).
- Informační systémy (IS STAG).

- Hardware (GPS v autě).

Pojmy na IT trhu + kdo to vlastně dělá?
Je dobré, než půjdeme na pohovor, zjistit si, co daná firma dělá. Následná aktivita je napojena na úvodní diskusi, jedná se o krátkou skupinovou práci:
Aktivita: vyhledejte na internetu následující pojmy a ke každému uveďte příklad firmy, která se problematikou zabývá – ERP, CRM, CMS, Business Intelligence, E-Banking. Výsledný seznam může vypadat následovně:

- ERP (peníze, lidé, schvalování, majetek. SAP, NetSuite).
- CRM (zákazníci. SAP, NetSuite).
- CMS (obsah – web, e-shopy. Kentico).
- Business intelligence (analýza dat. GoodData).
- E-Banking (internetové bankovnictví, EmbedIt).

Start IT projektu – přípravná fáze.
Má různé formy:

- Funkční firma zavádí novou službu nebo produkt (Czechitas – web s možností přihlašování na kurzy, Apple – Apple Pay v ČR).
- Startup – nová firma, buď uspěje, nebo ne (Kiwi – sběr a kombinování nabídek různých leteckých dopravců, AVG – antivirový program).
- Velké firmy a korporace (možnost najmutí si malé firmy na rozjetí nového nápadu – Zonky).
- Pomalé zavádění služeb státem (EET).

Ještě než projekt začne, často již běží přípravná fáze. Cílem je rozpracovat nápad a pochopit problém. Také je třeba si zodpovědět několik základních otázek: Co to přinese a vydělá? Jak dlouho to bude trvat? Kolik to bude stát? Půjde to udělat?
V tuto chvíli se jedná o více obchodní téma, nežli technické a je třeba zde angažovat nejzkušenější lidi. Realizovat špatný nápad je totiž překvapivě jednoduché, ale velmi drahé. Čím více je projekt inovativní, tím větší je riziko neúspěchu (přibližně 90% startupů zkrachuje).
Co to přinese a vydělá? Je třeba použít data. Vše řádně propočítat, nasimulovat, stavět na faktech (Co dělají ostatní? Proč od nás odcházejí současní zákazníci?). Toto neplatí jen o IT firmách, datová analytika má své důležité místo například v marketingu.
K čemu je nám přípravná fáze? Dává nám představu o všem – požadavky na realizaci, časování, obchodní plán i případná rizika.

Řízení

Jak koordinovat jednotlivé aktivity? Rozlišujeme dva způsoby organizace:

1. Vodopád

   - Plán je pevně stanovený. Aktivity jdou jedna za druhou (analýza $\longrightarrow$ požadavky, návrh $\longrightarrow$ architektura, vývoj $\longrightarrow$ aplikace).
     Organizačně se jedná o jednodušší způsob (plán/cena/smlouva), lidské obsazení se dobře plánuje, jelikož se ví, kdy je kdo potřeba. Problémy se zjišťují až na konci, kdy je nejpracnější je opravit. Vymyslí se něco, co vypadá na papíře pěkně, naprogramuje se to a je to špatně.
     Proč se na začátku hodně věcí neví?
     - Zadání se mění (a upřesňuje) za pochodu a často jsou jednotlivé verze protichůdné. Prolínají se různé pohledy a priority mají pro různé lidi různá uspořádání.
   - Jedná se o oblíbený přístup ve státní správě – je nastavena pevná cena, pevné termíny, konzervativní projekty a výsledky často neodpovídají očekávání.

2. Agilní vývoj

   - Není důležité naprogramovat to, co jsme si mysleli, že bude fungovat. Nesnažme se vymyslet všechno předem. Namísto plánů a dokumentů průběžně tvoříme kvalitní software, postupuje se po malých krocích. Iterace obsahuje všechny aktivity a jsou v ní zapojeni programátoři, zákazník, návrháři, ...
   - Po dokončení iterace se výsledek testuje. Buď se spouští další iterace, nebo ukončujeme. V další iteraci se řeší aktuální nejpalčivější problém.
   - Největší výhodou je řešení problémů včas se zapojením relevantních lidí. Organizačně je to však náročnější – řešení vyžaduje neustálé zapojení všech stran. Délka řešení ani cena se nedají říct dopředu. Postupuje se, dokud všichni nesouhlasí.

Diskuse: Co je pro vás důležité? Jak by vypadala stavba rodinného domu agilně vs. vodopádově?

Ve spojitosti s řízením projektů je velmi důležitá role *projektového manažera*. Jedná se o člověka, který je zodpovědný za úspěšnou realizaci projektu. Jeho úkolem je propojit všechny strany, podílející se na tvorbě projektu, dohromady. Nastavuje a vynucuje tzv. pravidla hry – na začátku řešení projektu panuje mnoho nejasností. Projektový manažer má za úkol rozplánovat práci a zkoordinovat jednotlivé spolupracovníky. Hlídá rozsah realizace a nastavuje způsob, jak řízeně dělat změny.

Analýza

Co se má udělat? Chceme se dozvědět co nejvíce o problému, který řešíme. Zmapovat stav a požadavky (co se od nás chce a očekává).
Mezi zúčastněnými nesmí chybět následující role: analytik, vlastník produktu (product owner, formuluje, jak bude nová věc vypadat), odborníci na danou oblast (fakturanti, účetní pro účetní software), zástupci uživatelů (testování, průzkumy).
Nový produkt je třeba řádně specifikovat, popsat jeho funkce i systém. Je důležité, aby mu rozuměli všichni zúčastnění.
Mezi využívanými technikami při analýze produktu se objevuje rozhovor, uživatelské testování a nejrůznější průzkumy.
Podíváme-li se na pozici analytika – jaký by měl být? Měl by být schopen systémově přemýšlet i aktivně naslouchat. Musí disponovat silnými komunikačními schopnostmi a schopností srozumitelně vysvětlovat a předávat informace.

Architektura, návrh

V této oblasti se seznámíme se základními principy konstrukce softwaru.

- **Problém:** Rozdělení zodpovědností (dekompozice) $\longrightarrow$ vrstvy („multi-tier architecture“, neboli vícebrstvá architektura označuje aplikace, jejichž funkčnost netvoří jeden celistvý program, ale více vzájemně spolupracujících vrstev, které běží zpravidla na různé výpočetní infrastruktuře) [7].
  - přístup „Rozděl a panuj“ – každá vrstva dělá něco jiného.
  - nejčastější rozdělení na Front-End (dále FE, abstrakce, která poskytuje přívětivé uživatelské rozhraní) a Back-End (dále BE, samotná výpočetní logika, která se pod tímto rozhraním skrývá) [8].
- **Aplikační logika:** Softwarové služby, rozhraní pro programování aplikací (Application Programming Interface, dále API).
- **Data a databáze:**
  - Co se stane, když „odkliknete“ objednávku v eshopu?
    BE ověří dostupnost, případně kredit a objednávku uloží (data jsou dohledatelná např. logistickým oddělením). Data jsou uložena do tzv. *databázové tabulky*, ve které se orientujeme, dotazujeme, případně s ní manipulujeme pomocí jazyka SQL.
    Aktivita: Seznámení se se základními SQL dotazy – select, join, ... a ukázka, jak práce s databází probíhá.

Implementace

V tomto bloku je cílem přiblížit posluchačům, jak programátor komunikuje s počítačem. Na světě existuje velmi mnoho programovacích jazyků, které nám umož-

ňují komunikovat s počítačem a zadávat mu jednotlivé příkazy. Programovat znamená psát nějaký „návod“, pro naši první představu to může být třeba recept. Je rozdíl, jestliže je v postupu receptu napsáno *nafiletuj rybu*, nebo podrobně popsán každý krok, který vede pokaždé ke stejnému výsledku.

Na tomto místě si ukážeme pouze krátký přehled nejpoužívanějších jazyků, detailněji se budeme programováním zabývat v modulu *Programování* (3.6).

- **Java, C#**: Java i C# jsou univerzální a velmi rozšířené programovací jazyky. Programy, které jsou psány v Javě/C# běží jak na Windows, MacOS, tak i na mobilu nebo v ledničce. Oba jazyky mají obrovskou základnu uživatelů a proto není velký problém, když se při programování na něčem „zaseknete“. Existují totiž fóra, jako například *StackOverFlow*, na kterých je spousta programátorů, kteří buď řešili obdobný problém, nebo jsou ochotni pomoci. Existuje v nich obrovské množství nástrojů. Nevýhodou je to, že nejsou snadné na naučení, tedy se nejedná o ideální jazyky pro začátky s programováním.

- **JavaScript**: Ač se název může zdát povědomý, JavaScript a Java nejsou stejné jazyky. Jedná se o jazyk využívaný hlavně pro uživatelská rozhraní. Jeho velký rozkvět následoval poté, co se webové stránky rozhýbaly a začaly být rychlejší. Jedná se o jazyk, který se velmi rychle mění.

- **HTML, CSS**: Díky nim jsme mohli programovat webové stránky i před existencí JavaScriptu. Podobu dokumentu, který jsme schopni vytvořit v editoru Word (text, barvy, odsazení), naprogramujeme s využitím HTML, CSS. JavaScript nám k tomuto dodá interakci a efekty.

- **Python**: Jedná se o univerzální a na naučení/používání velmi jednoduchý jazyk. Speciálně je vhodné jej využívat v oblasti datové analýzy. Zároveň v Pythonu lze programovat BE k jednoduchým aplikacím. Zkrátka, naučíte-li se Python, snadno pochopíte další jazyky. Nevýhodou je to, že je pomalejší.

- **C/C++**: Univerzální jazyky, které jsou při vhodném užití velmi rychlé, ale velmi složité.

Shrnutí – kterým jazykem mám začít? Jestliže chcete dělat webové stránky, začněte rozhodně HTML. Poté, až je budete chtít rozhýbat, přichází na řadu JavaScript. Pokud chceme přidat BE a umět univerzální jazyk, je vhodnou volbou Python. Pro psaní velkých aplikací se naučte Javu nebo C#.

Programátorské vychytávky

Asi každý při výkonu práce využívá nějaké nástroje, pomůcky a zlepšováky, které mu výkon usnadňují či zlepšují, Jinak tomu není ani u programátorů.

- **IDE**: Jedná se o pracovní (vývojové) prostředí pro programování (Integrated Development Environment). Různé části kódů umí zvýraznit či barevně odlišit, našeptává či ukazuje dokumentaci. Současně také kontroluje kód a umožňuje nám tzv. debugging (proces hledání a odstraňování chyb v programu). Umí program pro otestování rychle spustit a tímto vším programátorům hodně pomáhá.
- **GIT**: Verzovací systém, který umožňuje více lidem pracovat na jednom kódu, aniž by si vzájemně rušili či měnili své verze. Jedná se o sdílenou databázi, do které všichni přispívají. Než do ní přispěji, stáhnu si poslední verzi a ujistím se, že je vše v pořádku a vše pasuje. Pomáhá mi spojit mé změny se společnou verzí.

3.2 Metodika výuky

Obsah tohoto modulu je inspirován aktivitami realizovanými v předmětu *Teaching Lab*, který je pod vedením Ondřeje Přibyly otevírán na Fakultě informatiky Masarykovy univerzity v Brně (FI MUNI) a je určen pro učitele a cvičící jako soubor inspirace pro jejich přednášky a cvičení. Pro potřeby *Učitelské akademie* a této diplomové práce byly využity dojmy z předmětu převzaté od jeho absolventů a současný obsah byl vytvořen mnou s využitím uvedených zdrojů a literatury.

Cílem modulu je ukázat kantorům, jak efektivněji učit. Doporučit aktivity vhodné do hodin i to, jak je důležitá podpora, soudržnost a zpětná vazba mezi kolegy. Dále si ujasnit pojem učení, nastavit formu sebereflexe. Na ukázce výuky si vysvětlit, které postupy jsou lepší a které méně vhodné. V rámci tohoto bloku se budeme bavit i o normách, jak je zavádět, nastavovat či upravovat a jaká je jejich úloha ve výuce.
Tento modul je oproti ostatním formulován jinou formou. Jedná se o 8 předpřipravených hodin, podle kterých je možno vést seminář pro učitele od začátku do konce. Téměř každá hodina obsahuje nějakou úvodní aktivitu, která reflektuje hodinu zúčastněných učitelů z předchozích týdnů a pomáhá jim uvědomit si, co dělali dobře, či co by šlo vylepšit. Dále je obsahem těchto předpřipravených bloků spousta aktivit, které pomáhají zlepšit atmosféru ve výuce či přístup ke správnému zadávání práce či způsobu kladení otázek.
Forma výuky: frontální výuka, samostatná práce, skupinová práce, praktické ukázky, aktivity, projektová výuka.
Potřebný materiál: míček, učitelský deník pro všechny účastníky, počítač, projektor, flipchart/tabule a fixy.

Reflexe, normy ve výuce

Na začátku většiny výukových bloků v tomto modulu je, jak již bylo zmíněno, umístěna nějaká tzv. *otevírací aktivita*. Jedná se o krátkou hru, nebo diskusi, během které se učitelé seznamují, lépe poznávají, sdělují si zkušenosti z praxe a vzájemně se podporují. První otevírací aktivitou je *Seznamovací kolečko*. Účastníci sedí v kruhu, lektor má u sebe míček, řekne o sobě pár vět. Dále, kde a co učí a co jej v poslední době zaujalo. Míček posílá po kruhu dál a postupně se představí všichni zúčastnění. Po dokončení kolečka si všichni vymění místa, aby si do budoucna nespojovali jméno osoby s místem kde sedí, ale s jejím obličejem.

Jakými jsme učiteli?

Učení je jako dramaturgie[3]. Učíme tak, jak jsme byli učeni. Nejvíce napodobujeme to, co na nás nejvíce emocionálně zapůsobilo (většinou jsou to negativní vzpomínky, ale není to nutnost).

[3]umělecká činnost zaměřená na přípravu repertoáru a jeho částí [9]

Diskuse: Co z vyučovacího stylu našich učitelů se nám líbilo a co ne? Co fungovalo a co ne? Co z toho chceme taky dělat?
Během diskuse na tato témata zpravidla vyvstane několik bodů, které stojí za to si poznačit. Na základě průzkumu mezi pedagogy si dovoluji uvést několik příkladů:
Co ano?

- hodina je soutěživá – učitel zadává úkoly, které se žáci snaží vyřešit jako první.
- hodina probíhá formou scénky – je protkána vyprávěním příběhů či anekdot.
- dotazování žáků a následná práce s odpověďmi.
- vyučující je nadšený do obsahu hodiny.
- interakce s konkrétními jednotlivci.

Co naopak ne?

- nedodržování stanovených cílů.
- „death by powerpoint“ – slajdy plné textu, které vyučující pouze předčítá.
- učení faktů bez souvislostí.
- situace „Učitel se zeptá, studenti neodpoví a učitel tedy neposkytne odpověď“.
- nesmysluplnost, nesouvislost s předmětem.
- zesměšňování studentů za chybnou odpověď.

Reflexe

Je důležité reflektovat vlastní výuku. Proto účastníci obdrží pokyn, aby si vedli svůj „Učitelský deník“ [10], do kterého si budou po dobu běhu kurzu (ideálně i poté) psát poznámky týkající se výuky i přípravy na ni. Každý týden v deníku uvedou počet hodin, které strávili přípravou materiálů, úkolů či pokynů pro své žáky, rozmyslí si otázky, které chtějí žákům během hodiny položit. Dále nastíní strukturu jedné z odučených hodin (na časové ose, 0 – 90 minut), jaké jsou cíle této hodiny, proč chtějí své studenty učit zrovna toto. Tuto hodinu pak budou ve zbylém prostoru reflektovat (každý týden si mohou zvolit hodinu z jiné třídy). V deníku si zaznačí jak jsou s hodinou celkově spokojeni, jaký z ní mají pocit a co vidí na svých studentech. Dále si zkusí vyznačit k následujícím otázkám jeden až dva body:

1. Co se mi povedlo?
2. Co mohlo proběhnout lépe?

Na závěr v pár větách sami za sebe sepíší, jaká byla atmosféra na dané hodině, zda studenti všemu rozuměli atd.

Ukázka výuky: Formáty

V následující části výukového bloku je účastníkům demonstrována část výuky zaměřená na rychlost padajících objektů.

1. Mapování předchozích znalostí.
2. Uvedení do tématu a vyvolání zvědavosti.
3. Formulace otázky („Co se stane, když se soubor s příponou .jpg pokusím otevřít v poznámkovém bloku? “).
4. Prostor pro individuální zamyšlení.
5. První hlasování.
6. Diskuse v tříčlenných skupinkách (snažím se přesvědčit / změnit názor ostatních).
7. Druhé hlasování.
8. Veřejná debata, sepsání argumentů.
9. Třetí hlasování.
10. Experiment.
11. Dovysvětlení, předání dodatečných informací.

Rozbor ukázky:
Tím, že učitel ukáže, co by mohl udělat (spustit poznámkový blok, otevřít v něm místo textu obrázek a sledovat, jak pokus dopadne), ale neudělá to, vyvolá a udržuje napětí u posluchačů. Zároveň jsou díky skupinovým diskusím po prvním hlasování zapojeni do výuky všichni studenti. Hlasování přiměje k zamyšlení a utvoření názoru – výběru odpovědi („Učitel po nás chce, abychom přemýšleli, asi by nám neukázal výsledek zadarmo. Vymýšlení odpovědi – nějaká forma investice. Tu potom sledujeme, jelikož o ni nechceme přijít“). Je důležité při sepisování možností v rámci argumentace nedat žádným způsobem najevo, která z možností je správná. Dobré je také na závěr poskytnout jen jednoduché vysvětlení a dodat detailnější materiál k samostudiu. Uspoříme čas v hodině a zároveň mají studenti možnost pracovat svým vlastním tempem a do samostatně zvolené hloubky. Nechceme z nich mít pouze pasivní posluchače.

Normy

Jakmile se v rámci skupiny něco stane (vznikne *precedens*[4]), je velká šance, že se to stane znovu. Proto chceme podporovat a zesilovat žádané precedensy, aby se

[4]skutek, který slouží jako model pro něco podobného v budoucnosti [11]

z nich staly normy[5] a naopak tlumit nežádoucí precedensy, aby se, pokud možno, neopakovaly.
Mnoho precedensů se jednoduše stane a je potřeba na ně reagovat vhodně podle toho, jaká chcete, aby byla vaše výuka. Je však možnost je zavádět i cíleně (*Chci, aby se mě studenti vyptávali na otázky* – vyzvu skupinu, aby se mě zeptala na 3 dotazy, které zodpovím). Precedensy obvykle není nutné pojmenovávat.
Pokud řekneme všechna pravidla a normy, které chceme mít ve výuce, studenty zahltíme. Místo toho můžeme normami začít žít. Sice nejsou explicitně pojmenovány, ale implicitně se stávají součástí skupinové paměti a ovlivňují chování lidí.
Je potřeba formulovat a praktikovat – studenti si musejí zažít, že je v pořádku se ptát a chybovat, aby tomu uvěřili.

Diskuse: Co očekáváme od studentů? Od našich hodin? Co chceme aby studenti očekávali od nás?

- Prosby o pomoc (nemít strach z toho, že si nevím rady, i kdyby to mělo být až po skončení hodiny).
- Otázky od studentů (vybízet k dotazování, otázky nemusejí být jen dobré – TIP: nepokračovat ve výuce, dokud nepadnou alespoň 2 dotazy).
- Chodit včas a připraven.
- Hodina musí všem něco dát.
- Důvěra, autorita.

Diskuse: Co na hodinách nechceme?

- Chaos.
- Aby se student věnoval jiným aktivitám (mobil, web, ...).
- Vypadat neprofesionálně (neumět zapnout dataprojektor).
- Zapojení pouze skupiny stále stejných žáků.
- Vzbudit dojem, že chybovat je nežádoucí.

Příklady norem ve výuce:

- Dodržování času – když jednou přijdu pozdě, stane se to normou. Je třeba si stanovit, co s pozdě příchozími. Také je vhodné mít ujasněno, jak moc budu trestat nedodržování termínů odevzdání.
- Interakce s učitelem a ostatními – jak oslovuji své studenty? Mohou studenti klást otázky? A pokud ano, jakým způsobem? Je potřeba vymezit, zda se mají hlásit, skákat do řeči či se ptát na všechny dotazy na konci hodiny. Může student formou otázek ovlivnit obsah výuky?

[5] pravidla, která jsou ve skupině očekávána

- Návyky – budou studenti chodit k tabuli? Mám načrtnout algoritmus dříve, než začnu programovat? Sedí studenti stále na svých místech, nebo se během hodiny pohybují po prostoru?

Jak se lépe ptát?

Otevírací aktivitou druhého bloku jsou tzv. *diády*. Celá skupina stojí v kruhu se zavřenýma očima, každý před sebe natáhne jednu ruku a někoho se chytne. V takto vzniklých dvojicích pak probíhá aktivita.

- Reflexe některé z nedávno odučených hodin – vždy jeden z dvojice mluví souvisle o své hodině, druhý poslouchá a na nic se nedoptává. Účastník, který zrovna mluví, se snaží sdělovat co se povedlo i co se naopak nepovedlo. Zkrátka takové informace, na které by se jeho společník pravděpodobně chtěl zeptat. Poté mu jeho spolusedící převypráví, co mu přišlo nejdůležitější. Následně se role prohodí.

Diskuse: Kdo se dozvěděl něco nového? Pro koho bylo těžké říct, co se povedlo?

Hospitace

Je důležité se na průběh svých hodin umět podívat i jiným úhlem pohledu. Proto je dobré se o hodinách bavit s ostatními učiteli. Domluvte se s kolegou ze školy, aby se na vás přišel do hodiny podívat (samozřejmě po předchozí domluvě), pozoroval jak učíte, po hodině se s vámi sešel a poskytl vám zpětnou vazbu. Udělejte totéž pro některého z vašich kolegů.

Jak se lépe ptát?

Smyslem dotazování je podnítit samostatné přemýšlení studentů a zjistit informace, které jsou pro nás důležité.
Zamyslete se nad rozdílem v otázkách *Chcete začít názornou ukázkou nebo teorií?* a *Zvedněte ruku, kdo chce nejdříve vidět názornou ukázku.* Druhá otázka má konkrétnější zadání a jasnější způsob reakce.

Ukázka výuky: Zdrojový kód

Lektor promítne zdrojový kód, začne vysvětlovat jednotlivé řádky, záměrně odbočuje a zaplétá informace, na konci se zeptá „Chápete to? Můžeme jít dál?“

Diskuse: Proč jste se nepřihlásili, nebo nedali najevo, že chápeme či nechápeme?

Problémem je zejména to, že studenti neví, jak v takovýchto situacích reagovat. Nemohou přeci začít mluvit všichni najednou.
Dále také vyjadřování – Co je to „TO“? Řádky kódu, sémantika výkladu či použití programu? Co je to chápat? Musím být schopen kód napsat sám či někomu

vysvětlit, jak funguje?

Můžeme využít řadu pomocných nástrojů pro získávání odpovědí od studentů při pokládání tzv. *signaling question* (formulace KDO + PODMÍNKA + AKCE). Akce může vypadat následovně:

- Zvednutí ruky či vstanutí ze židle (rychlá a jednoduchá odpověď od všech současně, nelze aplikovat při větším množství možností).
- Teploměr – Zvedněte ruku tak vysoko či nízko, jak si myslíte, že jste připraveni na opakovací písemku? (odpověď všech najednou, zachycení různých možností).
- Co dělat, když se zapojují stále stejní studenti? Celou třídu můžeme rozdělit do skupinek, kde v každé bude alespoň jeden zástupce těch, kteří již látce rozumí. Ten pak svým kolegům ze skupiny problematiku vysvětlí.

Diskuse: Jakou otázku byste položili v těchto chvílích? (formulace konkrétních otázek ve tříčlenných skupinkách).

Ukázka výuky: Programování na čtyři způsoby

Potřebný materiál: Několik různých zdrojových kódů (počet dle počtu skupin).

Následující aktivity je možné obměňovat a kombinovat. Tyto variace mohou mít odlišné cíle (trénovat různé dovednosti) i předpoklady (existujících dovedností a znalostí). Po skončení aktivity přichází na řadu zpětná vazba formou diskuse obsahující následující body.

Diskuse: Co je pro každý typ úlohy typické? Čím se vzájemně liší? Který typ bývá studentům předkládán nejčastěji? Který typ by bylo třeba zařadit častěji?

1. **Napište program, který splňuje danou specifikaci:**
   Jedná se o tvůrčí činnost vyžadující kreativitu. Student začíná od nuly, nemá existující kód, o který by se mohl opřít (musí vymyslet architekturu, ale není svazován existujícím kódem a může si dělat věci po svém). Pro tento úkol je nutná znalost jazyka (syntaxe i standardní knihovny). Řešením procvičujeme programovací jazyk i schopnost výstavby programu.
2. **Zde je kód. Zjistěte a popište, co dělá:**
   Tato aktivita učí stručně formulovat myšlenky a pojmenovávat koncepty. Učí studenty číst cizí kód s porozuměním, což je do budoucna vede k tomu psát svůj vlastní kód srozumitelně. Je možné zadat škaredý kód, nebo naopak krásný a demonstrovat na něm elegantní řešení. Tato úloha může sloužit k seznámení s novými technikami nebo knihovnami, není zde nutná hluboká znalost syntaxe.

3. **Zde je kód. Upravte jej, aby splňoval danou specifikaci:**
   Stejně jako v předchozím úkolu, i zde se studenti učí pracovat s cizím kódem. Učí se přizpůsobit zavedené kultuře kódu (i když s ní vždy nemusí souznít nebo jim není sympatická), vystupuje z komfortní zóny, když nenavrhuje architekturu podle sebe.

   - Typ *Opravte kód:* zde je možné se poučit z cizích chyb, plnění úkolu vyžaduje smysl pro detail.
   - Typ *Přidejte funkcionalitu:* může vyžadovat znovupoužití existujícího kódu.

   Je možné udělat (3), aniž bych dokázal udělat (1).

4. **Zde je kód. Proveďte code review[6]:**
   Tato část úkolu trénuje pojmenování, argumentaci i stručné a výstižné vyjadřování. Trénuje vysokoúrovňový popis (zobecnění chyb, nedostatky v architektuře), současně zahrnuje i (2). Je možné se zde inspirovat dobrým kódem, při argumentaci může docházet k reflexi („Nedělám tutéž chybu taky?“) – toto může být součástí zadání. Jinou variantou této aktivity může být srovnání více kódů (výkon, vhodnost, kvalita).

Aktivity (1) – (4) mohou být propojeny v jednu velkou aktivitu.

1. Začneme rozdělením na několik skupin. Každá skupina dostane nějakou specifikaci, kterou má kód splňovat. V rámci skupiny je možno pracovat po jednotlivcích či dvojicích.

2. Každá skupina dostane kód jiné skupiny a bude jejich úkolem přijít na to, co dělá a jakým způsobem.

3. Poté musí kód upravit, aby odpovídal podobné specifikaci.

4. Každá skupina znovu dostane cizí upravený kód a udělá code review na originální i upravenou část. Revize se následně musejí dostat k původním autorům, aby měli k dispozici zpětnou vazbu.

Zakázka

Tento blok otevřeme opět aktivitou ve dvojicích, do kterých se tentokrát rozdělíme pomocí losování kartiček pexesa. Se svým partnerem se budeme bavit o proběhlých hodinách trošku jinak. Reflexe proběhne z pohledu studenta. Tento způsob přemýšlení o výuce je vhodné provozovat, jelikož studenti jsou to, oč tu běží především a to, jak hodina působí na ně, je přinejmenším stejně důležité jako to, jak hodina působí na učitele. V na slepo vzniklých dvojicích si účastníci

[6] Code review, neboli posouzení kódu, je proces systematického zkoumání počítačového zdrojového kódu. Účelem je najít programátorské chyby, které autor kódu přehlédl, čímž se zvýší kvalita software. [12]

vzájemně odpovídají na otázky: Co si student myslí, když přijde na hodinu? Vydrží dávat pozor celou dobu, po kterou něco vysvětluji? Chce se mu odpovídat na položené otázky?

Zakázka

Pomyslný *Duch vzdělání* nám zadal k vypracování zakázku. Jedná se o úkol, na kterém máme pracovat průběžně po celou dobu své učitelské kariéry. Jeho zadání obsahuje dva body:

1. Pracujte na tom, abyste ze svých hodin měli dobrý pocit a učení vás bavilo.
2. Pracujte na tom, aby vaše výuka byla pro studenty maximálně přínosná a studenti na ni vzpomínali jako na vynikající hodinu.

Samostatná práce: Samostatně se krátce zamyslete nad následujícími otázkami: *Jak tyto body naplnit? Chci je naplnit? Co pro mě znamenají? Naplňuji je už teď? Jaké jsou mé individuální cíle? Jaké jsou mezi body souvislosti?*
Po pěti minutách, kdy si každý sám rozmyslí odpovědi na otázky, se účastníci rozdělí do trojic a diskutují o svých odpovědích, o tom, co za svou kariéru postřehli ve spojitosti s danou tématikou či o tom, co mohou udělat pro to, aby byl Duch vzdělání se splněním zakázky spokojen.

Co dělá učitel během samostatné práce studentů?

Zatímco studenti řeší ve skupinkách zadaný úkol, odehrává se následující (je potřeba, aby se těmito body vyučující v průběhu celého kurzu a obzvlášť tuto hodinu opravdu řídil):

- Činnost studentů:
  - Při úvodním vysvětlování aktivity sledují pojmy a klíčová slova a nahrávají si je do aktivní paměti.
  - Poté, co začne práce ve skupinkách, si jejich členové vzájemně dovysvětlují principy, doptávají se, argumentují, mění své názory na řešení a samozřejmě přemýšlejí.
  - Ke konci aktivity o řešení diskutují vždy dvojice skupinek. Během této fáze se argumentuje, dochází ke změnám názoru a novým uvědoměním.
- Činnost učitele:
  - Na začátku vysvětlí aktivitu.
  - Během samostatné práce studentů obchází místnost a sleduje, jak skupinky postupují. Vyvažuje rychlost skupin, klade otázky, zadává další podúkoly a ověřuje, zda studenti chápou a dělají, co mají.
  - Aktivitu včas zastaví a vhodně ukončí, shrne vše podstatné.

Zpětná vazba

Potřebný materiál: předmět znázorňující „štafetu“, například plyšové zvíře

Cílem následující hodiny je mimo jiné ukázat si další metody práce se studenty – společný brainstorming, jehož výsledky se graficky znázorní tak, aby se z toho daly odvodit pro učitele cenné informace o pochopení látky.
V úvodu hodiny se účastníci opět rozdělí do dvojic. Tentokrát se budou bavit o tématu „učitelská moc“ a její použití v jejich hodinách. Jak moc jsme ovlivňovali výuku? Co jsme dělali, abychom toho dosáhli?

Myšlenková mapa

Následující aktivita je vykonávána formou brainstormingu, který je ale korigován vyučujícím. Učitel položí třídě otázku *Co vás napadne, když se řekne . . . ?* a spouští se debata. Studenti reagují, říkají pojmy, které je napadají. Vyučující všechny pojmy zapisuje na tabuli.
Další fází aktivity je propojování souvisejících pojmů mezi sebou, čímž ilustrujeme asociace studentů.
Proč je tato aktivita vhodná?
Můžeme se takto připravit na vysvětlování nového tématu. Získáváme cenné informace o tom, jak studenti pochopili předchozí látku (v případě, že na ni navazujeme) nebo jaký mají základ v novém tématu. Také tím nutíme studenty přemýšlet o dané problematice hlouběji (nutnost přemýšlet nad vazbami).
Počet hran v takto vzniklém asociačním grafu je dobrou měrou toho, jak studenti rozumí dané problematice.

Štafeta

Štafeta je výukový nástroj (v našem případě plyšák), který nám umožňuje držet přehlednou diskusi. Verbální aktivita studentů je podmíněna držením nějaké věci, která se předává mezi lidmi. Diskutující se pak nepřekřikují, je jasné kdo má slovo a kam mají ostatní upírat svou pozornost. Tím, že někomu předáme či hodíme štafetu, jej vyzýváme k mluvení.
Dalším benefitem užívání štafety je to, že většině lidí se lépe mluví, když drží v ruce něco, co mohou mačkat, otáčet, hrát si s tím a filtrovat nervozitu.

Diskuse (s užitím štafety): Co jste se dozvěděli v úvodní aktivitě? Jaký je váš názor na užívání učitelské moci a jaké s tím máte zkušenosti?

Zpětná vazba

U zpětné vazby rozlišujeme mezi *formativní* a *sumativní* zpětnou vazbou. Formativní zpětná vazba či hodnocení slouží hlavně studentovi, sumativní je pak formulované pro někoho dalšího jako výstup výsledků učení.
Příkladem sumativního hodnocení můžou být třeba známky na vysvědčení. Ty

	SUMATIVNÍ	FORMATIVNÍ
KDY	Na konci	V průběhu
ÚČEL	Měří výsledky učení	Podporuje učení
FORMA	Body, známky	Konzultace

Obrázek 2: Formativní a sumativní zpětná vazba – shrnutí

slouží jako informace pro rodiče či školu, na kterou se daný žák hlásí, v případě posledních ročníků. Samozřejmě se jedná o jistou formu informace i pro žáka, ale ten s výsledky v tuto chvíli již nemůže nic dělat.
Kdybychom se podívali na definice daných pojmů, můžeme říci, že smyslem sumativního hodnocení je „získat konečný přehled o dosahovaných výkonech nebo kvalitativně roztřídit celý posuzovaný soubor (žáků, pracovních výsledků apod.)“ [13]. Vedle toho, podstatou formativního hodnocení, je „časté, interaktivní hodnocení pokroku žáka v učení, porozumění učebním potřebám žáků a přizpůsobení výuky těmto potřebám“ [14].
Když si tyto poznatky shrneme, můžeme je zaznačit do jednoduché tabulky (viz obrázek 2), nebo si je sepsat následovně:
Formativní zpětná vazba je podporou procesu učení. Učitel dává možnost dělat chyby, které neovlivní hodnocení. Její formy mohou být například konzultace, párové aktivity či vzájemné opravování.
Sumativní zpětná vazba je hodnocení/kategorizace podle schopností (prošel/neprošel). Je zde snaha o objektivní měření výkonu/schopností.

Bloomova taxonomie

Potřebný materiál: 12 funkcí na samostatných lístečcích, přehledové infografiky k vlastnostem funkcí.

Aktivita: Vlastnosti funkcí

1. krok:
   Studenti dostanou název vlastnosti (surjektivní, injektivní, bijektivní) a příslušný obrázek odkazující z jedné množiny do druhé. Úkolem je správně je spárovat.

2. krok:
   Je rozdáno 12 funkcí a studenti je musí přiřadit ke správné vlastnosti (např. $y = x + 3$ pro $x \in \mathbb{R}$).

Obrázek 3: Grafické znázornění Bloomovy taxonomie [15]

Bloomova taxonomie [15]

Jedná se o teorii vzdělávacích cílů, kde pro znalost vyššího stupně je potřebné zvládat všechny nižší stupně (pro schopnost něco *analyzovat* je nutné to umět *použít*, což ale musíme nejdříve *pochopit* a pro pochopení je nezbytné *zapamatovat si*.

1. Vzpomenout si – Zapamatovat
   - Vybavit si fakta a základní pojmy.
   - Klíčová slovesa: definovat, vyjmenovat.
   - Např. vzpomenout si na definici relačních operátorů.
2. Pochopit – Porozumět
   - Vysvětlit myšlenky nebo pojmy.
   - Klíčová slovesa: zjistit příčinu.
   - Např. pochopit co dělají a jak fungují relační operátory.
3. Použít – Aplikovat
   - Aplikovat nabyté vědomosti v nových situacích.
   - Klíčová slovesa: vyřešit podle postupu, uvést příklad, převést do praxe.
   - Např. dokázat použít relační operátory v novém příkladu.
4. Analyzovat
   - Vidět spojitosti mezi pojmy.

- Klíčová slovesa: rozdělit na části.
- Např. znát pro každý operátor jeho klady a zápory.

5. Vyhodnotit – Hodnotit
    - Zdůvodnit postoj nebo rozhodnutí.
    - Klíčová slovesa: odborně kritizovat, argumentovat.
    - Např. rozhodnout, který operátor je nejvhodnější (nejefektivnější) pro řešení daného problému.
6. Vytvořit – Tvořit
    - Vyprodukovat nové/originální dílo.
    - Klíčová slovesa: navrhovat, komponovat, sumarizovat.
    - Např. vytvořit systém, který by implementoval jen nejvýhodnější operátory pro daný systém.

Diskuse: Co si odnášíte z Bloomovy taxonomie?

Struktura hodiny

Otevírací aktivitou tohoto bloku je společná diskuse v kruhu. Každý ze skupiny účastníků si vybere jednu pozitivní a jednu negativní zkušenost ze své učitelské praxe a sdělí ji ostatním.

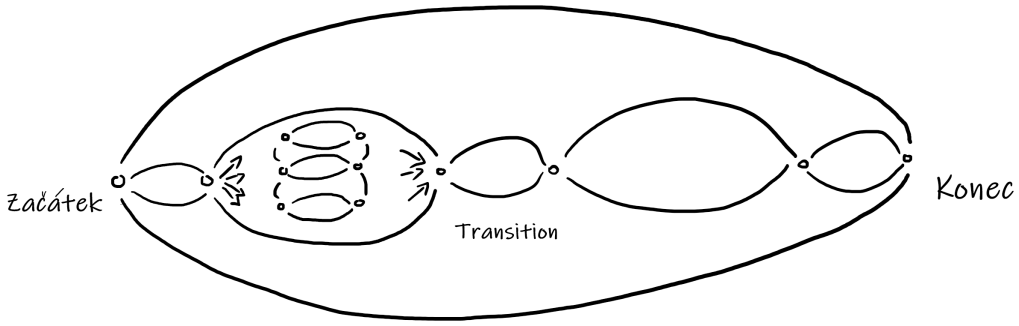
Strukturovaná hodina

Strukturovaná hodina je taková hodina, o které můžeme prohlásit, že student ví, co se děje, co je za námi, co je před námi a v jaké části probíraného tématu se nacházíme.
Je pro studenta i vyučujícího příjemnější, neboť nestrukturovaná hodina může na studenta působit zmatečně. Struktura není nutností, ale bez ní hodina vyžaduje mnohem větší režii ze strany kantora. Jak strukturovanou hodinu vytvořit?

Velká ryba (Big Fish)

Jedná se o koncept, který se snaží popsat jednotlivé kroky strukturované hodiny. Ke strukturování pomáhají čtyři základní praktiky – *situování, shrnování/uzavírání, sledování (tracking), přechod (transition)*. Každá je užitečná, ale ani jedna není jednoznačná. V některých situacích může dojít k jejich překryvu.

1. Situování
    - Říkám co děláme.
    - Říkám co bude.



Obrázek 4: Grafické znázornění Big Fish

- Říkám kontext.

2. Shrnování / Uzavírání
3. Tracking
    - Jak jsme se dostali tam kde jsme a co jsme nechali po cestě.
4. Transition
    - Přepnutí kontextu.
    - Pomáhají studentovi si uvědomit přechod na jinou činnost nebo změnu tempa práce.

Check list

Ukázka (viz obrázek 5) vhodného způsobu zaznamenání osnovy hodiny, průběžné značení toho, co se stihlo/nestihlo i rekapitulace na závěr.

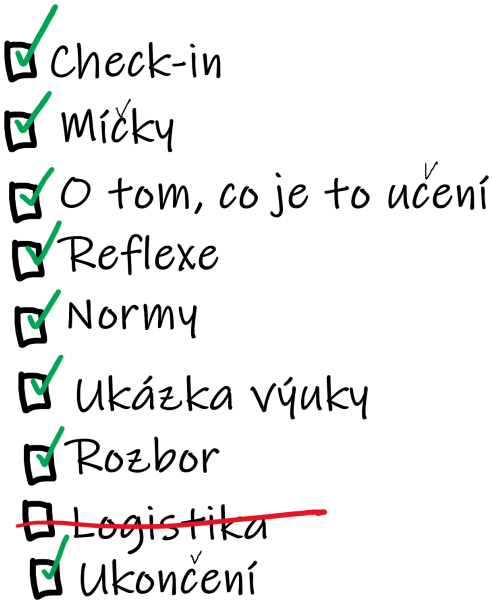
Parkoviště otázek

Způsob provedení je ponechán na dohodě mezi vyučujícím a studenty (např. využití slido[7] nebo psaní otázek na lístečky a následné losování). Umožňuje studentům přestat se trápit otázkou ve chvíli, kdy se mají soustředit na něco jiného a vyučující se mezitím může zamyslet nad odpovědí, případně na danou otázku nezapomene.

TPS (Think – Pair – Share) [16]

Typická je párová komunikace, při níž si studenti ve dvojicích vzájemně vymění názory. Výhodou je zapojení všech studentů, snížení nervozity z veřejného projevu, v prvním kroku je prostor pro promyšlení odpovědi i to, že si mohu vyslechnout jiný úhel pohledu (může vést ke zlepšení myšlenky).

[7] https://www.sli.do/



Obrázek 5: Struktura prvního výukového bloku znázorněna formou Check listu

MCQs (Multiple Choice Questions) [17]

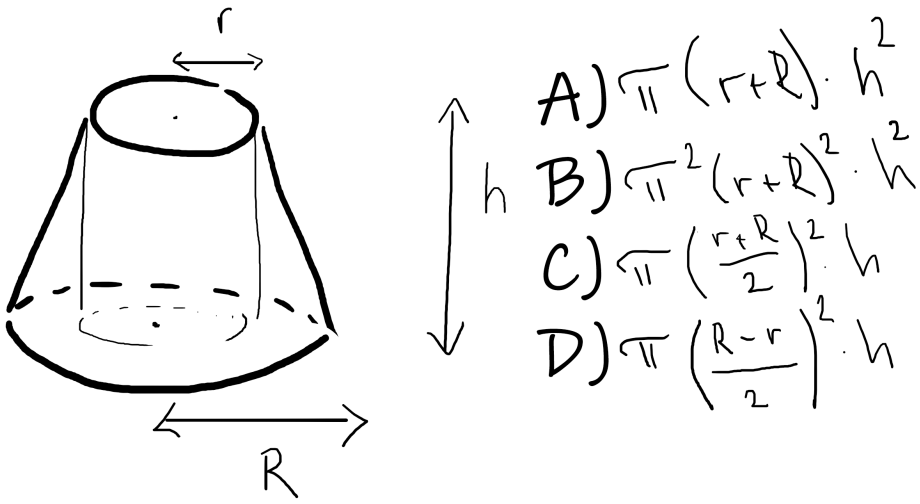
Početnou rodinu otázek s mnohočetným výběrem odpovědi (MCQ) můžeme rozdělit do dvou skupin položek. Do první patří ty, které od zkoušeného vyžadují vyznačit všechny vhodné odpovědi (ano/ne), do druhé pak položky, které od zkoušeného vyžadují, aby z nabídnutých odpovědí vybral jednu, nebo jiný předem stanovený počet odpovědí, které nejlépe odpovídají na otázku.

Obě techniky (TSP, MCQs) lze vzájemně kombinovat.

- nastíníme problém a studentům nabídneme 4 možnosti řešení. Necháme je hlasovat a poté ať si ve dvojicích prodiskutují, proč se rozhodli hlasovat pro danou možnost. Následně je můžeme nechat hlasovat znovu pro případ, že někdo změnil názor.
  Nakonec jednotlivé možnosti společně probereme a odhalíme tu správnou.
- Příklad aplikace postupu (viz obrázek 6): Hádejte, který z uvedených vzorců je nejblíže tomu, kterým lze spočítat objem „uříznutého“ (komolého) kužele.

Zadávání úkolů

Cílem posledního bloku tohoto výukového modulu je ujasnit si, jak NEzadávat úkoly a na co se u zadávání naopak soustředit. Kdo je to expert, kdo kouč a jaký



Obrázek 6: Příklad kombinace TPS + MCQs

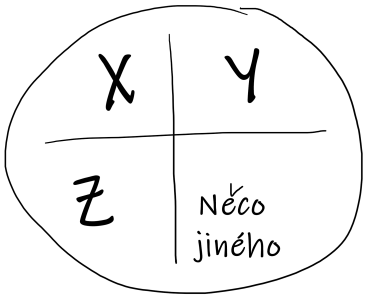
je rozdíl mezi jejich přístupy? Na základě společné aktivity chceme vybrat, jaká prostorová uspořádání jsou vhodná pro jaké činnosti.

Diskuse: Jak úkoly nezadávat? – východiska z diskuse sepsat

- Protichůdné zadání.
- „Toto zkoušet nebudeme, ale. . . “
- Změna zadání za běhu.
- Příliš dlouhé/složité zadání.
- Nepoužívat demotivující hlášky nebo výklad začínat slovy „To je primitivní“.
- Neúplné zadání, nevysvětlené pojmy.

Na co se u zadávání naopak soustředit?

- Výchozí bod.
- Výsledek (+ test výsledku).
- Forma odevzdání.
- Proces.
- Milníky.
- Účel (motivace).



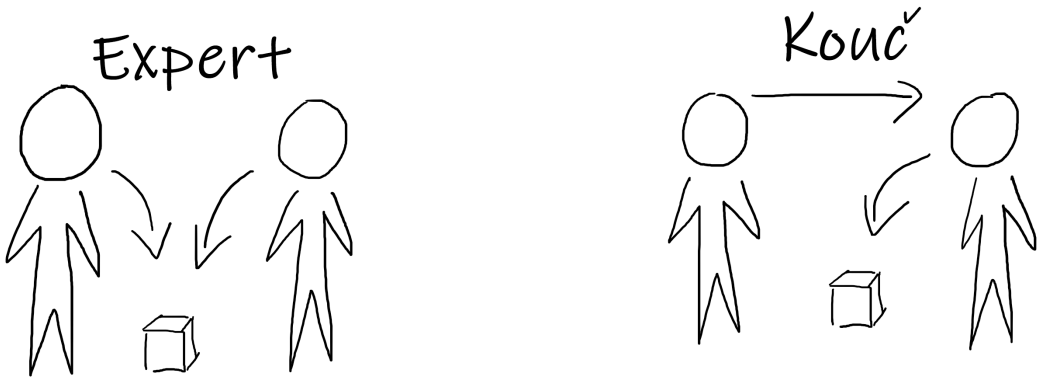
Obrázek 7: Redukce komplexity

Koučing (Coaching)

Jaké jsou rozdíly mezi přístupy experta a kouče?

1. Nástroje pro koučing:
    - Škálovací otázky
        - Jak moc máš jasno v tom to vyřešit na škále 1–10?
        - Co už máš, že jsi na X?
        - Co potřebuješ k tomu, abys byl na X+1?
    - „Představ si“ otázky:
        - Jak bude vypadat výsledek?
        - Jak to bude vypadat, když se to změní?
        - Kdo to pozná, že se něco změnilo?
        - Jakou změnu ten člověk uvidí?
    - Redukce komplexity:
        - Potřebuješ k řešení problému X, Y, Z nebo něco úplně jiného?
    - Časová sekvence:
        - Co potřebuješ k tomu, abys mohl udělat X?
        - Co se má stát, abys mohl udělat X?
    - Otázky na metakontext:
        - Co je na tom pro tebe důležité?

Musíme dávat pozor na způsob, jakým otázku pokládáme. Ptát se tak, abychom nezpochybnili toho druhého (např. nevhodně položená otázka: „Myslíš, že to dokážeš posoudit?“)



Obrázek 8: Přístup experta vs. přístup kouče

3.3 Digitální dovednosti

Původní obsah tohoto modulu byl z větší části pro účely Učitelské akademie vytvořen lektorem Czechitas Martinem Krčkem. Pro potřeby této diplomové práce byla však zachována pouze osnova a současný obsah byl vytvořen mnou s využitím uvedených zdrojů a literatury.

Cílem následujícího modulu je seznámit se s nástroji, které mohou učitelé efektivně využít nejen ve výuce, ale i jako pomoc a zjednodušení pro svou práci. Nikdy nevíme, kdy se nám do života postaví nějaká překážka a my na ni budeme muset okamžitě zareagovat například přesunem části výuky do online prostředí. K tomu nám skvěle mohou pomoci například „cloudová“ řešení námi využívaných služeb či tzv. *virtuální učebny*. Tomuto i jiným věcem se budeme věnovat právě v této části textu.
Forma výuky: Frontální výuka, samostatná práce, praktické ukázky, aktivity.
Potřebné pomůcky: notebook + myš.

Digitální online kancelář

V tomto bloku si představíme možnosti, které Google (případně Microsoft) poskytuje pro učitele, školu, neziskové organizace či firmy až po běžné uživatele.

Co je to ten Cloud?

Cloud computing (dále CC) by se dal vysvětlit jako doručování výpočetních služeb (serverů, úložišť, databází, sítí, softwaru, analytických nástrojů přes internet. Nabízí rychlejší inovace, flexibilitu prostředků a cenové výhody. Obvykle platíte jen za ty služby, které skutečně využíváte. [18]

Jaké jsou výhody CC?

- *Náklady* – CC eliminuje investiční náklady na nákup HW a SW či vytvoření a provoz místních datových center (stojany se servery, zdroje napájení).
- *Spolehlivost* – CC usnadňuje a snižuje náklady na zálohování dat, zotavení po případné havárii a zajištění provozní kontinuity.
- *Zabezpečení* – Řada poskytovatelů cloudu nabízí širokou škálu zásad, technologií a kontrolních prvků, které celkově posilují stav zabezpečení a tím pomáhají chránit data a aplikace před potenciálními hrozbami.

Jak se liší digitální online kancelář od té klasické?

V práci nebo doma máte nainstalovány programy a uložené dokumenty. Když nejste doma, nemáte k nim přístup. Když si je chcete vzít s sebou, musíte je nejdříve uložit na externí datové úložiště (např. flash disk) a na jiném počítači

použít.
V případě online kanceláře je vše uloženo online v cloudu, v bezpečném prostředí. A to nejen vlastní dokumenty, ale i nástroje, které nám umožňují je upravovat. A navíc, existuje jen jedna verze dokumentu – ta poslední. Na mobilu, tabletu, notebooku i PC.

Co nám nabízí Google?

Balíček G Suite (analogicky Microsoft – Office 365) [19]:

- Oblast SPOJENÍ
  - hlavní výhodou je to, že se vše nachází na jednom místě, vše je vždy dohledatelné a to včetně historie.
  - *Gmail* – Zabezpečený a soukromý email bez reklam, možnost zřízení adresy pro společnost ve vlastní doméně (tereza@czechitas.cz), skupinové emailové konference, možnost čtení i psaní zpráv v offline režimu (jsou odeslány ihned po opětovném připojení).
  - *Kalendář* – snadné sdílení a propojení s dalšími aplikacemi, chytré plánování schůzek (kontrola volného času kolegů, možnost zobrazení několika kalendářů zároveň – snadné plánování společné schůzky).
  - *Hangouts Chat* – přímé zprávy i skupinové komunikace, virtuální místnosti, možnost vyhledávání v předchozích konverzacích, integrace s kalendářem – snadné plánování schůzek.
  - *Hangouts Meet* – snadné připojení k pracovním videohovorům, možnost připojení i z jiného účtu (přes odkaz) nebo mimo internetové připojení (ke každé schůzce je přiřazeno telefonní číslo).
- Oblast PŘÍSTUP
  - máme k dispozici obrovské úložiště pro všechny typy dokumentů a souborů. Má skvělé možnosti prohledávání (včetně obrázků a PDF) i uvnitř dokumentů („Hledám zápis z porady, na které jsme řešili, že budeme hledat nového školníka.)“.
  - *Drive* – uschování, používání, sdílení i vytváření souborů na jednom zabezpečeném místě, využití technologie umělé inteligence k předvídání a zjišťování, co je pro uživatele důležité – rychlý přístup ...
    Pomocí sdílených disků uchovávání práce týmu na jednom místě – vlastníkem je celý tým, neustálý přístup k aktuálním verzím.
  - *Cloud Search* – vyhledávání v Gmailu, Drivu, dokumentech, tabulkách, prezentacích, kalendáři a dalších službách.
- Oblast VYTVÁŘENÍ

  - obdobné nástroje jako Word, Excel či PowerPoint, jen online a vždy k dispozici. Umožňují společné tvoření klidně i dvaceti osob. Pomocí formulářů můžeme pořádat různé ankety či průzkumy s okamžitým vyhodnocením, do nástroje Fotky pak ukládáme všechny své fotografie. I když je vyfotím telefonem, okamžitě k nim mám přístup z počítače.
  - *Dokumenty* – textový procesor pro týmy, dává nám možnost vytvářet a okamžitě upravovat dokumenty přímo v prohlížeči, není třeba instalace žádného specializovaného softwaru. Více lidí může pracovat současně, každá změna se průběžně ukládá. Také je zde možnost vkládat komentáře či vidět změny, které byly provedeny. Kdykoliv se můžeme vrátit ke starší verzi, otevírat či upravovat importované dokumenty (Word, PDF) a následně je exportovat do různých formátů (docx, pdf, odt, rtf, txt, html).
  - *Tabulky* – chytré a zabezpečené tabulky pro spolupráci v dynamické organizaci, spolupráce s kýmkoli, kdykoli a kdekoli. Navíc kompatibilita s MS Office.
  - *Prezentace* – vytváření úhledných prezentací přímo v prohlížeči bez potřeby jiného softwaru, možnost současné práce několika lidí, k dispozici spousta připravených šablon, možnost vkládat videa, obrázky, kresby i animované přechody.
  - *Formuláře* – vytváření formulářů pro průzkumy a dotazníky, možnost sestavování *písemek a testů*, export výsledných dat do tabulek, několik typů otázek, možnost měnit pořadí.
  - *Keep* – zaznamenávání inspirace a úkolů, spolupráce na poznámkách se členy týmu, nastavování připomenutí.
- Oblast KONTROLA
  - jedná se o oblast pro administrátory – nemusím mít externí firmu, sám (se základní znalostí IT) si přidávám další uživatele.
  - *Administrace* – snadné přidávání uživatelů, konfigurace bezpečnostních a jiných nastavení.
  - *Sejf* – uchovávání dat a vyhledávání informací v elektronických dokumentech.
  - *Mobil* – správa mobilních zařízení je zahrnuta v ceně služby G Suite/Cloud Identity
- Google CLASSROOM
  - jedná se o řešení pro školu – prostor pro vypsané kurzy a podklady pro studenty, místo pro odevzdávání domácích úloh.

– přiřazování domácích úloh, podkladů pro učení, testů (při psaní testu neumožňuje stránku opustit, tedy ani vyhledávat odpovědi na internetu, ale funguje pouze na chromebooku)

Největší výhodou celého balíku je rozhodně vyhledávání napříč aplikacemi a programy. Google vyhledávač využíváme k vyhledávání čehokoliv snad nejčastěji, proto jej budeme používat i tady.

Praktická cvičení – G Suite

Společně se podíváme na používání Gmailu. Ukážeme si jeho základní vlastnosti, jeho ovládání, kde najdeme jaké funkce a také si vyzkoušíme pár zajímavých vlastností, které Gmail má a mohly by se vám hodit.
Vše si prakticky vyzkoušíme na plánování školního výletu. Společně vyplníme online tabulku a na tomto příkladu si demonstrujeme, jak může být takový sdílený dokument užitečný a kolik práce nám může ušetřit.

Diskuse: Kolik z vás používá Gmail?

Zobrazení a odpověď na email

Jednoduchým kliknutím na řádek emailů se dostaneme na detail. Po přečtení emailu máme tři možnosti co dál.

1. Odpovědět odesílateli.
2. Odpovědět všem, kteří jsou uvedeni jako Odesílatel, ale i jako příjemce – a tím všechny informovat.
3. Přeposlat zprávu někomu dalšímu.

Když na email začnu odpovídat, nemusím mít strach, že se mi v průběhu smaže – průběžně se mi ukládá do konceptů. Můžu jej tedy dopisovat průběžně celý den. Kliknutím na *Odeslat* se email okamžitě odešle. Ale mohu nastavit i odeslání na pozdější dobu. Například, pracuji-li o víkendu, je vhodné odeslání nastavit na pondělních 8 hodin ráno. Stisknutím ikony popelnice vpravo email smažeme.

Hledání

Možnosti vyhledávání je to, co dělá Gmail jedinečným. Kromě předmětu zprávy mohu hledat i podle dalších parametrů, jako je Odesílatel, Datum odeslání emailu či časové rozmezí („bylo to někdy okolo Velikonoc“). Samozřejmě mohu hledat i podle slov, které zpráva obsahuje (například „školník“).

Hledání a štítky

Abych měla v emailech přehled, mohu je „oštítkovat“, tedy něco jako nalepit na ně POST–IT papírek. Poštu si pak mohu díky nastavení takovéhoto filtrování přehledně rozdělit do jednotlivých skupin (Matematika, Školení, Omluvenky, ...). Místo *Vyhledat* použiji při zadávání parametrů *Vytvořit filtr*. Od té doby se budou všechny emaily, splňující kritéria, samy označovat štítkem.

Archivace

Stalo se vám někdy, že jste hledali opravdu starý email nebo dokument? Nebo papír na stole, ale jste si skoro jisti, že jste ho před rokem vyhodili do koše? Proto Gmail nevyhazuje zprávy do koše, ale archivuje je.

Kancelářský balík:

Kromě Gmailu máme k dispozici celý kancelářský balík, který obsahuje *Dokumenty, Tabulky, Prezentace, ...* Kde je najdeme? V pravém horním rohu v prostředí Gmailu se nachází menu schované za ikonou „rubikovy kostky“.

Dokumenty

Pojďme si společně vyzkoušet *Tabulky* na již zmíněném příkladu – plánování školního výletu.
Jak to děláte dnes? Pravděpodobně získáte informace skrz mail/žákovskou knížku/formulář, nebo hůř, pošlete tabulku všem, aby ji vyplnili a vy to ručně přepíšete.
Nástroje balíčku G Suite nám umožňují vše vyplnit najednou a společně. Odkaz vám mohu buď poslat emailem (kde je nevýhoda toho, že odkaz zůstává navždy stejný, můžete se tedy do dokumentu dostat i za měsíc), nebo vám jej nasdílím jiným způsobem (proklik v prezentaci). Já si předem mohu nastavit, jaká práva v dokumentu vám udělím a jakým způsobem do něj můžete nahlížet, komentovat nebo měnit jeho obsah.
Každý, kdo má dokument otevřený, má automaticky přidělenou barvu a avatara (v případě propojení přes Gmail nejste anonymní a máte zobrazenou fotku, kterou máte ve svém profilu) v pravém horním rohu (v kolečku ohraničeném danou barvou.
Nyní si vyberte řádek a vyplňte informace do tabulky. U toho můžete sledovat, jak současně s vámi vyplňují ostatní. Kromě historie změn, která se neustále ukládá, můžete neustále přepisovat informace ostatních. V historii pak uvidíme, kdo v tabulce něco změnil a v případě potřeby se ke starší verzi můžeme vrátit.
Jak již bylo zmíněno, při sdílení dokumentu si jako vlastník můžete nastavit práva, s jakými dokument sdílíte mezi ostatní „přispívající“. Možnosti jsou následující:

1. Všichni editují/zobrazují/komentují.

2. Přístup pouze pro zvané.

Jednou ze součástí kancelářského balíku G Suite je služba *Formuláře*, která mimo využití pro tvorbu formulářů a následnému sběru dat, může být použita k výrobě písemek pro vaše žáky. Toho se bude týkat naše první samostatné cvičení.

Aktivita: Vytvořte pomocí Google Forms elektronickou písemku, nechte ji napsat své žáky, výsledky vygenerujte do tabulky a vyzkoušejte si, jak je pohodlné/nepohodlné jejich opravování.

Obecně mohu vytvářet dotazníky, formuláře či hlasování – online vidíme výsledky – výsledky nám „padají“ do tabulky, kterou mohu editovat.
Funkčnost formuláře si mohu samozřejmě sama ověřit ještě předtím, než jej rozešlu respondentům. Následně přes tlačítko *Odeslat* získám link, kterým se dostanu přímo k formuláři (ne jeho editovatelné podobě, ale do prostředí, kde respondenti odpovídají a ze kterého následně sbírám data).

Fotografie

Fotit je dnes extrémně jednoduché. Téměř každý má na svém mobilním telefonu fotoaparát výborné kvality, lepší, než svého času digitální fotoaparáty. Se službou *Fotografie* mohu své fotky okamžitě zálohovat, editovat, sdílet, vytvářet sdílená alba, do kterých mohou přispívat všichni oslovení spolupracovníci/spolužáci. Google vyhledávač už umí dokonce vyhledávat i ve fotografiích.
Fotoaparáty umí fotit i téměř ve tmě a nebo dělat nádherné portréty (s rozmazaným pozadím). To vše je výsledkem umělé inteligence. Ale umí toho ještě více – třeba *Google Lens*:

- Po namíření na rostlinu řekne, o jaký druh se jedná.
- Po namíření na produkt (např. kabelku) mi řekne, kolik kde stojí.
- Pro pochopení cizího textu stačí namířit a číst v češtině.

Google Classroom

Google učebna je služba, která je zdarma pro školy, neziskové organizace a všechny, kdo mají osobní účet Google. Usnadňuje propojení studentů s učiteli ve škole i mimo školu. Šetří čas a papír a můžete pomocí ní snadno vytvářet kurzy, zadávat úkoly, komunikovat a mít vše dobře zorganizované.
Používání učebny má spoustu výhod:

- *Snadné nastavení* – učitelé mohou studenty přidat přímo, nebo jim poslat kód, pomocí kterého se do kurzu zapíší sami. Nastavení trvá jen pár minut.
- *Úspora času* – díky jednoduchému postupu práce s úkoly, která probíhá čistě elektronicky, mohou učitelé úkoly na jednom místě vytvářet, opravovat i hodnotit.

- *Přehlednější organizace* – studenti vidí všechny své úkoly na stránce úkolů a všechny materiály kurzu (dokumenty, fotky, videa) jsou automaticky zařazeny do složek na *Drive*.
- *Lepší komunikace* – učebna učitelům umožňuje v mžiku poslat oznámení nebo zahájit diskusi. Studenti spolu mohou navzájem sdílet zdroje nebo ve streamu odpovídat na otázky.
- *Dostupnost a bezpečnost* – učebna, podobně jako všechny ostatní služby G Suite pro vzdělávání, neobsahuje reklamy, nevyužívá váš obsah ani data studentů k reklamním účelům a je zdarma.

Nezávisle na zařízení, všechny materiály a potřebné dokumenty máme neustále při sobě. Vše je dostupné a synchronizované přes Cloud. Tedy kromě toho, že jsou data kdykoliv a kdekoliv dostupná, jsou také dostatečně zabezpečená a dostupná z telefonu, počítače, nebo třeba tabletu.

Komunikace mezi lidmi a stroji

Lidé spolu chtějí komunikovat. Začalo to SMS zprávami a dnes jsou to zprávy online, chatBot, nebo se chtějí s něčím pochlubit většímu okruhu lidí – prostřednictvím sociálních sítí. Když si chci jen zahrát hru, už i tady je možnost chatovat a chlubit se.
Začalo to již zmíněnými SMS a s nástupem internetu se přidávaly další skupiny možností:

- ICQ (1996)
- Facebook Messenger, WhatsApp, Skype, Hangouts
- videohovory – titulky a překlady v reálném čase.

Některé z těchto služeb (Hangouts) mají tu výhodu, že se vše nachází v jednom prostředí.

Diskuse: Jak si myslíte, že je to s bezpečností?

End–to–end šifrování: Šifrování, při kterém je přenos dat zajištěn proti odposlechu správcem komunikačního kanálu i správcem serveru, přes který uživatelé komunikují. Další možností zabezpečení je například automatické mazání zpráv, či expirace po určité době (např. po dvou sekundách).
Chaty jsou dnes úplně všude. Měli bychom si uvědomit, že bezpečnost není pokaždé prioritou. Důležité je vědět, že je ve službě k dispozici chat – ne jen obrázky. Například oblíbená hra dětí (i rodičů) Minecraft – nejedná se pouze o hru jako např. Lego. Obsahuje chat, ve kterém si s hráčem může psát kdokoliv, ať už je to vrstevník nebo někdo, kdo se za něj vydává.
Stále více společností a stránek používá tzv. *ChatBota*. Jedná se o počítačový

program určený k automatizované komunikaci s lidmi. Nejčastější platformy využívající ChatBota jsou FB Messenger, Skype, Viber, WhatsApp, Slack, ...
V zákaznické sféře nahrazují operátory, lze s jejich pomocí nakoupit nějaké produkty či služby. Také nacházejí své uplatnění v oblasti HR (human resources – oddělení lidských zdrojů), kde jsou využíváni pro hledání a nábor zaměstnanců.

Sociální sítě

Jedná se o celosvětový, stále se rozvíjející fenomén. Sociální sítě patří mezi nejnavštěvovanější webové služby s miliony aktivních uživatelů, kteří si navzájem vyměňují data různého charakteru.

Diskuse: Jaké znáte sociální sítě?

Facebook

Facebook je již dlouhé roky číslo jedna. Proč?
Oblíbenost facebooku je již na ústupu, ale dlouho si držel (a stále drží) prvenství v tabulce oblíbenosti sociálních sítí. Je to díky velkému množství funkcí, multiplatformní podpoře a obrovské základně uživatelů, která nově příchozím umožňuje jednoduše navazovat kontakty či se spojit s přáteli z reálného života.
Jak si mohu lépe zabezpečit svůj účet?

1. Bezpečné heslo
   Pravidlo číslo jedna u všech internetových služeb vyžadujících přihlášení. Zásadám tvorby bezpečného hesla se budeme podrobněji věnovat v kapitole 3.4.

2. Dvoufázové ověření
   Kromě hesla by případný útočník potřeboval i váš telefon, aby se vám mohl nabourat do účtu.

3. Nastavení soukromí
   Je třeba rozlišovat, které informace sdílíte se všemi veřejně a které jen se svými přáteli.

4. Známí neznámí
   Přidáváním osob, které neznáte, mezi své přátele zvyšujete pravděpodobnost, že vámi sdílené údaje se dostanou k někomu, kdo jich může zneužít.

5. Dobrý sluha, špatný pán
   Nenechte sociální sítě, aby vám komplikovaly život. Každý uveřejněný příspěvek či komentář se může dostat ke komukoliv. Například k vašemu budoucímu (nebo současnému) zaměstnavateli.

Existují i profesní sociální sítě – největší je **LinkedIn**. Slouží k setkávání profesionálů a jejich diskusi o pracovních zájmech. Mezi její uživatele patří manažeři,

konzultanti, odborníci z různých oborů či celé firmy.
Nesmím zapomenout zmínit i největší internetový server pro sdílení videosouborů, který je sociální sítí i vyhledávačem zároveň. Jedná se samozřejmě o **YouTube**. Tzv. Youtubeři jsou fenoménem poslední doby, i když v poslední době jsou již nahrazováni influencery (Instagram) či musery (TikTok). Jedná se o tvůrce obsahu (videí), kteří kolem sebe budují komunitu fanoušků. V posledních několika letech se velmi rozmohlo jejich využití také v marketingu.

Diskuse: Co sledujete vy/vaše děti na Youtube?

Umělá inteligence v rukou učitele

Jak vnímáte a jak rozumíte následujícím pojmům?

1. *Umělá inteligence* [20]
   Systém nebo stroje, které napodobují lidskou inteligenci k plnění úkolů a mohou se iterativně vylepšovat na základě shromážděných informací. Vyskytuje se v několika podobách:

   (a) **Chatovací roboti** využívají umělou inteligenci k rychlejšímu uchopení problémů zákazníků a poskytování efektivnějších odpovědí.

   (b) **Inteligentní asistenti** využívají umělou inteligenci k analýze důležitých informací z velkých datových souborů s volným textem, aby zlepšili plánování.

2. *Strojové učení*
   Jedná se o podskupinu umělé inteligence, která se zaměřuje na vývojové systémy, které se učí, nebo zvyšují svou výkonnost, na základě dat, se kterými pracují. Umělá inteligence je široký pojem, který označuje systémy nebo stroje, které napodobují lidskou inteligenci.
   Strojové učení a umělá inteligence jsou často diskutovány společně a tyto pojmy jsou někdy používány zaměnitelně, ale neznamenají totéž. Důležitým rozdílem je to, že ačkoli veškeré strojové učení je umělou inteligencí, není veškerá umělá inteligence strojovým učením.
   v současnosti se strojové učení uplatňuje všude kolem nás. Když komunikujeme s bankou, nakupujeme na internetu nebo využíváme sociální sítě, vstupují do hry algoritmy strojového učení, jejichž účelem je zefektivnit, zjednodušit a zabezpečit naše aktivity. Strojové učení a s ním související technologie se rychle vyvíjejí.

3. *Neuronová síť*
   Umělá neuronová síť je jeden z výpočetních modelů používaných v umělé inteligenci. Vzorem je chování odpovídajících biologických struktur. Jedná se o strukturu určenou pro distribuované paralelní zpracování dat.
   Skládá se z umělých neuronů, jejichž vzorem je biologický neuron. Ty jsou

vzájemně propojeny a navzájem si předávají signály a transformují je pomocí určitých přenosových funkcí. Neuron má libovolný počet vstupů, ale jen jeden výstup.

Umělá inteligence v rukou učitele
Každá z uvedených aktivit je s účastníky procvičena na praktických příkladech, kde si všichni dané funkce samostatně vyzkouší.

Robote, co vidíš?

Vezmu-li obrázek a vložím jej do prostoru na stránce, stránka sama rozpozná, co se na něm nachází. Že je na něm dům. Ale dále podle okolí pozná, že se nachází v Praze. Dokonce je schopen rozpoznat i osoby na louce, či jej, v případě např. účtenky, převede na text.

Robote, co slyšíš?

Stroje v dnešní době rozumí i lidské řeči. A to v několika jazycích.
Speech to Text (STT) převede hlas na text. Toto si můžeme sami vyzkoušet v aplikaci *Dokumenty* (Nástroje – Diktování / Hlasové zadávání). Vhodná aplikace pro mobilní telefon může být například Live Transcribe, která je tímto způsobem skvělým nástrojem pro komunikaci s neslyšícími.
Text to Speech (TTS) předvádí skvělou výslovnost a generování slov v angličtině a jiných světových jazycích.

Mouse Age – Swipe Age

Jak to všechno začalo? Lidé potřebovali komunikovat s počítačem a nejvíce jim to usnadnila myš. Příchod iPhonu poté vše zjednodušil a miliony lidí začaly používat prsty k ovládání zařízení. Téměř nic se nemusí učit. Ale co je pro člověka nejvíce intuitivní?
Poslouchat a mluvit. Už v roce 1970 počítač rozuměl tisíci slov. Proto byl nasazen do výtahů, ústředen nebo aut. Bohužel se jedná o slovní zásobu přibližně tříletého dítěte.
S příchodem sociálních sítí se začal rychle učit. Měl mnoho vzorků na trénování a dnes umí více než milion slov, což je už více, než slovní zásoba jednoho jazyka. Dokonce zvládá při poslouchání dialogu vytěsnit okolní ruchy (jako projíždějící auto či mobilní telefon) a umí pochopit, co se např. na videu děje. Proč? Sociální sítě (např. Facebook) chtěly mít jistotu, že video, které na něj bylo nahráno, je bezpečné. Zda například neobsahuje návod pro domácí sestrojení bomby a jedná se pouze o balení narozeninového dárku.

Talking Age

V roce 2010 byla společností NC zveřejněna aplikace Siri. Fungovala skvěle, tak celý vývojový tým koupil Apple a o rok později ji integroval přímo do iPhone 4S.

Reklamou bylo to, že není třeba nic instalovat, stačí jen stiskem tlačítka asistenta aktivovat a po vyřčení hesla „Hey, Siri“ říci příkaz.
Bylo to poprvé, co je možno dělat reklamu na produkt, který okamžitě (během sledování reklamy) můžete začít používat. Na vývoji však brzy začaly pracovat i další společnosti.
Amazon chtěl také službu jako Siri, ale uzpůsobenou tak, aby slyšela z celé místnosti. Její vývoj zabral čas, ale v roce 2014 představil Alexu, která má od té doby největší náskok před ostatními. Google o dva roky později integroval hlasovou asistentku do všech telefonů.

Diskuse: Už jste někdy využili svou hlasovou asistentku? Jak?

Podobně jako asistentky Amazonu (Alexa) nebo Applu (Siri) reagují na daný pokyn, asistentku Googlu spouštíme příkazem „Hey Google“. Firma Google, obdobně jako Amazon, představila i reproduktor, ve kterém svou asistentku schovali například pro domácí použití.
Facebook představil koncem roku 2018 Portal poháněný Alexou. Je přes něj možné provozovat videohovory na FB Messengeru a WhatsAppu, nebo sdílet a prohlížet fotky/videa na FB nebo Instagramu.
Souboj velikánů: Nevýhodou asistentů/asistentek je obecně to, že nemluví česky. Cena jde u nás v současné době hodně dolů a je možné si je pořídit za přijatelnou částku. Umělá inteligence je dnes dostupná na mnoha zařízeních.

3.4 Bezpečnost v IT

Osnova tohoto modulu je inspirována obsahem kurzu *Jsem online bezpečně*, který pro potřeby Czechitas navrhl lektor Michal Kučera. Pro potřeby *Učitelské akademie* a této diplomové práce byla však zachována pouze osnova a současný obsah byl vytvořen mnou s využitím uvedených zdrojů a literatury.

V následujícím modulu se budeme věnovat úvodu do problematiky IT bezpečnosti, vysvětlení základních, avšak důležitých pojmů jako *hacking* či *social engineering*. Cílem je dozvědět se informace nezbytné pro zabezpečení našich účtů obsahujících nejen naše osobní data a seznámení se s riziky světa informačních technologií.
Forma výuky: frontální výuka, praktické ukázky.

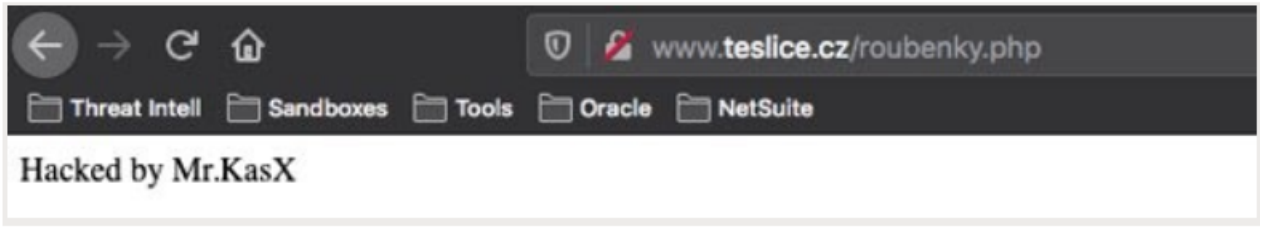
Napadení počítače

Technologie se stále vyvíjejí dopředu a s nimi se také vyvíjí zločin stejným, ne-li vyšším tempem. Tak jako si lidé chránili před digitalizací své peníze v trezorech, musíme se stejnou odpovědností přistupovat k ochraně našich peněz v bankách.

Aktivita: Sepište si na papír 5 věcí, které jsou pro vás v IT světě důležité, nebo věci, které byly za posledních 100 let digitalizované a měli bychom si je nějakým způsobem chránit.
Mohou to být zcela konkrétní věci jako například fotky mých dětí, nebo i abstraktní pojmy jako soukromí apod. Cílem je *uvědomit si*, co všechno pro nás může být důležité, co si konkrétně vy chcete chránit a na co si tedy dát pozor.

Hacking

Co to vlastně je? Pojem se dá přeložit jako *hledání a/nebo využití slabin systému nebo zařízení*. V reálném životě nám jako příklad mohou posloužit zadní vrátka, v IT světě se jedná například o součást kódu, jehož zneužití umožní vzdálený přístup do systému.
Kdo je to hacker? Hackerů, tedy lidí, kteří se zabývají takovýmito vniknutími do IT systému, je hned několik typů. Dělíme je do skupin *white hat, grey hat, black hat*. *White hat*, neboli etický hacker, je počítačový expert, který se specializuje na testování bezpečnostních opatření společnosti či systému. *Grey hat* je označení hackera, který může ale nemusí pracovat ilegálně. *Black hat* je označení užívané pro hackery pracující výhradně ilegálně.
Jaká je jeho motivace? Mohou to být peníze, strategická výhoda či touha po uznání.
Jsou hackeři dohledatelní? Ano i ne. Jejich dohledatelnost může být velice náročná, skrz několik VPN, maskování a podobně. Je důležité zdůraznit, že útoky jsou většinou masové.



Obrázek 9: Ukázka podpisu hackera

Co všechno se dá hacknout? Téměř cokoliv, třeba i teploměr v akváriu[8]
Jaké útoky jsou běžné (ve smyslu častého výskytu u běžných uživatelů)?

- (Spear) Phishing (více níže).
- Ransomware – škodlivý software, který šifruje disk a vyžaduje výkupné.
- (D)DoS – distribuovaný DoS (denial of services, neboli odepření služby), tedy útok, který zahltí daný server, ten spadne a lidé se na něj nedostanou.
- Kyberšikana.

Hacker často touží po tom být viděn, ukázat své dovednosti. Proto se hackeři po útoku různým způsobem podepisují (viz obrázek 9).

Social Engineering (sociální inženýrství)

Zahrnuje techniky, které mají za cíl zmanipulovat uživatele a donutit jej udělat určité kroky.

Modus Operandi

Účelem je

1. Vyvolat strach („Máme video/fotografie . . . “)
2. Donutit uživatele jednat pod časovým tlakem („Máte pět minut, jinak . . . “)

Cílem je donutit uživatele k akci, než se stihne zamyslet. Často se útočí na psychiku člověka, strach, zvědavost atd.

[8] https://thehackernews.com/2018/04/iot-hacking-thermometer.html

Phishing

Jedná se o útok na uživatele s cílem získat citlivé údaje, jako je například jméno a heslo, 2FA[9] nebo číslo platební karty. Útočník se vydává za autoritu (banku/-poskytovatele služeb) a pod různými záminkami od oběti získává potřebná data. Cílený phishing je označován jako *Spear Phishing*.
Zamyslete se nad zranitelností např. v emailu. Jedná se o zprávy obsahující zvláštní odkazy či odesílatele. Tyto emaily mají nějaký výhružný nebo zastrašující kontext – Někdo se vám dostal do bankovního účtu, je potřeba okamžitě obnovit heslo. apod. Cílem je oklamat uživatele a donutit jej k zadání údajů na podvodné stránce.
Dalším ze způsobů, jak podvést uživatele, je například vylákání telefonního čísla pod záminkou výhodné koupě. Na toto číslo lze poté posílat zpoplatněné SMS. Toto je běžná forma, kterou najdeme například při procházení obsahu stránek s kradenými filmy, většinou vyskočí jako reklamní okno před spuštěním filmu/-seriálu.
Útoky lze posílat i formou SMS zprávy. Může se jednat o žádosti navštívení různých stránek, výhry nebo i upozornění na vyzvednutí balíčku, který jste si ale neobjednali.
Telefonický phishing: V dnešní době lze provádět i telefonický phishing. Ten je zejména účinný v korporacích, kde se lidé navzájem neznají. Většinou jsou takové útoky cílené na HR a obecně lidi, kteří nemají kybernetickou gramotnost. Jako útočník si na LinkedInu mohu najít IT support dané firmy. Většinou na webových stránkách dané firmy, nebo i jinde, lze najít telefonní číslo a pomocí např. služby Spoofcard[10] číslo zfalšuji. Pak už jen zavolám do dané firmy, ohlásím se jako osoba, které má číslo náležet. Oběť vidí správné číslo i jméno a nenapadne ji, že by se mohlo jednat o útok. Pod záminkou aktualizace systému mohu donutit uživatele přejít na svou stránku, kde si stáhne a nainstaluje můj škodlivý malware.

Aktivita: Otevřete si Phishing Quiz[11], zadejte požadované údaje a odhalte phishingový útok.

Malware

Malware je obecné označení pro škodlivý software. Jedním z případů, jak jej získat, je například obdržení zprávy o napadení počítače virem. Pod touto záminkou byste si měli stáhnout zázračný software. Zkušenější a pozorný uživatel si však všimne, že podle URL adresy se nejedná o stránku Microsoftu, jak je ve zprávě uvedeno, ale s největší pravděpodobností jde o pokus o napadení.
Podobným způsobem lze narazit na podobné nabídky i na telefonech například při procházení stránek s pirátským obsahem. Součástí poplašné zprávy může být

[9] 2FA = dvoufaktorové ověření
[10] https://www.spoofcard.com/
[11] https://phishingquiz.withgoogle.com

například odpočet, který uživatele donutí k rychlejšímu rozhodnutí.

Blackmailing

Útočník, který vás údajně natočil v intimních chvilkách vám poslal mail, že bude tak hodný a nikomu dané video neukáže, pokud mu pošlete výkupné. Samozřejmě se jedná o podvod, tyto maily v sobě mají i přílohy, které jsou pojmenovány například vaším jménem. Nutno podotknout, že se nejedná o nic zvláštního, obzvlášť pokud máte svůj mail někde na webu včetně vašeho jména. Internet denně prochází stovky robotů, kteří mohou tyto údaje sbírat.

Ochrana soukromí

Co o vás vědí ostatní?
Aktivita: Motivační video[12] *+ následná diskuse – Co to ve vás vyvolalo?*

Různé sociální sítě nabízí různé možnosti ochrany. Například facebook umožňuje rozsáhlé nastavení ochrany soukromí. Dokonce je možnost i zobrazit vlastní profil tak, jak ho vidí veřejný uživatel.
Máte přehled o tom, jaké aplikace máte připojené k facebookovému účtu a jaká jsou jednotlivá oprávnění?
Ze svého facebookového účtu si také můžete stáhnout všechna svá data, fotky, zprávy, konverzace apod.
Aktivita: Projděte si nastavení facebooku a správně si nastavte ochranu soukromí. Projděte si oprávnění aplikací na telefonu – jaká aplikace má k čemu přístup.

Falešné profily

Podle čeho můžete poznat, že se jedná o falešný profil? Tyto profily jsou většinou sledovány placenými followery[13], samotný profil je bez věrohodné historie. Nachází se na něm umístěny cizí fotografie, které například úplně nesouhlasí s informacemi na profilu uvedenými (osoba na fotografii se dle profilu nachází v ČR, ale pozadí na fotografii je z USA).

Co se nevyplatí sdílet?

Není dobré sdílet například letenky s čárovým a identifikačním kódem. Dále pak platební karta či klíč, jelikož i ten se již dá z fotografie okopírovat. Díky dnešním technologiím a kvalitním fotoaparátům lze dokonce i prst vyfotit s takovou přesností, že je na základě takovéto fotografie možno duplikovat otisk prstu.

Diskuse: Co se nahraje na internet se jen tak nesmaže (představení modelového příkladu + následná reflexe).

[12] https://www.youtube.com/watch?v=F7pYHN9iC9I
[13] follower = sledující

Eva nafotila intimní fotografie a poslala je kamarádovi. Ten si je nenechal pro sebe a teď (po několika letech) je jich plný internet. Evu teď může kdokoliv vydírat, může mít problémy v práci či ve škole, hrozí jí riziko kyberšikany.

Chování v síti

Netiketa je souhrn všeobecných pravidel slušného chování na internetu. Pří pohybu v síti je třeba mít neustále na paměti to, že na druhém konci sedí také člověk. To, co napíšete do klávesnice, byste možná dotyčnému do očí nikdy neřekli.
Dodržujte obvyklá pravidla slušnosti běžného života. Co je nevhodné v normálním životě je samozřejmě nevhodné i na internetu. A v neposlední řadě odpouštějte ostatním chyby. I vy je děláte. Nevysmívejte se jim a nenadávejte za ně.

Co když mi někdo nadává nebo vyhrožuje?
Neodpovídejte dané osobě. Udělejte si záznam (screenshot) a konverzaci v žádném případě nemažte. Agresora ignorujte a nahlaste jeho chování provozovateli/policii. Informujte svou rodinu či přátele, nebojte se svěřit.

Na koho se můžu obrátit?
Možností existuje hned několik, mohou se lišit podle povahy závadného obsahu. V případě vydírání je určitě na místě obrátit se na policii ČR, dále například CSIRT.cz, v případě zneužití karty či účtu kontaktujte svou banku a pro všechny případy je zde samozřejmě k dispozici i Linka bezpečí, na které vás navedou na nejvhodnější řešení.
Závadný obsah je třeba hlásit (například prostřednictvím STOP online[14]).

Ochrana účtu

V tomto bloku se budeme zabývat přihlašováním do našich účtů a zabezpečením dat.

Co znám: Nejčastěji využívanou formou ochrany jsou hesla.
Co mám: Další možností ochrany je dvoufaktorové ověření, či pomocné nástroje jako Yubikey[15] nebo RSA SecurIS[16].
Jedná se o další vrstvy ochrany. Je možná otravné neustále zadávat nějaké kódy, ale je to výrazné zlepšení ochrany proti potenciálnímu útoku.
Co jsem: Skělou formou ochrany je biometrika – skenování oční duhovky či otisku prstu.

[14] https://www.stoponline.cz/cs/
[15] https://www.yubikey.cz/
[16] https://en.wikipedia.org/wiki/RSA_SecurID

Obrázek 10: Šifrovaná komunikace

Kontrola přihlášení

Je dobré si kontrolovat kdo a odkud je přihlášený (Nastavení účtu – Zabezpečení). V každé službě existuje sekce zabezpečení. Tu je dobré si projít a nastavit si lepší zabezpečení – minimálně dvoufaktorové ověřování.

Zabezpečená komunikace

Je na místě pokaždé, když zadáváte data jako osobní údaje, číslo platební karty, heslo či 2FA. Také když potřebujete věřit obsahu (např. v případě úřadů). Je třeba si všímat, zda jsou stránky zabezpečené. To zjistíme podle zeleného či šedého zámečku u URL adresy (viz obrázek 10), který nám ukazuje to, že je komunikace šifrovaná, tedy si ji nikdo nepřečte, pokud komunikaci odposlouchává. Z tohoto důvodu jsou velmi rizikové například veřejné wifi sítě.

Aktivita: Druhý faktor
Stáhněte si aplikaci Authy[17] *(iOS, Authy 2–Factor authentication – Android), vyberte si oblíbenou službu (Gmail/Outlook, Facebook, Instagram, Twitter, GitHub, . . .) a postupujte podle návodu.*
Registrujte si své telefonní číslo a ověřte jej. Zakažte podporu více zařízení. Poté přidejte účet (skenování QR kódu/manuální přidání), popisek a ikonu a uložte. Následně do služby přidáváte vygenerovaný kód.

Co dělat se záložními kódy? Záložní přístup do účtu, když všechno selže (ztratíte telefon, nebo vám jej ukradnou). Ideálním řešením je kódy vytisknout a v zalepené obálce uschovat v trezoru či uzamčeném šuplíku.

Heslo

Jak silné je vaše heslo? Jakých zásad byste se měli držet při jeho tvorbě?

Aktivita: Motivační video[18] *+ následná diskuse. Co to ve vás vyvolalo?*
Nejčastějším heslem je stále 123456. Dále pak Jméno partnera + nějaké číslo, jméno potomka + rok narození, telefonní číslo, rodné číslo, oblíbená kapela či jméno oblíbeného herce. Jestliže máte někde vymyšlené opravdu bezpečné heslo, je to ideální. Pokud však máte problém si jej zapamatovat, hlavně si jej nepiště na papírek a nelepte k počítači. Ani nikam jinam. Existují mnohem lepší způsoby, jak hesla uchovat. Jsou jimi speciální programy, tzv. password managery,

[17] https://authy.com/
[18] https://youtu.be/RfAdux3XidM

Number of Characters	Numbers only	Upper or Lower case letters	Upper or Lower case letters mixed	Numbers, Upper & Lower case letters	Numbers, Upper & Lower case letters, Symbols
3	instantly	Instantly	Instantly	instantly	instantly
4	Instantly	Instantly	Instantly	Instantly	instantly
5	instantly	instantly	instantly	3 secs	10 secs
6	instantly	instantly	8 secs	3 mins	13 mins
7	instantly	instantly	5 mins	3 hours	17 hours
8	Instantly	13 mins	3 hours	10 days	57 days
9	4 secs	6 hours	4 days	1 year	12 years
10	40 secs	6 days	169 days	106 years	928 years
11	6 mins	169 days	16 years	6k years	71k years
12	1 hour	12 years	600 years	108k years	5m years
13	11 hours	314 years	21k years	25m years	423m years
14	4 days	8k years	778k years	1bn years	5bn years
15	46 days	212k years	28m years	97bn years	2tn years
16	1 year	512m years	1bn years	6tn years	193tn years
17	12 years	143m years	36bn years	374tn years	14qd years
18	126 years	3bn years	1tn years	23qd years	1 qt years

Obrázek 11: Rychlost prolomení hesla [21]

které si budou hesla pamatovat za vás a dokonce vám je pomůžou i vytvářet (DVTV Apel[19]).

Kdo zná mé heslo? Užitečnou stránkou sdružující kradené databáze různých citlivých údajů, včetně karet a přihlašovacích údajů je Have I Been Pwned[20]. V rámci následující aktivity si zkontrolujte své účty.

Aktivita + Diskuse: Kdo zná mé heslo?
Otevřete si Have I Been Pwned a zadejte váš email. Notifikoval vás provozovatel o úniku? Co jste udělali? Pamatujete si to heslo? Kde všude jste takové heslo měli? Změnili jste si toto heslo i jinde?

Co by mělo heslo splňovat? Existuje množství tabulek (viz obrázek 11), která zobrazují rychlost prolomení hesla. *Jaké by tedy mělo být mé heslo?* Mělo by být dostatečně dlouhé (13 a více znaků), komplexní (malá i velká písmena, speciální a numerické znaky), nedohledatelné ve slovníku (ani jako zřetězení více slov). O generování a uložení takového hesla se postará již zmíněný správce hesel

[19] https://video.aktualne.cz/dvtv/nejcastejsi-heslo-je-porad-123456-proto-smejdi-nakupuji-vasi/r ef9d754c24d311e98c840cc47ab5f122/
[20] https://haveibeenpwned.com/

(například 1password[21] nebo LastPass[22]).

Několik důležitých pojmů a doporučení na závěr

- *Firewall* vás chrání před útoky zvenčí (a nejen to).
- *Router/Modem* se stará o připojení k internetu. Může být cílem útoků.
- *VPN* je virtuální privátní síť. Může se hodit například při cestování.
- *Free Wifi* se nachází například v kavárnách, na letištích či v MHD. Kdo poslouchá, slyší vše (co není šifrováno). Pokud používáte šifrované spojení (HTTPs, POP3s), můžete bezpečně brouzdat.
- *Wifi a sdílené heslo* – oproti síti bez hesla zde není téměř žádný rozdíl. Proč? Pokud používáte slabé heslo, dá se snadno rozlousknout. Řešením může být například speciální síť pro hosty.

[21] https://1password.com/
[22] https://www.lastpass.com/

3.5 Algoritmizace

V následující části textu se budeme věnovat tématu *algoritmizace*[23] a úvodu do *robotiky*. Ukážeme si několik nástrojů, které lze ve výuce využít a vše si vyzkoušíme na praktických příkladech. Také si představíme několik způsobů, jak vysvětlit problematiku algoritmizace bez využití počítače, konkrétně na problematice třídících algoritmů. Nastíníme, jak tenká je hranice mezi algoritmizací a programováním a pro přechod mezi nimi si ukážeme vhodné nástroje. Na závěr se seznámíme s velkým trendem ve výuce informačních technologií – roboty. Využití robotiky ve výuce si nastíníme na jedněch z nejužívanějších robotů, kterými jsou (pro první stupeň ZŠ) Ozoboti.
Forma výuky: frontální výuka, skupinová práce, projektová výuka, praktické ukázky.
Potřebný materiál: Notebook, tablet/chytrý telefon, Ozobot.

Algoritmus

Motivační Aktivita: Postup činnosti
Sepište seznam instrukcí pro

1. *napití se (výchozí – zavřená lahev s vodou, sklenice)*
2. *zabalení dárku (výchozí – role balicího papíru, lepicí páska, nůžky, krabice)*
3. *zavěšení nafouknutého balonku na kliku (výchozí – vyfouknutý balonek, klubko špagátu, nůžky)*
4. *vyhození nepořádku do koše (výchozí – zmačkaný papír ležící na zemi)*
5. *nakreslení domečku jedním tahem (výchozí – flipchart, fix).*

Studenti jsou rozděleni do skupin, každá skupina dostane jiný úkol (zadání stejné, činnost se liší). První seznam instrukcí píše každý sám, poté probíhá první test několika vybraných postupů. Následně všichni se stejnou činností dohromady sepisují jeden funkční postup a probíhá druhý test.

Definice algoritmu

Algoritmus je posloupnost instrukcí pro řešení problému. Takto bychom mohli zavést definici algoritmu. Bohužel se s ní pojí pár dalších pojmů, které v ní byly použity a které je potřeba vysvětlit. Co je to problém? Instrukce? Co to znamená řešit problém? [22]

[23]The man with a hammer sees every problem as a nail. Our age's grat hammer is the algorithm. (William Poundstone)

Problém

Diskuse: Uveďte příklady problémů
Cílem diskuse je poukázat na existenci různých typů problémů. Problémem může být například parkování auta, výpočet lineární rovnice či Izraelsko – Palestinský problém. Všem je již teď jasné, že ne všemi typy problémů se budeme zabývat ve spojitosti s algoritmy. Jak tedy vypadá problém, o kterém se zmiňujeme v definici algoritmu?
Popis problému:
Každý problém je jednoznačně popsán množinou přípustných zadání (*vstupů*), přiřazením (*předpisem/zobrazením*), které pro každé přípustné zadání (*vstup*) říká, jaké je odpovídající (správné) řešení (*výstup*). Shrneme-li tyto poznatky do definice, můžeme říci, že **problémem rozumíme přiřazení odpovídajícího výstupu ke každému přípustnému vstupu.**
Podíváme-li se na problémy, které byly zmíněny na začátku bloku, lze jednoznačně rozhodnout, které nás budou zajímat:

1. **Problém** (Parkování auta)
   Vstup: Popis počáteční dopravní situace (poloha auta, popis okolní situace včetně místa k parkování)
   Výstup: Popis správné koncové situace (auto je správně zaparkováno).

2. **Problém** (Řešení lineární rovnice $ax + b = 0$)
   Vstup: Dvojice (a, b) racionálních čísel a, b $(a \neq 0)$
   Výstup: Racionální číslo $-\frac{b}{a}$, jestliže $a * (-\frac{b}{a}) + b = 0$.

3. Izraelsko–Palestinský problém algoritmem nevyřešíme.

Instrukce

Diskuse: Uveďte příklady instrukcí
Podobně jako u *problému* je cílem diskuse poukázat na to, že s pojmem instrukce si každý může spojit něco jiného. Zmíněné příklady je vhodné rozdělit podle typu instrukce do několika skupin, například:

- Sečti čísla a a b (aritmetická instrukce).
- Do proměnné x ulož číslo 5 (instrukce přiřazení).
- Je-li $c > 0$, zvyš hodnotu b o 1 (podmíněná instrukce).
- Přečti číslo na vstupu (vstupně–výstupní instrukce).
- Pro každé $i = 1, 2, 3, 4, 5$ postupně proveď: Vytiskni hodnotu $2*i$ (instrukce cyklu, v němž je vstupně–výstupní instrukce).

Instrukcí rozumíme jednoznačný a srozumitelný pokyn.
Naše představa je taková, že existuje někdo (něco), kdo (co) instrukcím rozumí

a je schopen je mechanicky vykonávat.
Tento „vykonavatel instrukcí“ (člověk/počítač) je schopen instrukce vykonávat tak, jak jsou předepsány algoritmem (ve správném pořadí).

Řešení problému algoritmem

Algoritmus *řeší daný problém* právě když pro každý přípustný vstup I daného problému, jemuž odpovídá výstup O, platí: Vykonávání instrukcí podle algoritmu se vstupem I se po konečném počtu kroků zastaví a na výstupu je O.
Předpokládáme, že I je na začátku zapsáno na dohodnutém vstupním zařízení (disk, uživatelem přes klávesnici) a výstup se objeví na dohodnutém výstupním zařízení (disk, monitor).

Aktivita: Rozhodněte, zda se jedná o vhodný způsob přiřazení výstupu ke vstupu?
Vstup: $ax + b = 0$
Výstup:

1. *Pokud* $a \neq 0$*, zapiš na výstup číslo* $-\frac{b}{a}$.
   Pokud $a = 0$ *a* $b = 0$*, zapiš na výstup „Každé číslo je řešením“.*
   Pokud $a = 0$ *a* $b \neq 0$*, zapiš na výstup „Rovnice nemá řešení“.*

2. *Zkus odhadnout řešení.*
   Vyzkoušej, zda je správně. Pokud ano, zapiš jej na výstup. Pokud ne, zpřesni odhad a jdi dále.

Ve spojitosti se zmíněnou definicí algoritmu i souvisejících pojmů se pojí několik otázek k *diskusi:*

- *Je každý problém řešitelný algoritmem?*
- *Má smysl rozlišovat lehčí a těžší problémy?*
- *Lze rozlišovat algoritmy dle efektivity?*

Cílem této diskuse je zmínit se před posluchači o oblastech jako je vyčíslitelnost nebo složitost a čím se zabývají, dále netřeba téma rozebírat.

Zápis algoritmu

Existuje hned několik způsobů, jak lze algoritmy zapsat a znázornit. První z nich jsme si již vyzkoušeli v úvodní aktivitě, k dalším se samozřejmě dostaneme postupně.

1. *Přirozený jazyk:* tento popis je snadno srozumitelný i pro laiky, nevýhodou je však nejednoznačnost a zdlouhavost.

2. *Programovací jazyk:* popis je jednoznačný a následná tvorba programu je jednoduchá (programátoři zápisu rozumí), ale často obsahuje zbytečné detaily a je zdlouhavý.

3. *Pseudokód:* je snadno pochopitelný i neprogramátorům, je srozumitelný a přepis do programovacího jazyka je snadný. Pro implementaci je přepis dokonce nutný.

4. Grafické znázornění, například *vývojový diagram.*

Aktivity – třídící algoritmy

V následujícím bloku se budeme věnovat problému *třídění.* Jak jsme si již ukázali v předchozím textu, každý problém má jednotnou formu zápisu:
Vstup: posloupnost n čísel $(a_1, a_2, \ldots, a_n)$.
Výstup: permutace $(b_1, b_2, \ldots, b_n)$ vstupní posloupnosti.
Výstupní posloupnost vznikne přerovnáním prvků vstupní posloupnosti tak, aby v ní byly prvky setříděny podle velikosti.

Proč je problém třídění důležitý? Kvůli efektivnějšímu vyhledávání v rozsáhlých datech. Jedná se o častou úlohu při zpracování dat a velké množství složitějších algoritmů využívají algoritmy pro třídění.
Z definice problému třídění plyne následující postup: Procházej všechny možné permutace pole A, pro každou z nich ověř, zda je pole A setříděné. Pokud ano, skonči. Pokud ne, přejdi k další permutaci.

Základní přehled třídících algoritmů

- **Insertion sort (řazení vkládáním)** známe jako způsob řazení karet (po jedné), ležících v balíčku na stole.

  1. Mějme jeden prvek, ten je triviálně zařazen.
  2. Vezměme následující prvek a zařaďme jej na správné místo v již zařazených prvcích.
  3. Dokud pole obsahuje nezařazené prvky, opakuj (2).

- **Selection sort (řazení výběrem)** někteří z nás také využívají pro řazení karet, tentokrát však těch, které již máme rozdány v rukou. Postupně prochází pole a hledá nejmenší prvek, který posléze zařadí na začátek nesetříděné části.

- **Bubble sort (bublinkové řazení** je skvělým algoritmem pro motivaci žáků, kteří do problematiky zatím úplně nepronikli, jelikož se dá v rámci jednoduché aktivity demonstrovat přímo ve třídě bez toho, aniž by zúčastnění žáci něco věděli o algoritmizaci či problému třídění:
  Aktivita: Postavte všechny žáky do jedné dlouhé řady a dejte jim pokyn, aby se seřadili podle výšky od nejmenšího po největšího. Začíná se řadit od toho, kdo stojí úplně vlevo a každý smí komunikovat jen s osobou po své pravici.

- **Merge sort (řazení slučováním)** není zrovna algoritmem, který bychom mohli znát z běžného života, ale jeho znázornění prostřednictvím tance[24] je skvělým návrhem pro průřezové téma či školní představení.

Algoritmizace v praxi

Zásobníkem zajímavých úloh na procvičení algoritmizace může být například archiv bobříka informatiky[25], kniha Algorithmic Puzzles [23] nebo náš příklad z praxe zaměřený na spolubydlení [**Bydl**].

Problém: Spolubydlení

Představte si šest spolubydlících: Libora, Zuzanu, Petra, Pavlu, Ondru a Míšu. Žijí v jednom bytě a dělí se o náklady na společně používané věci jako je toaletní papír, mýdlo, prací prášek a podobně. Postupně své útraty zapisují do tabulky. Dejme tomu, že uplynulo například půl roku a spolubydlící se chtějí navzájem finančně vyrovnat. Vaším úkolem je vymyslet přesný postup, který mají následovat, aby došlo k celkovému vyrovnání všech lidí.
Důležité je tento postup vymyslet tak, abychom jej později dokázali sdělit počítači, což znamená, že náš postup musí být opravdu detailní. Nelze přikazovat věci typu „rozpočítej útratu mezi všechny účastníky“. Takové příkazy počítač nezná. Budeme muset pracovat s určitými omezeními.
Počítač si můžeme představit jako obyčejného úředníka s tužkou, papírem a kalkulačkou. Naprosto nerozumí pojmům jako účastník, částka, rozpočítat a podobně, vůbec neumí samostatně přemýšlet. Všechno mu musíme vysvětlit polopaticky a nevynechat žádný detail, na kterém by se mohl zaseknout. On umí pouze základní početní operace na kalkulačce, umí si něco poznamenat na papír a to je tak všechno.

Pokud se podíváte na naši tabulku výdajů, možná budete mít nutkání ji vyřešit tzv. metodou „prostě kouknu a vidím, nějak to tady přičtu, tady odečtu a hotovo“. To se může celkem povést pro tabulku o šesti lidech. Těžko ale takový postup uplatníme například pro menší Švýcarskou vesnici, která má 375 obyvatel a v tabulce výdajů je 8 822 záznamů. Pro představu, taková data zaberou oboustranně 197 listů A4 a těžko je můžeme zpracovat tímto způsobem.

Popis přirozeným jazykem:

1. Spočítej, kolik každý člen utratil celkem.
2. Spočítej průměrnou útratu na jednoho člena.
3. Spočítej rozdíly jednotlivých členů proti průměru.

[24] https://www.youtube.com/watch?v=XaqR3G_NVoo
[25] lhttps://www.ibobr.cz/test/archiv

4. Všechny peníze těch, kteří zaplatili podprůměr, dej do banku.

5. Bank rozděl mezi ty, kteří zaplatili nad průměr.

Pro počítač je tento návod pořád velmi složitý. Zapsali jsme tedy řešení problému *programovacím jazykem*, konkrétně pomocí Pythonu:

```
1  import statistics
2
3  seznamJmen = []
4  utraty = []
5
6  for vydaj in vydaje:
7    jmeno = vydaj[0]
8    utrata = vydaj[2]
9    if jmeno in seznamJmen:
10     index = seznamJmen. index (jmeno)
11     utraty[index] += utrata
12   else:
13     seznamJmen. append (jmeno)
14     utraty. append (utrata)
15
16 prumernaUtrata = statistics. mean (utraty)
17
18 for jmeno in seznamJmen:
19   index = seznamJmen. index (jmeno)
20   vyrovnani = round (utraty[index] -- prumernaUtrata)
21   if vyrovnani > 0:
22     print (jmeno + ' dostane\t' + str (vyrovnani))
23   else:
24     print (jmeno + ' ma dati\t' + str (--vyrovnani))
```

Seznamy proměnných seznamJmen a utraty představují tabulku celkových útrat pro každého jednoho člověka.

Od algoritmizace k programování, základy robotiky

Pro přechod od algoritmizace k programování je vhodné využít nejrůznějších nástrojů, na kterých můžeme trénovat algoritmický způsob myšlení, ale zároveň se již setkáváme s prvky programování. Pro tyto účely nám skvěle poslouží prostředí s tzv. bloky, které na sebe řetězíme jako skládačku a dohromady nám tvoří první jednoduché programy. Pro první setkání s bloky se nám krásně hodí Blockly Games[26], kde se žáci naučí s bloky zacházet a kdekoliv jinde to pro ně již bude známá věc [24].

Ozoboti ve výuce

Ozobot je malý robot, jehož je možné naprogramovat hned několika způsoby – pomocí nakreslených čar a barevných kódů (ozokódy) nebo díky jednoduchému

[26] https://blockly.games/

editoru Ozoblockly, v němž se skládají kousky kódu za sebe. Je tedy vhodný pro seznámení s programováním a robotikou. Rozlišujeme dva typy ozobotů, *Ozobot BIT* a *Ozobot EVO*.
BIT je menší robot s pěti světelnými čidly pro rozpoznání čar a barevných kódů s jednou RGB LED diodou. Svítí, bliká a nevydává žádné zvuky. Lze jej ovládat pomocí barevných kódů, umí sledovat čáru a přečíst barevné kódy, odbočovat na křižovatkách, točit se i zastavovat. Je také programovatelný pomocí PC/tabletu přes editor Ozoblockly.
EVO disponuje bluetooth s dosahem kolem 9 metrů. Pomocí několika LED diod svítí, bliká, umí vydávat zvuky a díky senzorům detekujícím vzdálenost umí rozpoznat předměty umístěné před/za ním, sledovat jiné objekty a také samozřejmě sleduje čáru a čte barevné kódy.

Jak ozobota udržovat?

Jako u jiných technologií a zařízení i o Ozoboty je třeba se správně starat. Pro nás jsou 4 kroky a to *nabít, chránit, čistit* a *kalibrovat*.

1. **Nabít:** Ozobot se nabíjí pomocí mikroUSB kabelu a trvá to necelou hodinu, vydrží pak nabitý přibližně jednu vyučovací hodinu (nabitý svítí zeleně, částečně nabitý bliká)

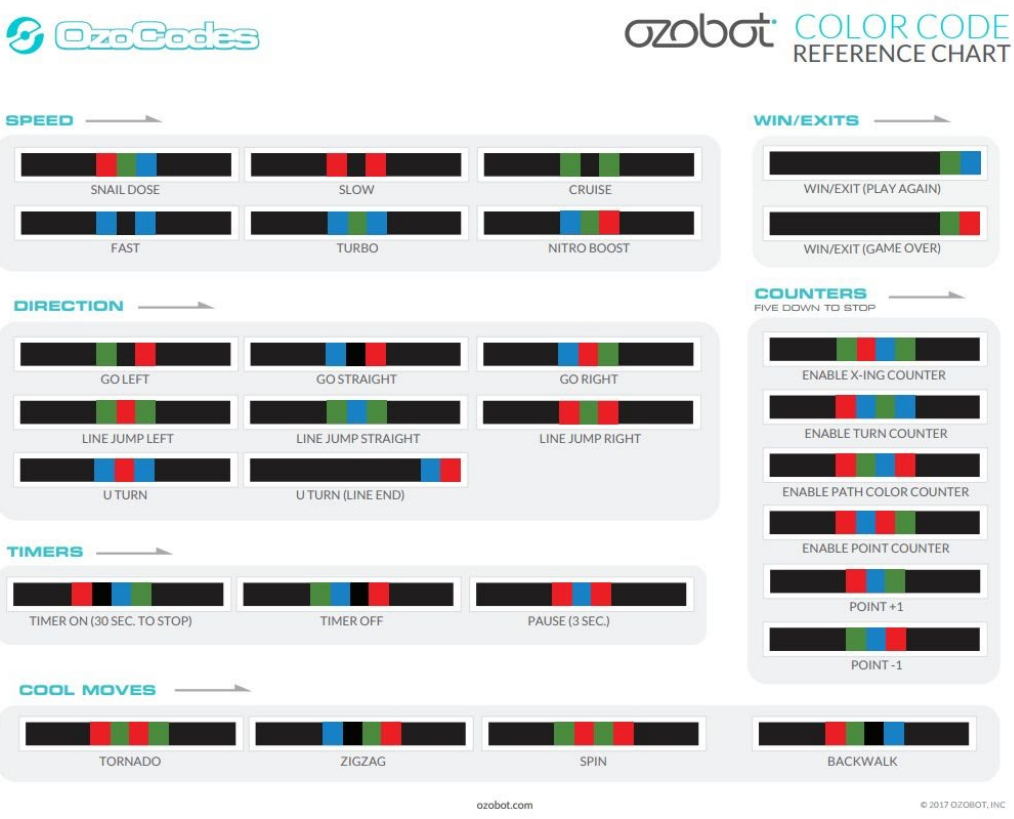
2. **Chránit:** Součástí balení je ochranný obal. Funguje jako cyklistická helma – tlumí nárazy a mírní následky. Dále ozobotovi nesvědčí přímé slunce, vlhko ani extrémní teploty.

3. **Čistit:** Čištění kol od prachu probíhá velmi jednoduše pomocí bílého papíru a je třeba provádět pouze u modelu BIT. Stačí jej položit na papír a opatrně s ním přejíždět dopředu a dozadu. Nepoužívá se voda ani žádné čistící prostředky

4. **Kalibrovat:** Kalibrace je jedním z nejdůležitějších kroků – infračervené senzory fungují jako oči. Proto je potřeba ozobota připravit na to, v jak ostrém světle bude pracovat a jakou barvu v daném světle mají kódy. Nutno kalibrovat při každé změně prostředí (změna světla, výměna papíru za tablet, . . .)

Offline programování

Na začátku tohoto bloku již bylo zmíněno, že ozoboty lze programovat pomocí kreslení čar doplněných o speciální kódy. I pro toto kreslení však existují určitá pravidla, která je třeba dodržovat.
Co není vhodné pro kreslení dráhy pro Ozobota, jsou pastelky, propisky, voskovky či křídy. Naopak nejlepší volbou jsou buď fixy přímo od výrobce Ozobota, nebo fixy na flipchart se seřízlou špičkou. Ozobot dobře reaguje i na některé typy zvýrazňovačů, to je však potřeba vyzkoušet s konkrétními kusy.



Obrázek 12: Skupiny ozokódů [25]

Ideální šířka dráhy je cca 5mm a pomohou ji nakreslit právě ploché fixy. Součástí dráhy s kódy bude i spousta zatáček. Pro práci s Ozobotem jsou ideální tupé a pravé úhly, ostré by nemusel zvládnout.
Aby neměl robot problém kód přečíst, je potřeba se držet základních pravidel. Barevné kódy lze umisťovat pouze na černou linku. Přestože Ozobot umí sledovat i barevné linky (a svítit barvou, kterou sleduje), číst kódy umí jen na černé čáře.
Šíře políček ozokódů musí být stejná a mezi jednotlivými barvami nesmí být bílé místo. Stejně tak se barvy nesmějí překrývat.
Pravidla pro práci s ozokódy se vztahují také k jejich umístění na dráze. Nikdy nesmějí být umístěny v zatáčkách, v křižovatkách nebo jejich těsné blízkosti ($< 3\ cm$) nebo příliš blízko jiným ozokódům.
Co to vlastně ty *ozokódy* jsou a jak vypadají? Jaké existují?
Jedná se o kódy měnící rychlost, směr, pracující s časem, „cool“ pohyby, počítáním či výhrou/ukončením hry. Přehled těch základních můžete vidět na obrázku 12.

Online programování

Online programování ozobota probíhá buď přes webové prostředí *Ozoblockly* (v němž lze programovat i model BIT, ale proces načtení kódu je složitější), nebo přes aplikaci *Evo by Ozobot*, přes kterou se ozobot páruje s telefonem/tabletem přes Bluetooth.

1. **Ozoblockly:** Jedná se o editor přístupný z webové stránky[27]. Program se spustí po stisknutí tlačítka „Get Started“.

   - V pravé části editoru se nachází sloupce s různými kartami. V kartě označené ozubeným kolečkem vybereme jazyk, který chceme (barvu pracovní plochy, ...).
   - *Důležité* je mít jas obrazovky nastavený na maximum a při nahrávání programu nehýbat s ozobotem.
   - Pro otevření nového prázdného prostředí využíváme v kartě Uživatel položku Nový program, chceme-li otevřít již uložený program, využijeme položku Otevřít program.
     v pravém sloupci nahoře nalezneme šipky Zpět / Vpřed / Smazat, v levém sloupci tlačítko pro výběr typu Ozobota, ukazatel úrovně příkazů a jednotlivé záložky s příkazovými bloky.
   - Ozoblockly umožňuje nastavit *úrovně příkazů* podle předchozích programátorských znalostí. Pro představu, v úrovni 1 pracujeme pouze s obrázky, je tedy vhodná pro malé děti. Naopak úroveň 5 již umožňuje užití polí, seznamů nebo tvorbu vlastních funkcí. Přecházení mezi úrovněmi je samozřejmě možné, jsou mezi sebou kompatibilní.

2. **Evo by Ozobot:** Účelem je údržba ozobota, jednoduché programování (pouze v angličtině, ale editor Blockly lze přepnout i do jiného jazyka), je zde nutnost povolení Bluetooth atd.

   - **Menu** (po přihlášení a kliknutí na ozobota):
     Přejmenování, Kalibrace, Více info, Odpojení.
   - **Údržba** (Více informací):
     Run Evo checkup (vyhledání nejnovějších aktualizací), Mute Evo (úprava hlasitosti), Brightness (intenzita světel). Nastavení je nutno potvrdit stiskem „Apply“.
   - **Ovládání ozobota z aplikace:**
     Úvodní stránka (tlačítko „Play“), drive (ovládání ozobota pomocí joysticku), pohyb, světelné a zvukové efekty, změna rychlosti či sledování čar.
     Ozoblockly editor má podobné prostředí, jako ozoblockly.com, ale není

[27] https://ozoblockly.com/

nutno ozobota kalibrovat nebo přikládat k obrazovce.
DISCOVER – tipy, seznamy kódů, inspirace, zvukové efekty.

Projektová výuka: Aktivity s ozobotem
Účastníci se rozdělí do skupin. Každá skupina pak pracuje na svém projektu, který má několik částí:

1. Vymyslete hru/aktivitu pro využití ozobota ve výuce (ne nutně v hodině IVT).
2. K této aktivitě navrhněte podklady (dráhy, zadání, vzorové řešení).
3. Tyto podklady převeďte do elektronické podoby (dráhu je možno „vyrobit“ například v editoru Canva, v Malování či v MS Excel).
4. Ke své aktivitě vytvořte propagační materiály (plakát, infografika (viz obrázek 13) . . .).

Ozobot ve vzdělávání:
Proč je ozobot dobrou investicí pro školu a skvělou volbou při rozhodování, co svým dětem pro rozvoj pořídit? Ozobot pomáhá rozvíjet hned několik důležitých znalostí a dovedností, mezi nimiž je například:

- Logické myšlení.
- Analytické myšlení.
- Kreativitu.
- Pomáhání při výuce optimalizace a hledání vhodného řešení problémů.

Proč se ozoboti dětem tak líbí?

Největšími výhodami je okamžitá interakce, rychlá odezva (brzy vidím výsledky své práce) a také jejich možnosti využití. Je to robotická hračka, která je skladná, navíc svítí, jezdí a v případě modelu Evo i vydává zvuky. Také máme možnost volit si práci online nebo offline.

- Při práci s ozoboty ve třídě nebo kroužku je vhodné mít k dispozici více Ozobotů, než je žáků.
- Je potřeba myslet na údržbu (nabíjet, aktualizovat, čistit) a také na to, že se musí během hodiny i po ní někde nabíjet (USB hub). Oba modely vydrží až jednu vyučovací hodinu, je ale důležité, aby nebyly pořád zapnuté (naučte žáky, aby je vypínali).
- Dále je třeba mít k dispozici smartphone nebo tablet se staženou aplikací Ozobot Evo pro případnou aktualizaci firmware a stahování aktualizací (pouze v případě vlastnictví modelu EVO).

Obrázek 13: Infografika – údržba ozobota

Když shrneme výhody vlastnictví ozobotů, pomáhají seznámit se se světem programování. Možnost si s ním hrát je prakticky kdekoliv. Je malý, takže se vejde i do kapsy a k jeho rozpohybování stačí papír a několik barevných fixů.
Nevýhodou je jeho výdrž – nabitý nám vydrží přibližně jednu vyučovací hodinu a poté se musí stejně dlouho dobíjet. V případě, že párujete více ozobotů skrze bluetooth s tablety, může se stát, že nastane problém s připojením správného ozobota ke správnému tabletu. Tomuto se dá vyhnout tak, že si předem každého ozobota pojmenujete a nalepíte na něj lísteček s jeho jménem, kterým jej máte pojmenovaného i v prostředí Evo by Ozobot.
V případě hledání vhodné alternativy pro hodiny robotiky na druhý stupeň základní školy mohu jedině doporučit programovatelné stavebnice LEGO. S výběrem té nejvhodnější pro danou věkovou skupinu může pomoci můj článek, který byl publikován ve speciálním vydání časopisu Computer a který je přílohou (B) mé diplomové práce.

3.6 Programování

Obsahem posledního výukového modulu je úvod do programování. Studenti se seznámí s pojmem *programování* a naučí se základy několika programovacích jazyků. Cílem modulu je navést učitele ke způsobu, jak uchopit výuku základů programování na základní i střední škole.
Forma výuky: frontální výuka, samostatná práce, praktické ukázky, aktivity.

Programování jako pojem

Diskuse: Co si představíte pod pojmem programování?
Cílem této diskuse je zjistit, co si účastníci o programování myslí. Zda si uvědomují, co vše se dá programovat (hardware, software, pračka apod.). V návaznosti na předchozí modul si připomínáme, že počítače a stroje obecně jsou velmi hloupé a programátor je ten, kdo jim dává „život“.
I když věnuji programování 1000 hodin, nemusí to nutně znamenat, že je ze mě perfektní programátor. Programovací jazyky a styly se neustále vyvíjejí a je třeba se přizpůsobit trendům, nebo alespoň vědět, kam se odkázat, kde sehnat správné a užitečné informace atd.

Kolik jazyků musím umět, abych byl dobrý programátor?

Toto je obecně velmi špatně položená otázka. Je potřeba se ptát, co potřebuji na konkrétní projekt, na kterém se chystám pracovat. Rozhodně není dobré se držet jednoho jazyka, ale vyzkoušet i jiný. Není ani ideální se učit několik let jeden jediný jazyk a pak zjistit, že pro další projekty budu potřebovat úplně jiný (na vývoj mobilních aplikací či webů použiji jiné jazyky).
V kapitole 3.1 byl již uveden základní přehled programovacích jazyků a taktéž u nich bylo zmíněno, že některé z nich jsou pro začátky čí položení základů lepší volbou, než jiné. V programování existuje spousta principů a syntaktických pravidel, která jsou přenositelná mezi jednotlivými jazyky.

Jaké nástroje využívám při programování?

1. **IDE (Integrated Development Environment):** viz str.22.
   Příkladem pro nás může být prostředí „Visual Studio“.

2. **Editor:** Nástroj, který nám, stejně jako IDE, dokáže podbarvovat kód a tím velmi zjednodušit programování. Je určen pro vývoj méně náročných projektů, jelikož neobsahuje žádné pokročilé nástroje, ale oproti IDE má velmi nízké nároky na hardware. Příkladem je editor „Visual Studio Code“.

Vizuální programování

Pro začátky s programováním již na základní škole je skvělým nástrojem tzv. *vizuální programování*. To zahrnuje skupinu programovacích jazyků/nástrojů, ve

kterých se zdrojový kód tvoří pomocí skládání předprogramovaných bloků za sebe (jako puzzle) a jejich spojováním vzniká náš program.

Dobrou příležitostí pro první seznámení s bloky může být prostředí **code.org** a v něm například hra Klasické bludiště[28] či Ledové království (Frozen)[29]. Hráči jsou po jednotlivých krocích přesně naváděni, co mají dělat, a během několika úrovní se naučí základní manipulaci s bloky i úvod do algoritmického myšlení.

S vizuálním programováním se úzce pojí také projekt Hour of Code[30], který má na svědomí právě nezisková organizace Code.org.

- Jedná se pravděpodobně o největší vzdělávací akci v dějinách informačních technologií.
- Jeden týden v roce po celém světě nejrůznější učitelé, děti či rodiče organizují pro své okolí aktivity, které umožňují nahlédnout do světa programátorů úplně každému.
- Kdokoli, kdekoli může uspořádat událost Hodina kódu.
- Jedná se o jednohodinové výukové programy ve více než 45 jazycích. Nejsou nutné žádné předchozí zkušenosti.
- Hodinu kódu dokonce provázejí i offline aktivity, takže je možnost se zapojit i se skupinou, která nemá k dispozici počítače.

Úvod do programování – Scratch

Další úrovní programování pomocí bloků může být Scratch. Teď už se bavíme o opravdovém programovacím jazyce, který na první pohled možná působí, že je vhodný pouze pro ty nejmenší, ale není tomu tak. Ve Scratchi se dokáže zabavit úplně každý a dají se v něm naprogramovat animace či hry od těch nejjednodušších až po ty složitější.
V příloze A se nachází metodika, která slouží jako podklad pro výuku deseti hodin úvodu programování ve Scratchi.

Úvod do programování – JavaScript (JS)

Co je to JavaScript? Jedná se o programovací jazyk, často označovaný jako skriptovací. V našem kontextu si představené jazyky rozdělíme následovně:

- HTML – obsah a struktura stránky.
- CSS – vzhled stránky.

[28] https://studio. code.org/hoc/1
[29] https://studio. code.org/s/frozen/stage/1/puzzle/1
[30] https://hourofcode.com/cz

- JavaScript – chování a interaktivita stránky.

JavaScript byl původně navržen pro přidávání drobných efektů na stránku. Dnes se však používá na webu téměř na vše (například velké aplikace typu Facebook, Gmail apod.).
Kam s ním v kódu? JavaScriptový kód se píše do souboru s koncovkou .js Tyto soubory se do HTML souboru připojují pomocí značky <script>, součástí níž je atribut „src“ odkazující na soubor s naším JS kódem. Tuto značku <script> můžeme umístit buď do hlavičky („head“), nebo těsně před konec „body“.

Proměnná

Proměnná je místo, kam si mohu uložit nějakou hodnotu pro pozdější použití. Můžeme si ji představit jako pojmenovaný „chlíveček“ v paměti počítače, ke kterému se mohu kdykoliv pomocí jména vrátit a k hodnotě přistoupit, přečíst ji nebo změnit.
Před použitím je potřeba proměnnou *deklarovat*, což provádíme pomocí klíčového slova „let“.

```
1  let jmeno.
2  jmeno = 'Tereza'.
3  /* let pouzijeme jen jednou na zacatku programu, neopakujeme jej
       pokazde, kdyz chceme do promenne vkladat novou hodnotu */
4  let vek = 25.
5  let pocet =  5.
6  pocet = pocet + 3.
7  osloveni = 'pani' + jmeno.
8  /* S promennymi lze provadet nejen matematicke operace */
9  const pi = 3.1415926.
10 /* Promenne, ktere se behem programu nemeni, se deklaruji pomoci
       klicoveho slova 'const' a nazyvaji se konstanty */
```

Datové typy

Datové typy jsou téma, které zde zmíním pouze okrajově. U hodnot proměnných v našem kódu rozlišujeme, zda se jedná o celé číslo, desetinné číslo, text a podobně. Touto problematikou se zabývají právě datové typy. Těchto typů je několik a my si jako příklad uvedeme právě ty tři, které jsem již zmínila:

- int – celočíselný datový typ.
- float – datový typ reprezentující desetinná čísla. V programování je důležité myslet na to, že se zde pro zápis využívá desetinná tečka, nikoli desetinná čárka.
- string – datový typ sloužící k uchování řetězce znaků a práci s nimi [26].

JavaScript patří mezi tzv. dynamicky typované jazyky, což znamená, že při deklaraci proměnné není třeba uvádět, o jaký datový typ se bude jednat. JavaScript

sám podle hodnoty proměnné a jejího zápisu určí, s jakým typem bude do budoucna pracovat.

Výpis do konzole

Jestliže chceme od JavaScriptu něco vypsat (hodnotu proměnné, výsledek operace, návratovou hodnotu funkce, ...), můžeme k tomu využít konzoli. V prohlížeči se k ní dostaneme přes „Prostředí pro vývojáře“ (F12). Vypisovat do konzole můžeme s pomocí funkce „console.log()“.

```
1 console.log (zprava);
2 console.log ("Hello world!");
3 console.log ("10" + "+" + "10" + "=" + (10+10));
4 let jmeno = "Tereza Nedjalkova";
5 let vek = 25;
6 console. log ("Jmenuji se" + jmeno + "a je mi" + vek + ". " );
```

Operace s proměnnými a operátory

Základní operátory, které při programování (nejen) v JavaScriptu budeme využívat můžeme rozdělit do následujících tří skupin:

- Operátor přiřazení „=“, pomocí kterého přiřazujeme hodnotu do konkrétní proměnné.
- Matematické operace „$+ - * / \%$“, které využíváme při nejrůznějších výpočtech tak, jak je známe z běžného života.
- Spojování textových řetězců „+“. Operátor '+' můžeme kromě sčítání hodnot využít jako „spojovník“ řetězců datového typu *string* s jinými řetězci.

Objekt Math

Pro složitější matematické operace a výrazy nám poslouží objekt Math. Díky němu můžeme volat funkce jako například generování náhodného čísla či výpočet dolní celé části. Oba tyto případy si nyní vyzkoušíme na příkladu.

Cvičení

Napište program, který bude fungovat jako hrací kostka, na které po každém novém spuštění programu padne číslo 1–6.
Využijte při tom Math.random() (generuje náhodné číslo z intervalu $\langle 0, 1)$), Math.floor(x), která dané číslo zaokrouhlí směrem dolů (dolní celá část čísla) a dále si poraďte sami.
Možné řešení:

```
function hodKostkou (){
  let hod = Math. round (Math. random * 6 + 1).
  console. log (hod).
}
```

Podmínky (rozhodování a větvení programu)

Podmíněné výrazy nám umožňují tzv. rozvětvit program. V určitém místě je možnost různých způsobů postupu a na základě splnění či nesplnění předepsané podmínky program pokračuje dále.

```
if (podminka) {
    \\ prikazy, ktere se vykonaji po splneni
}
```

Nejen v JavaScriptu je třeba brát v potaz, že ne každý výraz je vhodný na místo podmínky a ne vše se v syntaxi jazyka zapisuje stejně, jako tomu rozumíme intuitivně. Konkrétním příkladem je pro nás příkaz přiřazení a porovnání rovnosti hodnot.

- = vyjadřuje již zmíněný *operátor přiřazení*

  ```
  let x = 5.
  ```

- == je tzv. *equality operator*.
  JS porovnává hodnoty nehledě na datové typy (v případě porovnání různých datových typů vykoná *implicitní konverzi hodnot*, tedy převede číslo na text a porovnává dva textové řetězce).

  ```
  5 == '5'.   //true
  ```

- === je tzv. *identity operator*, který porovnává pouze hodnoty se stejnými datovými typy (liší-li se datový typ porovnávaných hodnot, výsledek porovnání je nepravdivý).

  ```
  5 === '5'.   //false
  5 !== '3'.  //true
  ```

Větev *else* podmínky: Existuje spousta případů, kdy testujeme podmínku, ale potřebujeme provést „něco“, když podmínka neplatí. K tomu nám slouží větev *else*, která se provede jen tehdy, je-li podmínka nepravdivá.

```
if (podminka) {
    //podminka plati
} else {
    //podminka neplati
}
```

Řetězení podmínek s *else if*:
Podmínka *if* může mít další větev. To využijeme v případě, že nám možnost *else* nestačí a chceme prověřit další podmínku.
Je nutné myslet na to, že na pořadí podmínek **záleží** a je nutné, aby postupovaly logicky za sebou. Ve chvíli, kdy program jednu z podmínek vyhodnotí jako pravdivou, zabývá se již pouze blokem kódu bezprostředně za touto podmínkou a další větve neřeší.

Logické operátory

Existují případy, kdy trvám na splnění více podmínek současně. Například se chystám jít na svatbu jako družička a potřebuji si koupit šaty, které:

1. Ladí s kyticí nevěsty.
2. Nemají moc odvážný střih.
3. Jsou v mých finančních možnostech.

Šaty si tedy koupím jen za předpokladu, že jsou splněny podmínky X (barva), Y (střih) i Z (cena).
K tvorbě takové kombinované podmínky využijeme tzv. *logické operátory*, kterými jsou *AND* (&&), *OR (||)* a *NOT (!)*. Pro jejich splnitelnost platí stejné podmínky, jako známe z matematiky pro logické spojky *konjunkce, disjunkce* a *negace*. Tedy podmínka skládající se z dvojice podmínek spojených operátorem AND je splněna tehdy a jen tehdy, když jsou obě dílčí podmínky pravdivé. Je-li podmínka spojením dvou podmínek s využitím operátoru OR, je splněna tehdy a jen tedy, je-li splněna alespoň jedna z dílčích podmínek. Poslední zmíněný operátor NOT, tedy negace, nám převrací pravdivostní hodnotu naší podmínky z pravdy na nepravdu a naopak.

```
1 if (teplota > 20 && teplota <= 30) {
2     console. log ("Dnes je prijemne pocasi na prochazku");
3 }
```

Cykly

Cykly využíváme v případě, že se nám nějaký kus kódu opakuje [27]:

1. Cyklus **FOR**: Tento cyklus má stanovený pevný počet opakování a obsahuje tzv. řídící proměnnou (celočíselnou), ve které se postupně během běhu cyklu mění hodnoty.

```
1     for (let i=0; i<10; i=i+1) {
2         //prikazy, ktere se vykonaji pri kazdem behu cyklu
3         console. log (i);
4     }
5     // klicove slovo (zavedeni pocitadla; podminka; krok
          pocitadla) {\dots }
```

2. Cyklus WHILE funguje jinak, jednoduše opakuje příkazy v bloku dokud platí podmínka.

```
    let a = 0;
    while (a < 10) {
        //prikazy, ktere se vykonaji pri kazdem behu cyklu
    }
    \\klicove slovo (podminka) {\dots }
```

Funkce

Funkce využíváme ve chvíli, kdy máme v našem kódu sadu příkazů, které chceme vykonávat opakovaně. Funkce má jméno, kterým ji volám.
V programování se obecně snažíme vyhnout opakování kódu. Funkce nám zjednodušují a zpřehledňují kód, zvyšují jeho čitelnost.
Jedná se o blok kódu, který jednou napíšeme a potom ho můžeme libovolně volat bez toho, abychom ho psali znovu a opakovali se. Funkci deklarujeme pomocí klíčového slova FUNCTION a obsahuje blok kódu ve složených závorkách[28].

```
//definice vlastni funkce
function nazev () {
    //sada prikazu
}
nazev ();    //volani funkce - jmeno + kulate zavorky
```

Funkce s parametrem:
Funkce může mít také libovolný počet vstupních parametrů, které píšeme do závorky v její definici a podle nich ovlivňujeme její chování. Funkce také může vracet zpět nějakou hodnotu. Slouží k tomu příkaz return.

```
function secti (a, b) {
    let c = a + b;
    return c;   //c - navratova hodnota
}
```

Výhoda funkcí je tedy v přehlednosti a úspornosti (můžeme napsat nějaký kód jednou a volat jej třeba stokrát na různých místech skriptu). Když se rozhodneme funkci změnit, provedeme změnu jen na jednom místě a tato změna se projeví všude, což značně snižuje riziko chyb [28].

Závěr

Tato diplomová práce je textem, který by měl současným i nastávajícím učitelům informatiky pomoci doplnit si znalosti v oblastech, o které se bude rozšiřovat rámcový vzdělávací program nebo které jsou pro výuku důležité, ale v současné době se na ně neklade příliš velký důraz. Práce sama o sobě není dostačujícím materiálem k rekvalifikaci těchto učitelů na učitele informatiky, ale může posloužit jako materiál, který dohromady slučuje nejdůležitější oblasti, které by kvalifikovaný učitel měl znát a ve kterých by se měl orientovat. Může napomoci k jejich poznávání a nasměrovat k dalšímu studiu.

V rámci vývoje projektu jsem se setkala se spoustou zajímavých lidí z různých oblastí vzdělávání, ať už se zástupci učitelů, ředitelů základních i středních škol, ministerstva školství, mládeže a tělovýchovy (MŠMT), vedoucími zájmových kroužků pro děti či odborníky na informační technologie. Část témat a přidružených aktivit jsem „otestovala“ na vzorku učitelů, kteří projevili zájem o takovéto dovzdělání. Myšlenku práce jsem taktéž pod záštitou organizace Czechitas prezentovala na několika konferencích a setkáních pro učitele po celé České republice. Každé jedno z těchto setkání pro mě bylo velmi přínosná a obohacující, stejně tak poté psaní této práce, ve které jsem se snažila zužitkovat co nejvíce znalostí nabytých právě po setkáních s těmito inspirativními lidmi.

Po dokončení práce mohu jen konstatovat to, co již delší dobu vím. Informatika je krásný, ale náročný obor, který je potřeba mít opravdu rád a mít v něm velký přehled, když jej člověk chce kvalitně vyučovat a předávat své znalosti ostatním. Doufám, že má práce by v tom mohla do budoucna někomu pomoci.

Conclusions

This diploma thesis is a text that should help current and future computer science teachers to supplement their knowledge in areas that will be expanded by the framework educational program or which are important for teaching, but currently not much emphasis is placed on them. The work alone is not sufficient material to retrain these teachers as computer science teachers, but it can serve as a material that brings together the most important areas that a qualified teacher should know and in which he should orient himself. It can help to get to know them and direct them to further study.

As part of the project development, I met a lot of interesting people from various fields of education, whether representatives of teachers, principals of primary and secondary schools, the Ministry of Education, Youth and Sports, leaders of interest groups for children or information technology experts. . I "tested" some of the topics and related activities on a sample of teachers who showed interest in such training. I also presented the idea of the work under the auspices of the Czechitas organization at several conferences and meetings for teachers throughout the Czech Republic. Each of these meetings was very beneficial and enriching for me, as well as the writing of this work, in which I tried to use as much knowledge acquired after meetings with these inspiring people.

After finishing the work, I can only state what I have known for a long time. Informatics is a beautiful but demanding discipline, which you need to really like and have a great overview of when you want to teach it well and pass on your knowledge to others. I hope that my work could help someone in the future.

A Základy programování – Scratch

Cílem následujícího textu je doporučit a metodicky rozebrat aktivity vhodné pro kroužek základů programování v programovacím jazyce Scratch. Scratch je vizuální programovací jazyk, který umožňuje vytvářet programy manipulací s grafickými programovými elementy, tzv. bloky. Jedná se o jazyk vhodný pro výuku programování u dětí a dospívajících ve věku 8–18 let.

Kurz se koná formou prezenční výuky v 10 hodinách (1 hodina = výukový blok o délce 45 minut).

Úvodní hodina

Popis lekce

Cílem hodiny je představit žákům problematiku programování a následně konkrétní programovací jazyk – Scratch. Ve zbývajícím čase si vyzkouší zahrát hru vytvořenou ve Scratchi.

- Úvod do programování
- Registrace do Scratche

Podrobný popis

První hodina probíhá, až na závěrečnou část, bez počítačů, ideálně v kruhu nebo na zemi.

1. **Seznámení**

Každý v kruhu postupně dostane míček (začíná lektor) a sdělí o sobě základní informace:

- Jméno
- Proč jsem tady a co bych se zde chtěl/a naučit
- Co mě baví
- Jaké mám zkušenosti s počítačem? S programováním?

2. **Diskuse**

- **Co je to počítač?** Chceme žáky navést k tomu, že i mobil/tablet/kalkulačka/lednička jsou počítače. **Kolik máte počítačů ve svém okolí? Co to dělá? Z čeho se to skládá? Jak se to ovládá?**
- **Co je to program?** Chceme naznačit rozdíl mezi programem a softwarem (synonymum), dále mezi softwaru a hardwaru. **Z čeho se skládá?** Z instrukcí/příkazů/povelů, příkazy se skládají z dílčích podpříkazů. Zadáváme tím počítači nějakou práci.

Aktivita: Programování lektora (inspirace[1]: čas 2:20–4:56), následná diskuse, co jsme viděli. Počítače jsou hloupé a udělají jen to, co jim řekneme (velmi konkrétně).

3. **Scratch:**

K dorozumění se s počítačem používáme různé tzv. programovací jazyky. Ten, který budeme využívat my, se jmenuje Scratch. Hra, kterou si zahrajeme pro motivaci: Na kočku a na myš[2].

4. **Domácí úkol – spolu s rodiči vytvořit účet ve Scratchi**

[1] https://www.youtube.com/watch?v=kLK_cD1M3Ew
[2] https://scratch.mit.edu/projects/248290657/

Úvod do PJ Scratch

Popis lekce

Studenti se seznámí s vývojovým prostředím Scratche. Vyzkouší si základní rozhraní, přidávání objektů a jejich ovládání.

Potřebný materiál

Notebook, myš

Podrobný popis

V úvodu hodiny se zeptáme, zda není potřeba dovysvětlit něco z minula a zda si všichni založili účet na stránkách Scratche. Otevřeme si stránku, necháme všechny přihlásit. Kontrola českého rozhraní v editoru

Co je to Scratch?

Scratch je vizuální programovací jazyk, ve kterém instrukce nepíši jako text, ale využívám k tomu skládání předpřipravených programových bloků. Vlastně tak trochu skládám puzzle. Popíšeme si prostředí editoru:

<table>
<tr>
<td rowspan="2">Scénáře
Zde se nachází bloky pro tvorbu našeho kódu. Dělí se do několika skupin podle toho, na co je využíváme (události, vnímání, ...).

Kostýmy
Každá postava ve Scratchi má možnost mít k dispozici několik kostýmů, které mezi sebou mohou během běhu programu přepínat. Můžeme pomocí nich například vytvářet animace.

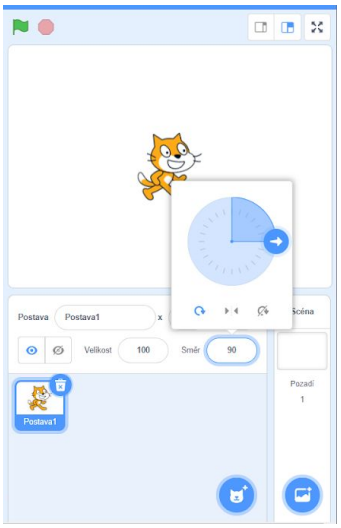
Zvuky
Kromě kostýmů patří mezi vlastnosti postav i zvuky, které mohou za běhu programu vydávat.</td>
<td rowspan="2">Vývojové prostředí
Zde skládáme bloky za sebe a vytváříme program pro jednotlivé objekty (postavy/pozadí).</td>
<td>Náš výsledný program
Zde se bude zobrazovat naše aplikace.</td>
</tr>
<tr>
<td>Postavy
Do našeho programu můžeme vkládat více postav. Každá bude mít svůj vlastní program, své kostýmy i zvuky.

Pozadí
I pozadí můžeme mít více. A i pozadí může mít svůj program!</td>
</tr>
</table>

Úkol: Cílem následujícího úkolu je vyzkoušet si základní rozhraní. Otevřete si editor, přidejte postavu auta (nebo jinou) a libovolné pozadí. Vytvořte program, při kterém budeme moci postavu ovládat šipkami.

- Šipky nahoru a dolů – krok vpřed a vzad
- Šipky vlevo a vpravo – otočka vlevo a vpravo
- Mezera – zatroubení (nebo jiný zvuk)

Co je to úhel a jak funguje?



Ve Scratchi ovládáme natočení postavy pomocí úhlu. Nechceme suplovat hodiny matematiky, ale některé děti se s úhlovou mírou ve škole ještě nesetkaly, proto je dobré vysvětlit, že "kolečko" nám tvoří 360 stupňů, můžeme jím otáčet zprava i zleva (poté se dostáváme do záporných hodnot).

V editoru je po vložení nové postavy úhel nastaven na 90°.

Cykly, souřadnice

Popis lekce

Studenti se seznámí s pojmem cyklus a vyzkouší si použití cyklů na příkladech. Dále se naučí pohybovat v souřadnicovém systému.

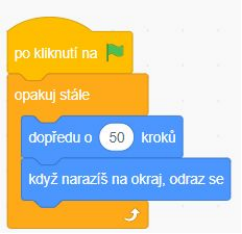
Potřebný materiál

Notebook, myš.

Podrobný popis

Jednou ze základních programátorských konstrukcí jsou **cykly**. Cykly opakují část kódu několikrát dokola. Často se využívají například pro animace, ke kterým se dostaneme později.

Jaký je rozdíl mezi **konečným** a **nekonečným** cyklem? Konečný cyklus obsahuje také tzv. ukončovací podmínku, nebo počet opakování, kolikrát se má vykonat.



Úkol: V editoru Scratch přidej novou postavu, například míč, a napiš program, po jehož spuštění bude míč poskakovat po prostoru a odrážet se od stěn.

Jak můžu umisťovat objekty na konkrétní pozice?

Orientaci v prostoru v editoru Scratch si vysvětlíme na krátké pohybové aktivitě. Všichni se zvedněte ze židle a opakujte po lektorovi.

- X – krab – chodí jen do stran a zároveň vypadá jako X *(nohy jako sumo bojovník, ruce nad hlavou)*
- Y – strom – roste do výšky a vypadá jako Y

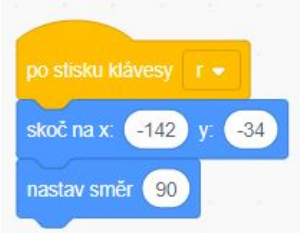
Polohu objektů určuji dvěma čísly – tzv. souřadnicemi x a y. To, kde je najdeme a jak se mění při pohybu postavy si ukažme v editoru.

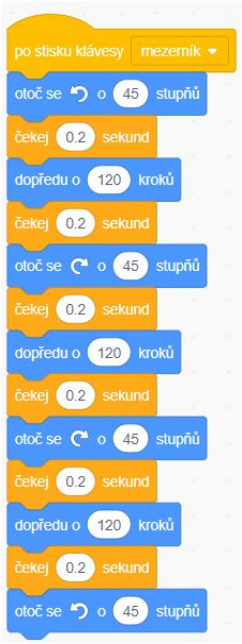
Střed se nachází na souřadnicích (0, 0). Plocha ve Scratchi je (-240, 240) pro osu x a (-180, 180) pro osu y. Celkově máme tedy k dispozici 480x360 bodů.

Aktivita: Slepá mapa – lektor ukazuje na ploše různé objekty a žáci tipují, jaké asi budou mít souřadnice.

Úkol: Přeskoč auto, aby kocoura nesrazilo!

1) Po stlačení písmene R přesuň hlavní postavu (kocoura) na začátek a nastav jí původní směr (90°).





2) Přidej druhou postavu (auto) a naprogramuj hlavní postavu tak, aby po stlačení mezerníku přeskočila přes tu druhou. Mezi jednotlivými pohyby udělej vždy krátkou pauzu.

První hra

Popis lekce

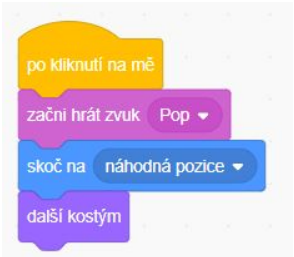
Žáci si naprogramují první hru, ve které využijí dovednosti, které se doposud naučili.

Potřebný materiál

Notebook, myš.

Podrobný popis

V první části hodiny si od dětí necháme v rámci opakování vysvětlit, jak fungují kostýmy, zvuky, cykly, posouvání pomocí změn souřadnic či úhly.

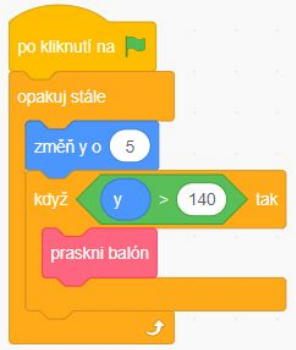


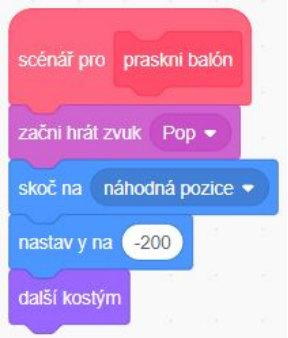
Úkol: Založ si nový projekt ve Scratchi a vytvoř v něm postavu balónku. Změň pozadí na nějaké hezké a barevné, ideálně přírodní. Napiš program, který po kliknutí myší na balónek přehraje zvuk prasklé bubliny (‘Pop’), balónek zobrazí na jiném (náhodném) místě a ještě změní jeho barvu.

Úkol: Změňte svůj program podle následujícího postupu:

1) Přidejte slunce, které se bude donekonečna točit
2) Přidejte mraky, které budou klouzat doleva a doprava
3) Balónek se pohybuje tak, že letí od spodního okraje k hornímu a pokud jej sami neprasknete kliknutím myší, praskne u horního okraje. Poté se přebarvený objeví opět u dolního okraje, ale na jiném místě a let vzhůru se opakuje.

Pozn.: Děti určitě vymyslí krkolomnější řešení, ale zadání splní. Následující řešení je s využitím vlastních bloků a podmínek, které se probírají později. Některé děti na to však přichází samy po projití všech bloků, které mají mezi scénáři k dispozici.





Dokonči příběh

Popis lekce

V této lekci se s dětmi zapojíme do projektu Dokonči příběh. Po shlédnutí videa každé se svou vlastní fantazií vytvoří krátkou animaci/hru inspirovanou videem a vycházející z předpřipraveného Scratch projektu.

Potřebný materiál

Notebook, myš, možnost promítání videa se zvukem.

Podrobný popis

Na adrese The Girl Who Hated Books[3] se nachází úvodní video k projektu 'The Girl Who Hated Books'. To lektor pustí dětem a následně jim zadá samostatnou práci, aby vytvořily vlastní pokračování daného příběhu s využitím zápletky z videa. K tomu jim poslouží předpřipravený materiál[4] od tvůrců projektu.

Je možno zpracovat animaci – rozhovor mezi Meenou a některou knižní postavou, příchod rodičů domů či například knižní kvíz, kterým bude Meena provázet. Fantazii se meze nekladou.

Inspiraci k plnění daného projektu lze najít na stránkách Scratche mezi řešeními jiných osob[5], které se do výzvy zapojily.

Nové efekty, které je zde vhodné využít:

- **Zobrazení a schování postavy** (Vzhled – skryj se/ukaž se)
- **Efekty** (barvy, velikost, víření, …)
- **Bubliny** (myšlenky a mluvení – nutno specifikovat jak dlouho)
- **Změny pozadí**

Pozn.: K podobnému zadání může sloužit jakékoli jiné video vybrané lektorem.

[3] https://www.youtube.com/watch?v=1Efrg23Sqe8
[4] https://scratch.mit.edu/projects/149857067/
[5] https://uploads.scratch.mit.edu/studios/3930335/

Podmínky

Popis lekce

Během této hodiny se žáci seznámí s podmíněnými výrazy a vytvoří si jednoduchou hru.

Potřebný materiál

Nootebok, myš.

Podrobný popis

Co je to **podmínka**? Jedná se o otázku, na kterou program odpoví buď ANO, nebo NE. Pokud odpoví ANO, je podmínka **splněná**, pokud odpoví NE, tak podmínka splněná není.

Úkol: V editoru Scratche si vytvořte 2 postavy – košík a jablko. Zvolte si nějaké hezké pozadí (ideálně se stromy). Pohyb košíku se dá ovládat šipkami vlevo/vpravo a jablko padá donekonečna dolů.

- Zapojení **podmínky**: Vnímání – dotek jiné postavy (Čeho chceme dosáhnout? Když je jablko v košíku, začne další padat shora)
- Jak na to? (chceme dojít k x–random, y = 200)

Úkol: Po chycení do košíku přidejte zvuk kousnutí do jablka. Přidejte i druhou podmínku – když jablko spadne na zem, přehraje se jiný zvuk a začne padat další jablko shora (y < -160)

Proměnné

Popis lekce

V této lekci se žáci seznámí s pojmem proměnná a pomocí něj si mohou vylepšit své doposud vytvořené hry.

Potřebný materiál

Notebook, myš.

Podrobný popis

K čemu využíváme **proměnné**? Proměnná je místo, ve kterém můžeme uchovávat nějakou informaci (číslo) při běhu programu a měnit její hodnotu. Práce s proměnnými je velice užitečná a často používaná například ve složitějších hrách nebo programech.

Úkol: Budeme pokračovat ve hře z minulé hodiny – chytání jablek do košíku. Budeme chtít počítat skóre a za každé chycené jablko obdrží hráč jeden bod. Přidejte tedy proměnnou SKÓRE, zobrazte ji ve hře a upravte program tak, aby při každém chyceném jablku přibyl jeden bod.

Úkol: Přidejte do hry podmínku, která při dopadu jablka na zem odečte ze skóre 5 bodů. Při začátku nové hry se skóre vynuluje.

Úkol: Přidejte proměnnou životy. Za každé jablko, které spadne mimo mísu se odečte život. Když máme 0 životů, jablko přestane padat. Dále přidejte do hry druhou postavu, která bude košík držet a pohybovat se s ním.

Úkol *: Zkuste změnit obtížnost hry – po dosažení skóre 20 zrychlit padání jablek nebo zpomalit pohyb misky. Dále vyzkoušejte přidat rozšíření – časovač. Po 30 sekundách ukončete hru (schovejte jablka).

Animace

Popis lekce

Vysvětlíme si, co je to animace. Princip animování si nejprve ukážeme na offline příkladu, poté nabyté vědomosti přesuneme do prostředí Scratche a zkusíme naše postavy rozpohybovat.

Potřebný materiál

Notebook, myš, čisté papíry, pastelky/fixy, nůžky.

Podrobný popis

Co je to **animace**? Jak fungují běžící obrázky? Jak vypadá páska filmu? Co je to fps?

Animace je pojem, který nám popisuje rozpohybování jednotlivých objektů. A to tak, že rychle za sebou pustíme několik obrázků. Počet obrázků, které se nám přehrají za jednu sekundu označujeme jednotkou fps (Frames Per Second).

Úkol: Nakresli na papír dvouobrázkovou, opakující se animaci[6].

Ukázka: https://cdn.mos.cms.futurecdn.net/mvDGxxU7dZrCoEFyhQxQ9f.gif

Jak probíhá animování ve Scratchi? Pomocí změny kostýmu. Jedna postava má k dispozici více kostýmů, které se od sebe v detailu liší (pohyb nohou). Zkusme ji tedy rozpohybovat.

Úkol: Rozpohybuj kocoura! Do nového projektu ve Scratchi si vlož postavu kocoura (nebo jinou, která má vhodnou volbu kostýmů). Poté napiš program tak, aby se při stisknutí šipky (dopředu/dozadu) kocour nejen posunul ve správném směru, ale aby udělal i opravdový krok (pohyb nohama).

Úkol: Na kolik fps běží Scratch? *(Nápověda:cyklus + další kostým + časovač)*

Úkol: Uprav svou hru s padajícím jablkem tak, aby postava, která nosí košík opravdu chodila. Pokud jsi to v poslední hodině nestihl, dokonči rozšířenou verzi hry (úkol *) obsahující změny rychlostí a po skončení (0 životů) vypiš na obrazovku text KONEC HRY.

[6] https://decko.ceskatelevize.cz/sikulove/navod?id=863

Klonování

Popis lekce

V této lekci si zadáme k tvoření zatím nejtěžší hru, kterou bude třeba dokončit doma. Při jejím programování se naučíme novou funkci Scratche – klonování, tedy vytváření více postav z jedné.

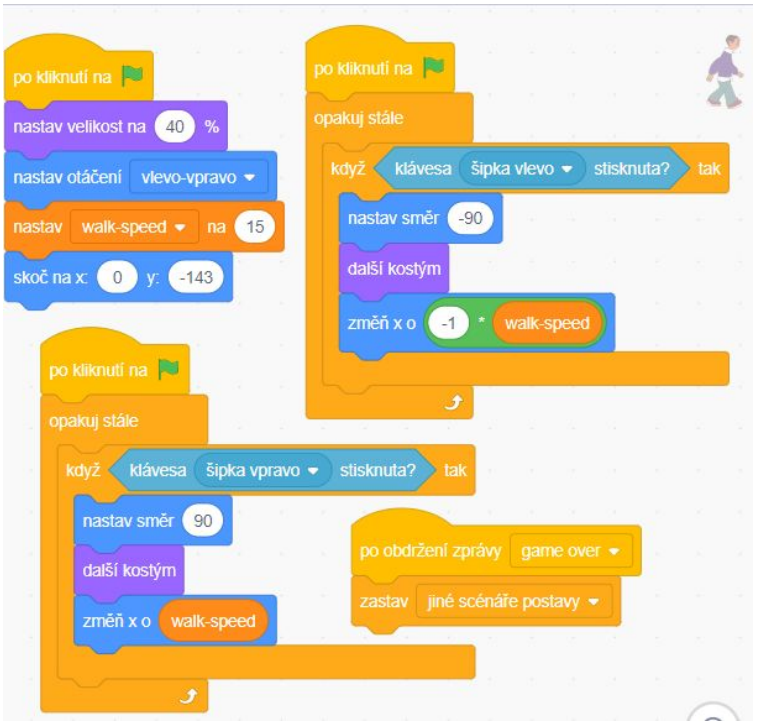
Potřebný materiál

Notebook, myš.

Podrobný popis

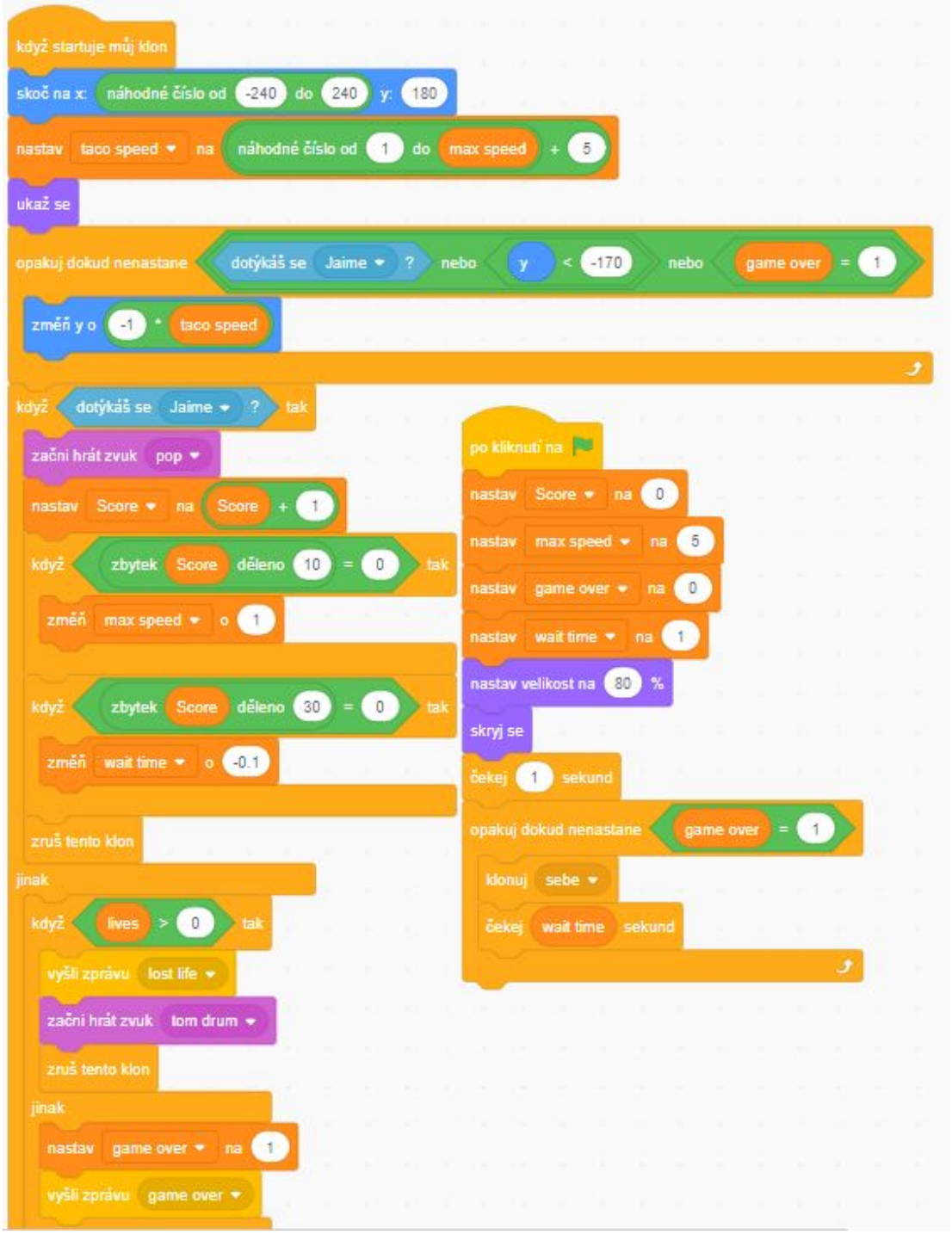
V této lekci se naučíme **klonovat**. Klonování nám umožní vytvořit z jedné postavy postav více, čili její kopie, neboli klony. Snadno tak mohu získat například spoustu mraků na pozadí nebo sněhovou vánici. Využiji k tomu bloky Klonuj sebe a poté pro nastavení vlastností Když startuje můj klon.

Úkol: Naprogramuj hru Taco Catcher. Můžeš vycházet ze základu, který máš ve hře s padajícím jablkem. Ovládání šipkami udělej plynulejší – když je stisknuta klávesa, postava se plynule pohybuje *(nekonečný cyklus)*. Padající jablko již není jen jedno, ale je jich mnohem více (po určité době se klonuje a klony se objevují na náhodných pozicích – vždy u horního okraje). Po ztrátě všech životů nezapomeň vypsat KONEC HRY a vše zastavit.

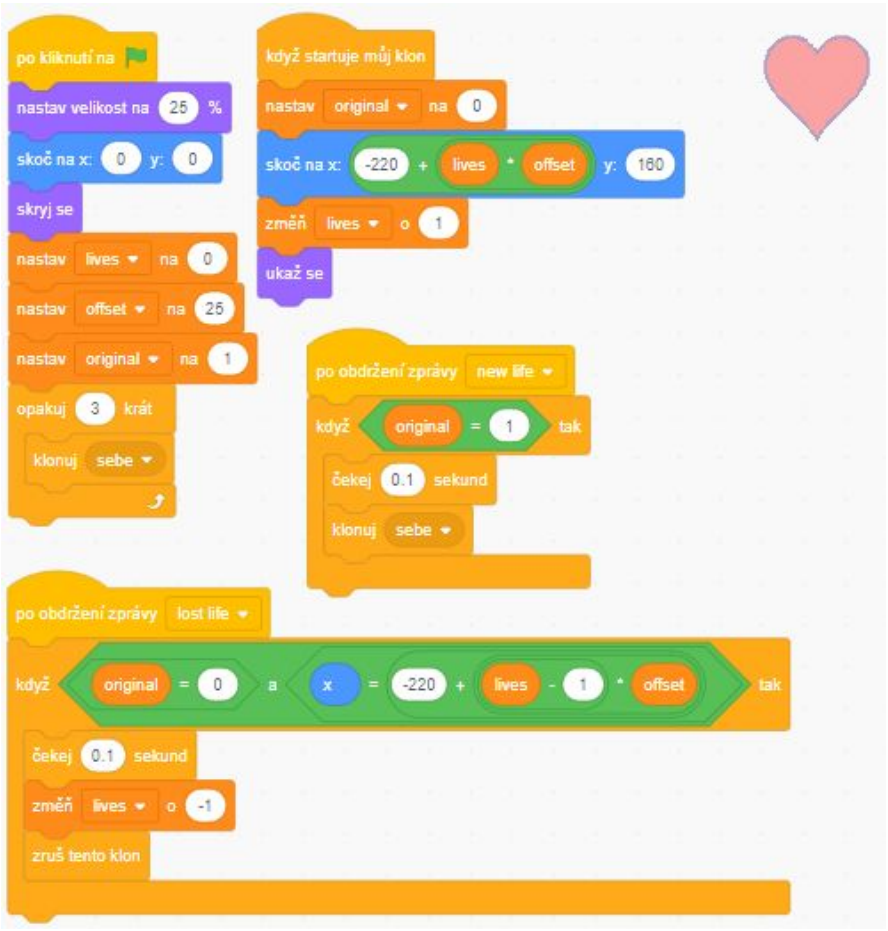


Úkol: Zkuste do hry přidat vizualizaci životů v podobě srdíček v jednom z horních rohů. Počet životů zde bude stabilně svítit jako 0–3 srdce, po ztrátě života srdce mizí. Jednou za čas mezi padajícími předměty (jablky/tacos) spadne srdce a my máme možnost získat život.

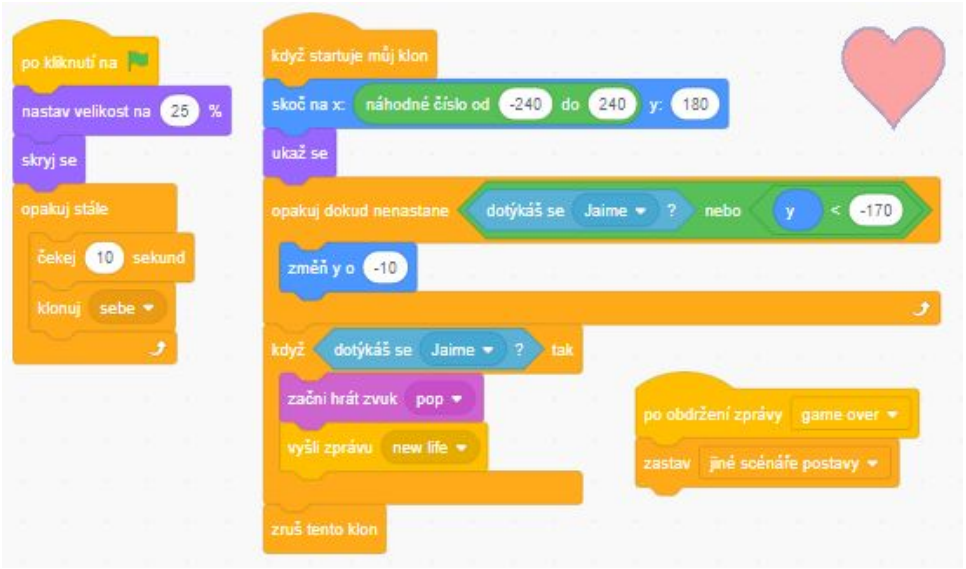
Taco:


když startuje můj klon
skoč na x: náhodné číslo od -240 do 240 y: 180
nastav taco speed na náhodné číslo od 1 do max speed + 5
ukaž se
opakuj dokud nenastane dotýkáš se Jaime ? nebo y < -170 nebo game over = 1
změň y o -1 * taco speed
když dotýkáš se Jaime ? tak
začni hrát zvuk pop
nastav Score na Score + 1
když zbytek Score děleno 10 = 0 tak
změň max speed o 1
když zbytek Score děleno 30 = 0 tak
změň wait time o -0.1
zruš tento klon
jinak
když lives > 0 tak
vyšli zprávu lost life
začni hrát zvuk tom drum
zruš tento klon
jinak
nastav game over na 1
vyšli zprávu game over
po kliknutí na
nastav Score na 0
nastav max speed na 5
nastav game over na 0
nastav wait time na 1
nastav velikost na 80 %
skryj se
čekej 1 sekund
opakuj dokud nenastane game over = 1
klonuj sebe
čekej wait time sekund

Život:



Padající život:



Zprávy, projekt (Moje Vánoce)

Popis lekce

Své dosavadní znalosti doplníme o poslední užitečnou věc a to jsou zprávy (sekce Události). Jedná se o způsob spuštění nějakého scénáře umožňující nám komunikaci mezi jednotlivými objekty. Následně všechny nabyté znalosti zužitkujeme při tvorbě animace na téma 'Má vánoční nadílka' *(lze upravit podle období v roce, kdy kroužek končí – jak jsem trávil/a velikonoční prázdniny, vzpomínky na léto, ...)*.

Potřebný materiál

Notebook, myš.

Podrobný popis

Představte si situaci, že vám všem nyní přijde úplně stejná SMS zpráva: "Do konce tydne odpada vyucovani, zadne domaci ukoly nemas, uzij si volno. "
Jak se po obdržení takovéto zprávy zachováte? *(Chceme se žáky dojít k tomu, že všichni obdrželi stejnou zprávu, ale každý se zachová jinak. Někdo se začne radovat a okamžitě plánuje, co s volným časem. Někdo jiný se bude vztekat, protože tak přijde o domluvený termín písemky, na kterou se už dva dny učí.)*.

S událostmi jsme se během lekcí již setkávali, jelikož každý scénář musí nějakou událostí začínat. Například Po kliknutí na vlajku... nebo Po stisku klávesy... a další. Scratch však také umožňuje vytváření vlastních událostí, tzv. Zpráv. Jakýkoliv scénář může poslat libovolnou zprávu všem ostatním, ty se pak chovají podle programu, který mají přiřazeny k bloku Po obdržení zprávy... .

Úkol: Vytvoř krátkou animaci na téma MÁ VÁNOČNÍ NADÍLKA, ve které ostatním sdělíš, co jsi našel pod stromečkem. Během jejího vytváření využij co nejvíce z toho, co jsi se v kroužku naučil.

Možné řešení: https://scratch.mit.edu/projects/357807266/

B Základy robotiky – Lego Mindstorms

V této příloze se nachází článek s názvem „Lego pro vývojáře“, který jsem napsala pro speciální vydání časopisu Computer, zaměřené na programování pro děti. Účel článku byl přiblížit různé stavebnice produkované firmou Lego a vytvořit tak stručný přehled jejich možností a srovnání, ať už pro rodiče, kteří chtějí některou ze stavebnic pořídit pro své dítě, tak pro učitele či ředitele, kteří chtějí roboty pořídit do školy, ale prozatím neví, který model zvolit.

KLASICKÁ STAVEBNICE JAKO UČEBNÍ POMŮCKA

LEGO pro vývojáře

Stavebnici LEGO zná snad každé dítě. Hra s plastovými kostkami však nemusí končit jen postavením domečku se zahrádkou. Dánská společnost se už řadu let věnuje i robotickým hračkám. Díky jejím programovatelným sadám si tak můžete vyrobit plnohodnotného robota.

[Tereza Nedjalková]

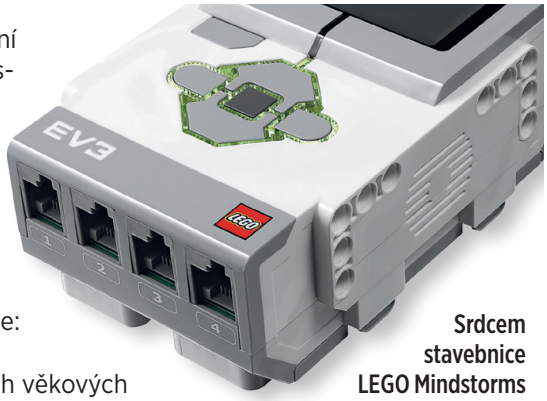
Technologie a počítače jsou všude kolem nás. Celý svůj život si zjednodušujeme práci vývojem a používáním techniky, která částečně nebo úplně zastane lidskou úlohu při výkonu dané činnosti. Od nástrojů usnadňujících obdělávání půdy jsme se dopracovali až k plné robotizaci některých výrobních procesů. Ale co je to vlastně ten robot? Kde se vzal a proč jich je v poslední době všude tolik?

Slovo „robot“ bylo poprvé použito v díle Karla Čapka s názvem R.U.R. Dále se tematika robotů v literatuře i filmové tvorbě objevila nesčetněkrát. Zajímavé je, že v kinematografii či vědecko-fantastické literatuře se ve spojitosti s roboty klade důraz na vzezření co nejvíce podobné člověku, a to jak po fyzické, tak po psychické stránce. Ve skutečném světě nás však mnohem více než vzhled robota zajímá jeho funkčnost

a efektivita. Jen pro zajímavost, první průmyslový robot byl vyroben v roce 1954 v USA společností Unimation.

Dánský gigant

Ale dost z obecné historie, podívejme se blíže na roboty určené pro výuku, konkrétně od společnosti LEGO. Tato dánská firma byla založena už v roce 1932. Její název byl odvozen z dánských slov leg godt, což se překládá jako „hraj si dobře“ a latinské slovíčko LEGO pak znamená „dát dohromady“. Přesně to vystihuje zaměření firmy, která se od roku 1949 specializuje na produkci nejrůznějších druhů stavebnic tvořených plastovými kostkami. Od roku 1986 pak LEGO vyvíjí produkty ovládané pomocí počítačové technologie. Když o dva roky později začala firma spolupracovat s MIT (Massachusetts Institute of Technology), začal vývoj inteligentní kostky, která v současnosti tvoří základ LEGO Mindstorms stavebnic.



Srdcem stavebnice LEGO Mindstorms je inteligentní kostka EV3

Pro koho jsou tyto stavebnice vhodné? Na tuto otázku se dá odpovědět velmi jednoduše: pro všechny. Zalíbí se klukům i holkám všech věkových kategorií. Jak těm, kteří již projevili zájem o svět informatiky, tak i takovým, kterým prozatím technologie nic moc neříkají. Pomocí těchto programovatelných stavebnic lze totiž vzbudit zvědavost a zájem o kódování opravdu v každém. Pokud se obáváte, že se nejedná o případ vašeho dítěte, jelikož jej zajímají spíše letadla na dálkové ovládání či štěňátka, nemusíte zoufat. Cestu k robotice a programování lze započít u libovolného koníčku, kterému se děti věnují. Každá jedna oblast zájmu jej totiž může inspirovat k postavení vlastního robota se specifickými vlastnostmi, který se může nějak hýbat, vydávat zvuky či napodobovat chování oblíbeného zvířete.

Sada LEGO BOOST (17101) je vhodná pro kluky i holky ve věku od 7 do 12 let. LEGO MINDSTORMS (31313) se pak hodí pro začínající i pokročilé děti od 10 let. Postavíte z ní až 17 robotů

Buldozer postavený z Lega Boost se pohybuje pomocí pásů a dokáže plnit nejrůznější úkoly

STAVEBNICE LEGO JAKO UČEBNÍ POMŮCKA

Kód pro roboty z Lega sestavujete pomocí bloků přímo v aplikaci

fantazie a vytvořit si vlastní projekt, všechny sady LEGO robotů obsahují i několik jednoduchých a srozumitelných návodů, co z přiložených součástek můžete sestavit a jakým způsobem svého robota následně rozpohybujete. Tyto sady existují dvě: LEGO Boost a LEGO Mindstorms. Sada **LEGO Boost** obsahuje návod k sestavení pěti různých robotů s různými funkcemi a je vhodná pro kluky i holky ve věku od 7 do 12 let. Oproti tomu sada **LEGO Mindstorms** disponuje návody a součástkami vhodnými ke stavbě sedmnácti robotických projektů a hodí se pro začínající i pokročilé děti od 10 let.

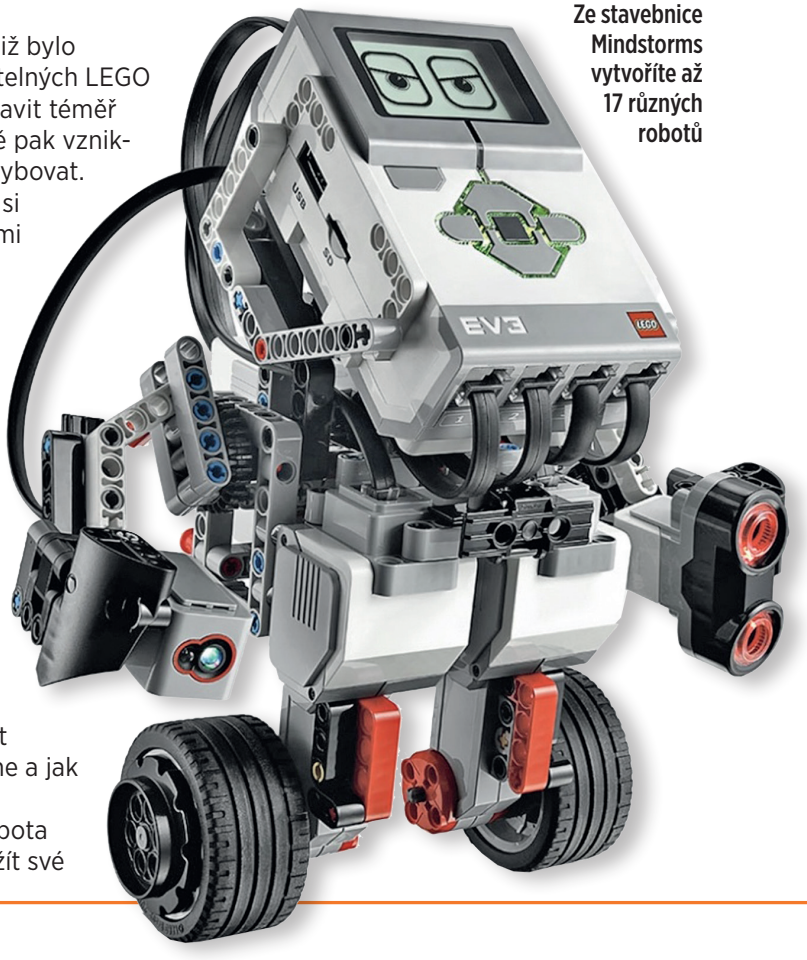
Obě zmíněné stavebnice jsou vhodné pro úplné začátečníky bez předchozích dovedností v oblasti programování a stejně tak jsou vhodné i pro dospělé, kteří se s nimi dokážou zabavit minimálně v takové míře jako děti.

Sady LEGO robotů doplňuje aplikace plná jednoduchých a srozumitelných návodů, co vše můžete ze součástek sestavit

Základy kódování

Jak LEGO roboti fungují? Jak již bylo řečeno, s pomocí programovatelných LEGO stavebnic můžete s dětmi sestavit téměř vše, co je napadne. A následně pak vzniklého robota jednoduše rozpohybovat. Děti všech věkových kategorií si díky speciálnímu softwaru velmi rychle osvojí kódovací dovednosti a s každým programem budou více a více rozvíjet své algoritmické myšlení. Použití tohoto softwaru je velice intuitivní. Ani vy ani vaše dítě nemusíte ovládat žádný složitý programovací jazyk, jelikož kód je sestavován pomocí bloků, které skládáte za sebe jako puzzle. Tyto bloky symbolizují různé pohyby, funkce či časování. Stačí je pouze naskládat za sebe v požadovaném pořadí, spustit program a sledovat, co se stane a jak váš robot ožívá.

Kromě toho, že při stavbě robota mohou malí programátoři využít své



Ze stavebnice Mindstorms vytvoříte až 17 různých robotů

Interakci s roboty zajišťuje prostřednictvím aplikace mobil nebo tablet

Jedním z pěti robotů, které můžete sestavit prostřednictvím sady LEGO Boost, je i chytrá elektrická kytara

Kytara nebo kocour

LEGO Boost je označení barevné sady, která se pyšní oceněním Technická hračka roku 2018. Jedná se o sérii kostek, dílků a návodů na stavění různých robotických modelů – elektrické kytary, kocoura Frankieho, nákladního auta, stroje na výrobu minirobotů a robota Vernieho.

V bezplatné aplikaci, ve které se mimo jiné nachází prostředí pro programování vašeho robota, najdete další pokyny a modely, které vašemu dítěti pomohou s návrhy a stavbou vlastních robotů, ať už zvířat, vozidel nebo čehokoli dalšího. Kódovací bloky v aplikaci fungují pro všechny modely stejně. Dále v ní můžete najít i zábavné programovací výzvy pro jednoho i více hráčů. Cílem je rozvíjet kreativitu a řešení problémů formou hry.

Stejně jako u sady Boost lze i roboty sestavené ze sady Mindstorms programovat prostřednictvím bezplatné aplikace, ve které se mimo užitečné návody a tipy nachází i vývojové prostředí pro tvorbu kódu pomocí programových bloků, které lze skládat za sebe. Na webových stránkách **www.lego.com** lze najít spoustu výukových materiálů doplněných o videa, která vás provedou základy základy programování a nabídnou nejrůznější tipy na stavění a vylepšení vašich robotů. Základní sada obsahuje 601 dílků, tři motory, řadu senzorů (např. infračervený, dotykový), ale hlavně má obrovskou uživatelskou základnu, díky které máte bezednou studnu inspirace stále po ruce.

Srdce robota vytvořeného v sadě Mindstorms je inteligentní kostka vybavená šesti tlačítky, černobílým displejem, vestavěným reproduktorem, portem USB, čtečkou karet, čtyřmi vstupními a čtyřmi výstupními porty. Kostka podporuje USB, Bluetooth a Wi-Fi komunikaci s počítačem a má programovací rozhraní.

Probuďte v sobě dítě

Technologie, počítače, roboti. To jsou slova, která nás obklopují denně a všude, a to se v nejbližší době nezmění. Jedna z věcí, které můžeme pro sebe i své děti udělat, je to, že půjdeme technologiím naproti a naučíme se s nimi zacházet, vynasnažíme se jim porozumět a umět s nimi fungovat. Jednou z cest, která nás k tomuto cíli může dovést, je právě využití robotických stavebnic, jako jsou LEGO Boost či LEGO Mindstorms. A nezapomínejte se bavit i vy, protože u dětí to zdaleka nekončí. Koneckonců, nepřestáváme si hrát, protože stárneme. Stárneme, protože si přestáváme hrát!

Literatura

[1] NÚV (sest.). *Návrh revizí ICT* [online]. Praha: Národní ústav pro vzdělávání, 2019 [cit. 2020-5-24]. 20 s. Dostupný z: ⟨http://www.nuv.cz/file/3362_1_1/⟩.

[2] FERRARI, Anusca (comp.). *DIGCOMP: A Framework for Developing and Understanding Digital Competence in Europe* [online]. Spain: Luxembourg: Publications Office of the European Union, 2013 [cit. 2020-5-24]. 50 s. Dostupný z: ⟨http://ftp.jrc.es/EURdoc/JRC83167.pdf⟩.

[3] PRŮCHA, Jan; WALTEROVÁ, Eliška; MAREŠ, Jiří. *Pedagogický slovník*. Praha: Portál, 2009. ISBN 978-80-7363-647-6.

[4] LESSNER, Daniel. Analýza významu pojmu „computational thinking. *Journal of Technology and Information Education* [online]. Olomouc, 2014, č. 6 [cit. 2020-5-24]. Dostupný z: ⟨http://www.jtie.upol.cz/pdfs/jti/2014/01/06.pdf⟩.

[5] ISTE; CSTA. Operational definition of computational thinking for K–12 education. 2011. Dostupný také z: ⟨https://id.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf⟩.

[6] RŮŽIČKOVÁ, Daniela. *Rozvíjíme ICT gramotnost žáků*. Praha: Národní ústav pro vzdělávání, školské poradenské zařízení a zařízení pro další vzdělávání pedagogických pracovníků (NÚV), 2011. ISBN 978-80-86856-94-0.

[7] WIKIPEDIA. *Vícevrstvá architektura* [online]. [cit. 2020-5-23]. Dostupný z: ⟨https://cs.wikipedia.org/wiki/V%5C%C3%5C%ADcevrstv%5C%C3%5C%A1_architektura⟩.

[8] WIKIPEDIA. *Front end a Back end* [online]. [cit. 2020-5-23]. Dostupný z: ⟨https://cs.wikipedia.org/wiki/Front_end_a_back_end⟩.

[9] ABZ.CZ, Slovník cizích slov. *Dramaturgie* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://slovnik-cizich-slov.abz.cz/web.php/hledat?cizi_slovo=dramaturgie&typ_hledani=prefix⟩.

[10] UKROP, Martin; ŠVÁBENSKÝ, Valdemar; NEHYBA, Jan. Reflective Diary for Professional Development of Novice Teachers. [online]. 2018, [cit. 2020-5-24]. Dostupný z: ⟨https://arxiv.org/pdf/1811.02965.pdf⟩.

[11] ABZ.CZ, Slovník cizích slov. *Precedens* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://slovnik-cizich-slov.abz.cz/web.php/hledat?cizi_slovo=precedens&typ_hledani=prefix⟩.

[12] WIKIPEDIA. *Code review* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://en.wikipedia.org/wiki/Code_review?oldid=753657014⟩.

[13] SLAVÍK, Jan. *Hodnocení v současné škole*. Praha: Portál, 1999.

[14] STARÝ, Karel. Formativní hodnocení: Zpráva o mezinárodním semináři OECD. *Pedagogika*. 2005, roč. 55, č. 4.

[15] WIKIPEDIA. *Pedagogická komunikace ve školní třídě* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://cs.m.wikipedia.org/wiki/Pedagogick%5C%C3%5C%A1_komunikace_ve_%5C%C5%5C%A1koln%5C%C3%5C%AD_t%5C%C5%5C%99%5C%C3%5C%ADd%5C%C4%5C%9B⟩.

[16] LITERACY, Adolescent. *Think, Pair, Share* [online]. [cit. 2020-5-17]. Dostupný z: ⟨http://www.adlit.org/strategies/23277/⟩.

[17] TEACHING EXCELLENCE, Centre for. *Multiple-Choice Questions* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://uwaterloo.ca/centre-for-teaching-excellence/teaching-resources/teaching-tips/developing-assignments/assignment-design/designing-multiple-choice-questions⟩.

[18] AZURE, Microsoft. *Cloud Computing* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://azure.%20microsoft.%20com/⟩.

[19] GOOGLE. *G Suite* [online]. [cit. 2020-5-17]. Dostupný z: ⟨https://gsuite.google.com/⟩.

[20] ORACLE. *Umělá inteligence* [online]. [cit. 2020-5-18]. Dostupný z: ⟨https://www.oracle.com/cz/artificial-intelligence/what-is-artificial-intelligence.html⟩.

[21] OVERBLOG. *HOW LONG DOES A BRUTE FORCE PASSWORD CRACK TAKE* [online]. [cit. 2020-5-18]. Dostupný z: ⟨https://www.google.com/url?sa=i&url=http%3A%2F%2Ftieketowhe.over-blog.com%2F2020%2F04%2FHow-Long-Does-A-Brute-Force-Password-Crack-Take.html&psig=AOvVaw039jQZPk59DbW9YtHZKOyP&ust=1589875822423000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCNCNx4_7vOkCFQAAAAAdAAAAABAD⟩.

[22] BĚLOHLÁVEK, Radim. *Algoritmická matematika 1* [online]. [cit. 2020-5-18]. Dostupný z: ⟨http://belohlavek.inf.upol.cz/vyuka/algoritmicka-matematika-1-1.pdf⟩.

[23] LEVITIN, Anany; LEVITIN, Maria. *Algorithmic Puzzles*. New York: Oxford University Press, 2011. ISBN 978-0199740444.

[24] GOOGLE, Code with. *Blockly Games* [online]. [cit. 2020-5-18]. Dostupný z: ⟨https://blockly.games/⟩.

[25] OZOBOT. *Ozobot Color codes* [online]. [cit. 2020-5-23]. Dostupný z: ⟨https://sites.google.com/a/dpsk12.net/samuelsstem/ozobot⟩.

[26] MUNI, FI. *Datový typ string* [online]. [cit. 2020-5-18]. Dostupný z: ⟨https://www.fi.muni.cz/~xsmerk/ib001/string.html⟩.

[27] ITNETWORK. *Cykly v JavaScriptu* [online]. [cit. 2020-5-18]. Dostupný z: ⟨https://www.itnetwork.cz/javascript/zaklady/javascript-tutorial-cykly-for-while⟩.

[28] ITNETWORK. *Funkce v JavaScriptu* [online]. [cit. 2020-5-18]. Dostupný z: ⟨https://www.itnetwork.cz/javascript/zaklady/tutorial-javascript-funkce⟩.