Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Kurz programování v jazyce Scratch pro děti

2022

Magdaléna Skácelová

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

Studijní obor: Informatika pro vzdělávání, prezenční forma

Bibliografické údaje

Autor:	Magdaléna Skácelová
Název práce:	Kurz programování v jazyce Scratch pro děti
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2022
Studijní obor:	Informatika pro vzdělávání, prezenční forma
Vedoucí práce:	doc. RNDr. Miroslav Kolařík, Ph.D.
Počet stran:	48
Přílohy:	1 CD/DVD
Jazyk práce:	český

Bibliograhic info

Author:	Magdaléna Skácelová
Title:	Scratch programming course for children
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2022
Study field:	Computer Science for Education, full-time form
Supervisor:	doc. RNDr. Miroslav Kolařík, Ph.D.
Page count:	48
Supplements:	1 CD/DVD
Thesis language:	Czech

Anotace

Práce se zabývá výukou základů programování pro děti. Je zaměřená na vizuální programovací jazyky, které jsou pro děti intuitivnější a graficky poutavější. V první části jsou popsány příklady jazyků vhodných pro výuku programování u dětí. Ve druhém oddíle je představen vizuální programovací jazyk Scratch. Ve třetí části je uveden plán kroužku programování pro děti. Kroužek je rozdělen do tematických celků. V jednotlivých blocích jsou představeny programátorské principy. Výstupem z každého bloku je projekt, v němž žáci uplatňují nabyté kompetence. Vlastní řešení projektů jsou přístupná na webových stránkách Scratche i na přiloženém disku. Součástí plánu kroužku jsou metodické pokyny pro lektora kroužku.

Synopsis

This theses deals with teaching the basics of programming to children. It is focused on visual programming languages that are more intuitive and graphically engaging for children. The first part describes examples of languages suitable for teaching programming to children. In the second part, the visual programming language Scratch is introduced. In the third part, there is a plan of a programming course for children. The course is divided into thematic blocks. Various programming principles are presented in individual blocks. The output of each block is a project in which students apply the acquired competences. Solutions to these projects are available on the Scratch website and on the attached disc. Methodical instructions for the lecturer are part of the class plan.

Klíčová slova: Scratch; programování; kurz programování; děti

Keywords: Scratch; programming; programming course; children

Děkuji panu doc. RNDr. Miroslavu Kolaříkovi, Ph.D., za vedení a cenné rady při vypracovávání mé bakalářské práce.

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

datum odevzdání práce podpis autora

Obsah

1 Úvod **8**

2 Vybrané programovací jazyky pro výuku programování u dětí **9**
2.1 LEGO Mindstorms 9
2.2 Baltík 9
2.3 MakeCode 9
2.4 Python 10

3 Jazyk Scratch **11**
3.1 Historie jazyka Scratch 11
3.2 Popis prostředí Scratche 12

4 Program kroužku **15**
4.1 Úvod do kroužku 15
4.2 Základní znalosti – příběh 17
4.3 Cykly – kreslení pravidelných n-úhelníků 21
4.4 Podmínky 25
4.5 Proměnné 28
4.6 Vlastní projekt 32
4.7 Klonování 33
4.8 Funkce 39
4.9 Závěrečný projekt 43

Závěr **45**

Conclusions **46**

A Obsah přiloženého CD/DVD **47**

Literatura **48**

Seznam obrázků

1 Prostředí pro vytváření projektu ve Scratchi. Prostor pro programování je vyznačený červenou barvou. Paleta bloků vyznačená modrou barvou. Scéna vyznačená zelenou barvou. Část pro výčet postav a pozadí vyznačeny žlutou barvou. 12
2 Základní sekce bloků, které jsou barevně odlišené. 13
3 Další přidané sekce bloků, které jsou odlišeny piktogramem. . . . 13
4 Umístění tlačítka pro přechod do složky projektů uživatele 16
5 Ukázka složky projektů s projekty uživatele 17
6 Sdílený projekt. Šipky ukazují na tlačítko pro vytvoření kopie projektu a tlačítko pro ukázání postav projektu a jejich scénáře. . . 18
7 Možnosti bloku se objeví poté, co na něj klikneme pravým tlačítkem myši. 19
8 Komentář ve Scratchi, který se vždy váže na konkrétní blok scénáře. 19
9 Umístění vlajek na ploše. Zelená vlajka pouští projekt a červená projekt zastaví. 20
10 Tvar úvodního bloku zajišťuje první místo ve scénáři. 20
11 Místa pro přidání nové postavy a jiného pozadí 21
12 Místo pro přidání skupiny bloků Pero 23
13 Ukázka hvězd nakreslených ve Scratchi. 24
14 Vnitřní blok scénáře 26
15 Blok výrazu – protáhlý osmiúhelník. Nelze použít samostatně. . . 27
16 Zaoblený hodnotový blok. Nelze použít samostatně. 27
17 Scénář pohybu nahoru pro postavu Hledače. Pohyb do ostatních směrů se naprogramuje obdobně. 28
18 Bloky sekce Proměnné 30
19 Řešení hry na chytání balónku. 31
20 Příklad zanořený bloků „spoj x y“ 32
21 Umístění Batohu na ploše. 34
22 Scénář pro funkcionalitu padání kapky. 35
23 Scénář pro chycení kapky vody krabem. 36
24 Hotový scénář pro původní postavu kapky 38
25 Hotový scénář pro klony postavy kapky 38
26 Červená sekce Moje bloky 40
27 Úvodní blok pro scénář funkce 40
28 Okno pro vytvoření vlastního bloku. 40
29 Scénář pro vykreslení květu s použitím vlastního bloku „Vykresli lístek“ 41
30 Okno pro vytvoření vlastního bloku „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“, kde *barva* a *délka* jsou parametry. . . 41
31 Scénář pro funkci „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“ 42

32 Okno pro vytvoření vlastního bloku „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“, kde *barva* a *délka* jsou parametry. . . 42

1 Úvod

Počítače se staly běžnou součástí lidského života a pro jejich dobré pochopení a plné využití již nestačí pouze uživatelské znalosti, ale důležitá je i znalost programování. Dříve bylo programování pojímáno jako velmi složitá činnost, jejíž výstupy jsou skryty krypticky vyhlížejícími programovacími jazyky. Dnes však již něco takového platit nemusí, a programovat se mohou učit už děti na základních školách. Výborným nástrojem pro ně mohou být programovací jazyky založené na vizuálním skládání bloků na sebe. Jedním z nejpopulárnějších jazyků z této skupiny je Scratch.

Tato bakalářská práce představuje kroužek programování v jazyce Scratch pro děti v rámci mimoškolní zájmové činnosti. Účelem kurzu je pomoci vyučujícímu postupně děti naučit všechny důležité aspekty a součásti programování v tomto jazyce. Děti se v jeho průběhu dozvědí základní koncepty a ověří si je při vytváření vlastních programů i her.

2 Vybrané programovací jazyky pro výuku programování u dětí

V této kapitole jsou představeny vybrané programovací jazyky, které jsou vhodné pro výuku programování. Každý z těchto jazyků přináší odlišné výhody a správná volba záleží například na věku žáků a plánované aplikaci naučených dovedností.

2.1 LEGO Mindstorms

LEGO Mindstorms není přímo programovací jazyk, ale je to programovatelná stavebnice, využívající kostek LEGO. Její hlavní částí je programovatelná kostka s mikročipem, ke které lze připojit motory a senzory. Všechny součásti jsou plně propojitelné s běžnými díly stavebnice LEGO, což umožňuje dětem vytvářet různorodé stroje a roboty, a to ať už podle předchystaných návodů, nebo dle vlastní fantazie. Doporučený věk dětí je určen podle konkrétní stavebnice, často je věková hranice pro jejich užívání určena od 10 let výše.

K Legu Mindstorms je dodáván vizuální programovací jazyk založený na skládání bloků, který je velmi podobný jazyku Scratch. S pomocí programovatelných senzorů a motorů mohou děti vidět práci svých programů na pohybujícím se robotovi, a kromě programování tak mohou získat i základní znalosti robotiky. Vzhledem k rozsáhlé komunitě a podpoře systému Mindstorms existuje mnoho projektů, jimiž se lze inspirovat. Pokročilejší žáci mohou stavebnici programovat i v klasických programovacích jazycích Python, Java a jiných. Podpora těchto jazyků byla přidána komunitou fanoušků tohoto projektu. Nevýhodou daného systému může být jeho cena a nutnost dokupovat další senzory a motory pro stavbu složitějších robotů. [1, 2]

2.2 Baltík

Baltík je blokový programovací jazyk s grafickým výstupem, pomocí něhož se programuje chování čaroděje. Výhodou tohoto jazyka pro děti je grafické zpracování bloků a čeština v nastavení prostředí programování. Vzhledem ke grafickému zpracování bloků je jazyk vhodný i pro děti, které ještě neumí číst. S tímto programovacím jazykem u nás probíhají i soutěže pro studenty. Baltík je placený s licencí na jeden rok, a jelikož existují i neplacené možnosti programování pro děti, není tak populární pro školy. Rovněž není tak graficky moderní a zajímavý jako jiné programovací jazyky. Nevýhodou je nutnost programovat s čarodějem a úzký výběr grafického prostředí. [3]

2.3 MakeCode

MakeCode je blokový programovací jazyk od společnosti Microsoft určený dětem. Je velmi podobný jazyku Scratch a disponuje i podobným webovým prostředím. MakeCode je k dispozici zdarma a s jeho pomocí lze programovat mikropočítač

Micro:bit. Jeho velkou předností je možnost jednoduchého přepnutí napsaného kódu do jazyka Python nebo JavaScript. Na základě zmíněné funkce se starší děti mohou snadno naučit využívat i tyto jazyky. Jeho největší nevýhodou pro učení dětí u nás je, že není přeložený do češtiny. [4]

2.4 Python

Python patří mezi vhodné textové programovací jazyky pro výuku programování především díky své snadné čitelnosti. Python pro strukturu programu využívá odsazení – na rozdíl od závorek, které uplatňují jazyky jako Java nebo C. Vynucuje tedy přehlednost napsaného kódu. K čitelnosti pomáhají i klíčová slova jazyka, která jsou podobná běžné mluvě v angličtině. Výhodou je i dynamické typování proměnných, a proto odpadá potřeba dopředu znát a deklarovat typy.

Pro Python také existuje velké množství snadno použitelných knihoven, pomocí kterých mohou uživatelé získat potřebnou funkcionalitu pro své programy. Díky nim je Python dnes velmi populární například v oblasti strojového učení a zpracování dat. Znalost tohoto jazyka tedy zdaleka přesahuje pouze výukové účely, neboť studenti pro něj snadno mohou najít využití i v praxi. Stále se však jedná o textový programovací jazyk s nutností znát a dodržovat syntax, a proto bych jej pro výuku doporučila spíše pro vyšší stupně středních škol. [5]

3 Jazyk Scratch

Scratch je v současnosti jedním z nejznámějších programovacích jazyků pro děti – věkové rozmezí, pro něž je určen, je od 8 – 16 let. Jeho hlavním autorem je Mitch Resnick a je patentován skupinou Lifelong Kindergarten Group, spadající pod Massachusetts Institute of Technology (MIT). Programování ve Scratchi probíhá skládáním grafických bloků za sebe. Tím odpadá nutnost psaní správné syntaxe, která může nové žáky tradičních textově založených programovacích jazyků snadno odradit.

Platforma, kterou Scratch přináší, představuje vše potřebné pro vytváření grafiky, animací, her, interaktivních příběhů a jiných prvků. Vše je prováděno čistým, přímočarým způsobem se zářivými barvami, pochopitelnými popisky a zajímavými postavami. To vše vytváří ideální základ pro vstup dětí do světa programování.

Další výhodou Scratche je obrovská komunita, která se na něm podílí. V současnosti existuje přes 70 milionů registrovaných uživatelů, jež nasdíleli přes 82 milionů projektů. Tyto projekty mohou děti vyzkoušet a nechat se jimi inspirovat při tvorbě vlastních programů.

Existuje i zjednodušená verze Scratche pojmenovaná ScratchJr a je určena pro děti ve věku od 5 do 7 let.

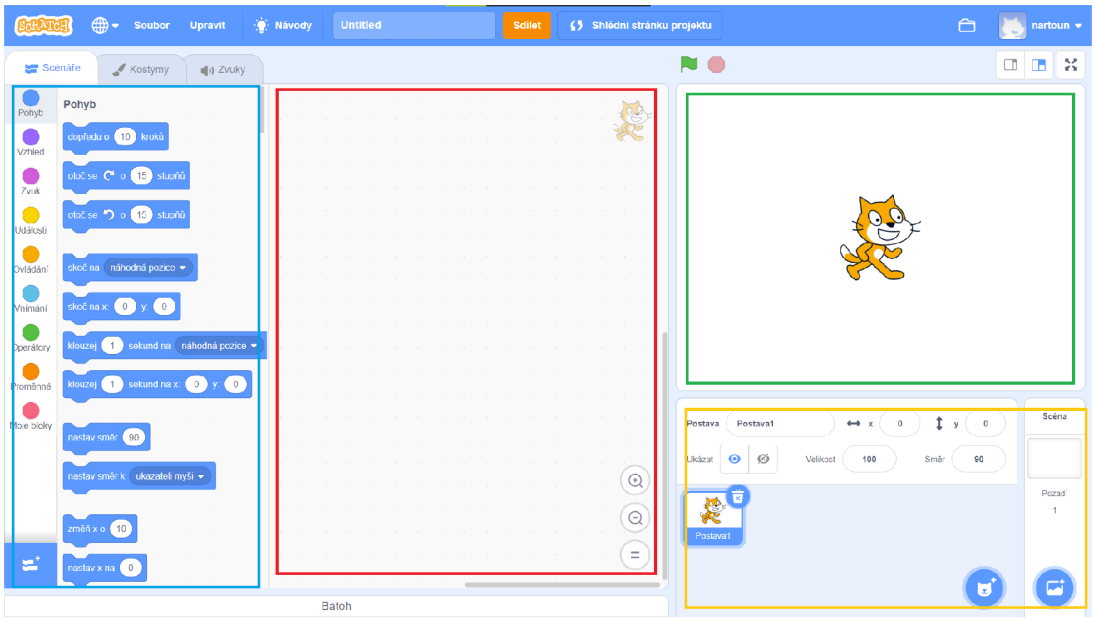
3.1 Historie jazyka Scratch

Od počátku roku 2000 vyvíjela skupina Lifelong Kindegarten (LLK), která je součástí MIT Media Lab, vizuální programovací jazyky pro děti. Mitchel Resnick, Yasmin Kafai a John Maeda získali grant National Science Foundation pro vývoj nového programovacího prostředí pro děti. LLK, vedená Michelem Resnickem ve spolupráci s týmem Yasmin Kafai na UCLA a Computer Clubhouses v Bostonu a Los Angeles, pracovali na vývoji Scratche. Vývoj probíhal s aktivním zapojením skupin dětí a mládeže v zájmových kroužcích Computer Clubhouses. Použití v těchto kroužcích sloužilo jako model pro ostatní zájmová centra jakožto ukázka toho, jak neformální výukové prostředí může pomoci k získání technologických dovedností.

Scratch byl zpočátku velmi základní programovací jazyk bez označených kategorií a bez zelené vlaječky – viz dále. Podobně jako AgentSheets zapojoval koncepty programování založeného na dotykovém ovládání a skládání bloků. Jeho úkolem byla výuka programování pro děti.

Filozofií Scratche jsou sdílení, znovupoužívání a kombinace částí kódu. S tím je spojen i slogan “Imagine, Program, Share” (vymysli, programuj, sdílej). Uživatelé mohou vytvořit své vlastní projekty nebo mohou zkombinovat již hotové projekty jiných uživatelů. Projekty vytvořené v jazyce Scratch podléhají licenci Creative Commons Attribution-Share Alike License a Scratch automaticky uvádí původního tvůrce projektu v horní části stránky projektu.

9. května roku 2013 byla vydána verze Scratch 2.0, která změnila vzhled prostředí a obsahovala jak online, tak i offline editor projektů. V rámci projektů



Obrázek 1: Prostředí pro vytváření projektu ve Scratchi. Prostor pro programování je vyznačený červenou barvou. Paleta bloků vyznačená modrou barvou. Scéna vyznačená zelenou barvou. Část pro výčet postav a pozadí vyznačeny žlutou barvou.

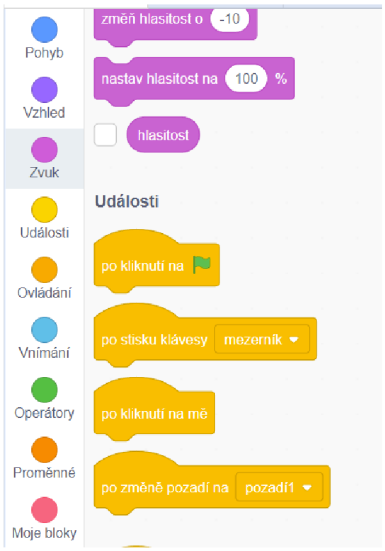
uživatelé mohli vytvářet vlastní bloky. Offline editor Scratch 2.0 byl dostupný ke stažení pro Windows, Linux a Mac z webových stránek Scratche. Existovala také neoficiální mobilní aplikace. Podpora Linuxu byla později opuštěna. V roce 2016 byla poprvé oznámena verze Scratch 3.0. Do ledna 2018 probíhaly testovací alfa verze a poté vznikla první beta verze použitelná ve většině internetových prohlížečů. První plná verze 3.0 byla vydána 2. ledna 2019. [6, 7]

3.2 Popis prostředí Scratche

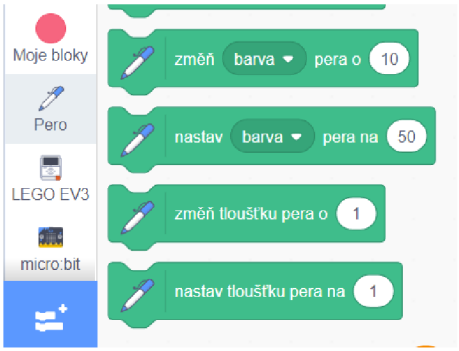
Pro Scratch existují dvě prostředí, –webové prostředí, nebo aplikace pro stolní počítače. Funkcionalita těchto prostředí je obdobná, ale můžou se lišit v určitých detailech.

Prostředí pro vytváření projektu ve Scratchi je rozdělené do několika částí - viz obrázek 1. Prostřední část je prostor pro programování, na obrázku 1 vyznačený červenou barvou. Do tohoto prostoru přetahujeme bloky, spojujeme je pod sebe, čímž vytváříme program neboli scénář. Všechny bloky jsou umístěny v paletě bloků nalevo, jež je na obrázku 1 vyznačená modrou barvou. Výstup projektu je vidět na scéně, která je na obrázku 1 vyznačená zelenou barvou. Pod scénou je umístěna část pro výčet postav a pozadí – na obrázku 1 vyznačeno žlutou barvou. Pod prostorem pro programování nalezneme tlačítko pro Batoh, kam lze ukládat části scénářů či celé postavy a přenášet je do jiných projektů.

Ve Scratchi jsou bloky pro přehlednost rozřazeny do skupin podle funkce.



Obrázek 2: Základní sekce bloků, které jsou barevně odlišené.



Obrázek 3: Další přidané sekce bloků, které jsou odlišeny piktogramem.

Tyto sekce jsou vypsány na bočním panelu nalevo od palety bloků – viz obrázek 2. Každá základní sekce je barevně odlišena a všechny bloky dané sekce mají právě tuto barvu.

Seznam základních sekcí bloků:

- modrá sekce Pohyb
- fialová sekce Vzhled
- růžová sekce Zvuk
- žlutá sekce Události
- oranžová sekce Ovládání
- světle modrá sekce Vnímání
- zelená sekce Operátory
- tmavě oranžová sekce Proměnné
- červená sekce Moje bloky

Do projektu lze přidat další specifické sekce bloků. Tyto další sekce nejsou odlišené barevně, ale disponují piktogramem. Piktogram je uveden v přehledu sekcí, stejně jako na začátku každého jednotlivého bloku. Právě zmíněné je vidět na obrázku 3. Specifické sekce jsou určeny pro možnost kreslení, ale lze přidat i bloky, s nimiž je možné složit program například pro stavebnici LEGO Mindstorm nebo mikropočítač micro:bit.

Scéna má kartézské souřadnice. Osa x má hodnoty od -240 do 240 a osa y od -180 do 180. Těsně nad scénou je umístěno tlačítko zelené vlajky, které spouští daný projekt, a tlačítko červené silniční značky stop, jež projekt zastaví.

Každá postava má určené své vlastní jméno, polohu a úhel otočení na scéně, velikost, kostýmy, zvuky a scénáře. Jméno by měla mít každá postava unikátní. Poloha se měří na osách x a y. Kostým udává vzhled postavy. Postava může mít více kostýmů, které pak ve scénáři může přepínat. Kostým má vždy ukázaný střed, od kterého se udává poloha a kolem něhož se i postava otáčí. Velikost postavy na scéně je zadaná v procentech a odvozuje se od velikosti kostýmu.

4 Program kroužku

Kroužek programování je určen především pro žáky druhého stupně základní školy, kteří chtějí začít s programováním. Kroužek je členěn do jednotlivých tematických bloků, jež mohou být rozděleny do více hodin podle rychlosti a schopnosti zúčastněných žáků. Předpokládaná délka bloku je vždy uvedena a počítá se s tím, že jedna vyučovací hodina kroužku bude trvat zhruba devadesát minut.

Pro každý blok jsou stanoveny předpoklady pro zvládnutí tématu. Pokud žáci požadované předpoklady nezvládají, např. je ještě neprobírali ve škole, je potřeba jim daná témata jednoduše vysvětlit. Projekty, které se řeší v rámci jednotlivých bloků, obsahují vlastní příklady. Ty jsou navrženy tak, aby byly přizpůsobené jednotlivým tématům, úrovni znalostí a byly pro žáky atraktivní. V plánu bloku jsou odkazy na projekty ve webovém prostředí Scratche. Projekty jsou také k dispozici na přiloženém disku. Metodické poznámky pro lektora jsou psány kurzívou.

Pro vytváření plánu kroužku jsem využívala svoji zkušenost s koučováním kroužku Programování ve Scratchi pod neziskovou organizací Czechitas, která se specializuje na výuku v oblasti informačních technologií pro ženy a děti. Jako další významný zdroj jsem využívala online kurz programování [8].

4.1 Úvod do kroužku

V úvodním bloku budou žáci seznámeni s organizací kroužku a s jeho celkovými cíli. Budou také stručně obeznámeni s prostředím Scratche a jeho komunitou.

Doporučený rozsah:

1 kroužek (1 x 90 minut)

Předpoklady pro zvládnutí bloku kroužku:

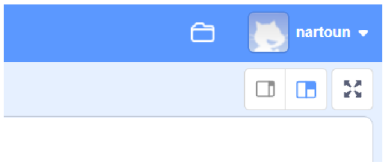
- základní práce s počítačem

Cíle bloku kroužku:

- Žák se zběžně seznámí s prostředím Scratche a jeho komunitou.
- Žák si samostatně prohlédne projekt ve Scratchi.

Domácí úkol z bloku kroužku:

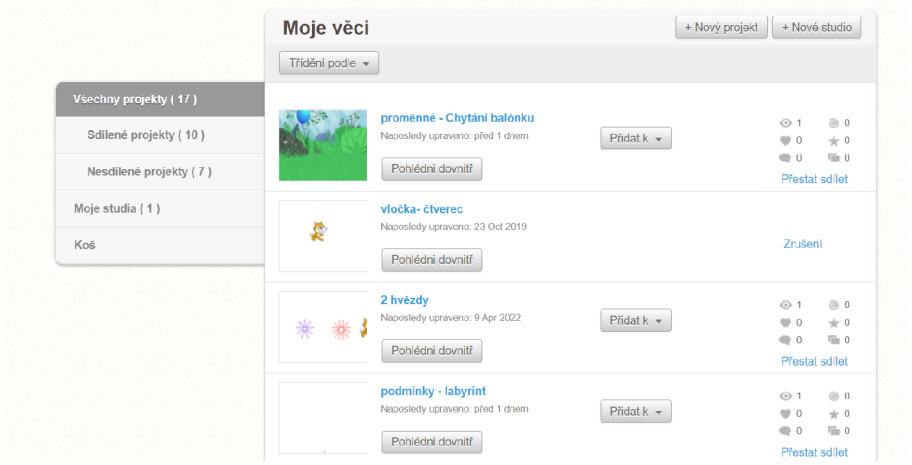
Každý žák si založí vlastní účet. Při založení je potřeba ověření, a to pomocí e-mailu rodiče. *Ideální by bylo připomenout i rodičům a upozornit na potřebu povolit žákovi sdílet jeho projekty.*



Obrázek 4: Umístění tlačítka pro přechod do složky projektů uživatele

Průběh bloku kroužku:

- Představení lektora a žáků pomocí některé seznamovací hry. *Každý řekne, proč se přihlásil; jestli má zkušenost s programováním; jaké zná programovací jazyky atd.*
- Stanovení si pravidel kroužku a jeho organizaci. *Kdy a kde bude kroužek probíhat, kde mohou žáci čekat, předání kontaktu na lektora, jednotlivé žáky a jejich rodiče. Pravidla: žádná otázka není hloupá, neskáčeme si navzájem do řeči apod.*
- Ukáže se stránka Scratche scratch.mit.edu s krátkým povídáním o něm. *Žáci mohou říct, jestli o Scratchi už dříve slyšeli a co o něm ví.*
- Stránka s programovacím oknem. Rozeberou se jednotlivé části: prostor pro kódovaní, scéna (výstupní okno), seznam postav a jejich hodnoty, nastavení pozadí, paleta bloků. Zatím se nebudou vysvětlovat kostýmy postav a zvuky, jak je vidět na obrázku 1.
- Vysvětlení přiřazení a důležitost názvu a automatického ukládání, případně ukládání ručního a možnost stáhnout si projekt do počítače. *Ideálně tento bod často opakovat a dbát na logické pojmenovávání každého projektu i postavy. Možnost stáhnout si projekt je užitečná především tehdy, pokud se žák nedokáže přihlásit na svůj účet nebo chce na projektu pracovat na desktopové verzi.*
- Každý uživatel potřebuje svůj vlastní účet pro ukládání a sdílení svých projektů.
- Domácí ukol: každý si založí svůj účet. Je nutná autorizace rodičů přes e-mail. *Možné je založení třídy ve Scratchi a účtů všech žáků učitelem. Momentálně jsou při této variantě omezené funkce třídy i jednotlivých účtů. Vývojový tým Scratche má v plánu pracovat na dalších funkcích, ale zatím není veřejně znám žádný termín.*
- Ukáže se profil uživatele a jeho složka projektů. *Složka projektů je místo, do něhož budou jednotlivé projekty uživatele uloženy. Obrázky číslo 4 a 5 ukazují tlačítko pro přesun do složky a její ukázku.*



Obrázek 5: Ukázka složky projektů s projekty uživatele

- Ukázka sdílených projektů ostatních uživatelů a komunity Scratche. Žáci si sami vyzkouší zahrát sdílené hry nebo se podívají na animaci. Je důležité, aby získali představu, co budou moci ve Scratchi vytvořit později samostatně. *V tuto chvíli je dobré zapnout počítač a sdílet podrobnosti k Wifi. Pokud by se počítače zapnuly dříve, studenti nebudou dávat pozor.*

- Upozorníme na možnost podívat se do kódu a vytvořit si vlastní kopii, kterou lze upravovat. Uživatelské rozhraní pro tuto funkcionalitu lze vidět na obrázku 6.

- Zmíníme možnost stáhnout si Scratch do počítače *Všichni žáci by měli na kroužku pracovat s webovou verzí Scratche, aby nevznikly odchylky a bylo možné sdílet projekty. Toto je vhodné, pokud chce někdo tvořit, aniž by použil internet. Ve stažené verzi Scratche se vyskytují odchylky od webové verze.*

4.2 Základní znalosti – příběh

Doporučený rozsah:

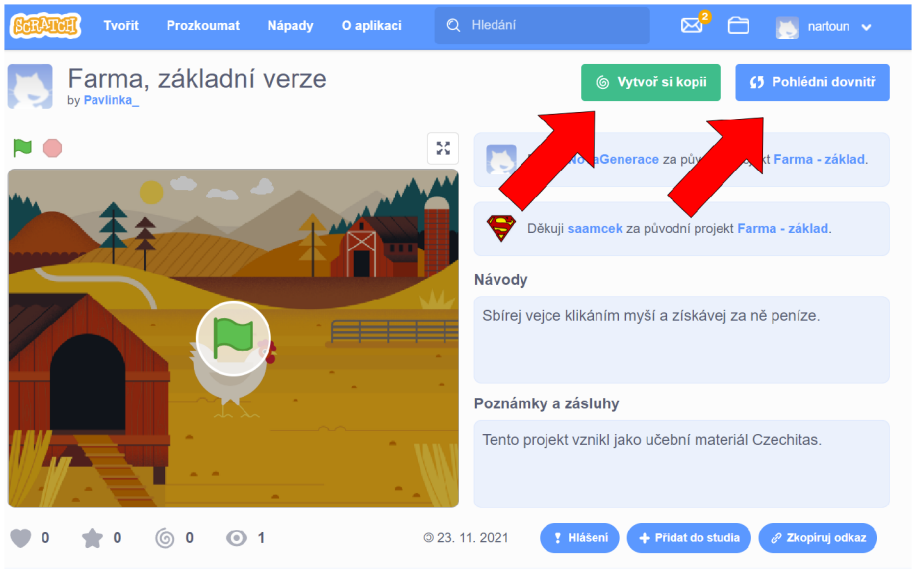
4 kroužky (4 x 90 minut)

Předpoklady pro zvládnutí bloku kroužku:

- základní práce s počítačem

Cíle bloku:

- Žák se seznámí se základními bloky Scratche.
- Žák vytvoří jednoduché scénáře postav.



Obrázek 6: Sdílený projekt. Šipky ukazují na tlačítko pro vytvoření kopie projektu a tlačítko pro ukázání postav projektu a jejich scénáře.

- Žák přidává a upravuje postavy a jejich kostýmy.

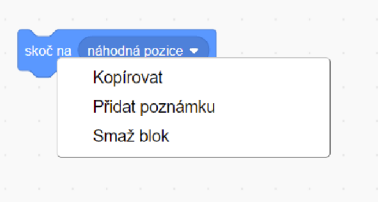
Průběh bloku:

- Ukázání některého příběhu jakožto motivace a vyzvání žáků k rozmyšlení si svého vlastního příběhu. *I když se bude příběh tvořit až na konci bloku, je dobré, aby měli žáci na vymýšlení dostatek času. Přiměřeně obtížný ukázkový příběh je lepší vybrat ze sdílených. Žáci tak neuvidí lektorem uměle vytvořený příběh.*

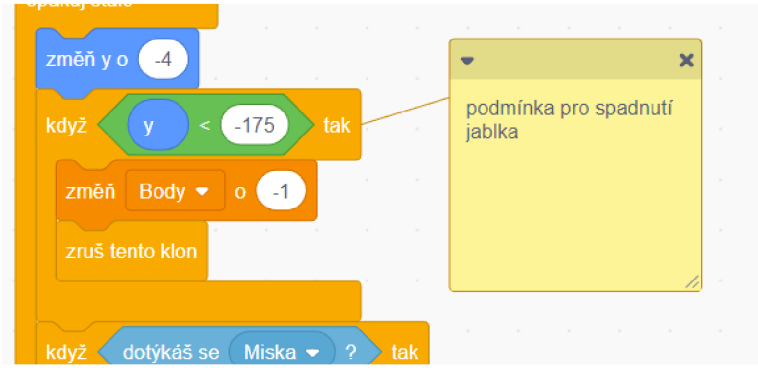
- Ukázání bloků pohybu z modré skupiny Pohyb. Předvedeme funkce bloků kliknutím na konkretní blok v panelu nebo poskládáním do scénáře pod sebe. Bloky do sebe tvarem zapadají jako puzzle. *Připomenutí fungování souřadnicového systému. Ukážeme minima a maxima pro osy, střed soustavy. Upozornit na rozdíl mezi bloky „nastav na …“ a „změnit o …“. Předvedení funkce těchto bloků i se zápornými hodnotami, na které zapomínají hlavně mladší žáci.*

- Sestavení scénáře a manipulování s ním. Ukážeme přeskládání bloků, mazání, duplikování a přidávání poznámek. Možnosti bloku jsou zobrazeny na obrázku 7. *S daným blokem se vždy drží i všechny bloky zapojené pod ním. Poznámky se vždy váží k jednomu bloku, jak je vidět na obrázku 8.*

- V diskuzi s žáky zjistíme, že se náš scénář nikdy sám nespustí. Předvedeme úvodní blok scénáře ze žluté skupiny událostí a správné zapínání a vypínání projektu přes zelenou a červenou vlajku, které jsou vidět na obrázku 9.



Obrázek 7: Možnosti bloku se objeví poté, co na něj klikneme pravým tlačítkem myši.



Obrázek 8: Komentář ve Scratchi, který se vždy váže na konkrétní blok scénáře.

Poukážeme na tvar úvodního bloku ukázaný na obrázku 10, který zajišťuje první pozici v každém scénáři.

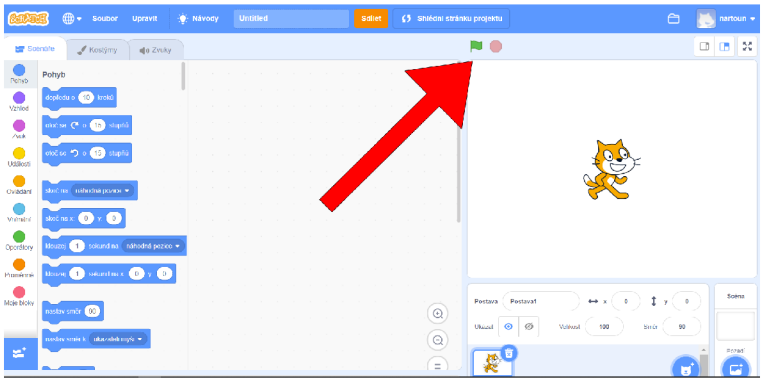
- Ukážeme, kde se přidává jiné pozadí scény, a upozorníme na pozadí se souřadnicemi pro lepší orientaci na ploše scény. *Pozadí si můžeme zvolit z nabídky ve Scratchi, nahrát vlastní obrázek, nebo jej nakreslit. Pozadí se souřadnicemi lze použít pro orientaci u každého projektu a nakonec je můžeme vymazat.*

- Ukážeme, kde se přidávají nové postavy a místo jejich pojmenování. Toto rozhraní je zobrazeno na obrázku 11. *Postavu si můžeme zvolit z nabídky ve Scratchi, nahrát vlastní obrázek, nebo ji nakreslit.*

- Předvedeme možnosti kostýmu postavy. Přidání nového kostýmu postavy, možnost jeho pojmenování a přejmenování. *Základní kočka má 2 kostýmy, některé postavy mají pouze jeden a jiné více.*

- Předvedeme funkci duplikace kostýmu a manipulaci s ním, a to vektorově. Ukážeme rozdílné možnosti manipulace obrázku převedeného do bitmapy. *Vektorově ukážeme např. změnu polohy končetin a výrazu obličeje. Při převedení obrázku do bitmapy můžeme docílit např. rozpůlení obrázku.*

- Samostatná práce: Namalovat nový kostým postavy. *Střed postavy, podle kterého se udávají její souřadnice na ploše, se zobrazuje jako šedý terč.*



Obrázek 9: Umístění vlajek na ploše. Zelená vlajka pouští projekt a červená projekt zastaví.

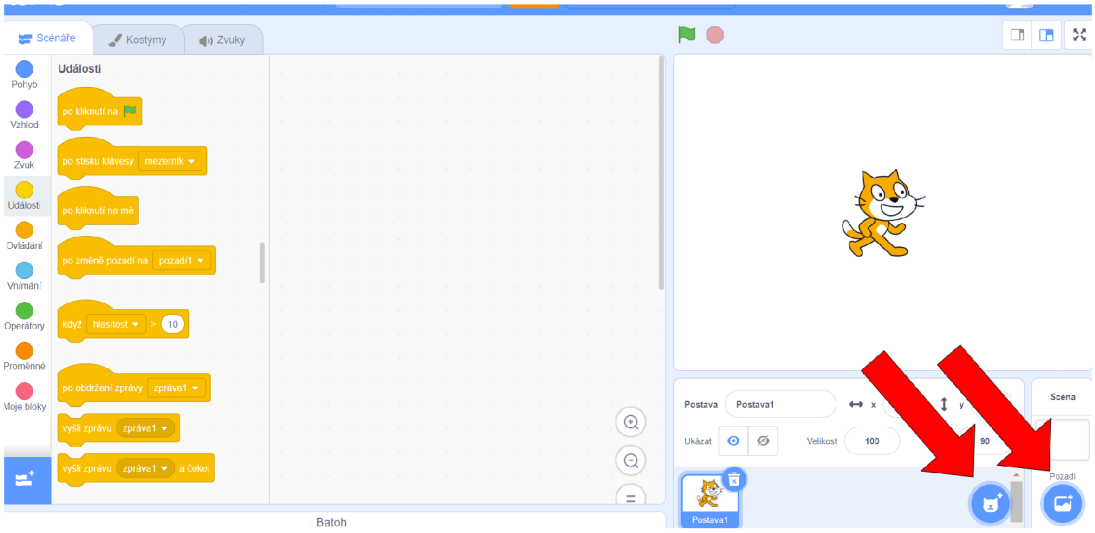
Obrázek 10: Tvar úvodního bloku zajišťuje první místo ve scénáři.

- Společně si vyzkoušíme bloky z fialové skupiny Vzhled. Bloky zkoušíme stejným způsobem jako bloky ze skupiny Pohyb. *Pokud nemají bubliny nastavený čas, zobrazují se napořád a jde je pouze přepsat jinou bublinou. Blok z této skupiny „jdi dopředu/dozadu o ...“ se vztahuje k vrstvě popředí, pozadí. Určuje, která postava překrývá kterou. Některé bloky mají seznam věcí, jež mohou měnit – např. blok „změň efekt barva o ...“ jde upravit na blok „změň efekt jas o ...“ Často se na tuto funkci zapomene, a tím se přichází o mnoho možných zajímavých kombinací.*

- Ukážeme blok „čekej ...“ z oranžové skupiny Ovládání. *Zvláště při sestavení animace je vhodně zvolené čekání důležité – např. jedna postava čeká na akci druhé.*

- Předvedeme růžové bloky ze skupiny Zvuků. *Při hodině je dobré zvuky nepoužívat ani ve sluchátkách, aby žáci věděli, co se kolem nich děje, a slyšeli lektora. Zvuky jsou velmi vítané u výsledného projektu, protože dokreslují atmosféru. Při vytváření projektu můžeme například nastavit hlasitost zvuku na nulu a později tento blok oddělat, nebo jen změnit jeho hodnotu. Upozorníme, že hlasitost má každá postava vlastní, takže je nutné všechny postavy ztlumit. Také upozorníme na rozdíl mezi „přehraj zvuk ... až do konce“ a „začni hrát zvuk ...“.*

- Ukázání vzorového projektu příběhu z komunity a následné společné probrání jeho kódu.



Obrázek 11: Místa pro přidání nové postavy a jiného pozadí

- Samostatná práce: vytvořit vlastní krátký příběh na dané téma. V příběhu by měla proběhnout výměna kostýmu a více než jedna pohybující se postava. Fantazii se meze nekladou. *Tématy mohou být např. pohádka*[1]*, co se dělo o prázdninách, povídání o své rodině. Podmínky úkolu by neměly být omezující, ale zároveň by měly zachovat určitý standard obtížnosti.*

4.3 Cykly – kreslení pravidelných n-úhelníků

V tomto bloku si představíme některé jednodušší druhy cyklů a budeme kreslit pravidelné mnohoúhelníky pomocí želví grafiky.

Doporučený rozsah:

3 kroužky (3 x 90 minut)

Předpoklady pro zvládnutí bloku:

- znalost úhlů
- zvládnutí předchozích bloků kroužku

Cíle bloku:

- Žák se seznámí s tzv. želví grafikou.
- Žák použije cyklus „opakuj ... krát“, což je ekvivalent cyklu „for“ v jiných programovacích jazycích.

[1]Příklad pohádky: https://scratch.mit.edu/projects/334676735

- Žák použije nekonečný cyklus „opakuj stále“. Tento cyklus je ekvivalent cyklu „while“ s tautologickou podmínkou v jiných programovacích jazycích

Želví grafika

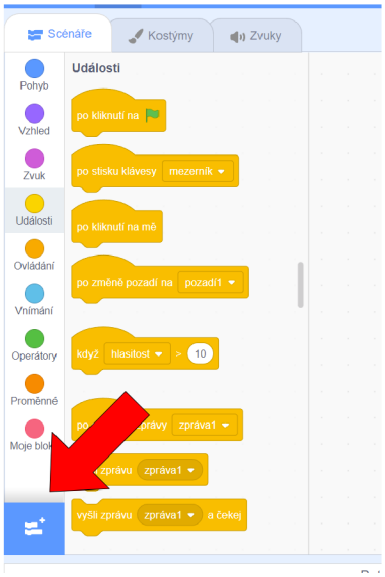
Želví grafika je metoda, která pomáhá vykreslovat různé obrázky pomocí několika základních příkazů („jdi dopředu“, „zatoč doprava“) a základních programátorských konstrukcí (cykly, proměnné, funkce). Začala se používat v 60. letech 20. století a je to osvědčený nástroj pro výuku programování a matematiky. Je velmi intuitivní a existuje řada prostředí, ve kterých je možné ji použít. Želví grafiku lze využít již při výuce dětí na prvním stupni základních škol, ale také se využívá pro procvičování netriviálních matematických a programátorských pojmů, jako jsou rekurze nebo goniometrické funkce.[9]

V želví grafice můžeme standardně použít příkazy :

- dopředu o x kroků *Dozadu se želva pohybuje se záporným x.*
- otočení doleva/doprava o daný úhel
- zvednutí a položení pera
- nastavení barvy pera
- nastavení tloušťky čáry
- vykreslení tečky
- uložení pozice želvy
- obnovení pozice želvy

Průběh bloku:

- Přidání skupiny Pero pro kreslení za použití tlačítek zobrazených na obrázku 12. *Tato skupina bloků se nezobrazuje při základním nastavení, ale jednoduše se přidá tlačítkem pod sekcemi bloků.*
- Vyzkoušíme funkčnost a možnosti bloků v sekci Pero. *Upozorníme, že postava kreslí vždy svým středem. Zopakujeme, kde má postava střed. Při přesunu postavy se mezi danými souřadnicemi nakreslí spojnice.*
- Vysvětlíme používání barvy, jasu, sytosti a průhlednosti a jejich číselné nastavení. *Ve Scratchi všechny nabývají hodnoty od 0 po 100.*
- Samostatná práce: nakreslíme čtverec pomocí sekvence příkazů.
- Zahájíme diskuzi, jestli by to šlo udělat lépe, a co je při tomto způsobu práce častou chybou. *Žáci mnohdy na řešení přijdou sami. Ukážeme možnost chyby při přepisu scénáře na větší čtverec – např. jednou se změní délka, zamění se blok apod.*



Obrázek 12: Místo pro přidání skupiny bloků Pero

- Představíme blok „opakuj ... krát“ a pomocí něj nakreslíme čtverec. *Takto nakreslený čtverec si uložíme, protože ho budeme dále potřebovat.*

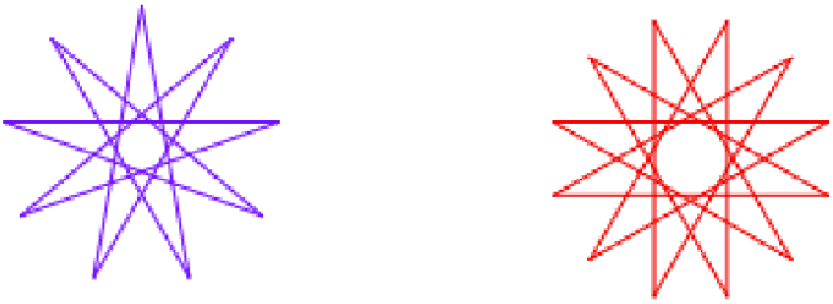
- Samostatná práce: nakreslit pravidelný šestiúhelník, přičemž každá strana bude mít jinou barvu. *Je vítané, aby žáci zkusili na správný úhel, tj.* 60^o*, přijít sami a vyzkoušeli, jaké barevné přechody získají v závislosti na parametru změny barvy pera. Rychlejší studenti mohou zkusit i jiné pravidelné n-úhelníky.*

- Několik studentů předvede svoji práci [2] a komentář – např. s čím měli problém, jaký barevný přechod se jim nejvíce líbil atd. *K předvedení se může použít sdílení projektu a jeho ukázání přes projektor. Každý, kdo je s prací hotov, může sdílet projekt a zařadit jej do společného studia lektora pro tento úkol. Žáci by měli být schopni přečíst kód jako věty. Toto bude důležité především u složitějších kódů, a je tedy třeba učit žáky této dovednosti už od začátku.*

- Samostatná práce: nakreslit vedle sebe dvě různé hvězdy různých barev tak, aby nebyly spojené. *Žáci by měli přijít sami na to, že potřebují mezi hvězdami zvednout pero. Hvězdy by měly mít přijatelnou velikost. Není špatně, pokud se hvězdy částečně překrývají, ale musí být celé vidět. Je potřeba nezapomenout ukázat možnosti různých hvězd pro inspiraci, jako jsou na obrázku číslo 13.*

- Zkontrolujeme řešení[3].

[2] odkaz na hotový projekt: https://scratch.mit.edu/projects/665137758
[3] odkaz na hotový projekt https://scratch.mit.edu/projects/673890108



Obrázek 13: Ukázka hvězd nakreslených ve Scratchi.

- Vrátíme se do projektu se čtvercem a zkusíme pomocí něj společně nakreslit hvězdu. Definujeme pojem vnořený cyklus a použijeme ho. *Žáci tím uvidí možnost mít cyklus v cyklu. V daném případě je mimořádně důležité, aby všichni žáci chápali, který blok se kdy provádí a proč. Scratch totiž nemá debugger a vždy se zvýrazňuje celý scénář, který se provádí, nikoliv však jednotlivý blok. Pro názornost je vhodné mít každý čtverec hvězdy jiné barvy.*

- Samostatná práce: nakreslit sněhovou vločku pomocí vnořeného cyklu. *Základem by neměl být čtverec, ale pokud má někdo problém s pochopením cyklů, je lepší čtverec ponechat. Vločka může být nakonec v odstínech modré, ale na začátku by pro názornost měl mít každý základní obrazec jinou barvu.*

- Několik žáků předvede své vločky[4] - ostatní pak hádají, jaký je výchozí obrazec.

Pedagogické výhody želví grafiky

Želví grafiku lze v programování zařadit v různých fázích zdatnosti studenta. Na uvedeném příkladu jsem jejím prostřednictvím ukázala jedny ze základních principů programování, ale lze na ní demonstrovat i tak složité programátorské téma, jako je rekurze. Studenti na různých stupních vzdělávání si s její pomocí mohou přiblížit v geometrii např. úhly, Pythagorovu větu i fraktály.

Jelikož je výsledkem vždy nějaký obrázek, je její použití pro studenty atraktivní. Vykreslený obrázek je také pro většinu lidí uspokojivější než správně vyřešený problém s písemným výstupem. Nesprávné řešení často vidí na první pohled i samotný student – např. obrazec se nespojí nebo je nějak posunut, obrazec je na jiném místě, než je očekáván apod.

Různé zajímavé příklady pro procvičení želví grafiky můžeme najít v knize Želví grafika Exkurze do programování, geometrie a umění od Radka Pelánka.

[4] odkaz na příklad hotového projektu https://scratch.mit.edu/projects/673888122

4.4 Podmínky

V tomto bloku se naprogramuje první hra a zavedou se podmínky.

Doporučený rozsah:

3 kroužky (3 x 90 minut)

Předpoklady pro zvládnutí bloku:

- zvládnutí předchozích bloků kroužku

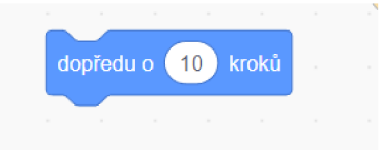
Cíle bloku:

- Žák se seznámí s podmínkami ve Scratchi.
- Žák se seznámí s výrazy, které mohou být uvedeny v podmínkách.
- Žák použije další úvodní bloky scénářů.

Průběh bloku:

- Předvedeme hru labyrint, kterou budeme následně programovat. *Je důležité hotové projekty dopředu nesdílet, protože by je žáci našli a pouze okopírovali. Je vhodné si hru i přejmenovat, aby nešla najít podle jména. Ve hře se objevují postava hledače, postava cíle a postava labyrintu. Labyrint je složen z čar jedné barvy, rozprostírajících se přes celou hrací plochu. Cíl bude čekat na jedné straně plochy a hledač se bude nacházet na straně druhé. Úkolem hráče je dostat hledače k cíli, přičemž nemůže překročit čáry labyrintu.*
- Některý z žáků vysvětlí fungování hry slovy. *Ostatním to pomůže si uvědomit, co má dělat která postava a posloupnost bloků ve scénářích*[5].
- Žáci si nahrají postavu labyrintu[6]. *Soubor s postavou se rozešle dopředu e-mailem, nebo si ho na hodině každý stáhne z flash disku, příp. může mít kroužek pro tyto případy sdílenou složku – např. na Google disk.*
- Postavu labyrintu si po pojmenování zvětšíme na celou výstupní obrazovku, aby byla ještě celá vidět, a vycentrujeme. *Pokud v kostýmu dostatečně oddálíme postavu, je vidět obdélník velikosti scény. V sekci kostýmu upozorníme, že se labyrint skládá jen z čar a mezi nimi žádná barva není. Ukazuje se jako kostkované pozadí.*
- Přidáme a zmenšíme postavu, která bude procházet labyrintem (hledač), a postavu sloužící jako cíl. *Postavy by se měly pohodlně vejít do uličky labyrintu. Každý si může zvolit vlastního hledače a cíl, ale někteří žáci často*

[5] odkaz na hotovou hru https://scratch.mit.edu/projects/712473154
[6] odkaz na postavu labyrintu https://inventwithscratch.com/mazegame/



Obrázek 14: Vnitřní blok scénáře

stráví dlouhou dobu vybíráním nejlepších postav. Pokud se někdo takový vyskytne, je lepší postavy určit.

- Vybereme a přidáme pozadí, které pojmenujeme Výhra. Na pozadí napíšeme nápis na téma „Vyhrál jsi!“ Toto pozadí se ukáže po úspěšném projití labyrintu. *Původní pozadí můžeme přejmenovat, ale není to nutné. Přidání postav i pozadí je dobré zadat jako samostatnou práci. Pokud by se na projektoru předvádělo to stejné, žáci by často jen bezmyšlenkovitě kopírovali, co vidí.*

- Začneme programování s postavou cíle, který je jednodušší. Diskuzí s žáky zjistíme, jak přesně by měl cíl fungovat. *Po dotknutí hledače se změní základní pozadí na pozadí Výhra a skryje se.*

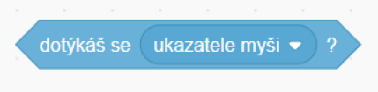
- Upozorníme, že zatím neznáme žádný blok, v němž bychom pokládali otázku. Ukážeme blok „když ... tak“ z oranžové skupiny Ovládání. *Poukážeme na podobnost s blokem cyklu. Do obou bloků je možné vnořit bloky jiné.*

- Rozebereme, jaké otázky můžeme pokládat pomocí podmínky. Zavedeme pojem výraz – otázka s uzavřenou odpovědí. *Mají odpověď ano, ne.*

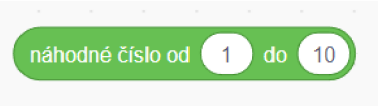
- V diskuzi vymyslíme příklady výrazu nejprve ze života a z matematiky, poté popřemýšlíme, které by se nám hodily ve Scratchi. *Matematické výrazy např. $<$, $>$ zařadíme, protože se v programování vyskytují velmi často a je dobré na ně upozornit dopředu.*

- Necháme žáky vyhledat příklady bloků výrazů. *Matematické a logické výrazy jsou v zelené skupině Operátory. Ostatní výrazy se nacházejí v modré skupině Vnímání.*

- Upozorníme na rozdílný tvar bloku výrazu (protáhlý osmiúhelník). Místo na výraz v podmínce je stejného tvaru. Vzhled bloků je zobrazen na obrázcích 10,14,15,16. *Když se doplňovalo číslo nebo text, byl tvar zaoblený. Žáci by si měli této skutečnosti všimnout sami. Pokud spustíme blok výrazu, zobrazí se nápis true nebo false – i při nastavení češtiny jsou v angličtině. Tyto bloky také nejde vkládat do scénáře samostatně, ale jako výplň jiných bloků. Protože jsou bloky tvarově odlišené, je to pro děti většinou intuitivní.*



Obrázek 15: Blok výrazu – protáhlý osmiúhelník. Nelze použít samostatně.



Obrázek 16: Zaoblený hodnotový blok. Nelze použít samostatně.

- V následné diskuzi přijde na řadu scénář pro postavu cíle. *Co by postava měla dělat, jsme si již řekli. Obtížné bude pro žáky si uvědomit, že podmínku musí dát dovnitř cyklu „Opakuj pořád“. Jinak bychom se na splnění podmínky zeptali jen jednou a scénář by skončil.*

- Samostatná práce: přidáme bloky, které nám hru vždy po spuštění uvedou do základního nastavení. *Postava cíle se ukáže, pozadí se nastaví na základní.*

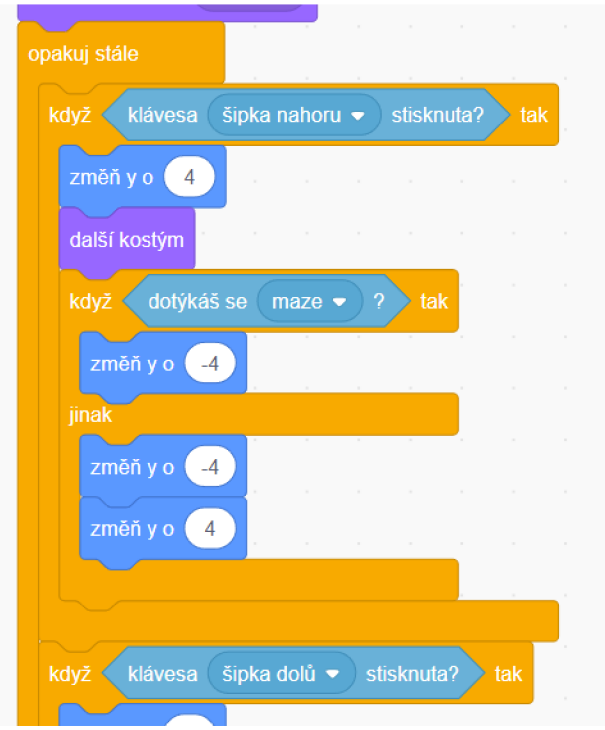
- Některý z žáků ukáže a popíše své řešení.

- Začneme řešit postavu hledače, u které je v této hře většina kódu. Společně s žáky rozhodneme, co by měl hledač dělat. Poukážeme na to, že postava by se měla pohnout pouze po stisku klávesy. Vyzveme žáky, aby takový blok našli. *Mohou najít dva vyhovující bloky: výraz „klávesa … stisknuta“ a úvodní blok „Po stisku klávesy …“ Kód je samozřejmě možné napsat i pomocí podmínky s výrazem v nekonečném cyklu, ale v tomto výukovém bloku se zaměříme na druhou možnost.*

- Vysvětlíme různé úvodní bloky a jejich možnosti použití. *Každý žák může vymyslet příklad k jednomu úvodnímu bloku. Vysílání zpráv zatím řešit nebudeme.*

- Samostatná práce: pohyb postavy hledače po zmáčknutí šipek. *Žáci musí mít prostor programovat sami a případné chyby najít a předělat. Velmi často se zapomíná, že v případě směrů dolů a doleva se musejí zadávat záporná čísla. Musíme zařídit, aby postava hledače neprocházela zdmi labyrintu. Vždy po posunu určitým směrem se zeptáme, jestli se postava dotýká labyrintu, a případně ji posuneme zpět. Posuny následují tak rychle po sobě, že je člověk ani nezaregistruje.*

- Ukážeme, jak by mělo vypadat řešení. *Na obrázku 17 je vidět scénář pro pohyb nahoru. Pohyb do ostatních směrů se naprogramuje obdobně. Pokud budou mít žáci dotazy ohledně řešení, bylo by užitečné, aby nejprve zkusili odpovědět ostatní.*



Obrázek 17: Scénář pohybu nahoru pro postavu Hledače. Pohyb do ostatních směrů se naprogramuje obdobně.

- Společně si připomeneme, co se stane při výhře. *Přepne se pozadí a schovají se postavy.*
- Samostatná práce: vytvoření scénáře pro konec hry
- Společně scénáře sestavíme. *Každá postava se po změně pozadí schová. Pro tuto možnost je ve Scratchi úvodní blok „po změně pozadí na …“*
- Žáci si hru zahrají a případně vylepší podle své fantazie. *Žáci budou mít hru uloženou na svém účtu a mohou si ji dovytvořit nebo zahrát i doma.*
- Hru je možné vylepšit přidáním dalších úrovní. Jednoduše lze přidat úroveň jako další kostým labyrintu, pokud bude mít začátek a konec na stejném místě. *Je dobré žákům možnost vylepšení připomenout a uvést příklady. Je vhodné vyčlenit v kroužku čas na ukázky vylepšení a vlastní projekty žáků. Mezi vylepšení se počítá i propracovanější grafika hry.*

4.5 Proměnné

Doporučený rozsah:

2 kroužky (2 x 90 minut)

Předpoklady pro zvládnutí bloku:

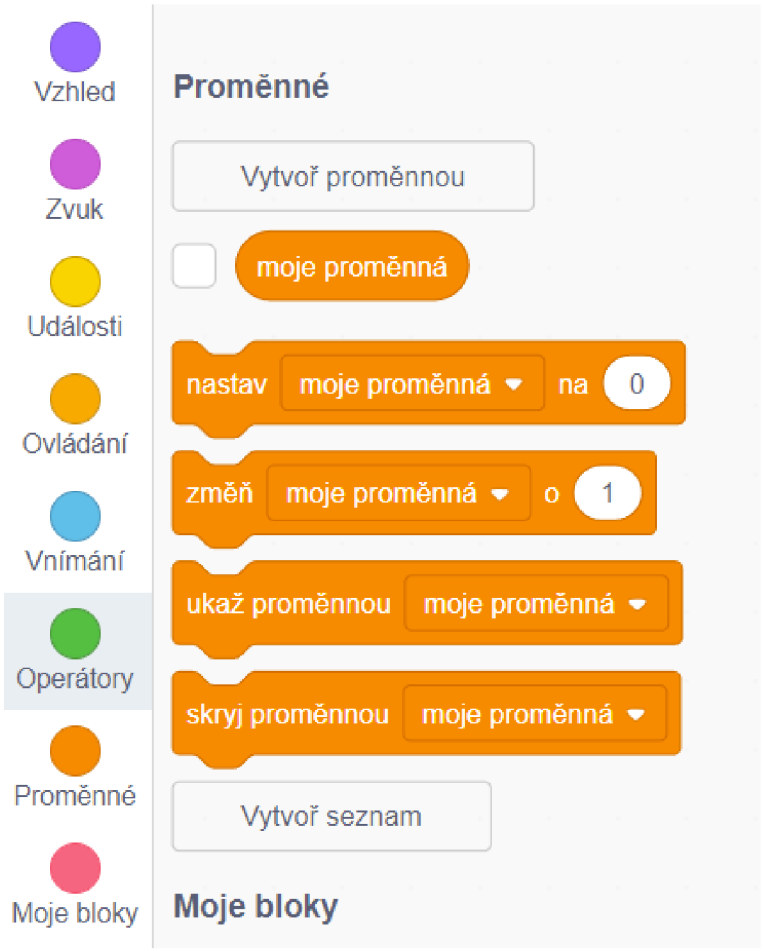
- zvládnutí předchozích bloků kroužku

Cíle bloku:

- Žák bude používat proměnné číselné i textové.
- Žák si procvičí výrazy.
- Žák použije blok „čekej, dokud nenastane ...“
- Žák bude schopen využívat údaje zadané uživatelem projektu.

Průběh bloku:

- Představíme si hru, kterou budeme v tomto bloku vytvářet. Jde o jednoduché chytání balónku a počítání skóre. *Tato hra je velmi jednoduchá na hraní i naprogramování, ale ukážeme si zde fungování a důležitost proměnných. Hra je postřehová – každý participant má za úkol klikat myší na náhodně se objevující se balónek.*
- odkaz na hru
- S žáky verbálně popíšeme, co se přesně ve hře děje. *Balónek se náhodně objeví na obrazovce. Poté, co na něj klikneme, přesune se na jiné – náhodné – místo, a přičte se jeden bod. Hra začíná s nula body.*
- Začneme vysvětlovat, jak fungují proměnné. Bloky proměnných jsou zobrazeny na obrázku 18. *Můžeme je přirovnat k pojmenované krabičce, do které lze ukládat různé informace. Ve Scratchi není proměnná vázaná na typ, takže do stejné proměnné ukládáme čísla i text. U této hry textovou proměnnou potřebovat nebudeme.*
- Necháme žáky vyzkoušet a popsat bloky z oranžové skupiny Proměnné. *Pokud budeme chtít proměnnou někde použít, stačí jen dát zaoblený blok s názvem příslušné proměnné do místa v bloku, kam jsme dosud psali čísla ručně. Tato místa mají opět stejný tvar, takže se jedná o velmi intuitivní postup. Zaškrtávací políčko u tohoto bloku značí, zda je proměnná vidět na scéně. Zaobleným blokům budeme říkat hodnotové, protože vždy představují hodnotu a nedají se použít samostatně ve scénáři. Ve Scratchi je vždy vytvořená jedna proměnná, kterou můžeme přejmenovat. Další proměnné si musíme vytvořit sami.*
- Ukážeme, že stejný tvar mají i přednastavené proměnné – např. „směr“, „číslo kostýmu“ atd. Většina těchto proměnných je specifická pro konkrétní postavu (lokální proměnné), ale nikoliv všechny. Například „x myši“ mají všechny postavy stejné – tyto proměnné nazveme globální. *Již předepsané*



Obrázek 18: Bloky sekce Proměnné

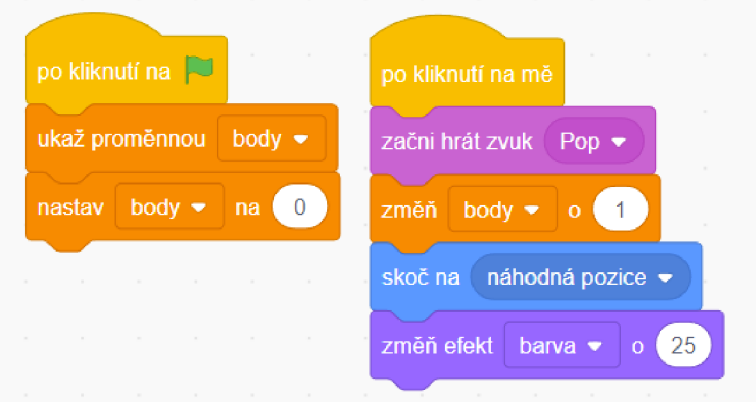
proměnné se často používají, a je tedy dobré je znát. U nově vytvořené proměnné zvolíme, jestli je určena pouze pro jednu, nebo pro všechny postavy. Proměnná pouze pro jednu postavu se používá především u klonování, které vysvětlíme později.

- Samostatná práce: každý žák najde alespoň dvě předdefinované proměnné lokální a dvě globální. *Musíme si uvědomit, že hlasitost je proměnná lokální, a tudíž každá postava může vydávat zvuky různě hlasitě.*

- Vyhodnotíme správnosti odpovědí. Na některých si ukážeme jejich správnost pomocí dvou postav. *Mezi ukázky zařadíme hlasitost, protože ta se žákům často plete.*

- Přidáme si do hry postavu balónku. Založíme si novou globální proměnnou – body.

- Samostatná práce: napsat scénář pro celou hru. *Hra je velmi jednoduchá a žáci by ji měli zvládnout. Pokud bude někdo rychlejší, vylepší hru o zvuky*



Obrázek 19: Řešení hry na chytání balónku.

prasknutí balónku a proměnu jeho barvy po kliknutí. Žáci by měli přijít vyzkoušením hry na fakt, že si po opětovném spuštění proměnná zachovává svoji hodnotu. Je třeba ji na začátku scénáře nastavit na nulu.

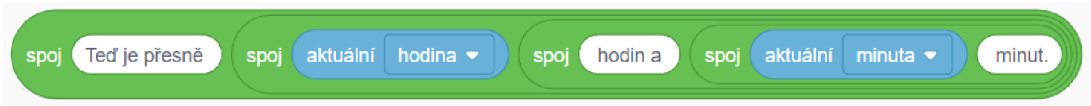
- Zkontrolujeme samostatnou práci[7]. Některý žák předvede správné řešení a okomentuje ho. Řešení úlohy je vidět na obrázku 19. *Žáci se musí v předvádění střídat. Když se objeví nějaké otázky, pokusí se je napřed zodpovědět ostatní a lektor je případně doplní.*

- V našem následujícím projektu si ukážeme textové proměnné a interakci s uživatelem. Předvedeme si projekt[8], kde budeme zadávat jméno a příjmení. Poté postava pozdraví uživatele jménem, příjmením a přezdívkou používanou ve Scratchi.

- Představíme blok „otázka ...“ a k němu patřící hodnotový blok „odpověď“. *Do bloku „otázka ...“ se doplní text, který se ukáže v bublině u dané postavy. Odpověď se zapisuje do pole, které se zobrazí v dolní části scény. Odpověď se potvrdí klávesou Enter nebo ikonou fajfky v pravé části pole. Je důležité zdůraznit, že po každé otázce se hodnota odpovědi změní. Pokud vyžadujeme více odpovědí, musíme danou odpověď pokaždé uložit na jiné místo - tj. potřebujeme proměnnou pro každou odpověď.*

- Společně naprogramujeme projekt. Ukážeme žákům blok „spoj x y“, který propojuje dva texty dohromady. Blok je užitečný, jestliže chceme mít text a hodnotu proměnné v jedné bublině. *Jelikož tímto blokem spojíme dohromady jen dva texty, je potřeba je propojovat, jak je vidět v projektu. Scratch nám tvarem bloků přímo udává, kam má přijít výraz a kam hodnota, proto s tímto aspektem nebývá problém. Žáci mívají potíže s možností zanoření jednoho hodnotového bloku do druhého, což je vidět na obrázku 20. Všechny*

[7] Celý projekt je dostupný na https://scratch.mit.edu/projects/676614146
[8] odkaz na projekt https://scratch.mit.edu/projects/670460870



Obrázek 20: Příklad zanořený bloků „spoj x y“

proměnné i hodnotu odpovědi necháme viditelné na scéně, aby bylo názornější přepisování hodnot.

- Jelikož bude uživatel zadávat své jméno a příjmení, tak na konci scénáře tyto proměnné přepíšeme. *U tohoto bodu je vhodné upozornit na zveřejňování citlivých údajů na internetu. Žáci se o bezpečném chování na internetu dozvídají ve škole, ale opakované upozornění si lépe zapamatují.*
- Projekt si společně vyzkoušíme a navrhneme vylepšení – např. přidání mezery mezi jméno a příjmení, vypsání aktuálního data, dotaz na další údaje atd.
- Samostatná práce: každý žák si do projektu přidá alespoň jedno vylepšení z diskutovaných. *V hotovém projektu jsou tato vylepšení již zahrnuta.*

4.6 Vlastní projekt

V tomto bloku bude lektor působit pouze jako poradce, protože každý žák si bude tvořit projekt podle vlastního návrhu.

Doporučený rozsah:

3 kroužky (3 x 90 minut)

Předpoklady pro zvládnutí bloku:

- dopředu vymyšlený projekt každého žáka
- zvládnutí předchozích bloků kroužku

Cíle bloku:

- Žák si procvičí doposud probraná témata.
- Žák si vyzkouší samostatně navrhnout celý projekt.
- Žák si procvičí logické uvažování.

Průběh bloku:

- Připomeneme žákům, co vše už umí. *Použijeme formu soutěže, přičemž se bude odpovídat na uzavřené otázky. Vhodné je využít některou aplikaci na kvízy, například Kahoot!*
- Samostatná práce: vytvořit si vlastní projekt. *Musíme ověřit, že každý pracuje na projektu přiměřené obtížnosti a rovněž zvládnutelném v časovém intervalu 2 kroužků. Pokud si některý žák nevymyslel svůj projekt, může vylepšovat projekty předchozí*[9]*. V tomto případě je dobré, aby si žák vymyslel vícero vylepšení k jednomu projektu. Žákům pomáháme s problémy a konzultujeme obsah projektu, ale snažíme se do tvorby příliš nezasahovat. Žáci mohou na projektech pracovat i doma.*
- Každý žák předvede svůj projekt, okomentuje ho a odpoví na otázky ostatních. *Na tento bod si necháme dostatek času, ideálně celý jeden kroužek, aby každý žák odprezentoval svůj projekt podle svého. Je vhodné, aby si žáci projekty navzájem vyzkoušeli.*
- Na závěr se uskuteční hlasování o nejzajímavější nápad, nejhezčí zpracování a nejzábavnější hru. *Hlasování by mělo být anonymní. Každý napíše na papír ke každé z jednotlivých kategorii dva různé projekty.*

4.7 Klonování

V tomto bloku ukážeme funkci klonování postav, která se ve Scratchi používá velmi často. Klonování nám do projektů přinese možnost mít na scéně více postav se stejným kódem, který napíšeme pouze jednou. Pomocí této techniky se snižuje redundance kódu.

Doporučený rozsah:

4 kroužky (4 x 90 minut)

Předpoklady pro zvládnutí bloku:

- zvládnutí předchozích bloků kroužku

Cíle bloku:

- Žák se naučí používat klonování postav.
- Žák si procvičí předchozí znalosti.

[9] Vylepšený projekt Labyrintu: https://scratch.mit.edu/projects/713431733

Obrázek 21: Umístění Batohu na ploše.

Průběh bloku:

- Představíme hru[10] Chytání a společně si ji popíšeme. *Ve hře chytá krab kapky vody a vyhýbá se bleskům. Úkolem je nasbírat co nejvíce bodů. Krab ve spodní části se pohybuje šipkami doleva a doprava. Shora padají kapky vody. Pokud se kapka dotkne kraba, zmizí a přičte se pět bodů. Jestliže kapku vody nechytneme, odečítá se jeden bod. Jednou za delší dobu padá kapka křišťálové vody za více bodů. Pokud se krabem dotkneme blesku, odečte se nám život. Začíná se s nula body a třemi životy.*

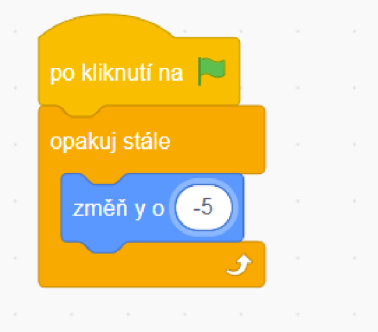
- Samostatná práce: přidat postavy kapky, kraba, blesku a změnit pozadí. Postavě kapka přidat nový kostým kapky křišťálové vody a poskládat scénáře pro postavu kraba. Potřeba bude také založit proměnné body a životy. *Další kostým kapky se lehce dotvoří jako obarvená kopie toho stávajícího. Zkontrolujeme, jestli všichni přebarvují kostým ve vektorovém módu. Obrázek v bitmapě by bylo možné přebarvit také, ale vektorový mód se pro tyto úpravy hodí lépe. Pohyb kraba je stejný jako u postavy hledače v projektu Labyrint.*

- Ukážeme správné řešení a velkou pomoc funkce Batohu. Její umístění je ukázáno na obrázku 21. *Do Batohu si můžeme uložit scénáře či jejich části, ale také celé postavy s jejími scénáři a kostýmy, takže si je můžeme přenášet z jednoho projektu do druhého. Po kliknutí na ikonu Batohu se otevře okno s momentálně uloženými kódy a postavami. Pokud si žáci napsali scénáře znovu, alespoň si je procvičili.*

- Sestavíme scénář kapky. Protože je scénář složitější, budeme ho doplňovat po funkčních celcích. V diskuzi si určíme, které funkce postavy budeme programovat. *Funkce jsou padání kapky, zmizení kapky po dopadu, zachycení kapky krabem, padání více kapek zároveň. Nejprve si naprogramujeme hru, v níž bude padat vždy jen jedna kapka. Poté si přidáme klonovaní.*

- Samostatná práce: sestavení scénáře pro funkci padání kapky vody. *Prozatím chceme jen scénář, kde bude kapka po zmáčknutí zelené vlajky v nekonečném cyklu snižovat svoji hodnotu souřadnice y. Ve Scratchi nikdy celá*

[10] odkaz na hotovou hru: https://scratch.mit.edu/projects/714459017



Obrázek 22: Scénář pro funkcionalitu padání kapky.

postava neodejde ze scény. Pokud postavu kapky ručně posuneme na scéně výše, znovu se začne posouvat dolů.

- Ukážeme správné řešení. Někdo z žáků je slovně okomentuje. *Parametr, o který budeme snižovat souřadnici y, udává, jak rychle bude kapka padat. Čím je jeho absolutní hodnota větší, tím rychleji bude padat. Pro pohodlné chycení kapky je vhodná hodnota parametru mínus pět. Výsledný scénář pro tuto funkcionalitu by měl vypadat obdobně jako na obrázku 22.*

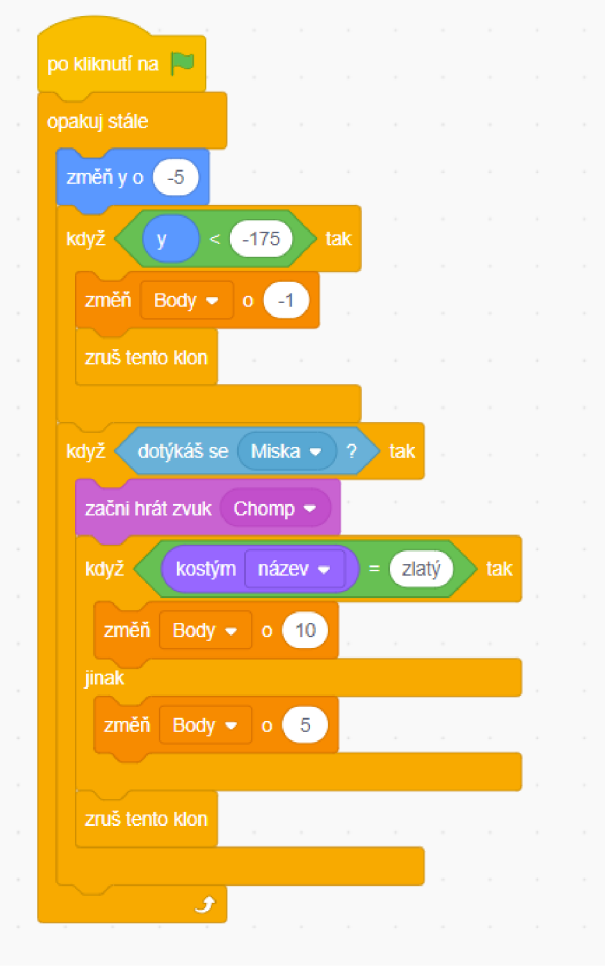
- Samostatná práce: přidat chování kapky při dopadu. *Ve scénáři se budeme ptát, jestli je souřadnice y menší, nebo rovna než spodní okraj scény. Pokud ano, přesune se postava na náhodné místo na horním okraji scény, a odečte se jeden bod. Žáci často zapomínají vnořit podmínku do cyklu – pak jim scénář nefunguje, protože se na otázku zeptají jen jednou.*

- Předvedeme možné řešení.

- Samostatná práce: naprogramování chování kapky při jejím chycení. *Do cyklu vnoříme podmínku s dotazem, jestli se dotýká postavy kraba. Pokud je podmínka splněna, zeptáme se na kostým a podle něj přičteme body. Žáci často zapomínají na různé bodové ohodnocení dvou typů kapky. Výsledný scénář je zobrazen na obrázku 23.*

- Někdo z žáků předvede a okomentuje své řešení.

- V tomto bodě máme fungující hru, v níž padá vždy jen jedna kapka. Zeptáme se žáků, jakým způsobem by přidali další padající kapky. Společně přijdeme na nápad, že postavu i s kódem několikrát duplikujeme. Vysvětlíme nebezpečí tohoto postupu. *Pokud jsme v původním scénáři udělali chybu nebo jej potřebujeme předělat, musíme přetvořit scénáře u všech kopií. Tento postup je časově náročný a velmi náchylný k dalším chybám. Postup názorně předvedeme a poukážeme na všechny jeho úskalí při více kopiích kódu.*



Obrázek 23: Scénář pro chycení kapky vody krabem.

- Ukážeme bloky pro klonování z oranžové skupiny Ovládání. Předvedeme je na jednoduchém příkladu postavy, která se naklonuje a posune – toto udělá opakovaně. *Klon se vždy vytvoří na místě původní postavy, takže pokud postavu neposuneme, klony budou poskládané přesně na sobě, a žákům se bude zdát, že neexistují. Scénář za úvodním blokem „když startuje můj klon“ se spustí každému konkrétnímu klonu jednou. Klon reaguje i na všechny scénáře postavy, které se spustí po jeho vytvoření – např. pokud úvodní blok je „po stisku klávesy …“*

- S žáky si ujasníme, jaké funkce by měla mít původní postava a jaké klony. *Postava by měla pouze vytvářet klony. Klon by měl plnit všechny funkce kapky ve hře. Oproti původní verzi s jednou kapkou by se klon po dotyku kraba nebo země měl zrušit.*

- Společně představíme scénáře s využitím klonování. *Nastavení proměnných necháme u původní postavy. Klonování musíme vložit do cyklu, protože jinak by padala jen jedna kapka. Musíme zajistit, aby kapky nepadaly ne-*

ustále. Přidáme blok „čekej … sekund“. Pro začátek můžeme zadat pevnou hodnotu, ale později přidáme – pro větší zajímavost hry – náhodné číslo. Podmínky se pouze přesunou a místo změny souřadnic postavy zrušíme klon.

- S žáky v diskuzi zjišťujeme, jaké funkce kapky nám chybí. *Kapka zatím nemění kostým, původní postava se zobrazuje v horní části scény.*

- Samostatná práce: zajistit, aby každá desátá kapka byla vylepšená. *Jak je běžné u programování, jde funkce začlenit různými způsoby. Funkce není složitá, ale žáci podobný úkol ještě neřešili, a budou potřebovat více času, možná i nápovědu od lektora. Je dobré žáky nechat chvíli přemýšlet samostatně nebo ve skupinkách.*

- Předvedeme možné řešení. Pokud bude mít některý žák jiné řešení, necháme ho způsob odprezentovat a okomentovat. *Jiné řešení lze provést například pomocí proměnné.*

- V této fázi stačí už jen zajistit, aby byly vidět jen klony, nikoliv původní postava. *Hned na začátku scénáře postavy tuto původní skryjeme. Klon se ukáže po svém vytvoření.*

- Hotový scénář pro původní postavu kapky může mít podobu jako na obrázku 24, scénář pro klony kapky by poté vypadal jako na obrázku 25.

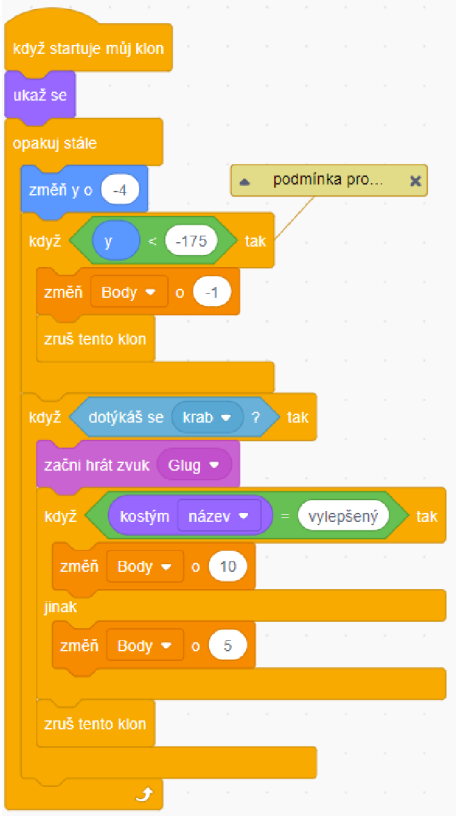
- Samostatná práce: naprogramovat postavu blesku. *Postava má velmi podobné scénáře jako postava kapky. Upravíme četnost padání – blesk bude padat méně často. Po spadnutí nebudeme ubírat body a při dotyku kraba se bude ubírat život. Postava nebude měnit kostým. Jestliže životy klesnou na nulu, hra se zastaví. Žáci se mohou společně radit a pomáhat si.*

- Ukážeme možnost správného řešení a probereme případné problémy, které žáci měli. *Pokud bude mít některý žák zájem, může řešení ukázat a okomentovat. Probereme i jiná řešení žáků, pokud se vyskytnou.*

- S žáky vymyslíme vylepšení hry a některé implementujeme. *Vhodné je např. přidat zvuky a zajistit různě dlouhou dobu mezi vytvořením klonů. Tato vylepšení jsou implementována i v ukázkové hře.*

- S žáky probereme, co se v bloku dozvěděli nového, co pro ně bylo na projektu nejtěžší a co nejzajímavější. *Můžeme přidat i zábavný kvíz s otázkami typu: co je na daném scénáři špatně, jaký výraz doplníme do podmínky apod. Můžeme využít aplikace na kvízy, které jsou pro žáky zábavné a vytvářejí i žebříček výsledků. Pomocí těchto otázek zjistíme, s čím mají žáci problémy a jestli je potřeba něco znovu dovysvětlit.*

Obrázek 24: Hotový scénář pro původní postavu kapky



Obrázek 25: Hotový scénář pro klony postavy kapky

4.8 Funkce

Kromě předdefinovaných bloků mohou uživatelé ve Scratchi vytvářet vlastní bloky, které zastávají podobnou úlohu jako funkce v jiných programovacích jazycích. Pomocí nich je možná znovupoužitelnost části kódu programu. Touto funkcionalitou je ve Scratchi implementován koncept zapouzdření.

Doporučený rozsah:

4 kroužky (4 x 90 minut)

Předpoklady pro zvládnutí bloku:

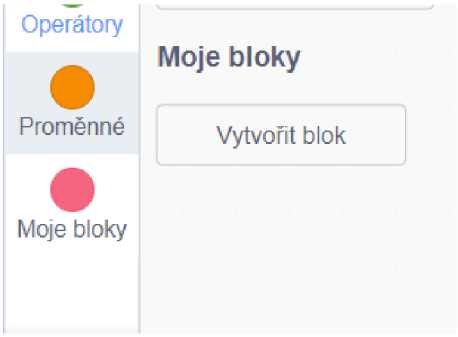
- zvládnutí předchozích bloků kroužku
- znalost základů želví grafiky z předchozího bloku

Cíle bloku:

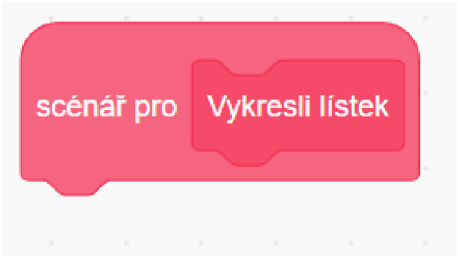
- Žák pochopí princip a důležitost opakovaného použití části kódu – již sestavených bloků.
- Žák se naučí vytvářet a používat vlastní bloky.

Průběh bloku:

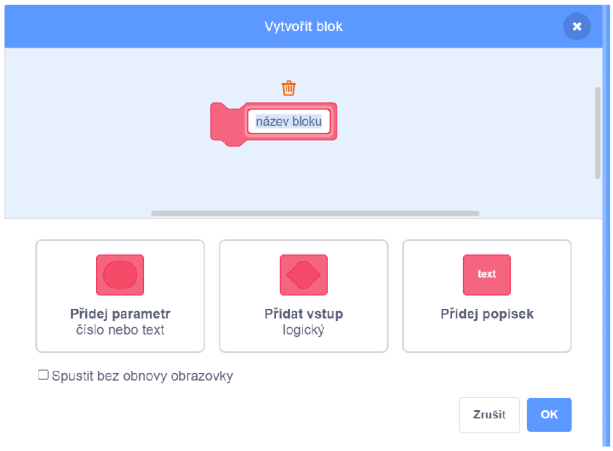
- Připomeneme želví grafiku. Žáci vyjmenují příkazy, které se používají. *Želví grafiku jsme používali, když jsme ukazovali cykly.*
- Samostatná práce: nakreslit květ, v němž každý okvětní lístek bude mít tvar kosočtverce. *Pro připomenutí – kosočtverec má všechny strany stejně dlouhé a vždy dva protilehlé úhly stejně velké. Součet vnitřních úhlů v kosočtverci je* 360 °. *Je vhodné nechat žáky napřed vytvořit kosočtverec samostatně. Po určité době jim s tvorbou daného útvaru pomůžeme.*
- Některý z žáků ukáže své řešení. *Tento úkol by měli žáci zvládnout bez problémů.*
- Samostatná práce: květu přidáme dva další okvětní lístky.
- Samostatná práce: každou stranu okvětního lístku prodloužíme o 10 kroků.
- Na těchto příkladech si ukážeme, že by bylo užitečné mít nějaký blok, který vykreslí celý okvětní lístek. *Upozorníme na redundanci kódu při vytváření dalších okvětních lístků. Případné úpravy je pak možné udělat na jednom místě a není nebezpečí zapomenutí nebo nesprávné úpravy v jednom z výskytů.*



Obrázek 26: Červená sekce Moje bloky

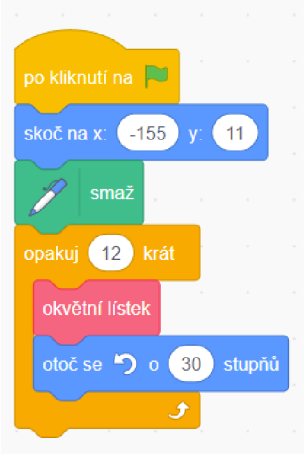


Obrázek 27: Úvodní blok pro scénář funkce

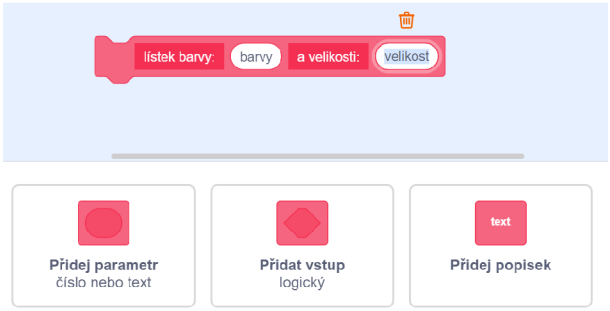


Obrázek 28: Okno pro vytvoření vlastního bloku.

- Ukážeme červenou sekci Moje bloky. *V sekci nejsou předdefinované bloky, vyskytuje se tam pouze tlačítko na vytvoření vlastního bloku, jak je vidět na obrázku 26.*
- Vytvoříme vlastní blok s názvem „Vykresli lístek“ a popíšeme vzniklé bloky. *Okno pro vytvoření vlastního bloku vidíme na obrázku 28. Přidáváním parametrů nového bloku se budeme zabývat později. Při zadání jména nového bloku se tento blok ukáže v sekci Moje bloky a na pracovní plochu se přidá úvodní blok „Scénář pro …“ s jeho jménem, jak je vidět na obrázku 27. Takto vytvořený blok reprezentuje funkci s daným jménem a scénářem.*
- Vysvětlíme, jak se tyto funkce používají. *Vhodné je představit funkce jako stroj – víme, co stroj vykoná, ale nezajímá nás jak. Zařazením bloku se jménem funkce se na daném místě v kódu vykoná scénář této funkce. Takto fungují i přednastavené bloky.*
- Samostatná práce: vytvořit scénář pro blok „Vykresli lístek“. *V původním kódu již žáci vykreslili okvětní lístek, takže lze tuto část scénáře pouze zkopírovat do scénáře funkce.*



Obrázek 29: Scénář pro vykreslení květu s použitím vlastního bloku „Vykresli lístek“



Obrázek 30: Okno pro vytvoření vlastního bloku „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“, kde *barva* a *délka* jsou parametry.

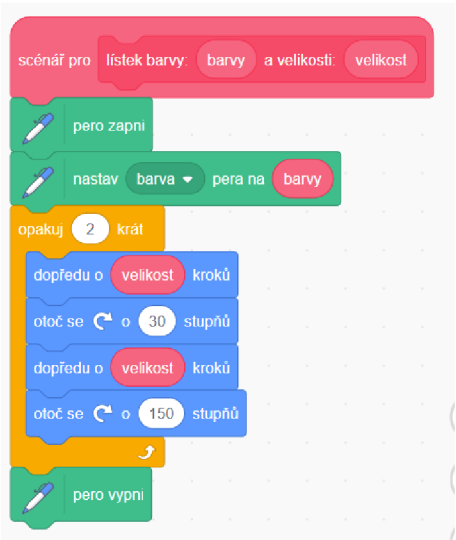
- Funkce by měla být vždy úplná, tak aby za všech okolností fungovala stejně. Diskutujeme s žáky o tom, co bychom měli do funkce doplnit. *V tomto konkrétním případě by funkce měla na začátku zapnout a na konci vypnout pero. Kdyby pero dopředu zapnuté nebylo, lístek by se nevykreslil. Funkce by také měla nastavit barvu pera. Funkce může být požitá pouze u postavy, u které je definovaná. Každá funkce musí disponovat unikátním jménem. Ve Scratchi sice jdou přidat dva vlastní bloky se stejným jménem, ale nejde je od sebe rozlišit.*

- Samostatná práce: začleníme nový blok „Vykresli lístek“ do scénáře postavy. *Tato část je pro většinu žáků intuitivní, ale někdy zapomenou odstranit původní kód.*

- Předvedeme správné řešení a odpovíme na případné dotazy. *Scénář je vidět na obrázku 29.*

- Takto můžeme vytvářet květ určité barvy a velikosti okvětních lístků. Abychom mohli vytvářet květy různých velikostí a barev, potřebujeme předat funkci parametry. Parametr je proměnná, která udává možnosti úpravy scénáře funkce, jako např. barvu a velikost okvětního lístku. Parametry zadáváme při použití bloku funkce. Scénář funkce pak proběhne v závislosti na zadaných parametrech. *Žákům můžeme parametry funkcí ukázat na příkladech bloků – např.„dopředu o x kroků“, přičemž parametr x je číslo.*

- Společně vytvoříme novou funkci „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“, přičemž *barva* a *délka* jsou parametry. Právě zmíněné je ukázáno na obrázku 30.



Obrázek 31: Scénář pro funkci „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“

Obrázek 32: Okno pro vytvoření vlastního bloku „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“, kde *barva* a *délka* jsou parametry.

- Samostatná práce: sestavit scénář pro funkci „Vykresli lístek barvy: *barva* a délky strany lístku: *délka*“. Parametry se ve scénáři funkce chovají jako proměnné.
- Ukážeme správné řešení, jak je vidět na obrázku 31. *Scénář je opět velmi podobný scénáři pro funkci „Vykresli lístek“, jen místo konkrétních hodnot udáváme hodnotové bloky parametrů.*
- Samostatná práce: nakreslit tři různé květy pomocí nové funkce. *Žáci mohou experimentovat s množstvím lístků a úhlem mezi nimi. Květ může vzniknout i z více řad okvětních lístků kolem stejného středu. Květy by neměly být spojeny čarou.*
- Několik žáků ukáže svoje řešení i se scénáři.
- Samostatná práce: každý žák si vybere jeden květ a vytvoří funkci pro jeho vykreslení. *Toto cvičení žákům názorně ukáže, že scénář funkce může v sobě používat bloky jiných funkcí.*
- Ukážeme možné řešení, jak je na obrázku 32, a okomentujeme ho.
- Na závěr bloku si zopakujeme vlastnosti funkcí a jejich vytváření. Poukážeme na to, jak funkce zpřehlednily scénář postavy.

4.9 Závěrečný projekt

Tímto blokem ukončíme kroužek programování. Žáci by měli být schopni zvládnout všechny základní úkoly ve Scratchi.

Doporučený rozsah:

4 kroužky (4 x 90 minut)

Předpoklady pro zvládnutí bloku:

- dopředu vymyšlený projekt každého žáka
- zvládnutí předchozích bloků kroužku

Cíle bloku:

- Žák si procvičí doposud probraná témata.
- Žák si vyzkouší samostatně navrhnout celý projekt.
- Žák si procvičí logické uvažování.

Průběh bloku:

- Se žáky probereme projekty, které jsme v kroužku vytvořili. Připomeneme funkcionality, které jsme použili. *Projdeme všechny větší projekty a pro připomenutí je zobrazíme na projektoru.*
- Každý žák si vymyslí vlastní projekt. V bodech si zapíše, co a jak bude programovat. *Pro inspiraci můžeme uvést příklady klasických jednoduchých her, např. pong*[11]*, had*[12]*, arkanoid*[13] *apod. Zkontrolujeme každý projekt a jeho náročnost. Musíme uvážit i zdatnost konkrétního žáka. V případě neúměrně těžkého, nebo naopak lehkého projektu žákovi nabídneme jeho obměnu.*
- Samostatná práce: vytvořit si vlastní projekt. *Žákům pomáháme s problémy a konzultujeme obsah projektu, ale snažíme se do procesu příliš nezasahovat. Žáci si mohou pomáhat navzájem, ale projekty jsou individuální. Žáci mohou na projektech pracovat i doma.*
- Každý žák předvede svůj projekt, okomentuje ho a odpoví na otázky ostatních. *Na tento bod si ponecháme dostatek času, ideálně po celý kroužek, aby každý žák odprezentoval vlastní projekt podle svého. Je vhodné, aby si žáci projekty navzájem vyzkoušeli.*

[11] Z komunity Scratche např. https://scratch.mit.edu/projects/2718643
[12] Z komunity Scratche např. https://scratch.mit.edu/projects/511466864
[13] Dostupný na adrese: https://scratch.mit.edu/projects/713115791

- Na závěr se uskuteční hlasování o nejzajímavější nápad, nejhezčí zpracování a nejzábavnější hru. *Hlasování by mělo být anonymní. Každý napíše na papír ke každé z jednotlivých kategorii dva různé projekty.*

- Závěrečné rozloučení. *Každý žák napíše na papír, co se mu na kroužku líbilo a co by naopak změnil, jestli kroužek splnil jeho očekávání a který projekt se mu líbil nejvíce. Tuto zpětnou vazbu může použít lektor pro další kroužky, a žáci si tak zpětně projdou celým procesem tvorby. Zpětnou vazbu žáci odevzdají anonymně.*

Závěr

V práci jsem popsala příklady jazyků vhodných pro výuku dětí. Z vizuálních programovacích jazyků se jako nejvhodnější pro výuku základů programování pro děti a mládež v České republice jeví jazyk Scratch. Pro děti je graficky intuitivní a zajímavý, je plně přeložen do českého jazyka. Jeho výhodou je bloková struktura, a proto nedisponuje složitou syntaxí. Popisu historie a webového prostředí se věnuje samostatná kapitola práce.

Hlavním přínosem práce je vytvořený plán uceleného kroužku programování pro děti na druhém stupni základních škol nebo jejich ekvivalentu. Plán je rozdělen do devíti tematických celků a jeho součástí jsou i metodické pokyny pro lektory kroužku. Ke každému bloku jsem vytvořila vlastní příklady hotových projektů v jazyce Scratch. Při vypracování práce jsem se opírala o své zkušenosti kouče kroužků programování pod neziskovou organizací Czechitas, která se věnuje výuce informačních technologií pro ženy a děti.

Kroužek programování by mohl být dobrým základem pro navazující kurz, v němž by se žáci učili složitější programovací techniky, jako je například rekurze, případně i jiné programovací jazyky.

Conclusions

In this thesis, I described examples of languages suitable for teaching children. Of the visual programming languages, the Scratch language appears to be the most suitable for teaching the basics of programming to children and youth in the Czech Republic. It is graphically intuitive and interesting for children and is fully translated into Czech. Its advantage is the block structure, and therefore it does not have a complex syntax. A separate chapter of the thesis is devoted to the description of the history and web environment.

The main contribution of the work is the created plan of a complete programming course for children in upper primary schools or their equivalent. The plan is divided into nine thematic units and it also includes methodological instructions for the course lecturers. I created my own examples of finished projects in the Scratch language for each block. When developing the thesis, I relied on my experience as a coach of programming courses under the non-profit organization Czechitas, which is dedicated to teaching information technology to women and children.

The programming course could be a good basis for a follow-up class, where students would learn more complex programming techniques such as recursion or other programming languages.

A Obsah přiloženého CD/DVD

`doc/`
: Text práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech příloh, a všechny soubory potřebné pro bezproblémové vygenerování PDF dokumentu textu (v ZIP archivu), tj. zdrojový text textu, vložené obrázky apod.

`hotové projekty`
: Složka hotových projektů, které žáci budou vytvářet během kroužku programování. Jednotlivé projekty lze spustit v desktopovém prostředí Scratche nebo nahráním do webového editoru Scratche [7].

`readme.txt`
: Instrukce ke spuštění projektů.

Literatura

[1] *A History of LEGO Education, Part 3: Mindstorms over matter*. Článek o historii Lego mindstorms v anglickém jazyce. Dostupný z: ⟨https://www.brothers-brick.com/2020/02/03/a-history-of-lego-education-part-3-mindstorms-over-matter-feature/⟩.

[2] *LEGO® MINDSTORMS® | Informace*. Oficiální webové stránky Lego mindstorms v českém jazyce. Dostupný z: ⟨https://www.lego.com/cs-cz/themes/mindstorms/about⟩.

[3] *Programovací jazyk Baltík*. Webové stránky jazyku Baltík v českém jazyce. Dostupný z: ⟨https://www.sgpsys.com/cz/⟩.

[4] *Programovací jazyk MakeCode*. Webové stránky jazyku MakeCode v anglickém jazyce. Dostupný z: ⟨https://www.microsoft.com/en-us/makecode⟩.

[5] *Python Introduction*. Webové stránky s úvodem do jazyka Python v anglickém jazyce. Dostupný z: ⟨https://www.w3schools.com/python/python_intro.asp⟩.

[6] *Web Scratch Foundation*. Scratch je spravován Scratch Foundation. Dostupný z: ⟨https://www.scratchfoundation.org/⟩.

[7] *Scratch*. Stránky Scratche. Dostupný z: ⟨https://scratch.mit.edu/⟩.

[8] SWEIGART, Al. *Scratch Game Programming*. Kurz je dostupný online zdarma pro přihlášené uživatele. Dostupný z: ⟨https://www.udemy.com/course/scratch-game-programming/⟩.

[9] PELÁNEK, Radek. *Želví grafika Exkurze do programování, geometrie a umění*. Brno: Computer Press, 2018. ISBN 978-80-251-4905-8.

[10] JANÍK, Tomáš; MAŇÁK, Josef; KNECHT, Petr. *Cíle a obsahy školního vzdělávání a metodologie jejich utváření*. Brno: Paido, 2009. Pedagogický výzkum v teorii a praxi. ISBN 978-80-7315-194-2.

[11] BRENNAN, Karen. Scratch-Ed: an online community for scratch educators. In. *Scratch-Ed: an online community for scratch educators*. 2009, s. 76–78.

[12] PELÁNEK, Radek. *Programátorská cvičebnice: [algoritmy v příkladech]*. Brno: Computer Press, 2012. ISBN 978-80-251-3751-2.

[13] TÖPFER, Pavel. *Algoritmy a programovací techniky*. Praha: Prometheus, 2007. ISBN 978-80-7196-350-9.

[14] YU, Brian; MALAN, David J. *CS50's Introduction to Programming with Scratch*. Kurz je dostupný online zdarma pro přihlášené uživatele. Kurz končí 31.12.2022. Dostupný z: ⟨https://learning.edx.org/course/course-v1:HarvardX+CS50S+Scratch/home⟩.

[15] KŘÍŽOVÁ, Bc. Kateřina. *Vizuální programovací jazyky ve výuce na středních školách*. 2019. Diplomová práce z Masarykovy Univerzity.