Univerzita Karlova

Pedagogická fakulta

Katedra informačních technologií a technické výchovy

DIPLOMOVÁ PRÁCE

Podpora výuky programování na ZŠ

Programming at elemetary school

Bc. Radek Vinický

Vedoucí práce:	PhDr. Jiří Štípek, Ph.D.
Studijní program:	Učitelství pro střední školy (N7504)
Studijní obor:	Učitelství VVP pro ZŠ a SŠ - Informační a komunikační technologie

2018

Odevzdáním této diplomové práce na téma Podpora výuky programování na ZŠ potvrzuji, že jsem ji vypracoval pod vedením vedoucího práce samostatně za použití v práci uvedených pramenů a literatury. Dále potvrzuji, že tato práce nebyla využita k získání jiného nebo stejného titulu.

V Praze dne 6. 12. 2018

.....

podpis

Rád bych vyjádřil poděkování PhDr. Jiřímu Štípkovi, Ph.D. za odborné vedení, cenné rady a věcné připomínky, které mi pomohly závěrečnou práci zkompletovat. Děkuji mé rodině, především manželce za toleranci a podporu, tolik potřebnou k sepsání práce.

NÁZEV

Podpora výuky programování na ZŠ

AUTOR

Bc. Radek Vinický

KATEDRA

Katedra informačních technologií a informační výchovy

VEDOUCÍ PRÁCE

PhDr. Jiří Štípek, Ph.D

ABSTRAKT

Diplomová práce se zabývá výukou programování na základních školách v České republice. Teoretická část práce se zaměřuje na mapování současného stavu výuky programování na základních školách. Dále jsou v práci mapovány aktuálně využívané zdroje, které jsou určeny pro podporu výuky algoritmizace a programování dětí. Tyto zdroje jsou dále hodnoceny vybranými evaluačními kritérii. Na teoretickém základu práce je navrhnut vlastní model výuky programování jako praktický výstup, a to včetně výukových materiálů a metodické příručky pro učitele. Návrh modelu výuky je ověřen v praxi.

KLÍČOVÁ SLOVA

algoritmizace, programování, model výuky, informační a komunikační technologie

TITLE

Programming at elemetary school

AUTHOR

Bc. Radek Vinický

DEPARTMENT

Department of information technology and education

SUPERVISOR

PhDr. Jiří Štípek, Ph.D

ABSTRACT

The diploma thesis deals with the teaching of programming at elementary schools in the Czech Republic. The theoretical part of the thesis focuses on mapping the current state of the teaching of programming at elementary schools. In addition, the currently used resources to support the teaching of algorithmization and programming for children are mapped. These resources are evaluated by selected evaluation criteria. On the theoretical basis of thesis, a teaching model focused on programming is designed as a practical output, including learning materials and a methodological manual for teachers. The design of the model of teaching is verified in practice.

KEYWORDS

algorithmization, programming, model of teaching, Information and Communication Technology

Obsah

Úvod

Společnost vyspělé části světa se ocitá v novém postindustriálním věku. Charakteristickým prvkem této doby je využívání informačních technologií a digitalizace obecně, a to ve většině odvětví lidské činnosti. Komerční společnosti zabývající se průmyslovou výrobou se postupně odklánějí od standardních procesů výroby na montážních a výrobních linkách a začínají ve výrobě využívat sítí řízené kyberneticko-fyzikální systémy, Internet věcí a umělou inteligenci. V souvislosti s tím zanikají pracovní pozice zaměstnancům s nízkou kvalifikací – operátorům výrobních linek a vznikají místa nová. Podobný trend je očekáván také ve stavebnictví, v důsledku nasazení 3D tisku, ale také v mnoha dalších odvětvích. Nová pracovní místa však budou obsazována kompetentními zaměstnanci s nutně vyšší kvalifikací, schopnostmi řešit problémy, umět pracovat s technologiemi, pracovat v týmech či týmy řídit, učit se novým poznatkům, dovednostem a dále se rozvíjet. Také děti postupně opouštějí hry s panenkami a autíčky. Ty jsou nahrazovány mobilními technologiemi, které u dětí získávají zásadní postavení.

V takové společnosti se mění i úkol škol. Hlavní úlohou je připravit (co možná nejlépe) své žáky na budoucí život občanů společnosti, kteří budou pro společnost prospěšní. Nikdo však přesně neví, co konkrétně děti v budoucnu čeká, jaké nové profese vzniknou. Krueger v článku na webu organizace ISTE [cit. 2018-10-17] informuje o statistice uvádějící: *„65 % dětí, které začínají plnit školní docházku, bude pracovat v zaměstnání, která dnes ještě neexistují.“* Jistý je rozhodně trend narůstání dostupnosti informací, a proto se práce s informacemi – informační gramotnost stává nedílnou součástí nutných dovedností každého aktivního občana. Ten by se v tomto prostředí měl dobře orientovat a využívat ho jak v profesním, tak v osobním životě.

Obsah výuky ve školách v 21. století by neměl z velké části směřovat k dovednosti ovládat kancelářský software, ale spíše k dovednostem, se kterými nás ve svém materiálu seznamuje ISTE/CSTA (2011), tedy formulovat a řešit problémy, automatizovat řešení, algoritmicky myslet, najít nejefektivnější možné řešení na základě identifikace a analýzy problému atd. Programování, respektive algoritmizace osvojování takových dovedností ve velké míře podporuje. Navíc, jak uvádí Resnick (2012) ve své řeči na TEDxBeaconStreet, tvorba kódu: *„pomáhá dětem učit se ve smysluplném kontextu. Ukazuje, že učení je proces, nikoli produkt. Učí děti být*

vytrvalými, i když věci nefungují dobře. Podporuje kreativitu a umožňuje komunikovat se světem kolem sebe.“

1. Cíle práce

Hlavním cílem práce je navrhnout model výuky algoritmizace a základů programování aplikovatelný v rámci informatického předmětu nebo zájmového útvaru. Do návrhu modelu výuky budou zařazeny aktuálně využívané a odborníky odkazované edukační zdroje, které jsou určeny pro podporu výuky algoritmizace a programování pro děti. Při výběru konkrétních zdrojů bude brán zřetel na preference a potřeby českých učitelů, ale především na věk žáků 6. a 7. ročníků základní školy, pro které bude model výuky navrhován a kteří se zúčastní ověřování v praxi. Zdroje budou redukovány na základě celkového hodnocení pomocí hodnotících kritérií. V rámci práce bude vytvořena webová stránka obsahující výukové materiály. Dále bude vytvořena metodická příručka pro učitele představující návrh, jak s žáky při průchodu výukovými materiály konkrétně postupovat. Navrhovaný model bude ověřen v praxi pilotním nasazením v rámci zájmového útvaru, a to v období devíti měsíců.

2. Výzkumné otázky a úkoly práce

V souvislosti s cíli diplomové práce byly stanoveny následující výzkumné otázky a výzkumné úkoly.

2.1 Výzkumné otázky

Otázka 1: Jaká je současná situace výuky algoritmizace a programování na základních školách?

Je na základních školách realizována výuka zaměřující se na oblast algoritmizace a programování? Je v tomto ohledu patrný rozdíl mezi velkými, středně velkými a malými školami? Kolik vyučovacích hodin této problematice učitelé věnují? Jaké zdroje při výuce využívají? Jaká prostředí preferují a proč? V rámci jakého ročníku s výukou algoritmizace či programování začínají?

Odpovědi na tyto dílčí otázky by nám měly pomoci s představou o současné situaci výuky algoritmizace a programování. Získané informace pak poslouží jako teoretická báze, na níž bude navrhovaný model stavěn.

Otázka 2: Jaké zdroje pro podporu výuky algoritmizace a programování určených dětem jsou vhodné?

V současnosti má učitel k dispozici značné kvantum zdrojů, které může ve výuce využívat, a to bez ohledu na preferované pojetí výuky. Rozdíl je především v kvalitě zdrojů. Proto je důležité znát názor odborné veřejnosti na vhodné edukační zdroje, respektive na možnosti jejich evaluace, neboť právě takové zdroje budou v práci nejprve hodnoceny pomocí evaluačních kritérií a zdroje s nejlepšími výsledky budou využívány v navrhovaného modelu výuky.

Otázka 3: Jsou do modelu vybrány aktivity přiměřené možnostem žáků 6. a 7. ročníku?

Náročnost aktivit v navrhovaném modelu výuky bude mít stoupající tendenci. Model by měl umožňovat jistou individuální přiměřenost[1]. Náročné aktivity budou vhodné především pro skupinu nadaných žáků. Otázkou však je, zda jsou navrhované aktivity svou obtížností přiměřené kognitivnímu vývoji žáků (11–13 let)?

Otázka 4: Zvýší se u žáků účastnících se ověřovacího procesu zájem o programování?

Bude mít absolvování výuky podle navrhovaného modelu na žáky pozitivní dopad? Vzroste jejich zájem o další rozvoj vlastních dovedností v oblasti programování? Odměnou za vynaloženou snahu by měly být funkční hry vytvořené vlastními silami, soutěže či úspěšný průchod všemi úrovněmi hry. Vliv na vnější motivaci žáků by měl mít způsob hodnocení založený na gamifikačních prvcích.

2.2 Výzkumné úkoly

Úkol 1: Zmapovat aktuální stav výuky programování na základních školách v České republice.

- Analyzovat rámcový vzdělávací program pro základní vzdělávání.
- Analyzovat empirické studie.
- Analyzovat diplomové práce.
- Porovnat účast českých škol v informatické soutěži Bobřík informatiky v určitém časovém období.

[1] Činnost je přiměřená s ohledem na osobní dispozice a možnosti žáka.

- Provést dotazníkové šetření.
    - Vytvořit dotazník vhodný k zjištění aktuálního stavu výuky programování na ZŠ.
    - Vybrat školy, kterým bude zaslán elektronický dotazník s prosbou o vyplnění.
    - Analyzovat výsledky dotazníkového šetření.

Úkol 2: Zmapovat dostupné zdroje vhodné pro podporu výuky algoritmizace a základů programování.

- Zmapovat situaci využívaných zdrojů pro podporu výuky algoritmizace a programování v ČR i v zahraničí.
- Vybrat aktuálně nejvíce využívané zdroje pro podporu výuky algoritmizace a programování.

Úkol 3: Zhodnotit vybrané zdroje pro podporu výuky algoritmizace a základů programování.

- Zmapovat evaluační kritéria vhodná pro hodnocení vybraných zdrojů a vytvořit vlastní evaluační systém.
- Zhodnotit vybrané zdroje.

Úkol 4: Navrhnout vlastní model výuky.

- Navrhnout strukturu modelu včetně hodinové dotace, očekávaných výstupů a návrhu na hodnocení žáků.
- Vytvořit výukové materiály.
- Realizovat přístup k vytvořeným výukovým materiálům.
- Vytvořit metodickou příručku pro učitele.

Úkol 5: Ověřit navržený model v praxi.

- Určit obtížnost aktivit navrhovaného modelu.
- Ověřit, zda bude mít pilotní nasazení na žáky pozitivní dopad ve smyslu zvýšeného zájmu o problematiku programování, a to i v rámci informálního učení.

Úkol 6: Stanovit závěry a případnou modifikaci modelu výuky.

3. Současný stav výuky programování na základních školách v České republice

Při návrhu vlastního modelu výuky by měly být brány v potaz aktuální reálné situace výuky. Před vytvořením modelu je tedy důležité bližší seznámení se a pochopení situace výuky oblasti algoritmizace a programování na státních základních školách v České republice. Při návrhu vlastního modelu pak budou zjištěné poznatky reflektovány, model bude vytvářen také s ohledem na preference a potřeby učitelů.

Zmapování současného stavu výuky programování v této práci probíhá ve čtyřech krocích:

- analýzou rámcového vzdělávacího programu pro základní vzdělávání,
- analýzou zdrojů zabývajících se stavem výuky základů programování a algoritmizace na základních školách,
- porovnáním účasti českých škol v informatické soutěži Bobřík informatiky v období 2010–2017,
- zmapováním současného stavu výuky algoritmizace a programování na ZŠ dotazníkovým šetřením.

Výše zmíněné první tři kroky plní především funkci předběžné teoretické analýzy, kterou je vhodné před samotným dotazníkovým šetřením uskutečnit (Chráska, 2007, s. 12).

3.1 Analýza rámcového vzdělávacího programu pro základní vzdělávání

Zatím stále platným hlavním kurikulárním dokumentem státní úrovně je od 1. 9. 2005 rámcový vzdělávací program pro základní vzdělávání (dále i „RVP ZV“). Průcha (2003, s. 197) definuje v pedagogickém slovníku RVP jako kurikulární dokumenty navazující na Bílou knihu, které vymezují cílové zaměření vzdělávání a které jsou charakterizovány prioritami, cíli, klíčovými kompetencemi a obsahem v oblastech vzdělávání. Rámcový vzdělávací program tvoří závazný rámec pro vytváření školních vzdělávacích programů – dokumentů školní úrovně. V RVP ZV se nově objevila také vzdělávací oblast Informační a komunikační technologie. Mezi cílovým zaměřením vzdělávací oblasti Informační a komunikační technologie vedoucí k osvojování

klíčových kompetencí žákem najdeme následující bod: *„schopnosti formulovat svůj požadavek a využívat při interakci s počítačem algoritmické myšlení.“* (RVP ZV, 2016, s. 38) Přestože jde již o třetí revizi RVP ZV, zůstává tato čtrnáct let stará vzdělávací oblast včetně obsahu vzdělávacího oboru bez výraznějších změn. Samotný obsah, tedy očekávané výstupy a učivo oboru Informační a komunikační technologie, problematiku algoritmizace nebo základů programování zcela vynechává. Jisté prvky algoritmizace bychom mohli vtěsnat do učiva tabulkový editor, prezentace informací (webové stránky). Výše zmíněný cíl oblasti je tak zatím jedinou částí, která algoritmizaci, potažmo informatické myšlení v RVP ZV zmiňuje. Vzdělávací obsah RVP ZV v oblasti Informační a komunikační technologie neodpovídá současným světovým trendům vzdělávání ani požadavkům společnosti, nereflektuje intenzivní vývoj informačních a komunikačních technologií. Na tom se již nějaký čas shodují vesměs všichni odborníci v oblasti vzdělávání (např. Neumajer, 2009). Problematickou se jeví také situace časové dotace oboru Informační a komunikační technologie, která na základní škole představuje 1 hodinu týdně na prvním a 1 hodinu týdně na druhém stupni v jednom vybraném ročníku.

Problematikou se aktuálně zabývá koncepce Strategie digitálního vzdělávání do roku 2020 (dále jen SDV), která se, kromě jiného, ve svém druhém bodě, konkrétně v opatřeních 2.1–2.3, zaměřuje na intervenci do oblasti rozvoje informatického myšlení a digitální gramotnosti žáků. Předkládá návrhy opatření, jež by měly situaci přežilého obsahu zlepšit, a to včetně revize RVP ZV (SDV, 2014, s. 22–25). Podle SDV měl být tento směr intervence realizován do konce roku 2017 (SDV, 2014, s. 41).

3.2 Analýza vědeckých prací a diplomové práce

Získat přehled a informovat se o stavu výuky algoritmizace a programování je možné ze závěrečných prací studentů univerzit, kteří danou oblast zájmu zmapovali. Také samotné univerzity, respektive katedry pedagogických fakult, realizují v určitých časových intervalech vědecké výzkumy týkající se stavu výuky vybraných oblastí a následně jistý pohled na situaci předkládají. V této práci jsou vybrány dvě empirické studie Katedry informačních technologií a technické výchovy Pedagogické fakulty Univerzity Karlovy:

- Informačně technologické kompetence dětí a jejich rozvoj na základních školách.

- Vybrané výsledky výzkumu rozvoje digitálních kompetencí žáků na ZŠ.

Dále jsou v práci představeny výsledky dotazníkového šetření, jež jsou součástí diplomové práce Výuka základů programování v prostředí Scratch.

3.2.1 Výběr výsledků empirické studie Informačně technologické kompetence dětí a jejich rozvoj na základních školách

Autoři Štípek a Vaňková (2014) v empirické studii informují o situaci výuky informačních předmětů na základních školách. Přináší pohled na výuku se zaměřením na výukové aktivity informatického charakteru, rozložení hodinových dotací povinného předmětu informatika, významnost obsahu (jednotlivých tematických celků), pohled učitelů na významnosti ICT, kompetencí žáků či hodnocení digitálních kompetencí učitelů.

Rozložení hodinových dotací a období začátku výuky algoritmizace a základů programování

Z nasbíraných dat vyplynulo, že na prvním stupni základních škol začíná výuka převážně v 5. ročníku s dotací 1 hodiny týdně. Na 2. stupni je výuka rozložena rovnoměrněji, nejčastěji je výuka zařazována do 6. ročníku. Na 2. stupni bylo možné pozorovat nárůst o jednu (disponibilní) hodinu týdně (Tabulka 1).

1. stupeň ZŠ						
		1. roč.	2. roč.	3. roč.	4. roč.	5. roč.
hodin týdně	1	2,7%	5,1%	12,1%	33,8%	**88,8%**
	2	0,4%	0,3%	0,5%	1,3%	2,9%
	3	0,1%	0,4%	0,3%	0,4%	0,3%
2. stupeň ZŠ						
		6. roč.	7. roč.	8. roč.	9. roč.	
hodin týdně	1	**75,7%**	60,3%	51,9%	48,3%	
	2	12,7%	12,5%	12,7%	14,3%	
	3	0,4%	1,0%	1,7%	2,0%	

Tabulka 1: Hodinová dotace na 1. a 2. stupni (Štípek a Vaňková, 2014)

74 % učitelů si není jistých nejvhodnějším obdobím začátku výuky algoritmizace a základů programování – rozvoje algoritmického myšlení. Jen 2,4 % z nich by výuku zařadilo již na 1. stupeň, 8,7 % by s výukou začalo na 2. stupni, a 14,9 % uvedlo, že s výukou by se mělo začít až od 3. stupně.

Tyto názory jsou však v rozporu se strategiemi výuky vyspělých států. Například Slovensko od roku 2007 zařazuje informatická témata již od 2. ročníku ZŠ. Národní kurikulum v Anglii doznalo v roce 2014 značné změny (Kemp, 2014, s. 4–5). Od školního roku 2014/2015 byl předmět ICT nahrazen předmětem Computing propojujícím tři navzájem propojené oblasti: Computer science (počítačovou vědu), Information technology (informační technologii), Digital literacy (digitální gramotnost). Nový studijní program cílí na programování, ale i na další aspekty počítačové vědy. Jádrem celého studijního programu se stává Computational thinking (informatické myšlení). Ve Finském kurilulu můžeme sledovat multidisciplinaritu algoritmizace a programování – pronikání do různých oblastí již od nejnižších ročníků, ve kterých lze spatřit základy programování (bez počítače), například ve výtvarné výchově – tvorba vzorů nebo pletení, v tělesné výchově – sestavování a obměňování sekvencí pohybů nebo v literatuře – správná formulace požadavků (Deruy, 2017). Například Dohnal (2009, s. 37) ve své diplomové práci odkazuje na článek Rudolfa Pecinovského z internetového portálu Česká škola, kde se uvádí: *„žáci jsou schopní pochopit základy algoritmizace a vytvářet jednoduché programy (cca 20 – 50 příkazů) již v první třídě.“*

Význam programování

Učitelé hodnotili míru významnosti tematických celků z hlediska rozvoje informační gramotnosti. Nejméně významným celkem byly podle učitelů (společně s tvorbou a využitím databází) algoritmizace a základy programování – rozvoj algoritmického myšlení. Téma bylo respondenty považováno za postradatelné z pohledu rozvoje příslušných kompetencí žáků. Autoři (Štípek a Vaňková, 2014, s. 57) uvádějí: *„Ve skupině „okrajových“ celků je pozoruhodné především postavení Algoritmizace a základů programování, kdysi dominantní a cílové dovednosti v oblasti počítačového vzdělávání. Z hlediska geneze zařazování počítačových technologií do výuky na školách dokládají získané výsledky transformaci pojetí založeného na klíčovém významu algoritmizace a programování směrem k pojetí založenému na uživatelském zvládnutí základních aplikací a nástrojů.“*

Algoritmizace a programování stálo podle výsledků výzkumu před čtyřmi roky na samém okraji zájmu učitelů informační a komunikační technologie. Učitelé kladi jen velmi malý důraz na prohlubování schopností (ICT kompetencí) žáků myslet

algoritmicky, umět formulovat pokyny a instrukce, což potvrzuje také Vaníček (2012, s. 17) ve výzkumu postojů budoucích učitelů k zařazení informatických témat do výuky.

3.2.2 Vybrané výsledky výzkumu rozvoje digitálních kompetencí žáků na ZŠ

Autoři (Štípek, Rambousek a Vaňková, 2015) představují výsledky výzkumu, který byl realizován na Pedagogické fakultě UK v rámci šetření projektu GAČR. Výzkum byl orientován na problematiku informatické výchovy na ZŠ ČR. Ve výzkumu se autoři zabývají významností 11 tematických celků, včetně Algoritmizace a základů programování, jejich náročností a atraktivitou, a to jak z pohledu učitelů, tak žáků. Respondenti odpovídali na otázky určením významu na stupnici 0–100. Šetření se účastnilo celkem 1183 učitelů informaticky orientovaných povinných předmětů a 2173 žáků ze 112 škol.

Výsledky výzkumu opět nejsou pro oblast programování příliš příznivé. Z pohledu významu tematického celku Algoritmizace a základy programování v porovnání s ostatními tematickými celky ze strany učitelů dopadl nejhůře. Učitelé tematickému celku přiřadili nejnižší významnost. Chápou programování jako nejobtížnější z tematických celků z pohledu náročnosti na učení žáka, považují tematický celek jako nejméně atraktivní v porovnání s bezpečností, autorským právem, etikou, hardwarem a softwarem počítačů, prací s počítačovou grafikou, tabulkovým kalkulátorem, textovým editorem, zvukem a videem na počítači, vytvářením webových stránek, prezentací, vyhledáváním a získáváním informací nebo se základními uživatelskými dovednostmi a správou souborů.

3.2.3 Výběr výsledků dotazníkového šetření diplomové práce Výuka základů programování v prostředí Scratch

Krejsa (2014, s. 13–17) v rámci své diplomové práce zmapoval stav výuky programování formou dotazníkového šetření, kterého se zúčastnilo 401 respondentů. Z šetření vyplynulo, že jen 57 učitelů základních škol (14%) zařazovalo do výuky informatiky obsah základy programování, a to především v devátém (34 odpovědí) a osmém (20 odpovědí) ročníku. 344 učitelů (86 %) se základům programování ve výuce nevěnovalo, přičemž se nejvíce odvolávali na nepřítomnost obsahu v RVP ZV

(220 dopovědí). Učitelé, kteří vyučovali základy programování, pro výuku nejčastěji využívali prostředí:

- Baltík/Baltie (24 odpovědí),
- Imagine Logo (11 odpovědí),
- PHP, Easy Logo, Pascal (po 2 odpovědích).

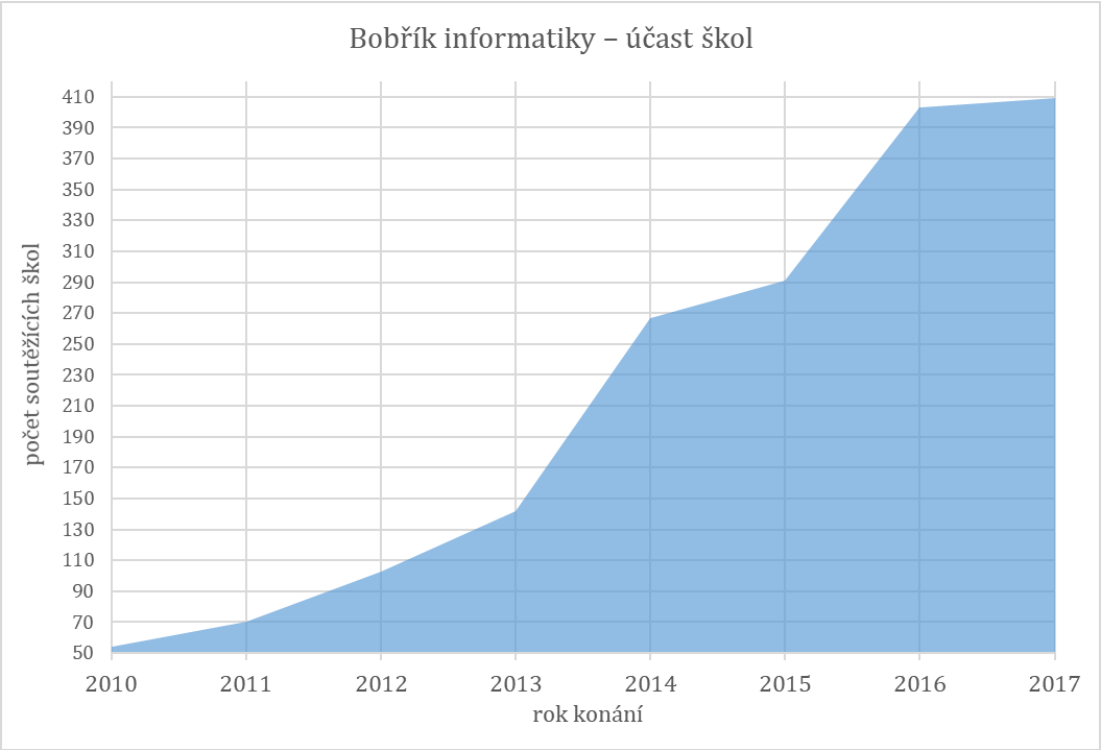
Přestože bylo Krejsovo šetření realizováno v roce 2014, ani jeden respondent nezmínil fenomén block-based programování Scratch, přičemž v tomto období již byla k dispozici také online verze 2.0.

Výsledky šetření s sebou přinesly pozitivní predikci stavu výuky programování, neboť z 344 učitelů, kteří nezařazovali programování do výuky, jich o tom 28 (8 %) aktuálně uvažovalo a 139 (40 %) učitelů o tom bylo ochotno alespoň uvažovat.

3.2.4 Analýza účasti českých škol v informatické soutěži Bobřík informatiky

Bobřík informatiky je soutěž, která cílí na seznamování žáků i jejich učitelů s informatickými problémy. Řešení těchto problémů vede k osvojování informační gramotnosti. Pro žáky základních škol jsou k dispozici tři kategorie rozlišující věk žáků. Řešitelům poskytuje nejen okamžitou zpětnou vazbu ve formě správného řešení, ale také informace o souvislosti úlohy s vědním oborem Informatika.

Účast škol v soutěži je dobrovolná. Počet škol (z celku 4155 škol v roce 2017) mezi roky 2010 až 2017 naznačuje trvalý růst významnosti informatiky, informatického myšlení, světa algoritmů a základů programování na školách (Graf 1). Sebraná data o počtu škol jsou dostupná na webu soutěže ibobr.cz, v sekci Bobří školy na mapě [cit. 2018-09-15].



Graf 1: Účast základních škol v soutěži Bobřík informatiky za období 2010–2017

3.3 Zmapování aktuálního stavu výuky algoritmizace a programování na ZŠ dotazníkovým šetřením

Již bylo zmíněno, že je algoritmizace v rámcovém vzdělávacím programu zúžena na absolutní minimum. Tematické plány však mohou překročit rámec učiva definovaný v RVP. Učitelé tak mohou problematiku programování do školních dokumentů zapracovat. Učitelé také využívají možnosti začlenit výuku algoritmizace, programování a informatického myšlení do zájmových útvarů, které realizují v rámci školních klubů. Skutečný stav se tedy může od závazného rámce odlišovat „nestandardním" učivem v rámci výuky povinného, či volitelného předmětu, ale také v rámci volnočasové aktivity pod hlavičkou školního klubu.

3.3.1 Výběr vzorku

Zmapování aktuálního stavu výuky programování na základních školách v České republice bylo uskutečněno kvantitativní metodou. Metodou sběru dat bylo zvoleno CAWI – Computer Assisted Web Interviewing[2] cílené na učitele základních škol. V prostředí Google Forms byl sestaven dotazník (Příloha 1). K získání emailových

[2] CAWI – Computer Assisted Web Interviewing – dotazníkové šetření v prostředí webu.

adres respondentů bylo využito adresáře škol a školských zařízení z webu MŠMT CZ *„Výběr z adresáře škol a školských zařízení“* [cit. 2018-03-29]. Samotné dotazníkové šetření bylo realizováno v období od 4. 4. 2018 do 16. 4. 2018. Náhodným losem[3] bylo vybráno a osloveno 2000 škol z celkového počtu 2404 běžných základních škol[4] emailovou výzvou. Na výzvu odpovědělo, a dotazníkového šetření se zúčastnilo celkem 489 respondentů – učitelů informačních a komunikačních technologií, resp. vedoucích zájmových útvarů základních škol z České republiky (jeden respondent za školu). Návratnost dotazníkového šetření činila 24,45 % z celkového počtu oslovených škol. Bližší rozložení počtu respondentů podle velikosti škol je v tabulce níže (Tabulka 2). Dělení škol podle počtu žáků vychází z článku České školní inspekce (2015).

Velikost školy (podle počtu žáků)	Malé školy (do 100 žáků)	Středně velké školy (101 až 300 žáků)	Velké školy (nad 300 žáků)
Absolutní četnost respondentů	77	158	254
Relativní četnost respondentů	15, 75 %	32,31 %	51,94 %

Tabulka 2: Četnosti respondentů podle velikosti školy

Celkový počet základních škol ve školním roce 2017/2018 činil podle MŠMT CZ, odboru školské statistiky, analýz a informační strategie (2018) celkem 3823 běžných základních škol, z toho s druhým stupněm 2404 škol. Referenční vzorek se tak rovnal 20,34 % škol s druhým stupněm z různých částí České republiky. Nebyl však zajištěn vzorek škol odpovídající svou strukturou reálné situaci počtu škol podle velikosti či geografického rozložení. Nelze tedy hovořit o reprezentativním vzorku.

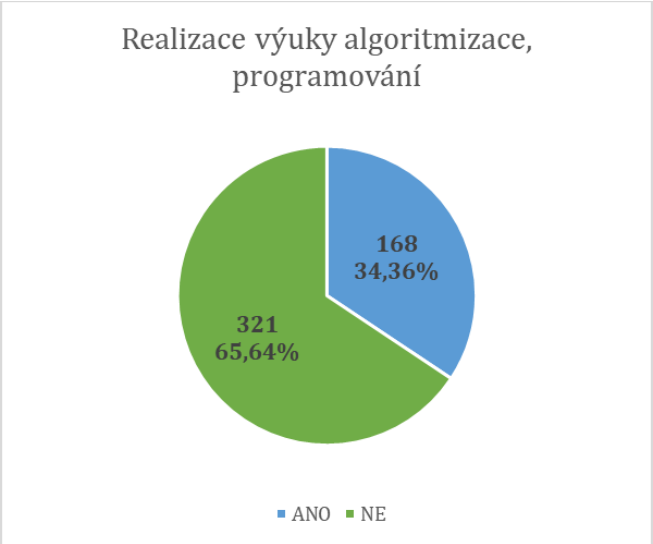
[3] K losování byl využit Microsoft Office Excel 2016, konkrétně funkce RANDBETWEEN.

[4] Státní škola umožňující ukončení povinné školní docházky – nižší sekundární vzdělání (mezinárodně označované jako stupeň ISCED 2). Ve škole jsou žáci vyučováni podle školního vzdělávacího programu závazně vycházejícího z RVP ZV.

3.3.2 Výsledky dotazníkového šetření

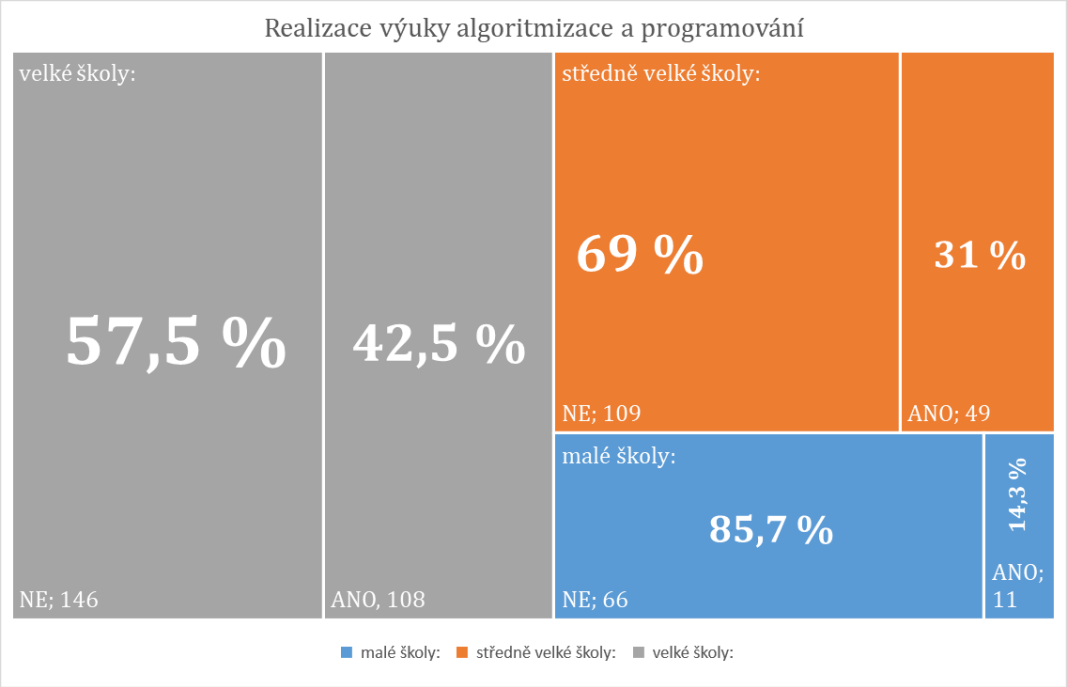
Realizace výuky

Celkem 321 respondentů v dotazníku uvedlo, že se na jejich škole výuka programování nerealizuje, a to v žádné formě. Naopak na 168 základních školách je výuka realizována, a to alespoň v minimálním rozsahu jedné hodiny (Graf 2).



Graf 2: Realizace výuky algoritmizace, programování

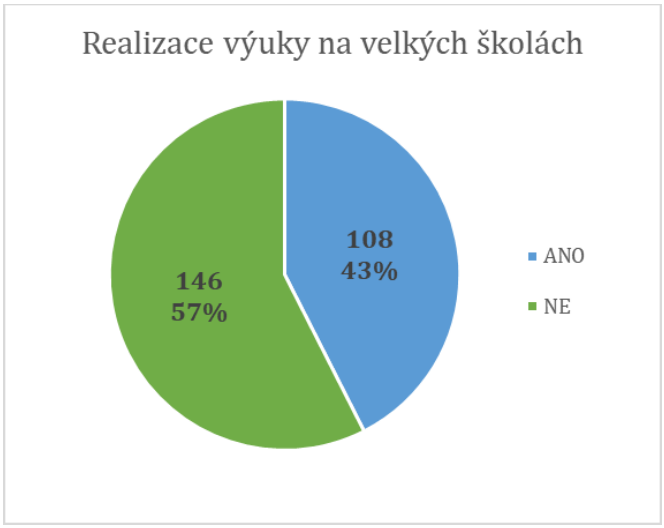
Při bližším pohledu na odpovědi respondentů je patrné, že s rostoucí velikostí školy z pohledu počtu žáků také vzrůstají tendence zařazovat programování do výuky. Ony tendence jsou patrné také v následujícím grafu (Graf 3).



Graf 3: Realizace výuky algoritmizace na programování

Na malých školách byly kladné odpovědi k záporným v poměru 11 : 66, tedy pouze 14,3 % škol zařazuje programování do výuky, zatímco 85,7 % ne. Odpovědi učitelů středně velkých škol pak byly v poměru 49 : 109. 31,0 % škol se programováním zabývá, 69,0 % ne. Na velkých školách je situace výuky programování ještě lepší. Poměr kladných odpovědí k záporným činí 108 : 146, tedy 42,5 % škol programování do výuky zařazuje, 57,5 % nikoliv.

Algoritmizaci a programování se na základních školách věnuje 108 z 254 učitelů velkých škol, což činí 42,51 %. Přestože počet škol vyučujících algoritmizaci a programování nepřekročil 50 %, je i tato situace vcelku příznivá (Graf 4). Nicméně vzorek není reprezentativní, proto nelze na základě uvedených zjištění usuzovat na skutečnou situaci v ČR. Zapojení respondentů bylo navíc dobrovolné a lze předpokládat, že učitelé, kteří se ve své výuce programování věnují, se zapojili do dotazníkového šetření ve větší míře, než odpovídá jejich skutečnému zastoupení mezi všemi učiteli Informatiky.



Graf 4: Realizace výuky algoritmizace a programování na velkých školách (nad 300 žáků)

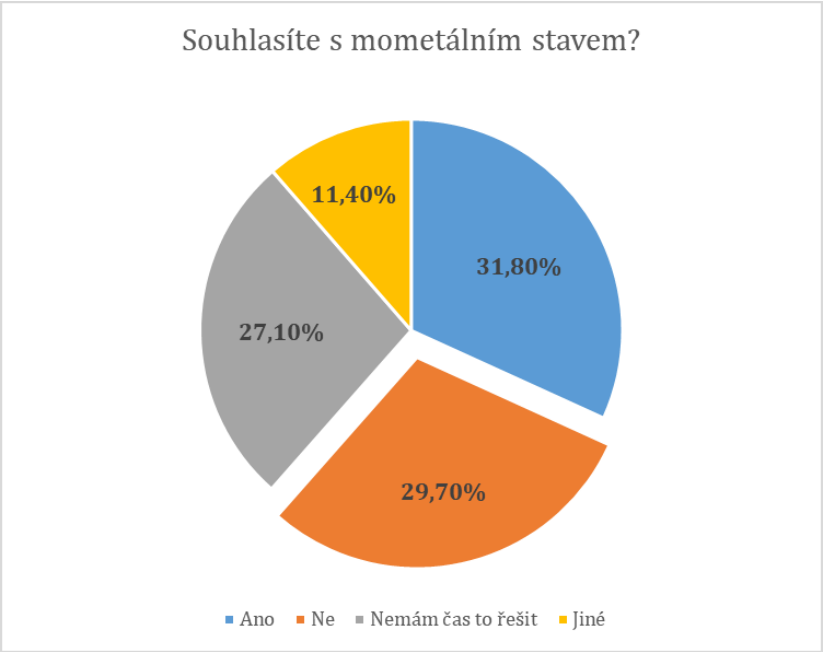
Nejčastěji zmíněné důvody opomíjení výuky algoritmizace a programování na základních školách jsou obsaženy v následující tabulce (Tabulka 3) a seřazeny podle výskytu. Vyučující nadpoloviční většiny nezařazují algoritmizaci a programování do výuky z důvodu absence obsahu v rámcovém vzdělávacím programu a na něj navazujících dokumentech. Dalšími zmíněnými faktory jsou náročnost výuky a nízká odborná kvalifikace (dříve aprobovanost) vyučujících informační a komunikační technologie na základních školách. Ostatní důvody vyjádřené respondenty dotazníku jsou obsahem přílohy (Příloha 2).

Důvod	Absolutní četnost odpovědi	Relativní četnost odpovědi
Problematika není v RVP (ŠVP)	166	51,71 %
Náročnost výuky	121	37,69 %
Neaprobovanost vyučujícího	120	37,38 %
Malá podpora – chybí učebnice, metodiky, …	100	31,70 %
Algoritmizace a programování nepatří na ZŠ	44	13,70 %
Nezájem žáků	42	13,08 %

Tabulka 3: Nejčastější důvody opomíjení výuky programování

Problematika neaprobovaných učitelů oblasti ICT je patrná také z výročních zpráv České školní inspekce. Ve výroční zprávě za školní rok 2013/2014 dosahovala aprobovanost výuky v oblasti ICT pro 1. stupeň 30,2 % a pro 2. stupeň 31,6 %, což byla nejnižší hodnota ze všech sledovaných oblastí a předmětů (Zatloukal, 2014, s. 142). Výroční zpráva za školní rok 2015/2016 již data o aprobovanosti učitelů v oblasti ICT pro 1. stupeň neposkytuje. Aprobovanost učitelů 2. stupně vzrostla v průběhu dvou let na 41,0 % (Zatloukal, 2016, s. 190). Přestože nárůst aprobovanosti byl během dvou let téměř 10%, oblasti ICT v tomto ohledu patří stále poslední příčka.

Pokud respondenti odpověděli na otázku týkající se realizace výuky negativně, byla jim na závěr poskytnuta možnost tento stav reflektovat. Ukazatelem možné změny směrem ke zlepšení stavu a zavedení obsahu algoritmizace a programování do výuky je odmítavý postoj bezmála 30 % respondentů (Graf 5).



Graf 5: Postoj k aktuálnímu stavu výuky algoritmizace a programování ve škole

Jiné, individuální odpovědi jsou obsahem přílohy (Příloha 3). Lze je zobecnit do tří skupin:

1. nízká hodinová dotace to znemožňuje;
2. problematika neaprobovanosti, chybí učitel;
3. programování neučit plošně, ale jen na specializovaných základních školách s rozšířenou výukou.

Na školách všech typů převládá výuka algoritmizace a programování jako okrajové téma povinných předmětů. Neočekávaný je značný rozptyl počtu hodin, v extrémech 1 až 150 hodin. Nejvíce času věnují tematickému celku na velkých základních školách, nejméně pak na malých školách (Tabulka 4).

Počty vyučovacích hodin						
Typ školy	Výskyt odpovědí		Průměrný počet hodin	Směrodatná odchylka	Extrémní hodnoty	
	Maximálně 5 hodin	Minimálně 6 hodin			Minimum	Maximum
Malé školy	9	2	8,82	6,87	3	20
Středně velké školy	24	25	14,87	18,95	1	100
Velké školy	29	77	19,86	23,54	1	150
Školy celkem	62	104	17,69	21,79	1	150

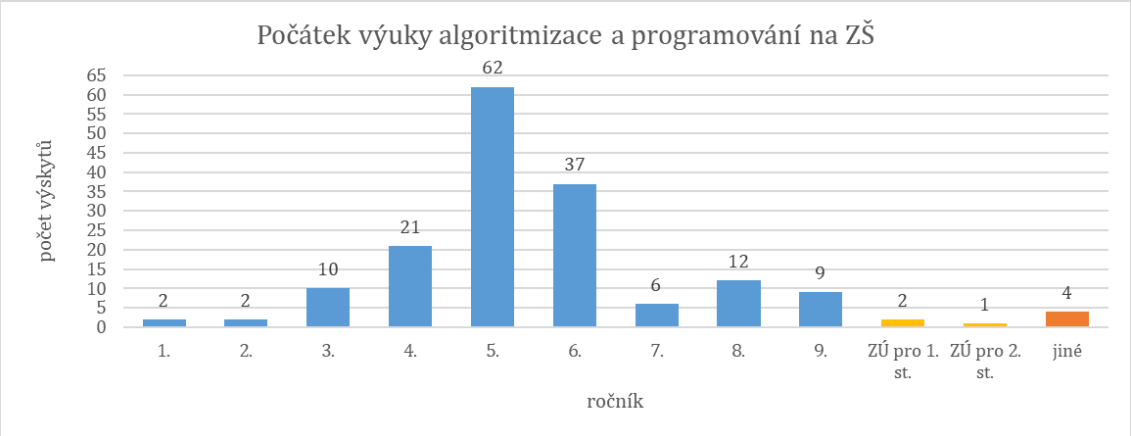
Tabulka 4: Počty hodin věnovaných výuce algoritmizace a programování

Celkem 104 respondentů uvedlo, že počet hodin, ve kterých se věnují problematice algoritmizace a programování, přesahuje pět vyučovacích hodin. Čtyři respondenti odhadli počet vyučovacích hodin na 100, jeden respondent uvedl dokonce 150 vyučovacích hodin, a to v kombinaci povinného předmětu a zájmového útvaru. Pokud je na základních školách realizována výuka algoritmizace a programování, většinově se tímto tematickým celkem zabývají více než 5 hodin.

Algoritmizace a programování je na základních školách vyučována nejvíce v 6. ročníku (106 výskytů), dále pak v 5. a 9. ročníku (92 a 90 výskytů). Učitelé ale nejčastěji zařazují výuku již do 5. ročníku 1. stupně (Tabulka 5), (Graf 6).

Počáteční období realizace výuky algoritmizace a programování									
Ročník	1.	2.	3.	4.	5.	6.	7.	8.	9.
Výskyt výuky v ročníku	2	4	13	31	92	106	82	79	90
Počátek výuky	2	2	10	21	62	37	6	12	9

Tabulka 5: Realizace výuky v ročnících



Graf 6: Počátek výuky algoritmizace a programování na ZŠ

Již bylo zmíněno, že realizace výuky algoritmizace a programování nemusí nutně probíhat jen v hodinách ICT. Školy mohou zahrnout problematiku programování nad rámec běžné výuky – do výuky v zájmových útvarech.

Zájmové útvary, ve kterých se edukátoři věnují algoritmizaci a programování, jsou zřizovány především na velkých školách (Tabulka 6).

Počet škol realizující výuku algoritmizace a programování v rámci zájmového útvaru				
Velikost školy	Rámec povinného předmětu společně se zájmovým útvarem		Rámec zájmového útvaru[5]	
	Absolutní četnost	Relativní četnost	Absolutní četnost	Relativní četnost
Malé školy	2	18,18 %	0	0,00 %
Středně velké školy	8	16,33 %	0	0,00 %
Velké školy	27	25, 00%	4	3,70%
Školy celkem	37	22,02 %	4	2,38%

Tabulka 6: Realizace výuky programování v rámci zájmových útvarů

Ze 108 velkých škol, které se zaobírají problematikou, se 27 z nich věnuje algoritmizaci a programování právě v rámci zájmového útvaru. Ve čtyřech z nich pak jen v rámci zájmového útvaru. V malých a středně velkých školách se problematice věnují spíše v hodinách ICT nebo kombinovaně – v hodinách ICT i zájmových útvarech. Na žádné z malých nebo středně velkých škol však nenastala situace, že by se algoritmizací a programováním žáci zabývali jen v zájmovém útvaru.

Preferované způsoby výuky a využívané zdroje

Trendem ve výuce prvostupňových dětí jsou (vedle programování pomocí bloků, ze kterých děti skládají funkční programy, například ScratchJr., Scottie Go!) programovatelné robotické hračky pro rozvoj logického a tvořivého myšlení, modelování a řešení problémů, výuka základů programování a také matematiky, a to i přes finanční zátěž pro školy. Šetření však ukazuje, že nejvyužívanějšími prostředky pro výuku jsou metody rozvíjející schopnosti informatického myšlení a programování bez počítače – unplugged, respektive řešení soutěžních úloh z oblasti informatického myšlení a algoritmů, úloh zaměřujících se na řešení problému (Tabulka 7).

Preference prostředí na 1. stupni ZŠ – povinné informatické předměty						
Způsob	1. r.	2. r.	3. r.	4. r.	5. r.	Celkem na prvním stupni
rozvíjení algoritmického myšlení bez programování, bez počítače – unplugged aktivity (Bobřík informatiky aj.)	0	1	8	22	59	**90**
block-based programování ve vizuálních programovacích jazycích (Scratch, Snap!, Blockly)	1	1	3	7	38	50
kurzy na webu code.org (Hour of Code)	0	0	2	6	28	36

[5] Škola realizuje výuku algoritmizace a programování jen v rámci zájmového útvaru.

programování ve 3D vizuálních programovacích jazycích (Kodu Game Lab, Alice aj.)	0	0	1	5	12	18
programování robotů (BEE-BOT, Ozobot, LEGO WeDo, Mindstorms aj.)	1	1	2	2	11	17
popis a tvorba algoritmu (slovně, vývojové diagramy aj.)	0	1	2	3	10	16
textové "turtle" programování s vizuální zpětnou vazbou příkazů (LOGO, Imagine LOGO aj.)	0	0	0	2	7	9
programování ve "hrách" (Lightbot, Hakitzu aj.)	0	0	0	1	5	6
jiný[6]	0	1	1	1	2	5
textové programování v programovacích jazycích (Python, Pascal/Delphi, C/C++, C#, Java aj.)	0	0	0	0	1	1
programování jednočipových počítačů (Arduino, Raspberry Pi aj.)	0	0	0	0	0	0

Preference prostředí při zájmových útvarech pro prvostupňové žáky	Celkem
block-based programování ve vizuálních programovacích jazycích (Scratch, Snap!, Blockly aj.)	**11**
rozvíjení algoritmického myšlení bez programování, bez počítače – unplugged aktivity (Bobřík informatiky aj.)	**11**
programování robotů (BEE-BOT, Ozobot, LEGO WeDo, Mindstorms aj.)	8
popis a tvorba algoritmu (slovně, vývojové diagramy aj.)	7
programování ve 3D vizuálních programovacích jazycích (Kodu Game Lab, Alice aj.)	4
kurzy na webu code.org (Hour of Code)	5
programování ve "hrách" (Lightbot, Hakitzu aj.)	2
textové "turtle" programování s vizuální zpětnou vazbou příkazů (LOGO, Imagine LOGO aj.)	1
textové programování v programovacích jazycích (Python, Pascal/Delphi, C/C++, C#, Java aj.)	1
programování jednočipových počítačů (Arduino, Raspberry Pi aj.)	0

Tabulka 7: Preference prostředí – prostředků pro výuku na 1. stupni

Výuka z pohledu programování se na druhém stupni velmi podobá výuce na stupni prvním. Opět jsou nejvíce využívány metody pro rozvoj algoritmického myšlení bez programování a unplugged metody (Tabulka 8).

Preference prostředí na 2. stupni ZŠ – povinné informatické předměty					
Způsob	6. r.	7. r.	8. r.	9. r.	Celkem na 2. stupni
rozvíjení algoritmického myšlení bez programování, bez počítače – unplugged aktivity (Bobřík informatiky aj.)	60	52	45	54	**211**
block-based programování ve vizuálních programovacích jazycích (Scratch, Snap!, Blockly aj.)	44	37	26	28	135
kurzy na webu code.org (Hour of Code)	34	28	23	26	111

[6] Respondenti uvádí konkrétně: *„PSPad – základy tvorby webových stránek; Easy Logo, ve třídě s pěti počítači."*

popis a tvorba algoritmu (slovně, vývojové diagramy aj.)	17	9	14	23	63
programování ve 3D vizuálních programovacích jazycích (Kodu Game Lab, Alice aj.)	16	12	13	13	54
programování robotů (BEE-BOT, Ozobot, LEGO WeDo, Mindstorms aj.)	13	10	11	11	45
textové "turtle" programování s vizuální zpětnou vazbou příkazů (LOGO, Imagine LOGO aj.)	9	8	9	15	41
textové programování v programovacích jazycích (Python, Pascal/Delphi, C/C++, C#, Java aj.)	4	3	6	13	26
programování ve "hrách" (Lightbot, Hakitzu aj.)	6	5	4	7	22
jiný[7]	3	3	4	6	16
programování jednočipových počítačů (Arduino, Raspberry Pi aj.)	0	2	4	4	10

Preference prostředí při zájmových útvary pro druhostupňové žáky	Do 12 let	Do 15 let
block-based programování ve vizuálních programovacích jazycích (Scratch, Snap!, Blockly aj.)	**9**	**18**
rozvíjení algoritmického myšlení bez programování, bez počítače – unplugged aktivity (Bobřík informatiky aj.)	7	10
programování robotů (BEE-BOT, Ozobot, LEGO WeDo, Mindstorms aj.)	8	10
popis a tvorba algoritmu (slovně, vývojové diagramy aj.)	3	6
programování ve 3D vizuálních programovacích jazycích (Kodu Game Lab, Alice aj.)	5	7
kurzy na webu code.org (Hour of Code)	4	2
textové "turtle" programování s vizuální zpětnou vazbou příkazů (LOGO, Imagine LOGO aj.)	1	4
programování ve "hrách" (Lightbot, Hakitzu aj.)	1	2
textové programování v programovacích jazycích (Python, Pascal/Delphi, C/C++, C#, Java aj.)	0	3
programování jednočipových počítačů (Arduino, Raspberry Pi aj.)	0	3
jiný[8]	0	1

Tabulka 8: Preference prostředí – prostředků pro výuku na 2. stupni

Celkově lze usuzovat, že nejfrekventovanější formou výuky algoritmizace a programování na základních školách jsou unplugged metody a rozvíjení informatického myšlení bez programování. Naopak programování jednodeskových mikropočítačů je na základních školách spíše raritou (Tabulka 9).

[7] Respondenti uvedli konkrétně: *„Baltík, PSPad editor – tvorba HTML, umimeprogramovat.cz, pouze na webových stránkách pomocí her."*

[8] Respondent uvedl konkrétně: *„MySQL – představa o databázích"*

Využívání forem výuky programování na základních školách				
Způsob	1.–5. ročník	6.–9. ročník	Zájmový útvar	Celkem
rozvíjení algoritmického myšlení bez programování, bez počítače – unplugged aktivity (Bobřík informatiky aj.)	90	211	28	**329**
block-based programování ve vizuálních programovacích jazycích (Scratch, Snap!, Blockly aj.)	50	135	38	223
kurzy na webu code.org (Hour of Code)	36	111	11	158
popis a tvorba algoritmu (slovně, vývojové diagramy aj.)	16	63	16	95
programování robotů (BEE-BOT, Ozobot, LEGO WeDo, Mindstorms aj.)	17	45	26	88
programování ve 3D vizuálních programovacích jazycích (Kodu Game Lab, Alice aj.)	18	54	16	88
textové "turtle" programování s vizuální zpětnou vazbou příkazů (LOGO, Imagine LOGO aj.)	9	41	6	56
programování ve "hrách" (Lightbot, Hakitzu aj.)	6	22	5	33
textové programování v programovacích jazycích (Python, Pascal/Delphi, C/C++, C#, Java aj.)	1	26	4	31
programování jednočipových počítačů (Arduino, Raspberry Pi aj.)	0	10	3	13

Tabulka 9: Celkový přehled preference prostředí na ZŠ

Vedle označení obecných skupin či přístupů k výuce měli respondenti možnost označit konkrétní prostředí, které využívají, a představit jeho výhody. V tabulce níže (Tabulka 10) jsou zaznamenány odpovědi (řazené sestupně) podle relativní četnosti výskytu. Přestože respondenti v předešlých odpovědích preferovali především unplugged aktivity, výskytu dobrovolných odpovědí vévodí block-based programování ve Scratchi a web neziskové organizace Code.org společně s herními aktivitami z celosvětové akce Hour of Code, (celkem 87 výskytů) následované 3D herním programovacím prostředím Kodu Game Lab od Microsoft Research (28 výskytů).

Preferované prostředí	Četnost	Výhody
Scratch **(ScratchJr)**	**63** z toho **(2)**	- líbívost, online, zdarma, příjemné prostředí; - jednoduché, pro všechny věkové kategorie; - nemusí se soustředit na syntaxi… věnují se plně pochopení principu; - nenáročný, intuitivní prostředí, grafika blízká dětem, dá se nahlédnout do již hotových projektů a inspirovat se; - učení se hrou; - logická posloupnost MIT App Inventor – okamžitá odezva na smartphony
Code.org **(Hour of Code)**	**47** z toho **(24)**	- líbivost; - podrobně vypracované postupně rozvíjející kurzy; - zajímavé pro děti, baví je; - zajímavé, jednoduché, zábavné; - připravené materiály; - učení se hrou; - okamžitý feedback, názornost
Kodu Game Lab	**28**	- zábavné, rozvíjí tvořivost; - poměrně jednoduchý vstup dětí; - virtuální svět, graficky zajímavé pro žáky
Baltík (Baltie)	**25**	- možnost výjezdních soutěží; - vhodný pro 3. třídu
Imagine (Easy) Logo	**11**	- jednoduchost, učebnice
PSPad (HTML + CSS + JavaScript)	**11**	- zdarma, rychlý, jednoduchý
Lego Mindstorms	**10**	- nezmíněny
Bobřík informatiky	**8**	- nezmíněny
Ozoblockly (Ozobot)	**7**	- možná odezva přes roboty
Karel	**7**	- jednoduché, česky; - snadno pochopitelné
Visual Studio (C#)	**6**	- základy tvorby aplikací; přehledné prostředí (verze 2010), VB jazyk je poměrně jednoduchý, a proto pro výuku základů programování vhodný
Lego WeDo	**6**	- nezmíněny
Blockly	**5**	- jednoduché, pro všechny věkové kategorie
Alice	**3**	- kreativní, zábavné; - jednoduché prostředí, efektivní, líbivý přístup oproti Scratch, freeware, lze použít jako propedeutikum k Javě
Pascal	**3**	- nezmíněny
Vývojové diagramy (papír)	**3**	- nezmíněny
Umíme programovat	**3**	- nenáročnost na vybavení, jednoduchost
Java	**3**	zdarma
Minecraft EDU; BeeBot; Micro:Bit – Java Block editor; MS Excel; Arduino; PHP	**po 2**	- nezmíněny
Quirkboti – brčkoboti; Králíček Péťa; Swift Playgrounds; Tynker; Mblock; Code Combat; ProBot; NC Lab;	**po 1**	- nezmíněny

Tabulka 10: Preferovaná prostředí

Respondenti měli možnost představit zdroje informací a online prostředí, jež využívají při přípravě na vyučování a při samotné výuce. Celkem 13 respondentů uvedlo, že při výuce algoritmizace a programování využívá učebnice, přičemž nejvíce jsou využívány učebnice Jiřího Vaníčka (Příloha 4). Naopak 110 odpovědí směřuje k hypermediím – webům zaměřeným na programování nebo umožňujícím samotné učební aktivity. Podle odpovědí lze obecně soudit, že učitelé využívají rozmanité webové stránky zaměřující se na problematiku spojenou s programováním a algoritmizací. Ovšem téměř nikdo nevyužívá možnost propojení a spolupráce s ostatními učiteli v rámci sociálních sítí s možností sdílet své poznatky a postupy s ostatními. Například pouze 1 respondent zmínil využívání materiálů ze sociálních sítí (GEG a facebookových skupin). Kompletní seznam webů sloužících k výuce je k dispozici v příloze (Příloha 5).

3.3.3 Současný stav výuky algoritmizace a programování vyplývající z dotazníkového šetření

Významnost tematického celku algoritmizace a programování na základních školách roste. K výuce celku, jako k nepovinnému obsahu, se postupem času přiklání stále větší počet škol, což, pokud vezmeme v úvahu chystané změny v kurikulárních dokumentech státní úrovně, je vhodný postup v přípravě na změny. Postupný nárůst je patrný z počtu zapojených škol do informatické soutěže Bobřík informatiky, ale i z porovnání Krejsova dotazníkového šetření z roku 2014 s tímto šetřením. Zatímco v roce 2014 zařazovalo algoritmizaci a programování do výuky pouhých 14 % oslovených učitelů, v roce 2018 je to již více než 34 %. Vliv na rozdíl samozřejmě může mít skutečnost, že vzorek respondentů není reprezentativní (u dotazníkových šetření, kde se respondenti zapojují dobrovolně, je reprezentativita problematická).

Více než polovina učitelů, kteří neimplementují téma do výuky, se nejčastěji odvolávají na absenci obsahu ve státních kurikulárních dokumentech. Téměř třetina učitelů nerealizujících výuku programování však s tímto stavem nesouhlasí.

Samotná realizace výuky tématu bývá nejčastěji zařazována poprvé do pátého a šestého ročníku základních škol. Celková časová dotace věnovaná výuce algoritmizace a programování roste s velikostí školy. Velké školy, které se problematice algoritmizace a programování věnují, vyhrazují v učebních plánech pro tuto oblast cca 20 vyučovacích hodin, středně velké cca 15 hodin a malé školy méně než 10 vyučovacích hodin. Jestliže budeme brát v úvahu počet vyučovacích hodin ICT

plynoucí z RVP ZV, jsou výsledky šetření pozitivní, neboť celkový počet hodin ICT, který nařizuje MŠMT ČR v RVP ZV se pohybuje na hranici maximálně 80 všech vyučovacích hodin za školní rok.

Výuka probíhá především formou výuky unplugged a účastí škol v informatických soutěžích či mezinárodních akcí. Stále oblíbenějším se stává block-based programování. Respondenti na prvních dvou místech preferovaných prostředí uvedli právě ta, která stavějí na block-based programování. I zde je patrný rozdíl oproti roku 2014, kdy učitelé upřednostňovali především prostředí Baltík a Imagine Logo.

Faktory, jež učitelé spatřují v preferovaném prostředí jako výhodné, jsou především:

- líbivost, zábavnost a atraktivita prostředí jako motivační prvek;
- přístupnost odkudkoli, kdykoli, z libovolné platformy;
- žádné nebo minimální finanční náklady;
- jednoduchost a absence syntaxe jako možnost soustředit se jen na řešení problému, pochopení algoritmů a programovacích konstrukcí;
- názornost a okamžitá zpětná vazba, možnost nahlédnutí do programu jiných projektů;
- možnost pracovat v prostředí nezávisle na věku, resp. ve značném věkovém rozptylu.

Uvedené výhody jsou logické a bude na ně v této práci brán značný zřetel, a to především při hodnocení dostupných online zdrojů vhodných pro podporu výuky algoritmizace a programování a také při návrhu vlastního modelu výuky.

4. Zmapování dostupných online zdrojů určených pro podporu výuky algoritmizace a programování dětí.

Aktuální situace v počtu zdrojů pro podporu výuky v online prostředí – na webových stránkách či v obchodech s aplikacemi pro mobilní platformy App Store nebo Google Play <https://play.google.com/> je problematice algoritmizace a programování nakloněna. Například obchod Google Play [cit. 2018-10-07] vyhledá a nabídne pro pojem „programming“ 245 aplikací zabývajících se problematikou – od aplikací vhodných pro malé děti (3+) po aplikace cílící na teenagery a dospělé. Počet vyhledaných aplikací byl navíc filtrován pro možnost stažení zdarma a pro hodnocení čtyřmi a více hvězdami. Platforma sdílených videí Youtube <https://www.youtube.com/> po zadání stejného pojmu zobrazí tisíce videí s tematikou programování.

4.1 Pojem zdroj

Zdroj je v této práci chápán jako dostupný prostředek a nástroj, který je vhodný pro podporu výuky algoritmizace a programování.

Práce si bere za jeden z hlavních cílů zmapovat online zdroje, které jsou zdarma k dispozici a následně tyto zdroje zhodnotit podle navržených evaluačních kritérií. Cíleno je především na online aplikace a webová prostředí, která představují vhodné prostředky výuky a poskytují nástroje pro výuku základů programování, algoritmizace, stejně jako umožňují rozvíjet informatické myšlení u žáků základní školy.

Do výběru nejsou zahrnuty zdroje cílící na programování jako na prostředek řízení hardwaru, u nichž je zpětná vazba spojena s externí robotickou hračkou nebo stavebnicí. Důvodem je pořizovací cena robotů a stavebnic, která se pohybuje v řádech tisíců korun českých.

4.2 Mapování online zdrojů

Ne všechny online zdroje (blogy, videa, webové stránky a aplikace poskytující informace o problematice či nabízející výuková prostředí a nástroje, na které lze v prostředí Internetu narazit) jsou vhodným prostředkem pro výuku algoritmizace

a základů programování pro děti. Druhá generace webu (Web 2.0[9]) již delší dobu umožňuje všem uživatelům s přístupem k internetu tvořit a publikovat na webu vlastní obsah různé kvality a relevance. Tento stav s sebou přináší značné zvýšení počtu zdrojů rozmanité kvality a zaměření, ale také horší orientaci v jejich velkém počtu. Z aktuálního stavu dále plyne obtížnější rozhodování, jaký prostředek pro výuku či učení vlastně zvolit.

K získání přehledu vhodných a využívaných výukových online zdrojů je v práci využito:

- preferencí zdrojů vyplývajících z dotazníkového šetření;
- analýzy elektronických článků dostupných na portálech se zaměřením na edukaci.

Elektronické články jsou vybírány především z předních portálů, jež se zabývají implementací technologií do vzdělávání, konkrétně EdSurge a Edutopia. Dále jsou vybírány články z portálů, na které je z EdSurge a Edutopie odkazováno. Z českého prostředí je pak vybrán přední metodický portál RVP.CZ.

4.2.1 Výběr dostupných online zdrojů z pohledu učitelů informatických předmětů

Učitelé v dotazníkovém šetření uvedli celkem 58 různých zdrojů (Příloha 5), které ve výuce preferují. Pouze 7 zdrojů bylo zmíněno více než dvěma učiteli, a to následující:

- **Code.org**, dostupný na <https://code.org>;
- **Hour of Code**, dostupný na <https://hourofcode.com/cz>;
- **Scratch**, dostupný na <https://scratch.mit.edu>;
- **Bobřík informatiky**, dostupný na <https://www.ibobr.cz>;
- **Ozoblockly**, dostupný na <https://ozoblockly.com/>;
- **Karel**, dostupný na <http://karel.oldium.net>;

[9] Termín Web 2.0 vytvořil a poprvé použil Tim O'Reilly v roce 2004 jako označení pro další vývojové stádium webu. [cit. 2018-10-07]

- **Umíme programovat**, dostupný na <https://www.umimeprogramovat.cz>.

Zbylých 51 zdrojů bylo zmíněno jen individuálně, jedním nebo dvěma učiteli.

Obecně se dá říci a z reakcí respondentů vyplývá, že učitelé preferují především prostředky block-based vizuálního programování a unplugged metody výuky, respektive využívají archivních soutěžních otázek z Bobříka informatiky, které u žáků rozvíjejí informatické myšlení.

4.2.2 Výběr dostupných online zdrojů analýzou elektronických článků a dokumentů zaměřených na edukaci

EdSurge je přední americký informační portál, jenž se v rámci svého působení snaží seznamovat návštěvníky s novými možnostmi aplikování technologií do vzdělávacího procesu, sdílí podrobné informace o nových technologiích a o jejich implementaci do výuky, do učení žáků (kategorie K-12) i do vyššího vzdělávání. V sekci *„EdSurge Guide“*, článku *„Výuka kódování pro děti“* [cit. 2018-10-10] je porovnáno více než čtyřicet nástrojů, které EdSurge doporučuje jako vhodné pro začátečníky v oblasti programování. Nástroje jsou rozděleny do šesti kategorií, z nichž jsou do této práce vybrány kategorie *„Výuka logiky v programování“*, *„Vizuální programování s bloky“* a *„Hry a nástroje pro naprogramování vlastní hry“*.

Z nabízených možností jsou vybrány pouze ty, které jsou dostupné zdarma.

Vybrané nástroje k výuce logiky a programování:

- **Alice**, dostupný na <http://www.alice.org/>;
- **BotLogic**, dostupný na <http://botlogic.us/>;
- **CargoBot**, dostupný na <https://twolivesleft.com/CargoBot/>;
- **Code.org Studio**, dostupný na <https://studio.code.org>;
- **CS Unplugged**, dostupný na <https://csunplugged.org>;
- **Turtle Academy**, dostupný na <http://turtleacademy.com/>.

Vybrané nástroje k výuce block-based programování:

- **App Inventor**, dostupný na <http://appinventor.mit.edu>;
- **Hopscotch**, dostupný na <https://www.gethopscotch.com/>;
- **Scratch 2.0**;
- **Snap!**, dostupný na <https://snap.berkeley.edu/>;
- **Tynker**, dostupný na <https://www.tynker.com>.

Vybrané nástroje pro výuku algoritmizace a programování formou hry nebo tvorbou vlastní hry:

- **Code Combat**, dostupný na <https://codecombat.com/>;
- **Hakitzu**, dostupný na <https://kuatostudios.com/>;
- **Kodable**, dostupný na <https://www.kodable.com/>;
- **Kodu**, dostupný na <https://www.kodugamelab.com/>;
- **Stencyl**, dostupný na <http://stencyl.com/>.

Vzdělávací nadace George Lucase – Edutopia se věnuje především transformaci vzdělávání pro věkové kategorie K-12. Představuje učitelům, rodičům i studentům vhodné postupy k osvojování kompetencí ve vzdělávání. Davis (2016) ve svém článku na Edutopii předkládá několik zdrojů a organizací zaměřujících se na výuku základů programování, jež jsou podle něj vhodné [cit. 2018-10-10]. Z nich jsou vybrány následující:

- **Google Made w/ Code**, dostupný na <https://www.madewithcode.com/>;
- **Scratch**;
- **Tynker**, dostupný na <https://www.tynker.com/hour-of-code/>;
- **Code.org**, dostupný na <https://code.org/learn>;
- **Khan Academy**, dostupný na <https://www.khanacademy.org/computing>;
- **Stencyl**, dostupný na <http://stencyl.com/>;
- **Code Monster**, dostupný na <http://www.crunchzilla.com/code-monster>;
- **MIT App Inventor**, dostupný na <http://appinventor.mit.edu/explore/>;
- **Hour of Code**;

- **Code.org**;
- **Black Girls Code**, dostupný na <http://www.blackgirlscode.org/>;
- **CoderDojo**, dostupný na <https://coderdojo.com/>;
- **CS Unplugged**.

ESchool News – Daily Tech News & Innovations je pravidelný měsíčník zaměřující se na pokrytí problematiky technologií ve vzdělávání, ve všech jejích aspektech, a to v celém vzdělávacím systému severoamerického školství nesoucí označení K-20. Evans (2015) [cit. 2018-10-10] v článku *„Jak vybrat správný programovací jazyk pro studenty“* předkládá 3 vhodná prostředí pro výuku základů programování:

- **Turtle Art**, dostupný na <http://turtleart.org>;
- **Scratch**;
- **Tynker**.

Common Sense Education je další neziskovou organizací, která se věnuje poskytování opor ve vzdělávání. V článku *„Nejlepší aplikace a webové stránky pro učení se programování a kódování“* [cit. 2018-10-10], sekci *„Nejlepší výběr“* je prezentován seznam 43 aplikací a webů vhodných pro výuku programování.

Z článku jsou vybrány jen ty zdroje, které jsou zdarma a pro věkovou kategorii K-12, tedy kategorii žáků stejného věku, jako je věk žáků druhého stupně základní školy:

- **Code.org**;
- **Scratch**;
- **Kodu Game Lab**;
- **Google CS First**, dostupný na <https://csfirst.withgoogle.com/en/home>;
- **Made w/ Code**;
- **App Inventor**, dostupný na <http://www.appinventor.org/>;
- **Thunkable**, dostupný na <https://thunkable.com/#/>;
- **Mozilla Thimble**, dostupný na <https://thimble.mozilla.org/cs/>;
- **Swift Playgrounds**, dostupný na <https://www.apple.com/uk/swift/playgrounds/>.

Edudemic je webový server pokrývající problematiku vzdělávání a zabývající se vhodnými postupy začleňování technologií do vzdělávání.

Dustin Le (2015), člen Edudemic, ve svém článku *„Tři nejlepší zdarma dostupné webové stránky zaměřené na programování pro děti“* [cit. 2018-10-10] představuje pět hlavních kritérií, podle kterých stránky vybíral, a to: jednoduchost prostředí, estetika a design, zábavnost, efektivita, cena.

Podle výše zmíněných kritérií vybral následující zdroje:

- **Code Avengers**, dostpné na <https://www.codeavengers.com>;
- **Lightbot**, dostupné na <http://lightbot.com/flash.html>;
- **Code Combat**.

Akademie kódování CodaKid (sídlící v USA) cílí na vzdělávání nové generace programátorů, a to formou volnočasové aktivity nebo prostřednictvím online kurzů. V jednom z publikovaných článků s názvem *„7 špičkových kódovacích jazyků roku 2018“* představuje Dodge (2018) jazyky, které pokládá za opravdu nejlepší. Svůj výběr podporuje tvrzením [cit. 2018-10-10]: *„Na CodaKid jsme učili programovat desítky tisíc studentů a vyzkoušeli jsme snad každý zdroj vhodný k výuce programování pro děti. V následujícím článku uvádíme seznam nejlepších:“*

- **Scratch**;
- **Blockly**, dostupné na <https://developers.google.com/blockly/>;
- **Swift Playgrounds**.

CodeCamp je přední australskou organizací zaměřující se na programování jako volnočasovou aktivitu dětí ve věku 5 až 13 let. Děti se v kempech trvajících 2 až 4 dny učí informatickému myšlení, řešení problémů a logice. Tomu všemu hravou formou – programováním her a aplikací. Člen tutorského týmu Young (2018) [cit. 2018-10-13] předkládá v článku *„20 nejlepších programovacích jazyků pro děti,“* prostředí a jazyky, které s dětmi v kempech programování nejvíce využívají. V článku také informuje rodiče o vhodném výběru prostředí, v němž by se děti mohly zdokonalovat v dovednostech programování. Upozorňuje (podobně jako Resnick) na fakt, že děti by měly začínat programovat v prostředí podporující vizuální programování, které je nebude demotivovat nefunkčností programu,

například kvůli zapomenutému středníku. Dále v článku zmiňuje vyšší programovací jazyky i skriptovací jazyky, které jsou děti od věku 13 let schopny zvládnout, a ve kterých v kempech programují. Tyto jazyky, konkrétně Basic, Perl, PHP, Java, JavaScript, Ruby a C++ však nebudou dále do procesu mapování a hodnocení online zdrojů zařazeny, neboť jsou svou podstatou vhodné spíše do prostředí výuky programování na středních školách.

Výběr prostředí, která Young pokládá za ideální:

- **Alice**,
- **Blockly**,
- **Scratch**,
- **CoderZ**, dostupný na <http://ftc.coderzworld.com/#/account/login/>;
- **Kodu**,
- **Swift Playgrounds**.

CS First je jednou z mnoha aktivit společnosti Google, která se zaměřuje na podporu informatiky ve vzdělávání. Podněcuje učitele k výuce programování a děti k jeho učení. CS First představuje celkem devět témat – vyprávění příběhů, umění, design hry, móda, hudba, přátelství, sociální média, sport a animace. Pro každé téma je připraven učební plán obsahující osm lekcí [cit. 2018-10-10]. Všechna témata jsou představena na <https://csfirst.withgoogle.com/c/cs-first/en/curriculum.html>. Jako prostředek tvorby je využit **Scratch**.

Metodický portál RVP.CZ je v oblasti vzdělávání v České republice přijímán a využíván jako prostředí pro inspiraci a pro sdílení zkušeností mezi učiteli. Učitelé a studenti pedagogických fakult představují v článcích věnujících se problematice algoritmizace a programování různé nové i osvědčené zdroje, které hodnotí a porovnávají s ostatními. V článcích představují následující online zdroje, které jsou pro přehlednost uvedeny do tabulky (Tabulka 11).

Zdroj	Autor článku	Rok
Scratch	Krejsa	2014
	Vancl	2017
	Krisl	2013
	Čechová Humpolcová	2012
	Němec	2018
	Holečková	2017
Alice	Vít	2015
Hour of Code	Šandová	2013
Pencil Code Gym	Šambazov	2015
Blockly Games		
Imagine LOGO	Podzimek	2013
Code.org Studio		2014
Comenius Logo	Kozák	2015
Kodu	Humpolcová	2011
Turtle Pond	Bečvář	2009
Robot Karel	Frýbert	2008
Lightbot	Kocourková	2015
Code Monster		
Code Combat		
Code Kingdoms		
Code Monkey		
	Khas	2014
CS Unplugged	Naske	2008

Tabulka 11: Vybrané zdroje autorů článků na metodickém portálu RVP.CZ

4.2.3 Vlastní výběr skupiny online zdrojů

Na základě mapování zdrojů byla sestavena tabulka se všemi vyskytujícími se prostředky, které jsou seřazeny podle četnosti výskytu (Příloha 6). Tabulka níže (Tabulka 12) představuje ty z nich, které byly zmíněny dva a vícekrát.

Prostředek	Četnost výskytu	Stručný popis	Podpora češtiny
Scratch	8	Scratch je zřejmě světově nejznámějším online prostředkem pro výuku základů programování s více než 32 miliony registrovaných uživatelů, jak uvádí Scratch Statistics [cit. 2018-10-15]. Využívá block-based vizuálního programování. Scratch byl vyvitut na Massachusetts Institut of Technology (dále jen MIT) skupinou Lifelong Kindergarden – MIT Media Lab v roce 2007.	ANO
Code.org, Studio Code.org	6	Web neziskové organizace Code.org patří mezi špičku v oblasti zdrojů vhodných pro výuku algoritmizace a základů programování pro děti v celosvětovém měřítku. Edukátorům a dětem nabízí rozmanité možnosti využití prostředků webu, od jednohodinové akce (Hour of Code) po ucelené celoroční kurzy. Na webu je dostupná kurikulární dokumentace pro kurzy i lekce, hodinové plány i správné odpovědi.	ANO
Kodu	4	Kodu (Kodu Game Lab) od společnosti Microsoft je vizuálně programovatelný, zdarma dostupný engine pro tvorbu her ve 3D prostředí. Samotné programování je realizováno skládáním grafických ikonek. Programovací logika je založena na podmínce if–then, v Kodu reprezentovanou jako when–do. Dále umožňuje vytvořit nekonečnou smyčku (ve hře) always–do. Kodu navíc nabízí možnost překladu vytvořeného kódu do jazyka C#.	ANO
Blockly (Blockly Games)	3	Společnost Google na webu Google for Education [cit. 2018-10-15] představuje Blockly jako open source knihovnu, která umožňuje přidávat block-based programovací nástroj do webových a mobilních aplikací, a tak využít všech kladů vizuálního programování. Dokáže reprezentovat koncepty jako proměnná, příkaz, cyklus a další. Blockly je do značné míry využíváno jako programovací nástroj v online vzdělávacích projektech, například na webu organizace Code.org i Hour of Code, v App Inventor, Made with Code nebo při programování robůtka Ozobota na <https://ozoblockly.com/>.	ANO
Code Combat	3	Code Combat je webovou platformou, která umožňuje uživateli vybrat si k výuce herní prostředí. K výběru jsou jazyky Python nebo Javascript. Zdarma je poskytnuta pouze první herní mapa (Kobka Kithgardu) s 42 úrovněmi. Uživatel je v této mapě seznamován se základní syntaxí, cykly, proměnnou, funkcí a parametry funkce nebo s řetězci. To vše formou RPG[10] hry.	ANO

[10] RPG (angl. Role-Playing Game) je hrou na hrdiny. Hráč na začátku vybírá svého hrdinu, kterého v průběhu hry zdokonaluje, a to jak hrdinovu dovednostní složku, tak výbavu (výzbroj a výstroj).

CS Unplugged	3	CS Unplugged představuje možnost výuky informatiky bez počítače formou aktivit využívajících herní prvky. Ke všem aktivitám je k dispozici velmi podrobná metodika, včetně videí a materiálů k vytištění. Web <https://csunplugged.org> představuje v pěti hlavních tématech úvod do problematiky – binární soustavy, programování (robotů), algoritmů, sítí, detekce a opravy chyb, a to hravou formou vhodnou pro děti.	NE
Hour of Code	3	Akce organiace Code.org, jež se zaměřuje na propagaci dětského programování v rámci jedné vyučovací hodiny. Web akce poskytuje zdarma značné množství herních aktivit v různých podobách a využívající různých prostředků, například Blockly, Tynker, JavaScript, Python, Scratch, Java, atd.	ANO
Swift Playgrounds	3	Swift Playgrounds je aplikace společnosti Apple pro iPad. Výuka probíhá v programovacím jazyku Swift ve třech kurzech nazvaných Learn to Code1, 2 a 3. Kurzy seznamují děti se základními koncepty programování: s příkazy a sekvencemi příkazů, debuggingem, podmínkami a logickými operátory, řídícími strukturami, proměnnými, funkcemi, parametry, poli a dalšími.	NE
Tynker	3	Tynker je Scratchi podobné prostředí společnosti Neuron Fuel, jež podporuje výuku dovedností v block-based programování. Nabízí velké množství kurzů různé obtížnosti pro žáky různých věkových kategorií a dovedností. Součástí je okamžitý překlad programu z bloků do JavaScriptu. Nevýhodou je nutnost platby za plnou funkčnost všech nabízených nástrojů.	NE
App Inventor	3	App Inventor je online block-based (Blockly) programovací prostředí z dílny MIT zaměřené na tvorbu mobilních aplikací s operačním systémem Android. Web <http://appinventor.mit.edu> nabízí tutoriály pro začátečníky i mírně pokročilé, metodickou dokumentaci pro učitele. Vytvořenou aplikaci je možné sledovat přímo na mobilním zařízení nebo v Android emulátoru.	NE
Alice	2	Software umožňuje výuku programování ve 3D prostředí. Projekt Alice je reakcí učitelů vytvořit prostředí, které by sloužilo pro výuku objektově orientovaného jazyka Java, což především poslední verze programu Alice 3 splňuje, neboť je s Javou plně kompatibilní. Neumožňuje však ani dědičnost, ani polymofismus.	NE
Code Monster	2	Code Moster je zdarma dostupný online nástroj pro výuku a procvičování syntaxe jazyka JavaScript. Žák má k dispozici celkem 59 lekcí, ve kterých plní požadavky monstra na změnu kódu.	NE
Google CS First	2	CS First ve všech projektech využívá k výuce prostředí Scratch. Jednotlivá témata jsou k dispozici v podobě Scratch studií, které obsahují různé projekty od CS First.	NE

Google Made with Code	2	Made w/ Code (Made with Code) je genderově zaměřená iniciativa společnosti Google, jež se soustředí na přiblížení a zatraktivnění programovaní dívkám. Web nabízí výuku programování v 19 projektech, ve kterých je využíváno vizuální programování pomocí bloků, konkrétně Blockly.	NE
Lightbot	2	Lightbot kombinuje hru s prvky programování. Byl vytvořen pro neziskovou organizaci Code.org. Lightbot je „programován" pomocí ikonek, zaměřuje se na principy, které by mohly být v začátcích pro děti obtížně pochopitelné a které jsou běžné v různých vyšších programovacích jazycích.	ANO
Robot Karel	2	Česká online verze prostředí a programovacího jazyka určeného pro výuku dětí – Karel the Robot. Karel umožňuje vytvářet nové příkazy, které se kombinují ze čtyř původních nebo již vytvořených.	ANO
Stencyl	2	Software, který se zaměřuje na tvorbu her pomocí block-based programovacího jazyka. Pro práci v programu je však nutná insalace. Základní verze je k dispozici zdarma.	NE

Tabulka 12: Nejčastěji zmiňované online zdroje

Nezkrácená verze tabulky je součástí přílohy (Příloha 7).

Do další části práce budou vybrány a uváděny pouze ty zdroje, které plně podporují češtinu. Důvody eliminace ostatních jsou především dva související s problematikou:

- porozumění – eventuálně špatná interpretace, nepochopení atp.
- plné koncentrace na problém a řešení problému, nikoliv na překlad.

4.2.4 Porovnání vybraných online zdrojů z pohledu návštěvnosti webových stránek

Posledním krokem mapování online zdrojů je jejich vzájemné porovnání z pohledu:

- celkového počtu návštěv webu za poslední tři měsíce (celkem);
- počtu měsíčních návštěv webu (měsíčně);
- celkového počtu jednotlivých návštěvníků (celkem IP);
- průměrného počtu návštěv webu jedním návštěvníkem (IP);
- průměrného času stráveného na webu (čas);

- míry okamžitého opuštění webu (bounce rate[11]);

pomocí nástroje SimilarWeb zaměřeného na analýzu webových stránek, dostupného na <https://www.similarweb.com/>, a to v období 3 měsíců (od července do září 2018). Výsledky jsou zpracovány v tabulce níže (Tabulka 13).

Doména	Celkem	Měsíčně	Celkem IP	IP	Čas [mm:ss]	Bounce rate [%]
scratch.mit.edu	43 800 000	14 600 000	6 300 000	2,32	08:32	39,86
code.org	12 800 000	4 255 000	1 700 000	2,46	12:32	33,50
codecombat.com	2 500 000	817 482	293 829	2,78	08:45	52,29
hourofcode.com	1 100 000	380 956	168 272	2,26	02:40	51,58
blockly-games.appspot.com	537 485	179 162	66 736	2,68	10:47	31,89
lightbot.com	468 600	156 208	69 350	2,25	01:49	45,07

Tabulka 13: Výsledky analýzy webových stránek nástrojem Similarweb

Ze vzájemného porovnávání byly vyřazeny Kodu a Robot Karel především z důvodu objektivity. Zatímco Kodu <www.kodugamelab.com> poskytuje na webu, vedle metodické podpory a tipů, především možnost stažení instalačního souboru, pak online verze Karla <http://karel.oldium.net> je tak specificky česká záležitost, že se u ní nepředpokládá přístup z jiných států. Pro představu jsou výsledky těchto dvou webů představeny v tabulce níže (Tabulka 14).

Doména	Celkem	Měsíčně	Celkem IP	IP	Čas [mm:ss]	Bounce rate [%]
kodugamelab.com	183 004	61 001	26 570	2,30	02:10	53,09
karel.oldium.net	4 700	1567	< 5 000[12]	3,55	01:40	55,77

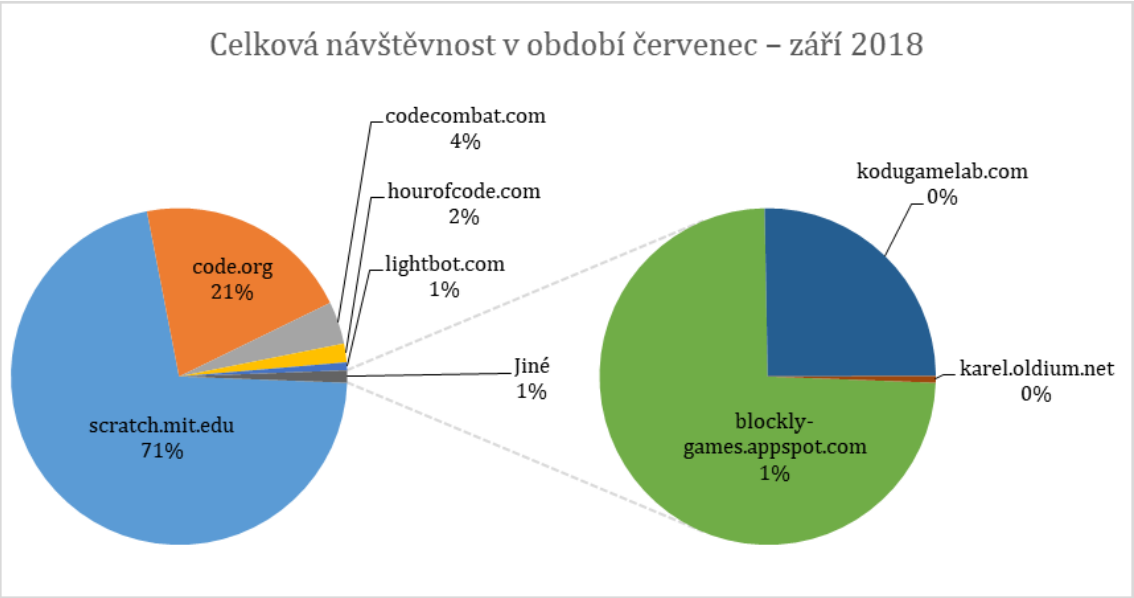
Tabulka 14: Výsledky analýzy Kodu a Karel nástrojem Similarweb

Výsledky analýzy korelují s dříve zjištěným stavem. Extrémních hodnot dosahuje Scratch. V porovnání všech zmíněných domén za poslední tři měsíce patří doméně scratch.mit.edu více než dvě třetiny všech přístupů na domény zařazené do analýzy (Graf 7), což svědčí o jeho výsostném postavení mezi ostatními prostředky. Dalším

[11] Slovník MediaGuru [cit. 2018-10-11] představuje pojem Bounce rate jako „procento návštěvníků webu, kteří web ihned po vstupu opustili (zjistili, že konkrétní web není ten, který hledali apod.). Například v Google Analytics bounce rate znamená, že příchozí návštěvník navštívil na serveru jen jednu stránku a server opustil."

[12] Nelze přesně určit, neboť SimilarWeb údaj neposkytuje.

zajímavým údajem je výsledek celosvětového žebříčku (Global Rank) představující návštěvnost webových stránek, ve které patří doméně code.org celkové 6 668 místo (v USA pak 3 112 místo). Tento výsledek jistě svědčí o popularitě organizace a programování ve světě.



Graf 7: Celková návštěvnost vybraných domén

Z ryze českých zdrojů nelze opomenout doménu umimeprogramovat.cz (zmíněnou třemi učiteli v dotazníkovém šetření), neboť podle analýzy návštěvnost domény činí ve třech měsících téměř 30 tisíc přístupů (9 931 za poslední měsíc), čímž výrazně převyšuje nejen doménu karel.oldium.cz, ale i další české zdroje. *„Umíme programovat“* je jednou z šesti částí projektu *„Umíme to“* zajišťující především repetičně-fixační funkci.

4.2.5 Souhrn výsledků zmapovaných online zdrojů určených pro podporu výuky algoritmizace a programování

Tato část práce se zabývala zmapováním současného stavu dostupných online zdrojů, které jsou vhodným prostředkem k podpoře výuky algoritmizace a programování. Zdroje byly vybrány na bázi výsledků plynoucích z výběru zdrojů zmíněných v dotazníkovém šetření a dále byly vybírány na základě analýzy článků předních světových portálů a metodického portálu RVP.CZ, jež se obecně zabývají problematikou vzdělávání. Nejčastěji se vyskytující zdroje byly seřazeny podle

četnosti, poté došlo k eliminaci těch, které nepodporují češtinu z důvodů zmíněných v kapitole 4.2.3. Na závěr byly domény vybraných zdrojů podrobeny analýze návštěvnosti (Tabulka 13), a to v celosvětovém měřítku, který SimilarWeb poskytuje. Výsledky analýzy všech vybraných zdrojů z pohledu návštěvnosti lze nalézt v příloze (Příloha 8).

Podle analýzy návštěvnosti se nejčastěji využívají zdroje: Scratch, Code.org, dále zdroje pro tvorbu mobilních aplikací – MIT App Inventor a Thunkable následovanými Tynkerem. Všechny zmíněné zdroje jsou měsíčně navštíveny více než jedním milionem uživatelů. Z pohledu průměrného času stráveného jedním uživatelem v průběhu jednoho přístupu na webu je situace odlišná. Nejdelší dobu tráví uživatelé na webu Bobříka informatiky <ibobr.cz>, a to bez 1 sekundy 13 minut. Další zdroje, které si udržují čas nad 10 minut, jsou Code.org (čas 12:32), Blockly Games (čas 10:47) a Thunkable (čas 10:42). SimilarWeb však analyzuje jen údaje měřitelné v prostředí webu a nelze jím měřit počet přístupů k aplikacím na mobilních zařízeních ani čas strávený v těchto aplikacích.

Autoři článků představující zdroje pro výuku programování stanovují v kombinaci výčet 39 zdrojů (Příloha 6), které chápou jako vhodné pro výuku základů programování. Podle četnosti výskytu se nejčastěji objevují Scratch, Code.org, Kodu, Blockly a Blockly Games, Code Combat, CS Unplugged, Hour of Code, Swift Playground a Tynker. Všechny tyty zdroje byly autory zmíněny tři a vícekrát.

Průnikem obou částí, tedy analýzou návštěvnosti webů (celková návštěvnost nad 1 milion uživatelů za poslední tři měsíce) a četností výskytu zdrojů v článcích (3 a více) získáme vcelku objektivní výsledek nejčastěji využívaných zdrojů, představených v tabulce níže (Tabulka 15).

zdroj		Scratch	Code.org	Code Combat	Tynker	Hour of Code
Pořadí	v návštěvnosti webu	1	2	6	5	8
	v počtu výskytů v článcích	1	2	5	9	7

Tabulka 15: Nejčastěji využívané zdroje

Do další části práce, tedy k hodnocení byly vybrány následující zdroje: **Scratch**, **Code.org**, **Kodu Game Lab**, **Blockly Games**, **Code Combat** a **Lightbot**. Prvním důvodem eliminace ostatních zdrojů je jejich malá významnost v porovnání s vybranými. Druhým důvodem je absence češtiny. Výběr byl proveden také s ohledem na věk žáků ZŠ (11 až 13 let) a jejich zkušenosti v oblasti programování (minimální nebo žádné). Hour of Code bude z hodnocení vyřazena, neboť v principu využívá ostatních zdrojů, například Scratch, Tinker, Code Combat, Blockly Games nebo Lightbot a především kreativních studií Code.org. Tím se jako samostatný zdroj stává nehodnotitelným.

5. Zhodnocení dostupných online zdrojů určených pro podporu výuky algoritmizace a programování dětí.

V úvodu hodnocení budou stanovena evaluační kritéria, která nám poslouží pro hodnocení a následný výběr vhodných online zdrojů určených pro podporu výuky. Janík, Knecht a Najvar (2010, s. 43) představují evaluaci jako proces shromažďování empirických dat a vyhodnocování těchto nashromážděných dat na pozadí teoretických východisek s následkem zisku relevantní informace pro rozhodování a praktické jednání. Autoři tvořící evaluační kritéria a soustředící se na online zdroje, hodnotí především dva typy:

- výukové programy a aplikace;
- výukové webové stránky.

Vzhledem k povaze vybraných zdrojů bude v této práci vycházeno především z evaluačních kritérií výukových programů a aplikací s přihlédnutím na evaluační kritéria webových stránek.

5.1 Zmapování evaluačních kritérií

Nejprve je vhodné seznámit se s kritérii od autorů, kteří se touto problematikou ve svých pracích zabývají. Kritériem, které již nebude bráno v úvahu, je jazyková mutace. Všechny zdroje bez podpory češtiny již byly vyřazeny.

5.1.1 Kritéria evaluace výukových programů podle Vaníčka

Vaníček v článku *„Kritéria evaluace výukových programů pro vyučování matematiky pomocí počítače“* [cit. 2018-10-07] apeluje na potřebu nejprve posoudit výukové programy a až teprve poté program koupit. K posouzení by měla posloužit kritéria, která jsou pro naši potřebu zredukována na kritéria obecná. Zanedbána jsou kritéria oborová – ta, která se týkají matematiky.

Obecná kritéria jsou následující: validita, obsahová správnost, ergonomie, věkově přiměřené didaktické metody, motivační aspekty, robustnost, možnost změny nastavení parametrů, nápověda, zpracovaná metodika použití (manuály nestačí), cena, styl výuky.

5.1.2 Hodnotící kritéria podle Klementa

Klement (2005, s. 26) představuje čtrnáct hodnotících kritérií, jež dělí na kritéria nezbytná a kritéria doplňková. K nezbytným hodnotícím kritériím řadí: ovládání, aktuálnost a terminologickou správnost obsahu, přehlednost a snadnou ovladatelnost, obousměrnou komunikaci (interaktivitu), dostatek procvičovacích prvků, nabídku více možných řešení, způsob využití ve výuce, soulad s příslušným kurikulem a uplatňování mezipředmětových vztahů.

Mezi doplňková kritéria Klement řadí: možnost modifikace obsahu, přítomnost multimédií a nápovědy, logické uspořádání obsahu a přiměřenost cílové skupině uživatelů, podněty pro představivost, zapamatování, porozumění a pro praktické využití získaných poznatků.

5.1.3 Evaluační kritéria aplikací podle Walkera

Walker (2012) se zabýval aplikacemi vhodnými do výuky. Z důvodu značného množství aplikací na trhu vytvořil evaluační tabulku. Tabulka obsahuje celkem sedm domén, které autor chápe jako stěžejní, a to konkrétně soulad s kurikulárními dokumenty, autentičnost prostředí (původnost), zpětná vazba, diferencovanost (flexibilita), přátelské uživatelské prostředí, motivace a účinnost aplikace na výkon žáka.

5.1.4 Vyhodnocování výukových programů podle Brdičky

Brdička (1995) v hypertextové učebnici Učení s počítačem představuje další pohled na hodnocení výukových programů. Podle autora jsou nejdůležitější prvky pro posuzování kvality první dojem, cena, nápověda (nejdůležitější prvek programu), uživatelské rozhraní, dostupnost všech potřebných funkcí ze všech míst, diferencovaný přístup k žákům, plnění stanovených cílů.

5.1.5 Evaluační kritéria aplikací podle Vincenta

Vincent (2012) se na svém webu *„Learning in hand“* v článku *„Cesty k evaluaci výukových aplikací“* zabývá vhodností zařazení aplikací do vzdělávacího procesu. K tomuto účelu vytvořil hodnotící tabulku, dostupnou na <https://learninginhand.com/rubric>, do které zařadil tato evaluační kritéria: relevantnost, přizpůsobivost, zpětnou vazbu, kognitivní dovednosti, motivaci,

sledování výkonu studenta, použitelnost. Dalším nástrojem, který Vincent představuje, je hodnotící seznam (checklist), v němž najdeme:

- Odpovídá záměru a potřebám studenta.
- Součástí je nápověda nebo tutoriál.
- Vhodný obsah pro studenty.
- Informace jsou bezchybné, věcné a spolehlivé.
- Obsah lze exportovat, kopírovat nebo tisknout.
- Nastavení a/nebo obsah aplikace lze přizpůsobit.
- Přizpůsobený obsah lze převést do jiných zařízení.
- Uchovává studijní výsledky.
- Design aplikace je funkční a vizuálně stimulující.
- Student může kdykoli ukončit aplikaci bez ztráty progresu.
- Pracuje s možnostmi usnadnění jako je VoiceOver a Speak Selection.
- Je zdarma.
- Pro plnohodnotné použití aplikace nejsou nutné žádné nákupy v aplikaci.
- Rychle se načítá a je stabilní.
- Neobsahuje reklamy.
- Aktualizována za posledních 6 měsíců.
- Podporuje kreativitu a představivost.
- Poskytuje příležitosti k využívání vyšších kognitivních funkcí.
- Podporuje spolupráci a sdílení nápadů.
- Poskytuje užitečnou zpětnou vazbu.

5.1.6 Evaluační kritéria vhodných zdrojů podle Dustina Le

Le (2015) si při výběru vhodných zdrojů pro svůj článek na Edudemic.com stanovil následující kritéria: jednoduchost prostředí, estetika a design, zábavnost, efektivita, cena.

5.1.7 Kritéria hodnocení podle Řezníčkové

Řezníčková (2014, s. 26 – 28) se ve své diplomové práci zabývá tvorbou vlastního výukového programu. Před jeho tvorbou se seznamuje s kritérii, podle nichž jsou programy hodnoceny. V práci uvádí následující: kvalitní dokumentace, metodický návod jako popis a rady jak program do výuky začlenit, jednoduchost a přirozenost

ovládání, možnost zpětné vazby, podpora multimédií, řízení průchodu programem na základě reakce uživatele, cena, pravidelná aktualizace, nápověda.

5.1.8 Kritéria hodnocení online forem vzdělávacích materiálů podle Pexy

Pexa (2011) se ve výzkumné části své disertační práci zabýval především návrhem hodnotících kritérií pro edukační webové stránky. Šedesát vybraných kritérií bylo předloženo k hodnocení vzorku učitelů základních a středních škol v ČR. Ti hodnotili nejvýše následující kritéria: obsahově a gramaticky správný obsah, výskyt multimediálních a dynamických prvků, snadná orientace, využití v různých formách výuky (frontální, individualizovaná).

5.1.9 Kritéria vyplývající z vlastního dotazníkového šetření

Faktory, jež učitelé spatřují v preferovaném prostředí jako výhodné (Tabulka 10), jsou především:

- líbivost, zábavnost a atraktivita prostředí jako motivační prvek;
- přístupnost odkudkoli, kdykoli, z libovolné platformy;
- žádné nebo minimální finanční náklady;
- jednoduchost a absence syntaxe jako možnost soustředit se jen na řešení problému, pochopení algoritmů a programovacích konstrukcí;
- názornost a okamžitá zpětná vazba, možnost nahlédnutí do programu jiných projektů;
- možnost pracovat v prostředí nezávisle na věku, resp. ve značném věkovém rozptylu.

5.2 Vlastnosti programovacího jazyka v edukačním prostředí podle Paperta

Na závěr nelze nevzpomenout na Seymoura Paperta. Autoři (Resnick, Silverman, Kafai, aj., 2009, s. 63) v článku *„Scratch: Programming for All“* připomínají myšlenku Seymoura Paperta – otce dětského programování, který vyslovil základní kritéria programovacích jazyků v edukačním prostředí, a to že programovací jazyk by měl mít nízkou podlahu (low floor) – měl by být snadný pro začátečníka, vysoký strop (high ceiling) – měl by umožnit tvorbu komplexních projektů, a dále by měl mít

široké zdi (wide walls) – měl by podporovat typově rozmanité projekty, protože děti mají mnoho různých zájmů a také různé styly učení. Tyto vlastnosti by měl mít dobrý prostředek vhodný pro výuku programování, a proto budou zohledněny ve výběru evaluačních kritérií.

5.3 Vybraná evaluační kritéria

Již ve 4. kapitole této práce bylo využito prvního z kritérií k výběru vhodných zdrojů. Tímto výběrovým kritériem byla lokalizace českého jazyka. V průběhu mapování byly vyřazeny všechny zdroje, které nepodporují češtinu.

Vybraná kritéria vyplývající ze zmapovaných evaluačních kritérií autorů jsou v této práci grupována do tří hlavních skupin:

- vlastnosti a cena;
- design a uživatelská přívětivost;
- didaktické aspekty.

První skupina nesoucí název *„vlastnosti a cena“* představuje soubor kritérií, který se zaměřuje na evaluaci obsahové části. Obsah by měl být svou obtížností uzpůsoben věku dítěte a měl by být pro žáka srozumitelný. Do tohoto rámce kritérií je také řazen způsob tvorby kódu. Při tvorbě by se dítě mělo plně věnovat řešení problému, nikoli řešit syntaktické chyby, jak uvádí Resnick (2010), Young (2018). Měl by umožňovat práci se základními primitivy, a to práci s příkazy, sekvencemi, s voláním procedur nebo funkcí, a dále by měl umožňovat práci s řídícími strukturami – selektivním nebo opakovaným prováděním příkazů. Přístup ke zdroji by měl být umožněn z libovolné platfomy. Měl by být také k dispozici zdarma, neměl by být zpoplatněn přístup k dalším částem obsahu. Zdroj by měl kromě nenáročnosti na tvorbu kódu umožnit vytvoření komplexních a různorodých projektů, tedy měl by se držet Papertových požadavků.

Označení kritérií:

K1: přístup – možnosti přístupu uživatele ke zdroji (odkudkoli, kdykoliv), přístup je umožněn z libovolné platformy;

K2: věková přiměřenost – obsah je svou náročností uzpůsoben věku uživatele, a to včetně stylu tvorby kódu;

K3: primitiva – pokrývá práci se základními primitivy (sekvence, selekce, iterace);

K4: cena;

K5: komplexně-různorodé projekty.

Druhá skupina pojmenovaná *„design a uživatelská přívětivost"* představuje soubor kritérií, který se soustředí na evaluaci funkčnosti designu zdroje. Každý zdroj by měl obsahovat nápovědu nebo ucelený systém tutoriálů. Přítomnost nápovědy zdůrazňují například Brdička (1995), Vaníček [cit. 2018-10-07], Řezníčková (2014), Klement (2005), Vincent (2012) nebo Pexa (2011). Nápověda je podstatným elementem, který představuje pomoc při potížích s ovládáním nebo pomoc s orientací v prostředí. Jednotlivé prvky uživatelského rozhraní by měly být vhodně umístěny tak, aby se žák v prostředí rychle a snadno zorientoval. Všechny potřebné funkce jsou dostupné ze všech míst. Celé prostředí by mělo být dostatečně jednoduché a mělo by zohledňovat cílovou skupinu – žáky. Žák by při práci neměl být příliš závislý na učiteli. Design by měl respektovat specifika věkové skupiny, například vhodnou velikostí písma, počtem slov v řádku, barevností. Design by měl být zajímavý, zábavný, měl by dostatečně motivovat, neboť podle Rambouska (2014, s. 27) lze motivaci chápat jako soubor činitelů, které podněcují, řídí a udržují žáka v činnosti, kterou žák provádí.

Označení kritérií:

K6: ergonomie – vhodné rozložení elementů grafického uživatelského rozhraní;

K7: pedagogicko-psychologické aspekty – prostředí by mělo být jednoduché, vhodné pro rychlou orientaci a usnadnění práce se zdrojem, písmo (typ fontu, velikost písma aj.) by mělo být uzpůsobeno věku uživatele, stejně, jako barevnost grafického uživatelského rozhraní;

K8: vizuální aspekt – atraktivní prostředí;

K9: motivace – schopnost udržet si uživatele, který zdroj s oblibou využívá;

K10: průvodnost – usnadňuje práci se zdrojem a orientaci v prostředí například pomocí nápovědy nebo tutoriálů.

Poslední skupinou je skupina *„didaktické aspekty"*, která představuje soubor kritérií hodnotícího zdroje především z pedagogického pohledu implementace zdroje

do vyučovacího procesu. Zdroj by měl disponovat kurikulem představujícím obsah uspořádaný do logicky navazujících částí a metodickými dokumenty, které představují popisy a rady, jak práci se zdrojem do výuky implementovat v rámci své interní struktury, nebo odkazovat na jiné zdroje, které metodickou dokumentaci obsahují. Důležitou funkcí je z psychologicko-pedagogického i didaktického pohledu funkce kontrolně-diagnostická, která, jak upozorňuje Rambousek (2014, s. 2), prolíná celý vyučovací proces. Hlavním elementem, který zajišťuje tuto funkci je zpětná vazba. Zdroj by měl zprostředkovávat okamžitou zpětnou vazbu, v ideálním případě vizuální povahy. Dále by měl umožňovat aplikaci různých organizačních forem výuky (frontální, kooperativní, projektová, individualizovaná atd.) a výukových přístupů (instruktivní, konstruktivistický) podle potřeb učitele. Zdroj by měl podporovat kreativitu, představivost, a především poskytovat příležitosti k využívání vyšších kognitivních funkcí.

Označení kritérií:

K11: zpětnovazební prostředky – zprostředkovává zpětnou vazbu;

K12: dostupnost kurikula a **metodické dokumentace** – k dispozici je výukový design představující logicky navazující části, dále přestavuje způsoby, jak zdroj začlenit do výuky;

K13: organizační formy výuky – diferencovaný přístup umožňující kombinaci různých forem výuky;

K14: výukové aktivity – podporuje kreativitu a představivost, poskytuje příležitosti k využívání vyšších kognitivních funkcí, řešení problémů, umožňuje učení hrou;

K15: personalizace – umožňuje vytvoření účtu a přihlášení, uchovává studijní výsledky, sleduje výkon dítěte;

K16: sociální aspekty – v rámci zdroje existuje komunita uživatelů, zdroj umožňuje sdílení vytvořených programů.

5.4 Důležitost vybraných kritérií

Zdroje, vybrané v kapitole 4.2.5 budou hodnoceny patnácti evaluačními kritérii, přičemž pro každé kritérium je nastavena váha z intervalu $\langle 1; 3 \rangle$, která vyjadřuje

důležitost evaluačního kritéria v porovnání s ostatními (Tabulka 16). Důležitost kritérií je určena podle subjektivního dojmu.

Skupina	Kritérium	Váha kritéria
Vlastnosti a cena	K1: přístup	1
	K2: věková přiměřenost	3
	K3: primitiva	2
	K4: cena	1
	K5: komplexně-různorodé projekty	2
Design a uživatelská přívětivost	K6: ergonomie	2
	K7: pedagogicko-psychologické aspekty	1
	K8: vizuální aspekt	1
	K9: motivace	3
	K10: průvodnost	1
Didaktické aspekty	K11: zpětnovazební prostředky	3
	K12: dostupnost kurikula a metodické dokumentace	2
	K13: organizační formy výuky	2
	K14: výukové aktivity	2
	K15: personalizace	1
	K16: sociální aspekty	2

Tabulka 16: Nastavení důležitosti jednotlivých kritérií (váhy kritérií)

5.5 Proces hodnocení vybraných oline zdrojů evaluačními kritérii

Vybrané zdroje jsou seřazeny do žebříčku podle míry splnění daného kritéria. Konkrétní pozice v žebříčku může nabývat hodnot z intervalu $\langle 1; 6 \rangle$. V případě shodnosti, respektive významné podobnosti v rámci daného kritéria, je zdrojům udělena stejná hodnota (Tabulka 17).

Kritérium	Scratch	Code.org	Kodu	Blockly Games	Code Combat	Lightbot
K1	4	1	6	1	1	4
K2	1	1	1	5	6	1
K3	2	1	5	2	4	6
K4	1	1	1	1	6	5
K5	1	3	2	4	4	6
K6	1	1	1	5	6	1
K7	1	1	1	6	1	1
K8	5	3	1	6	1	3
K9	4	4	1	6	3	1

K10	1	4	1	6	4	1
K11	1	1	1	1	1	1
K12	1	1	3	6	4	5
K13	1	3	1	4	4	6
K14	1	3	1	4	4	4
K15	1	1	4	4	1	6
K16	1	2	2	5	4	5

Tabulka 17: Žebříček zdrojů podle míry splnění jednotlivých kritérií

Pro všechny zdroje je vypočítán vážený průměr podle vzorce

$$\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i},$$

kde w_i představuje váhy jednotlivých kritérií a x_i představuje konkrétní pozice v kriteriálních žebříčcích. Vážený průměr je vypočítán také pro všechny tři hlavní skupiny. Výsledky výpočtů včetně finálního žebříčku jsou uvedeny v následujících tabulkách (Tabulky 18 až 21).

Celkový výsledek zahrnující všechna kritéria						
zdroj	**Scratch**	Code.org	Kodu	Blockly Games	Code Combat	Lightbot
$\bar{x}$	**1,62**	1,97	1,83	4,14	3,59	3,28
pořadí	**1**	3	2	6	5	4

Tabulka 18: Výsledné hodnoty a pořadí zdrojů

Výsledek pro skupinu kritérií „vlastnosti a cena“						
zdroj	Scratch	**Code.org**	Kodu	Blockly Games	Code Combat	Lightbot
$\bar{x}$	1,56	**1,44**	2,67	3,22	4,56	4,00
pořadí	2	**1**	3	4	6	5

Tabulka 19: Výsledek hodnocení podle skupiny kritérií "vlastnosti a cena"

Výsledek pro skupinu kritérií „design a uživatelská přívětivost“						
zdroj	Scratch	Code.org	**Kodu**	Blockly Games	Code Combat	Lightbot
$\bar{x}$	2,63	2,75	**1,00**	5,75	3,38	1,25
pořadí	3	4	**1**	6	5	2

Tabulka 20: Výsledek hodnocení podle skupiny kritérií "design a uživatelská přívětivost"

Výsledek pro skupinu kritérií „didaktické aspekty“						
zdroj	**Scratch**	Code.org	Kodu	Blockly Games	Code Combat	Lightbot
$\bar{x}$	**1,00**	1,83	1,75	3,75	3,00	4,08
pořadí	**1**	3	2	5	4	6

Tabulka 21: Výsledek hodnocení podle skupiny kritérií „didaktické aspekty“

Z výsledků hodnocení vyplývá, že nejlépe využitelným zdrojem je Scratch, a to podle celkového průměru všech kritérií. Kromě toho je Scratch nejlépe hodnoceným také v oblasti didaktických aspektů zdrojů. Mírné nedostatky jsou spatřovány v oblasti designu a uživatelské přívětivosti, jež jsou způsobeny především komplexností a možnostmi nastavení, které Scratch umožňuje, a dále množstvím příkazů, jež lze využívat. Problematickým kritériem se také jeví kritérium K1. Online verze Scratche pro svůj běh využívá Flash Player, který například nepodporují zařízení Apple. Pro tato zařízení je však k dispozici offline verze. Nová verze Scratche, známá pod číselným označením 3.0, již funguje plně na platformě HTML5. Tato, zatím ještě testovaná beta verze, je k dispozici na <https://beta.scratch.mit.edu/>. Scratch je nejlépe hodnoceným zdrojem v didaktické oblasti, a to u všech šesti kritérií, které na oblast didaktických aspektů cílily.

Dalším zdrojem vhodným pro výuku programování je podle výsledků Kodu vytvořený skupinou Microsoft's FUSE Lab v roce 2009. Také Kodu neuspělo v hodnocení kritériem K1. Vzhledem k povaze Kodu jako vývojového prostředí pro tvorbu her s využitím vlastního vizuálního programovacího jazyka je uživatelům umožněn pouze offline přístup. Navíc 3D prostředí klade vyšší nároky na hardware uživatele. Další nedostatek je shledán v hodnocení kritériem K3. Vizuální programování je založeno na gramatice podmínka – akce (when – do, always – do), využívané především u edukačního programování robotů, například Lego Mindstorms. Kodu je vhodé vyzdvihnout především v oblasti grafického designu a uživatelské přívětivosti. U všech hodnotících kritérií (K6 až K10) dosáhl v porovnání s ostatními nejlepších výsledků.

Třetím nejlépe hodnoceným zdrojem je podle výsledků web organizace Code.org (založená v roce 2013). Code.org je nejlepším z vybraných zdrojů v oblasti obsahové. Umožňuje využívat širokou řadu aktivit či kurzů rozdělených podle věku nebo schopností žáků. Všechny aktivity jsou k dispozici zcela zdarma.

V podstatě záleží na učiteli, zda upřednostní spíše graficky propracované prostředí Kodu, které bude žáky motivovat v práci s ním, ale oželí přitom výuku vedoucí k pochopení problematiky procedur a jejích parametrů nebo k pochopení principu proměnných v programu. Nebo naopak využije prostředků Scratche či Code.org k vysvětlení výše zmíněného učiva na úkor krásného 3D prostředí. Učitelé také mohou preferovat výuku v některém profesionálně využívaném programovacím nebo skriptovacím jazyce, například v Pythonu nebo JavaScriptu. Potom se jako vhodný jeví spíše Code Combat nebo některá z aktivit Hour of Code.

6. Návrh vlastního modelu výuky

Hlavním cílem této práce je návrh vlastního modelu výuky, který by mohl být aplikovatelný ve vzdělávání na základní škole. Model bude stavět na teoretických základech, výsledcích dotazníkového šetření a výsledcích hodnocení zdrojů.

V úvodu návrhu nového modelu výuky je vhodné odpovědět si na několik základních otázek.

Jaké prostředí a nástroje budou využívány?

Návrh vychází ze získaných poznatků z předchozích kapitol. Model by měl sloužit skupině učitelů, která má jen minimální zkušenosti s výukou zaměřenou na tematický okruh algoritmizace a programování. Proto budou do modelu výuky zvoleny především ty zdroje, které patří k nejvyužívanějším – zdroje, pro které je zdarma k dispozici podpora v podobě různých druhů pedagogických materiálů, například kurikulární nebo metodická dokumentace, takové zdroje, kterých využívá široká základna aktivních uživatelů (učitelů), neboť rozsáhlá aktivní základna představuje vhodnou podporu ve formě rad a návodů pro začínající učitele, například na webech zaměřených na metodiku, diskusních fórech nebo na sociálních sítích.

Zvolenými zdroji budou v první řadě Scratch (online verze) a Code.org kurz. Do modelu budou začleněny unplugged metody vybrané z Hour of Code (Code.org), upravené úlohy ze soutěže Bobřík informatiky a CS Unplugged. Dále bude využito herní prostředí Lightbot a Lightbot 2. K tvorbě vývojových diagramů bude využita aplikace PS Diagram. PS Diagram představuje Bartyzal [cit. 2018-10-30] jako: *„Aplikaci pro rychlou a pohodlnou tvorbu algoritmů v podobě vývojových diagramů. Vývojové diagramy lze pak rozpohybovat a pozorovat, jak se např. mění parametry cyklu, jakým způsobem algoritmus rozhoduje, a jak se mění proměnné.“* Aplikace PS Diagram je vybrána nezávisle na výsledcích zmapovaných zdrojů. Důvodem k zařazení aplikace, respektive vývojových diagramů je seznámit žáky s reálně využívanou možností grafického znázornění algoritmů.

Jaké pojetí výuky bude preferováno?

V modelu bude upřednostňováno konstruktivistické pojetí výuky s důrazem na sociální dimenzi učení (práce ve dvojicích, skupinách) a na okamžitou vizuální

zpětnou vazbu. Se základními pojmy a uživatelským rozhraním pro jednotlivá prostředí budou žáci seznamováni tradiční cestou – transmisivně, a to především z důvodu efektivnosti a úspory času.

Jakou strukturu bude model mít a jaké konkrétní aktivity bude model obsahovat?

Model bude svou strukturou vycházet především ze dvou zahraničních publikací a z vlastní zkušenosti vyučujícího. Berry (2013, s. 15) v návrhu výuky předmětu Computing v Anglii uvádí pořadí: *„želví grafika; tvorba animací pomocí kódu; vytvoření matematického kvízu; tvorba počítačové hry.“* Kurikulární průvodce organizace Code.org na <http://fliphtml5.com/pcrbk/tfbw/basic> v části *„Úvod do programování“* [cit. 2018-10-20] představuje měsíční plán obsahující celkem 10 lekcí. V prvním týdnu se zaměřuje na vysvětlení pojmů algoritmů, programování a jejich důležitosti. Plán dále obsahuje aktivity bez počítače (unplugged), orientované na algoritmizaci. V druhém týdnu jsou žáci seznámeni se sekvencemi, využíváním funkcí a tvorbou vlastních funkcí. Třetí týden je zaměřen na parametry funkcí, cykly a náhodná čísla. Poslední týden pracují žáci ve skupinách na vlastním projektu, ve kterém uplatňují získané znalosti.

Model výuky bude sestaven ze čtyř hlavních výukových bloků, které budou reflektovat výše zmíněná pořadí v představených plánech. Každý blok se bude od ostatních lišit především ve využívaných prostředcích a různorodosti zařazených aktivit.

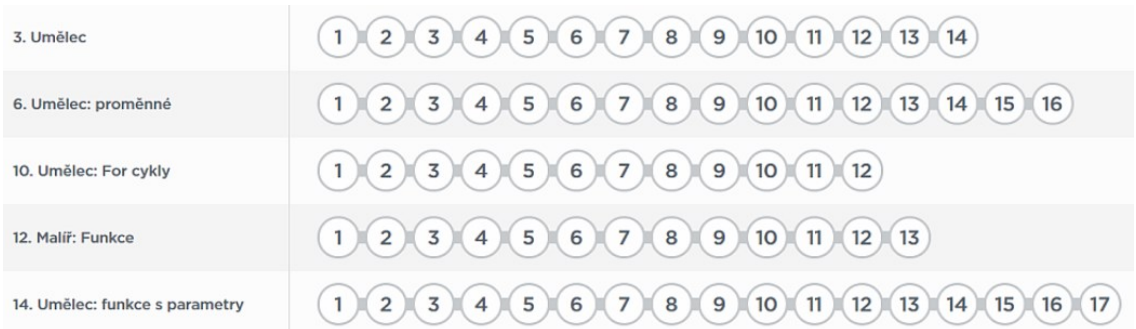
V prvním bloku budou žáci seznámeni se základními pojmy – algoritmem a vlastnostmi algoritmu, programem, programováním, programovacím jazykem a vývojovým diagramem, jako standardně využívaným grafickým zápisem algoritmu. První blok bude veden instruktivně. Pro potřebu zpětné vazby bude využito testu prostřednictvím aplikace Plickers. Testové otázky se zaměří na pochopení pojmů a jejich vzájemných vztahů, například na pochopení vzájemného vztahu mezi algoritmem a programem. Alternativou k testu by mohla být například tvorba pojmové mapy.

Druhý blok bude dále rozdělen na tři části. Do první části druhého bloku budou zařazeny unplugged aktivity. Aktivity budou zaměřeny především na aplikaci znalostí získaných v prvním bloku. Žáci by hned v úvodních cvičeních měli prožít,

jak je nezbytné přesně a jednoznačně formulovat jednotlivé pokyny v programu. Prožitek by měly zajistit aktivity bez počítače, ve kterých žáci instruují stroj k činnosti. Prožitek by měl mít také inverzní charakter – z pohledu stroje, který se instrukcemi řídí.

V druhé části druhého bloku se žáci seznámí s pojmem podprogram. Dále si vyzkouší řídit robota skládáním limitovaného počtu ikonek s instrukcemi ve hře Lightbot a Lightbot 2.0. Tato aktivita je vybrána především pro svou jednoduchost – žáci by měli uspět na všech úrovních. Pocit z úspěchu by tak měl zvyšovat vnitřní motivaci žáků. Kromě motivace je Lightbot vybrán jako první, neboť žáka vede k tvorbě efektivního kódu s eliminací redundantních příkazů. Hravou formou představuje procedury jako nástroj strukturovaného programování, který navíc podporuje abstraktní myšlení. Zpětná vazba má vizuální charakter a je poskytována okamžitě v podobě chování robota. Ladění programu se tak stává snazším. Zároveň je však nutné upozornit na fakt, že Lightbot představuje dětem rekurzi (volání funkce v těle samotné funkce) jako cyklus, což není didakticky vhodné řešení. Alternativou by mohla být například aplikace Cargo-Bot. Tvůrčí aktivitu budou žáci uplatňovat v závěru druhé části, v prostředí hry Lightbot 2.0, která umožňuje vytvářet vlastní herní úrovně, a tím u žáků podpoří tvořivost.

Ve třetí a poslední části druhého bloku budou žákům představeny pojmy proměnná a parametry funkce. Znalost nových pojmů budou žáci aplikovat v dalších aktivitách. Tyto aktivity jsou součástí kurzu 4 studia organizace Code.org. Kurz umožňuje procházet úrovně lekcí ve dvojicích, čehož by mělo být využito. Počet lekcí v kurzu bude snížen z 22 na 5, a to především z časových důvodů. Konkrétní výběr lekcí je patrný z obrázku níže (Obrázek 1).



Obrázek 1: Vybrané lekce (PrintScreen z webu <https://studio.code.org/s/course4> [cit. 2018-11-3])

Vybrané lekce svým pojetím vycházejí z principů Loga, respektive želví grafiky. Želvu nahradilo studio Code.org malým malířem, kód je tvořen prostřednictvím Blockly. Jednotlivé lekce se učivem zaměřují na sekvence, cykly, proměnné, cykly se známým počtem opakování, funkce a funkce s parametry. Žák má opět okamžitou vizuální zpětnou vazbu, která napomáhá pochopení jednotlivých bloků v programu, a dále usnadňuje dětem ladění svých programů. V neposlední řadě bude v závěrečné části druhého bloku zvoleno block-based programování z důvodu snazšího přechodu do prostředí Scratch, jenž bude primární náplní obsahu výuky nejen třetího bloku, ale také celého navrhovaného modelu.

Třetí blok bude využívat prostředí Scratch – online verze dostupné na adrese <https://scratch.mit.edu/>. Scratch je zvolen především pro svou univerzálnost, kterou poskytuje, od práce s obrazem a zvukem přes tvorbu animovaných interaktivních příběhů či simulaci reálných situací až po tvorbu komplexních her a projektů. Z pedagogického hlediska lze vyzdvihnout především rozvíjení tvořivého myšlení, plánování, navrhování a v neposlední řadě učení se spolupracovat ve skupině s ostatními. Třetí blok bude rozdělen do jednotlivých lekcí, které by měly obsahem zahrnovat řídicí struktury, podmínky využívající operátory a logické operátory, práci s proměnnou a podprogramy, a to se vzrůstající náročností. Žáci budou ve dvojici (případně samostatně) tvořit interaktivní animace, soutěže a hry. V závěru bloku žáci ve dvojici naplánují a vytvoří vlastní projekt.

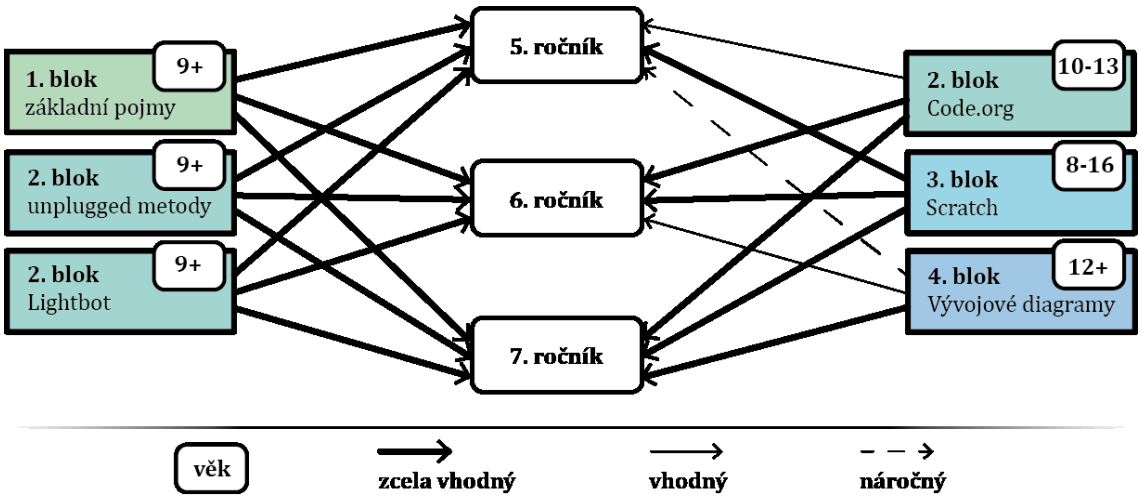
Čtvrtý blok se svým obsahem bude orientovat na reprezentaci algoritmů, a sice na vývojové diagramy. Žák se nejprve seznámí s využívanou symbolikou a pravidly tvorby diagramů. Naučí se diagramy jak číst, tak tvořit. Tvorbu a následnou vizuální zpětnou vazbu by mělo zajišťovat prostředí aplikace PS Diagram. Funkčnost sestaveného algoritmu si žáci ověří v prostředí Scratch nebo v Code.org studiu.

V samotném závěru budou žákům představeny další webové projekty a software, které mohou využívat k dalšímu – informálnímu učení, například Kodu, Scratch 3.0 (HTML5 verze), komplexnější Stencyl – software pro tvorbu her nebo stavebnici Lego Mindstorms.

Pro jaký ročník, respektive věk žáků bude model vytvořen?

Aktuální situace ve vyspělých zemích posouvá hranice počátku výuky algoritmizace již do mateřských škol, respektive prvních ročníků základních škol. Avšak u předškoláků a žáků prvních ročníků se předpokládá, že ve výuce budou využívat především programovatelné hračky či prostředky tvorby kódu, které jsou svým designem přizpůsobeny kognitivnímu vývoji malých dětí, například včelku Bee-Bota nebo aplikaci ScratchJr.

Model výuky bude navržen jako úvod do algoritmizace a programování pro začátečníky, bez vymezení konkrétního ročníku. Obsahově však bude cílit především na žáky ve věku 11 a více let. Tímto krokem je reagováno na poznatky zjištěné v dotazníkovém šetření, ve kterém učitelé označili jako počáteční období realizace výuky algoritmizace a programování 5. a 6. ročník ZŠ (Tabulka 5). Jednotlivé bloky mohou být zařazeny do výuky v rámci jednoho ročníku, nebo mohou být rozvrhnuty do tematických plánů několika ročníků. Model může být také využit v rámci zájmového útvaru. Je však doporučeno dodržovat pořadí jednotlivých bloků. Obtížnost obsahu jednotlivých bloků a jejich předpokládané nasazení je představeno na obrázku níže (Obrázek 2).



Obrázek 2: Navrhované možnosti nasazení bloků do ročníků

Jaké požadavky jsou kladeny na obsah výuky?

Obecně se dá říci, že obsah výuky bude ve hlavním bloku číslo tři vybírán v souvislosti s obsahem výuky matematiky především 6. ročníku (celá čísla, dělitelnost, velikost úhlu). Jednotlivé aktivity tak mohou mít mezipředmětový přesah právě do matematiky. Žáci 6. ročníku ještě nemají příliš zkušeností s pojmy

z oblasti počítačové grafiky ani zkušenosti s tvorbou obrázků v rastrových či vektorových editorech. Proto budou pro většinu aktivit ve Scratchi předpřipraveny již hotové objekty.

Jaké by měly být očekávané výstupy v navrhovaném modelu?

Jelikož RVP ZV nás s žádným konkrétním obsahem pro oblast programování neseznamuje, bude tato práce vycházet ze tří pedagogických dokumentů, respektive publikací.

Prvním dokumentem je návrh rámce očekávaných výstupů nového předmětu – Informatika, jenž je dostupný na webu Národního ústavu pro vzdělávání <http://www.nuv.cz/file/3361/>. Dokument pokrývá tematický celek algoritmizace a programování na základních a středních školách. Návrh pro tematický celek Algoritmizace a programování určený pro základní vzdělávání je představen v tabulce níže (Tabulka 22).

Tematický celek	*1. stupeň ZŠ*	*2. stupeň ZŠ*
Algoritmizace a programování	*formuluje otázky, odpovídá; řeší problémy, úkoly a situace, myslí kreativně, předkládá „nápady“; nalézá nová řešení nebo alternativní k běžným; postupuje a učí se podle pokynů a instrukcí, popisuje známé postupy v účelném pořadí jednotlivých kroků; stanovuje postupy/kroky řešení elementárních/jednoduchých problémů; určuje příčiny a následky v pozorovaných dějích; sleduje a vypráví příběh, pohádku;*	*přečte textový nebo symbolický zápis algoritmu a vysvětlí jeho jednotlivé kroky*
		popíše jednoduchý problém, navrhne a popíše jednotlivé kroky jeho řešení
		upraví připravený postup pro obdobný problém; ověří správnost jím navrženého postupu, najde a opraví v něm případnou chybu
		rozpozná různé algoritmy, které vedou ke stejným výsledkům
		v blokově orientovaném programovacím jazyce sestaví program; program otestuje a opraví v něm případné chyby
		rozpozná opakující se vzory, používá opakování a připravené podprogramy; používá události ke spouštění podprogramů

Tabulka 22: Navrhovaný rámec očekávaných výstupů předmětu Informatika [cit. 2018-10-20]

Druhým zdrojem je publikace průvodce pro anglické učitele primárního vzdělávání *„Computing in the national curriculum: A guide for primary teachers“*, vytvořeným sdružením Computing At School, stejně jako třetí publikace *„Computing in the national curriculum: A guide for secondary teachers“*, která pro

změnu cílí na učitele sekundárního vzdělávání. Z prvostupňového průvodce – klíčových fází 1 a 2 je vybrán následující obsah pro oblast algoritmizace a programování (Berry, 2013):

- Pochopí, co jsou to algoritmy a zná souvislost mezi algoritmem a programem.
- Pochopí důležitost přesných a jednoznačných pokynů v programu.
- Vytvoří jednoduché programy a odladí je.
- S využitím logického myšlení dokáže předvídat chování jednoduchých programů.
- Navrhne, napíše a odladí programy splňující konkrétní požadavky.
- Popíše, jak se bude chovat hotový program.
- Řeší problémy rozkladem do menších částí.
- Při tvorbě programu využívá sekvence, selekce a opakování; pracuje s proměnnými a různými formami vstupů a výstupů.
- Využívá logiky ke zjištění a opravě chyb v algoritmech.

Autor druhostupňového průvodce Kemp (2014) v klíčové fázi 3 představuje následující obsah:

- Navrhne, použije a vyhodnotí modely stavu a chování reálných problémů a fyzikálních systémů s využitím abstrakce.
- Pochopí několik klíčových algoritmů (například třídící a vyhledávací), které reflektují informatické myšlení.
- Vybere vhodný algoritmus z více alternativ, které řeší stejný problém.
- Využívá dvou a více programovacích jazyků, z nichž je alespoň jeden textový k vyřešení informatického problému.
- Vhodně využívá datovou strukturu, například seznamy, tabulky, pole.
- Navrhne a vytvoří program, ve kterém využívá procedury a funkce.
- Chápe základy Booleovy logiky – chápe význam AND, OR, NOT a jejich využití v programování.

Z předkládaných možností jednotlivých pramenů jsou do navrhovaného modelu vybrány či spojeny následující části vzdělávacího obsahu:

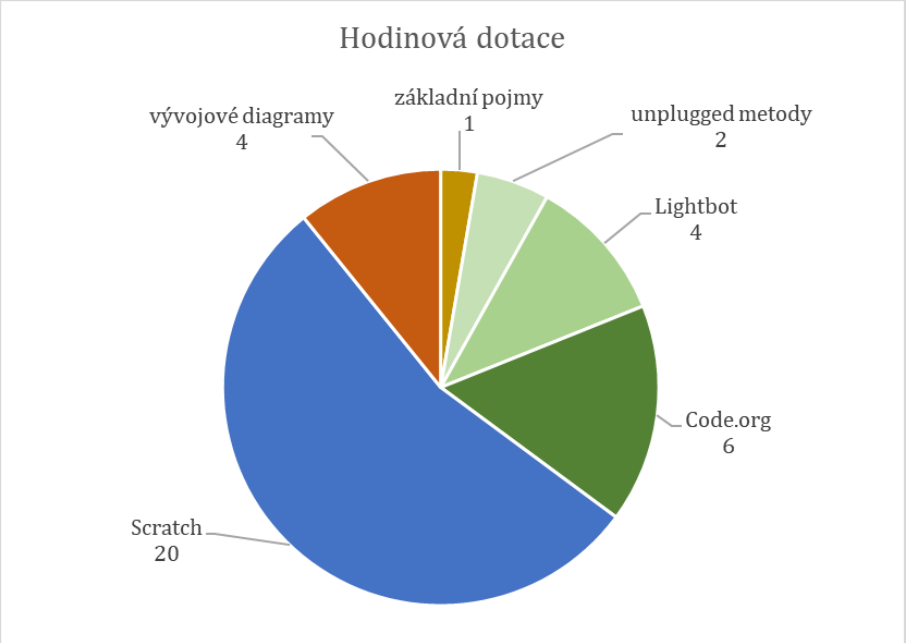
- Pochopí, co jsou to algoritmy a zná souvislost mezi algoritmem a programem.
- Pochopí důležitost přesných a jednoznačných pokynů v programu.

- Využívá informatického myšlení v aktivitách bez počítače.
- Přečte textový nebo symbolický zápis algoritmu a vysvětlí jeho jednotlivé kroky.
- Vybere vhodný algoritmus z více alternativ, které řeší stejný problém.
- Řeší problémy rozkladem do menších částí.
- Navrhne, napíše a odladí programy splňující konkrétní požadavky.
- V blokově orientovaném jazyce sestaví program, ve kterém využívá sekvence, funkce (procedury) a vlastní funkce včetně parametrů funkcí, cyklů a náhodných čísel.
- Rozpozná opakující se vzory, používá opakování a připravené podprogramy; používá události ke spouštění podprogramů.
- Chápe základy Booleovy logiky – chápe význam AND, OR, NOT a jejich využití v programování.
- Upraví připravený postup pro obdobný problém; ověří správnost jím navrženého postupu, najde a opraví v něm případnou chybu.
- Popíše jednoduchý problém, navrhne a popíše jednotlivé kroky jeho řešení.

Jaká hodinová dotace bude pro model výuky a jeho jednotlivé části nastavena?

Jak již bylo zmíněno, model bude rozdělen do čtyř hlavních bloků. Prvnímu bloku – výkladu základních pojmů bude vymezena 1 vyučovací hodina. První části druhého bloku (unplugged metodám) budou vymezeny alespoň 2 vyučovací hodiny, v jejichž průběhu by měli žáci pochopit důležitost přesného a jednoznačného vyjadřování požadavků. Druhé části – hře Lightbot a kurzu na Code.org pak bude vymezeno nejméně 10 vyučovacích hodin, přičemž aplikaci Lightbot a Lightbot 2.0 budou vymezeny konkrétně čtyři hodiny, což je odhadovaný čas k úspěšnému průchodu všech úrovní aplikace Lightbot a průchodu některých z úrovní hry Lightbot 2.0, a to včetně vytvoření a otestování vlastní úrovně hry i otestování úrovně vytvořené spolužákem. Zbylých šest hodin je vymezeno k úspěšnému průchodu pěti výukovými lekcemi kurzu na webu organizace Code.org. Třetímu bloku, který se bude zabývat především tvorbou programů ve Scratchi, bude vymezeno nejméně 20 vyučovacích hodin, v jejichž průběhu by žáci měli absolvovat celkem devět sekcí obsahujících rozdílný počet jednotlivých aktivit s různou náročností. Čtvrtý blok, zaměřený na tvorbu vývojových diagramů, bude realizován alespoň

ve 4 vyučovacích hodinách, které by měly postačit k seznámení žáků s vývojovými diagramy. Hodinová dotace bude pro jednotlivé bloky rozdělena následovně (Graf 8):



Graf 8: Hodinová dotace jednotlivých částí

Celková časová dotace tak činí nejméně 37 hodin, které je možné, vzhledem k povaze modelu, rozložit také do více ročníků.

Časově-tematický plán obsahující vybrané očekávané výstupy pro navrhovaný model je součástí přílohy (Příloha 9).

Jak budou hodnoceny výkony žáků?

Žáci budou hodnoceni především formativně. Pro potřebu klasifikace ve formě sumativní, ale také formativní je možné vycházet například z Screawnovy tabulky hodnocení [cit. 2018-10-28], jež je dostupná na webu <https://jts.design/scratch-project-rubrics/>. Součástí hodnocení aplikovaného v tomto modelu bude k oceňování práce žáků využíváno gamifikačních prvků. Žáci budou v průběhu celé výuky sbírat body – imaginární puzzle dílky. V závěru konkrétních lekcí nebo v závěru každého bloku budou podle počtu nasbíraných puzzle dílků odměněni odznakem (tzv. *„stickerkou"*) nebo oboustrannou herní kartou s tématikou bloku výuky (Příloha 10). Konkrétní rozvržení možných zisků puzzle dílků, odznaků a karet pro všechny části modelu je k dispozici v příloze práce (Příloha 12). Gamifikační prvky, jako odměny za výkon, mají podporovat především funkci

motivační, respektive podporovat pozornost žáka a jeho snahu k dosažení vytyčeného cíle, a to pomocí příjemného prostředí, ve kterém výuka probíhá a do kterého se bude rád vracet (Pávková, 2014, s. 35).

6.1 Realizace přístupu k výukovým materiálům

Obsah výukových materiálů již byl stanoven výše, především v reakci na otázku spojenou s konkrétními aktivitami, které by měl model obsahovat. Dalším krokem je zajistit přístup k těmto materiálům.

Výukové materiály by měly být dostupné odkudkoli a kdykoli, měly by být přístupné veřejnosti a měly by být nezávislé na platformě. Všechny tyto požadavky budou splněny vytvořením webové stránky, jež bude výukové materiály obsahovat. K vytvoření webu byla vybrána možnost využít předpřipravené šablony, konkrétně šablony *„Bell“* [cit. 2018-11-03], která je zdarma dostupná na webu autora šablony. Šablona byla následně upravena především po obsahové stránce. Architektura stránky pracuje pouze s prohlížečem na straně klienta, tím pádem nebylo nutné řešit vhodný webhosting a stránka mohla být nasazena na školní server.

Obsah je dostupný na adrese <https://www.3zskadan.cz/prgm/> v podobě webové stránky zahrnující všechny použité elektronické výukové materiály nebo odkazy na ně.

6.2 Sylabus výuky

- Představení webové stránky *„Programování na ZŠ“* a plánu výuky žákům
- Pojmy algoritmus, program, programování, programovací jazyk, vývojový diagram
- Unplugged metody
- Pojem podprogram
- Lightbot
- Lightbot 2.0
- Pojmy proměnná, parametry funkce
- Code.org kurz
    - Sekvence, proměnné, cykly, funkce, parametry funkce
- Scratch
    - Informace o Scratchi

    - Vytvoření uživatelského účtu a seznámení s prostředím, orientace v multiblocích a blocích
    - Princip tvorby programu, testování, ladění
    - Sekvence, proměnná, spojování řetězců, nekonečný cyklus, podmínky, cykly, zastavení programu, náhodné číslo
    - Paralelní procesy, události
    - Nový blok (podprogram)
    - Podmínky a operátory
    - Podprogramy s parametry
    - Rekurze
    - Klonování a pořadí objektů (hloubka)
- Vývojové diagramy
    - Pravidla tvorby a symbolika
    - Čtení a tvorba
    - PS Diagram – seznámení s prostředím
    - Tvorba v PS Diagramu
- Seznámení s dalšími možnostmi podporujícími prohlubování dovedností

6.3 Metodika

Žák by měl dosáhnout osvojení elementárních znalostí a dovedností z oblasti algoritmizace a programování s využitím edukačních programovacích jazyků Scratch, Blockly a herních aplikací. Osvojení dovedností by mělo být dosaženo především aktivní konstruktivní činností žáků a komunikací mezi žáky při řešení problému. V rámci této práce vznikla metodická příručka, která je součástí přílohy práce (Příloha 13). Příručka může být využita učiteli, kteří by se rozhodli zařadit navržený model nebo jeho části do své výuky. Detailněji popisuje cíle hodin, navrhované i alternativní postupy, očekávané výstupy, požadavky na nutné vybavení, náročnost aktivit, hodinovou dotaci, způsoby ověření výstupů jednotlivých částí, možná řešení a navrhované přidělení odměn.

7. Ověření návrhu modelu výuky na ZŠ

Ověření návrhu modelu výuky (odhalování jeho nedostatků, chyb ve výukových materiálech či přílišnou obtížnost aktivit) bylo realizováno pilotním nasazením modelu na základní škole. Pilotní nasazení probíhalo v období školního roku 2017/2018, konkrétně od října 2017 do června 2018, každé úterý od 14:30 do 15:30, a to v rámci zájmového útvaru vedeného autorem práce. Důvodem výběru zájmového útvaru byla předpokládaná hodinová dotace – téměř 40 vyučovacích hodin.

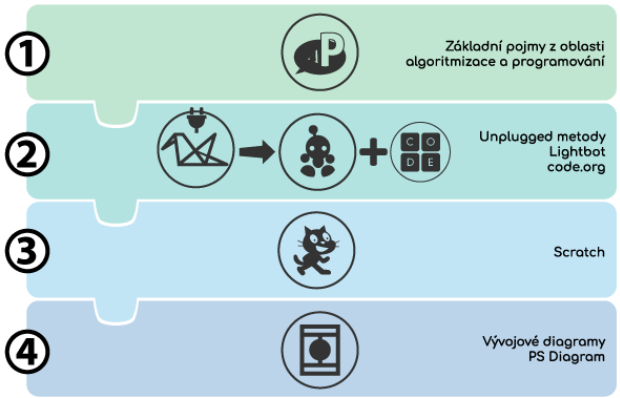
V ověřovacím procesu bylo využíváno metod soustavného pozorování a analýz prací jednotlivých skupin nebo jednotlivců. V závěru pilotního nasazení se žáci zúčastnili dotazníkového šetření jako respondenti.

7.1 Vzorek žáků zapojených do pilotního nasazení

Do ověřování navrhovaného modelu výuky bylo zapojeno celkem jedenáct žáků (10 chlapců a 1 dívka) ze šestých a sedmých tříd, kteří se přihlásili do zájmového útvaru nesoucího název *„Programování na základce“*. Vznikla tak malá heterogenní skupina žáků ve věku 11 až 13 let se značně rozdílnými schopnostmi, dovednostmi, rozdílným pracovním tempem a úrovní inteligence.

7.2 Proces ověření

Výuka probíhala podle navrženého sylabu (Kap. 6.2), konkrétněji podle časově-tematického plánu (Příloha 9) v učebně ICT na základní škole (Obrázek 3).



Obrázek 3: Plán výuky

7.2.1 Úroveň obtížnosti

V rámci ověřovacího procesu modelu výuky byla průběžně vyhodnocována úspěšnost žáků a obtížnost jednotlivých aktivit obsažených ve výukových materiálech.

Slovní vyjádření obtížnosti bylo určováno v závislosti na výsledném aritmetickém průměru procentuální úspěšnosti (Tabulka 23) vzorku žáků, kterého dosahoval v rámci tvůrčích aktivit.

Úspěšnost [%]	100–91	90–76	75–51	50–26	25–0
Navrhovaná obtížnost	velmi snadná	snadná	normální	obtížná	velmi obtížná

Tabulka 23: Návrh slovního hodnocení obtížnosti aktivity

Princip stanovení obtížnosti unplugged aktivit

Procentuální úspěšnost byla určena vztahem:

$$úspěšnost = \frac{k_s}{k_c} \cdot 100\,,$$

kde k_s je počet správných kroků vedoucích k úspěšné konstrukci a k_c je celkový počet kroků v zápisu postupu konstrukce.

Aritmetický průměr úspěšnosti byl vypočítán podle vzorce

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i\,.$$

Úspěšnost v unplugged aktivitách může být ovlivněna vnějším faktorem závislým na výběru složitosti konstrukce. Žáci byli na tuto skutečnost předem upozorněni, a dále na ni nebyl brán ohled. Druhým faktorem ovlivňujícím výsledek mohou být schopnosti členů ověřovacích skupin. Proto byly zápisy postupu revidovány také učitelem.

Princip stanovení obtížnosti aktivit v Lightbot a Lightbot 2

Obtížnost hry Lightbot a Lightbot 2 byla hodnocena podle průměrné úspěšnosti skupin v jednotlivých částech hry. V případě, že nebylo v silách skupiny úroveň dokončit, byla jí poskytnuta nápověda od učitele. Za každou nápovědu bylo skupině odečteno 10 % z celkové 100% úspěšnosti. Obtížnost poslední aktivity ve hře Lightbot 2 (tvorba vlastní úrovně bludiště) byla hodnocena slovně samotnými žáky.

Princip stanovení obtížnosti vybraného kurzu Code.org

Obtížnost vybraných lekcí zařazených do kurzu byla hodnocena podle aritmetického průměru procentuální úspěšnosti vzorku žáků vycházející z progresu skupin v povinných úrovních každé z lekcí kurzu. Každý samostatně splněný úkol byl hodnocen 100 %. Úkol splněný za pomoci učitele nebo úkol dokončený s nadbytečným množstvím použitých bloků (překročným limitem) byl hodnocen 50 %. Pokud byl úkol vynechán nebo nedokončen, byl hodnocen 0 %.

Princip stanovení obtížnosti tvůrčích aktivit ve Scratchi

Obtížnost tvůrčích aktivit třetího bloku byla hodnocena žáky v reflexi zařazené v závěru každé lekce. Žáci hodnotili obtížnost každé ze sedmi zadaných lekcí přiřazením hodnoty v intervalu $\langle 0; 6 \rangle$, přičemž slovní ekvivalent hodnot je následující (Tabulka 24).

0	1	2	3	4	5	6
velmi snadné	snadné	jednodušší	normální	trochu obtížnější	obtížné	velmi obtížné
program jsme vytvořili bez cizí pomoci		program jsme vytvořili s cizí pomocí		vytvořili jsme částečně funkční program		program byl nefunkční

Tabulka 24: Slovní ekvivalent přiřazovaných číselných hodnot

Princip stanovení obtížnosti tvorby vývojových diagramů

Obtížnost tvorby diagramů ve čtvrtém bloku byla určena podle správnosti návrhů vývojových diagramů. Za každou chybu ve vývojovém diagramu bylo odečteno 10 % z celkových 100 %.

7.2.2 Úvodní hodina – první blok výuky

V první části úvodní hodiny žáci odpovídali na čtyři otázky zadané s cílem vytvořit si představu, jak jsou žáci motivováni a jaké mají povědomí o problematice programování:

- *„Proč ses přihlásil(a) na tento kroužek?“*
- *„Co si představíš pod pojmem program?“*
- *„Co si představíš pod pojmem programování?“*
- *„Už jsi někdy programoval(a)?“*

Odpovědi dvou žáků na první otázku byly překvapivé:

Odpověď 1: *„Abych se naučil stříhat videa na youtube.“*

Odpověď 2: *„Protože se doma nudím.“*

Naopak, další dva žáci měli jasnou představu o tom, co je to program i programování. Přihlásili se proto, aby byli schopni společně naprogramovat strategickou hru, jejíž scénář již společně vytvořili.

V další části hodiny byla žákům představena webová stránka *„Programování na základce“* obsahující vytvořené výukové materiály. Při zběžném průchodu webem byli žáci seznámeni s plánem výuky, jednotlivými bloky a s principem odměn za odvedenou práci.

Následně byly žákům představeny pojmy algoritmus a základní vlastnosti algoritmů, program, programování a programovací jazyk. Důraz byl kladen především na pochopení souvislosti mezi algoritmem a programem.

V závěru hodiny byla provedena kontrola pochopení učiva formou testu prostřednictvím aplikace Plickers. Okamžitě po ukončení testu byli žáci seznámeni se správnými odpověďmi. Úspěšní žáci vysvětlili správnost odpovědi spolužákům, kteří chybovali.

Úspěšnost v testu byla následující (Obrázek 4):

[redacted]	71 %	C	B	D	B	B	C	B
[redacted]	86 %	B	D	D	B	B	A	B
[redacted]		-	-	-	-	-	-	-
[redacted]	86 %	C	D	D	B	B	D	B
[redacted]	86 %	B	D	D	B	B	A	B
[redacted]	86 %	B	D	D	B	B	A	B
[redacted]	86 %	C	C	D	B	B	A	B
[redacted]	100 %	C	D	D	B	B	A	B
[redacted]	86 %	B	D	D	B	B	A	B
[redacted]	86 %	A	D	D	B	B	A	B
[redacted]	71 %	C	B	D	B	B	B	B
[redacted]	100 %	C	D	D	B	B	A	B

Obrázek 4: Úspěšnost žáků v testu (PrintScreen <https://www.plickers.com/scoresheet>)

Všichni žáci v testu odpověděli na více než 70 % otázek správně.

Shrnutí výsledků úvodní hodiny

Žáci ve skupině mají velmi rozdílné vstupní znalosti a zkušenosti s tvorbou kódu. Žáci jsou různě motivovaní. Na základě zjištění došlo u některých aktivit k rozšíření o alternativní možnosti (jednodušší a obtížnější varianty).

7.2.3 Druhý blok výuky – unplugged aktivity

Unplugged aktivity byly zařazeny především proto, aby žáci při jejich plnění pochopili důležitost přesné, jednoduché a jednoznačné formulace instrukcí.

První aktivita

První aktivita druhého bloku – origami je orientovaná na slovní nebo symbolický zápis postupu skládání papíru, které vede do vybraného tvaru. Žáci byli v úvodu seznámeni s náročností úkolu a s vhodností výběru lehké skládačky. Pracovali ve dvojicích a jedné trojici.

Následující tabulka představuje, zda se dvojici podařilo postup popsat dostatečně jasně (Tabulka 25).

Označení skupiny	SK1	SK2	SK3	SK4	SK5
Hotový postup	ANO	ANO	ANO	ANO	ANO
Složitost origami [počet kroků]	10	8	12	8	12
Ověření skupinou s úspěšností [%]	S2 60	S3 100	S4 41,6	S5 100	S1 75
Opětovné ověření skupinou s úspěšností [%]	S3 40	S4 100	S5 25	S1 100	S2 83,3
Kontrola řešení – úspěšnost učitele [%]	60	100	41,6	100	83,3

Tabulka 25: Úspěšnost skupin (1. unplugged aktivita)

Aktivita vycházela z úlohy informatické soutěže Bobřík informatiky nazvané *„Folding paper“* [cit. 2018-11-08].

Shrnutí výsledků první unplugged aktivity

V rámci aktivity si žáci poprvé vyzkoušeli důležitost správné a jednoznačné formulace instrukcí. Pouze dvěma skupinám se podařilo přesně popsat postup, podle kterého bylo možné origami složit. Skupiny SK1, SK3 měly v zápisu postupu chyby a nejednoznačný popis následných kroků. Skupina SK5 v závěru postupu udělala menší chybu (nejednoznačnost dvou finálních kroků). Skupina SK2 byla přiřazena do SK3, skupina SK4 do SK1, ve kterých plnily funkci poradců. Žáci byli

po prvním neúspěchu opět upozorněni na fakt, že ne vždy (spíše naopak) se podaří vytvořit zcela funkční program hned napoprvé. Po opravě vedly všechny postupy k úspěšnému složení zvoleného origami. Na pozici poradců (při hledání chyb a opravách postupů neúspěšných skupin) výborně pracovali žáci skupin SK2 a SK4.

Úspěšnost vzorku v první části činila 72,49 %. Obtížnost aktivity byla vyhodnocena jako **normální**.

Druhá aktivita

Druhá unplugged aktivita se opět zaměřovala na správný zápis postupu, tentokrát pomocí grafických instrukcí, respektive na tvorbu obrázků pomocí číselných řad. Žáci pracovali ve skupinách.

Skupiny SK2 a SK4 vytvářely rastrové obrázky v rastrové mřížce (20×20 px). Postup kódovaly do číselných řad. Aktivitu představuje Bell, Witten a Fellows (2015, s. 16–25) pod názvem *„Colour by numbers.“*

Ostatní skupiny pracovaly podle zadání dostupného na webové stránce. K tvorbě kódu měli žáci k dispozici pouze omezený počet grafických instrukcí (Obrázek 5).

Obrázek 5: Povolené grafické instrukce

Aktivita vycházela z první lekce kurzu dostupného na webu organizace Code.org pod názvem *„Graph Paper Programming“* [cit. 2018-11-08].

Shrnutí výsledků druhé unplugged aktivity

V průběhu práce se u skupin SK1, SK3 a SK5 objevil problém při přechodu na nový řádek. Na základě návrhů žáků bylo stanoveno pravidlo střídání směrů při tvorbě obrázku (sudé řádky ve směru doleva, liché doprava). Všem skupinám se podařilo postup správně zapsat a také správně interpretovat postup spolužáků.

Úspěšnost žáků byla 100%. Aktivita byla hodnocena jako **velmi snadná**.

Třetí unplugged aktivita

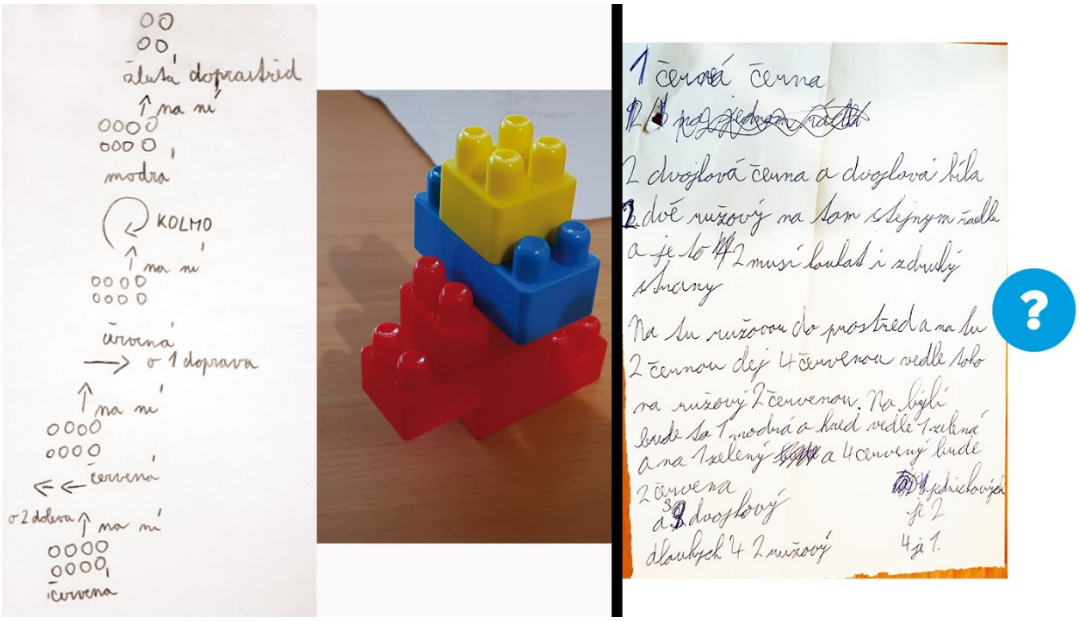
Třetí z unplugged aktivit se jako poslední opět zaměřovala na přesný a jednoznačný zápis postupu. Žáci pracovali ve dvojicích. Skupiny stavěly konstrukce z LEGO® kostek a zároveň zapisovaly postup vedoucí k finální konstrukci. Přestože byli žáci upozorněni, aby tvořili konstrukce maximálně z 5 kostek (byl připomenut problém první aktivity), většina skupin tvořila z většího počtu, čímž si již vědomě ztížila úlohu. Výsledky úspěšnosti korektního zápisu postupu je představena v následující tabulce (Tabulka 26).

Označení skupiny	SK1	SK2	SK3	SK4	SK5
Hotový postup	ANO	ANO	ANO	ANO	ANO
Složitost konstrukce [počet kostek]	8	5	10	7	7
Ověření skupinou s úspěšností [%]	S2 50	S3 100	S4 0	S5 28,6	S1 42,9
Opětovné ověření skupinou s úspěšností [%]	S3 37,5	S4 100	S5 0	S1 42,9	S2 42,9
Kontrola řešení – úspěšnost učitele [%]	50	100	0	42,9	42,9

Tabulka 26: Úspěšnost skupin (3. unplugged aktivita)

Shrnutí výsledků třetí unplugged aktivity

Obrázek níže představuje rozdílný přístup žáků k zápisu postupu (Obrázek 6). Zápis postupu v levé části patří skupině SK2, zápis v pravé části patří skupině SK3.



Obrázek 6: Zápis postupu dvou skupin

Aktivita vycházela z cvičení zařazeného do publikace organizace Code.org (2018).

Úspěšnost žáků činila 44,48 %. Aktivita je hodnocena jako **obtížná**.

Čtvrtá unplugged aktivita

Čtvrtá aktivita je zaměřena na hledání pravidel, podle kterých je možné jednotlivé objekty třídit. Žáci pracovali samostatně nebo ve dvojicích. S touto aktivitou neměli žáci žádné potíže. Úspěšnost žáků byla 100%. Aktivita je hodnocena jako **velmi snadná**.

7.2.4 Druhý blok výuky – Lightbot

Hry Lightbot a Lightbot 2 byly do modelu zařazeny především z důvodu rozvíjení informatického myšlení.

První aktivita Lightbot

Žáci procházeli všemi dvaceti úrovněmi hry Lightbot. Aktivity se zúčastnilo celkem devět žáků, kteří pracovali ve dvojicích (SK1–SK4). Jeden žák pracoval samostatně (J1). Progres žáků ve všech částech hry Lightbot je uveden v tabulce níže (Tabulka 27).

Úrovně		Úspěšnost skupin [%]				
Část	Číslo úrovně	SK1	SK2	SK3	SK4	J1
Část BASICS	1	100	100	100	100	100
	2	100	100	100	100	100
	3	100	100	100	100	100
	4	100	90	100	100	100
	5	100	100	100	100	100
	6	100	100	100	100	100
	7	100	100	100	100	100
	8	100	100	100	100	100
Část PROCEDURES	1	100	100	100	100	100
	2	90	90	100	100	100
	3	100	100	100	100	100
	4	90	90	100	100	100
	5	80	70	100	100	100
	6	80	50	90	100	90
Část LOOPS	1	100	100	100	100	100
	2	100	100	100	100	100
	3	100	100	100	100	100
	4	90	90	100	100	100
	5	80	70	80	100	100
	6	70	70	90	100	90
Průměrná úspěšnost skupin		94	91	98	100	99

Tabulka 27: Progres v první aktivitě Lightbot

Shrnutí výsledků první aktivity Lightbot

Úspěšnost vzorku byla více než 90%. Jisté problémy s dokončením bylo možné sledovat především v závěrečných úrovních druhé a třetí části hry. Aktivitu lze hodnotit jako **velmi snadnou**.

Druhá aktivita Lightbot 2

Žáci měli za úkol projít nejméně čtyřmi úrovněmi částí *„Conditionals“* a *„Expert“* hry Lightbot 2. Aktivity se zúčastnilo 8 žáků pracujících ve dvojicích (SK1–SK4). Progres žáků v zadaných úrovních hry (dvě skupiny zvládly také další úrovně) je uveden v následující tabulce (Tabulka 28).

Úrovně		Úspěšnost skupin [%]			
Část	Číslo úrovně	SK1	SK2	SK3	SK4
Část CONDITIONALS	1	100	100	80	90
	2	100	90	40	90
	3	70	0	50	60
	4	100	80	20	30
Část EXPERT	1	60	60	30	60
	2	100	80	30	100
	3	100	60	20	80
	4	90	60	30	80
Průměrná úspěšnost skupin		90	66,25	37,5	73,75
Obtížnost aktivity		snadná	normální	obtížná	normální

Tabulka 28: Progres v druhé aktivitě Lightbot 2

Shrnutí výsledků druhé aktivity Lightbot 2

Skupiny se potýkaly s problémy téměř ve všech úrovních. Podle slov žáků jim největší problém činily podmínky realizované obarvením instrukcí.

Celková úspěšnost vzorku v rámci druhé aktivity činila 66,88 %. Obtížnost aktivity byla stanovena jako **normální**.

Třetí aktivita

Třetí aktivita cílila především na vlastní experimentování žáků v prostředí hry Lightbot 2. Aktivity se zúčastnilo celkem 11 žáků rozdělených do dvojic (SK1–SK4) a jedné trojice (SK5). Výsledky, kterých žáci, dosáhli, jsou následující (Tabulka 29).

Označení skupiny	SK1	SK2	SK3	SK4	SK5
Obsahuje jednu rozhodovací podmínku?	ANO	ANO	ANO	ANO	ANO
Obsahuje volání jedné procedury?	ANO	ANO	ANO	ANO	ANO
Obsahuje přesný počet míst pro instrukce?	NE	NE	ANO	NE	ANO
Je úroveň hratelná?	ANO	ANO	ANO	ANO	ANO
Ověřeno skupinou	SK2	SK3	SK4	SK5	SK1
Žáky navržená obtížnost aktivity	normální	normální	velmi snadná	snadná	velmi snadná

Tabulka 29: Výsledky třetí aktivity Lightbot 2

Obtížnost aktivity byla na základě žákovských reakcí vyhodnocena jako **snadná**.

7.2.5 Druhý blok výuky – Code.org kurz

Žáci spolupracovali ve dvojicích, které byly nastaveny v prostředí kurzu Code.org. Každé lekci byla věnována jedna hodina.

V tabulkách níže (Tabulka 30 až Tabulka 34) je graficky zobrazen progres skupin v jednotlivých lekcích. Tmavá barva (100% úspěšnost) symbolizuje samostatně splněný úkol. Světlá barva (50% úspěšnost) symbolizuje úkol, k jehož splnění bylo skupině napovězeno nebo byl při tvorbě kódu překročen limit povolených bloků. Bez barvy (0% úspěšnost) jsou ty úkoly, které nebyly vyřešeny. Pro každou lekci byla vypočítána průměrná úspěšnost v rámci celého vzorku (všech skupin).

Lekce 3: Umělec													
	Úkol č.	1	2	3	4	5	6	7	8	9	10	11	12
Skupina	SK1												
	SK2												
	SK3												
	SK4												
	SK5												
	J1												

Tabulka 30: Progres skupin ve 3. lekci kurzu Code.org

Celková procentuální úspěšnost žáků ve 3. lekci kurzu činila 78,5 %. Náročnost lekce zaměřené na využívání sekvencí a cyklů byla na základě výsledné úspěšnosti žáků vyhodnocena jako **snadná**.

Lekce 6: Umělec – proměnné														
	Úkol č.	1	2	3	4	5	6	7	8	9	10	11	12	13
Skupina	SK1													
	SK2													
	SK3													
	SK4													
	SK5													
	J1													

Tabulka 31: Progres skupin v 6. lekci kurzu Code.org

Celková procentuální úspěšnost žáků v 6. lekci kurzu činila 83,3 %. Náročnost lekce zaměřené na práci s proměnnou byla na základě výsledné úspěšnosti žáků vyhodnocena jako **snadná**.

Lekce 10: Umělec – for cykly											
	Úkol č.	1	2	3	4	5	6	7	8	9	10
Skupina	SK1										
	SK2										
	SK3										
	SK4										
	SK5										
	J1										

Tabulka 32: Progres skupin v 10. lekci kurzu Code.org

Celková procentuální úspěšnost žáků v 10. lekci kurzu činila 63,3 %. Náročnost lekce zaměřené na cykly s pevným počtem opakování byla na základě výsledné úspěšnosti žáků vyhodnocena jako **normální**.

Lekce 12: Malíř – funkce												
	Úkol č.	1	2	3	4	5	6	7	8	9	10	11
Skupina	SK1											
	SK2											
	SK3											
	SK4											
	SK5											
	J1											

Tabulka 33: Progres skupin ve 12. lekci kurzu Code.org

Celková procentuální úspěšnost žáků ve 12. lekci kurzu činila 81,1 %. Náročnost lekce zaměřené na práci s funkcemi byla na základě výsledné úspěšnosti žáků vyhodnocena jako **snadná**.

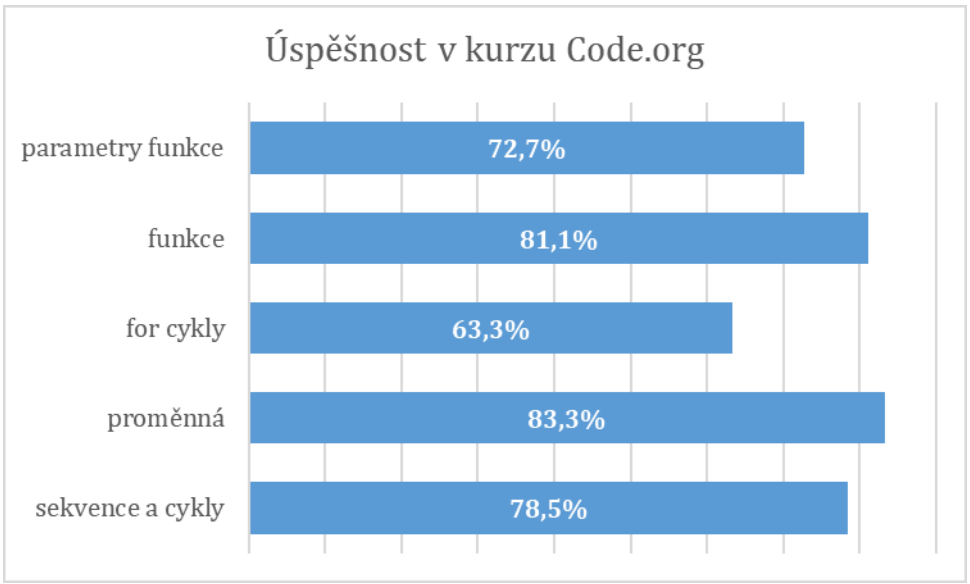
Lekce 14: Umělec – funkce s parametry																
	Úkol č.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Skupina	SK1															
	SK2															
	SK3															
	SK4															
	J1															

Tabulka 34: Progres skupin ve 14. lekci kurzu Code.org

Celková procentuální úspěšnost žáků ve 14. lekci kurzu činila 72,7 %. Náročnost lekce zaměřené na práci s parametry funkce byla na základě výsledné úspěšnosti žáků vyhodnocena jako **normální**.

Shrnutí výsledků kurzu Code.org

Kurz byl zařazen především z důvodu seznámení žáků s block-based programováním, s řídicími strukturami, funkcemi a proměnnými. Největší problém činila žákům lekce 10 zaměřená na cykly s přesným počtem opakování (63,3% úspěšnost). Naopak, nejméně problematické bylo pro žáky využívání proměnných v programech (83,3% úspěšnost). Úspěšnost žáků v jednotlivých lekcích kurzu Code.org je představena v grafu níže (Graf 9).



Graf 9: Úspěšnost vzorku žáků v rámci kurzu Code.org

Celkově nejproblematičtější (20% úspěšnost) se jevil úkol číslo 15 v poslední lekci zaměřené na parametry funkce. Ani jedna z dvojic jej nedokázala vyřešit bez nápovědy. Druhá nejnižší úspěšnost (25 %) byla zaznamenána v úkolu číslo 9 v 10. lekci kurzu. Tuto úlohu vyřešila jedna dvojice bez nápovědy a jedna s nápovědou. Pro nejúspěšnější skupinu (SK1) byla nejproblematičtější lekce zaměřená na parametry funkcí.

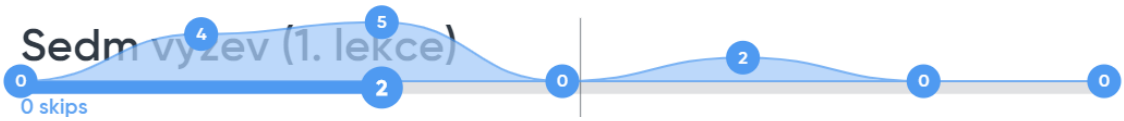
7.2.6 Třetí blok výuky – Scratch

Žáci hodnotili obtížnost jednotlivých lekcí slovně a s využitím aplikace Mentimeter <https://www.mentimeter.com>. Obtížnost byla žáky hodnocena vždy v závěru každé lekce. Navrhovaná obtížnost jednotlivých lekcí je tak do značné míry ovlivněna subjektivními pocity žáků. Do hodnocení obtížnosti nebyl zařazen závěrečný projekt. Žáci v průběhu třetího bloku pracovali ve skupinách po dvojicích a v jedné trojici. Žáci byli vedeni k samostatnosti. V případě výskytu problému s konstrukcí programu byli žáci vždy nejprve odkázáni na nápovědu ve Scratchi. Napovězeno bylo žákům pouze při přetrvávajících problémech s nefunkčním kódem. Z tohoto důvodu byly do nulté lekce zařazeny, kromě založení účtu, především aktivity podporující samostatnou činnost, například: *„prostudujte prostředí“, „najděte bloky“.*

První lekce třetího bloku

První lekce žáky blíže seznamovala s tvorbou kódu pomocí bloků ve Scratchi. Podle slov žáků nebyla tvorba kódu v novém prostředí příliš problematická: *„Tvoří se to skoro stejně jako v tom kurzu s malířem. Neměli jsme s tím moc problém, jenom jsme si museli hlídat, kterou postavičku máme vybranou.“* Většina skupin (80 %) dokázala vyřešit všech sedm výzev, dvě skupiny zcela samostatně.

Navrhovaná obtížnost lekce (Obrázek 7):



Obrázek 7: Obtížnost první lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

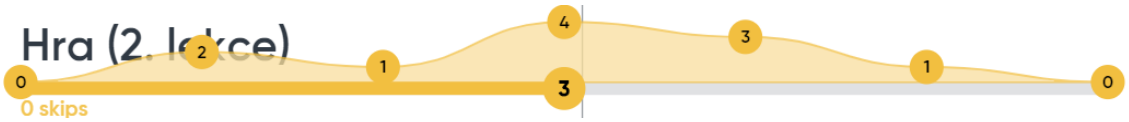
Čtyři žáci (36 %) hodnotili první lekci jako snadnou. Tvorbu všech sedmi programů dokázali realizovat bez cizí pomoci. Pět žáků (46 %) hodnotilo lekci jako jednodušší. Při tvorbě kódu zřídka požádali o pomoc učitele. Dva žáci (18 %) hodnotili první

lekci jako trochu obtížnější. Žákům se nepovedlo vytvořit poslední dva programy. S průměrem 2,0 byla obtížnost lekce vyhodnocena jako **jednodušší**.

Druhá lekce třetího bloku

Druhá lekce byla složena ze čtyř částí (tří povinných, jedné dobrovolné), které žáky postupně vedly k vytvoření jednoduché funkční hry. Jedna skupina (20 %) dokázala vytvořit funkční hru zcela bez pomoci. Dvě skupiny (40 %) vytvořily hru s pomocí nápověd učitele. Dvěma skupinám (40 %) se podařilo vytvořit částečně funkční hru.

Navrhovaná obtížnost lekce (Obrázek 8):



Obrázek 8: Obtížnost druhé lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

Na hodnocení obtížnosti lekce se projevoval subjektivní dojem žáků, proto bylo možné v rámci jedné skupiny spatřovat rozdílná hodnocení obtížnosti. Dva žáci (18 %) hodnotili lekci jako snadnou, jeden žák (9 %) ji hodnotil jako jednodušší. Pro čtyři žáky (36 %) měla lekce normální obtížnost. Tři žáci (27 %) ji hodnotili jako trochu obtížnější, jeden žák (9 %) jako obtížnou. S průměrem 3,0 byla obtížnost lekce vyhodnocena jako **normální**.

Třetí lekce třetího bloku

Ve třetí lekci žáci pracovali na tvorbě soutěží, při které využívali funkci náhodného čísla (Math.random). Jednalo se o aktivitu s mezipředmětovým přesahem do matematiky. Jedné skupině (20 %) se podařilo vytvořit obě soutěže, přičemž se pokusila o bonusové modifikace programů (u první soutěže úspěšně). Dvě skupiny (40 %) vytvořily funkční soutěže s pomocí nápověd učitele. Dvěma skupinám (40 %) se podařilo vytvořit první plně funkční a druhou částečně funkční soutěž.

Navrhovaná obtížnost lekce (Obrázek 9):

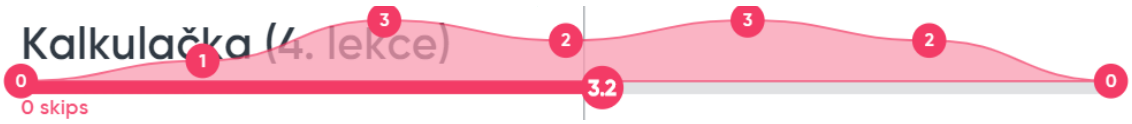
Obrázek 9: Obtížnost třetí lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

Nejčastěji (45 %) byla obtížnost hodnocena jako jednodušší. Jeden žák (9 %) lekci vyhodnotil jako velmi obtížnou. S průměrem 2,9 byla obtížnost lekce vyhodnocena jako **normální**.

Čtvrtá lekce třetího bloku

Čtvrtá lekce byla zaměřena na využití vnořených cyklů, událostí, proměnných a operátorů při tvorbě funkční kalkulačky. Lekce byla pro žáky náročnější než předchozí, neboť v rámci tvorby neměli předpřipravené žádné bloky ani objekty. Přesto se třem skupinám podařilo vytvořit funkční kalkulačky. Při práci žáci mohli nahlížet do jiných projektů a čerpat z nich inspiraci.

Navrhovaná obtížnost lekce (Obrázek 10):



Obrázek 10: Obtížnost čtvrté lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

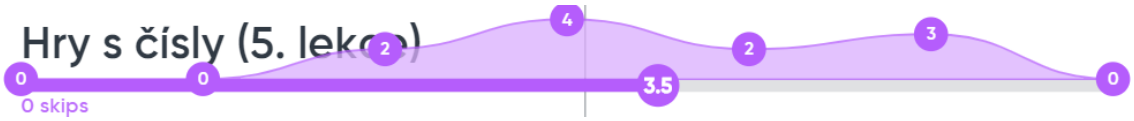
Dvěma skupinám (40 %) se nepodařilo sestavit zcela funkční program (navržené kalkulačky pouze sčítaly nebo odčítaly). Obě skupiny stavěly program s využitím vnořených cyklů, což se jim nepodařilo. Naopak dvě skupiny (40 %) dokázaly (kromě vytvoření funkčních kalkulaček) ošetřit problematiku nuly v děliteli.

Náročnost lekce se také odrazila ve výsledném hodnocení. S průměrem 3,2 byla obtížnost lekce vyhodnocena jako **normální**.

Pátá lekce třetího bloku

V páté lekci se objevil problém s obtížností jedné z aktivit. Jednalo se aktivitu *„Odpočet“*, kterou bez návodných rad učitele nebyla schopna vyřešit žádná ze skupin. Tato aktivita byla prozatím vynechána. Aktivita *„Sudá – lichá“* obsahovala dvě různě náročné varianty. Dvě slabší skupiny (40%) pracovaly na jednodušší variantě, kterou obě skupiny zvládly dovést do zcela funkčního programu. Ve zbylém čase se věnovaly opravě kódu kalkulaček. Zbylé tři skupiny (60 %) zvládly vytvořit obě varianty. Následně se pokoušely o zvládnutí náročnější aktivity *„Math Master“*. Dvě skupiny byly v tomto počinu úspěšné.

Navrhovaná obtížnost lekce (Obrázek 11):



Obrázek 11: Obtížnost páté lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

Přestože se žákům podařilo téměř samostatně sestavit své programy, hodnotili lekci následovně: dva žáci jako jednodušší, čtyři žáci jako normální, dva žáci jako trochu obtížnější a tři žáci jako obtížnou. Svůj podíl na hodnocení měla jistě aktivita *„Odpočet“*, kterou jeden z žáků popsal slovy: *„Tohle se fakt nedá.“* S průměrem 3,5 byla obtížnost lekce vyhodnocena jako **trochu obtížnější**.

Šestá lekce třetího bloku

Šestá lekce byla zaměřena na volání funkcí s parametry. Lekce nebyla pro žáky náročná, a to (podle slov žáků) především z důvodu absolvování Code.org kurzu. Čtyři skupiny (80 %) byly schopny vytvořit funkční program, ve kterém využily volání funkce s parametry podle zadání. Pátá skupina (20 %) vytvořila částečně funkční program, ve kterém byla volána funkce se vstupními parametry tloušťka pera a délka strany. Skupině byla poskytnuta rada učitele o vztahu mezi počtem stran a vnitřním úhlem. Zbylé čtyři skupiny následně upravovaly kód problematické aktivity *„Odpočet“*, kterou vyřešily dvě skupiny bez nápovědy, dvě s nápovědou. Žádné skupině se nepodařilo vytvořit plně funkční dobrovolný projekt *„Vykreslení n-úhelníků“*.

Žáky navrhovaná obtížnost lekce (Obrázek 12):

Obrázek 12: Obtížnost šesté lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

Pět žáků hodnotilo obtížnost lekce jako jednodušší, dva žáci určili obtížnost jako normální, dva jako trochu obtížnější. Jeden žák hodnotil obtížnost lekce jako snadnou. Jeden žák ji vyhodnotil jako obtížnou. S průměrem 2,7 byla obtížnost lekce vyhodnocena jako **normální**.

Žáci, kteří se pokusili o sestavení funkčního programu v projektu *„Vykreslení n-úhelníků“*, hodnotili tuto aktivitu jako **příliš obtížnou**.

Sedmá lekce třetího bloku

Sedmá lekce se skládala ze šesti prvků (projektů), které by měly postupně vést k vytvoření funkční hry. Problematický byl především pátý prvek *„enemáci, kolize a konec hry“*. Tuto část musel s žáky realizovat učitel. Nejprve byl žákům představen funkční kód, který četli, vysvětlovali a porovnávali se zadáním. Následně se kód ve skupinách pokusili sestavit. Třem skupinám (60 %) se podařilo vytvořit plně funkční hru.

Navrhovaná obtížnost lekce (Obrázek 13):

Obrázek 13: Obtížnost sedmé lekce podle žáků (PrintScreen <https://www.mentimeter.com>)

Šest žáků hodnotilo obtížnost této lekce jako obtížnou, dva jako velmi obtížnou a dva žáci jako trochu obtížnější. Jeden žák hodnotil obtížnost jako jednodušší. S průměrem 4,7 byla obtížnost lekce vyhodnocena jako **obtížná**.

Osmá lekce třetího bloku

V osmé lekci žáci ve skupinách aplikovali získané znalosti a dovednosti ve Scratchi. Žáci byli omezeni pouze časem (120 minut). Svůj výsledný projekt skupiny nechaly zanalyzovat v online nástroji Dr. Scratch na <http://www.drscratch.org> vytvořeném právě za účelem analýzy a hodnocení Scratch projektů. Nástroj analyzuje kód v sedmi úrovních, které jsou hodnoceny body z intervalu ⟨0; 3⟩:

- **Algoritmizace** (Flow Control) – využití cyklů, vnořených cyklů aj.
- **Reprezentace dat** (Data reprezentation) – využití proměnných, seznamů aj.
- **Abstrakce** (Abstraction) – využití klonů, funkcí aj.
- **Interaktivita** (User interactivity) – využití prvků vnímání, např. otázka, přepni video aj.
- **Synchronizace** (Synchronization) – využití synchronizačních prvků, např. čekej
- **Paralelní procesy** (Parallelism) – využití událostí, např. po obdržení zprávy aj.
- **Logika** (Logic) – využití logických operátorů, podmínek aj.

Body jsou následně sečteny a projekt ohodnocen:

- 0–7 bodů z 21 bodů: **basic**;
- 8–14 bodů z 21 bodů: **developing**;
- 15–21 bodů z 21 bodů: **master**.

Projekty skupin byly ohodnoceny následovně (Tabulka 35).

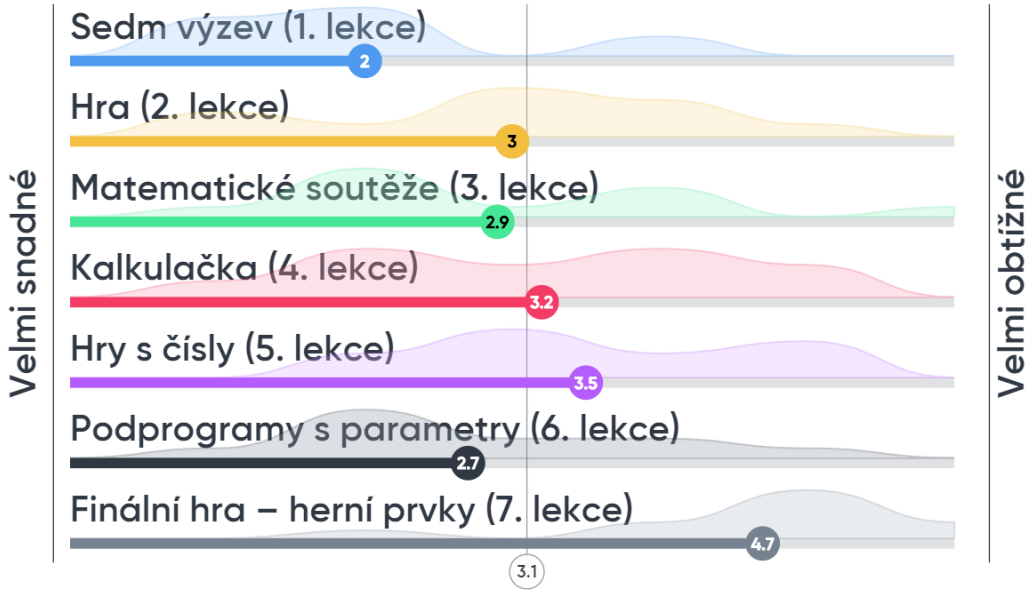
Skupina	Algoritmizace	Reprezentace dat	Abstrakce	Interaktivita	Synchronizace	Paralelní procesy	Logika	Celkem bodů	Hodnocení
SK1	2	2	3	2	2	3	3	17	**Master**
SK2	2	2	3	2	2	3	2	16	**Master**
SK3	1	0	1	1	2	2	0	7	**Basic**
SK4	3	1	2	2	1	2	1	12	**Developing**
SK5	2	1	1	2	1	2	1	10	**Developing**
Σ	10	6	11	9	8	11	6	-	-

Tabulka 35: Výsledky analýzy vlastních projektů

Výsledky analýzy mohou podstatně přiblížit skutečnost, jak jsou žáci v tvorbě kódů ve Scratchi zdatní. Analýza také upozorňuje na případné chyby či duplicitu v kódu. Je však vhodné počítat s určitým zkreslením výsledků v závislosti na charakteru projektu. Například v interaktivním příběhu (SK3) budou spíše využívány prvky synchronizace, interaktivity a paralelních procesů, než proměnné a funkce s parametry.

Shrnutí výsledků třetího bloku Scratch

Výsledné hodnocení obtížnosti jednotlivých lekcí žáky je představeno na obrázku níže (Obrázek 14).



Obrázek 14: Hodnocení obtížnosti Scratch lekcí žáky (PrintScreen <https://www.mentimeter.com>)

Žáky navržená obtížnost třetího bloku byla celkově vyhodnocena jako **normální** (aritmetický průměr činil 3,1). Žáci v průběhu plnění lekcí identifikovali jako příliš náročné projekty *„Odpočet“* (pátá lekce), *„Vykreslení n-úhelníků“* (šestá lekce) a *„pátý prvek – enemáci, kolize a konec hry“* (sedmá lekce). Žádná z aktivit tohoto bloku nebyla žáky vyhodnocena jako příliš lehká.

7.2.7 Čtvrtý blok výuky – vývojové diagramy

Vývojové diagramy byly do modelu výuky začleněny především z toho důvodu, aby byli žáci seznámeni s reálně využívaným prostředkem, který je možné aplikovat při návrzích algoritmů a programových struktur.

První aktivita – Čteme a tvoříme diagramy

První aktivita se zaměřovala na čtení a tvorbu jednoduššího vývojového diagramu. Žáci pracovali ve dvojicích, jeden žák pak samostatně. Všichni žáci dokázali oba diagramy přečíst i určit účel. Procentuální úspěšnost této aktivity činila 100 % a aktivita byla podle úspěšnosti žáků vyhodnocena jako **velmi snadná**, což potvrdili slovně i samotní žáci.

V další části první aktivity měli žáci sestavit vývojový diagram, který by představoval algoritmus řešící zadaný problém. Skupiny měly na výběr mezi lehčí a těžší variantou. Za chybu se nepovažovala záměna symbolu vstupu/výstupu s příkazem.

Dvě skupiny si vybraly těžší variantu, ostatní si vybraly lehčí variantu. Vypracované vývojové diagramy obsahovaly (kromě skupiny SK2) funkční chyby. Žáci byli na chyby upozorněni a byla jim umožněna oprava chyb. Výsledky aktivity jsou představeny v následující tabulce (Tabulka 36).

Varianta	Těžší varianta (pokoj)		Lehčí varianta (čištění zubů)			
Označení skupiny	SK1	SK2	SK3	SK4	SK5	J1
Bylo realizováno?	ANO	ANO	ANO	NE	ANO	ANO
Počet chyb v 1. pokusu	2	1	3	-	2	3
Úspěšnost 1. pokusu [%]	80	90	70	0	80	70
Úspěšnost 2. pokusu [%]	100	100	90	90	90	90
Obtížnost podle žáků	snadná	velmi snadná	snadná	normální	velmi snadná	snadná

Tabulka 36: Výsledky 1. aktivity - Čteme a tvoříme diagramy

Úspěšnost žáků činila 65 %, po opravě chyb v diagramech pak 93,3 %. Obtížnost aktivity byla vyhodnocena jako **snadná**.

Druhá aktivita – Úprava

Úkolem žáků bylo nalézt a opravit chyby v hotových vývojových diagramech. Žáci absolvovali povinně pouze první dva (snazší) příklady. Výsledky aktivity jsou představeny v tabulce níže (Tabulka 37).

Označení skupiny		SK1	SK2	SK3	SK4	SK5	J1
1. příklad							
Úprava diagramu	Opraven výstup	ANO	ANO	ANO	ANO	ANO	ANO
	Výpočet obvodu	ANO	ANO	ANO	ANO	ANO	ANO
	Výstup	ANO	ANO	ANO	NE	ANO	NE
Úspěšnost [%]		100	100	100	70	100	70
Obtížnost podle žáků		velmi snadná	velmi snadná	velmi snadná	normální	velmi snadná	normální

2. příklad							
Úprava diagramu	Přidána podmínka	ANO	ANO	ANO	ANO	NE	NE
	Upraven výstup	NE	ANO	ANO	NE	NE	NE
Úspěšnost [%]		50	100	100	50	0	0
Obtížnost podle žáků		normální	velmi snadná	snadná	normální	velmi obtížná	velmi obtížná

Tabulka 37: Výsledky 2. aktivity – úprava diagramů

Při úpravě prvního diagramu byli žáci celkově úspěšnější (90 %), při úpravě druhého diagramu dosáhli přesně 50% úspěšnosti. Obtížnost byla vyhodnocena jako **normální**.

Třetí aktivita – tvorba

Žáci měli za úkol vytvořit vývojový diagram, který by představoval algoritmus na porovnávání čísel. Žáci pracovali v pěti skupinách (čtyři dvojice, jedna trojice). Jedna skupina si vybrala obtížnější variantu – porovnání tří čísel. Zbylé čtyři skupiny si vybraly lehčí variantu – porovnání dvou čísel. Výsledky aktivy jsou uvedeny v následující tabulce (Tabulka 38).

Varianta	Těžší varianta (3 čísla)	Lehčí varianta (2 čísla)			
Označení skupiny	SK1	SK2	SK3	SK4	SK5
Bylo realizováno?	ANO	ANO	ANO	ANO	ANO
Počet chyb v 1. pokusu	3	1	2	3	2
Úspěšnost 1. pokusu [%]	70	90	80	70	70
Úspěšnost 2. pokusu [%]	70	100	100	70	90
Realizováno ve Scratchi?	NE	ANO	ANO	NE	NE
Obtížnost podle žáků	normální	snadná	normální	obtížná	normální

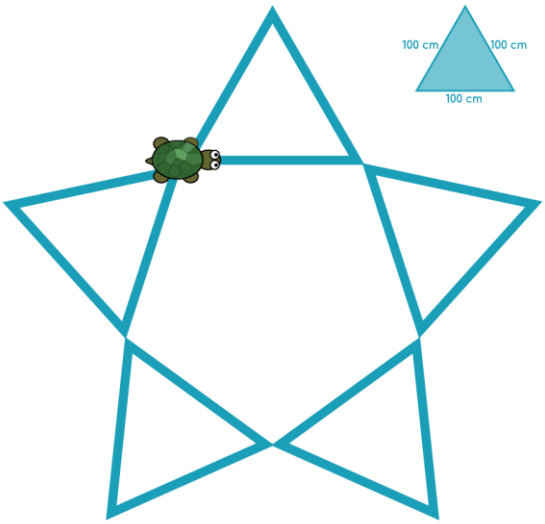
Tabulka 38: Výsledky 3. aktivity – tvorba

Úspěšnost žáků po prvním pokusu činila 76 %. Po opravě došlo ke zlepšení. Průměrná úspěšnost skupin ve druhém pokusu (po opravě) činila 86 %. Obtížnost, s ohledem na návrhy obtížnosti aktivity žáků, byla vyhodnocena jako **normální**.

Čtvrtá aktivita – želvička a hvězda

Poslední aktivita byla navržena se snahou pokrýt co nejvíce problematiku, která byla obsahem modelu výuky (Obrázek 15):

- návrh algoritmu realizovaný vývojovým diagramem nebo jiným způsobem (na papír nebo v PS Diagramu),
- želví grafiku (kurz Code.org, Scratch),
- programování (Scratch nebo Code.org studio).



Obrázek 15: Želvička hvězda

Žáci pracovali zcela samostatně. Při práci měli k dispozici veškerou podporu, kterou poskytovala webová stránka *„Programování na základce“*. Pokud byla v návrhu algoritmu chyba a žáci nebyli (na základě této chyby) schopni vytvořit funkční kód programu, byla s pomocí učitele odstraněna. Zároveň bylo skupině odečteno 10 % z celkové úspěšnosti návrhu. Výsledky závěrečné aktivity jsou představeny v následující tabulce (Tabulka 39).

Označení člena	Realizace návrhu algoritmu		Úspěšnost návrhu [%]	Zásah učitele do návrhu	Realizace programu v prostředí		Úspěšnost realizace [%]	Průměrná úspěšnost [%]
	papír	PS Diagram			Scratch	Code.org		
Z1	ANO	ANO	100	-	-	ANO	100	100
Z2	ANO	ANO	100	-	-	ANO	100	100
Z3	ANO	-	100	-	-	ANO	100	100
Z4	ANO	-	90	ANO	ANO	-	100	95
Z5	ANO	-	100	-	ANO	-	100	100
Z6	ANO	-	60	ANO	-	ANO	100	80
Z7	ANO	-	**50**	ANO	-	ANO	100	**75**
Z8	ANO	-	100	-	ANO	-	100	100
Z9	ANO	-	100	-	-	ANO	100	100
Z10	ANO	-	90	ANO	-	ANO	100	95
Z11	ANO	-	100	-	ANO	-	100	100
Σ	11	2	**90**	4	4	7	**100**	**95**

Tabulka 39: Výsledky 4. aktivity – Želvička a hvězda

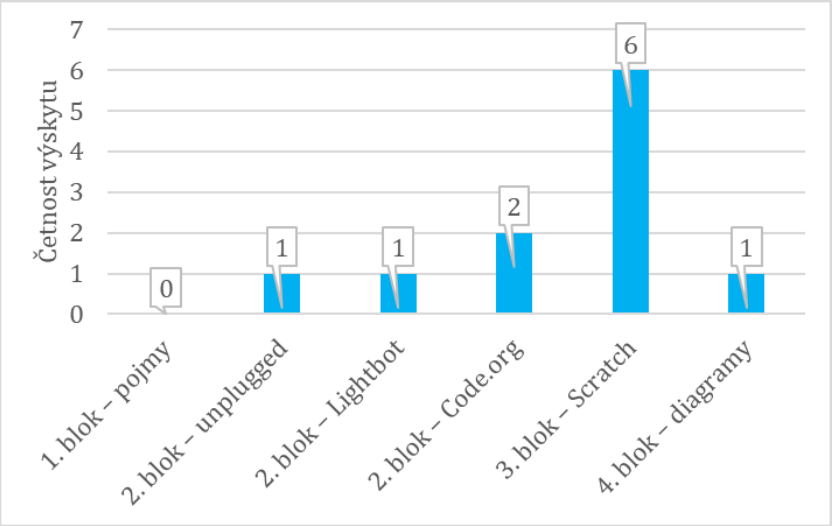
Všichni žáci využili k realizaci návrhu papír. Pouze dva žáci svůj návrh navíc převedli do elektronické podoby. Čtyřem žákům (36%) musel být navržený algoritmus učitelem upraven. Úspěšnost tvorby algoritmu činila 90 %. Všichni žáci dokázali napsat zcela funkční program. K tvorbě kódu si čtyři žáci vybrali prostředí Scratch, sedm žáků si vybralo Code.org studio. Průměrná úspěšnost žáků byla 95%. Aktivita byla vyhodnocena jako **snadná**.

7.3 Dotazníkové šetření

Dotazníkové šetření bylo koncipováno podle požadavků (Chráska, 2007). Šetření se v červnu 2018 zúčastnilo všech 11 žáků. Bylo rozhodnuto, že dotazník bude anonymní z důvodu vyšší validity odpovědí. Žáci v průběhu závěrečné hodiny vyplnili dotazníky obsahující 12 otázek (Příloha 11), ve kterých hodnotili zpětně výuku programování, respektive navržený model výuky.

7.3.1 Výsledky dotazníkového šetření

Otázka 1: Který ze čtyř bloků tě zaujal nejvíce? Proč?



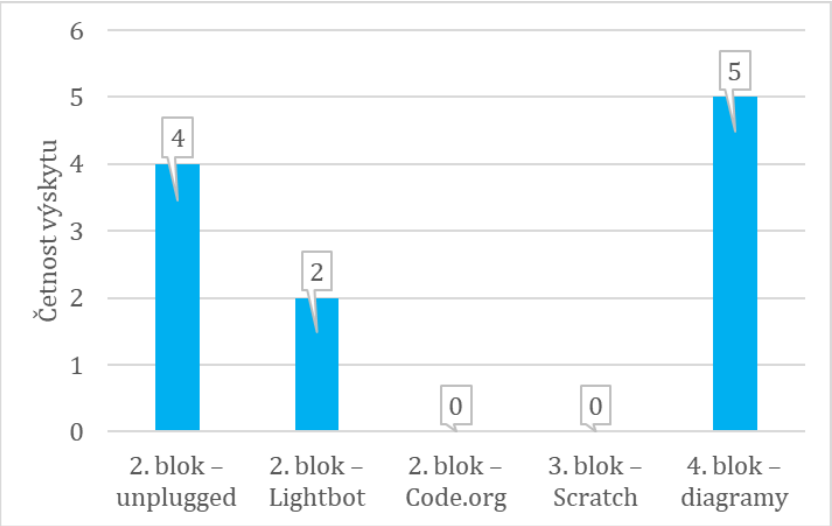
Graf 10: Odpovědi žáků na 1. otázku

Více než polovina žáků označila 3. blok (55%) s odůvodněním:

- *„Nejzajímavější.“* (dvakrát)
- *„Bylo to v něm nejlepší pracovat.“*
- *„Vhodné pro programování.“*
- *„Připadalo mi, že se to nejvíc blíží skutečné tvorbě hry.“*
- *„Mohl jsem tvořit, co jsem chtěl. Bylo to hodně na mě.“*

Žáci ocenili možnost volné kreativní tvorby, kterou Scratch poskytuje (Graf 10). Ostatní bloky byly žáky vybrány rovnoměrně.

Otázka 2: Který z bloků (kromě prvního) tě zaujal nejméně? Proč?



Graf 11: Odpovědi žáků na 2. otázku

Žáci označili především dva bloky (82 %) – vývojové diagramy a unplugged aktivity s následujícími důvody:

- *„Nebavilo mě to.“* (dvakrát)
- *„Moc mi to nešlo.“* (dvakrát)
- *„Nedalo se tam hrát ani vymýšlet hry.“*
- *„Příliš těžké.“*
- *„Moc snadné.“*
- *„Nudné.“*
- *„Scratch a Lightbot byly lepší.“*

Z odpovědí vyplývá, že žáci v tomto věku preferují především na hru orientovaná prostředí (Graf 10 a Graf 11). Tvorba vývojových diagramů v PS Diagram se všemi pravidly a symbolikou byla pro některé z žáků zřejmě příliš nezáživná.

Otázka 3: Která konkrétní aktivita tě bavila nejvíce?

Odpovědi žáků byly opravdu pestré. Pouze jedenkrát došlo ke shodě:

- *„Vlastní projekt.“* (dvakrát)
- *„Ta první hra ve Scratchi.“*
- *„Bavilo mě odpovídat v Menti a v Plickers.“*
- *„Lekce 1 ve Scratchi.“*
- *„První lekce. Výzvy ve Scratchi.“*

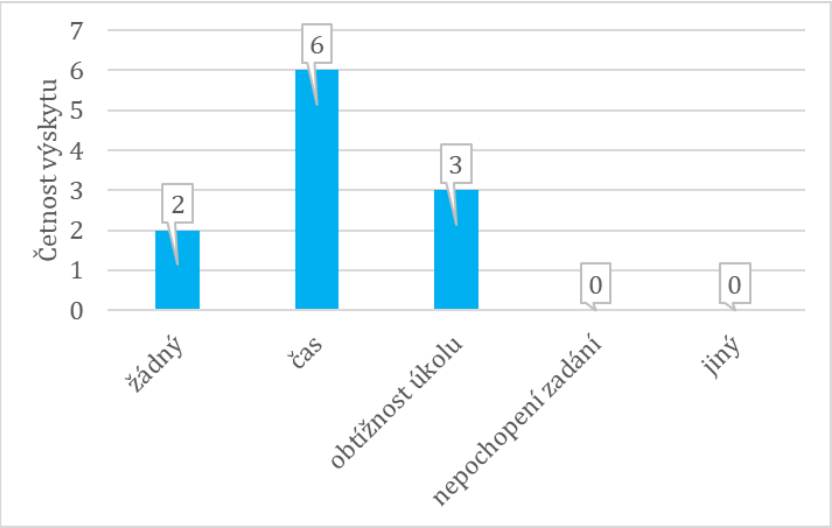
- *„Kalkulačka ve Scratchi.“*
- *„Jak jsme kreslili obrázek podle kódu.“*
- *„Nejvíce mě asi bavil Lightbot. Ta první verze.“*
- *„PSDiagram 3 – želvička a hvězda.“*
- *„Malíř v kurzu.“*

Otázka 4: Která konkrétní aktivita tě bavila nejméně (nebavila)? Proč?

- *„Poslední lekce ve Scratchi. Nemohli jsme se dohodnout, co uděláme a pak jsme to nestíhali.“*
- *„Rekurze a odpočet.“*
- *„Lightbot, protože byl moc snadný.“*
- *„Lightbot. Druhá část byla příliš obtížná.“*
- *„Vývojový diagram na porovnávání.“*
- *„Unplugged, protože to bylo bez počítače.“*
- *„Hra ve Scratchi, protože to bylo těžké.“*
- *„Pravidla vývojových diagramů. To, jak se kreslí.“*
- *„Porovnávání čísel v PS Diagramu.“*
- *„Vytvořit vývojový diagram na porovnávání čísel.“*
- *„Všechny těžké bez počítačů.“*

Z odpovědí vyplývá, že neoblíbené aktivity byly především ty s nevyváženou obtížností. Podle zaměření pak byla nejméně oblíbená tvorba vývojových diagramů (4 výskyty).

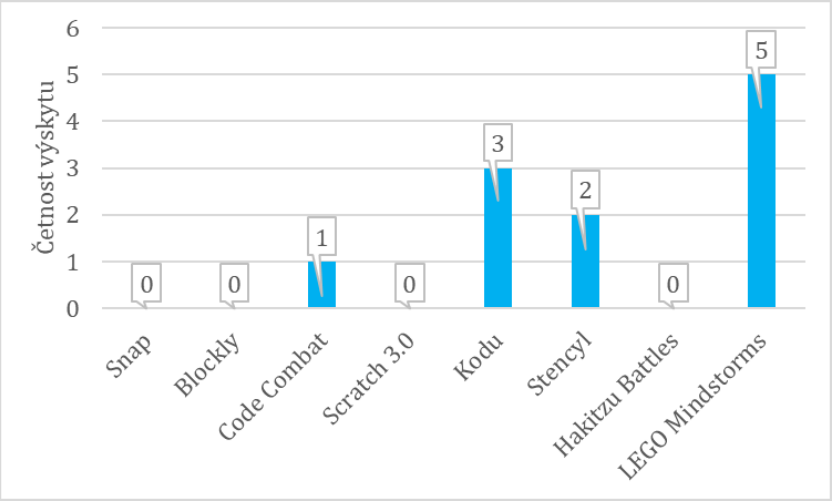
Otázka 5: Který z důvodů ti nejvíce bránil v dokončení práce (funkčních programů ve Scratchi, na Code.org, úrovní Lightbota atd.)?



Graf 12: Odpovědi žáků na 5. otázku

Žáci vybrali možnosti nedostatek času (55 %), obtížnost úkolů (27 %) nebo žádný z důvodů (18 %). Nejčastěji byl žáky vybrán důvod nedostatku času (Graf 12), což nás přivádí k otázce, zda by nebylo vhodné zvýšit hodinovou dotaci nebo snížit počet aktivit.

Otázka 6: Které z prostředí, programů nebo stavebnic (v závěru výuky) tě zaujaly natolik, že bys v nich/s nimi rád pracoval?



Graf 13: Odpovědi žáků na 6. otázku

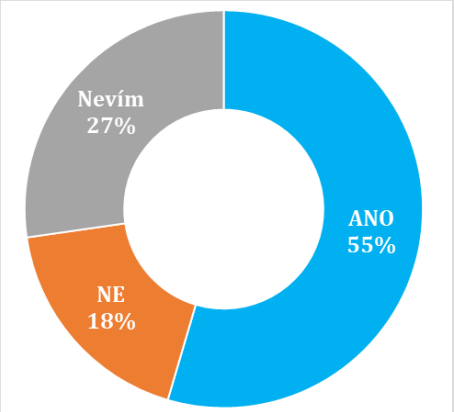
Nejzajímavější je pro žáky robotická stavebnice LEGO Mindstorms (45,5 %). Žákům přišlo také zajímavé prostředí Kodu (27,3 %). Naopak, žáky nezaujaly Snap, Blockly, Hakitzu ani nově připravovaná verze Scratche (Graf 13).

Otázka 7: Pracoval jsi na některých školních projektech doma?

Čtyři žáci (36 %) pracovali na školních projektech také doma. Sedm žáků (64 %) na projektech doma nepracovalo.

Otázka 8: Budeš v programování pokračovat i po ukončení kroužku?

Z odpovědí (Graf 14) vyplývá, že na šest žáků (55 %) měl model výuky pozitivní vliv ve smyslu dalšího informálního učení v oboru programování. Dva žáci (18 %) tuto možnost dopředu odmítají, tři žáci (27 %) neví.



Graf 14: Odpovědi žáků na 8. otázku

Otázka 9: Byl(a) jsi s náplní práce spokojen(a)? Splnil kroužek tvé představy?

Všech jedenáct žáků vybralo variantu *„Ano, byl. Ano, splnil.“*

Otázka 10: Odradila tě účast v kroužku od dalšího programování? Pokud ano, proč?

Dva žáci (18 %) odpověděli kladně. Žáci odpověď vysvětlili následovně:

- *„Nečekal jsem, že je programování těžké.“*
- *„Moc těžké. Potom už mě to přestávalo bavit.“*

Otázka 11: Vyhovovala ti práce ve dvojici (trojici)?

Všech jedenáct žáků uvedlo, že práce ve dvojici (trojici) byla vhodnou volbou.

Otázka 12: Vyhovoval ti systém hodnocení – odměn?

Všichni žáci byli spokojeni s výběrem odměn formou gamifikačních prvků. Sedm žáků (64 %) se ohradilo proti způsobu udělování karet, které označili za nedosažitelné, například:

„Nešlo nasbírat tolik puzzle, abych získal karty.“

„Asi bych snížil, kolik puzzle musím mít nebo by to pan učitel mohl vymyslet jinak.“

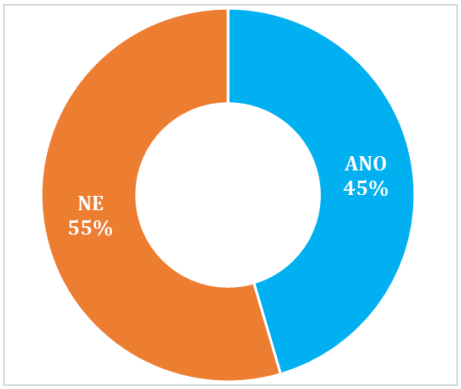
Listopadová otázka

Všem jedenácti žákům, kteří se zúčastnili ověřovacího procesu, byla v průběhu listopadu 2018 položena otázka, zda se doma či jinde věnují programování.

Odpovědi žáků:

- *„Už ne. Nemám na to čas. V Bobříkovi jsem měl o 30 bodů více, než jsem míval.“*
- *„Ne, neprogramuju.“* (třikrát)
- *„K narozeninám jsem dostal to LEGO ev3, tak stavím roboty.“*
- *„Ne, ale bavilo mě to. Možná, že zase začnu.“*
- *„Ne. Zkoušel jsem Kodu, ale přestal jsem. Teď se musím učit.“*
- *„Programujeme ve Stencyl. Zkoušeli jsme Game Maker, ale bylo to moc těžké.“* (dvakrát)
- *„Občas zapnu Scratch, ale málokdy. Tak jednou za měsíc.“*
- *„Pracuju v novém Scratchi. Už je v češtině. Jen mě štve, že se nedá ukládat online.“*

Podle odpovědí žáků lze v porovnání s červnovými reakcemi spatřit nepatrný pokles zájmu o programování. Žádný z žáků, kteří v červnu nebyli rozhodnuti (vybrali odpověď *„Nevím.“*), doma v programování nepokračoval (Graf 15). Naopak, pět žáků se po pěti měsících stále programování věnuje.



Graf 15: Listopadové odpovědi žáků na 8. otázku

7.4 Shrnutí pilotního nasazení

Pilotního nasazení se zúčastnilo jen jedenáct žáku, což velmi malý vzorek žáků.

Práce ve skupinách – sociální dimenze učení

Žáci označili práci ve skupinách jako dobrou volbu, na které by nic neměnili. Na práci ve skupinách oceňovali především možnost vzájemné spolupráce, lepšího pochopení látky, možnost poradit se a tvořit společně.

Fungující spolupráce byla v hodinách zájmového útvaru jasně patrná. Žákům bylo umožněno pracovat také samostatně, což ve většině situací odmítli.

Prostředky a nástroje – výběr zdrojů

Přestože žáci Scratch nepovažovali za nejjednodušší nástroj pro tvorbu programů, z výstupů dotazníkového šetření vyplývá, že si Scratch oblíbili. Náročnost tvorby programu v Scratchi nepřímo potvrdili výběrem nástroje při realizaci závěrečné aktivity (čtvrtá aktivita – želvička a hvězda). Scratch si vybralo jen 36 % žáků, zatímco Code.org studio 64 % žáků. Na Scratchi žáci oceňovali rozsah možností tvořit prakticky cokoliv, co je napadlo – od jednoduchých animací po komplexní hry. Jako oblíbené prostředí žáci označili také Code.org studio. Na studiu oceňovali především podobnost se Scratchem a jednoduchost, respektive kvalitní zpětnou vazbu, kterou poskytuje.

Nejméně oblíbenými bloky výuky byly pro žáky unplugged aktivity (36 % žáků) a vývojové diagramy (64 % žáků). Unplugged aktivity žáci považovali za nejméně důležité, avšak náročné, a to především první a třetí unplugged aktivitu.

Hodinová dotace – časový plán

Z odpovědí žáků na 5. otázku vyplývá, že časová dotace určená jednotlivým aktivitám nebyla pro většinu žáků dostačující. Přestože byla hodinová dotace stanovena tak, že 10 minut bylo vždy vyhrazeno úvodu – seznámení s cílem hodiny a aktivováním žáků, závěrečných 10 minut bylo vymezeno shrnutí učiva a reflexi. Čas vymezený jednotlivým aktivitám byl stanoven jako nejméně trojnásobek času potřebný k zvládnutí aktivity učitelem. Problém s časovou dotací určenou pro jednotlivé aktivity (především aktivity ve Scratchi) byl pozorován také při samotné výuce. Týkal se především skupin žáků, které se plně nesoustředily na konkrétní příklady, ale zkoušely experimentovat či zkoumat kódy jiných uživatelů.

Systém hodnocení – gamifikace

Systém hodnocení v podobě sbírání puzzle dílků, odznaků a kartiček byl pro žáky velmi motivující a splnil očekávání. Přestože žáci nesouhlasili s množstvím puzzle dílků, kterých bylo potřeba nasbírat k získání karet, nebude systém modifikován.

Obtížnost aktivit

Obsah výuky s optimální obtížností byl jedním ze zásadních elementů při tvorbě návrhu modelu výuky, neboť výběr příliš snadných aktivit by mohl mít za následek nemotivovanost žáků, zatímco výběr příliš obtížných aktivit by mohl žáky demotivovat.

Obsah výuky byl vybírán s ohledem na obsah výuky matematiky 6. ročníku (celá čísla, dělitelnost, velikost úhlu). Většina žáků 6. ročníků neměla zkušenosti s tvorbou vektorových či rastrových obrázků. Výběr aktivit byl podle reakcí žáků na 5. otázku dotazníkového šetření volen správně. Také v průběhu výuky nebyly zpozorovány nedostatky, které by se týkaly použití matematického aparátu.

Pro každou z aktivit byla při tvorbě výukové webové stránky určena předpokládaná obtížnost (v tabulce níže označená jako PředOb) označená počtem hvězdiček. V průběhu pilotního nasazení bylo ověřováno, zda skutečná obtížnost (v tabulce níže označená jako SkutOb) aktivity odpovídá předpokládané obtížnosti (Tabulka 40).

Aktivita	PředOb	SkutOb	Aktivita	PředOb	SkutOb
Origami	1,5	normální	Scratch 1	1	jednodušší
Bitmapový stroj	1	velmi snadná	Scratch 2	2,5	normální
LEGO	1	obtížná	Scratch 3	2,5	normální
Příšerky	0,5	velmi snadná	Scratch 4	2	normální
Lightbot 1	1,5	velmi snadná	Scratch 5	3	trochu obtížnější
Lightbot 2	2	normální	Scratch 6	2,5	normální
Lightbot 3	2,5	snadná	Scratch 7	3	**obtížná**

Umělec (3. lekce)	2,5	snadná	Scratch 8	4	nehodnoceno[13]
Umělec: proměnné (6. lekce)	2,5	snadná	Čteme a tvoříme diagramy	1	snadná
Umělec: for cykly (10. lekce)	2,5	normální	Úprava diagramů	1,5	normální
Umělec: fukce (12. lekce)	2,5	snadná	Tvorba diagramů	2,5	normální
Umělec: parametry (14. lekce)	2,5	normální	Želvička a hvězda	2,5	snadná

Tabulka 40: Porovnání předpokládané a reálné obtížnosti

Předpokládaná obtížnost sedmnácti aktivit neodpovídala pozorované obtížnosti. U těchto aktivit bude obtížnost modifikována na novou (pozorovanou) hodnotu.

Výsledky ukazují na fakt, že splnění jednotlivých aktivit bylo v možnostech žáků, kteří se zúčastnili pilotního nasazení navrženého modelu do výuky. Problematickými se jevily aktivity *„Odpočet“*, *„Vykreslení n-úhelníků“* a 5. prvek komplexní hry – *„enemáci, kolize a konec hry“* ve třetím bloku, které většina žáků nebyla schopna dokončit. *„Vykreslení n-úhelníků“* je vybranou aktivitou pro nadané žáky, která nebude vyřazena. Aktivita *„Odpočet“* byla úspěšně dokončena čtyřmi skupinami, které ke splnění potřebovaly jen více času, než bylo vymezeno. V případě aktivity zaměřené na herní prvky *„enemáci, kolize a konec hry“* je vhodné přemýšlet o případné modifikaci – snížení náročnosti nebo prodloužení času potřebného k vyřešení problému.

Zájem žáků o další rozvoj vlastních dovedností v oblasti programování

Žádný z žáků, kteří se zúčastnili ověřovacího procesu, neměl v úvodu výuky zkušenosti v oblasti programování. Naopak, jeden žák 6. ročníku měl o programování zcela mylnou představu (střih videa). Výsledek závěrečného dotazníkového šetření naznačuje, že více než polovina žáků by se ráda programování věnovala i nadále. Po dalších pěti měsících byla všem účastníkům položena otázka ověřující skutečnou situaci. Pět žáků zaujalo programování natolik, že se mu ve svém volném čase stále věnují, a to buď pravidelně, nebo okrajově.

[13] Osmá lekce 3. bloku (Scratch) je vlastním projektem žáků. Její obtížnost závisí na konkrétním výběru projektu. Nelze objektivně určit konkrétní obtížnost aktivity.

7.4.1 Návrh na modifikaci modelu

Z pilotního nasazení vyplynuly potřeby modifikace, které je vhodné uskutečnit. První bude modifikováno pořadí unplugged aktivit tak, aby žáci v úvodu bloku začínali snazšími aktivitami. Další změna se týká zvýšení časových dotací v blocích zaměřených na block-based programování, především pak navýšení hodinové dotace 3. bloku. Při navrhování časových dotací pro jednotlivé aktivity nebyla zohledněna zvídavost žáků ani zájem experimentovat. Změny se budou týkat druhého, třetího a čtvrtého bloku, konkrétně deseti aktivit. Navýšení časových dotací pro jednotlivé aktivity jsou uvedeny v tabulce níže (Tabulka 41). Navýšení vychází z vypozorovaných časových tísní, do kterých se skupiny dostávaly v průběhu pilotního nasazení. Navrhovaná navýšení činí celkově 180 minut.

Blok	2. (Code.org kurz)				3.					4.
Aktivita	3. lekce	10. lekce	12. lekce	14. lekce	Lekce #1	Lekce #2	Lekce #4	Lekce #5	Lekce #7	Čtení a psaní
Původní časová dotace [min]	60	60	60	60	40	80	60	120	180	20
Nová časová dotace [min]	70	80	80	80	50	100	80	140	210	40
Nárůst [min]	60				100					20

Tabulka 41: Navrhované navýšení časových dotací

Alternativou k navyšování časových dotací by mohla být redukce počtu aktivit, které byly v rámci shrnutí jednotlivých aktivit pilotního nasazení vyhodnoceny buď jako velmi snadné, nebo příliš obtížné. Realizována však bude první varianta (navýšení časové dotace).

Další změnou, která bude realizována, je změna obtížnosti jednotlivých aktivit. Změny v obtížnosti jsou uvedeny v tabulce níže (Tabulka 42).

Aktivita	Původní obtížnost [hvězdičky]	Nová obtížnost [hvězdičky]	Aktivita	Původní obtížnost [hvězdičky]	Nová obtížnost [hvězdičky]
Bitmapový stroj	1	0,5	Scratch 2	2,5	1,5
LEGO	1	2,5	Scratch 3	2,5	1,5
Lightbot 1	1,5	0,5	Scratch 4	2	1,5
Lightbot 3	2,5	1	Scratch 5	3	2
Umělec (3. lekce)	2,5	1	Scratch 6	2,5	1,5
Umělec: proměnné (6. lekce)	2,5	1	Scratch 7	3	2,5
Umělec: for cykly (10. lekce)	2,5	2	Tvorba diagramů	2,5	2
Umělec: fukce (12. lekce)	2,5	1	Želvička a hvězda	2,5	1
Umělec: parametry (14. lekce)	2,5	2	-	-	-

Tabulka 42: Navrhované změny v obtížnosti

Poslední modifikace se týká aktivity „Bitmapový stroj“, ve které měli žáci problém při přechodu na nový řádek. Do výukového materiálu (webové stránky) bude zařazen návrh žáků v grafické podobě.

8. Závěr

Diplomová práce se zabývala problematikou programování na základních školách v České republice. Přestože zatím nevešel v platnost nově revidovaný rámcový vzdělávací plán, který má zásadně změnit postavení tematické oblasti algoritmizace a programování v RVP, existuje množství učitelů základních škol, kteří se ve výuce problematikou spojenou s algoritmizací a programováním ve větší, či menší míře zabývají. Učitelé přitom využívají celou řadu prostředků, nástrojů a metodických přístupů. Nejčastěji jsou na českých školách využívány unplugged aktivity a úlohy ze soutěže Bobřík informatiky, ale stále častěji je do výuky zařazováno programování ve vizuálních (blokově orientovaných) programovacích jazycích. Školy se však nesnaží vychovat armádu mladých programátorů, kteří by zaplnili pracovní pozice na trhu práce. Snahou škol je vybavit žáky schopnostmi být kreativní, řešit problémy a k jejich řešení využívat efektivních postupů – informaticky myslet. Pokud učitelé mají zájem učit žáky informaticky myslet, je vhodným prostředkem právě algoritmizace, tvorba kódu, respektive programování.

V rámci diplomové práce vznikl návrh modelu pro podporu výuky programování na základních školách. Samotnému návrhu modelu výuky předcházelo několik kroků. Konkrétně se jednalo o zmapování aktuálního stavu výuky programování na základních školách. Další krok vedl k ujasnění si nepřehledné situace v prostředcích a nástrojích, kterých je možné ve výuce využívat, které ze zdrojů jsou učiteli a odborníky preferovány a proč. Preferované zdroje byly následně zhodnoceny vybranými evaluačními kritérii. Tato kritéria byla vybrána na základě analýzy evaluačních kritérií pro hodnocení edukačních programů, aplikací, online zdrojů a webových stránek. Do návrhu modelu výuky pro podporu programování byly vybrány nejlépe hodnocené zdroje.

Následně byl navrhnut rámec očekávaných výstupů a vytvořen časově-tematický plán obsahující očekávané výstupy a učivo. Výběr očekávaných výstupů vycházel ze tří zdrojů – návrhu rámce očekávaných výstupů nového předmětu – Informatika, dále z publikací Computing in the national curriculum pro učitele prvního a druhého stupně škol ve Velké Británii.

Výše provedené kroky se staly východiskem pro realizaci vlastního návrhu modelu výuky. Obsah byl publikován formou webové stránky zahrnující konkrétní výukové

materiály. Tato webová stránka je učitelům, žákům i veřejnosti k dispozici na <https://www.3zskadan.cz/prgm/>. Současně s webovou stránkou vznikla metodická příručka pro učitele (Příloha 13), jako návod, jak s žáky při průchodu navrženým modelem postupovat.

Ověření proběhlo pilotním nasazením na základní škole v rámci zájmového útvaru, a to po dobu 9 měsíců. Pilotního nasazení se zúčastnil pouze malý vzorek žáků (11 žáků) šestého a sedmého ročníku ve věku 11–13 let.

V rámci ověřovacího procesu byly zjištěny konkrétní nedostatky, především rozdíly v navrhované a reálné obtížnosti jednotlivých aktivit. Také časové dotace vyhrazené pro plnění jednotlivých aktivit nekorespondovaly s reálnými potřebami žáků. Zjištěná skutečnost se týkala především deseti aktivit. Všechny zjištěné nedostatky budou v návrhu modelu přepracovány.

S ohledem na úspěšnost žáků v konkrétních aktivitách jednotlivých bloků a s ohledem na reakce žáků, ať už přímo pozorované v průběhu jednotlivých vyučovacích hodin, nebo zprostředkované odpověďmi na otázky obsaženými v závěrečném dotazníkovém šetření, lze konstatovat, že navržený model splnil svůj účel – žáci si osvojili elementární znalosti a dovednosti v oblasti algoritmizace a programování. Celkem 45,5 % žáků se programování stále věnuje, a to i po uplynutí pěti měsíců od ukončení pilotního nasazení.

Seznam informačních zdrojů

CHRÁSKA, Miroslav. *Metody pedagogického výzkumu: základy kvantitativního výzkumu*. Praha: Grada, 2007. Pedagogika (Grada). ISBN 978-80-247-1369-4.

JANÍK, Tomáš, Petr KNECHT a Petr NAJVAR. *Nástroje pro monitoring a evaluaci kvality výuky a kurikula.* Brno: Paido, 2010. Pedagogický výzkum v teorii a praxi. ISBN 978-80-7315-209-3.

PÁVKOVÁ, Jiřina. Psychologie pro pedagogy: sociální a pedagogická psychologie. V Praze: Univerzita Karlova, Pedagogická fakulta, 2014. ISBN 978-80-7290-676-5.

PRŮCHA, Jan, Jiří MAREŠ a Eliška WALTEROVÁ. Pedagogický slovník. 4. aktualiz. vyd. Praha: Portál, 2003. ISBN 80-7178-772-8.

RAMBOUSEK, Vladimír. *Materiální didaktické prostředky*. V Praze: Univerzita Karlova, Pedagogická fakulta, 2014. ISBN 978-80-7290-664-2.

Seznam elektronických informačních zdrojů

BARTYZAL, Miroslav. PS Diagram: Algoritmus diagramem snadno a rychle [online]. [cit. 2018-10-30]. Dostupné z: http://www.psdiagram.cz/

Beaver Informatics Malaysia: Malaysian computational thinking and informatics contest [online]. *[cit. 2018*-11-08]. Dozstupné z: http://beaver.my

BELL, Tim, Ian H. WITTEN a Mike FELLOWS. *CS Unplugged: An enrichment and extension programme for primary-aged students* [online]. 2015 [cit. 2018-10-20]. Dostupné z: https://classic.csunplugged.org/wp-content/uploads/2015/03/CSUnplugged_OS_2015_v3.1.pdf

BERRY, Miles. *Computing in the national curriculum: A guide for primary teachers* [online]. Computing at School, 2013 [cit. 2018-10-20]. ISBN 978-1-78339-143-1. Dostupné z: https://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

Bootstrapmade. Bell – Free Multipurpose Bootstrap 4 Template [online]. [cit. 2018-11-03]. Dostupné z: https://bootstrapmade.com/bell-free-bootstrap-4-template/

BRDIČKA, Bořivoj. *Učení s počítačem: Vyhodnocování výukových programů* [online]. 1995 [cit. 2018-10-08]. Dostupné z: http://it.pedf.cuni.cz/~bobr/ucspoc/

BRENNAN, Karen, Christan BALCH a Michelle CHUNG. *CREATIVE COMPUTING* [online]. Harvard Graduate School of Education [cit. 2018-11-20]. Dostupné z: http://scratched.gse.harvard.edu/guide/files/CreativeComputing20141015.pdf

Code.org- Course 4. In: Youtube [online]. [vid. 2018-11-08]. Playlist uživatele iLearn Code. Dostupné z: https://www.youtube.com/channel/UCMdRUTHi8mpPC3ia1JTQbIA/playlists

[Code.org]. *Course D: Programování Na Milimetrovém Papíru #1* [online]. 2018 [cit. 2018-11-08]. Dostupné z: https://studio.code.org/s/coursed-2018/stage/1/puzzle/1

[Code.org]. *CS Principles Curriculum Guide 2018 - 2019* [online]. 2018, 55 s. [cit. 2018-10-20]. Dostupné z: http://fliphtml5.com/pcrbk/tfbw/basic

[Code.org]. *CS Principles 2018: Intro to Programming* [online]. 2018 [cit. 2018-11-08]. Dostupné z: https://curriculum.code.org/csp-18/unit3.pdf

Common Sense Education: Best Apps and Websites for Learning Programming and Coding [online]. [cit. 2018-10-10]. Dostupné z: https://www.commonsense.org/education/top-picks/best-apps-and-websites-for-learning-programming-and-coding

Česká školní inspekce dnes zahájila zjišťování výsledků žáků. *ČŠI: Česká školní inspekce* [online]. Praha, 2015 [cit. 2018-09-26]. Dostupné z: https://www.csicr.cz/Stredni-cast/Tiskove-zpravy/Ceska-skolni-inspekce-dnes-zahajila-zjistovani-vys

DAVIS, Matt. *Edutopia: Teach Your Kids to Code: 6 Beginner's Resources for Parents* [online]. 2016 [cit. 2018-10-10]. Dostupné z: https://www.edutopia.org/blog/teach-kids-coding-resources-parents-matt-davis

DERUY, Emily. In Finland, Kids Learn Computer Science Without Computers: Students can learn the basics with a set of knitting needles. The Atlantic [online]. 2017 [cit. 2018-10-07]. Dostupné z: https://www.theatlantic.com/education/archive/2017/02/teaching-computer-science-without-computers/517548/

DODGE, David. *Codakid: Top 7 Kids Coding Languages of 2018* [online]. 2018 [cit. 2018-10-10]. Dostupné z: https://codakid.com/kids-coding-languages/

DOHNAL, Pavel. *Programování na základních školách* [online]. Brno, 2009 [cit. 2018-11-16]. Dostupné z: https://is.muni.cz/th/yb97y/Diplomova_prace_Pavel_Dohnal.pdf. Diplomová práce. Masarykova univerzita, Pedagogická fakulta. Vedoucí práce Martin Dosedla.

EdSurge Guide: Teaching Kids to Code [online]. [cit. 2018-10-10]. Dostupné z: https://www.edsurge.com/research/guides/teaching-kids-to-code

EVANS, Frank. *ESchool News: How to choose the right programming language for students* [online]. 2015 [cit. 2018-10-10]. Dostupné z: https://www.eschoolnews.com/2015/12/10/programming-language-330/2/?all

Google Developers: Blockly – Introduction to Blockly [online]. 2018 [cit. 2018-10-15]. Dostupné z: https://developers.google.com/blockly/guides/overview

KEMP, Peter. *Computing in the national curriculum: A guide for secondary teachers* [online]. Computing at School, 2014 [cit. 2018-10-20]. ISBN 978-1-78339-376-3. Dostupné z: https://www.computingatschool.org.uk/data/uploads/cas_secondary.pdf

KLEMENT, Milan. Možnosti evaluace výukových programů. In: *Trendy technického vzdělávání 2005* [online]. 2005, 17 - 29 [cit. 2018-10-07]. Dostupné z: https://www.researchgate.net/publication/292983376_MOZNOSTI_EVALUACE_VYUKOVYCH_PROGRAMU

KREJSA, Jan. *Výuka základů programování v prostředí Scratch* [online]. 2014 [cit. 2018-10-07]. Dostupné z: https://theses.cz/id/b5f11x/DP_Krejsa_Scratch.pdf. Diplomová práce. Jihočeská univerzita v Českých Budějovicích, Pedagogická fakulta, Katedra informatiky. Vedoucí práce Jiří Vaníček.

KRUEGER, Nicole. Preparing students for jobs that don't exist. *ISTE* [online]. 2017 [cit. 2018-10-17]. Dostupné z: https://www.iste.org/explore/articledetail?articleid=1002

LE, Dustin. *Edudemic: The Three Best Free Coding Websites for Kids* [online]. 2015 [cit. 2018-10-10]. Dostupné z: http://www.edudemic.com/the-three-best-free-coding-websites-for-kids/

Lightbot: How does Lightbot teach programming? [online]. [cit. 2018-10-07]. Dostupné z: http://lightbot.com/hoclearn.html

MediaGuru: Bounce rate [online]. [cit. 2018-10-11]. Dostupné z: https://www.mediaguru.cz/slovnik-a-mediatypy/slovnik/klicova-slova/bounce-rate/

Mitchel Resnick: Scratch. In: *Youtube* [online]. 29.10.2010 [cit. 2018-05-25]. Kanál uživatele spotlightdml. Dostupné z: https://www.youtube.com/watch?v=oNSao-amctk

Návrh revizí rámcových vzdělávacích programů v oblasti informatiky a informačních a komunikačních technologií [online]. Národní ústav pro vzdělávání, 2018 [cit. 2018-10-20]. Dostupné z: http://www.nuv.cz/file/3362/

NEUMAJER, Ondřej. Proč a jak inovovat pojetí ICT v rámcových vzdělávacích programech. *Metodický portál RVP.CZ: inspirace a zkušenosti učitelů* [online]. 2009 [cit. 2018-10-07]. Dostupné z: https://clanky.rvp.cz/clanek/o/z/2989/proc-a-jak-inovovat-pojeti-ict-v-ramcovych-vzdelavacich-programech.html/?oblibene=1

[ISTE/CSTA]. *Operational Definition of Computational Thinking: for K–12 Education* [online]. 2011 [cit. 2018-10-17]. Dostupné z: http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf

O'REILLY, Tim. What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. *COMMUNICATIONS & STRATEGIES* [online]. O'Reilly Media, 2007, (1), 21 [cit. 2018-10-08]. Dostupné z: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1008839

PEXA, Petr. *Kritéria hodnocení elektronických vzdělávacích materiálů zobrazovaných webovými prohlížeči* [online]. 2011 [cit. 2018-10-08]. ISSN 1803-537x. Dostupné z: http://jtie.upol.cz/pdfs/jti/2011/02/09.pdf

RESNICK, Mitchel. Let´s teach kids to code: TED Talk. In: *TED* [online]. 2012 [cit. 2018-10-17]. Dostupné z: https://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code

RESNICK, Mitchel, Brian SILVERMAN, Yasmin KAFAI, et al. Scratch. *Communications of the ACM* [online]. 2009, 52(11) [cit. 2018-10-08]. DOI: 10.1145/1592761.1592779. ISSN 00010782. Dostupné z: http://portal.acm.org/citation.cfm?doid=1592761.1592779

ŘEZNÍČKOVÁ, Blanka. *Využití VISUAL BASIC ve stereometrii na ZŠ* [online]. Olomouc, 2014. Diplomová práce (Mgr.). Univerzita Palackého v Olomouci, Pedagogická fakulta, Katedra matematiky [cit. 2018-10-07]. Dostupné z: https://theses.cz/id/umqoay/Diplomova_prace.pdf

Scratch Statistics – Imagine, Program, Share [online]. 2018 [cit. 2018-10-15]. Dostupné z: https://scratch.mit.edu/statistics/

SCREAWN, Julian. Scratch Project Rubrics [online]. [cit. 2018-10-28]. Dostupné z: https://jts.design/scratch-project-rubrics/

Statistická ročenka školství - 2017/2018 - výkonové ukazatele [online]. 2018 [cit. 2018-10-08]. Dostupné z: http://toiler.uiv.cz/rocenka/rocenka.asp

STRATEGIE DIGITÁLNÍHO VZDĚLÁVÁNÍ DO ROKU 2020 [online]. 2014 [cit. 2018-11-14]. Dostupné z: http://www.vzdelavani2020.cz/images_obsah/dokumenty/strategie/digistrategie.pdf

ŠTÍPEK, Jiří a Petra VAŇKOVÁ. Vybraná zjištění výzkumu stavu a pojetí rozvoje informačně technologických kompetencí na základních školách. Orbis Scholae [online]. Praha: Karolinum, 2014, 8(1), 47-64 [cit. 2018-10-07]. Dostupné z: http://karolinum.cz/ink2_stat/dload.jsp?prezMat=103636

ŠTÍPEK, Jiří, Vladimír RAMBOUSEK a Petra VAŇKOVÁ. Vybrané výsledky výzkumu rozvoje digitálních kompetencí žáků na ZŠ. *Pedagogika* [online]. 2015, 65(3), 259-273 [cit. 2018-10-07]. ISSN 2336-2189. Dostupné z: http://pages.pedf.cuni.cz/pedagogika/?p=11256&lang=cs

TOMCSÁNYIOVÁ, Monika. Potvůrky. In: *BOBŘÍK INFORMATIKY: Výběr z úloh národních kol soutěže 2010–2014* [online]. Praha: NÚV, 2015 [cit. 2018-11-11]. ISBN 978-80-7481-142-5. Dostupné z: http://www.nuv.cz/uploads/Publikace/bobrik2017.pdf

VANÍČEK, Jiří. *Kritéria evaluace výukových programů pro vyučování matematiky pomocí počítače* [online]. Jihočeská univerzita, Pedagogická fakulta [cit. 2018-10-07]. Dostupné z: http://amper.ped.muni.cz/~svobodka/useful/kriteria_sw.pdf

VANÍČEK, Jiří. *Potenciální a skutečný dopad informatické soutěže do změn kurikula ICT v České republice* [online]. 2012, 15-24 [cit. 2018-10-07]. Dostupné z: https://www.bebras.org/sites/default/files/documents/publications/Vanicek-2012.pdf

VINCENT, Tony. Learning in Hand with Tony Vincent: Ways to Evaluate Educational Apps [online]. 2012 [cit. 2018-10-08]. Dostupné z: https://learninginhand.com/blog/ways-to-evaluate-educational-apps.html

Výběr z adresáře škol a školských zařízení [online]. 2018 [cit. 2018-03-18]. Dostupné z: http://stistko.uiv.cz/registr/vybskolrn.asp

WALKER, Harry. *Evaluation Rubric for Mobile Applications (APPS)* [online]. Johns Hopkins University, 2010 [cit. 2018-10-07]. Dostupné z: https://docs.google.com/document/d/169iZMhNMB_ycJLRT2zSRbozAR_1TiGQ1-FmjfKzoppk/edit

YOUNG, Josh. *CodeCamp: 20 Best Programming Languages for Kids* [online]. 2018 [cit. 2018-10-13]. Dostupné z: https://www.codecamp.com.au/blog/20-best-programming-languages-kids

ZATLOUKAL, Tomáš. *Výroční zpráva České školní inspekce za školní rok 2013/2014* [online]. Praha: Česká školní inspekce, 2014 [cit. 2018-10-07]. ISBN 978-80-905632-7-8. Dostupné z: https://www.csicr.cz/Csicr/media/Prilohy/PDF_el._publikace/V%C3%BDro%C4%8Dn%C3%AD%20zpr%C3%A1vy/Vyrocni_zprava_CSI_2013_2014.pdf

ZATLOUKAL, Tomáš. *Výroční zpráva České školní inspekce za školní rok 2015/2016* [online]. Praha: Česká školní inspekce, 2016 [cit. 2018-10-07]. ISBN 978-80-88087-09-0. Dostupné z: https://www.csicr.cz/getattachment/cz/Dokumenty/Vyrocni-zpravy/Vyrocni-zprava-Ceske-skolni-inspekce-za-skolni-(2)/Vyrocni_zprava_CSI_2015-2016.pdf

Seznam zkratek

CAWI	Dotazníkové šetření v prostředí webu (angl. Computer Assisted Web Interviewing)
CS	Počítačová věda – Informatika (angl. Computer Science)
CSTA	Asociace učitelů informatiky (angl. Computer Science Teachers Association)
GAČR	Grantová agentura České republiky
GEG	Google Educator Group
ICT	Informační a komunikační technologie (angl. Information and Communication Technology)
ISTE	Mezinárodní společnost pro technologie ve vzdělávání (angl. International Society for Technology in Education)
MŠMT ČR	Ministerstvo školství, mládeže a tělovýchovy České republiky
RPG	Hra na hrdiny (angl. role-playing game)
RVP	Rámcový vzdělávací program
SDV	Strategie digitálního vzdělávání do roku 2020
ŠVP	Školní vzdělávací program
TED	Technology, Entertainment, Design
UK	Univerzita Karlova
ZŠ	Základní škola
ZÚ	Zájmový útvar
ZV	Základní vzdělávání

Seznam obrázků, grafů a tabulek

Seznam příloh

Seznam elektronických příloh

PŘÍLOHA 13: CD-ROM

- DIPLOMOVÁ PRÁCE (FORMÁT PDF)
- METODICKÁ PŘÍRUČKA (FORMÁT PDF)
- WEBOVÁ STRÁNKA (FORMÁT ZIP)

Příloha 1: Struktura dotazníku – dotazníkové šetření I

Algoritmizace a programování na ZŠ

Dotazník má jen 7 krátkých otázek, vyplnění Vám nezabere víc než 5 minut Vašeho času a je anonymní. Jako odměnu za vyplnění jsou v závěru dotazníku odkazy na výukové webové stránky pro výuku počítačové grafiky a výuku programování (v přípravě).
Dotazník je určen ke sběru informací o výuce programování na základních školách.
Analýza informací z dotazníku bude součástí mé diplomové práce na KITTV Pedf UK.

*Povinné pole

1. **Název školy**

2. **Hrubý odhad počtu žáků školy** *

3. **Vyučujete na škole algoritmizaci a programování?** *

Označte jen jednu elipsu.

- ◯ Ano. (v jakékoliv formě) *Přeskočte na otázku 4.*
- ◯ Ne. (v žádné formě) *Přeskočte na otázku 9.*

4. **Jak často vyučujete algoritmizaci a programování a v rámci jakého předmětu?** *

Zaškrtněte všechny platné možnosti.

- ☐ Ano, ale jen okrajově (1 – 5 hodin) v rámci povinného předmětu ICT.
- ☐ Ano, nejméně 5 hodin v rámci povinného předmětu ICT.
- ☐ Ano, ale jen okrajově (1 – 5 hodin) v rámci jiného povinného nebo volitelného předmětu.
- ☐ Ano, nejméně 5 hodin v rámci jiného povinného nebo volitelného předmětu.
- ☐ Ano, v kroužku programování
- ☐ Jiné: _____

Přeskočte na otázku 5.

5. **1. Uveďte hrubý odhad celkového počtu hodin, které věnujete výuce algoritmizace a programování.** *

6. **2. V jaké třídě se výuce programování věnujete? Jaké prostředí preferujete?** *

Zaškrtněte všechny platné možnosti.

	rozvíjení algoritmického myšlení bez programování, bez počítače – unplugged aktivity (Bobřík informatiky aj.)	popis a tvorba algoritmu (slovně, vývojové diagramy aj.)	"block-based" programování ve vizuálních programovacích jazycích (Scratch, Snap!, Blockly aj.)	programování ve 3D vizuálních programovacích jazycích (Kodu Game Lab, Alice aj.)	textové "turtle" programování s vizuální zpětnou vazbou příkazů (LOGO, Imagine LOGO aj.)	textové programování v programovacích jazycích (Python, Pascal/Delphi, C/C++, C#, Java aj.)	programování robotů (BEEBOT, Ozobot, LEGO WeDo, Mindstorms aj.)	programování jednočipových počítačů (Arduino, Raspberry Pi aj.)	kurzy na webu code.org (Hour of Code)	programování ve "hrách" (Lightbot, Hakitzu aj.)	jiný
1.třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
2. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
3.třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
4. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
5. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
6. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
7. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
8. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
9. třída	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
kroužek (1. stupeň)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
kroužek (věk do 12 let)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
kroužek (věk do 15 let)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
jiný	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

7. 3. Uveďte konkrétně, v jakých prostředích s dětmi programujete. Případně, v čem spatřujete výhody oproti jiným prostředím. *

8. 4. Uveďte učebnice či webové stránky, které využíváte při výuce programování. V případě, že nic nevyužíváte, napište prosím "nic". *

Přeskočte na "Závěr."

Děkuji.

Na závěr vyberte hlavní důvody, proč se na škole programování nevyučuje.

9. Jaké jsou hlavní důvody? *

10. Souhlasíte s tímto stavem?

Označte jen jednu elipsu.

- ◯ Ano.
- ◯ Ne.
- ◯ Nemám čas to řešit.
- ◯ Jiné: _____

Přeskočte na "Závěr."

Závěr

Velmi Vám děkuji za poskytnuté informace.

Jako skromnou odměnu za vyplnění dotazníku Vám nabízím odkaz na autorské webové stránky, které se týkají výuky počítačové grafiky na ZŠ: <http://3zskadan.cz/grafika>. Jedním z výstupů závěrečné práce budou webové stránky zaměřené na výuku programování, které vznikají na <http://3zskadan.cz/prgm>. Přeji Vám hezký den. S pozdravem Bc. Radek Vinický

Příloha 2: Důvody opomíjení výuky programování na základních školách

„Není dostatečné technické vybavení.“

„Pro některé děti náročné.“

„Nízká dotace, jsou důležitější oblasti - základy IT“

„Výuka se zaměřuje především na práci s PC jako uživatele. Přirovnal bych to k situaci, kdy na to, abych získal řidičský průkaz, tak si musím umět sám vyrobit a seřídit auto.“

„Polovina má problém uživatelsky zvládnout programy Office a užití funkcí a výpočtů v Excelu, neumím si představit, jak budu učit žáky s IVP grafický zápis algoritmu a povely progr. jazyka např. Pascal na výpočet obvodu kruhu, programátorem se stane 1 z okresu, nač to na ZŠ učit všechny. Tím ať se zabývají odborné školy a gymnázia, nedokážeme děti učit řádně matematice a ještě tohle, jejich zájem je spíše sociální sítě mobil hry a užít si to, ne přijít na to, jak to udělat… těch je málo.“

„Nízká časová dotace na velmi široké spektrum toho, co spadá pod IT.“

„1 hodina informatiky na 1. i 2. stupni v rámci RVP. I po dotaci v rámci disponibilních hodin nedává smysl vyučovat programování s dotací 1 hodina týdně.“

„Dáváme přednost jiným dovednostem: psaní 10 prsty, grafice.“

„Máme na II. stupni výuku ICT pouze v 6. a 9. ročníku. Tím pádem na algoritmizaci a programování už není čas.“

„Jiné priority v rámci IKT; časová roztříštěnost (školní projekty, šest různých vyučovaných předmětů, spoluodpovědnost za ŠVP, dvě malé děti a spousta dalších aktivit).“

„Řešíme problémy se zvládnutím rozhodovacích i jiných funkcí Excelu, tak i se znalostmi matematiky a fyziky.“

„Plánujeme v příštím školním roce.“

„Náročnost takové výuky, žáci ještě nejsou dostatečně vyspělí.“

„Sama o tom nic moc nevím.“

„Málo času v rámci tematického plánu zařadit i toto téma.“

„Obsah RVP je na 2 h informatiky za 2 roky tak nabitý, že na to není prostor.“

„Úroveň žáků.“

„Nízká časová dotace na výuku informatiky.“

„Zastaralé vybavení ICT technikou, programové vybavení.“

„Není na to v tuto chvíli vhodná doba vzhledem k revizi ŠVP.“

„Informatika ma v dnesni dobe uplne jinou napln. Tim nesnizuji prinosnost umeni programovani, ale dnes asi spise jako nejaky povinne volitelny predmet.“

„Týdenní dotace 1h 5. ročníky, 1h 6. ročníky!!!“

„30 žáků na hodině.“

„Nízká hodinová dotace, kdy se zvládá základ.“

„Časová náročnost s ohledem na přidělenou hodinovou dotaci v souvislosti s dalším obsahem ŠVP.“

„Programy, které znám jsou buď moc lehké, nebo příliš těžké. Nic postupného.“

„Máme třídy, kde takové téma nemá co dělat - sláva inkluzi.“

„Nelze smysluplně vyučovat ve skupině s 25 žáky, na dělení nejsou ani učebny, ani vyučující.“

„Informatika má málo prostoru, není čas.“

Příloha 3: Postoj respondentů k aktuálnímu stavu

Opomíjení výuky algoritmizace a programování

„Pokud by se ukázalo jako prospěšné, ráda absolvuji školní, a budu předávat dále žákům.“

„Žáci nezvládají základní úkony s pc. Pro základní školu je nejdůležitější, aby žák uměl s textovými editory, tabulkovými, prezentaci udělat apod., že počítač není jen pro youtube a on-line hry. Nicméně naše škola nabízí kroužek "Mediální výchovu", ve které se žáci učí základy grafiky.“

„Učím na 1. stupni, proto jsem o tom vůbec neuvažovala.“

„Myslím si, že pro ZŠ je to nadstavba, zařadit jako nepovinný předmět pro IT nadané žáky.“

„Díky neaprobovanosti se necítím kompetentní k odpovědi.“

„Ano i ne, je potřeba zájem ze strany žáků.“

„Nezamýšlela jsem se nad tím, nejsem aprobovaná a jsem ráda, že zvládám, co předávám.“

„Časová dotace 1 h/týdně je strašně málo (ještě jich x odpadne).“

„Výuka informatiky probíhá v 5. a 6. ročníku, hodina týdně.“

„V ZŠ je to oblast pro velmi úzký okruh žáků.“

„Částečně, algoritmizace by se dala zařadit, ale plošně to není vhodné, já se např. pokoušel o lehké programování www a radši jsme to nechali (nezájem, těžké).“

„V případě odborně vybaveného kolegy v rámci volitelných předmětů jsem pro.“

„Na jednu stranu bych výuku uvítal, ale organizačně je to velmi náročné a není, kdo by toto učil.“

„V dané časové dotaci na předmět informatika mi to nepřijde reálné.“

„Programování bych zařadil jen na vybrané (specializované) ZŠ.“

„Jen velmi malá část žáků má potřebné předpoklady, těm doporučujeme individuální vzdělávání ve specializovaných kurzech. 1 hod. týdně stačí sotva tak na získání potřebných základů práce s PC a obsluhu jednoduchých editorů.“

Příloha 4: Seznam učebnic využívaných respondenty pro podporu výuky algoritmizace a programování

Název (autor)	Počet výskytů
Informatika I, II, III (Jiří Vaníček)	6
Informatika pro 1. stupeň ZŠ (Jiří Vaníček)	3
Informatika pro ZŠ (Libuše Kovářová)	1
Moderní programování (Radek Vystavěl)	1
Imagine Logo (A. Blaho, P. Tomcsanyi, I. Kalaš)	1
S počítačem nejen k maturitě (Pavel Navrátil)	1
Příklady a cvičení z informatiky a VT (Pavel Navrátil)	1
Informatika pro základní školy (autor neuveden)	1

Příloha 5: Seznam zdrojů, které dotazovaní učitelé při výuce algoritmizace a programování preferují

Název webu	Počet výskytů
code.org	29
hourofcode.org	15
scratch.mit.edu	14
ibobr.cz	4
ozoblockly.com	4
karel.oldium.net	3
umimeprogramovat.cz	3
codecombat.com	2
microla.cz	2
kodugamelab.com	2
jakpsatweb.cz	2
ozobot.sandofky.cz	2
youtube.com	2
blockly-games.appspot.com	2
sgpsys.com (SPG.cz)	2
blockly	1
playcodemonkey.com	1
pc-hratky.webnode.cz	1
sgpsys.com (SPG.cz)	2
robomise.cz	1
microbit.org	1
techsoup.cz	1
lego.com/bits-and-bricks	1
programovaniprodeti.cz	1
itnetwork.cz	1
tib.cz	1
mblock.cc	1
allcancode.com/web	1
lightbot.com	1
codeacademy.com	1
zdrojak.cz	1
ozobot.com	1
cojsemvyzskousela.cz	1
logo a baltík	1
GEGy a facebookové skupiny (materiály)	1
alice.org	1
ict-2.webnode.cz/a9-rocnik/	1
www.zsstraz.cz/index.php?p=516	1
blíže nespecifikované weby, vlastní materiály, aj.	20

Příloha 6: Tabulka zdrojů

Zdroj	Počet výskytů
Scratch	**8**
Code.org	**6**
Kodu	**4**
Blockly a Blockly Games	**3**
Code Combat	**3**
CS Unplugged	**3**
Hour of Code	**3**
Swift Playground	**3**
Tynker	**3**
Alice	**2**
App Inventor	**2**
Code Monster	**2**
Google CS First	**2**
Google Made with Code	**2**
Lightbot	**2**
Robot Karel	**2**
Stencyl	**2**
BotLogic	1
Black Girls Code	1
CargoBot	1
Code Avengers	1
Code Kingdoms	1
Code Monkey	1
CoderDojo	1
CoderZ	1
Comenius Logo	1
Hakitzu	1
Hopscotch	1
Imagine LOGO	1
Khan Academy	1
Kodable	1
MIT App Inventor	1
Mozilla Thimble	1
Pencil Code Gym	1
Snap!	1
Thunkable	1
Turtle Academy	1
Turtle Art	1
Turtle Pond	1

Příloha 7: Nejčastěji zmiňované online zdroje

Nezkrácená verze

Prostředek	Výskyt	Stručný popis	Lokalizace češtiny
Scratch	8	Scratch je zřejmě světově nejznámějším online prostředkem pro výuku základů programování s více než 32 miliony registrovaných uživatelů (Scratch, 2018). Využívá block-based vizuálního programování. Scratch byl vyvitut na Massachusetts Institut of Technology (dále jen MIT) skupinou Lifelong Kindergarden – MIT Media Lab v roce 2007 a jeho puzzle bloky, pomocí kterých je program tvořen, se staly průlomovými v oblasti výuky programování zaměřené především na děti. Vedoucím týmu a duchovním otcem Scratche je Mitchel Resnick – mimo jiné žák Seymoura Paperta. Resnick charakterizuje Scratch jako zdroj umožňující uživatelům Scratche – *„Scratcherům“* vytvářet interaktivní příběhy, animace, simulace i hry a následně sdílet své výtvory s komunitou po celém světě. Ve Scratchi i dalších block-based programovacích jazycích se počítačový program tvoří podobně, jako se staví konstrukce z kostek Lega. Žáci se nemusejí obávat takových věcí, které s sebou obvykle přináší učení se programovacím jazykům – *„kam dát hranaté závorky nebo středník,“* podotýká Resnick v rozhovoru dostupném na youtube [cit. 2018-10-15].	ANO
Code.org (Studio)	6	Webové stránky neziskové organizace Code.org patří mezi špičku v oblasti výuky algoritmizace a základů programování pro děti v celosvětovém měřítku. Edukátorům nabízí rozmanité možnosti využití prostředků webu od jednohodinové akce (Hour of Code) po ucelené celoroční kurzy. Na webu je dostupná kurikulární dokumentace pro kurzy i lekce, hodinové plány i výsledky správných odpovědí. Code.org Studio představuje řadu 4 hlavních kurzů odlišné náročnosti cílící na různé věkové kategorie. Současně v kurzech kombinuje výuku na počítači s unplugged metodami výuky. Velmi zajímavé nástroje poskytuje učitelský účet, pod kterým lze vytvořit a spravovat třídy, pravidelně kontrolovat progres svých žáků v reálném čase, dokonce učiteli umožňuje tvorbu vlastních kurzů.	ANO

Kodu	4	Kodu (Kodu Game Lab) od společnosti Microsoft je vizuálně programovatelný, zdarma dostupný engine pro tvorbu her v atraktivním 3D prostředí, který se soustředí na děti toužící programovat. Samotné programování je realizováno skládáním grafických ikonek. Programovací logika je založena na podmínce if–then, v Kodu reprezentovanou jako when–do. Dále umožňuje vytvořit nekonečnou smyčku (ve hře) always–do. Kodu navíc nabízí možnost překladu složeného kódu do jazyka C#. Instalační soubor je možné stáhnout na webu <https://www.kodugamelab.com/>. Kromě možnosti stažení jsou zde k dispozici především hodinové plány a metodická dokumentace i tzv. Kodu Worlds, tedy světy vytvořené jinými uživateli, které je možno stáhnout a v enginu vyzkoušet, prozkoumat a remixovat. Češtinu do Kodu lze stáhnout z webu autora překladu <https://preklady.buchtic.net>.	ANO
Blockly (Blockly Games)	3	Společnost Google na webu Google for Education <https://developers.google.com/blockly/guides/overview> [cit. 2018-10-15] představuje Blockly jako open source knihovnu, která umožňuje přidávat block-based programovací nástroj do webových a mobilních aplikací, a tak využít všech kladů vizuálního programování. Dokáže reprezentovat koncepty jako proměnná, příkaz, cyklus a další. Blockly navíc umožňuje překlad sestaveného programu do jazyků JavaScript, Python, PHP, Lua a Dart. Blockly je do značné míry využíváno jako programovacího nástroje v online vzdělávacích projektech, například na webu organizace Code.org i Hour of Code, v App Inventor, Made with Code nebo při programování robůtka Ozobota na <https://ozoblockly.com/>. Pro žáky, jako začínající programátory, jsou také připraveny webové stránky Blockly Apps <http://seriousgames.lip6.fr/blockly/apps/> či známější Blockly Games <https://blockly-games.appspot.com> pro úplné začátečníky, dostupné zdarma.	ANO
Code Combat	3	Code Combat je webovou platformou, která umožňuje uživateli vybrat si k výuce herní prostředí. K výběru jsou jazyky Python nebo Javascript. Zdarma je poskytnuta pouze první herní mapa (Kobka Kithgardu) s 42 úrovněmi. Vývojáři uvádějí herní dobu mapy na jednu až tři hodiny. Uživatel je v této mapě seznamován se základní syntaxí, cykly, proměnnou, funkcí a parametry funkce nebo s řetězci. To vše formou RPG. Po úspěšném dokončení úrovně přechází hráč na další – složitější úroveň. Hra podporuje češtinu.	ANO
CS Unplugged	3	CS Unplugged představuje možnost výuky informatiky bez počítače formou různých výukových aktivit využívajících herní prvky. Web <https://csunplugged.org> disponuje několika aktivitami, které je možné ve výuce využívat. Ke všem aktivitám je navíc k dispozici velmi podrobná metodika, včetně videí a materiálů k vytištění. Web představuje v pěti hlavních tématech úvod do problematiky – binární soustavy, programování (robotů), algoritmů, sítí, a detekce a opravy chyb, a to hravou formou vhodnou pro děti.	NE

Hour of Code	3	Akce organiace Code.org se zaměřuje na propagaci dětského programování v rámci jedné vyučovací hodiny. Web akce <https://hourofcode.com/cz> poskytuje zdarma značné množství herních aktivit v různých podobách a využívající různých prostředků, například Blockly, Tynker, JavaScript, Python, Scratch, Java, atd.	ANO
Swift Playgrounds	3	Swift Playgrounds je aplikace společnosti Apple pro iPad (pro běh vyžaduje iOS 10). Aplikace je dostupná zdarma na App Store. Výuka probíhá v programovacím jazyku Swift ve třech kurzech nazvaných Learn to Code1, 2 a 3, jejichž náročnost postupně roste. Kurzy seznamují děti se základními koncepty programování – s příkazy a sekvencemi příkazů, debuggingem, podmínkami a logickými operátory, řídícími strukturami, proměnnými, funkcemi, parametry, poli a dalšími. Jedná se o povedený projekt Applu, jehož nevýhodami v českých podmínkách je jazyková lokalizace (podpora 6 jazyků, ale ne češtiny), nutnost vlastnit iPad a také výuka programovacího jazyka, který je výhradně určený pro produkty Apple.	NE
Tynker	3	Tynker je velmi povedené, Scratchi podobné prostředí společnosti Neuron Fuel, jež podporuje výuku dovedností v programování. Nabízí velké množství kurzů různé obtížnosti pro žáky různých věkových kategorií a dovedností. Jeho nevýhodou je však absence jazykových mutací (dostupné je jen v angličtině). Také nutnost platby za plnou funkčnost všech nabízených nástrojů (například přístup k privátnímu serveru hry Minecraft) či prémiovým lekcím (kompletní knihovně) brání v masivní implementaci Tynkeru do výuky na českých školách. Vzhledem k prostředkům, které do jisté míry převyšují Scratch (například okamžitého překladu do JavaScriptu), má Tynker potenciál Scratch předčit.	NE
App Inventor	3	App Inventor je online block-based (Blockly) programovací prostředí z dílny MIT zaměřené na tvorbu mobilních aplikací s operačním systémem Android. Web <http://appinventor.mit.edu> nabízí tutoriály pro začátečníky i mírně pokročilé, metodickou dokumentaci pro učitele. Odkazuje také na MOOC Coursera kurz, který je věnovaný právě tvorbě aplikací v App Inventoru. Samotné prostředí je dostupné na webu <http://ai2.appinventor.mit.edu>. Vytvořenou aplikaci je možné sledovat přímo na mobilním zařízení nebo v emulátoru Androidu v počítači. App Inventor je k dispozici zdarma.	NE
Alice	2	Multiplatformní software pro Windows, Linux a Mac OS, který umožňuje výuku programování ve 3D prostředí. Projekt je reakcí na snahu učitelů vytvořit prostředí, které by sloužilo pro výuku objektově orientovaného jazyka Java, což především poslední verze programu Alice 3 splňuje, neboť je s Javou plně kompatibilní. Neumožňuje však ani dědičnost, ani polymofismus. Využití na základní škole pro výuku programování začátečníků je značně diskutabilní, přihlédneme-li k obtížnosti tvorby i faktu, že je celé prostředí v anglickém jazyce. Starší Alice 2 i nová 3 jsou zdarma ke stažení na webu projeku Carnegie Mellon University <https://www.alice.org/get-alice/> [cit. 2018-10-15].	NE

Code Monster	2	Code Moster je zdarma dostupný online nástroj pro výuku a procvičování syntaxe jazyka JavaScript. Žák má k dispozici celkem 59 lekcí, ve kterých plní požadavky monstra na změnu kódu. Dobrou zpětnou vazbou je okamžitá vizuální reakce na změnu kódu. Code Monster je podle webu <codewizardshq.com> vhodný pro děti od třinácti let.	NE
Google CS First	2	CS First ve všech projektech využívá k výuce prostředí Scratch. Jednotlivá témata jsou k dispozici v podobě Scratch studií, které obsahují různé projekty od CS First, vhodné k remixu. Další informace o CS First byly popsány výše.	NE
Google Made with Code	2	Made w/ Code (Made with Code) je genderově zaměřená iniciativa společnosti Google, jež se soustředí na přiblížení a zatraktivnění programovaní dívkám. Web nabízí výuku programování v 19 projektech (například Music Mixer, Avatar, Zalévací robot, Wonder Woman a další), ve kterých je využíváno vizuální programování pomocí bloků, konkrétně Blockly. Bohužel, Made with Code postrádá českou lokalizaci.	NE
Lightbot	2	Lightbot je další z platforem, kterou je možné využívat při výuce základů programování pro děti. Integruje zábavnou hru s technikami programování, učebními principy a pojmy. Lightbot byl vytvořen Dannym Yaroslavskim pro neziskovou organizaci Code.org. Lightbot se nesoustředí na konkrétní vyšší jazyk (programuje se pomocí ikonek), zaměřuje se na principy abstraktního myšlení programování, které by mohly být v začátcích pro děti obtížně pochopitelné a které jsou běžné v různých vyšších programovacích jazycích. Ve všech úrovních hry nutí dítě přemýšlet nad vhodným postupem a tvořit jen efektivní program s eliminací redundantních příkazů, a to tak, že umožňuje vkládat pouze omezený počet ikonek. Současně jim umožňuje vidět chování robota plnícího instrukce programu, což představuje okamžitou zpětnou vazbu s možností kvalitního odstraňování chyb v programu. Na webu <http://lightbot.com/hoclearn.html> [cit. 2018-10-07] je učitelům představena možnost pro starší nebo nadanější děti, při které si mohou své programy vytvořené pomocí ikon transformovat do slovních instrukcí.	ANO
Robot Karel	2	Česká online verze prostředí a programovacího jazyka určeného pro výuku dětí – Karel the Robot. Karel umožňuje vytvářet nové příkazy, které se kombinují ze čtyř původních nebo již vytvořených.	ANO
Stencyl	2	Software, který se zaměřuje na tvorbu her pomocí block-based programovacího jazyka. Hry je možné vyvíjet pro platformy iOS, Android, Windows, Mac, Linux, Flash a HTML5. Pro práci v programu je však nutná insalace. Základní verze je k dispozici zdarma ke stažení na oficiálním webu <http://www.stencyl.com>.	NE

Příloha 8: Analýza zdrojů podle SimilarWeb

Doména[14]	Celkem	Měsíčně	Celkem IP	Čas [mm:ss]	Bounce rate [%]
scratch.mit.edu	43800000	14600000	6300000	08:32	39,86
code.org	12800000	4255000	1700000	12:32	33,50
ai2.appinventor.mit.edu (MIT App Inventor)	5330000	1777000	524773	08:16	23,04
tynker.com	3050000	1017000	448251	06:47	47,33
csfirst.withgoogle.com	1930000	642105	314581	03:09	51,65
codecombat.com	2500000	817482	293829	08:45	52,29
thunkable.com	3145000	1048000	286496	10:42	18,93
hourofcode.com	1100000	380956	168272	02:40	51,58
stencyl.com	818619	272873	108350	03:22	53,64
thimble.mozilla.org	749435	249812	105696	04:07	45,19
playcodemonkey.com	593876	197959	80997	08:26	42,08
snap.berkeley.edu	837433	279191	78669	03:21	53,93
codeavengers.com	576979	192326	77370	06:56	44,36
lightbot.com	468600	156208	69350	01:49	45,07
blockly-games.appspot.com	537485	179162	66736	10:47	31,89
coderdojo.com	304243	123481	52411	01:34	57,18
alice.org	347128	115709	51036	02:05	54,20
madewithcode.com	284926	94975	47516	04:03	43,45
kodable.com	325769	108590	44052	04:28	28,72
codekingdoms.com	335501	111834	42449	02:58	43,78
appinventor.org (Google)	315567	105189	40239	02:29	49,48
gethopscotch.com	332433	110811	38661	03:21	53,93
csunplugged.org	248812	82937	35815	02:41	41,99
kodugamelab.com	183004	61001	26570	02:10	53,09
turtleacademy.com	108114	36038	12166	02:57	44,58
crunchzilla.com (Code Monster)	85375	28458	11184	00:55	51,77
umimeprogramovat.cz	29792	9931	< 5000	04:11	58,40
gym.pencilcode.net	27981	9327	< 5000	01:56	29,16
ftc.coderzworld.com (CoderZ)	8696	6379	<5000	05:00	-
botlogic.us	18987	6329	< 5000	01:52	24,00
blackgirlscode.org	< 5000	< 5000	< 5000	00:11	29,98
ibobr.cz	6910	< 5000	< 5000	12:59	30,15
turtleart.org	9937	< 5000	< 5000	01:43	31,26
karel.oldium.net	4700	< 5000	< 5000	01:40	55,77

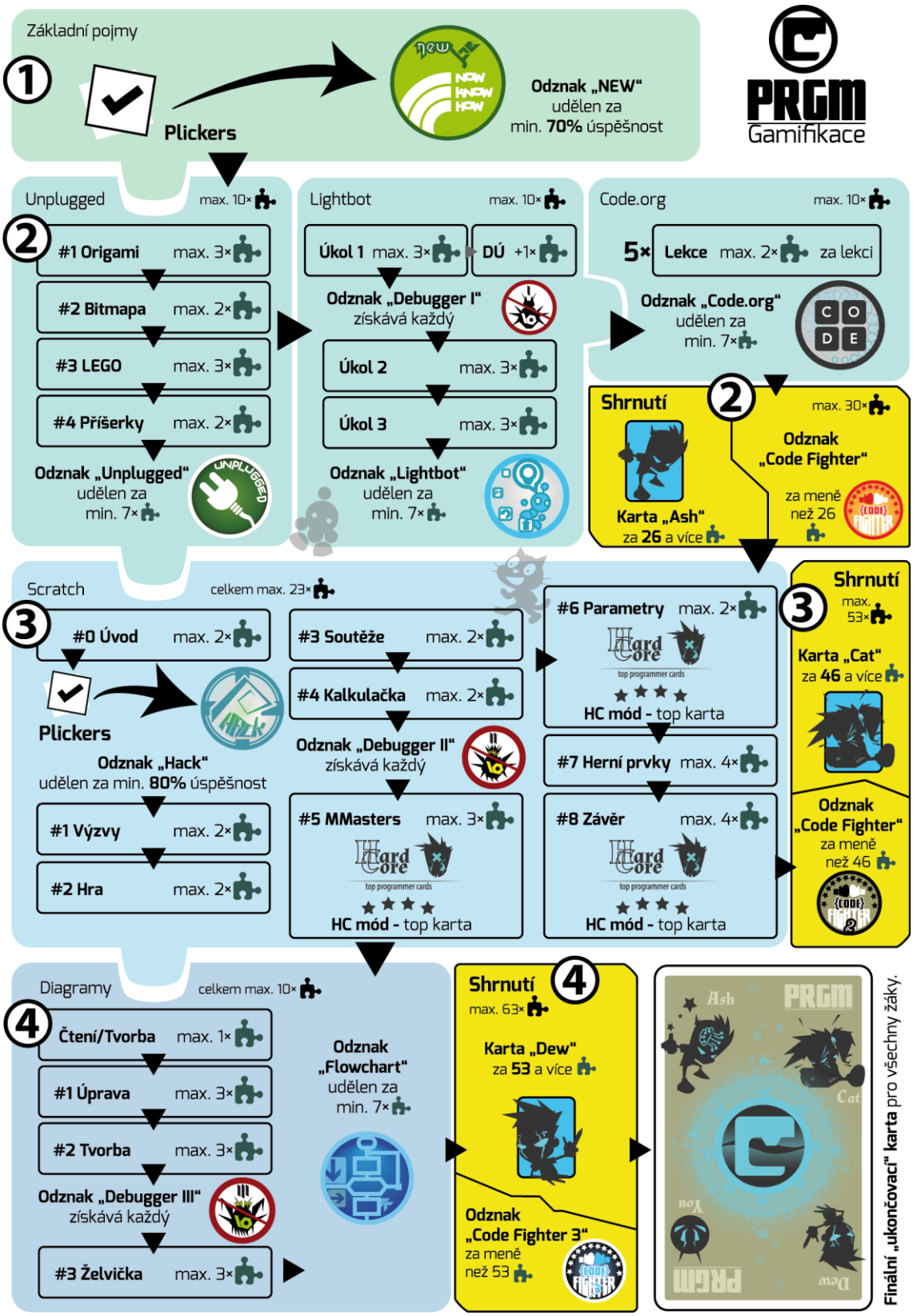
[14] Nelze analyzovat Swift Playgrounds, Hakitzu, CargoBot, Imagine LOGO, Turtle Pond, Comenius Logo, Khan Academy

Příloha 9: Časově-tematický plán modelu

Blok	Očekávané výstupy	Učivo	Hodinová dotace
1. blok	Žák pochopí, co jsou to algoritmy a zná souvislost mezi algoritmem a programem	Pojem algoritmus vlastnosti algoritmu Pojem program, programování a programovací jazyk	1
2. blok – unplugged, Lightbot, Code.org	Žák pochopí důležitost přesných a jednoznačných pokynů v programu Žák využívá informatického myšlení v aktivitách bez počítače	Unplugged výuka algoritmizace	2
	Žák rozpozná opakující se vzory, používá opakování a připravené podprogramy	Lightbot – řešení průchodu bludištěm Pojem podprogram (procedura)	4
	Žák navrhne, napíše a odladí programy splňující konkrétní požadavky. Žák v blokově orientovaném jazyce sestaví program, ve kterém využívá sekvence, funkce (procedury) a vlastní funkce včetně parametrů funkcí a cyklů	Pojem proměnná Pojem parametr funkce Sekvence příkazů (Blockly)	1
	Žák rozpozná opakující se vzory, používá opakování a připravené podprogramy Žák upraví připravený postup pro obdobný problém; ověří správnost jím navrženého postupu, najde a opraví v něm případnou chybu.	Výběr z kurzu č. 4 Studia Code.org: - L3. umělec - L6. umělec: proměnná - L10. umělec: for cykly - L12. malíř: funkce - L14. umělec: funkce s parametry	5

3. blok - Scratch	Žák navrhne, napíše a odladí programy splňující konkrétní požadavky. Žák chápe základy Booleovy logiky – chápe význam AND, OR, NOT a jejich využití v programování. Žák v blokově orientovaném jazyce sestaví program, ve kterém využívá sekvence, funkce (procedury) a vlastní funkce včetně parametrů funkcí, cyklů a náhodných čísel. Žák vybere vhodný algoritmus z více alternativ, které řeší stejný problém. Žák řeší problémy rozkladem do menších částí. Žák rozpozná opakující se vzory, používá opakování a připravené podprogramy; používá události ke spouštění podprogramů	Účet na scratch.mit.edu, UI Scratche Pojmy proměnná, cyklus, spojování řetězců, rozhodování Multibloky – seznámení	2
		Tvorba jednoduchých programů (předpřipravené bloky). Nový blok, událost, náhodné číslo	1
		Herní prvky a tvorba hry (předpřipravené bloky)	2
		Tvorba soutěží založených na náhodě	2
		Kalkulačka	1
		Hry s čísly - odpočet - sudá/lichá	3
		Math Master (NADANÍ ŽÁCI)	N
		Podprogramy a parametry - n-úhelníky - rekurze	2
		Rekurze (NADANÍ ŽÁCI)	N
		Herní prvky – tvorba funkční hry - změna hráče - paralax - skok a hudba - ninja pohyb - kolize, ukončení hry - další level	4
		Komplexní projekt	3
4. blok – PS Diagram	Žák popíše jednoduchý problém, navrhne a popíše jednotlivé kroky jeho řešení. Žák vybere vhodný algoritmus z více alternativ, které řeší stejný problém. Žák řeší problémy rozkladem do menších částí. Žák upraví připravený postup pro obdobný problém; ověří správnost jím navrženého postupu, najde a opraví v něm případnou chybu.	Čtení a tvorba vývojového diagramu	1
		Úprava vývojového diagramu	1
		Porovnávání čísel – vývojový diagram vedoucí k řešení	1
		Návrh řešení želví grafiky vývojovým diagramem, ověření ve Scratchi/Code.org	1

Příloha 10: Gamifikační plán odměn



Příloha 11: Závěrečný dotazník – Dotazníkové šetření II

Tvůj věk: _____

1. Který ze čtyř bloků tě zaujal nejvíce? Proč?

Který: _____ Proč: _____

_____

2. Který z bloků (kromě prvního) tě zaujal nejméně? Proč?

Který: _____ Proč: _____

_____

3. Která konkrétní aktivita tě bavila nejvíce?

Která: _____ Proč: _____

_____

4. Která konkrétní aktivita tě bavila nejméně (nebavila)? Proč?

Která: _____ Proč: _____

_____

5. Zakroužkuj, který z důvodů ti nejvíce bránil v dokončení práce (funkčních programů ve Scratchi, na Code.org, úrovní Lightbota atd.)?

žádný čas obtížnost úkolu nepochopení zadání

nějaký jiný: _____

_____

6. Které z prostředí, programů nebo stavebnic (v závěru výuky) tě zaujaly natolik, že bys v **nich/s nimi rád pracoval?**

_____

7. Zakroužkuj. Pracoval jsi na některých školních projektech doma?

ANO NE

8. Budeš v programování pokračovat i po ukončení kroužku?

_____

_____

9. Byl(a) jsi s náplní práce spokojen(a)? Splnil kroužek tvé představy?

_____

_____

10. Zakroužkuj. Odradila tě účast v kroužku od dalšího programování? Pokud ano, proč?

ANO NE NEVÍM

Proč:

_____

_____

11. Zakroužkuj. Vyhovovala ti práce ve dvojici (trojici)?

ANO NE JINÉ: _____

12. Zakroužkuj a popiš. Vyhovoval ti systém hodnocení – odměn?

ANO NE

Popis: _____

_____

Děkuji za vyplnění.

Příloha 12: Odznaky a karty

Ash

Schopnosti

PRGM

A

PRGM

H

Dew

PRGM

You

Cat

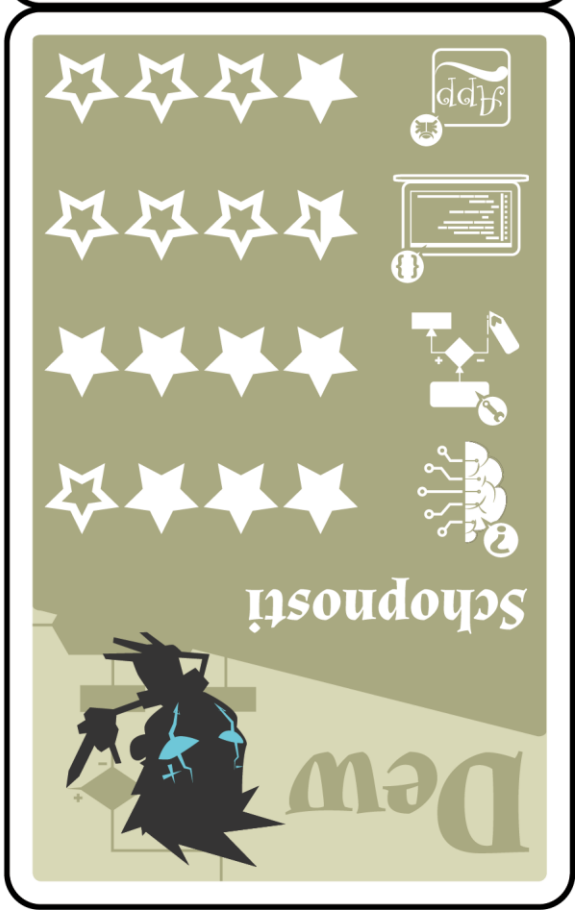
Ash

PRGM


C
SCRATCH
PRGM


D
x1<=1
PRGM


isonqobS


schoposti

Hard Core

Schopnosti

App

Hard Core

Schopnosti

App

HARD CORE

PROGRAMMER

HARD CORE

PROGRAMMER


Hard
Core
Schopnosti
App
Hard
Core
Schopnosti
App
HARD
CORE
PROGRAMMER
HARD
CORE
PROGRAMMER