Univerzita Hradec Králové

Přírodovědecká fakulta

Katedra kybernetiky

Dětské programovací jazyky ve výuce informatiky

Diplomová práce

Autor: Bc. Michaela Kunhartová

Studijní program: N 1101 Matematika

Studijní obor: Učitelství pro střední školy - informatika

Učitelství matematiky pro střední školy

Vedoucí práce: PhDr. Musílek Michal, Ph.D.

Hradec Králové 12. dubna 2017

Univerzita Hradec Králové

Přírodovědecká fakulta

Zadání bakalářské práce

Autor:	Bc. Michaela Kunhartová
Studijní program:	N 1101 Matematika
Studijní obor:	Učitelství pro střední školy - informatika
	Učitelství matematiky pro střední školy
Název práce:	Dětské programovací jazyky ve výuce informatiky
Název práce v AJ:	Children's Programing Languages in Teaching of Informatics
Garantující pracoviště:	katedra kybernetiky Přírodovědecké fakulty UHK
Vedoucí práce:	PhDr. Musílek Michal, Ph.D.
Oponent:	doc. RNDr. Štěpán Hubálovský, Ph.D.
Datum zadání práce:	26. 11. 2015
Datum odevzdání práce:	12. dubna 2017

Prohlášení:

Prohlašuji, že jsem diplomovou práci vypracovala samostatně a že jsem v seznamu použité literatury uvedla všechny prameny, z kterých jsem vycházela.

V Hradci Králové dne Michaela Kunhartová

Poděkování:

Ráda bych poděkovala panu. Michalu Musílkovi za odborné vedení, vstřícnost, trpělivost a ochotu, s jakou se mi věnoval v průběhu zpracování mé diplomové práce.

Anotace

KUNHARTOVÁ, M. *Dětské programovací jazyky ve výuce informatiky.* Hradec Králové, 2017. Diplomová práce na Přírodovědecké fakultě Univerzity Hradec Králové. Vedoucí diplomové práce Michal Musílek. 81 s.

Práce formou literární rešerše poskytuje základní informace o dětských programovacích jazycích (např. LOGO, SCRATCH) a o možnostech rozvoje logického myšlení žáků během výuky, s důrazem na výběr vhodných úloh. Rešerše je doplněna základním popisem vývojových prostředí a nástrojů, které lze při práci s dětskými programovacími jazyky využívat. Praktické části představuje soubor řešených algoritmických úloh, vhodných k procvičování logického a algoritmického myšlení a zaměřených na různá témata. Součástí je i didaktický popis jednotlivých úloh a procesu jejich řešení, včetně stanovení výukových cílů a návrhů k rozvíjení klíčových kompetencí. Závěrečná empirická část práce ověřuje přínosu řešení vybraných úloh z praktické části se zvolenou skupinou žáků

Klíčová slova:

Dětské programovací jazyky, SCRATCH, výuka programování, výuka informatiky, logické myšlení, algoritmické myšlení

Annotation

KUNHARTOVÁ, M. *Children's Programing Languages in Teaching of Informatics*. Hradec Králové, 2017. Diploma Thesis at Faculty of Science Univerzity of Hradec Králové. Thesis Supervisor Michal Musílek. 81 s.

The theoretical part of the bachelor thesis is a literature review which includes basic information about educational programming language (e. g. LOGO, SCRATCH) opportunities of development logical thinking during lessons with emphasis on choice of suitable tasks. Literature rewiev is going to be completed with basic description of development of environment and tools which we can use during working with educational programming language. Practical part present to develop a set of solved algoritmical problems suitable for practising of logical and alorithmical thinking with focus on a different theme. It includes a description of the individual tasks and process of their solutions, including the definition of learning aims and proposals for the development of key competencies. Empirical part is the verifies of benefits of solutions of selected tasks from the practical part for on the selected group of pupils

Keywords:

Educational programming language, SCRATCH, programming teaching, informatics teaching, logical thinking, alogrithmic thinking

Obsah

Úvod

Ve výuce informatiky stojí před vyučujícím nejeden obtížný úkol. Má žáky nejen naučit určitým teoretickým znalostem a praktickým dovednostem, ale také u svých žáků rozvíjet rozumovou stránku, kultivovat, zušlechťovat jejich způsob myšlení a uvažování. Je vždy těžké najít mezi těmito úkoly rovnováhu a nezaměřit se jen na jednu část. Není vždy snadné obsáhnout ve výuce informatiky všechny tyto aspekty. Jednou z mnoha možností, jak naplnit tyto cíle, je začlenit do výuky informatiky hodiny programování.

Tématem této diplomové práce jsou dětské programovací jazyky a jejich výuka v hodinách informatiky. Jedná se o komplexní téma, ve kterém je ve skutečnosti zahrnuto několik dílčích okruhů. Jednotlivým okruhům se podrobněji věnuji v teoretické části. Prvním okruhem je výuka informatiky podle Rámcového vzdělávacího programu (RVP) z pohledu časové dotace, rozsahu učiva a očekávaných výstupů. Druhým dílčím tématem je již samotná výuka programování. Ta spolu s rozvojem logického myšlení patří mezi významné aspekty všestranného rozvoje žáka. Posledním dílčím tématem v teoretické části jsou mezipředmětové vztahy a jejich využití ve výuce informatiky.

V teoretické části jsou dále uvedeny některé programovací jazyky, které řadíme do skupiny dětských programovacích jazyků. Každému z nich je věnován krátký přehled o jeho historii, motivaci pro vznik daného jazyka, přehled jednotlivých parametrů pro použití ve výuce a nechybí ani didaktický pohled.

V praktické části je představeno programovací prostředí SCRATCH, je zmíněna jeho historie doplněná o krátký popis grafického rozhraní. Následuje sbírka úloh pro tvorbu v programovacím prostředí SCRATCH. Sbírka obsahuje celkem jedenáct úloh, většina z nich je doplněna o možné rozšíření, či vylepšení tvořeného programu. U jednotlivých úloh je uveden název úlohy, časová náročnost, použitá algoritmická konstrukce, výchovně-vzdělávací cíle, rozvíjené klíčové kompetence, tematický celek, zadání pro tvorbu programu, průřezová témata a mezipředmětové vztahy. Autorské řešení úloh je uvedeno v příloze. Empirickou část práce představuje ověření několika vybraných úloh ve výuce. Cílem je posoudit časovou náročnost a obtížnost pro žáky.

Cíle diplomové práce

Cílem teoretické části je zpracování literární rešerše. Ta bude rozčleněna na dvě části. První část je rešerše, která definuje základní pojmy problematiky spojené s výukou informatiky, např. RVP, mezipředmětové vztahy, konstruktivismus a jeho možnosti implementace ve výuce, rozvoj logického a abstraktního myšlení.

Dalším cílem teoretické části je literární rešerše historie programování a programovacích jazyků, Doplněná o přehled v základních pojmech, které souvisejí s programováním, jako jsou pojmy program, větvení programu, typy cyklů a další náležitosti programu.

Praktická část má několik cílů. Jedním z cílů empirické části je krátké představení programovacího prostředí SCRATCH. Toto prostředí bude představeno z pohledu funkcí a parametrů tohoto prostředí SCRATCH, vybraných s ohledem na praktické didaktické využití tohoto prostředí. Druhým cílem praktické části je vytvoření krátké sbírky úloh pro výuku programování prostřednictvím programovacího prostředí SCRATCH.

Cílem empirické části je posoudit časovou náročnost výuky a kvalitativně prozkoumat, které prvky zadané úlohy a pokyny při jejich řešení činí žákům potíže.

1 Národní program rozvoje vzdělávání v České republice

Žijeme v době prudkých a obtížně předvídatelných změn, které se projevují na úrovni politické, společenské, lidské, technické a řadě jiných oblastí. Jedním z klíčových faktorů, který způsobuje tyto změny, je rozvoj informačních a komunikačních technologií. Všechny tyto změny mají nezpochybnitelný vliv na výchovu, vzdělávání a školský systém. Na tyto změny je potřeba reagovat a to v souladu s požadavky Ministerstva školství, mládeže a tělovýchovy (MŠMT). To zveřejnilo dokument „Koncepce vzdělávání a rozvoje vzdělávací soustavy v České republice“. *Ministerstvo se touto koncepcí přihlásilo k zásadě, že rozvoj školství a všech dalších vzdělávacích institucí a aktivit, podílejících se na utváření národní vzdělanosti, se má v budoucnosti vyvozovat z obecně přijatého rámce vzdělávací politiky a jasně vymezených střednědobých a dlouhodobých záměrů, které mají být veřejně vyhlášeny v podobě závazného vládního dokumentu, „Bílé knihy“.* [1]

Tento dokument vychází z mnoha analýz českého školství, z odborných diskuzí pracovníků MŠMT, Ministerstva práce a sociálních věcí (MPSV), České školní inspekce (ČŠI), Rady vysokých škol (RVŠ), České konference rektorů (ČKR) a řady vzdělávacích institucí i jednotlivých pedagogických pracovníků.

Česká Bílá kniha je pojata jako systémový projekt, formulující myšlenková východiska, obecné záměry a rozvojové programy, které mají být směrodatné pro vývoj vzdělávací soustavy ve středně dobém horizontu. [1]

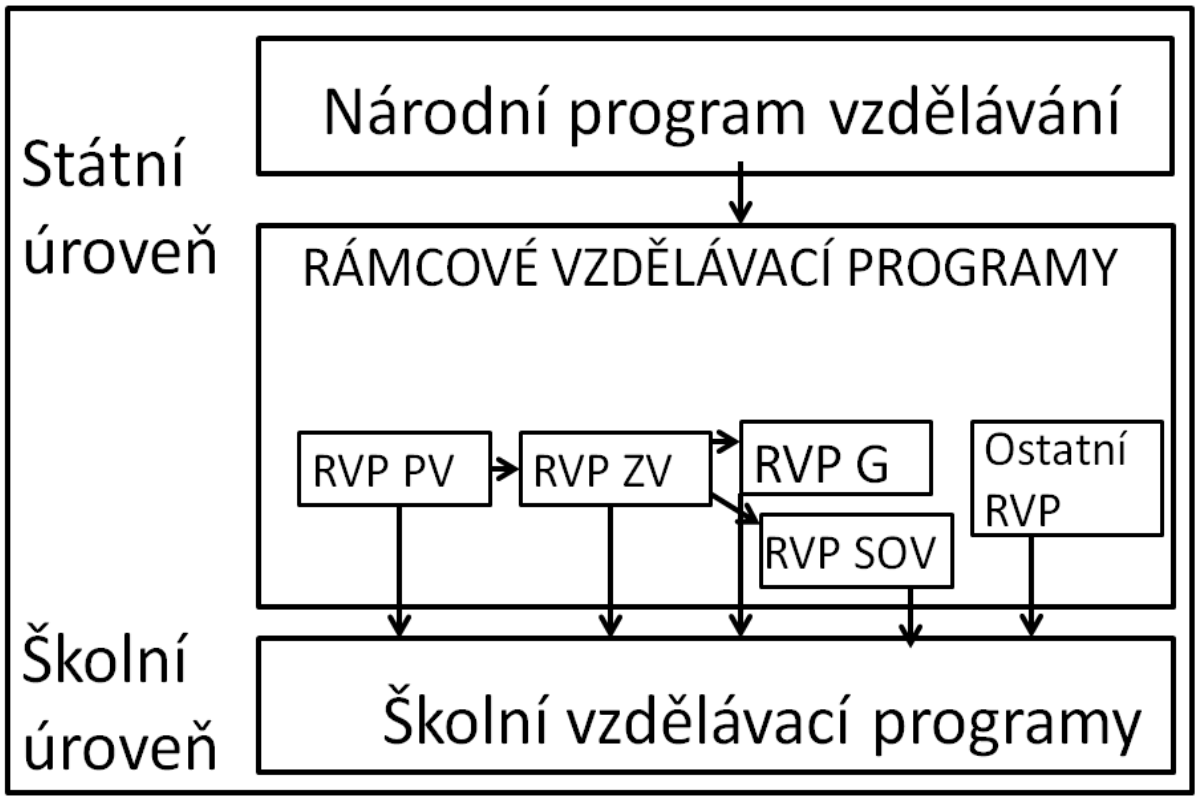
1.1 Kurikulum školy a Rámcové vzdělávací programy

Latinské slovo curriculum lze přeložit několika způsoby. Prvním z nich je běh, druhým je často životopis. V pedagogice chápeme kurikulum jako „*obsah vzdělávání (učivo) v širším slova smyslu a proces jeho osvojování, tj. jako veškerou zkušenost žáka (učícího se), kterou získává ve školském (vzdělávacím) prostředí, a činnosti, které jsou spojeny s jeho osvojováním a hodnocením*“. [2]

Kurikulární dokument je takový pedagogický dokument, který určuje koncepci, cíle a vzdělávací obsah dané etapy vzdělávání. Východiskem kurikulárním dokumentů

je Národní program rozvoje vzdělávání v ČR (tzv. Bílá kniha) a Zákon o předškolním, základním, středním, vyšším odborném a jiném vzdělávání. Do vzdělávací soustavy byl zaveden nový systém kurikulárních dokumentů pro vzdělávání žáků od 3 do 19 let. Kurikulární dokumenty byly vytvořeny na dvou úrovních, tj. státní úroveň a školní úroveň. *Státní úroveň v systému kurikulárních dokumentů představují Národní program vzdělávání a rámcové vzdělávací programy (dále jen RVP). Školní úroveň představují školní vzdělávací programy (dále jen ŠVP), podle nichž se uskutečňuje vzdělávání na jednotlivých školách*. [3] Všechny kurikulární dokumenty jsou veřejně přístupné pro odbornou pedagogickou veřejnost i pro nepedagogickou veřejnost.

Systém kurikulárních dokumentů



Obrázek 1Systém kurikulárních dokumentů

1.2 Klíčové kompetence a průřezová témata

Klíčové kompetence představují souhrn vědomostí, dovedností, schopností, postojů a hodnot důležitých pro osobní rozvoj a uplatnění každého člena společnosti. [3]

Praktické osvojení těchto klíčových kompetencí je náročný, složitý a dlouhodobý proces, který není vázán ke konkrétnímu stupni nebo ročníku dané školy a ukončen spolu se školní docházkou. Jedná se o základ pro celoživotní učení žáka a jeho úspěšný vstup do pracovního života.

Mezi klíčové kompetence řadíme kompetence k učení, kompetence k řešení problémů, kompetence komunikativní, kompetence sociální a personální, kompetence občanské a pracovní kompetence.

Průřezová témata jsou povinnou součástí základního vzdělávání, jejich rozsah a způsob realizace stanovuje ŠVP. Tematické okruhy průřezových témat procházejí napříč vzdělávacími oblastmi a pozitivně ovlivňují proces utváření a rozvíjení klíčových kompetencí žáků. [4]. Předpokladem pro účinné začlenění průřezových témat do výuky je jejich provázanost a propojenost se vzdělávacím obsahem vyučovacích předmětů a souvislost s další činností žáků, která může být realizována i mimo školu. V etapě základního vzdělávání jsou vymezena tato průřezová témata: Osobnostní a sociální výchova, Výchova demokratického občana, Výchova k myšlení v evropských a globálních souvislostech, Multikulturní výchova, Environmentální výchova a Mediální výchova. [3]

1.3 Vzdělávací oblasti

„Vzdělávací obsah základního vzdělávání je v RVP ZV orientačně rozdělen do devíti vzdělávacích oblastí. Jednotlivé vzdělávací oblasti jsou tvořeny jedním vzdělávacím oborem nebo více obsahově blízkými vzdělávacími obory.“ [3]

Vzdělávací oblast	Vzdělávací obor (obory)
Jazyk a jazyková komunikace	Český jazyk a literatura Cizí jazyk
Matematika a její aplikace	Matematika a její aplikace
Informační a komunikační technologie	Informační a komunikační technologie

Člověk a jeho svět	Člověk a jeho svět
Člověk a společnost	Dějepis Výchova k občanství
Člověk a příroda	Fyzika, Chemie Přírodopis, Zeměpis
Umění a kultura	Hudební výchova Výtvarná výchova
Člověk a zdraví	Výchova ke zdraví Tělesná výchova
Člověk a svět práce	Člověk a svět práce

Tabulka 1Vzdělávací oblasti a obory podle RVP

1.4 Mezipředmětové vztahy

Dnešní svět je nabitý všemožnými informacemi, a aby je mohli žáci všechny pojmout a pochopit, je důležité, aby výuka vedla k nácviku pochopení souvislostí pomocí integrace obsahů vzdělávání. Pokud chce učitel ve výuce pracovat s mezipředmětovými vztahy, je nezbytné, aby si uvědomoval vzájemný vztah mezi jednotlivými oblastmi učiva, předměty a tématy, a učivu se věnovat právě v těchto vazbách. Výše zmiňované klade na učitele podmínku odborné znalosti vyučovaného předmětu a také pečlivou přípravu na vyučovací hodinu, což je pro učitele časově náročné.

Pedagogický slovník definuje mezipředmětové vztahy jako *„vzájemné souvislosti mezi jednotlivými předměty, chápání příčin a vztahů, přesahujících předmětový rámec, prostředek mezipředmětové integrace*“. [4]

Pokud budeme přemýšlet o mezipředmětových vztazích pouze v souvislosti s přírodními vědami, pak Jiří Škoda chápe mezipředmětové vztahy jako „*didaktickou modifikaci vztahů mezivědních, které jsou objektivní zákonitostí integrace přírodních věd. Mezioborové vztahy mohou tedy být charakterizovány jako vzájemné souvislosti mezi jednotlivými předměty, chápání příčin a vztahů, přesahující předmětový rámec, jako prostředek mezipředmětové integrace“.* [5]

Josef Janás uvádí následující charakteristiky mezipředmětových vztahů [6]:

- Jsou nezbytné k vytvoření ucelené představy žáků o přírodě a společnosti.
- Usnadňují systematizaci poznatků z různých předmětů.
- Napomáhají odstranit nežádoucí dublování učiva v jednotlivých předmětech.
- Umožňují vytvářet dovednost syntézy i transferu poznatků a pracovních metod z jednoho předmětu do druhého.

Důležitým úkolem mezipředmětových vztahů a jejich začlenění do výuky je odstranění „izolovanosti" předmětů, podpoření myšlení a uplatňování znalostí a dovedností žáků z různých vzdělávacích oblastí najednou při řešení komplexního úkolu.

2 Konstruktivismus

Jedním z mnoha poslání školního vyučování je předání nových vědomostí a poznatků svým studentům, žákům. Během vyučování k tomu dochází pomocí využití několik didaktických principů. Ty jsou závislé na mnoha faktorech, například na věku žáků, na faktické úrovni dovedností těchto žáků, na přístupu vyučujícího ke svým žákům. Přístup k žákovi ve školní výuce se během historie vyvíjí v souvislosti s různými filozofickými, kulturními a pedagogickými směry. Jedním z nich je konstruktivismus. Pedagogický směr konstruktivismus vychází z filozofických myšlenek Jeana Piageta. Ten studoval strukturu a stádia myšlenkových, kognitivních operací u dětí během vývoje. Zajímalo ho, jak se u dětí vyvíjí jejich chápání prostoru, času a kauzality (tj. příčiny a důsledku).

„*Konstruktivismus je směr druhé poloviny 20. století, který zdůrazňuje aktivní úlohu člověka, význam jeho vnitřních předpokladů a důležitost jeho interakce s prostředím a společností.*" [7]

Během druhé poloviny 20. stolení se konstruktivismus i nadále vyvíjel a došlo ke vzniku několika proudů konstruktivismu. Prvním z těchto proudů je tzv. radikální konstruktivismus, který je reprezentován Ernestem von Glasersfeldem, druhým je kognitivní konstruktivismus, mezi jeho hlavní představitele patří Jean Piaget a John Dewey, další odnoží je didaktický konstruktivismus, který je prezentován

Milanem Hejným a Františkem Kuřinou, dále sociální konstruktivismus reprezentován v díle Lva Vygotského.

Mezi charakteristické znaky konstruktivismu patří aktivita žáků během výuky. Žák je ten, kdo klade otázky a snaží se na ně najít uspokojivé odpovědi. Častý je také dialog žáků mezi sebou. Učitel má roli mentora, průvodce, režiséra. Převládající forma výuky je skupinové vyučování a individuální práce žáka. Úkol školy jako vzdělávací instituce je rozvoj kompetencí a talentu u všech žáků.

Mezi důsledky konstruktivismu patří nalezení smyslu a pochopení jevů na základě mentální struktury a individuálními zkušenostmi žáka. Učitel není chápán pouze jako jediný zdroj informací. Tyto dvě skutečnosti mohou někdy vést ke snížení některých obsahů vzdělávání. Tento fakt často vede k praktickému zaměření výuky a k praktické aplikaci získaných poznatků.

Karla Hrbáčková [8] uvádí sedm bodů, na které konstruktivistický přístup klade důraz:

1 „R*ozhodující je aktivní role žáka.*
2 *Učení je proces kognitivního konstruování,*
3 *Učení probíhá nejefektivněji prostřednictvím aktivní manipulace s předměty, jejich modely apod.*
4 *Nové učení začíná aktualizací předchozího porozumění.*
5 *Učení se navozuje nejlépe v podnětném a komplexním prostředí.*
6 *Navození významných problémových situací podporuje smysluplnost učení a motivaci žáků.*
7 *Sociální a kulturní kontext je významný pro porozumění věcem a jevům.*“

Jana Cachová a Naďa Stehlíková [9] v souvislosti s didaktickým konstruktivismem předkládají pět tezí pro výuku matematiky, které zachycují konstruktivistické přístupy k výuce z pohledu vyučujícího. Tyto teze lze využít i při výuce informatiky.

1 *„Učitel probouzí zájem dítěte o matematiku a její poznávání.*
2 *Učitel předkládá žákům podnětná prostředí (úlohy a problémy) a vhodně s nimi pracuje.*
3 *Učiteli jde především o žákovu aktivní činnost.*

4 *Učitel nahlíží na chybu jako na vývojové stádium žákova chápání matematiky a impulz pro další práci.*

5 *Učitel se u žáků orientuje na diagnostiku porozumění spíše než na reprodukci odpovědi.“*

Teze číslo jedna úzce souvisí s motivací žáka. Irena Lokšová a Josef Lokša [10] rozlišují motivační činitele podněcující výkonnost žáka na vnitřní činitele a na vnější činitele. Mezi vnitřní činitele řadíme například potřebu úspěchu a prestiže, snahu vyhnout se neúspěchu. Mezi vnější činitele zařazujeme známky, různé odměny a tresty.

Druhá teze si klade za cíl tzv. problémové vyučování, kdy učitel předloží žáků nějaký problém, který je často prezentován problémovou úlohou, u které není na první pohled patrný postup řešení a ani možný výsledek. Milan Hejný a Naďa Stehlíková [11] chápou roli učitele jako toho, kdo dětem nepředává hotové kusy poznání, ale ukazuje mu cesty, kterými se on sám k takovému poznání může dopracovat.

Čtvrtá teze ostře souvisí se způsobem, jak kantor nakládá s chybou a jak s ní dále pracuje ve výuce. Podle J. Cachové a N. Stehlíkové [9] strach z chyby, z chybného výsledku, či postupu vnáší do atmosféry práce napětí a nervozitu, tím se narušuje vztah mezi učitelem a žákem i mezi dětmi navzájem. Přitom pro řešení úloh je zapotřebí vytvořit ve třídě příznivé pracovní klima, ve kterém se žáci nesmí bát navrhovat hypotézy, které nemusí nutně vést ke správnému výsledku. Pokud jsou ale děti svázány obavami, zda jsou jejich úvahy vhodné, těžko je pak možné od nich očekávat podnětné tvůrčí návrhy.

Pro úspěšnou a správnou diagnostiku toho jak žák rozumí učivu, doporučuje M. Hejný dodržování následujícího desatera, které je sice původně zamýšlené pro matematiku, ale lze využít i během výuky informatiky. Žák, který učivu rozumí, dokáže:

1 *„Objasnit paradox,*

2 *rekonstruovat zapomenutý vzorec,*

3 *objasnit selhání standardního postupu,*

4 *obhájit standardní postup vůči námitce,*

5 *najít chybu v úvaze,*

6 *aplikovat poznatek v praxi,*

7 *rozhodnout o platnosti hypotézy,*

8 *najít objekt požadovaných vlastností,*

9 *řešit nestandardní úlohu,*

10 *objasnit některé pojmy, souvislosti, symboliku atd.*“. [12]

Opakem konstruktivistického přístupu k výuce je transmisivní přístup. Tento přístup chápe učitele jako nositele informací, které je potřeba přenést na žáka, který nemusí být během procesu výuky aktivní a pochopit plně souvislosti v učivu.

Rozdíly mezi konstruktivistickým a transmisivním vyučováním shrnuje M. Hejný a kol. [12] v následující tabulce.

	Konstruktivistické vyučování	Transmisivní vyučování
Hodnota poznání	Kvalita	kvantita
Motivace	Vnitřní	vnější
Trvalost poznání	Dlouhodobá	krátkodobá
Vztah učitel – žák	Partnerský	submisivní
Klima	Důvěry	strachu
Nositel kvality	Žák	učitel
Činnost žáka	Tvořivá	imitativní
Poznatek žáka	Produktivní	reproduktivní
Nosná otázka	CO? A PROČ?	JAK?

Tabulka 2 Porovnání konstruktivistického a transmisivního vyučování

3 Badatelsky orientovaná výuka

Efektivitou výukového procesu v souvislosti s aktivitou žáka ve výuce se zabývalo mnoho významných pedagogů a filozofů. Pro ilustraci jmenujme několik velikánů pedagogiky, kterými jsou Maria Montessori, Lev Vygotskij, John Dewey a Jean Piaget. Zejména během 20. století se vlivem humanistické filozofie a pedocentrismu začaly v pedagogické praxi objevovat různé snahy o změny tradiční role žáka v transmisivní formě výuky. V návaznosti na tyto snahy vznikaly

různé styly výuky. Jedním z těchto nově vzniklých směrů je kromě konstruktivismu i badatelsky orientované výuka.

Pojem badatelsky orientovaná výuka (BOV) pochází z anglického pojmu Inquiry Based Learning (IBL). Co to tedy ta badatelsky orientovaná výuka vlastně je? *„Badatelsky orientovaná výuka se zaměřuje na rozvíjení procesů myšlení a postojů potřebných pro zvládnutí života v budoucnosti, o které přesně nevíme, co nám přinese. Základem badatelského přístupu je žákova aktivita a kladení otázek.“* [13] Během badatelsky orientované výuky žáci vyhledávají informace, kladou otázky, hledají na ně uspokojivé odpovědi. Role učitele během výuky je proaktivní, podporuje úsilí žáků a rozvíjí jejich odpovědnost.

Podle Miroslava Papáčka [14] je cílem badatelsky orientované výuky zaměření se na bádání (inquiry) a odklonění se od výuky založené na memorování faktů. Mezi přínosy badatelsky orientované výuky z pohledu žáka patří nutnost a možnost klást otázky, značná míra aktivity během vyučování, možnost zažít pocit úspěchu z objeveného a poznání nového, a tím prožít tzv. „aha efekt“.

O významnosti takto orientované výuky svědčí i množství mezinárodních projektů, které vznikly ve spolupráci s několika univerzitami v České republice. Pro ilustraci uveďme projekt Mathematics and Science in Life: Inquiry Learning and the World of Work (MaSciL), který vznikl ve spolupráci s Přírodovědeckou fakultou Univerzity Hradec Králové, dále Projekt FIBONACCI, spolupráce s Jihočeskou univerzitou, a projekt PROFILES, který vznikl ve spolupráci s Masarykovou univerzitou.

4 Kritické myšlení a jeho rozvoj

Přestože termíny kritické a logické myšlení a jejich rozvoj jsou velmi často používané, není jednoduché tyto termíny definovat. Existuje několik možností jak definovat pojem kritické a logické myšlení a myšlení samotné.

Jiří Průcha [15] chápe myšlení jako *„poznání, nalézání a tvoření vazeb mezi předměty, které tvoří problémovou situaci“.* Myšlení podle J. Průchy také slouží k modelování vnějšího světa.

„Logické myšlení je myšlení na základě prokázaných znalostí a informací, které jsou jisté a přesné. Logické myšlení je základem pro moderní technologie a je běžně označováno jako myšlení levé mozkové hemisféry. Logické myšlení k řešení problému používá přímo fakta.“ [16]

Michael Scriven & Richard Paul definují kritické myšlení jako *„intelektuálně disciplinovaný proces aktivního a vhodného konceptování, uplatňování, analyzování, syntetizování a/nebo vyhodnocování informací získaných z/nebo generované pozorováním, zkušenostmi, reflexí, úvahou, či komunikací, jako vodítko k přesvědčení a činnosti. Ve své extrémní formě je založeno na univerzálních intelektuálních hodnotách, které přesahují předmět oblasti: přehlednost, přesnost, přesnost, důslednost, relevance, spolehlivých důkazů, dobrých důvodů, hloubka, šíři a spravedlnosti“.* [17]

Oproti tomu autoři knihy *Critical thinking for education student* [18] chápou kritické myšlení jako proces *„složený z tvorby informací, hodnocení úsudků o požadavcích a argumentech“.*

K čemu je takové kritické myšlení dobré? Kritické myšlení je způsob uvažování, který má každému člověku pomoci při řešení libovolně složitých a náročných problémů v jeho osobním i pracovním životě s využitím jeho znalostí, dovedností a zkušeností. Tyto dovednosti jsou u žáků a studentů systematicky rozvíjeny během doby studia. Schopnosti kriticky myslet ve své práci využívají zejména manažeři a ostatní vedoucí pracovníci.

Například podle [18] by člověk s rozvinutým kritickým myšlením měl být schopný:

- být vnímavý k nápadům a myšlenkám ostatních,
- udělat pozitivní a negativní soudy,
- rozlišit mezi zdroji informací a rozpoznat jejich autoritu a autenticitu,
- presentovat své myšlení a argumenty ve vhodné formě,
- být flexibilní při zvažování jiných možností a názorů,
- být čestný tváří v tvář svým vlastním předsudkům a předpojatosti.

Je důležité si uvědomit, že kritické myšlení není přímočará aktivita, může zde být mnoho opakujících se cyklů výkladu, rozboru, hodnocení, odvozování, vysvětlování a metakognicí.

Výklad (interpretace) - Pokud něco čteme, pak je naše porozumění závislé na našich vlastních osobních zkušenostech a získaných znalostí.

Rozbor (analýza) - Aby bylo možné kriticky přemýšlet, je nezbytné, aby tyto prvky byly zahrnuté do našeho přemýšlení.

Hodnocení (evaluace) - Hodnocení je proces, během kterého si každý člověk vytváří úsudek o svém dosavadním poznání a jeho zdroji.

Odvozování - Odvozování je část kritického myšlení, během které dochází k pevnění našich znalostí a dovedností.

Vysvětlování - Vysvětlování je nedílnou součástí kritického myšlení. Je to schopnost jasně a souvisle objasnit své požadavky a důvody.

Metakognice - Metakognice klade velký důraz na to jak sebevědomí a prosazení mohou ovlivnit náš závěr přemýšlení.

4.1 Řešení problémů s využitím kritického a logického myšlení

Pokud si položíme otázku: „Co je to problém?“, můžeme dostat řadu různých odpovědí. Problém je zaparkovat v některých částech města, problém je uvařit oběd pro velkou rodinu, problém je přesně narýsovat kružnici vepsanou do trojúhelníku. Obecně můžeme říci, že problém je obtížná či problematická situace, kterou se snažíme úspěšně vyřešit s využitím dostupných možností.

Pro řešení problémových situací používáme různé metody řešení. S tímto přístupem se můžeme setkat v matematice, kde se při řešení daného problému hledají způsoby, které využívají různých pravdivostních tvrzení. Hledání řešení problému jako zkouška v oblasti kritického myšlení není založeno pouze na formálních důkazech, ale častěji se zaměřuje na řešení, která jsou velmi jednoduchá a nevyužívají za všech okolností čísla. [19]

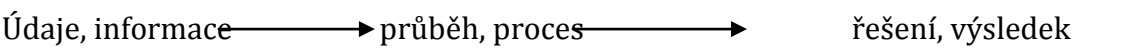
Autoři knihy *Thinking skills Critical thinking and Problem Solving* [19] uvádějí přehled způsobů, které využíváme pro řešení problémů

- určit, které části informací jsou důležité, vzhledem k množství informací, které jsou nepodstatné,

- spojit části informací, které se zdají být nepravděpodobné, pro získání nové informace,
- vztahovat jeden celek informací k jiné sadě v rozdílné formě. To zahrnuje využívat zkušeností: vztáhnout nový problém k jinému, již vyřešenému.

Při řešení problému se můžeme setkat s informacemi v různých formátech. Například jako text, čísla, grafické znázornění případně obrázek.

Jaký je tedy odborný postup při řešení problému? Na níže uvedeném schématu je tento proces rozdělen na nakolik dílčích kroků, které dohromady vytvářejí proceduru.



Tímto jednoduchým schématem ukazuje metodu, kterou lze řešet mnoho problémových situací.

Autoři [19] přinášejí metody využívané při řešení problémů:

- jednoznačně a jasně stanovit, určit požadované řešení. Přečíst si pečlivě otázku a porozumět jí je nezbytné,
- prohlédnout si dostupné informace. Následně rozlišit, které z nich jsou podstatné a které jsou nepodstatné,
- možnost provést mezivýsledek při hledání řešení,
- zkušenosti s obdobnými problémy pomáhají. Pokud jsme se nesetkali s tímto typem úkolu, bude nám jeho řešení trvat o něco déle,
- výše uvedený problém řešit systematicky.

Podle Průchy je při řešení problémů také důležitá tvořivost. Tu chápe jako *„komplex specifických schopností člověka vytvářet nové, originální až jedinečné produkty myšlenkové nebo materiální, jež jsou společensky hodnotné a komunikovatelné“* [15]

5 Výuka Informačních a komunikačních technologií podle RVP

Informační a komunikační technologie (ICT) jsou jednou z devíti vzdělávacích oblastí podle RVP. V tomto dokumentu je uvedena charakteristika vzdělávací

oblasti, cílové zaměření vzdělávací oblasti, vzdělávací obsah vzdělávacího oboru, učivo doplněné o očekávané výstupy. Vzdělávací oblast Informační a komunikační technologie podle RVP *„umožňuje všem žákům dosáhnout základní úrovně informační gramotnosti - získat elementární dovednosti v ovládání výpočetní techniky a moderních informačních technologií, orientovat se ve světě informací, tvořivě pracovat s informacemi a využívat je při dalším vzdělávání i v praktickém životě. Vzhledem k narůstající potřebě osvojení si základních dovedností práce s výpočetní technikou byla vzdělávací oblast Informační a komunikační technologie zařazena jako povinná součást základního vzdělávání na 1. a 2. stupni. Získané dovednosti jsou v informační společnosti nezbytným předpokladem uplatnění na trhu práce i podmínkou k efektivnímu rozvíjení profesní i zájmové činnosti“.*[3]

Cílem vzdělávání v této vzdělávací oblasti podle RVP je pomoci žáku rozvíjet a utvářet klíčové kompetence tím, že se žák naučí využívat moderní informační a komunikační technologie, formulovat svůj požadavek a využívat při interakci s počítačem algoritmické myšlení, respektovat právo duševního vlastnictví při práci se softwarem a také šetrné práci s výpočetní technikou.

Vzdělávací obsah tohoto oboru zahrnuje několik dílčích témat pro výuku. Prvním z nich jsou Základy práce s počítačem, kde se žáci učí ovládat počítač a jeho periferie, bezpečnému chování při práci s hesly. Mezi další témata patří vyhledávání informací a komunikace, zpracování a využívání informací, vyhledávání informací a komunikace, posledním tématem je zpracování a využití informací.

Minimální časová dotace pro výuku této vzdělávací oblasti je jedna vyučovací hodina na 1. stupni a jedna vyučovací hodiny na 2. stupni. [20]

5.1 Výuka ICT jako nástroj pro rozvoj gramotnosti

Slovem gramotnost rozumíme osvojenou dovednost číst a psát. Tuto dovednost si žáci osvojí obvykle v počátcích školní docházky. Současný rozvoj společnosti však klade na člověka řadu požadavků, které přesahují základní "trivium" - umět číst, psát a počítat. Takto rozšířená vybavenost člověka pro jeho uplatnění v životě se označuje jako funkční gramotnost. [21]

Funkční gramotnost má několik rovin:

- čtenářská gramotnost,
- matematická gramotnost,
- přírodovědná gramotnost,
- finanční gramotnost,
- ICT gramotnost. [22]

Radek Blažek [23] a další autoři uvádí ještě několik dalších dílčích složek gramotnosti. Jedná se nejčastěji o pohybovou gramotnost, jazykovou gramotnost, manažerskou gramotnost, numerickou a informační gramotnost.

5.2 Informační gramotnost a počítačová gramotnost

Téma „informační gramotnost“ se zpopularizovalo s rozvojem informačních a komunikačních technologií v posledních třech desetiletích. Je mnoho definic, které vymezují pojem informační gramotnost. Pro porovnání uveďme dvě definice. První z nich je jednou z nejpoužívanějších, která byla zveřejněna roku 1989 ve zprávě Komise pro informační gramotnost (Presidential Committee on Information Literacy - součást American Library Association) „*Informačně gramotní lidé se naučili, jak se učit. Vědí, jak se učit, protože vědí, jak jsou znalosti pořádány, jak je možné informace vyhledat a využít je tak, aby se z nich mohli učit i ostatní. Jsou to lidé připravení pro celoživotní vzdělávání, protože mohou vždy najít informace potřebné k určitému rozhodnutí či k vyřešení daného úkolu*.“ [24]

Oproti tomu uvádí ČŠI definici informační gramotnosti s větším důrazem na kompetence žáka. Ta chápe informační gramotnost jako „*schopnost identifikovat a specifikovat potřebu informací v problémové situaci, najít, získat, posoudit a vhodně použít informace s přihlédnutím k jejich charakteru a obsahu, zpracovat informace a využít je k znázornění (modelování) problému, používat vhodné pracovní postupy (algoritmy) při efektivním řešení problémů, účinně spolupracovat v procesu získávání a zpracování informací s ostatními, vhodným způsobem informace i výsledky práce prezentovat a sdílet, při práci dodržovat etická pravidla, zásady bezpečnosti a právní normy, to vše s využitím potenciálu digitálních technologií za účelem dosažení osobních, sociálních a vzdělávacích cílů“.* [25]

Informační gramotnost je nadřazený pojem k ICT gramotnosti. Tu také označujeme jako počítačovou gramotnost. [26] V souvislosti s počítačovou gramotností také mluvíme o digitálním vzdělání.

ICT gramotnost chápeme v širším pojetí jako soubor kompetencí jedince daných určitou situací, vycházíme z konceptu kompetencí jako souhrnu vědomostí, dovedností, schopností, postojů a hodnot důležitých pro osobní rozvoj a uplatnění každého člena společnosti. [22] Význam počítačové gramotnosti jako soubor kompetencí potřebných pro život a pracovní uplatnění narůstá. V této souvislosti můžeme pozorovat významný nesoulad mezi reálným životem, vzdělávacím systémem a školou. Rozdíl mezi využitím ICT v osobním životě žáků a využitím ICT ve škole se stále zvětšuje.

„*Digitálním vzděláváním reaguje na změny ve společnosti související s rozvojem digitálních technologií a jejich využíváním v nejrůznějších oblastech. Zahrnuje jak vzdělávání, které účinně využívá digitální technologie na podporu výuky a učení, tak vzdělávání, které rozvíjí digitální gramotnost žáků a připravuje je na společenské a pracovní uplatnění ve společnosti*.“ [27]

6 Historie počítačů a výpočetní techniky

6.1 Početní pomůcky, mechanické početní stroje a počítače

Jednou z motivací pro vývoj a konstrukci počítačů byla matematika a její aplikační využití. Motivací bylo nejen zjednodušit některé výpočty, např. v astrologii, ale na druhé straně i zpřesnit tyto výpočty, s přesností na několik desetinných míst, a zrychlit, tj. zkrátit dobu potřebnou pro výpočet. Dalším důvodem byla snaha omezit chyby ve výpočtech způsobené lidským faktorem, například numerické chyby.

Proto se v průběhu historie objevují různé snahy o konstrukci takového stroje. Zpočátku se jednalo o různé početní pomůcky a mechanické stroje. Tyto snahy vyústily v průběhu několika staletí ke konstrukci stroje zvaného počítač.

Jak ale definovat, vysvětlit co to počítač je? Přestože tento termín používáme denně, definice není jednoduchá. Jaromír Matucha [28] definuje pojmy počítač, mechanický stroj a početní pomůcka následujícím způsobem.

„Počítač je zařízení splňující následující podmínky:

- *provádí výpočty (početní úkony)*
- *početní úkony provádí digitálně*
- *provádí celý početní úkon, není to jen pomůcka pro lidskou paměť,*
- *pracuje samostatně podle předem zadaného programu a bez asistence člověka dokáže rozhodnout, jak v daném okamžiku pokračovat.*

Početní pomůcka (tzv. kalkulátor)- z výše uvedených podmínek nesplňuje 3. a 4. podmínku Analogová zařízení nesplňují 2. podmínku. Mechanické počítací stroje splňují pouze první tři uvedené podmínky".

Jako příklad početní pomůcky může uvést počítadlo, abaku, sčot, liny. Při práci s těmito pomůckami musel počtář provádět výpočty sám, tyto uvedené pomůcky mu pomáhají pouze při zapamatování mezi výpočtů.

6.2 Stručná historie početních pomůcek a mechanických početních strojů

Významným milníkem byl objev logaritmů Britem Johnem Napierem v roce 1614 [29]. Několik let po tomto objevu uveřejnil Henry Briggs podrobnější logaritmické tabulky, ve kterých přešel od tzv. přirozených logaritmů k dekadickým. Tento objev vedl ke zjednodušení výpočtů v astrologii, geodezii a mechanice. John Napier využil svých znalostí ke konstrukci tzv. Napierových kostek. Tyto objevy měly vliv na konstrukci logaritmického pravítka, které se stalo významnou početní pomůckou, využívanou nejen ve fyzice.

Další kategorií početních pomůcek jsou různí předchůdci dnešní kalkulačky. Jednu z prvních kalkulaček sestrojil Wilhelm Schickard. Jednalo se o mechanickou kalkulačku, která uměla násobit a dělit, přičemž obě tyto operace převáděla pomocí Napierových kostek na sčítání a odčítání.

Další významná konstrukce mechanického počítacího stroje je připisována Blaise Pascalovi. Šlo o mechanický počítací stroj, který uměl pouze provádět operace sčítání a násobení. Motivací pro konstrukci tohoto stroje byla snaha ulehčit výběrčím daní sčítání jednotlivých vybraných položek. Tento mechanický stroj se jmenoval Pascaline. [28]

Dokonce i významný německý matematik Gottfried Wilhelm von Leibniz sestrojil krokový kalkulátor, který uměl sčítat, odčítat, násobit, dělit a provádět druhé mocniny.

Mezi další významné osobnosti ve vývoji počítačů patří Angličan Charles Babbage. Ten chtěl nejprve minimalizovat chyby, ke kterým docházelo při "ručních" výpočtech prováděných pomocí matematických tabulek. V roce 1832 Babbage dokončil fungující prototyp svého diferenčního stroje, ale místo dalšího pokračování na něm se raději rozhodl pro mechanický matematický analytický stroj, který by byl schopen provádět jakékoli numerické výpočty.

Díky Babbageově spolupracovnici Adě Augustě z Lovelace obsahoval řídicí program možnost podmíněných a nepodmíněných skoků a také princip podprogramů. Ta se starala především o správu financí jeho výzkumu. Jelikož znala konstrukci a funkčnost stroje, mohla sestavit seznam instrukcí, čímž se stala první programátorkou. Na její počest pak v 80. letech americké ministerstvo obrany pojmenovalo po ní nový programovací jazyk ADA.

6.3 Stručná historie počítačů

Velký vliv na vznik, rozšíření a vývoj počítačů měly obě světové války. Mezi hlavní důvody patří dva – potřeba matematických výpočtů pro vojenské účely dělostřelectva a pro organizační a informační potřeby armády. Jak se později ukázalo v praxi, střelba z kanonu podléhá známým matematickým rovnicím. Tyto výpočty se dají zjednodušit pomocí mechanických počítacích strojů. Větším oříškem byla potřeba šifrovat a dešifrovat interní tajné informace před nepřítelem, a naopak snaha o prolomení šifrovacího kanálu nepřítele. [30]

Od meziválečného období přes období druhé světové války, studené války až do dnešní doby probíhá v oblasti vývoje počítačů intenzivní a překotný vývoj.

V průběhu této doby byly vyvinuty čtyři generace počítačů. Pro klasifikaci jednotlivých generací a jejich rozlišení mezi sebou vycházíme z několika dílčích parametrů. Jaroslav Pelikán [31] rozděluje generace počítačů podle jejich konfigurace, rychlosti, tj. počtu provedených operací za sekundu, a podle součástky, která je základním prvkem. Tomáš Pitner [32] mezi charakteristické rysy jednotlivých generací přidává popis velikosti vnější paměti, uživatelského rozhraní a metodiky zápisu algoritmů u jednotlivých generací. Michal Musílek [33] jako jeden z parametrů pro rozlišení jednotlivých generací počítačů uvádí i pokroky v softwarovém vybavení.

Generace	Rychlost (operace/s)	Součástka	Softwarové vybavení
1.	několik set	elektronky	strojový kód počítače
2.	Tisíce	tranzistory	jazyky symbolických adres, assemblery
3.	Desetitisíce	integrované obvody	vyšší programovací jazyky, operační systémy
4.	desítky milionů	mikroprocesory	aplikační software dostupný na uživatelské úrovni

Tabulka 3 Generace počítačů

6.4 Programování počítačů, programovací jazyky a jejich vývoj

Vývoj programování je neoddělitelně spjat s celkovým vývojem výpočetní techniky a vývojem hardwarového vybavení počítače. V souvislosti s vývojem programů můžeme také mluvit o generacích.

Do nulté generace můžeme zařadit mechanické stroje. Jejich programovacím jazykem byla konstrukce složená z různých ozubených koleček a válců. Šlo tedy o jakési předskokany programování.

V první generaci byly vytvářeny jednoúčelové programy, zejména pro využití v matematice a fyzice. Způsob zápisu programu byl pomocí strojového kódu. Pavel Kříž [34] uvádí „strojový kód, tedy přímo ten kód, se kterým pracují elektronické součástky. Algoritmy byly zapsány v paměti počítače přímo v číselném kódu, který zpracovával příslušný procesor. Tento kód je ve své podstatě binární, z důvodů lepší čitelnosti programu se však často místo binárního kódu používal osmičkový nebo šestnáctkový kód“.

Během vývoje se brzy ukázalo, že software (tj. strojový kód) zaostává za možnostmi hardwaru. Tento fakt vedl k vývoji nového způsobu programování tak, aby se plně využila výkonnost počítačů. Tyto snahy vyústily v konstrukci tzv. jazyka symbolických adres (assembly language). Zde se místo číselných kódů pro jednotlivé instrukce procesoru používaly symbolické zkratky, které nevyžadovaly absolutní alokaci paměti, místo ní používaly symbolické adresy, jejichž konkretizaci pak zajistil překladač typu assembler. [33] Když tento nový způsob programování byl významným ulehčením v oblasti informatiky. Tato skutečnost se stala charakteristikou druhé generace.

Dalším milníkem ve vývoji programování byla třetí generace a tzv. vyšší programovací jazyky. Jedná se o strojově nezávislé programovací jazyky, podporující metody strukturovaného programování, během kterého dochází k rozčlenění programu do autonomních funkčních celků – modulů. Každý modul obsahuje rozhraní a tělo. [34] Existují dvě různá vysvětlení pro slovo „vyšší“. Jednou verzí je, že je to protiklad k nízké úrovni strojového kódu nebo k jazyku symbolických adres. Druhou verzí je, že tyto programovací jazyky kladou vysoké požadavky na algoritmické a logické přemýšlení programátora.

Mezi typické reprezentanty této generace programovacích jazyků patří jazyky FORTRAN, COBOL a PASCAL. Nyní se na ně podíváme jednotlivě.

Programovací jazyk FORTRAN (FORmula TRANslator) byl vyvinut firmou IBM v 50. letech. Důraz ve vývoji tohoto jazyka byl kladen na rychlost a efektivnost různých vědecko-technických výpočtů. Vývoj tohoto jazyka pokračoval do počátku 60. let. Později byl jazyk FORTRAN normalizován a doplněn o nové verze.

Programovací jazyk COBOL (Common Business Oriented Language), přestože nedosahoval po formální stránce přesnosti jako FORTRAN, ale byl snadno implementovaný a použitelný programovací jazyk vhodný pro využité v komerční sféře, stal se velmi rychle rozšířeným.

Programovací jazyk PASCAL byl vyvinut v roce 1970 Niklausem Wirthem. Mezi přednosti tohoto programovacího jazyka patří jeho strukturovanost a snadná implementace pro další pracování. Motivací pro vznik tohoto jazyka byla potřeba vytvořit programovací jazyk vhodný k výuce programování.

Do této generace řadíme i rodinu tzv. objektově orientovaných programovacích jazyků. Mezi její představitele patří například jazyky Jawa, C++, Smalltalk, CLOS, Objekt Pascal, Eiffel.

Čtvrtou generací programovacích jazyků lze charakterizovat pomocí práce s grafickým uživatelským rozhraním. Místo vypisování jednotlivých příkazů dovolují komunikovat s počítačem pomocí vizuálních prostředků – nabídek, dialogů, obrázků, ikon označující data nebo programy, které je možné pomocí myši přesouvat, kopírovat, označovat a podobně. [34]

7 Základní pojmy v oblasti programování

7.1 Algoritmus

Slovo algoritmus přejala informatika z aritmetiky. Toto slovo pochází ze jména významného perského matematika Abū ʻAbd Allāh Muhammad ibn Mūsā al-Chwārizmīho. Tento učenec ve svém díle vytvořil systém zápisu početních algoritmů prostřednictvím arabských číslic a základů algebry. Jeho jméno bylo do latiny převedeno jako Algorismé, algoritmus, a původně znamenalo „provádění aritmetiky pomocí arabských číslic"; stoupenci této myšlenky se jmenují algoristé a počítají pomocí algoritmů.

Ale co znamená slovo algoritmus dnes v informatice?

„Algoritmus je přesný popis, popisující určitý proces, který vede od měnitelných vstupních údajů k požadovaným výsledkům. Jinými slovy – algoritmus je jednoznačný a přesný popis řešení problému." [35]

Eva Milková v knize Algoritmy [36] uvádí, že algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy.

Vlastnosti algoritmu [36]:

Hromadnost - znamená, že algoritmus lze použít pro řešení obecné úlohy, tj. že nepopisujeme postup jedné úlohy, ale poslouží k řešení libovolné úlohy, která patří do jisté třídy úloh

Determinovanost - v každém kroku je jednoznačně určen způsob pokračování práce algoritmu

Konečnost - algoritmus se nezacyklí, po určitém počtu kroků skončí

Resultativnost - po konečném počtu kroků dospěje k řešení (vrátí třeba jen chybové hlášení)

Marta Bechyňová [35] doplňuje tyto vlastnosti ještě o následující dvě:

Správnost - výsledek, který vznikne použitím algoritmu, musí být správný

Opakovatelnost - algoritmus vede vždy ke stejným výsledkům, jsou-li zadána stejná data.

Algoritmus je tedy posloupnost příkazů (série pokynů), která uvádí postup, jak vyřešit daný problém. Zjednodušeně řečeno lze říci, že algoritmus je postup, který vede k určitému cíli. Existuje několik způsobů, jak zapsat algoritmus. Michal Krátký a Jiří Dvorský uvádějí [37] následující způsoby:

- přirozený jazyk (slovní popis),
- grafické znázornění (např. vývojový diagram),
- speciální jazyk (pseudojazyk), programovací jazyk.

Algoritmizace je přesný postup, který se používá při tvorbě programu pro počítač, jehož prostřednictvím lze řešit nějaký konkrétní problém. [35]

Programovací jazyk vysvětluje Marek Běhálek [38] jako „standardizovaný nástroj pro komunikaci s počítačem“. Oproti tomu M. Bechyňová používá přesnější definici k vysvětlení pojmu programovací jazyk. Uvádí „programovací jazyk = umělý jazyk, jenž se používá pro definování sekvence programových příkazů, které lze zpracovat na počítači.“ [28]

7.2 Programování

Programování je proces, během kterého člověk (programátor) přepisuje algoritmus pro řešení daného problému do programovacího jazyka. Tento proces lze rozdělit do čtyř kroků:

- přesná definice problému,
- sestavení algoritmu,
- tvorba programového kódu,
- ověření funkce programu. [39]

V prvním kroku je důležité položit si otázky typu: Co má být výsledkem? Jaké mám dosavadní informace? Které informace mi pomohou při řešení problému? Rozumím problému, který mám vyřešit?

Ve druhém kroku sestavím jednoznačný sled kroků, pokynů, které mají řešit daný problém. Algoritmus přesně popisuje postup zpracování daného úkolu, nedává však odpověď na daný problém, ale pouze postup, jak ji získat.

Ve třetím kroku sestavíme na základě algoritmu řešené úlohy program (zdrojový text) v konkrétním programovacím jazyce.

Ve čtvrtém kroku procesu ověřujeme to, že je navržený program funkční, tj. řeší zadaný problém. Ke správné práci programu je potřeba provést tzv. odladění. Cílem odladění je odstranění chyb z programu. Nejčastější chyby jsou chyby v zápise, tzv. syntaktické - ty většinou odhalí překladač. Závažnější jsou logické chyby, vyplývající z nesprávně navrženého algoritmu, nebo chyby, vzniklé špatným předpokladem v etapě formulace nebo analýzy úlohy - ty se projeví nesprávnou činností programu nebo špatnými výsledky - při odstraňování těchto chyb může pomoci ladící program (debugger) umožňující sledování aktuálního stavu proměnných a krokování. Teprve po odstranění všech druhů chyb můžeme program použít k praktickému řešení úloh. [40]

7.3 Základní algoritmické konstrukce

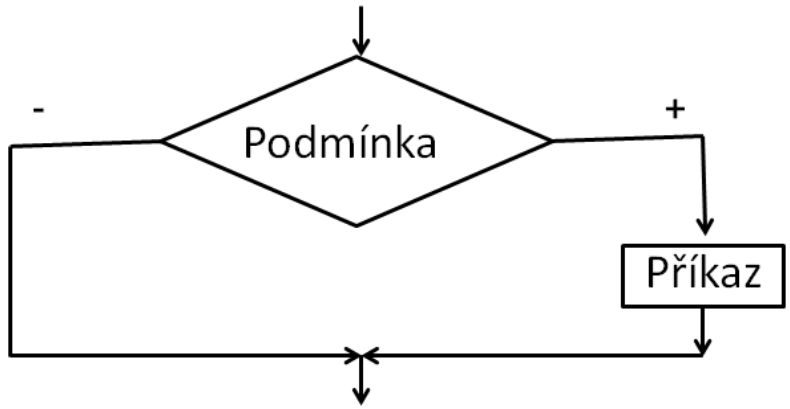
Posloupnost příkazů je algoritmická konstrukce, která obsahuje příkazy, které jsou vykonány v uvedeném pořadí. [36]

7.3.1 Větvení programu

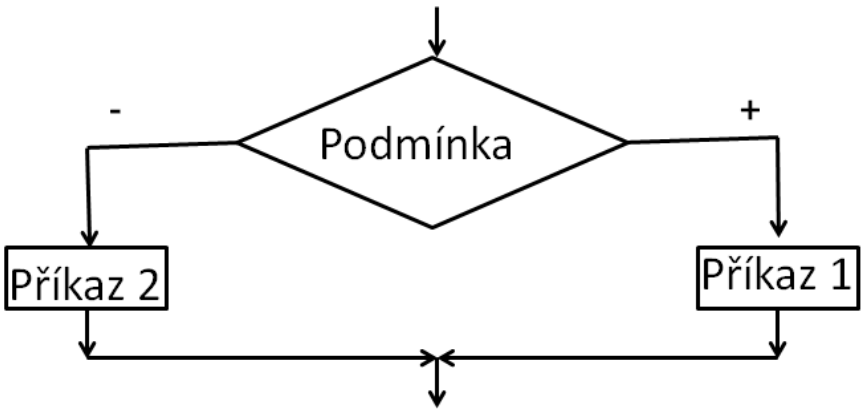
Větvení programu můžeme rozumět jako výběru mezi několika variantami následujících příkazů na základě splnění nebo nesplnění určité podmínky výběru. [41] E. Milková [36] definuje příkaz větvení jako *„algebraickou konstrukci, která*

obsahuje za klíčovým slovem „jestliže“ podmínku, na jejímž splnění nebo nesplnění závisí vykonání příkazu (buď jednoduchého, nebo strukturovaného) uvedeného za příslušným klíčovým slovem (pak, jinak)“. Rozlišujeme dva základní typy větvení, větvení úplné a větvení neúplné.

Neúplné větvení provede program tak, že nejprve vyhodnotí podmínku a pokud je podmínka splněna, provede se Příkaz1, pokud není splněna, neprovede se nic, příkaz je bez účinku. A program pokračuje dál. Úplné větvení programu, na rozdíl od neúplného, řeší možnost vykonání příkazu i tehdy, jestliže podmínka není splněna. Realizace programu proběhne tak, že se nejprve vyhodnotí podmínka, pokud je splněna, vykoná se Příkaz1, ale pokud není splněna podmínka, provede se Příkaz2.



Obrázek 2 Neúplné větvení

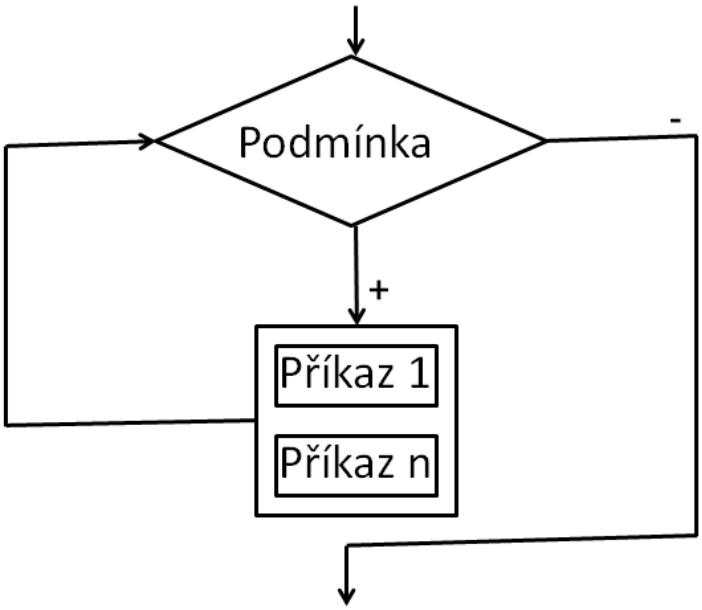


Obrázek 3 Úplné větvení

7.3.2 Cyklus

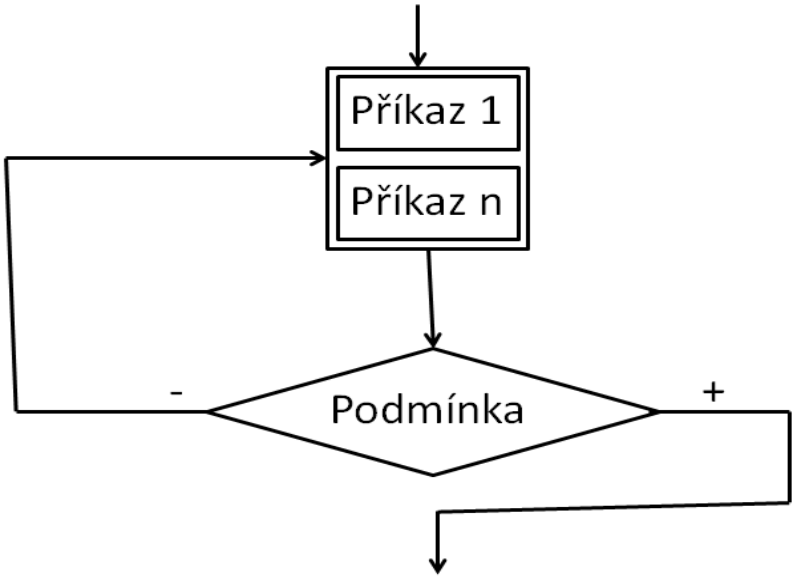
„Cykly jsou strukturované příkazy umožňující, aby se některé příkazy programu mohly opakovat vícekrát. Všechny mají ve své struktuře: tělo cyklu – tj. příkazy, které se mají opakovat, a podmínku cyklu – která rozhoduje, zda se má v provádění cyklu pokračovat, nebo už skončit.“ [42]

Cykly můžeme rozdělit na dva typy. Prvním typem je cyklus s podmínkou na začátku ověřování, testování podmínky proběhne ještě před prvním provedením příkazů v cyklu. Jestliže je splněna podmínka, program vykoná příkaz (případně více příkazů) a opět se bude ověřovat podmínka. Tato činnost se stále cyklicky opakuje, dokud je podmínka splněna. Až nebude podmínka splněna, přestane opakování se daných příkazů.



Obrázek 4 Cyklus s podmínkou na začátku

Druhým typem cyklů je cyklus s podmínkou na konci. Ověření podmínky proběhne až po prvním vykonání příkazů v těle cyklu. Což v praxi znamená, že program vykoná příkazy v těle cyklu a poté ověří podmínku. Dokud není splněna podmínka, vykonává se opakovaně příkaz, příkazy. Opakování příkazu skončí, když je podmínka splněna.



Obrázek 5 Cyklus s podmínkou na konci

Pro ilustraci si uveďme příklad z praxe. Algoritmy, jak již bylo řečeno, vedou k dosažení určitého cíle. Cílem může být vypočítat příklad, narýsovat kružnici nebo umýt nádobí. Pokud myjeme nádobí a talíř je špinavý, musíme po jeho povrchu přejet houbičkou opakovaně, tj. více než jednou, případně použít sílu a přitlačit. To vše je závislé na okolnostech (podmínkách).

8 Přehled programovacích jazyků užívaných pro výuku programování

V dobách počátku svého vzniku vyžadovaly počítače odborný přístup ze strany technické obsluhy vzhledem ke svým technickým parametrům. Mezi skupinu profesí, která se starala o obsluhu a chod počítače, můžeme zařadit programátory a technický personál, který například měnil součástky. Mezi první programátory patřili nejčastěji výborní matematici, fyzici. Tuhle skutečnost lze vysvětlit i v rámci historických souvislostí, kdy do konce druhé světové války neexistovala informatika jako samostatný vědní obor, a tím pádem ani odborníci, programátoři v oboru informatiky. Tato skutečnost se začala velmi měnit v 50. a 60. letech, kdy mohlo vzniknout několik významných pracovišť zaměřujících se na využití počítačů. Tím vzniká i poptávka po prvních kvalifikovaných programátorech. S tím jde ruku v ruce otázka, jak tyto lidi „vyučit řemeslu“ a jaké k tomu zvolit prostředky. Aneb jaký programovací jazyk je vhodný pro výuku programování.

Mezi první programovací jazyk, který byl zamýšlen pro výuku programování, je pravděpodobně Pascal. Tento programovací jazyk konstruoval Niklause Wirth. Tento jazyk je podle mnohých považován za ideální jazyk pro výuku programování na vysokoškolské, případně středoškolské úrovní [6a, 6b,6c]. Pascal je vhodný pro výuku programování u začátečníků z několika důvodů. Prvním z nich je srozumitelnost zápisu, rozvoj systematického a strukturovaného myšlení, rozvoj pečlivosti při zápisu programu a může se stát vstupní branou pro další programovací jazyky.

8.1 Kritéria pro výběr programovacího jazyka

Při volbě zařazení programovacího jazyka do výuky musí vyučující během procesu rozhodování zohlednit několik faktorů. Mezi první oblast rozhodování patří oblast obsahu výuky a cíle výuky. Zde by si vyučující měl odpovědět na několik otázek Rostislav Fojtík [43] jako vodítko uvádí:

- Je cílem naučit základům algoritmizace? Pak je vhodné využívat jednoduché názorné prostředky.
- Je cílem výuky zvládnout konkrétní jazyk?
- Znají základy programování?
- Je cílem naučit se programovat podle moderních měřítek? To znamená zaměřit se na objektově orientovaný přístup, který je uplatňován v současné praxi nejčastěji?

Další oblast má charakter více zaměřený na programovací jazyky. Alena Halousková uvádí [44] několik zásad pro otestování konkrétního vhodného programovacího jazyka. Jde například o ověření toho, do jaké míry je programovací jazyk:

- spolehlivý a efektivní,
- podobný syntaxi s jinými jazyky,
- zachází s datovými typy,
- rozšiřitelný, zda je volně a snadno dostupný,
- nepoužívá se jen k výuce, má i reálné využití v praxi,
- má uživatelskou komunitu nabízející podporu,
- jsou k dispozici dobré výukové materiály,
- stav vývoje.

9 Přehled dětských programovacích jazyků

Otázkou zůstává, jaký zvolit programovací jazyk pro začátek výuky programování a algoritmizace, pokud jde o děti. Touto otázkou se zabývalo nemálo učitelů informatiky. V průběhu několika desetiletí bylo vytvořeno několik programovacích jazyků pro výuku dětí, ale učitelé je mohou využít i v začátcích výuky programování u dospělých a adolescentů. Souhrnně bývají nazývány jako „dětské programovací jazyky“ [45]. V anglicky psané literatuře je též obvyklý termín „educational programming language“.

9.1 Program Logo

Tento programovací jazyk je vlajkovou lodí mezi dětskými programovacími jazyky. Byl vytvořen roku 1967 skupinou spolupracovníků - Danielem Bobrowem, Wallym Feurzeigem, Seymourem Papertem a Cynthiem Solomonem. Papert patřil mezi významné žáky Jeana Piageta, proto byl ve své tvorbě ovlivněn konstruktivismem. Program je postaven na myšlence ovládání pohybu želvy (malého robota) pomocí pokynů programu. Program Logo stojí za vznikem tzv. želví geometrie [46]. Tento jazyk má i řadu implementací. Jednu z nich vytvořili Andrej Blaha, Ivan Kalaš a Petr Tomcsány. [45]

9.2 Program Karel

Program Karel navrhl Richard Pattis v roce 1981. Cílem programu Karel je rozpohybovat robota ve světě v podobě čtvercové sítě, čtyř základních příkazů (krok, vlevo v bok, polož a zvedni) a dvou testů (je zeď?, je značka?) a příkazů pro vytvoření struktury (podmínka, cyklus).

9.3 Program Baltík

I v českém prostředí byl vytvořen program pro výuku programování u dětí. Autorem je informatik Bohumír Soukup. Ideovou linkou je svět čaroděje Baltíka, ve kterém mu děti pomáhají při plnění různých úkolů.

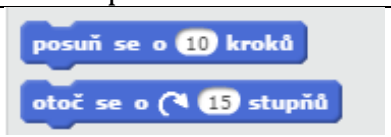
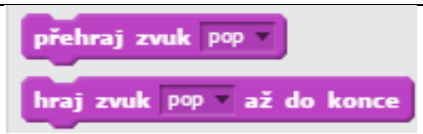
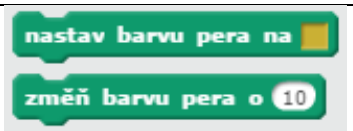
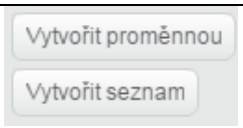
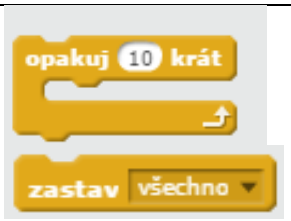
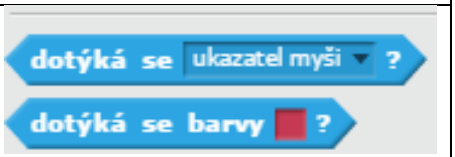
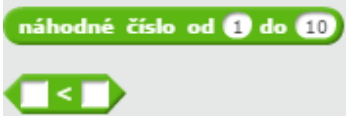
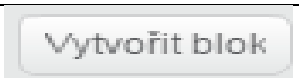
9.4 Program SCRATCH

Program SCRATCH navrhl Mitchel Resnick ze skupiny Lifelong Kindergarten (v překladu „Celoživotní mateřská školka“), působící v rámci Massachusetts Institute of Technology (MIT).

Program SCRATCH je dostupný online a bezplatně na webových stránkách https://scratch.mit.edu/. Tyto internetové stránky jsou téměř kompletně přeložené do češtiny, až na několik stránek webu, které kombinují český překlad a původní angličtinu. Na tomto webu se každý zájemce o programování může zaregistrovat do komunity uživatelů. K registraci je potřeba zadat své uživatelské jméno, stát, ve kterém uživatel žije, měsíc a rok narození, pohlaví a kontaktní email. Cílem této komunity je možnost sdílení vlastních vytvořených programů s jinými uživateli, inspirovat se tvorbou, případně si vzájemně poradit při řešení nových projektů. Česká komunita uživatelů programu SCRATCH má 28 800 uživatelů [47].

9.4.1 Prostředí programu SCRATCH

Prostředí programu SCRATCH je rozděleno do několika částí. Prvním z nich je bílý obdélník umístěný v levé horní části obrazovky, který tvoří jakési hřiště či dvorek, ve kterém se pohybuje kočka (kocour) SCRATCH. V levé části obrazovky jsou umístěny pokyny pro ovládání programu SCRATCH. Tyto pokyny jsou rozděleny do tří záložek, a to „Scénáře“, „Kostýmy“ a „Zvuky“. Záložka „Scénáře“ obsahuje několik sekcí, jako například Pohyb, Data, Události, Vnímání. Každá tato sekce je označena jinou barvou. Přehled viz dále uvedená tabulka, která je pro přehlednost uvedena na další stránce.

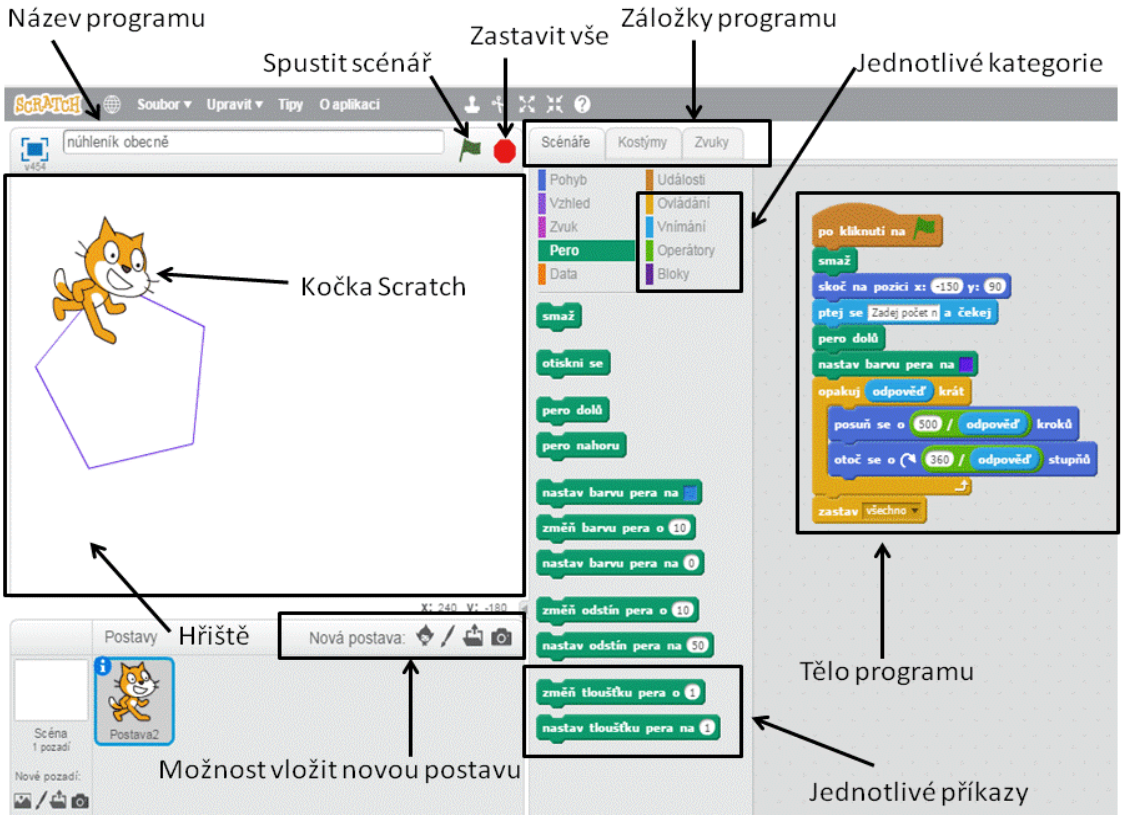
Kategorie	Poznámka	Ukázka příkazů
Pohyb	Zajišťuje pohyby a natáčení postav	
Vzhled	Ovlivňuje vzhled postav, změny jejich velikosti, komiksové bubliny, změny pozadí	
Zvuk	Přehrává zvukových souborů a programovatelných nástrojů a tónů.	
Pero	Nabízí možnost kreslení perem s volitelnou tloušťkou, barvou a intenzitou.	
Data	Slouží k vytváření proměnných a práce s jejich hodnotami.	
Události	Počáteční bloky určují spouštění události pro navazující posloupnost bloků.	
Ovládání	Obsahuje výběr možnosti využití podmíněného příkazu, který umožňuje částečné i úplné větvení programu, cyklů a zastavení programu	
Vnímání	Umožňuje postavám reagovat na kontakt s okolím, které uživatel vytvořil.	
Operátory	Umožňuje užití aritmetických a logických operátorů, základní výpočty a generaci náhodných čísel.	
Bloky	Uživatelské procedury (bloky) a řízení připojených externích zařízení.	

Tabulka 4 Kategorie v prostředí SCRATCH

V záložce „Kostýmy“ může uživatel měnit vzhled postavičky SCRATCH. K tomu může využít předdefinované možnosti nebo nahrát, namalovat vlastní kostým, případně nahrát z knihovny. V záložce „Zvuky“ může uživatel namluvit zvuk, případně ho vybrat ze souboru.

9.4.2 Popis prostředí SCRATCH

Prostředí SCRATCH je rozděleno do několika částí. Grafické rozhraní tohoto prostředí nabízí pro uživatele snadnou orientaci. Významnou částí, kde se program odehrává, je bílý obdélník. Jedná se o jakési hřiště, kde se pohybuje kočka SCRATCH, případně jiná postava. Nad touto lištou je umístěna ikonka zelené vlaječky. Po kliknutí na tuto vlaječku dojde ke spuštění scénáře (programu). Vlevo od této ikony se nachází ikonka červeného osmiúhelníku. Ta má funkci zastavit probíhající program. Dole pod bílým obdélníkem je umístěno několik ikon, které umožňují vložení nové postavy. V prostřední části obrazovky v horizontálním směru se nacházejí jednotlivé záložky programu. Uživatel si zde přepíná mezi záložkou „Scénáře“, „Kostýmy“, a „Zvuky“. Každá z těchto záložek obsahuje několik kategorií. Vlevo od záložek je šedý obdélník, který slouží jako pracovní stůl. Zde uživatel vytváří svůj program z jednotlivých příkazů, které jsou reprezentovány jako dílky skládačky. Ilustrační obrázek je pro větší názornost uveden na další stránce.



Obrázek 6 Prostředí SCRATCH

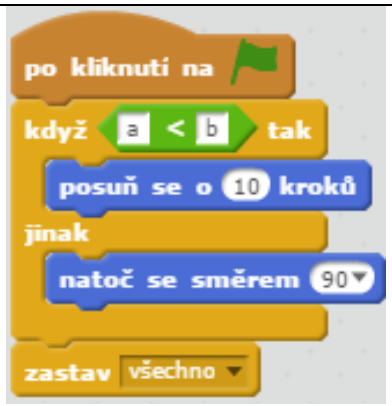
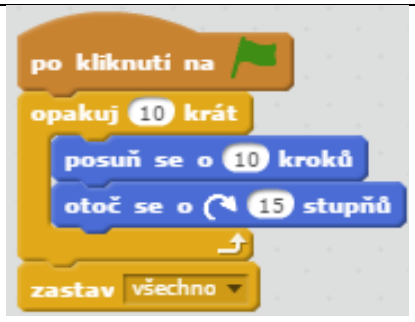
9.4.3 Programátorský pohled na prostředí SCRATCH

Prostředí SCRATCH podle [48] rozvíjí klíčové kompetence k řešení problémů a projektové dovednosti. Tento rozvoj probíhá díky zapojení logického uvažování a abstrakce, dělení složitých problémů na jednodušší části, rozvoj nápadu od počáteční fáze až po dokončení celého projektu. Dále tento dokument uvádí, že prostředí SCRATCH pomáhá uživatelům pochopit základní principy počítačů a programování. Konkrétně se jedná o následující zásady:

- uživatel musí přesně říci počítači, co má udělat,
- každý může vytvářet počítačové programy (nejen IT experti),
- psaní počítačových programů nevyžaduje žádné speciální odborné znalosti, jen jasné a pečlivé myšlení.

Prostředí SCRATCH podporuje následující programové koncepce[44] sekvence příkazů, příkazy cyklu, podmíněné příkazy, proměnné, seznamy (pole), reakce na události, vlákna, koordinace a synchronizace (zprávy a čekání), vstup z klávesnice, náhodná čísla a booleovskou logiku. V následující tabulce si ukážeme vybrané

programové koncepce z pohledu její logické konstrukce, jejího využití s připojením ukázky sestrojeného programu.

Programová koncepce	Vysvětlení – rozvíjí	Ukázka sestrojeného programu
Posloupnost	Systematické myšlení, ohledně pořadí kroků v programu	
Neúplné větvení	logické myšlení	
Úplné větvení	logické myšlení, práci s alternativním řešením	
Smyčka	slouží k opakování řady instrukcí při splnění dané podmínky	

Události	Události jsou posloupnosti příkazů spouštěné uživatelem nebo další části programu.	
Vlákna	Slouží ke spuštění dva vláken současně, která jsou vykonávána paralelně.	Pozn.: Každé vlákno programu patří k jedné postavě. Proto může v programu SCRATCH být zapojeno více postav.

Tabulka 5 Programové koncepce v prostředí SCRATCH

A. Halousková [44] ve své diplomové práci mezi nevýhody prostředí SCRATCH z programátorského hlediska uvádí dva argumenty. Prvním z nich je skutečnost, že SCRATCH není v pravém slova smyslu programovací jazyk. Druhou nevýhodou podle A. Halouskové jsou specifické druhy cyklů, které SCRATCH používá.

9.4.4 Didaktický pohled na prostředí SCRATCH

SCRATCH je programovací jazyk, který řadíme do skupiny tzv. dětských programovacích jazyků. SCRATCH už od počátku byl konstruován, vytvářen za účelem výuky programování u dětí. Tuto skutečnost je potřeba zohlednit v didaktických aspektech tohoto programu. V následující kapitole si prohlédneme SCRATCH z didaktického hlediska.

Pokud chceme posoudit, zda daný materiál (učebnice, pracovní listy, prezentace, video) je v hodný zařadit do výuky, je vždy důležité předtím zhodnotit daný materiál z pohledu didaktiky. Didaktika je podle Oldřicha Šimoníka [49] věda o vzdělání, která zkoumá cíl, obsah, prostředky a podmínky vyučování. Z těchto poznatků vycházejí didaktické zásady. Mezi didaktické zásady (principy) řadí O. Šimoník [49] princip výchovnosti, princip cílevědomosti, princip uvědomělosti, princip aktivity, princip názornosti, princip soustavnosti, princip posloupnosti, princip přiměřenosti, princip individuálního přístupu k žákům, princip zpětné vazby, princip spojení teorie s praxí a princip spojení teorie se životem. Robert

Fisher [50] k uvedeným principům přidává princip zpětné vazby. Nyní se vrátíme k programovacímu prostředí SCRATCH a popíšeme si toto prostředí z pohledu již zmíněných didaktických principů.

Zásada názornosti je v prostředí SCRATCH dodržena pečlivě. To dokazuje paralelní zobrazení „dvorku“ pro pohyb kočky SCRATCH v levé části obrazovky a zobrazení pracovní plochy pro tvorbu algoritmů, programu na pravé části obrazovky. Navíc prostředí SCRATCH nabízí i možnost opakované změny v těle programu. Touto cestou lze lépe pochopit jednotlivé příkazy programu, sled těchto příkazů a v konečném důsledku může tento způsob práce vést k odstranění příkazů, které daný program zbytečně opakuje, jsou mezi sebou ve vzájemném logickém rozporu nebo nejsou dostatečně efektivní.

V prostředí SCRATCH je také dobře zakomponována zpětná vazba. Tato zásada je v praxi realizovaná například pomocí bezprostředního vykonání programu po jeho spuštění, možností zapojit se do komunity uživatelů a mít možnost získat i od nich zpětnou vazbu ve formě komentáře ke svému vytvořenému programu.

Při hodnocení vědeckosti v prostředí SCRATCH je několik faktorů, které budeme sledovat. Velmi kladně bych hodnotila skutečnost, že uživatel již pracuje s příkazy, které jsou graficky předpřipravené ve formě jednotlivých „kousků, dílků“ skládačky, které do sebe zapadají. Tento fakt jistě velmi ocení začínající uživatelé při tvorbě programu, který obsahuje cyklus, případně podmínku, protože SCRATCH „myslí“ za ně a nabízí „dílek“, který nezapomíná například u podmínky na využití větvení. Osobně mě během práce v programovacím prostředí SCRATCH zarazil fakt, že zde není nutné použít „dílek“, který by symbolizoval konec programu, tak jak to například známe u řady jiných programovacích jazyků. Tento dílek není ani nazván „Konec“, ale jako „Zastavit“. Uživatel si může vybrat ze dvou možností. První možnost je „Zastavit vše“, druhou je „Zastavit tento scénář“.

10 Výuka programování

Jednou z mnoha dílčích kompetencí, které řadíme do digitálního vzdělání je i dovednost programování. Výuka programování, ale není zahrnuta do povinného učiva v rámci RVP pro základní školy, do výuky na středních školách je výuka

programování zahrnuta v RVP pro gymnázia a u některých typů odborných středních škol. Tento fakt znepokojuje řadu odborníků. Na tuto situaci reaguje například nevýdělečný program code.org, jehož hlavním cílem je zpřístupnit bezplatně co nejvíce lidem možnost pro online výuku programování. Mezi hlavní důvody patří skutečnost, že programování je způsob, jak rozvíjet schopnosti a potenciál každého člověka.[51]

Základní znalost některého z programovacích jazyků se dnes i pro běžného uživatele počítače stává stále citelnější potřebou. Hlavním cílem výuky programování není vychovávat programátory, kteří zvládnou i náročné algoritmizační úlohy, ale především získat žáky a studenty, kteří budou schopni použít nabyté vědomosti a zkušenosti s programováním k vytváření výukových aplikací dotvářejících celkovou koncepci pojetí moderní výuky.[52]

Výuka programování má velký vliv na rozvoj logického myšlení u žáků na jejich budoucí pracovní uplatnění, na rozvinutí lidského kapitálu a ekonomický rozvoj společnosti

Steve Jobs na otázku: Proč se učit programovat? odpověděl: „Myslím, že by se všichni měli naučit programovat, protože vás programování naučí přemýšlet“. [53]

11 Vytvořené podklady pro výuku programování

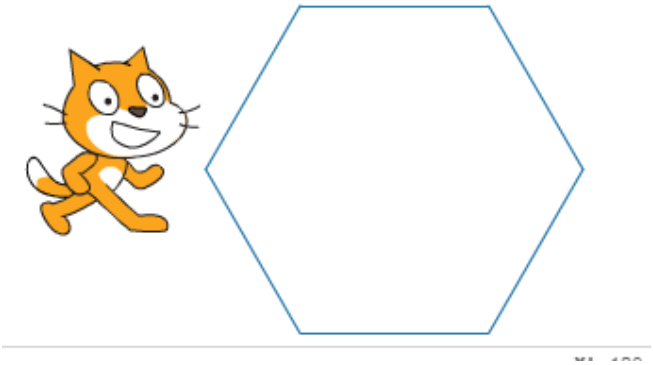
Cílem empirické části této diplomové práce bylo vytvoření souboru úloh, který by mohl vyučující zařadit do výuky programování pomocí dětských programovacích jazyků. V rámci diplomové práce bylo vytvořeno celkem deset úloh. U každé úlohy je uveden její název, přibližná časová náročnost, použitá algoritmická konstrukce, výchovné a vzdělávací cíle, zadání úlohy a mezipředmětové vztahy pro danou úlohu. Řešení všech úloh je uvedeno na přiloženém CD. Případně dostupné online na stránkách scratch.mit.edu, kde jsou tyto úlohy dohledatelné pod uživatelským jménem dpkunhartova.

Předkládané úlohy zlepšují u žáků následující klíčové kompetence, kterými jsou personální (osobnostní) a sociální výchova, kompetence k řešení problémů a kompetence pracovní. Žáci jsou učitelem vedeni k hledání řešení, samostatnému řešení, hledání způsobu řešení problému. Pracovní kompetence jsou rozvíjeny

samostatnou prací žáků a přemýšlením nad postupem práce. Společným tematickým celkem je úvod do programování, základy algoritmizace. Mezi začleněná průřezová témata do uvedených úloh patří osobnostní a sociální výchova a mediální výchova. Jako pomůcka pro řešení jednotlivých úloh slouží počítač případně tužka a papír pro případné náčrtky.

11.1 Pravidelný n-úhelník

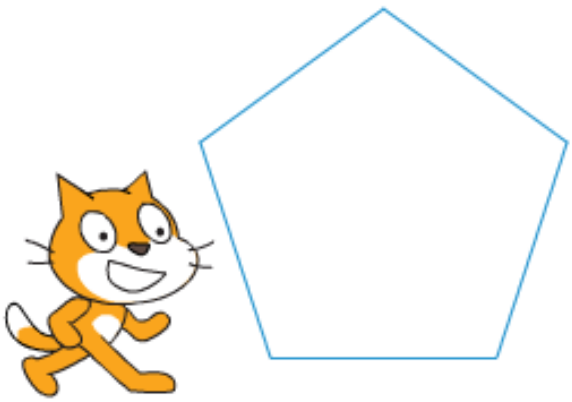
Název programu	Pravidelné n-úhelníky
Očekávaná časová náročnost	5 minut
Použitá algoritmická konstrukce	cyklus s předem daným počtem opakování
Výchovné a vzdělávací cíle	Žák umí vysvětlit pojem pravidelný n-úhelník, žák umí načrtnout pravidelný n-úhelník, žák umí vypočítat vnitřní úhel v pravidelném n-úhelníku
Zadání pro tvorbu programu	Vytvořte program, který umí vykreslit pravidelný trojúhelník (tj. rovnostranný trojúhelník), pravidelný čtyřúhelník (tj. čtverec), pětiúhelník, šestiúhelník.
Mezipředmětové vztahy	Matematika



Obrázek 7 N-úhelník

11.2 Pravidelné n-úhelníky obecně

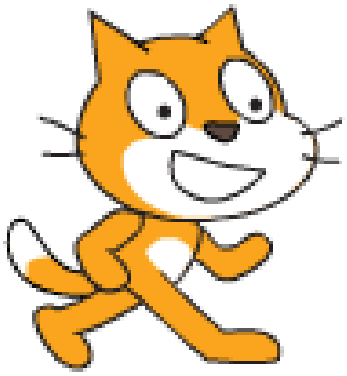
Název programu	Pravidelné n-úhelníky obecně
Očekávaná časová náročnost	15 minut
Použitá algoritmická konstrukce	cyklus s předem neznámým počtem opakování
Výchovné a vzdělávací cíle	Žák umí vysvětlit pojem pravidelný n-úhelník, žák umí načrtnout pravidelný n-úhelník, žák umí vypočítat vnitřní úhel v pravidelném n-úhelníku.
Zadání pro tvorbu programu	Vytvořte program, který se nejprve zeptá uživatele, kolika úhelník chce vytvořit, uživatel zadá libovolné n, následně SCRATCH vykreslí daný n-úhelník.
Mezipředmětové vztahy	Matematika



Obrázek 8 Pravidelný pětiúhelník

11.3 Kočka a mezerník

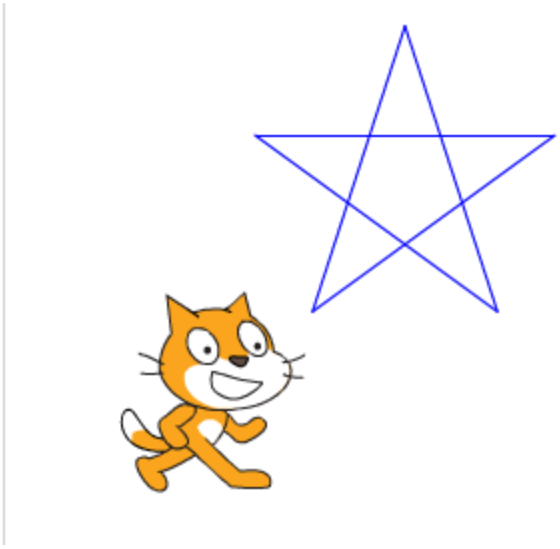
Název programu	Kočka a mezerník
Časová náročnost	10 minut
Použitá algoritmická konstrukce	neúplné větvení, podmínky
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení.
Zadání pro tvorbu programu	Vytvořte program, ve kterém se kočka pohybuje libovolně po celé ploše, pokud kočka narazí do zdi, odrazí se a pokračuje dál. Celý program může kdykoliv ukončit uživatel stisknutím mezerníku.
Mezipředmětové vztahy	Fyzika



Obrázek 9 Kočka SCRATCH

11.4 Pravidelná hvězda s pěti, sedmi a patnácti cípy

Název programu	Pravidelná hvězda
Časová náročnost	20 minut
Použitá algoritmická konstrukce	cyklus s předem daným počtem opakování
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení.
Zadání pro tvorbu programu	Vytvořte program, který vykreslí pěticípou hvězdu, sedmicípou hvězdu a patnácticípou hvězdu.
Mezipředmětové vztahy	Matematika



Obrázek 10 Pěticípá hvězda

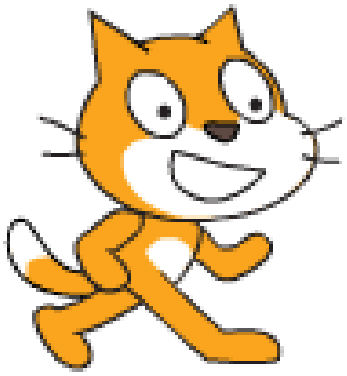
11.5 Komentátor

Název programu	Komentátor
Časová náročnost	10 minut
Použitá algoritmická konstrukce	neúplné větvení
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení.
Zadání pro tvorbu programu	Vytvořte program, během kterého kočka SCRATCH jde vodorovně po obrazovce. Na začátku kočka řekne: „Už jdu!“, dojde do poloviny obrazovky, kde řekne: „Už jsem v polovině“, poté pokračuje dál a na konci řekne: „Už jsem v cíli!“
Mezipředmětové vztahy	čeština, fyzika

Obrázek 11 Komentátor

11.6 Cestovatel

Název programu	Cestovatel
Časová náročnost	10 minut
Použitá algoritmická konstrukce	neúplné větvení
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení.
Zadání pro tvorbu programu	Vytvořte program, během kterého kočka SCRATCH jde vodorovně po obrazovce. Poté, co dojde do poloviny obrazovky, řekne: „Hurá!“ a zmizí. Pro efektivitu lze změnit pozadí.
Mezipředmětové vztahy	čeština, fyzika



Obrázek 12 Cestovatel

11.7 Závodník

Název programu	Závodník
Časová náročnost	10 minut
Použitá algoritmická konstrukce	neúplné větvení, paralelní programy
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení. Umí naprogramovat činnost dvou postav.
Zadání pro tvorbu programu	Vytvořte program, během kterého kočka SCRATCH jde vodorovně po obrazovce v autíčku. Poté, co dojde do poloviny obrazovky, řekne „To je jízda!“ a jede dál. Pro efektivitu lze změnit pozadí.
Mezipředmětové vztahy	matematika, fyzika

Obrázek 13 Závodník

11.8 Stopař

Název programu	Stopař
Časová náročnost	15 minut
Použitá algoritmická konstrukce	neúplné větvení, paralelní programy
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení. Umí naprogramovat činnost dvou postav.
Zadání pro tvorbu programu	Vytvořte program, ve kterém kočka SCRATCH vystupuje jako stopař. Nejprve čeká na autíčko, poté přijede autíčko, kočka SCRATCH nastoupí a odjíždí. Možné rozšíření: Během čekání může kočka říci: „To je ale doba!“, poté co nastoupí, může říci: „Konečně!“
Mezipředmětové vztahy	matematika, fyzika, čeština

Obrázek 14 Stopař

11.9 Nehoda

Název programu	Nehoda
Potřeby a pomůcky	Počítač, papír na případné náčrtky
Časová náročnost	20 minut
Použitá algoritmická konstrukce	neúplné větvení, paralelní programy, cykly
Výchovné a vzdělávací cíle	Žák umí sepsat program s využitím větvení a cyklů. Umí naprogramovat činnost dvou postav.
Zadání pro tvorbu programu	Vytvořte program, ve kterém kočka SCRATCH vystupuje jako řidič autíčka. Autíčko jede, proti němu vyjede jiné autíčko, obě autíčka jedou proti sobě, ale dojde ke srážce. Možné rozšíření: Srážka může být doplněna o zvukové, případně o vizuální efekty.
Mezipředmětové vztahy	matematika, fyzika,

Obrázek 15 Nehoda

12 Ověření navržených materiálů

12.1 Příprava na výuku a vytvoření materiálů

Cíle empirické části byly dva. Prvním z nich bylo ověření časového rozvržení u jednotlivých úkolů a druhým cílem bylo ověření, zda úroveň navržených úkolů je přiměřené pro žáky druhého stupně základní školy. Použitou metodou pro ověření cílů praktické části této diplomové práce bylo účastněné pozorování, analýza prací, které vytvořili žáci během dané dvouhodinovky a analýza odpovědí ze získaných dotazníků. Celé vlastní ověření navržených úkolů v praktické části této diplomové práce probíhalo na ZŠ Úprkova v Hradci Králové během školního roku 2016/2017. Informatika je zde standardně vyučována ve dvouhodinových blocích (dále označovaných jako dvouhodinovky). Časová dotace hodin informatiky je v některých ročnících (pátý, šestý a sedmý ročník) jedna hodina týdně, proto jsou žáci rozděleni na poloviny, tyto skupiny se střídají, a mají, z pohledu žáka, dvě hodiny jednou za čtrnáct dní. Ve vyšších ročnících (osmý a devátý ročník) je informatika vyučována jako volně volitelný předmět s časovou datací dvě hodiny týdně (tj. jako jedna dvouhodinovka týdně)

Ověření praktické části diplomové práce proběhlo po dohodě s vyučujícím během dvouhodinovky. Výuka byla naplánována na pátek během páté a šesté vyučující hodiny. Pro ověření byl po dohodě s vyučujícím vybrán osmý ročník, protože se jedná o jejich volitelný předmět a v osmém ročníku mají žáci rozvinuté logické a abstraktní myšlení. Učitel tuto třídu hodnotí jako velmi různorodou, vesměs snaživou a kázeňsky zvládnutelnou.

Pro přípravu na samotnou výuku bylo vytvořeno pět úkolů. Prvním z nich bylo vytvoření pravidelných n-úhelníků. Konkrétně se jednalo o rovnostranný trojúhelník, čtverec, pětiúhelník, šestiúhelník a osmiúhelník. Pro jednoduché numerické dopočítání velikosti vnitřních úhlů byly vybrány právě tyto uvedené n-úhelníky. Pro vyřešení tohoto úkolu je nezbytné vědět vzoreček pro výpočet vnitřního úhlu, případně si tento vzorec umět dohledat na internetu. Z pohledu vytvoření algoritmické konstrukce je důležité, aby si žák uvědomil možnost použít cyklus s předem daným počtem opakování.

Na tento úkol navazoval druhý úkol. Zde bylo cílem vytvořit program, který se uživatele zeptá, jaký n-úhelník chce vykreslit, uživatel zadá číslo n a program tento n-úhelník vykreslí. Tento úkol klade na žáky vyšší nároky na algoritmické a abstraktní myšlení.

Třetím úkolem bylo pro žáky vytvořit program, kde se kočka SCRATCH pohybuje po bílém poli. V případě, že narazí na konci pole do zdi, se odrazí, změní směr a jde jinam. Celá hra končí v momentě, kdy uživatel stiskne mezerník. Během vypracování tohoto programu je důležité, aby si žák uvědomil jednotlivé podmínky uvedené v zadání a vhodným způsobem s nimi pracoval.

Čtvrtým úkolem bylo vytvoření pravidelné hvězdy, která má pět, devět nebo patnáct cípů Pro počet cípů u hvězdy byla záměrně volena tato čísla, pro snadné numerické dopočítání vnitřního úhlu v jednotlivých cípech. Tento úkol klade na žáky nároky zejména na jejich matematické přemýšlení, prostorovou představivost a sekvenční myšlení.

Pátý úkol volně navazuje na předešlý. Jedná se o zobecnění předešlého úkolu. Úkolem je vytvořit program, kde na začátku uživatel zadá počet cípů a program danou hvězdu vykreslí. Tento úkol byl plánován pouze pro žáky, kteří mají rychlejší pracovní tempo.

Úkol	Vstupní kompetence	Algoritmické konstrukce	Mezipředmětové vztahy
Pravidelné n_úhelníky 1	určení velikosti vnitřního úhlu	cyklus s předem daným počtem opakování	Matematika
Pravidelné n_úhelníky 2	určení velikosti vnitřního úhlu	cyklus s předem neznámým počtem opakování	Matematika
Kočka a mezerník		větvení, podmínky	Fyzika
Hvězda 1	určení velikosti vnitřního úhlu	cyklus s předem daným počtem opakování	Matematika
Hvězda 2	určení velikosti vnitřního úhlu	cyklus s předem neznámým počtem opakování	Matematika

Tabulka 6 Úkoly pro ověření ve výuce

12.2 Průběh výuky

Začátek samotné výuky byl organizačně náročnější. Protože k této skupině žáků, kteří měli mít výuku informatiky, byli připojeni žáci ze sedmého ročníku, kteří nejeli na lyžařský výcvik. Prvních pět minut trvalo, než se všichni usadili k počítačům. Žáci ze sedmého ročníku měli samostatnou práci, proto byli do ověřování navržených úkolů zařazeni žáci osmých ročníků. Počet žáků zapojených do ověřování byl dvanáct (1 dívka, 11 chlapců).

Každý žák dostal na začátku dvouhodinovky písemné zadání, kód umožňující další zpracování a na konci dvouhodinovky i anonymní dotazník pro zhodnocení dané dvouhodinovky. Na úvod dvouhodinovky proběhlo krátké seznámení s programovacím prostředím SCRATCH. Žáci si vyzkoušeli jednotlivé záložky a vybrané pokyny.

Jako první byl zařazen úkol na vytvoření programu, který by vykreslil pravidelný 3-úhelník, tj. rovnostranný trojúhelník. Tuto úlohu měli žáci za úkol nejprve vyřešit samostatně poté, co mají pocit, že se jim povede vytvořit tento program, se přihlásí. Tento úkol jsme si demonstrovali názorně. Jeden žák byl vybrán jako SCRATCH, druhý žák mu dával pokyny a První žák SCRATCH je měl plnit. Dále žáci plnili zadané úkoly samostatně s možností se kdykoliv zeptat. U tohoto úkolu žáky nejvíce bavilo měnit barvu a tloušťku pera pro vykreslení požadovaných n úhelníků.

Druhým úkolem bylo vytvoření programu, který se uživatele zeptá, na počet vrcholů u pravidelného n-úhelníku, uživatel zadá požadované číslo n, a kočka SCRATCH požadovaný n-úhelník vykreslí.

Jako třetí úkol byl zařazen program, ve kterém by se kočka SCRATCH pohybovala nahodile a pokud by narazila do zdi, otočí se a jde dál. Celý program může kdykoliv uživatel ukončit stisknutím mezerníku. Během tvorby tohoto programu se žáci mezi sebou začali potichu bavit a ukazovat si své výtvory, ve třídě tak vznikl hluk, proto byli žáci několikrát napomenuti. Několik žáků zjistilo, že lze změnit vzhled postavy a tak místo kočky SCRATCH měli na obrazovce letadlo, černoch či jiná animovaná postava.

Jako poslední úkol byl zařazen program na vytvoření pěti, sedmi a patnácti cípe hvězdy. Protože byl tento úkol zařazen na konci dané dvouhodinovky a pozornost žáků byla již značně snížená, byl tento úkol pro většinu z nich velmi obtížný.

Během dvouhodinovky se ukázala rozdílná úroveň v pracovním nasazení jednotlivých žáků i v úrovni jejich matematických a algoritmických dovedností.

12.3 Zhodnocení výuky

12.3.1 Reflexe na konci hodiny

Na konci hodiny jsem provedla sebereflexi hodiny. Jako podklad k této sebereflexi jsem použila vybrané otázky z diplomové práce Anny Culkové [54], která uvádí několik oblastí pro po hospitační rozbor. Jedná se o oblasti přípravy a řízení výuky, hodnocení žáků, kázně a sebehodnocení. Pro potřeby této diplomové práce byly vybrány níže uvedené otázky.

1 Jaké výukové cíle jste si stanovil pro tuto vyučovací hodinu?

2 Byli si i žáci vědomi cílů, kterých mají ve výuce dosáhnout?

3 Připravil jste si úlohy potřebné ke zjištění, zda bylo cílů dosaženo?

4 Byly vaše pokyny a výklady jasné a přiměřené z hlediska věku žáků?

5 Stačíte v průběhu hodiny sledovat pokrok žáků v učení, případně upravovat učební postupy tak, aby bylo dosaženo výukových cílů?

6 Pomáhá způsob řízení výuky udržovat a podněcovat zájem žáků?

7 Stačíte sledovat pokrok jednotlivých žáků a poskytovat individuální pomoc?

8 Mohou žáci vnímat váš zájem o pokrok každého žáka?

9 Je vaše komunikace se žáky cílevědomě zaměřena na udržování kladného klimatu třídy?

Odpovědi na otázky

1 Výukovým cílem bylo naučit žáky základní orientaci v prostředí SCRATCH. Žáci umí vytvořit v prostředí SCRATCH pravidelný n-úhelník. Žák zvládne

vytvořit animaci. Žák zvládne použít cyklus s předem daným počtem opakování.

2 Ano, byli si vědomi cíle hodiny, protože na začátku hodiny jsem tento cíl hodiny řekla.

3 Pro zjištění, zda bylo dosaženo daných výukových cílů, byly využity dva prostředky pro hodnocení. Prvním z nich bylo, že žáci své práce odesílali na můj pracovní email a druhým prostředkem byl dotazník na konci hodiny.

4 Snažila jsem se, aby byly.

5 Sledování činnosti žáka je základem pro dobrou pedagogickou práci. Proto se snažím práci každého žáka sledovat pokud možno systematicky. Proto procházím během času, kdy mají žáci zadanou samostatnou práci, třídou, případně ze zadní řady sleduji práci jednotlivých žáků na jejich monitorech.

6 Zde je z mého pohledu důležitá práce s chybou. Snažím se vést výuku tak, aby žáci měli možnost nad svojí chybou přemýšlet a zkusili sami přijít na způsob opravy, případně se doptali svých spolužáků.

7 S žáky mám vždy stanovené pravidlo, že pokud něco neví, něco jim není jasné, srozumitelné, ať se hned zeptají. Navíc jsem od učitele, který tuto skupinu učí, věděla o dovednostech jednotlivých žáků.

8 Jedním z prostředků, které k tomu během výuky používám, je oslovování žáků jejich křestními jmény. Pokud se žákovi povede vyřešit problém, nad kterým pracoval nějakou dobu a vyvinul pro řešení značné úsilí, tak chválím za toto úsilí, někdy i před celou třídou.

9 Mezi znaky kladného klimatu ve třídě patří vzájemná důvěra mezi žákem a učitelem, pocit bezpečí a odvaha žáka se zeptat. Každý žák má právo se během hodiny zeptat na cokoli, co souvisí s probíranou látkou nebo organizací výuky. Pokud se zeptá na něco, co je v probíraném učivu důležité, tak jej pochválím. Nikdy cíleně nesnižuju hodnotu žáka, tím, že bych mu řekla, že je neschopný, hloupý a nemá ty správné schopnosti.

12.4 Hodnocení žákovských prací

Žáci své práce posílali na můj pracovní email jako přílohu. Během hodnocení výsledků jejich práce jsem u každého žáka prošla všechny jím vytvořené programy v prostředí SCRATCH. Pro přehlednost zaznamenávání výsledků jsem použila tabulkový procesor Excel. Tabulka je uvedena v příloze. [?]

Nejčastěji se podařilo

Z hodnocení žákovských prací vyplynulo, že žáků se nejvíce dařilo vytvořit úkol č. 3 s názvem Kočka klávesnice. Dokonce i během průběhu bylo zřejmé, že žáky tento úkol baví.

Extra se podařilo

Někteří žáci během hodiny své práce vylepšili nebo doplnili o nějaké prvky, které nebyly zadány. Příklady: žák s kódem 103-8AB-21 vytvořil program pro mazání pracovní plochy, navíc u tvorby pravidelného n-úhelníka zjistil, že pro n = 45 SCRATCH vytvoří kružnici.

Žák (103-8AB-24) v úkolu Kočka a klávesnice přidal obrázek (Sprite) Letadla, ve kterém tato kočka SCRATCH létala. Žák (103-8AB-25) také přišel na podmínku, že pro n = 45 program SCRATCH vykreslí kružnici, navíc u druhého úkolu poté co je vykreslen požadovaný n-úhelník, řekne kočka SCRATCH co nakreslila. Dva žáci (103-8AB-21, 103-8AB-27) začali pracovat na rozšiřujícím úkolu. Oběma se podařilo vykreslit jeden typ hvězdy. Vybrané vzorově řešené žákovské práce jsou uvedeny v příloze.

Nejčastější problémy

Žákům největší problémy dělalo využívat opakování a cykly v programech. Například u prvního úkolu, tvorba pravidelného n-úhelníku, opakování nepoužilo pět žáků. Tato skutečnost se projevila na vypracování druhého úkolu, který nevypracovali čtyři žáci.

12.4.1 Dotazníky

Vyplnění dotazníku proběhlo na konci dvouhodinovky. Žáci odpovídali anonymně. Dotazník byl rozdělen do tří tematických bloků. První z nich se týkal programovacího prostředí SCRATCH. Náplní druhého bloku otázek bylo zjištění jejich povědomí o programování a jejich dosavadní zkušenosti s programováním. Poslední blok otázek byl zaměřen na oblast logického a sekvenčního myšlení. Jeden žák odevzdal prázdný dotazník, proto počet respondentů je jedenáct. Nevyplněný dotazník spolu s přehlednými výsledky je uveden v příloze.

12.4.2 Programové prostředí SCRATCH očima žáků

Žáci měli na číselné stupnici od jedné do deseti ohodnotit programové prostředí SCRATCH, když deset je maximum. Grafické rozhraní prostředí SCRATCH žáci průměrně ohodnotili na 4,62. Zde se projevil velký rozdíl v bodování u jednotlivých žáků. Grafické rozhraní získalo 9, 7, 6, 5 a 1 bodů.

U hodnocení náročnosti jednotlivých úkolů jeden žák všechny úkoly ohodnotil deseti body, přestože během hodiny věnoval pozornost mnoha věcem mimo výuku. Tento fakt bylo možné zpětně dohledat. Žáci měli své práce uložit do složky, učitel pak vidí uživatelské jméno toho, kdo danou práci uložil. V následující tabulce je přehled průměrné náročnosti daných úkolů podle žáků. Hodnotící stupnicí byla opět škála od jedné do deseti, kde deset je maximum.

Název úkolu	N-úhelníky	N-úhelníky obecně	Klávesnice a myš	hvězda
Obtížnost	3,45	5	3,64	5,64

Tabulka 7 Obtížnost úkolů očima žáků

Na otázku, co konkrétního bylo během hodiny největší problém, odpovídali žáci různě. Tři žáci napsali, že všechno. Dva naopak neměli žádný problém, dva žáci na tuto otázku neodpověděli, jeden žák uvedl, že problémem byla jeho lenost. Tři žáci uvedli, že problém pro ně bylo vytvoření hvězdy. Jeden žák uvedl, že měl problémy se soustředěním. Vybrané vyplněné dotazníky jsou uvedeny v příloze.

12.4.3 Druhý blok otázek – Programování očima žáků

Odpovědi žáků na otázku, co je to programování, by se daly rozdělit do dvou skupin. První by vystihovala subjektivní hodnocení žáků, co to programování je, a druhou skupinu tvoří odpovědi, kde se žáci pokoušejí popsat programování a jeho funkci. Dva žáci neodpověděli. Čtyři žáci uvedli, že programování je těžké, sranda, super. Šest žáků se pokusilo o odborný popis toho, co je programování.

Druhá otázka se žáků ptala, jaké mají zkušenosti s programováním. Tři žáci na tuto otázku neodpověděli, tři žáci uvedli, že zkušenosti s programováním mají pouze ze školy. Jeden žák uvedl, že neví jaké má zkušenosti s programováním. V oblasti užívání počítače převládá u této skupiny uživatelská úroveň. Žáci používají počítač k hraní her, streamování, sledování youtube, případně referáty do školy. Na otázku, proč se učit programovat, mě zaujaly dvě odpovědi. První žák odpověděl prakticky a pragmaticky, protože se to bude hodit. Druhý žák odpověděl: „Aby z nás něco bylo."

12.4.4 Řešení logické úlohy

Posledním úkolem v dotazníku měli žáci za úkol překreslit známý domeček jedním tahem a napsat popis. Jeden žák tento úkol nevyplnil. Dva žáci tento úkol nedokončili. U dalších dotazníků dvou žáků chybí postup. Žáci nejčastěji očíslovali hrany, někteří přidali i šipky. V příloze je jako ukázka uvedeno řešení od žáků 103-8AB-27, 103-8AB-40. Dva žáci (103-8AB-37, 103-8AB-22) postup sepsali obdobným způsobem jako geometrickou konstrukci v matematice. Ukázky jejich zápisu řešení jsou také uvedeny v příloze.

13 Zhodnocení cílů pro empirickou část

Pro dvouhodinovku bylo naplánováno celkem pět úloh, s tím, že čtyři budou shodné pro všechny. A jedna obtížnější úloha bude pro bystřejší žáky, kteří mají rychlejší pracovní tempo. Toto časové rozplánování se ukázalo jako správné a z časového hlediska realistické. V následující tabulce je jednoznační porovnání očekávané časové náročnosti a reálného času, který žáci potřebovali pro vypracování jednotlivých úloh. Reálná časová náročnost je přibližným průměrem času, který žáci potřebovali k vypracování dané úlohy. Tento čas je zaokrouhlen na

celé minuty. Úloha „Kočka a mezerník“ mnohá žáky bavila, proto vymýšleli různé vylepšení a proto strávili nad touto úlohou záměrně více času.

Název úlohy	Očekávaná časová náročnost	Reálná časová náročnost
Pravidelné n-úhelníky	10 minut	13 minut
Pravidelné n-úhelníky obecně	15 minut	21 minut
Kočka a mezerník	15 minut	10 minut
Pravidelná hvězda	20 minut	30 minut

Tabulka 8 Porovnání časové náročnosti

Druhým cílem bylo ověření, zda navržené úkoly mají odpovídající úroveň obtížnosti. Splnění tohoto úkolu je obtížně měřitelné. Jedním z kritérií je porovnání očekávané a reálné časové dotace a druhým kritériem je hodnocení jednotlivých žákovských prací.

U každé práce bylo hodnoceno, zda žák použil správnou algoritmickou konstrukci, zda program, který vytvořil, není z logického pohledu chybný, či nevykonává nesmyslné pokyny a počet pokynů, které žák zadal programu. Shrnutí hodnocení jednotlivých žákovských prací je uvedeno v následující tabulce.

	n-úhelník	n-úhelník obecně	kočka a klávesnice	hvězda
103-8AB-21	funkční program	funkční	funkční	fuknční
103-8AB-23	nevyhovující formát	nevyhovující formát	nevyhovující formát	nevyhovující formát
103-8AB-29	chybí opakování	není	nefunkční	není
103-8AB-22	chybí opakování	pro každý typ extra	funkční	část cyklu
103-8AB-37	chybí opakování	pouze n=5	více programů	není
103-8AB-24	nevyhovující formát	není	funkční	není
103-8AB-40	chybí opakování	není	nefunkční	není
103-8AB-27	chybí opakování	není	není	část cyklu
103-8AB-38	nevyhovující formát	nevyhovující formát	nevyhovující formát	nevyhovující formát
103-8AB-39	funkční program	pouze část cyklu	funkční	není
nevyplněn	funkční program	pro každý typ extra	funkční	část cyklu
nevyplněn	funkční program	nevyhovující formát	nevyhovující formát	není

Tabulka 9 Hodnocení jednotlivých žákovských prací – shrnutí

Dva žáci veškeré programy uložili v nevyhovujícím formátu, který se zpětně nepodařilo otevřít. Nejčastěji žákům dělalo problémy během programování využít cyklus. Tato skutečnost se projevila zejména v první úloze. Poslední úlohu, část

žáků nestihla dokončit pro rozdílné pracovní tempo. V této úloze měli žáci největší problém s výpočtem konvexního úhlu v jednom cípu dané hvězdy. Mnohým nepomohlo ani doporučení, na kterých internetových stránkách mohou tento vztah a vzoreček pro výpočet těchto úhlů najít.

Závěr diplomové práce

Dnešní svět se rychle mění. Jedním z oborů, který podněcuje a vede ke změnám, je informatika. Informační technologie získávají své nezastupitelné místo prakticky ve všech oborech lidské činnosti. Proto škola jako instituce, která má připravovat mladé lidi do života, se snaží na tento pokrok reagovat adekvátním způsobem. Pro úspěšný a kvalitní způsob života je nezbytné rozvíjet dovednosti žáků. Jedna z nich je schopnost neustále se učit nové věci a zpracovávat informace.

Prvotním cílem teoretické části byla literární rešerše, která měla definovat základní pojmy problematiky spojené s výukou informatiky, konstruktivismu a badatelsky orientované výuky, rozvojem logického a abstraktního myšlení během výuky. Cíle praktické části byly dva. Prvním z nich bylo stručné představení historie programování a jeho vývoje. Druhým cílem bylo vytvoření stručného literární rešerše, která poskytuje přehled v základních pojmech, které souvisejí s programováním.

Pro empirickou část práce bylo stanoveno několik cílů. Jedním z nich bylo stručné představení programovacího prostředí SCRATCH, dalším bylo vytvoření sbírky úloh, které by sloužila jako inspirace pro učitele. Vybrané úlohy byly ověřeny v rámci experimentální výuky.

Všechny vytyčené cíle jsem podle svého názoru splnila. Hodnotit kvalitu jejich naplnění již není mým úkolem.

Ke všem úlohám, které jsou uvedeny v této práci, je na přiloženém CD vytvořeno vzorové řešení. Tyto úlohy mohou sloužit učitelům jako inspirace pro další práci s programem SCRATCH ve výuce. Všechny úlohy je možné spustit online na adrese https://scratch.mit.edu/projects/editor/?tip_bar=getStarted.

Pro případné zájemce je umožněn online přístup na webu scratch.mit.edu. Vypracované úlohy jsou zveřejněné pod jmény uvedenými v diplomové práci. Všechny úlohy jsou zpřístupněny pod uživatelským účtem dpkunhartova. Vše dostupné na následující adrese https://scratch.mit.edu/mystuff/#shared.

Diplomová práce může sloužit jako inspirace pro vyučující informatiky, kteří by rádi začali vyučovat programování, případně hledají další podněty pro svoji práci a možné zpestření pro žáky během vyučování.

Výuka programování má svůj významný vliv na rozvoj logického, sekvenčního a abstraktního myšlení. Možná proto je programování mnoha žáky vnímáno jako náročné a určené pouze pro hrstku výjimečných jedinců. Tento názor jim může pomoci změnit program SCRATCH, který díky svému příjemnému grafickému rozhraní a intuitivnímu ovládání umožňuje uživateli plně se soustředit na tvorbu programu a jeho logickou strukturu. Navíc díky možnosti vložení zábavných prvků (změna kostýmů, vložení zvuku) umožňuje tvorbu mnoha pro žáky atraktivních programů.

Zkušeností a dovedností, které jsem získala během psaní diplomové práce, bych ráda využila ve své budoucí pedagogické praxi.

Seznam použité literatury

[1] Národní program rozvoje vzdělávání v České republice: bílá kniha. Praha: Tauris, 2001. ISBN 80-211-0372-8.

[2] MAŇÁK, J., JANÍK T. a ŠVEC V.: *Kurikulum v současné škole*. Brno: Paido, 2008. Pedagogický výzkum v teorii a praxi. ISBN 978-80-7315-175-1.

[3] Rámcový vzdělávací program pro základní vzdělávání. Stařeč: Infra, 2004. ISBN 80-86666-24-7.

[4] Průcha, J., Walterová, E., Mareš, J.: *Pedagogický slovník*. Praha, Portál 2003. ISBN 80-7178-772-8 .

[5] ŠKODA, J.: *Současné trendy v přírodovědném vzdělávání*. 1. vyd. Ústí nad Labem: Univerzita J. E. Purkyně v Ústí nad Labem, 2005. ISBN 80-7044-696-X.

[6] JANÁS, J.: *Mezipředmětové vztahy a jejich uplatňování ve fyzice a chemii na základní škole*. 1. vyd. Brno: Univerzita J. E. Purkyně v Brně, 1985. ISBN 55-965-85.

[7] Hartl, P., Hartlová, H.: *Psychologický slovník*. Praha: Portál, 2000. ISBN 978-80-7367-569-1.

[8] Nevalová, D. a kol. *Konstruktivismus a jeho aplikace v integrovaném pojetí přírodovědného vzdělávání – Úvodní studie*. 1.vyd. Olomouc, Univerzita Palackého, 2006. ISBN 80-244-1258-6.

[9] Cachová, J., Stehlíková, N., *Konstruktivistické přístupy k výuce a praxe* [online], Jednota českých matematiků a fyziků 2006 [cit. 1. 2. 2017]. Dostupné z: class.pedf.cuni.cz/NewSUMA/FileDownload.aspx?FileID=89.

[10] Lokša, J., Lokšová, I.: *Pozornost, motivace, relaxace a tvořivost dětí ve škole,* Praha: Portál. 1999. ISBN 80-7178-205-X.

[11] Hejný, M.; Stehlíková, N.: *Číselné představy dětí*. Praha, PedF UK. 1999. ISBN 80-86039-98-6.

[12] Hejný, M., Novotná, J., Stehlíková, N. *25 kapitol z didaktiky matematiky*, Praha UK: Pedagogická fakulta, 2004. ISBN 80-7290-189-3.

[13] Bílek, M.: *Projekt MaSciL*, [online] mascil projekt, 2013, [cit. 1. 4. 2017]. Dostupné z: https://ris2.uhk.cz/mascil/project.html

[14] Papáček M.: *Badatelsky orientované přírodovědné vzdělání – cesta pro biologické vzdělávání generací Y, Z a alfa?* Scienta in educatione. 2010, roč. 1, č. 1, s. 33 – 49, ISSN 1804 – 7106. Dostupné z: http://www.scied.cz/index.php/scied/article/viewFile/4/5

[15] Průcha, J.: *Metody tvůrčího myšlení*, Akademie Jana Ámose Komenského, Praha, 1993. ISBN: 80-704-8070-X.

[16] Reference, *What is logical thinking?* [online] [cit. 21. 2. 2017] dostupné z: https://www.reference.com/world-view/logical-thinking-4de952f121bd4376#.

[17] The critical thinking community, *definic critical thing,*[online] [cit. 14. 2. 2017] dostupné z: http://www.criticalthinking.org/pages/defining-critical-thinking/766

[18] Eales – Reynolds L-J, Judge B., McCreery E., Jones P., *Critical thinking for education student, 2. vyd. 2013,* ISBN: 9781446268414.

[19] Butterworth, J., Thwaites, G., *Thinking skills Critical thinking and Problem Solving,* 2. vyd, 2013, Cambridge university press, ISBN 97811107606302.

[20] Rámcový učební plán. *Digifolio.rvp.cz*. [online], Praha, 2. ledna 2016, [cit. 18. 11. 2016]. Dostupné z: http://digifolio.rvp.cz/view/view.php?id=10846

[21] VÁCLAVÍK, V.: *Kolikrát jsme gramotní?* [online] Hradec Králové, 3. 9. 2000, [cit. 18. 11. 2016] dostupné z: http://kmen.uhk.cz/kmen/dvpp/clanky/gramotnosti.html

[22] ALTMANOVÁ, J., FALTÝN, J., NEMČÍKOVÁ K. a ZELENDOVÁ E. (eds.). *Gramotnosti ve vzdělávání: příručka pro učitele*. [online] Praha: Výzkumný ústav pedagogický, 2010.[cit. 18. 11. 2016] ISBN 978-80-87000-41-0. Dostupné z: http://www.vuppraha.cz/wp-content/uploads/2011/03/Gramotnosti-ve-vzdelavani11.pdf

[23] Blažek, R.: *Gramotnost: definice pojmu.* [online], Praha, 16. 1.2014, [cit. 18. 11. 2016] dostupné z: http://slideplayer.cz/slide/2742129/

[24] American Library Association. *Presidential Commission on Information Literacy. Final report.*[online], Washington, D. C., January 10, 1989. [cit. 18. 11. 2016] Dostupné z: http: http://www.ala.org/acrl/publications/whitepapers/presidential

[25] Česká školní inspekce, *Metodika pro hodnocení rozvoje informační gramotnosti.*[online], Praha, červen 2015. [cit. 18. 11. 2016]. Čj.: ČŠIG-2981/15-G2. Dostupné z: http://www.niqes.cz/Niqes/media/Testovani/KE%20STA%C5%BDEN%C3%8D/V%C3%BDstupy%20KA1/IG/Metodika-pro-hodnoceni-rozvoje-IG.pdf

[26] Dombrovská, M., Landová, H. a Tichá, L., *Informační gramotnost – teorie a praxe v ČR,* [online], Národní knihovna ČR, 2004, roč. 15, č. 1, s. 7 – 18, [cit. 18. 11. 2016]. ISSN 1214-0678. Dostupné z: http://full.nkp.cz/nkkr/NKKR0401/0401007.html

[27] MŠMT, *Strategie digitálního vzdělávání do roku 2020*, [online], Ministerstvo školství, mládeže a tělovýchovy, 31. Října 2014,[cit. 18. 11. 2016]. Dostupné z: http://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-digitalniho-vzdelavani-do-roku-2020

[28] MATUCHA, J. *Programovací jazyky: Historie počítačů* [online]. Olomouc, 2006 Absolventská práce. Vyšší odborná škola a Střední průmyslová škola elektrotechnická Olomouc, Vedoucí práce Jiří Rudolf a Josef Kolář. [cit. 7. 2. 2017]. Dostupné z: <http://kormus.cz/mvt/data/historie_pocitacu_matucha.pdf>

[29] Dalakov G, *Biography of John Napier* [online]. History-computer.com [cit. 7. 2. 2017]. Dostupné z : http://history-computer.com/People/NapiersBio.html

[30] Nygrýn, P., *Historie počítačů: Od elektronky po internet* [online] zive.cz [15. srpna 2011] živě.cz [cit. 7. 2. 2017]. Dostupné z: http://www.zive.cz/Clanky/Historie-pocitacu-Od-elektronky-po-internet/sc-3-a-147343/default.aspx

[31] PELIKÁN, J. *Historie počítačů* [online] fi.muni.cz [cit. 7. 2. 2017] Dostupné z: http://www.fi.muni.cz/usr/pelikan/ARCHIT/TEXTY/HISTOR.HTML

[32] PINTER, T., *Výuka programování na základní a střední škole* [online] fi.muni.cz [cit. 2017-02-07]. Dostupné z: http://www.fi.muni.cz/~tomp/semuc/text_pitner.html

[33] MUSÍLEK, M., *Kapitoly z dějin informatiky*. Hradec Králové: Gaudeamus, 2011. ISBN 978-80-7435-129-7.

[34] KŘÍŽ, P., *Vývoj programování a programovacích jazyků.* [online] fi.muni.cz, 2002 [cit. 2017-02-07]. Dostupné z: http://www.fi.muni.cz/usr/jkucera/pv109/2002/xkriz1.htm

[35] Bechyňová, M, *1. Úvod algoritmus a programovací jazyky*, [online] Gymnázium Vlašim, [5. 10. 2012] [cit. 23. 2. 2017] Dostupné z: http://www.ivt.mzf.cz/algoritmizace-a-programovani/uvod-do-algoritmu/1-uvod-algoritmus/

[36] MILKOVÁ, E. *Algoritmy: objasnění, procvičení a vizualizace základních algoritmických konstrukcích,* Praha, Alfa nakladatelství 2008. ISBNA 978-80-87197-10-3.

[37] Krátký, M., Dvorský, J., *Úvod do programování* [online], Vysoká škola báňská – Technická univerzita Ostrava, 2005, [cit. 23. 2. 2017] Dostupné z: http://www.cs.vsb.cz/kratky/courses/2004-05/udp/presentation/udp-2_6.pdf

[38] Běhálek, M., *Programovací jazyky*, [online], VŠT-TUO, [cit. 23. 2. 2017] Dostupné z: http://wiki.cs.vsb.cz/images/4/4c/Progjaz.pdf

[39] PŠENČÍKOVÁ, J., *Algoritmizace*. Vyd. 2. Kralice na Hané: Computer Media, 2009. ISBN 9788074020346.

[40] Bechyňová, M., 2. *Algoritmizace* [online] Gymnázium Vlašim, 3. 10. 2012, [cit. 23. 2. 2017] Dostupné z: http://www.ivt.mzf.cz/algoritmizace-a-programovani/uvod-do-algoritmu/2-algoritmizace/

[41] PADRTA, D., *Strukturované programování* [online]6. 9. 2001 [cit. 9. 3. 2017] dostupné z: http://www.isd.cz/pascal/2strukt.html#odkaz22

[42] Urban, O., *Cykly*, [online] Gymnázium Boženy Němcové, [cit. 23. 2. 2017] Dostupné z: http://www.gybon.cz/~urban/pascal/ucebnice/cykly.html

[43] Fojtík, R., *Moderní přístupy k výuce programování,* [online], Časopis pro technickou a informační výchovu, 1/2013 [cit. 9. 3. 2017], ISSN 1803-537X. Dostupné z: http://jtie.upol.cz/pdfs/jti/2013/01/08.pdf

[44] Halousková, A., *Učebnice jazyka Scratch*, [online]. Brno, 2012 [cit. 9. 3. 2017]. Diplomová práce. Masarykova univerzita, Vedoucí práce Tomáš Pitner. Dostupné z: http://is.muni.cz/th/346458/fi_m/DP_Halouskova.pdf.

[45] Musílek, M. *Dějiny informatiky 3*, [online], Univerzita Hradec Králové, duben 2010, [cit. 28. 2. 2017]. Dostupné z: http://black-hole.cz/cental/wp-content/uploads/2010/06/Dejiny_informatiky_3.pdf

[46] Wikipedia*, Logo (programming language),* [online], 3. 2. 2017, [cit. 28. 2. 2017]. Dostupné z: https://en.wikipedia.org/wiki/Logo_(programming_language)

[47] Scratch, *Community statistics at a glance,* [online], [cit. 1. 3. 2017]. Dostupné z: https://scratch.mit.edu/statistics/

[48] MIT Media Lab, *Programing concepts and skills supported in SCRATCH,* [online].[cit. 8. 3. 2017]. Dostupné z: http://web.media.mit.edu/~mres/scratch/documentation/Scratch-CS-concepts.pdf

[49] ŠIMONÍK, O.: *Úvod do didaktiky základní školy,* Brno, 2005. ISBN 80-86633-330.

[50] FISHER, R., *Učíme děti myslet a učit se: Praktický průvodce strategiemi vyučování,* Portál 1997. ISBN 80-7178-120-7.

[51] Voříšek, L., *Bill Gates a Mark Zuckerberg mluví o novém projektu Code.org pro online výuku programování, zkuste jej také,* [online], cdr.cz, 27. 2. 2013, [cit. 18. 11. 2016]. Dostupné z: http://cdr.cz/clanek/predstaveni-codeorg-vyuka-programovani-pro-kazdeho-zdarma-a-online

[52] Klement, M., Klement, J., Lavrinčík, J., *Metody optimalizace a hodnocení výuky základů programování,* [online] 2014, Olomouc, [cit. 18. 11. 2016]. Dostupné z: http://www.kteiv.upol.cz/uploads/soubory/publikace_wos_2014/Klement_PDF%20UP_metodika%20PROS.pdf

[53] Schön, O, *Programovat může každý, stačí chtít a být zvědavý,* [online] 20. 10. 2016, Hospodářské noviny, [cit. 18. 11. 2016]. Dostupné z: http://archiv.ihned.cz/c1-65483860-programovat-muze-kazdy-staci-chtit-a-byt-zvedavy#disqus

[54] Culková, A., *Hospitace a její funkce*, [online]. Brno, 2008 [cit. 9. 3. 2017]. Bakalářská práce. Masarykova univerzita, Vedoucí práce Jan Šťáva. Dostupné z: https://is.muni.cz/th/94041/pedf_b/Bakalarska_prace_-_AC.pdf

Seznam obrázků a tabulek

Všechny uvedené obrázky byly vytvořeny jako podklad pro tuto diplomovou práci v programech MS PowerPoint a Malování.

Seznam příloh

Zadání práce v prostředí SCRATCH Můj Kód:___ - _____-____

Pozorně si přečti zadání.
Zkus během tvorby programu použít co nejméně příkazů.
Jakékoliv postřehy, poznámky, vzorečky piš na tento papír. Odpověz na otázky. Pokud neznáš odpověď, zkus ji zjistit na internetu. Každý program bude začínat kliknutím na zelenou vlaječku.

- **Vytvoř program, který vytvoří pravidelné n-úhelníky. Pro n = 3, 4, 6, 8, 5. Tj. Vytvoř rovnostranný trojúhelník, čtverec, pravidelný šestiúhelník, osmiúhelník a pětiúhelník s délkou strany 100.**

  Zajisti, aby daný útvar se vešel na obrazovku celý.
  Jakou velikost mají vnitřní úhly u jednotlivých n-úhelníků?
  Jaký vzorec platí pro velikost vnitřních úhlů v zadaných n-úhelnících?
  Program ulož do svého počítače s názvem. „n-úhelníky“tvůj kód.

Vytvoř program, který se uživatele zeptá, jaký n-úhelník chce vytvořit. Uživatel napíše číslo a program tento požadovaný n-úhelník vytvoří.

Program ulož do svého počítače s názvem. „n-úhelníky obecně“tvůj kód.

Vytvoř program, ve kterém kočka Scratch ujde 10 kroků. Jde tak dlouho, dokud nenarazí do zdi. Když narazí do zdi, tak se odrazí a jde dál. Celou hru může kdykoliv ukončit uživatel, když stiskne mezerník.

Jaké podmínky musíš zohlednit při tvorbě programu?

Program ulož do svého počítače s názvem. „kočka a klávesnice“tvůj kód.

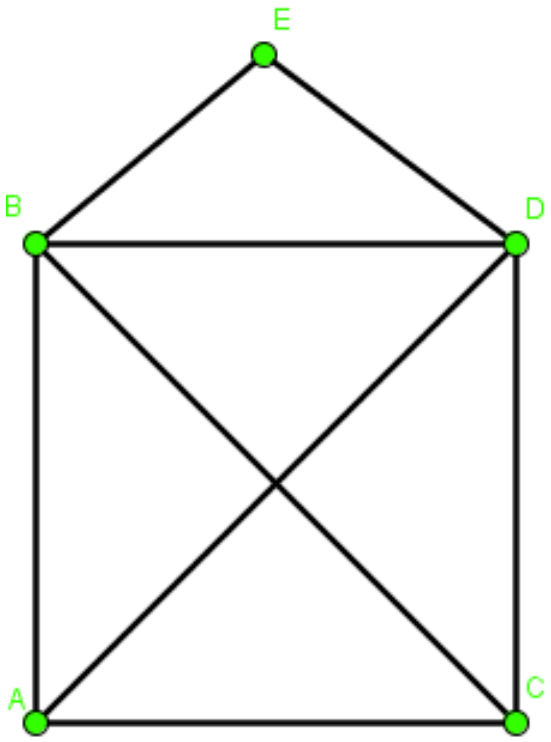
Vytvoř program, který vykreslí hvězdu s 5, 9 a 15 vrcholy. Délka hrany je 150.

Zajisti, aby daný útvar se vešel na obrazovku celý.
Jakou velikost mají vnitřní úhly u jednotlivých cípů hvězdy?
Jaký vzorec platí pro velikost vnitřních úhlů tyto cípy?
Program ulož do svého počítače s názvem. „Hvězdy“tvůj kód

Vytvoř program, který se uživatele zeptá, kolika cípou hvězdu chce vytvořit. Uživatel napíše číslo a program požadovanou hvězdu vytvoří. Pozor, uživatel musí zadávat pouze lichá čísla od 3 výše.

Program ulož do svého počítače s názvem. „Hvězdy obecně “tvůj kód.

Nakresli domeček jedním tahem. Napiš symbolický postup



Dotazník pro hodnocení výuky SCRATCHE Kód:

Na škále od 0 po 10 ohodnoť následující výroky. 10 je maximum.

Grafické rozhranní prostředí SCRATCH _____

Náročnost jednotlivých úkolů: N-úhelníky _____-

n- úhelníky obecně _____

kočka a klávesnice _____

Hvězda ____

Hvězda obecně _____

Co konkrétního mi během této hodiny dělalo největší problém?

Co je to programování?

Jaké máš zkušenosti s programováním?

K čemu nejčastěji používáš počítač?

Proč se učit programovat?

Dotazník pro hodnocení výuky SCRATCHE Kód: 103-8AB-22

Na škále od 0 po 10 ohodnoť následující výroky. 10 je maximum.

Grafické rozhraní prostředí SCRATCH 7

Náročnost jednotlivých úkolů: N-úhelníky 5

n- úhelníky obecně 4

kočka a klávesnice 3

Hvězda 3

Hvězda obecně 3

Co konkrétního mi během této hodiny dělalo největší problém?

SKORO VŠECHNO

Co je to programování?

UMĚNÍ, CO MÁ URČITÁ PRAVIDLA DĚLAT

Jaké máš zkušenosti s programováním?

SKORO ŽÁDNÉ

K čemu nejčastěji používáš počítač?

K PSANÍ

Dotazník pro hodnocení výuky SCRATCHE Kód: 103-8AB-22

Na škále od 0 po 10 ohodnoť následující výroky. 10 je maximum.

Grafické rozhraní prostředí SCRATCH 5

Náročnost jednotlivých úkolů: N-úhelníky 0

n- úhelníky obecně 0

kočka a klávesnice 0

Hvězda 1

Hvězda obecně 1

Co konkrétního mi během této hodiny dělalo největší problém?

Nic

Co je to programování?

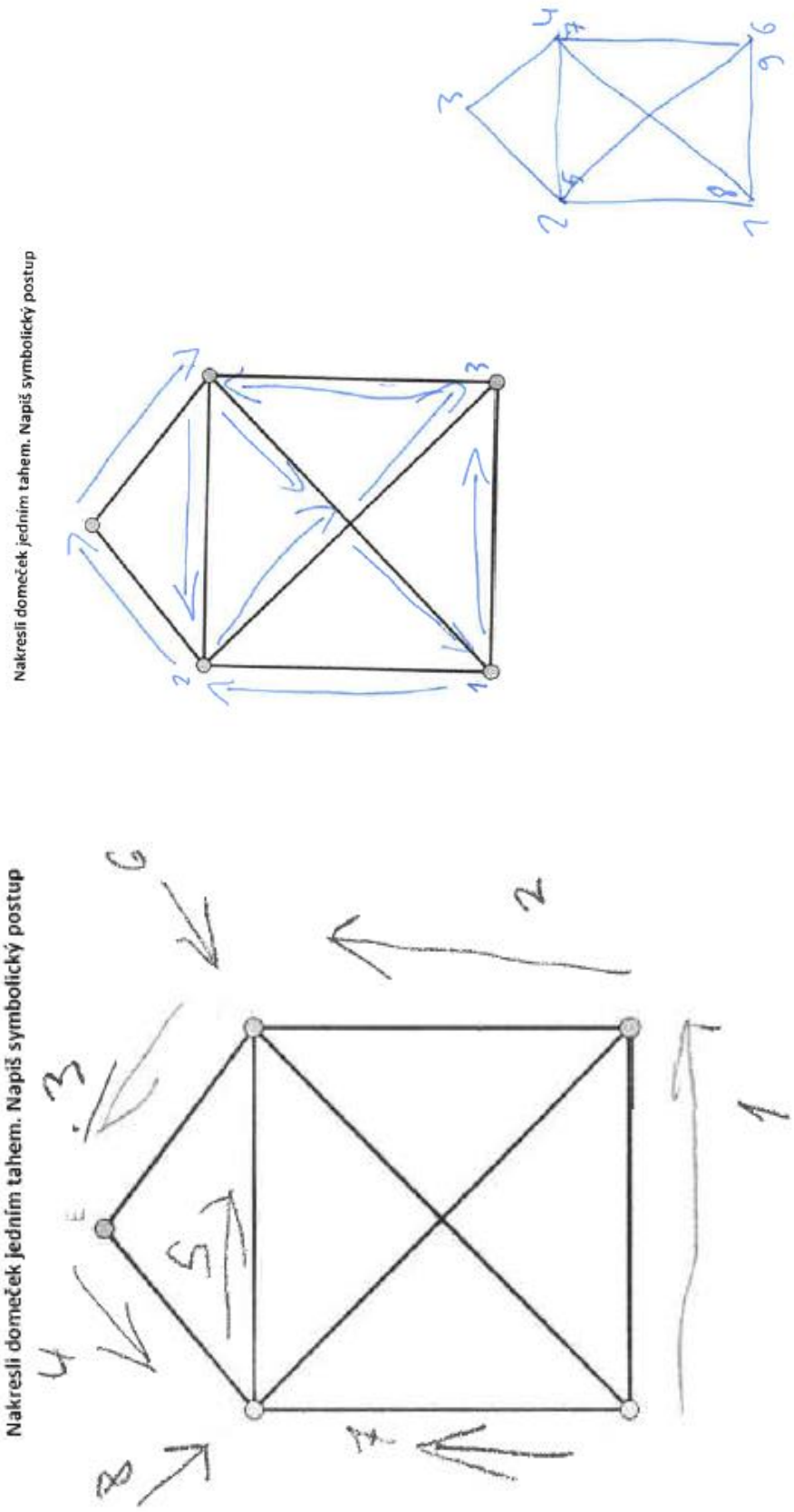
Asi nic

Jaké máš zkušenosti s programováním?

Jedné jsem se ve škole

K čemu nejčastěji používáš počítač?

K prezentacím

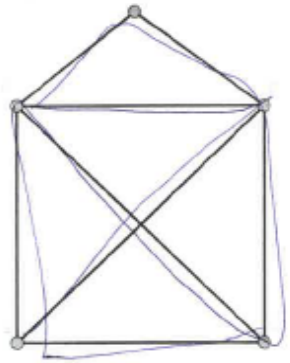


2 Žákovská řešení

Vytvoř program, který se uživatele zeptá, kolika cípou hvězdu chce vytvořit. Uživatel napíše číslo a program požadovanou hvězdu vytvoří. Pozor, uživatel musí zadávat pouze lichá čísla od 3 výše.

Program ulož do svého počítače s názvem. „Hvězdy obecně "tvůj kód.

Nakresli domeček jedním tahem. Napiš symbolický postup

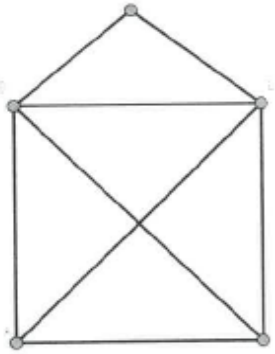


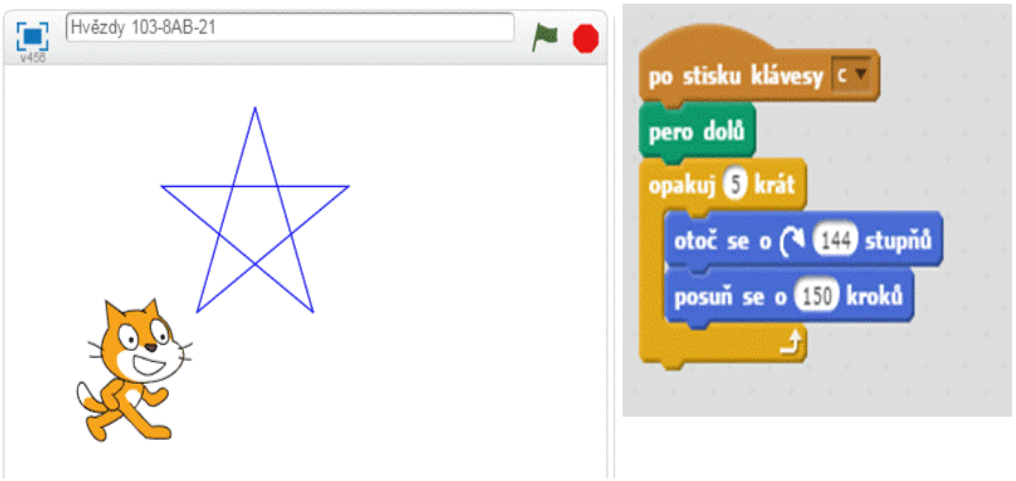
1. [redacted] A→D
2. D→B
3. B→E
4. E→D
5. D→C
6. C→B
7. B→A

Vytvoř program, který se uživatele zeptá, kolika cípou hvězdu chce vytvořit. Uživatel napíše číslo a program požadovanou hvězdu vytvoří. Pozor, uživatel musí zadávat pouze lichá čísla od 3 výše.

Program ulož do svého počítače s názvem. „Hvězdy obecně "tvůj kód.

Nakresli domeček jedním tahem. Napiš symbolický postup





3Ukázky vybraných žákovských prací v programu SCRATCH