UNIVERZITA PALACKÉHO V OLOMOUCI

PEDAGOGICKÁ FAKULTA

Katedra technické a informační výchovy

**Bakalářská práce**

David Květoň

**Sada úloh pro výuku programování ve Scratchi pro ZŠ**

Olomouc 2023 vedoucí práce: doc. RNDr. Petr ŠALOUN

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a uvedl jsem v ní veškerou literaturu a ostatní informační zdroje, které jsem použil.

V Olomouci dne XX. XX. XXXX …………………….

vlastnoruční podpis

Poděkování

Chtěl bych poděkovat panu docentu Šalounovi za cenné rady a ochotu při spolupráci.

David Květoň

**Anotace**

**Abstract**

**Obsah**

[Úvod 6](#_Toc148447919)

[1 Informatické myšlení 7](#_Toc148447920)

[1.1 Vymezení a definice pojmu informatické myšlení 7](#_Toc148447921)

[1.2 Složky informatického myšlení 9](#_Toc148447922)

[1.3 Využití informatického myšlení v běžném životě 10](#_Toc148447923)

[2 Digitální vzdělávání 12](#_Toc148447924)

[2.1 Vize digitálního vzdělávání 12](#_Toc148447925)

[2.2 Strategie digitálního vzdělávání v ČR 13](#_Toc148447926)

[2.3 Cíle strategie vzdělávací politiky ČR do roku 2030+ 13](#_Toc148447927)

[2.4 Nové pojetí výuky informatiky na ZŠ 14](#_Toc148447928)

[3 Vizuální programovací jazyk (VPL) 16](#_Toc148447929)

[3.1 Scratch 16](#_Toc148447930)

[4 PRAKTICKÁ ČÁST BAKALÁŘSKÉ PRÁCE 17](#_Toc148447931)

[4.1 Rozdělení úloh 17](#_Toc148447932)

[Závěr 19](#_Toc148447933)

[Seznam použitých zdrojů 20](#_Toc148447934)

[Seznam obrázků 23](#_Toc148447935)

[Seznam grafů 24](#_Toc148447936)

[Seznam tabulek 25](#_Toc148447937)

Úvod

# INFORMATICKÉ MYŠLENÍ

Myšlení je poznávací proces, který je charakteristický hned několika body. Skládá se z vnitřních, implicitních myšlenkových operací. Probíhá jednak na vědomé, kontrolované a řízené úrovni (myšlení logické, induktivní, deduktivní), jednak na neuvědomované úrovni (myšlení intuitivní). Obvykle se dá usměrňovat vůlí (myšlení volní). Může však probíhat bez volního úsilí (myšlení asociativní), dokonce i proti volnímu úsilí (myšlení vtíravé). [1]

## Vymezení a definice pojmu informatické myšlení

Spojení informatické myšlení by se zjednodušeně dalo vysvětlit, jako schopnost myslet jako informatik při řešení problémů. Computational thinking teacher resources uvádí relativně dobře srozumitelnou a konkrétní definici: Informatické myšlení je postup řešení problému, který zahrnuje mimo jiné následující charakteristiky:

* Formulovat problémy způsobem, který umožňuje jejich strojové řešení.
* Logicky uspořádat a zkoumat data.
* Reprezentovat data prostřednictvím abstrakcí, jako jsou modely a simulace.
* Automatizovat řešení pomocí algoritmického myšlení (jako posloupnost kroků).
* Odhalit, prozkoumat a provést možná řešení s cílem odhalit nejúčinnější kombinaci činností a zdrojů.
* Zobecňovat a přenášet tento postup řešení problémů do nejrůznějších dalších oblastí. [16]

Všechny tyto dovednosti podporuje další nezbytná součást informatického myšlení, a tou jsou předpoklady a postoje. Jedinec by měl být sebejistý tváří v tvář složitosti, vytrvalý při řešení obtížného problému. Měl by také snášet nejednoznačnosti a být schopný vypořádat se s otevřenými problémy. A v neposlední řadě je důležité, aby jedinec dokázal dorozumět a také spolupracovat s ostatními, a tak dosáhnout společného cíle. [2]

Do českého vzdělávání se požadavek na rozvoj informatického myšlení žáků dostává v roce 2014 prostřednictvím vládního dokumentu Strategie digitálního vzdělávání do roku 2020, který řadí rozvoj informatického myšlení žáků mezi své tři prioritní cíle. [3]

Selby a Woolard chtěli na diskuzi o informatickém myšlení vrhnout nové světlo. Proto se rozhodli zformulovat jednu ne příliš širokou definici a shrnout používanou terminologii. Pro tyto účely zanalyzovali celkem 22 různých definic pojmu „computational thinking“. Součástí výstupu jejich výzkumu je tabulka s přehledem pojmů, které bývají s informatickým myšlením spojovány či dokonce ztotožňovány, včetně vyjádření, zda příslušný pojem může být zahrnut do definice informatického myšlení či nikoliv: [4]

**Tabulka 1:** Terminologie spojená s definicí informatického myšlení

|  |  |  |
| --- | --- | --- |
| **Pojem** | **Lze zahrnout?** | **Zdůvodnění** |
| Myšlenkový proces | ano | Literární zdroje se shodují. |
| Abstrakce | ano | Literární zdroje se shodují. |
| Dekompozice (rozklad) | ano | Literární zdroje se shodují. |
| Logické myšlení | ne | Příliš široký pojem, nedostatečně definovaný. |
| Algoritmické myšlení | ano | Dobře definovaný napříč různými obory. |
| Řešení problémů | ne | Příliš široký pojem, je důkazem použití IM, nerozvíjí ho. |
| Hodnocení (evaluace) | ano | Dobře definovaný napříč různými obory. |
| Zobecnění (generalizace) | ano | Dobře definovaný napříč různými obory. |
| Navrhování systémů | ne | Důkaz použití IM. |
| Automatizace | ne | Důkaz použití IM. |
| Informatický obsah | ne | Důkaz použití IM. |
| Modelování a simulace | ne | Důkaz použití IM |

Selby a Woolard na základě svých zjištění navrhuji definici tohoto znění: *Informatické myšlení je činnost, typicky orientovaná na výsledek, spojována, ale ne výlučně omezena, na řešení problémů. Jedná se o kognitivní proces, který odráží schopnost:*

* *abstrahovat,*
* *rozkládat problém na podproblémy (dekompozice),*
* *myslet algoritmicky,*
* *hodnotit,*
* *zobecňovat (generalizace).* [5]

Jinými slovy je informatické myšlení dle Selbyho a Woolarda přístup zaměřený na řešení problémů zahrnující myšlenkový proces používající abstrakci, dekompozici, algoritmický přístup, hodnocení a zobecňování.

Je tedy zřejmé, že informatické myšlení nemá téměř nic společného s obsluhou počítače, s uživatelským přístupem k technologiím, protože „takové používání počítače informatické myšlení nerozvíjí“ [6].

E.W. Dijkstra, známý nizozemský informatik, který bývá řazen i mezi průkopníky informatiky, dokonce řekl: „Informatika není o počítačích o nic víc než astronomie o dalekohledech.“ [6]

Tento známý citát zmiňuje i Pelánek a objasňuje ho následovně: „Počítače, stejně jako dalekohledy, jsou jen prostředek. Informatik, jenž se stará jen o svůj počítač, je na tom stejně jako hvězdář, který se pro všechnu starost o svůj dalekohled zapomněl dívat na nebe.“ [7]

## Složky informatického myšlení

Pro rozvoj informatického myšlení žáků je třeba vymezit, jaké složky koncept informatického myšlení zahrnuje. Do velké míry nám s tímto může pomoci definice Selbyho a Woolarda (2013, s. 5), neboť právě schopnost abstrakce, dekompozice, algoritmického myšlení, zobecňování a hodnocení jsou považovány za základní předpoklady informatického myšlení.

Cílem **abstrakce** je problém zjednodušit, a to určením částí problému, které jsou důležité a podstatné, a částí, které jsou naopak nepodstatné. Prostředkem pro znázornění abstrakce může být model, simulace, diagram, abstraktní jazyk apod. Jako příklad použití abstrakce v matematice lze uvést slovní úlohy, u kterých k vyřešení napomáhá vyjádřit si klíčové informace úlohy více abstraktním jazykem, například algebraicky.

**Dekompozice** je proces, při kterém je problém rozdělen na dílčí podproblémy. Tento přístup má mnoho výhod. Zaprvé z velkých zdánlivě neřešitelných problémů učiní sérii nebo strukturu menších problémů, jejichž řešení je snazší, protože je řešiteli problému například už známé. Zároveň rozklad na podproblémy vytváří vhodné podmínky pro týmovou práci.

**Algoritmické myšlení** představuje zejména proces tvorby algoritmů a algoritmických řešení, která neslouží jen k řešení jedné úlohy, ale jsou řešením celé skupiny úloh, které se od sebe liší vstupními údaji. Vzhledem k tomu, že je algoritmické myšlení vnímáno jako dominantní složka informatického myšlení.

**Generalizace** neboli **zevšeobecňování** je přístup založený na rozeznávání vzorů, podobností a spojitostí, které vedou k pochopení podstaty zkoumaného jevu. Generalizace umožňuje rychleji vyřešit nový problém na základě zkušeností z řešení předchozího podobného problému. Neinformatickým příkladem generalizace může být učení se správné výslovnosti anglických slov. Když se žák v rámci určitého slova naučí vyslovovat určitou 15 posloupnost písmen, dokáže ji správně vyslovit i v případě, kdy na ni narazí v jiném dosud neznámém slově. Poslední ze základních složek informatického myšlení je hodnocení. Jeho cílem je ověření toho, že navržené řešení je dobré a účelné. Za tímto účelem je potřeba vždy zhodnotit řešení z různých hledisek jako je správnost fungování, rychlost, efektivita, náročnost použití řešení z pohledu uživatele, řešení nestandardních situací apod. Součástí procesu hodnocení je i tzv. debugging tedy vyhledávání a ladění chyb.

## Využití informatického myšlení v běžném životě

Pokud bychom chtěli přiblížit využití informatického myšlení v běžném životě obyčejného člověka, museli bychom hledat nějaký často opakovaný a jednotvárný proces, při kterém se pracuje s množstvím nějakých položek. Například proces nakupování si můžeme urychlit uspořádáním položek na nákupním seznamu dle rozmístění zboží v konkrétní prodejně. Tím předejdeme složitému hledání potravin v seznamu. Dalším příkladem by mohl být výběr pokladny v nákupním centru, abychom čekali co možná nejkratší dobu ve frontě, či uspořádání potravin v lednici dle data trvanlivosti. [3]

Díky informatickému myšlení můžeme zachraňovat i lidské životy. Například při řetězové transplantaci ledvin. Právě zapojení informatického myšlení na straně organizátorů dárcovského systému vedlo k rozpoznání a posouzení jeho řešitelnosti. Následná spolupráce s informatiky vedla ke zlepšení situace a nalezení efektivního algoritmu k vyřešení celého problému. Díky sestavenému řetězci je možné zachránit hned několik životů najednou. V Česku probíhají řetězové transplantace již několik let. [11]

# DIGITÁLNÍ VZDĚLÁVÁNÍ

V současnosti jsme svědky mnoha společenských proměn, které zasahují bez výjimky všechny oblasti lidské činnosti. Abychom byli schopni na tyto proměny reagovat, je třeba na ně adekvátně připravit vzdělávací systém. Vzhledem k významným změnám ve společnosti, způsobeným dynamickým rozvojem, je nutné tomuto vývoji přizpůsobit obsah, metody a formy vzdělávání.

Stejně jako je tomu v ostatních oborech lidské činnosti, lze pozorovat technologické trendy, které ovlivňují pedagogiku, školství a vzdělávání. Digitální technologie mají zásadní podíl na nutnosti přehodnotit vzdělávací cíle a měnit zažité postupy a vazby v našem vzdělávacím systému. [10]

## Vize digitálního vzdělávání

Vizí digitálního vzdělávání je, aby vzdělávací systém vybavil každého jedince bez rozdílu takovými kompetencemi, které mu umožní se uplatnit v informační společnosti a využívat nabídky otevřeného vzdělávání v průběhu celého života. [11]

Koncepce Digitální Česko v. 2.0, kterou v roce 2013 schválila Vláda České republiky, konkrétně uvádí: *„Informační technologie by měly postupovat celým procesem výuky na základních školách, nikoli, jen v předmětech typu Práce s počítačem. Plné zapojení moderních technologií do výuky všech předmětů vnímá stát jako nezbytné v rámci posunu vzdělávacího systému od prostého memorování faktů k důrazu na čtenářskou gramotnost, komunikační dovednosti a logické myšlení.“*

Součástí usnesení vlády k této koncepci je soubor opatření, z nichž se jedno opatření týká problematiky vzdělávání a ukládá MPSV ve spolupráci s MŠMT vypracovat strategii pro zvýšení digitální gramotnosti a rozvoj elektronických dovedností občanů Cílem strategie je nastavit podmínky a procesy ve vzdělávání, které toto digitální vzdělávání umožní realizovat. [12]

## Strategie digitálního vzdělávání v ČR

Vláda České republiky reaguje na přetrvávající rychlý vývoj v oblasti digitálních technologií

a je si vědoma nutnosti a potřeby implementovat moderní technologie do výuky. V souvislosti s rychle postupující digitalizací společnosti je bezpochyby žádoucí, aby vzdělávací systém,

s přihlédnutím k dynamice těchto změn, byl dostatečně flexibilní a adekvátně připraven.

Tomuto má napomoci Strategie vzdělávací politiky ČR do roku 2030+. Jedná se o aktuálně platný klíčový dokument, který navazuje na strategický dokument pro oblast vzdělávání vydaný pod názvem Strategie vzdělávací politiky České republiky do roku 2020. Strategie 2030+ je stěžejním dokumentem pro rozvoj vzdělávacího systému v ČR pro nadcházející desetiletí let 2020 – 2030. Zmíněná strategie si klade za cíl zmodernizovat vzdělávací systém tak, aby děti

i dospělí obstáli v dynamickém a neustále se měnícím světě 21. století, dále připravit ho na nové výzvy a řešit přetrvávající problému, které v česku panují. Dokument vymezuje dva hlavní strategické cíle a pět strategických linií, které představují cesty a nástroje k realizace těchto cílů. [13]

## Cíle strategie vzdělávací politiky ČR do roku 2030+

Dnešní žáci se velmi výrazně liší oproti svým předchozím vrstevníkům. Za společný socializační znak soudobé generace je považováno hlavně využívání digitálních technologií. Předmětná strategie si proto klade následující cíle:

* využívat moderní technologie k dosažení nově definovaných vzdělávacích cílů,
* vytvořit podmínky pro rozvoj digitálního vzdělávání všech žáků a učitelů,
* zvýšit úroveň kompetencí v oblastech užívání digitálních technologií, informatického myšlení a digitální gramotnosti,
* uzpůsobit vzdělávací systém, aby byl schopen přiměřeně se přizpůsobit dynamickému prostředí a pokroku spojeného s rozvojem nových technologií, digitalizace

a internacionalizace,

* snažit se zvýšit úroveň digitálních dovedností a informatického myšlení,
* zahrnout informační a datovou gramotnost, komunikaci a spolupráci, mediální gramotnost, tvorbu digitálního obsahu, bezpečnost v on-line prostředí, ale i řešení problémů a kritické myšlení do procesu vzdělávání. [13]

## Nové pojetí výuky informatiky na ZŠ

Doposud se žáci věnovali většinou práci s textovými, tabulkovými a prezentačními editory. Nové pojetí výuky informatiky kromě základů uživatelských dovedností pro práci se zařízeními a aplikacemi přináší a klade důraz:

* na informatické myšlení,
* strukturovaně přemýšlet (i s využitím počítačů a aplikací),
* žáci by se měli učit pracovat s informacemi,
* žáci by měli umět popsat problém, analyzovat jej a hledat funkční řešení,
* porozumět principům digitálních technologií,
* bezpečně a eticky využívat digitálních technologie a týmovou práci. [13]

Inovovat obsah vzdělávací oblasti informatika s důrazem na rozvoj informatického myšlení žáků výrazně přispěl projekt, který byl realizován v rámci Operačního programu Výzkum, vývoj a vzdělávání. Konkrétně se jedná o projekt „Podpora rozvíjení informatického myšlení“ (dále jen ve zkratce PRIM), na jehož spolufinancování se podílela Evropská unie. Hlavním příjemcem a garantem projektu, jehož realizace byla úspěšně ukončena k datu 30. 11. 2020, byla Jihočeská univerzita v Českých Budějovicích. Mezi další spolupracující partnery se řadí všechny pedagogické fakulty v ČR a Národní ústav pro vzdělávání.

V rámci tohoto projektu se také podařilo úspěšně vytvořit ucelené sady materiálů pro výuku,

a to pro všechny stupně škol. Skutečnosti, proč využít zrovna, a právě prostředí vizuálního programovacího jazyka Scratch pro praktickou část mé bakalářské práce přispívá i fakt, že byl zařazen projektem PRIM, jakožto výchozí produkt, ve kterém by se měli začít učit programovat již žáci od 1. stupně ZŠ a plynule v něm navázat a pokračovat na 2. stupni ZŠ.

**Obrázek 1:** Ukázka ucelené sady výukových materiálů dle projektu PRIM (2021)



# VIZUÁLNÍ PROGRAMOVACÍ JAZYK

V této kapitole se budeme zabývat pojmem vizuální programovací jazyk (ve zkratce VPL – anglicky Virtual Programming Language). Jak už název vypovídá, nebude se jednat o klasické konvenční programování, kde je zdrojový kód zapisován pomocí konstruktů datového programovacího jazyka a reprezentován zejména v textové podobě. V případě vizuálního programování je kód reprezentován graficky, pomocí obrázků. Ve výpočetní technice vizuální programovací jazyk je programovací jazyk, který uživatelům umožňuje vytvářet programy manipulací s programovými prvky graficky na rozdíl od klasického programování, kde se program vytváří pomocí textové syntaxe. [17] VPL umožňuje programování s vizuálními výrazy, prostorovým uspořádáním textu a grafických symbolů, které se používají buď jako prvky syntaxe nebo sekundární notace. Například mnoho VPL je založeno na myšlence „boxů a šipek“, kde se s boxy nebo jinými objekty obrazovky zachází jako s entitami spojenými šipkami, čarami nebo oblouky, které představují relace (nebo také vztahy) mezi těmito prvky. [18]

## Motivace a cíl VPL

První motivací pro vznik vizuálního programovacího jazyka je lepší pochopení syntaxe kódu programu a celkové zjednodušení programování pomocí různých vizuálních pomůcek. Příkladem může být zvýrazňování syntaxe ve vývojových prostředí pomocí různých barev, které odlišují proměnnou od funkce apod. V **Obrázek 2** můžete vidět názornou ukázku zvýraznění syntaxe pomocí odlišností barev v jazyce C#. Další motivací je snaha přivést k programování grafiky a designéry, ale zároveň u toho neodradit programátory. Ve vizuálním programování je totiž kladen důraz hlavně na estetiku výsledných programů. [20]

**Obrázek 2:** Ukázka zvýraznění syntaxe ve vývojovém prostředí Visual Studio Code

Obsah obrázku text, snímek obrazovky, software, Multimediální software

Popis byl vytvořen automaticky

Obecným cílem VPL je zpřístupnit programování začátečníkům a podporovat programátory na třech různých úrovních: [19]

1. **Syntax:** VPL využívají ikony/bloky, formuláře a diagramy s cílem snížit nebo dokonce eliminovat možnost syntaktických chyb a pomáhají s uspořádáním programovacích primitiv k vytvoření správně utvořených programů
2. **Sémantika:** VPL mohou poskytovat mechanismy k odhalení významu programovacích primitiv. To může zahrnovat pomocné funkce poskytující dokumentační funkce zabudované do programovacích jazyků
3. **Pragmatika:** VPL podporují studium toho, co programy znamenají v konkrétních situacích. Tato úroveň podpory uživatelům umožňuje umístit artefakty vytvořené pomocí VPL do určitého stavu, aby zkoumali, jak program zareaguje na tento stav. Například s programovacím jazykem pro robota Thymio mohou uživatelé dostat robota do určitého stavu, aby viděli, jak zareaguje, tedy které senzory budou aktivovány.

## Příklady vizuálních programovacích jazyků

Za jeden z nejznámějších a nejvíce využívaných vizuálně programovacích jazyků v oblasti vzdělávání se dá považovat Scratch. A jelikož se praktická část mé bakalářské práce zaměřuje právě na tento jazyk, věnuji mu samostatnou podkapitolu a popíšu jej hlouběji. Existuje však mnoho dalších vizuálních programovacích jazyků, které jsou navrženy s různými cíli   
a zaměřením. Proto jich níže pár uvedu a stručně popíšu jejich využití.

### Blockly

Byl vyvinutý společností Google. Jedná se o javascriptovou knihovnu, která je open source, tedy může být využívaná širokou veřejností zdarma. Blockly poskytuje vizuální rozhraní pro vytváření vlastních programovacích editorů pro tvorbu specifických typů aplikací, jako jsou vzdělávací nástroje, hry nebo nástroje pro vývoj softwaru. Knihovnu Blockly využívá i Scratch. Scratch byl totiž postaven do roku 2019 na technologii Flash. Po skončení podpory této technologie přešel Scratch právě na technologii Blockly.

**Obrázek 3:** Ukázka kódu v Blockly Dostupné z: https://upload.wikimedia.org/wikipedia/commons/9/95/Blockly\_Conditions\_main.png

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automaticky

### App Inventor

Tento VPL umožňuje uživatelům vytvářet mobilní aplikace pro platformu Android pomocí vizuálního programování. Tento nástroj byl vyvinut na MIT (Massauchusetts Institute of Technology). Je free a open source, tedy přístupný všem zdarma. Díky tomuto nástroji mohou uživatelé bez hlubší znalosti klasického programování vytvářet funkční aplikace. Jelikož je tento nástroj vizuálně založen, usnadňuje porozumění základům tvorby aplikací, a proto je také vhodný pro výuku ve školách a vzdělávacích institucích.

**Obrázek 4:** Ukázka vizuálního prostředí App Inventoru. Dostupné z: https://cdn-learn.adafruit.com/guides/cropped\_images/000/001/277/medium640/AppInvent09.PNG?1535067563

Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automaticky

### Lego Mindstorms EV3 / NXT Software

Dsadsadsadsadsadsadsadsa

### Tynker

## Scratch

Jedním z předních zástupců vizuálního programování je právě programovací jazyk Scratch. Je to intuitivní protředek pro vstup do světa programování, navržený pro začátečníky a děti. Byl vytvořen na MIT (Massachusetts Institute of Technology). Scratch umožňuje vytvářet interaktivní příběhy, hry a všemožné animace pomocí přetahování a spojování bloků, které reprezentují programovací příkazy. Jeho vizuální rozhraní a bloková struktura odstraňují potřebu psát kód, což umožňuje uživatelům snadno porozumět základním principům, programování a algoritmů. Jazyk Scratch nejenom podporuje rozvoj programovacích dovedností, ale také kreativitu, logické myšlení a řešení problémů. Díky svému interaktivnímu prostředí umožňuje uživatelům experimentovat s různými nápady a vidět okamžitě výsledky svého úsilí. Tento jazyk se stal oblíbeným nástrojem pro výuku programování ve školách   
a vzdělávacích institucích po celém světě díky své přístupnosti   
a schopnosti zaujmout a motivovat nové programátory a tvůrce.

### Blockly

### Kodu

# PRAKTICKÁ ČÁST BAKALÁŘSKÉ PRÁCE

V této části mé bakalářské práce vytvořím sadu úloh v programovacím jazyce Scratch. Sada bude obsahovat celkem 18 úloh v prostředí programovacího jazyka Scratch. Úlohy budou mít za cíl, aby si žák osvojil základní programovací kompetence a dostatečně se tak připravil na programování ve vyšších programovacích jazycích jako je Python, Java, C# apod. Tato sada je určená pro druhý stupeň ZŠ, tedy pro žáky 5-9 třídy. Sadou úloh bude žáky provázet postavička se jménem „Bajtík“. Jedná se o animovaného netopýra a jeho jméno je odvozené od pojmu „Bajt“, což je v informatice základní jednotka kapacity počítačové paměti. Všechny vypracované úlohy budou popsány níže s podrobným popisem zadání, řešení a případných doplňujících úkolů. Ke každé úloze bude taky k dispozici metodický list, který bude sloužit jako pomůcka pro pedagogy.

## Rozdělení úloh

Úlohy budou rozděleny do tří kategorií. V každé kategorii si žák osvojí určité znalosti v programovacím jazyce Scratch a bude tak rozvíjet svou znalost nejen v jazyce Scratch, ale taky obecně v programování. Tyto kategorie se budou od sebe odlišovat určitou obtížností, která je uzpůsobená schopnostem žáka. Kategorie na sebe navazují, a to tím způsobem, že v obtížnostně těžších kategoriích budou použity nabité znalosti z kategorií jednodušších. Každá kategorie bude obsahovat celkem 5 úloh + 1 závěrečnou úlohu ve formě většího projektu, který bude sumarizovat danou kategorii a budou tam použito vše, co se žák naučil. Výjimkou je akorát poslední kategorie „Profík“, ve které bude úloh pouze 5, které však budou koncipovány ve formě větších projektů.

**Kategorie Elév**

V této části jsou úlohy koncipovány tak, aby se žák zejména seznámil s prostředím programovacího jazyka Scratch. Pracuje se zde převážně s jednoduchými cykly, kostýmy, pozadím a zvuky. Žáci se zde i naučí pracovat s rozšířením pero, které je v rámci Scratche přístupné volně. Řešení takových úloh vychází časově na cca 30 minut.

**Kategorie Kadet**

Kategorie Kadet se bude věnovat složitějším cyklům, podmínkám a práce s proměnou. Budou zde i využity dříve naučené znalosti z kategorie Kadet. Řešení takových úloh vychází časově na cca 45 minut, tedy 1 vyučovací jednotku.

**Kategorie Profík**

V poslední kategorii Profík bude 5 větších projektů, kde budou žáci využívat vše, co se naučili v předchozích kategoriích. Projekty budou většinou ve formě vytvoření nějaké hry. Řešení takových projektů zabere žákům cca 90 minut, což přesahuje 1 vyučovací jednotku a mohou sloužit právě jako dlouhodobější projekty v rámci výuky.

# ÚLOHY KATEGORIE ELÉV

## Létající Bajtík

**Tabulka 3:** Metodický list - Létající Bajtík

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Letající Bajtík |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 30 minut |
| Rozsah: | 1 pozadí, 1 postava, 2 scénáře, 10 bloků |
| Klíčová slova: | animace, pohyb, kostým, postava |
| Anotace materiálu: | Žák vytvoří animaci pohybu postavy Bajtík. |

**Zadání:** Rozpohybuj Bajtíka, aby náhodně létal po celé obrazovce a vytvoř mu animaci, která bude simulovat mávání jeho křídel.

**Řešení:** Žák bude mít k dispozici před vytvořenou postavu Bajtík a pozadí. Postava bude měnit dva kostýmy pomocí cyklu „opakuj stále“. Poté v druhém cyklu „opakuj stále“ nastavíme, aby se postava posouvala dopředu o 10 kroků a nastavíme vlastnost, aby se po nárazu na okraj odrazila.

**Obrázek 5** Kód Bajtík

**Obsah obrázku text, snímek obrazovky, Grafický software, Multimediální software

Popis byl vytvořen automaticky**

**Obrázek 6** Postava Bajtík

**Obsah obrázku snímek obrazovky, text, Multimediální software, Grafický software

Popis byl vytvořen automaticky**

**Obrázek 7** Pozadí

**Obsah obrázku snímek obrazovky, text, Multimediální software, software

Popis byl vytvořen automaticky**

**Doplňující úkoly:** Zkus změnit nebo přidat nové pozadí, Zkus přidat novou postavu a pokus se o animaci pohybu této postavy.

## Pohyb pomocí tlačítek

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Pohyb pomocí tlačítek |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 30 minut |
| Rozsah: | x pozadí, 1 postava, 6 scénářů, 18 bloků |
| Klíčová slova: | animace, pohyb, tlačítko, pozadí |
| Anotace materiálu: | Žák naprogramuje pohyb postavy „Bajtík“ pomocí tlačítek a animaci mávání křídel. |

**Zadání úlohy:** Naprogramuj pohyb Bajtíka pomocí tlačítek na klávesnici, aby mohl létat vpravo, vlevo, nahoru a dolů. Zamez tomu, aby se při změně směru neotáčel a zároveň, aby se při nárazu na okraj odrazil. Naprogramuj i animaci, aby Bajtík mával křídly.

**Řešení úlohy:** Pomocí bloků „po stisku klávesy“ nastavíme pohyb postavy a přepnutí pozadí. Je dobré seznámit žáky se souřadnicemi. Souřadnice X určuje horizontální pozici postavy (zleva doprava). Souřadnice Y určuje vertikální pozici postavy (nahoru dolů). Pro postavu je potřeba i nastavit, aby se po nárazu na okraj odrazila a také aby se neotáčela. Poté už jen přidáme dříve naučenou animaci pohybu.

**Obrázek 8:** Kód Bajtík

Obsah obrázku text, snímek obrazovky, design

Popis byl vytvořen automaticky

**Obrázek 9:** Kostým Bajtík

Obsah obrázku snímek obrazovky, text, Multimediální software, Grafický software

Popis byl vytvořen automaticky

**Doplňující úkoly:** Přidej si do programu více pozadí a naprogramuj, aby se po stisku klávesy mezerník změnilo pozadí.

## Divadlo

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Divadlo |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 4 postavy, 4 scénářů, 8 bloků |
| Klíčová slova: | tlačítko, vyslání zprávy, obdržení zprávy, kliknutí na tlačítko |
| Anotace materiálu: | Žák vytvoří přehlídku kostýmů a pomocí dvou tlačítek bude měnit u postav tělo nebo hlavu. |

**Zadání úlohy:** Vytvoř divadlo z kostýmů zvířátek. Vyber si několik zvířecích postav z dostupné nabídky Scratche a uprav si zvířata tak, abys oddělil hlavu a tělo. Poté naprogramuj před vytvořená tlačítka „Tlačítko Hlava“ a „Tlačítko Tělo“, aby se po kliknutí na ně vyslala zpráva, pomocí které se potom bude měnit příslušná partie (hlava, tělo).

**Řešení úlohy:** Zde se žáci seznámí s novým blokem „vyšli zprávu“. Pomocí bloku „vyšli zprávu“ dokážou po kliknutí na tlačítko vyslat zprávu pro vykonání určité události. V tomto případě, pokud bude kliknuto např. na tlačítko „Hlava“. Vyšle se zpráva a v kódu postavy „Hlava“ si nastavíme, že po obdržení zprávy změníme kostým.

**Obrázek 10:** Kód Tlačítko Tělo

**Obsah obrázku text, Písmo, logo, Grafika

Popis byl vytvořen automaticky**

**Obrázek 11:** Kód Tlačítko hlava

**Obsah obrázku text, Písmo, logo, žlutá

Popis byl vytvořen automaticky**

**Obrázek 12:** Kód Tělo

**Obsah obrázku text, Písmo, snímek obrazovky, logo

Popis byl vytvořen automaticky**

**Obrázek 13:** Kód Hlava

**Obsah obrázku text, Písmo, logo, snímek obrazovky

Popis byl vytvořen automaticky**

Obrázek 14: Kostýmy Hlava

**Obsah obrázku text, snímek obrazovky, software, Grafický software

Popis byl vytvořen automaticky**

**Obrázek 15:** Kostýmy Tělo

**Obsah obrázku text, snímek obrazovky, software, Počítačová ikona

Popis byl vytvořen automaticky**

**Doplňující úkoly:** Přidej více postav. Zkus změnit hudbu, která hraje na pozadí. Zkus si vytvořit a naprogramovat vlastní tlačítko, které bude například po kliknutí měnit pozadí.

## Mandala

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Mandala |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 20 minut |
| Rozsah: | 1 pozadí, 1 postava, 1 scénář, 12 bloků |
| Klíčová slova: | cyklus, pero, otáčení, stupně, pohyb |
| Anotace materiálu: | Žák naprogramuje mandalu pomocí rozšíření pero a za využití cyklu. |

**Zadání úlohy:** Pokus se nakreslit mandalu pomocí možného rozšíření Scratche „Pero“ a za využití cyklu s určitým počtem opakování.

**Řešení úlohy:** Zde budou žáci seznámeni s rozšířením „Pero“. Postavě „Pencil“ tedy dáme možnost, aby po příkazu „pero zapni“ začalo kreslit dle pokynů. Mandalu poté nakreslíme pomocí cyklu s určitým počtem opakování. Žáci zde budou mít svobodu v tom jaké parametry si nastaví (každému se může vykreslit něco jiného). V obrázcích níže je příklad kódu a výstup programu.

**Obrázek 16:** Kód pro vytvoření Mandaly

Obsah obrázku snímek obrazovky, text, diagram, design

Popis byl vytvořen automaticky

**Obrázek 17:** Výsledná Mandala

Obsah obrázku kruh, umění, design

Popis byl vytvořen automaticky

**Doplňující úkoly:** Zkus upravit počet opakování nebo stupeň otáčení se, aby si vytvořil jiný zajímavý obrázek.

## DJ pult

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | DJ Pult |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 6 postav, 6 scénářů, 12 bloků |
| Klíčová slova: | tlačítko, kliknutí na tlačítko, zvuk |
| Anotace materiálu: | Žák vytvoří 6 postav, které si upraví jako tlačítko. Každému tlačítko přidá i zvířecí zvuk, který se přehraje po kliknutí na dané tlačítko. |

**Zadání úlohy:** Pomož DJ Bajtíkovi vytvořit tu největší zvířecí diskotéku. Vytvoř 6 postav, které budou sloužit jako tlačítka. Přidej je do DJ pultu a naprogramuj tlačítka tak, aby po kliknutí na ně se přehrál tebou zvolený zvuk zvířete.

**Řešení:** Žáci si zde primárně nacvičí úpravu kostýmů. Vytvoří si svůj DJ pult se 6 tlačítky. Využijeme bloku „Po kliknutí na mě“, který provede přehrání zvoleného zvuku – Žáci si tady mohou vyhrát s volbou zvuků. V obrázcích níže je ukázka vytvořeného DJ pultu.

**Obrázek 18:** 6 vytvořených tlačítek, které jsou umístěné na DJ Pultu

**Obsah obrázku snímek obrazovky, kreslené, kruh, Grafika

Popis byl vytvořen automaticky**

**Obrázek 19:** Kód pro tlačítko

**Obsah obrázku text, snímek obrazovky, Operační systém, design

Popis byl vytvořen automaticky**

**Doplňující úkoly:** Přidej a naprogramuj více tlačítek. Zkus změnit hudbu v pozadí.

## Malování

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Malování |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 60 minut |
| Rozsah: | 1 pozadí, 5 postav, 11 scénářů, 51 bloků |
| Klíčová slova: | Pero, Kreslení, Cyklus, Podmínka, Tlačítko, Barva |
| Anotace materiálu: | Žák naprogramuje ve Scratchi primitivní program malování. |

**Zadání úlohy:** Vytvoř primitivní program Malování. Na předem vytvořenou paletu přidej sebou zvolené barvičky. Naprogramuj barvičky, aby se po kliknutí na danou barvu změnila barva pera. Pero naprogramuj, aby kreslilo, dokud bude stisknuta myš.

**Řešení úlohy:** Na začátku projektu bude mít žák k dispozici paletu, která je součástí pozadí a celkem dvě postavy - „Tuzka“ a „Cerna“. Postava „Cerna“ reprezentuje černou skvrnu na paletě. Po kliknutí na skvrnu se vyšle zpráva. Postava „Tuzka“ po obdržení zprávy změní barvu pera na danou barvu. Poté pomocí cyklu „opakuj stále“ nastaví, aby pero kreslilo po dobu, co uživatel má stisknutou myš. Tímto způsobem žák přidá a naprogramuje i ostatní barvičky. V obrázcích níže můžete vidět řešení takového programu.

**Obrázek 20:** Přidané barvičky na paletě

**Obsah obrázku snímek obrazovky, text, design

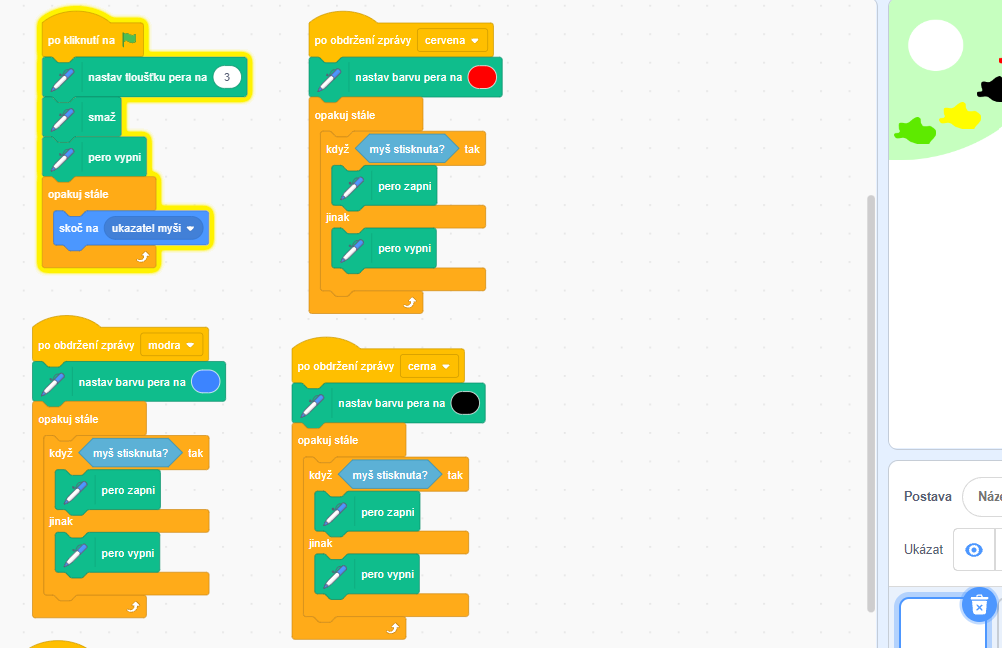
Popis byl vytvořen automaticky**

**Obrázek 21:** Kód pro barvu Modra

**Obsah obrázku text, Písmo, logo, snímek obrazovky

Popis byl vytvořen automaticky**

**Obrázek 22:** Kód postavy Tuzka

****

**Doplňující úkoly:** Zkus vytvořit a naprogramovat tlačítko, které bude upravovat tloušťku pera.

# ÚLOHY KATEGORIE KADET

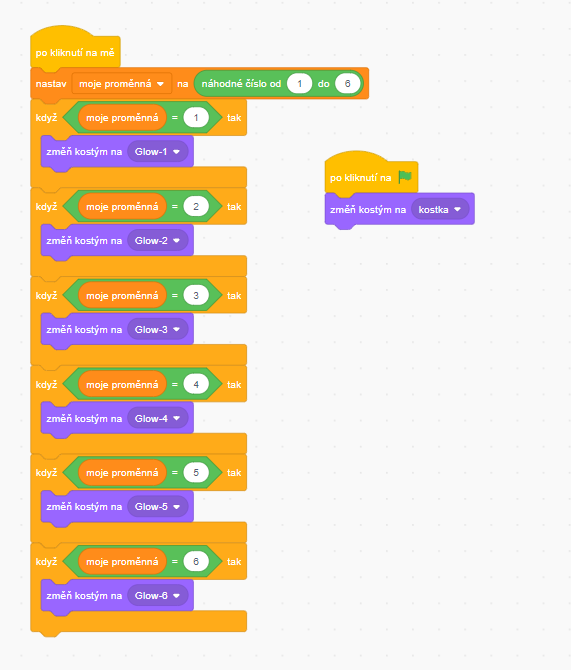
## Kostka

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Kostka |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 1 postava, 2 scénáře, 22 bloků |
| Klíčová slova: | kostka, náhodný generátor čísel, proměnná, podmínka, cyklus |
| Anotace materiálu: | Žák vytvoří kostku, která bude generovat náhodná čísla 1-6. |

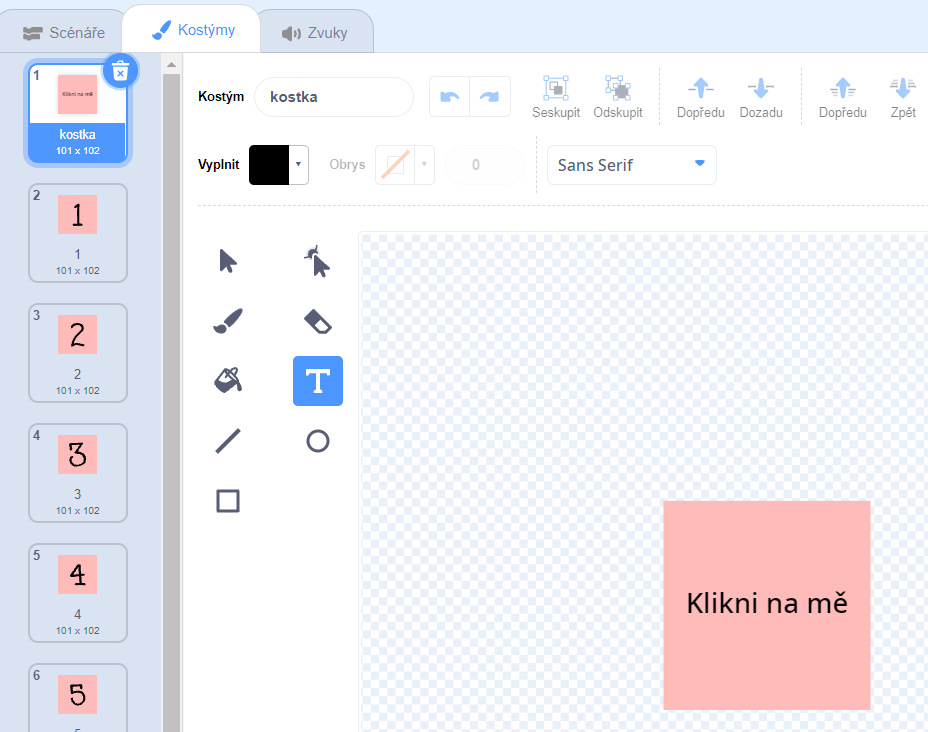
**Zadání:** Vytvoř kostku, která bude po kliknutí generovat čísla od 1 do 6.

**Řešení:** V této úloze se žáci seznámí s proměnnou, podmínkou a blokem „náhodné číslo od do“. Je potřeba jim vysvětlit, jak proměnná funguje. Doporučuji vysvětlit jako krabici, do které něco vložíme/uložíme. V tomhle případě budeme do proměnné ukládat pomocí generátoru náhodných čísel, číslo od 1 do 6. Poté pomocí podmínky budeme ověřovat jaké číslo je v proměnné uložené. Pokud např. padne číslo 6, změníme kostým kostky, aby se zobrazila 6. V obrázcích níže je ukázka kódu.

**Obrázek 23:** Kód Kostka

****

**Obrázek 24:** Kostýmy

****

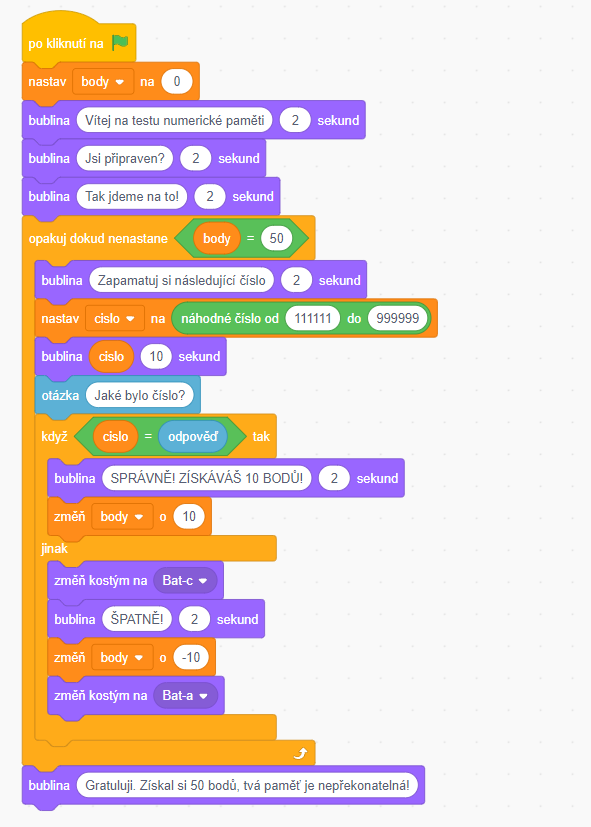
## Numerická paměť

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Numerická paměť |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 1 postava, 1 scénář, 18 bloků |
| Klíčová slova: | náhodný generátor čísel, proměnná, podmínka, cyklus, otázka |
| Anotace materiálu: | Žák vytvoří test numerické paměti, kde bude generovat náhodná vícero ciferná čísla a poté uživateli podá otázku, zdali si pamatuje vygenerované číslo. |

**Zadání:** Vytvoř test numerické paměti. Vygeneruj 6 ciferné číslo, které po 10 sekundách zmizí. Poté se uživatele zeptáš, jaké bylo číslo. Pokud uživatel odpoví správně, získá 10 bodů. Pokud špatně, ztrácí 10 bodů. Jakmile uživatel získá bodů 50, ukonči program.

**Řešení:** Vytvoříme proměnnou body a proměnnou číslo. Do bodů budeme ukládat skóre za správnou/špatnou odpověď. Do proměnné číslo budeme vždy ukládat náhodně vygenerované číslo, které bude mít 6 cifer. Pomocí cyklu „opakuj dokud nenastane“ vložíme podmínku, aby se cyklus opakoval dokud proměnná body nebude mít hodnotu 50. V samotném cyklu potom vygenerujeme dané číslo, uložíme ho do proměnné a vypíšeme uživateli po 10 sekund. Poté číslo zmizí a zeptáme se pomocí bloku „otázka“ jaké bylo číslo. Co uživatel napíše se uloží do bloku „odpověď“. Srovnáme odpověď s proměnnou číslo a pokud se rovná – přičteme 10 bodů, pokud ne – ubereme 10 bodů. V obrázku níže můžete vidět ukázku kódu.

**Obrázek 25:** Kód testu numerické paměti

****

**Doplňující úkoly:** Zkus naprogramovat, aby se cifra vygenerovaného čísla postupně navyšovala.

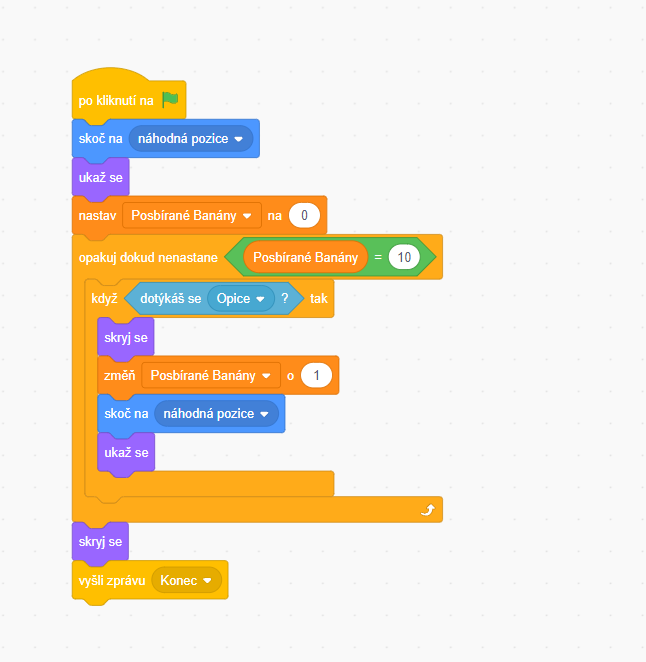
## Sběr banánů

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Sběr banánů |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 2 postavy, 6 scénářů, 27 bloků |
| Klíčová slova: | podmínka, cyklus, dotyk postavy, proměnná, pohyb postavy |
| Anotace materiálu: | Žák naprogramuje opičku, která bude mít za úkol sbírat banány. Bude zapotřebí naprogramovat i postavu banány, která se bude objevovat a mizet dle potřeby. |

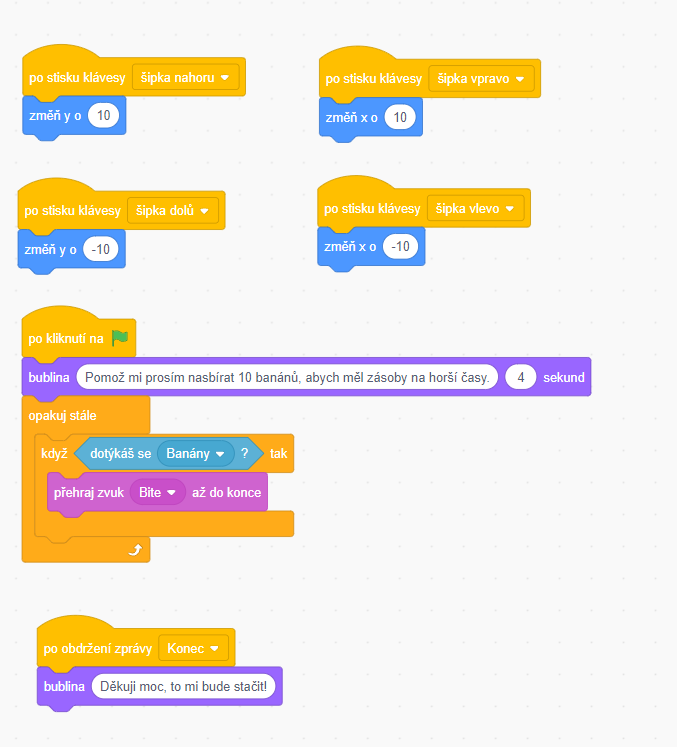
**Zadání:** Naprogramuj hru, kde opička bude mít za úkol nasbírat 10 banánů. Po spuštění hry se vždy na náhodné pozici zobrazí banán. Jakmile se postava opičky dotkne banánu, banán zmizí a znova se objeví na jiné náhodné pozici. Až opička nasbírá 10 banánů, ukonči hru.

**Řešení:** Postava Opice bude mít v sobě kód pro pohyb postavy pomocí bloků „po stisku klávesy“. Poté je potřeba přidat cyklus opakuj stále, kde se pomocí podmínky bude ověřovat, jestli se postava dotýká postavy Banány. V postavě Banány vytvoříme proměnnou „Posbírané Banány“, do které budeme ukládat počet posbíraných banánů. Pomocí cyklu „opakuj dokud nenastane“, nastavíme aby se cyklus opakoval, dokud proměnná nebude mít hodnotu 10. V samotném cyklu je potom podmínka – pokud se dotýkáš postavy Opice, zvyš hodnotu proměnné o 1, skryj postavu Banány a přemísti na náhodnou pozici. V obrázcích níže můžete vidět řešení úlohy.

**Obrázek 26:** Kód Banány

****

**Obrázek 27:** Kód Opice

****

**Doplňující úkoly:** Ozvuč postavy, přidej efekty, pohraj si s pozadím. Pokus se o to, aby se na obrazovce objevilo zároveň více banánů.

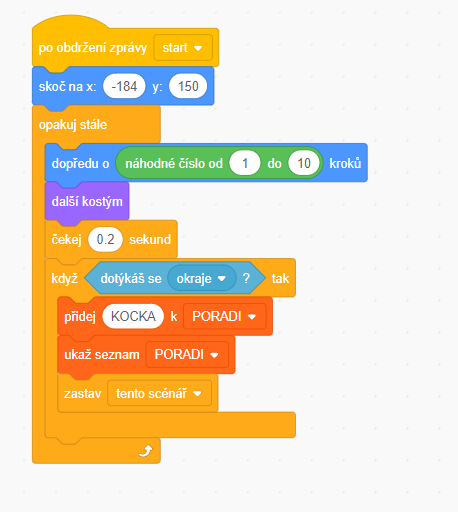
## Dostihy

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Sběr banánů |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 5 postav, 5 scénářů, 50 bloků |
| Klíčová slova: | podmínka, cyklus, dotyk postavy, proměnná, seznam, náhodný generátor čísel |
| Anotace materiálu: | Žák naprogramuje dostihy. Vytvoří 5 postav a bude upravovat jejich rychlost pomocí náhodného generátoru čísel. Jaká postava se první dotkne cíle, zvítězí. Pořadí se potom vypíše ve vytvořeném seznamu. |

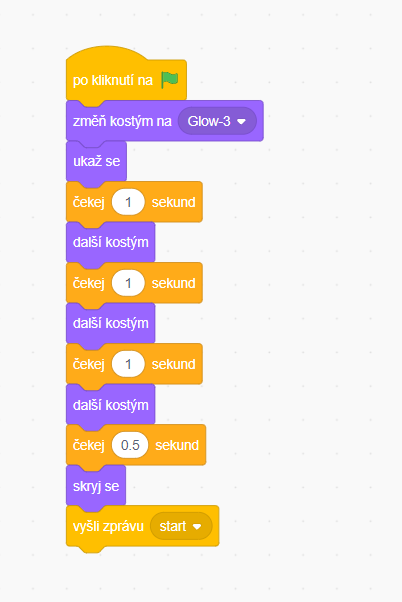
**Zadání:** Naprogramuj dostihový závod zvířátek. Vytvoř 5 zvířecích postav a náhodně upravuj jejich rychlost pohybu. Jakmile se postava dotkne cíle, ulož jí do seznamu a zobraz tak finální pořadí závodníků.

**Řešení:** V této úloze se žáci naučí pracovat se seznamem. Funguje stejně jako proměnná, akorát se do něj dá uložit více věcí zároveň. Každá postava bude mít téměř stejný kód. V každé bude cyklus „opakuj stále“, ve kterém se bude náhodně upravovat rychlost postavy od 1 do 10. Pokud se postava bude dotýkat okraje, znamená to, že je v cíli a název dané postavy se uloží do seznamu „PORADI“. Na konci závodu budou tedy v seznamu vypsány všechny postavy v pořadí, ve kterém dorazili do cíle. V obrázku níže ukázka kódu jedné z postav.

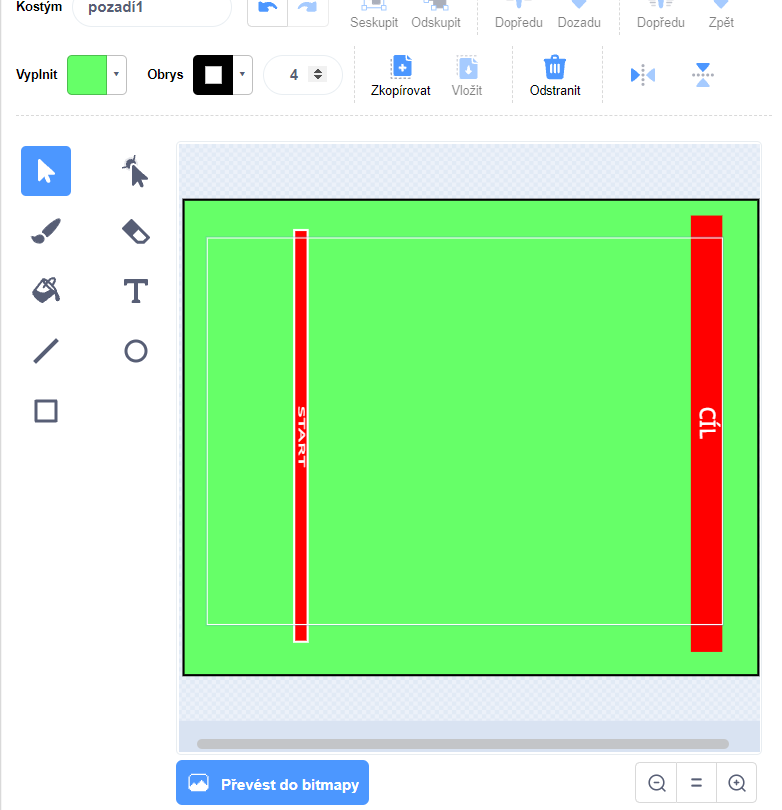
**Obrázek 28:** Kód Kočka



**Obrázek 29:** Kód animace startu



**Obrázek 30:** Pozadí



**Doplňující úkoly:** Pohraj si s grafickou stránkou programu, přidej animace a zvuk pro každou postavu. Než začne závod, odstartuj ho pomocí jednoduché animace prolínání čísel a zvuku výstřelu.

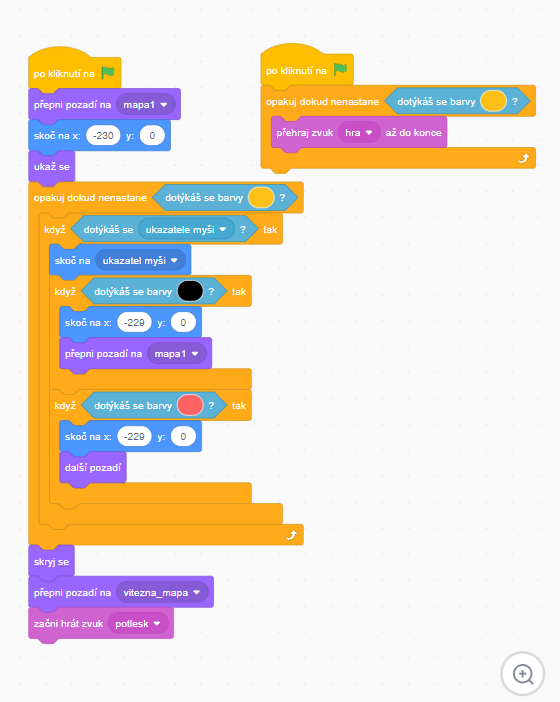
## Minihra s kuličkou

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Minihra s kuličkou |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 5 postav, 5 scénářů, 50 bloků |
| Klíčová slova: | podmínka, cyklus, dotek barvy |
| Anotace materiálu: | Žák vytvoří minihru, kde bude za úkol přesunout kuličku přes vytvořenou dráhu, aniž by se kulička dotkla hrany vytyčené dráhy. |

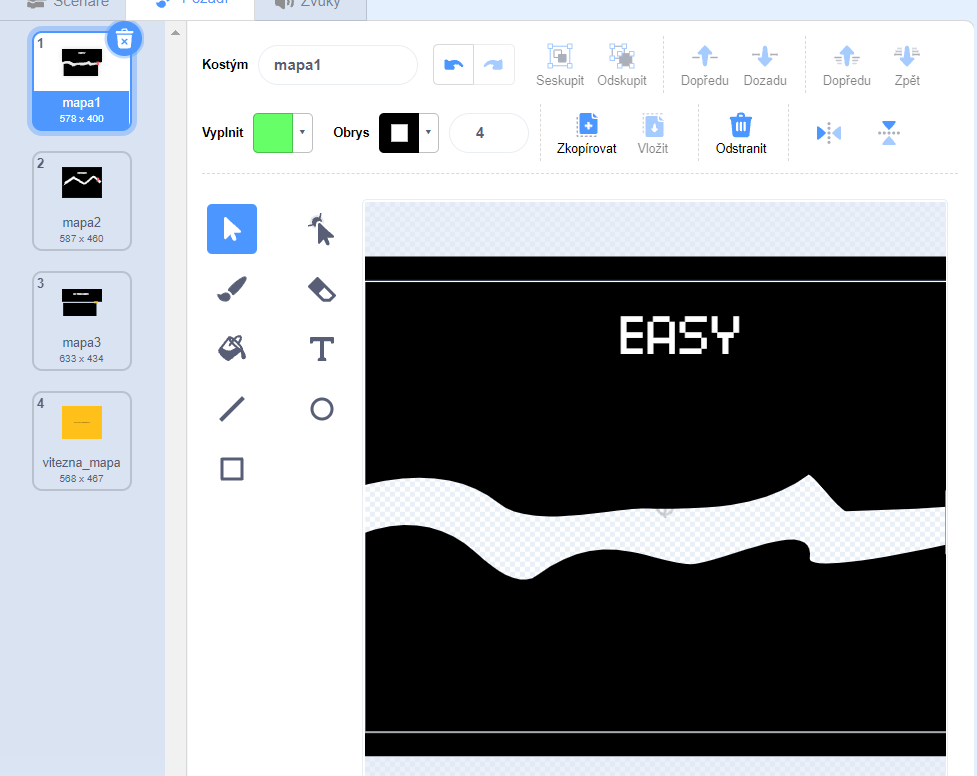
**Zadání:** Vytvoř minihru pro červenou kuličku. Hra bude obsahovat celkem tři dráhy, které nakreslíš (Ideálně, aby se stupňovala náročnost drah). Červená kulička poté bude muset postupně projít všechny dráhy, aniž by se dotkla okraje dráhy.

**Řešení:** Nejdříve vytvoříme dráhy. Postačí nám dva černé obdélníky, které si upravíme dle své volby. Kulička bude procházet po bílé ploše a pokud se dotkne černé barvy, skočí na začátek mapy. Ovládání kuličky nastavíme na ukazatel myši. Využijeme bloků „dotýkáš se barvy“, „opakuj dokud nenastane“ a „skoč na ukazatel myši“. V obrázku 28 můžete vidět řešení takové úlohy.

**Obrázek 31:** Kód pro kuličku

****

**Obrázek 32:** Ukázka dráhy

****

**Doplňující úkoly:** Vytvoř a přidej více map do programu. Pokus se uživateli po úspěšném překonání všech map vypsat jeho počet chyb.

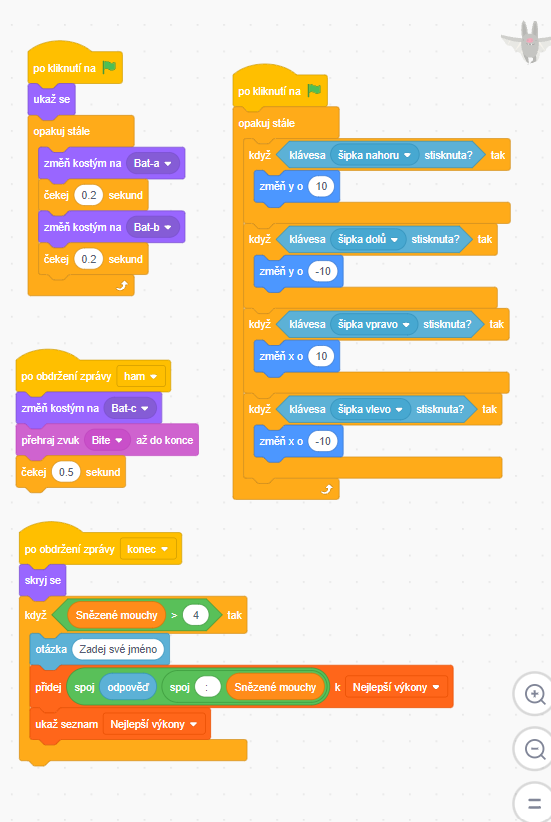
## Lov hmyzu

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Lov hmyzu |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 90 minut |
| Rozsah: | 1 pozadí, 2 postav, 6 scénářů, 47 bloků |
| Klíčová slova: | podmínka, cyklus, pohyb postavy, interakce s druhou postavou, seznam, stopky, vyšli zprávu |
| Anotace materiálu: | Žák naprogramuje hru, ve které netopýr loví mouchy, které náhodně poletují po obrazovce. Hra potrvá minutu, poté bude mít hráč možnost zapsat své skóre do seznamu výsledků. |

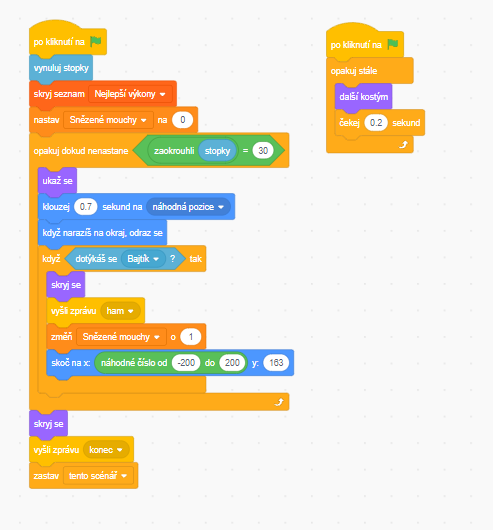
**Zadání úlohy:** Vytvoř hru, kde Bajtík bude ovládán pomocí šipek a jeho úkol bude lovit mouchy. Mouchu naprogramuj tak, aby létala náhodně po celé obrazovce. Pokaždé, co se Bajtík dotkne mouchy, započítá se mu bod. Hra potrvá celkem 1 minutu a po konci si hráč bude mít možnost zapsat své skóre na seznam výsledků (ale pouze v tom případě že nasbíral alespoň 10 much).

**Řešení:** Pro postavu Bajtík nám bude stačit, když naprogramuje pohyb – tentokrát pomocí cyklu „opakuj stále“ a podmínky s blokem „klávesa x stisknuta?“, tento způsob pohybu postavy je plynulejší. Dále tu máme dva scénáře po obdržení zprávy „ham“ a „konec“. Po obdržení zprávy „ham“ přehrajeme zvuk a změníme kostým. Po obdržení zprávy „konec“ provedeme uložení jména hráče a jeho skóre do seznamu „Nejlepší výkony“ (viz Obrázek 30). V postavě Moucha je hlavní část programu, kde se kontroluje čas pomocí bloku „stopky“, který měří čas programu od spuštění. Hra je tedy nastavená na 30 sekund. Moucha bude náhodně klouzat po obrazovce, když narazí na okraj, odrazí se. Pokud se dotýká postavy Bajtík, vyšle se zpráva „ham“, změníme proměnou „Snězené Mouchy“ a skočíme na náhodnou pozici, odkud moucha vylétne. V Obrázku 31 najdete ukázku řešení.

**Obrázek 33:** Kód postavy Bajtík

****

**Obrázek 34:** Kód postavy Moucha



**Doplňující úkoly:** Přidej do hry zvukové efekty, animace postav, nakresli vlastní pozadí.

# KATEGORIE PROFÍK

## Trénink přesnosti

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Trénink přesnosti |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 45 minut |
| Rozsah: | 1 pozadí, 1 postava, 3 scénáře, 32 bloků |
| Klíčová slova: | čas, stopky, seznam, průměr, cyklus |
| Anotace materiálu: | Žák naprogramuje trénink přesnosti, kde bude za úkol 30x kliknout na postavu, která se po každém kliknutí přemístí na náhodnou pozici. Po každém kliknutí se měří čas reakce, poté se vypočítá průměrný čas reakce. |

**Zadání úlohy:** Vytvoř program, který otestuje tvou rychlost přesnosti v kliknutí na určitý cíl. Test bude spočívat v tom, že uživatel bude muset 30x kliknout na postavu. Při každém kliknutí se postava přemístí na jinou pozici a zaznamenáš čas uživatelovi reakce. Poté vypočítáš průměrný čas jeho reakce, vypíšeš výsledek a dáš uživateli možnost zapsat se do listiny výsledků.

**Řešení:** Vytvoříme seznam „reakce“, do kterého se vždy bude ukládat čas reakce kliknutí na postavu Bajtík. Dále vytvoříme proměnnou „Zbývá nakliknutí“, která nám bude ukazovat kolik zbývá pokusů a nastavíme ji na hodnotu 30. V samotném programu pak jen za pomocí bloku „po kliknutí na mě“ naprogramuje průběh (viz Obrázek 32). Po obdržení zprávy „konec“ se provede finální výpočet. Budeme postupně brát prvky ze seznamu „reakce“ a přičítat je postupně do proměnné „soucet“. Musíme však začít od druhého prvku, protože první kliknutí na postavu slouží jako start testu, tedy se nemůže započítávat. Takhle projdeme celý seznam a poté provedeme aritmetický průměr (součet reakcí / počet pokusů) a uložíme jej do proměnné výsledek, kterou uživateli vypíšeme. Poté se pomocí bloku „otázka“ zeptáme na jméno uživatele a uložíme jeho výsledek do seznamu „Listina výsledků“ (viz Obrázek 33).

**Obrázek 35**: Kód postavy Bajtík 1

**Obsah obrázku text, snímek obrazovky, Písmo, diagram

Popis byl vytvořen automaticky**

**Obrázek 36:** Kód postavy Bajtík 2

**Obsah obrázku text, snímek obrazovky, Písmo, logo

Popis byl vytvořen automaticky**

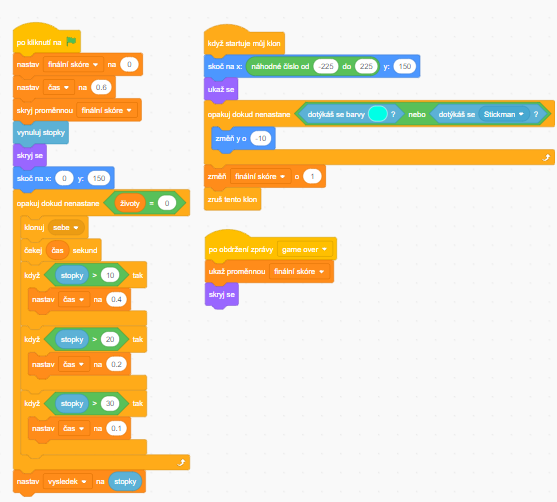
## Dodgeball

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Dodgeball |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 90 minut |
| Rozsah: | 2 pozadí, 2 postava, 6 scénářů, 56 bloků |
| Klíčová slova: | klon, klonuj sebe, interakce s postavou, čas |
| Anotace materiálu: | Žák vytvoří hru, kde se postava snaží vyhýbat míčům, které padají shora dolů. Každých 10 sekund se bude intenzita padajících míčů zvyšovat. |

**Zadání úlohy:** Vytvoř hru, kde budeš ovládat postavu zleva doprava. Její úkol bude vyhýbat se míčům, které budou padat se shora dolů. Padající míče naprogramuj tak, aby se po 10 sekundách změnila intenzita jejich padání. Postava bude mít 3 životy. Za každé uhnutí míče, započítej jeden bod. Za každé trefení míčem uber život. Na konci vypiš skóre.

**Řešení úlohy:** V této úloze poprvé využijeme blok „klonuj sebe“. Tento blok vytvoří kopii postavy a pomocí bloku „když startuje můj klon“ nastavíme co se s danou kopií bude dít. V tomto případě nastavíme, aby klon skočil na náhodnou horizontální pozici, vertikálně je nastavený staticky na souřadnici y = 150. Klon se bude měnit svou vertikální souřadnici směrem dolů, tedy bude padat dokud nenastane, že se dotýká postavy Stickman a nebo barvy, která je na dolní plošině pod postavou Stickman, pak se klon zruší a skočí se do hlavního cyklu, který probíhá dokud má hráč víc jak 0 životů. Za využití stopek budeme měřit čas programu. Po uplynutí každých 10 sekund budeme měnit proměnou „čas“, která bude určovat, jak dlouho se bude čekat na vytvoření dalšího klonu. V Obrázku 34 můžete vidět řešení takového programu. V Obrázku 35 je program pro postavu Stickman, kde je scénář pro pohyb postavy a poté cyklus, který hlídá počet životů postavy.

**Obrázek 37:** Kód postavy Ball



**Obrázek 38:** Kód postavy Stickman

Obsah obrázku text, snímek obrazovky, Barevnost, design

Popis byl vytvořen automaticky

## Tetris

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Tetris |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 90 minut |
| Rozsah: | 2 pozadí, 1 postava, 3 scénáře, 38 bloků |
| Klíčová slova: | klon, klonuj sebe, interakce s postavou, otáčení postavy |
| Anotace materiálu: | Žák naprogramuje napodobeninu známé hry tetris. |

**Zadání úlohy:** Vytvoř napodobeninu hry Tetris. Jde o padající bloky různých tvarů. Bloky zkus vytvořit v jiném grafickém programu a nahraj je do Scratche.

**Řešení úlohy:** Bloky můžeme vytvořit např. v jednoduchém programu Malování (popř. pokročilejší programy jako GIMP, Adobe Photoshop, Adobe Illustrator) a nahrajeme je do kostýmů (Obrázek 37). V hlavním cyklu programu budeme využívat bloku „klonuj sebe“. Na každý další klon se bude čekat pomocí proměnné „konec“. Pro každý klon potom nastavíme, aby se se zvolil náhodný kostým bloku a dokud se nedotýká černé barvy (tzn. dokud se nedotýká spodku hracího pole nebo dalšího bloku), bude blok klouzat směrem dolů. Pomocí šipek nastavíme pohyb doleva, doprava a pomocí mezerníku umožníme otáčet s blokem o 90 stupňů. Pokud se klon bude dotýkat vršku hracího pole znamená to konec a vyšleme zprávu „game over“, která změní pozadí hry a je konec. Obrázek 36 ukazuje řešení takového programu.

**Obrázek 39:** Kód postavy Blok

**Obsah obrázku snímek obrazovky, text, design

Popis byl vytvořen automaticky**

**Obrázek 40:** Kostýmy postavy Blok

**Obsah obrázku snímek obrazovky, software, Počítačová ikona, Multimediální software

Popis byl vytvořen automaticky**

## Flappy Bajtík

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Flappy Bajtík |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 90 minut |
| Rozsah: | 1 pozadí, 2 postavy, 4 scénáře, 30 bloků |
| Klíčová slova: | skákací hra, skok postavy, proměnná, interakce s postavou |
| Anotace materiálu: | Žák vytvoří hru podobnou známé hře „Flappy Bird“. Jde o skákací hru, kde se postava snaží vyhnout překážkám. |

**Zadání úlohy:** Vytvoř hru, podobnou známé hře „Flappy Bird“. Místo ptáčka ale nahradí náš známý Bajtík. Jeho úkolem bude přelétnout mezi dvěma sloupy. Jediné tlačítko pohybu bude mezerník, po jehož stisknu postava poskočí. Pokud se postava dotkne sloupu, ukonči hru a vypiš skóre.

**Řešení úlohy:** Vytvoříme sloupy, které se budou vždy pohybovat proti postavě Bajtík v náhodné výšce. Pokud se nedotknou postavy Bajtík a jejich souřadnice bude menší jak -230 (tzn budou na levém kraji) – skočí zpátky doprava na náhodnou souřadnici y (viz Obrázek 39). Postavu Bajtík naprogramujeme tak, aby dokud se nedotýká postavy „Sloupy“, skákala po stisknutí klávesy mezerník. To dosáhneme pomocí proměnné „skok“. Na začátku programu je nastavená na 0 a po každé stisknutí mezerníku se změní o 5 (pokud je hodnota proměnné menší jak 5). V hlavním cyklu se potom v každém opakování snižuje o hodnotu -0.2. Díky tomu se dá hezky simulovat animace skoku postavy. V Obrázku 38 můžete vidět řešení programu.

**Obrázek 41:** Kód postavy Bajtík

**Obsah obrázku snímek obrazovky, text, design

Popis byl vytvořen automaticky**

**Obrázek 42:** Kód postavy Sloupy

**Obsah obrázku text, snímek obrazovky, logo, Písmo

Popis byl vytvořen automaticky**

**Obrázek 43:** Kostým Sloupy

**Obsah obrázku text, snímek obrazovky, diagram, software

Popis byl vytvořen automaticky**

## Útok asteroidů

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Útok asteroidů |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 120 minut |
| Rozsah: | 1 pozadí, 6 postav, 14 scénářů, 100 bloků |
| Klíčová slova: | klonuj sebe, cyklus, podmínka, proměnná, vyšli zprávu, výroková logika |
| Anotace materiálu: | Žák naprogramuje hru, kde vesmírná loď bude mít za úkol odrazit útok asteroidů, které padají směrem na planetu. |

**Zadání úlohy:** Naprogramuj hru, kde vesmírná loď bude bránit planetu od padajících asteroidů. Loď bude mít 3 životy a jako zbraň bude mít laser, který může vystřelit po asteroidu a zničit jej. Planeta bude mít 100 životů, po každém poškození asteroidem se planetě ubere 10 životů. To stejné platí pro styk vesmírné lodě s asteroidem – loď ztrácí 1 život. Pokud planeta dokáže přežít – pogratuluj hráčovi. Pokud planeta nepřežije, zmizí a hráčovi se vypíše „Game Over“.

**Řešení:** V této hře figurují čtyři postavy. Vesmírná Loď, Planeta, Asteroid a Laser. Vesmírné lodi naprogramujeme pohyb pomocí cyklu „opakuj stále“. Dále budeme pomocí stejného cyklu kontrolovat proměnnou životy – pokud se postava dotkne postavy Asteroid, ubere se jeden život, pakliže už postava Vesmírná Loď nemá životy, provede se animace výbuchu a postava zmizí. Obrázek 41 ukazuje kód této postavy. Postavu Asteroid naprogramujeme tak, aby vždy skočila na náhodnou souřadnici x, souřadnice y bude staticky nastavena na vrchní část obrazovky. V hlavním cyklu „opakuj x krát“ nastavíme počet opakování na 50. V samotném cyklu poté budeme vytvářet klon postavy Asteroid, který bude padat směrem dolů, dokud nenarazí do postavy Planeta, Vesmírná loď, nebo Laser – poté zmizí (viz Obrázek 43). Pokud skončí všech 50 opakování, vyšle se zpráva „victory“ a hra skončí výherně (viz Obrázek 45). Postavu Laser naprogramujeme tak, aby skočila na pozici postavy Vesmírná loď, poté po stisku klávesy mezerník se vytvoří klon, který bude měnit svoji y souřadnici směrem nahoru dokud nenarazí na okraj nebo do Postavy Asteroid (viz Obrázek 42). Postava Planeta bude mít proměnnou životy a pomocí cyklu „opakuj dokud nenastane“ bude pouze snižovat proměnnou o 10, pokud se bude dotýkat postavy Asteroid (viz Obrázek 44). Po skončení cyklu vyšle zprávu „game over“ a hra skončí prohrou (viz Obrázek 45).

**Obrázek 44:** Kód postavy Vesmírná loď

**Obsah obrázku text, snímek obrazovky, design

Popis byl vytvořen automaticky**

**Obrázek 45:** Kód postavy Laser

**Obsah obrázku text, snímek obrazovky, design

Popis byl vytvořen automaticky**

**Obrázek 46:** Kód postavy Asteroid

**Obsah obrázku snímek obrazovky, text, diagram, design

Popis byl vytvořen automaticky**

**Obrázek 47:** Kód postavy Planeta

**Obsah obrázku text, snímek obrazovky, Písmo, diagram

Popis byl vytvořen automaticky**

**Obrázek 48:** Kód postavy Konec hry

**Obsah obrázku text, snímek obrazovky, Písmo, design

Popis byl vytvořen automaticky**

## Had

|  |  |
| --- | --- |
| **Metodický list** | |
| Název materiálu: | Had |
| Vzdělávací oblast: | Informační a komunikační technologie |
| Vyučovací předmět: | Informatika |
| Věkové určení: | 2. stupeň ZŠ |
| Časová dotace: | 120 minut |
| Rozsah: | 4 pozadí, 3 postavy, 21 scénářů, 134 bloků |
| Klíčová slova: | klonuj sebe, cyklus, podmínka, proměnná, vyšli zprávu, výroková logika, interakce s postavou |
| Anotace materiálu: | Žák naprogramuje hru Had, která bude fungovat pro dva hráče. Had sbírá potravu - po každém snězení potravy se had zvětšuje. Pokud had narazí do druhého hada nebo narazí na kraj hracího pole, vyhrává ten druhý had a naopak. |

**Zadání úlohy:** Naprogramuj hru Had pro dva hráče. Pravidla jsou následující:

Pokud Had1 narazí do těla Had2 (stejně i naopak - vyhrává Had2.

Pokud Had1 narazí do hlavy Had2 (stejně i naopak) – vyhrává ten had, který nasbíral více jídla.

Pokud Had1 nebo Had2 narazí na kraj hracího pole – vyhrává automaticky ten druhý had.

**Řešení úlohy:** Pohyb Hada nastavíme pomocí proměnné „směr“, kterou nastavíme na hodnoty nahoru, dolu, doleva a doprava podle toho jaká klávesa šipky je stisknuta. Poté pomocí proměnné „delka“ budeme zvyšovat délku hada. Funguje to tak, že pokud se postava Had dotkne postavy Jidlo, proměnná „delka“ se změní o 0.1 (viz Obrázek 48). V hlavním cyklu postavy Had se potom bude neustále klonovat tělo hada přesně o aktuální nastavené délce. (viz Obrázek 47). Konec hry je vyřešen tak, že se čeká, dokud se had dotkne buď černé barvy (tzn. okraje mapy), a nebo druhého hada, který se odlišuje barvou. Poté se vyšle příslušná zpráva, kdo vyhrál. Pokud se Had1 dotkne hlavou postavy Had2, tak se vyhodnotí skóre dvou hadů a opět pomocí „vyšli zprávu“ se určí vítěz a ukončí se hra. Obrázek 47 je řešení kódu pro postavu hada. Obrázek 48 ukazuje kód pro postavu jídla, které slouží jako potrava pro hada.

**Obrázek 49:** Pohyb postavy Had1

**Obsah obrázku text, Písmo, logo, snímek obrazovky

Popis byl vytvořen automaticky**

**Obrázek 50:** Hlavní kód postavy Had1

**Obsah obrázku snímek obrazovky, žlutá, Barevnost, design

Popis byl vytvořen automaticky**

**Obrázek 51:** Kód postavy Jídlo

**Obsah obrázku text, snímek obrazovky, žlutá, design

Popis byl vytvořen automaticky**

**Obrázek 52:** Kód pozadí

**Obsah obrázku text, snímek obrazovky, Písmo, logo

Popis byl vytvořen automaticky**

# UMÍSTĚNÍ ÚLOH

Sada úloh bude umístěná na dvou místech. Prvním místem bude webová stránka scratch.mid.edu, což je webová platforma určena pro tvorbu interaktivních příběhů, animací, her a dalších multimediálních projektů. Je vyvinuta a provozována MIT Media Lab. Tato platforma je primárně zaměřena na vzdělávání v oblasti programování a počítačového myšlení pro děti a začátečníky. Úlohy s řešením pro učitele jsou nahrané na mém profilu. Zde je odkaz na sadu úloh - <https://scratch.mit.edu/users/kvetda00/projects/>.

Druhým místem je cloudové online úložiště GitHub, což je webová platforma pro správu verzí a správu kódových projektů. Je to jeden z nejpopulárnějších nástrojů pro správu verzí a poskytuje služby pro spolupráci na vývoji softwaru. Umožňuje vývojářům spolupracovat na projektech, sledovat změny ve zdrojovém kódu, navrhovat úpravy, spravovat problémy a mnoho dalšího. GitHub je postaven na systému správy verzí Git, který umožňuje ukládat historii změn v kódu a zpřístupňuje nástroje pro správu těchto změn. Uživatelé mohou vytvářet repozitáře, které obsahují zdrojový kód jejich projektů a umožňují lidem přispívat kódem, navrhovat změny a diskutovat o vylepšeních či chybách. Úlohy s řešením pro učitele i úlohy upravené pro žáky budou nahrané v repozitáři mého účtu, který je zveřejněn.

Zde je odkaz - <https://github.com/davidkveton/kveton_bc>.

Závěr

Seznam použitých zdrojů

**Knižní:**

**Online:**

**Obrázky:**

1. PRŮCHA, Jan, WALTEROVÁ, Eliška a MAREŠ, Jiří (2013). Pedagogický slovník. 7., aktualiz. a rozš. vyd. Praha: Portál, ISBN 978-80-262-0403-9.m
2. *International Society for Technology in Education* (2012). [online]. [cit. 2022-11-17]. Dostupné z: <http://www.assnstrategies.com/pdf/ISTEPositionProfileFinal.pdf>
3. *Strategie digitálního vzdělávání do roku 2020*. Praha: Ministerstvo školství, mládeže a tělovýchovy České republiky, 2014. Dostupné také z: <https://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-digitalniho-vzdelavani-do-roku-2020>
4. *Informatické myšlení: Co je informatické myšlení* (2018). [online]. Jihočeská univerzita v Českých Budějovicích, [cit. 2022-11-17]. Dostupné z: <https://www.imysleni.cz/informaticke-mysleni/co-je-informaticke-mysleni>
5. SELBY, Cynthia a John WOOLARD. UNIVERSITY OF SOUTHAMPTON. *Computational Thinking: The Developing Definition*. Southampton, 2013. [online]. [cit. 2022-11-17]. Dostupné z: <https://eprints.soton.ac.uk/356481/1/Selby_Woollard_bg_soton_eprints.pdf>
6. VANÍČEK, Jiří, ed. Výuka algoritmizace patří především do informatiky. In: ROSECKÝ, Jan. *Počítač ve škole 2016 – sborník příspěvků*. Nové Město na Moravě: Gymnázium Vincence Makovského se sportovními třídami, Nové Město na Moravě, 2016. ISBN 978- 80-905765-6-8.
7. COHEN, Avi a Bruria HABERMAN*. Computer science: A language of technology*. ACM SIGCSE Bulletin. 2007, 39(4), 65-69.
8. PELÁNEK, Radek. *Jak to vyřešit?: logické úlohy a hry*. 1. vyd. Praha: Portál, 2011. ISBN 978-80-7367-872-2.
9. LESSNER, Dan (2018). *Hledání dárců ledvin* [online]. Jihočeská univerzita v Českých Budějovicích, [cit. 2022-11-17]. Dostupné z: [https://www.imysleni.cz/clanky/priklady/23-hledani-darculevin?fbclid=IwAR3wpkcVVV0JxcB\_up9tCbKz38dAilA3huMyAysUM6pToVQSBuJ P7PKKQCM](https://www.imysleni.cz/clanky/priklady/23-hledani-darculevin?fbclid=IwAR3wpkcVVV0JxcB_up9tCbKz38dAilA3huMyAysUM6pToVQSBuJ%20P7PKKQCM)
10. UNESCO. *Towards knowledge societies for peace and sustainable   
    development, first final recommendations* [online]. 2013 [cit. 2022-11-21]. Dostupné z: <http://bit.ly/1r8eNpC>
11. HYLÉN, Jan. *Open educational resources: Opportunities and challenges. Oecd* [online]. 2006 [cit. 2022-11-21]. Dostupné z: <http://bit.ly/1sFikJX>
12. MŠMT. *Rozpracovaný koncept digitální gramotnosti* [online]. Září 2018, verze 2.0 [cit. 2022-11-21]. Dostupné z: <https://digigram.cz/files/2019/06/VM1.1-Koncept-DG.pdf>
13. Ministerstvo práce a sociálních věcí. *Strategie vzdělávací politiky ČR do roku 2030+* [online]. ©2013-2021, 2020 [cit. 2022-11-21]. Dostupné z: <https://www.msmt.cz/uploads/Brozura_S2030_online_CZ.pdf>
14. BARR, HARRISON a CONERY. *Computational Thinking: Digital Age* [online]. International Society for Technology in Education, 2011 [cit. 2023-11-17]. ISSN-1082-5754. Dostupné z: <https://files.eric.ed.gov/fulltext/EJ918910.pdf>
15. JOST, KETTERL, BUDDE a LEIMBACH. *Graphical Programming Environments for Educational Robots: Open Roberta - Yet Another One?* IEEE, 2014. ISBN 978-1-4799-4311-1. DOI 10.1109/ISM.2014.24. Dostupné z: <https://ieeexplore.ieee.org/document/7033055>
16. KUHAIL, FAROOQ, HAMMAD a BAHJA. *Characterizing Visual Programming Approaches for End-User Developers: A Systematic Review*. IEEE, 2021. ISSN 2169-3536. Dostupné z: <https://ieeexplore.ieee.org/document/9320477>
17. REPENNING. *Moving Beyond Syntax: Lessons from 20 Years of Blocks Programing in AgentSheets* [online]. Journal of Visual Languages and Sentient Systems [cit. 2023-11-20]. Dostupné z: doi:[10.18293/vlss2017-010](http://ksiresearchorg.ipage.com/vlss/journal/VLSS2017/vlss17paper_10.pdf)
18. ZMRZLÝ, Adam. *Vizuální programovací jazyk* [online]. MUNI [cit. 2023-11-20]. Dostupné z: <https://is.muni.cz/th/k10pq/prilohy/Prezentace_LaSAr/LaSArIS_2014.pdf>

Seznam obrázků

[**Obrázek 1:** Ukázka ucelené sady výukových materiálů dle projektu PRIM (2021) 15](#_Toc151115923)

[**Obrázek 2** Kód Bajtík 20](#_Toc151115924)

[**Obrázek 3** Postava Bajtík 20](#_Toc151115925)

[**Obrázek 4** Pozadí 21](#_Toc151115926)

[**Obrázek 5:** Kód Bajtík 23](#_Toc151115927)

[**Obrázek 6:** Kostým Bajtík 23](#_Toc151115928)

[**Obrázek 7:** Kód Tlačítko Tělo 25](#_Toc151115929)

[**Obrázek 8:** Kód Tlačítko hlava 25](#_Toc151115930)

[**Obrázek 9:** Kód Tělo 25](#_Toc151115931)

[**Obrázek 10:** Kód Hlava 25](#_Toc151115932)

[Obrázek 11: Kostýmy Hlava 26](#_Toc151115933)

[**Obrázek 12:** Kostýmy Tělo 26](#_Toc151115934)

[**Obrázek 13:** Kód pro vytvoření Mandaly 28](#_Toc151115935)

[**Obrázek 14:** Výsledná Mandala 28](#_Toc151115936)

[**Obrázek 15:** 6 vytvořených tlačítek, které jsou umístěné na DJ Pultu 30](#_Toc151115937)

[**Obrázek 16:** Kód pro tlačítko 30](#_Toc151115938)

[**Obrázek 17:** Přidané barvičky na paletě 32](#_Toc151115939)

[**Obrázek 18:** Kód pro barvu Modra 32](#_Toc151115940)

[**Obrázek 19:** Kód postavy Tuzka 33](#_Toc151115941)

[**Obrázek 20:** Kód Kostka 35](#_Toc151115942)

[**Obrázek 21:** Kostýmy 35](#_Toc151115943)

[**Obrázek 22:** Kód testu numerické paměti 37](#_Toc151115944)

[**Obrázek 23:** Kód Banány 39](#_Toc151115945)

[**Obrázek 24:** Kód Opice 39](#_Toc151115946)

[**Obrázek 25:** Kód Kočka 41](#_Toc151115947)

[**Obrázek 26:** Kód animace startu 41](#_Toc151115948)

[**Obrázek 27:** Pozadí 42](#_Toc151115949)

[**Obrázek 28:** Kód pro kuličku 44](#_Toc151115950)

[**Obrázek 29:** Ukázka dráhy 44](#_Toc151115951)

[**Obrázek 30:** Kód postavy Bajtík 46](#_Toc151115952)

[**Obrázek 31:** Kód postavy Moucha 47](#_Toc151115953)

[**Obrázek 32**: Kód postavy Bajtík 1 49](#_Toc151115954)

[**Obrázek 33:** Kód postavy Bajtík 2 49](#_Toc151115955)

[**Obrázek 34:** Kód postavy Ball 51](#_Toc151115956)

[**Obrázek 35:** Kód postavy Stickman 51](#_Toc151115957)

[**Obrázek 36:** Kód postavy Blok 53](#_Toc151115958)

[**Obrázek 37:** Kostýmy postavy Blok 53](#_Toc151115959)

[**Obrázek 38:** Kód postavy Bajtík 55](#_Toc151115960)

[**Obrázek 39:** Kód postavy Sloupy 55](#_Toc151115961)

[**Obrázek 40:** Kostým Sloupy 55](#_Toc151115962)

[**Obrázek 41:** Kód postavy Vesmírná loď 57](#_Toc151115963)

[**Obrázek 42:** Kód postavy Laser 58](#_Toc151115964)

[**Obrázek 43:** Kód postavy Asteroid 58](#_Toc151115965)

[**Obrázek 44:** Kód postavy Planeta 59](#_Toc151115966)

[**Obrázek 45:** Kód postavy Konec hry 59](#_Toc151115967)

[**Obrázek 46:** Pohyb postavy Had1 61](#_Toc151115968)

[**Obrázek 47:** Hlavní kód postavy Had1 61](#_Toc151115969)

[**Obrázek 48:** Kód postavy Jídlo 62](#_Toc151115970)

[**Obrázek 49:** Kód pozadí 62](#_Toc151115971)

Seznam grafů

Seznam tabulek

Anotace

|  |  |
| --- | --- |
| **Jméno a příjmení:** |  |
| **Katedra:** | Katedra technické a informační výchovy |
| **Vedoucí práce:** |  |
| **Rok obhajoby:** | 2023 |
|  |  |
| **Název práce:** |  |
| **Název v angličtině:** |  |
| **Anotace práce:** |  |
| **Klíčová slova:** |  |
| **Anotace v angličtině:** |  |
| **Klíčová slova v angličtině:** |  |
| **Přílohy vázané v práci:** | 2 přílohy:  *Dotazník*  *Obrázek základní desky MSI* |
| **Rozsah práce:** | stran |
| **Jazyk práce:** | čeština |