Umíme informatiku

Blokové programování

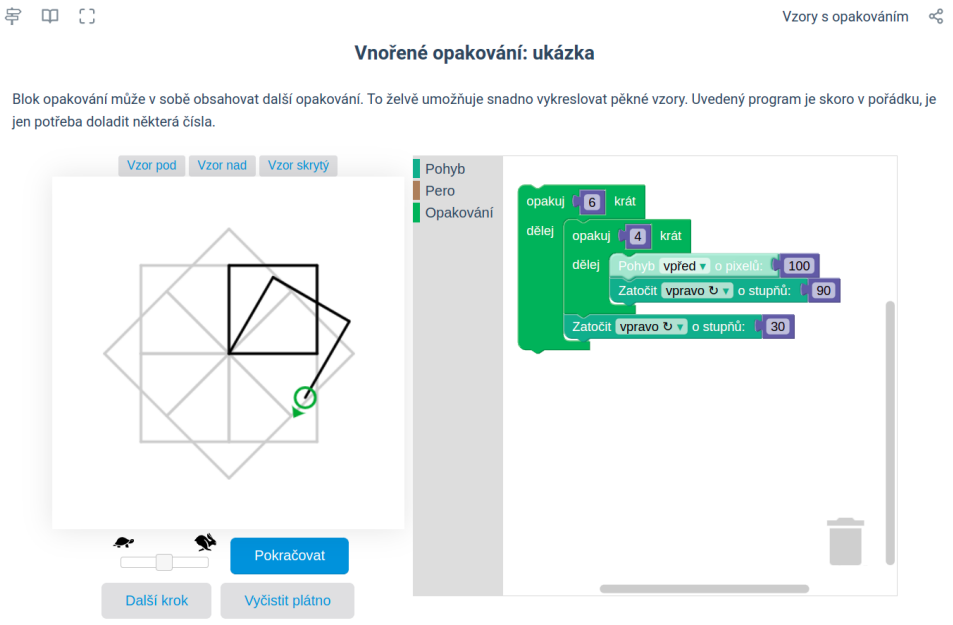
Blokové programování (též grafické programování) je dnes standardní přístup k výuce algoritmického myšlení a základů programování. Používá se jako předstupeň před programováním v běžných textových programovacích jazycích (např. Python). Může být použito také jako nástroj pro procvičení algoritmického myšlení bez toho, aby následně navazoval přechod na textový programovací jazyk.

Blokové programování se využívá především ve spojení s **mikrosvěty**: jednoduché prostředí (pravidelná mřížka, 2D plátno), ve kterém se pohybuje agent ovládaný programem (robot, želva, malíř).

Blokové programování v Umíme informatiku

Umíme informatiku využívá standardní postupy pro znázornění a ovládání blokového programování. Dostupná cvičení tak lze snadno kombinovat s aktivitami v jiných prostředích. Blokové programování v Umíme informatiku má následující specifika a silné stránky:

- **úlohy s jednoznačným zadáním**, které jsou **automaticky opravované** – žáci tak mají konkrétní cíl a jasnou zpětnou vazbu, zda cíle dosáhli,
- **několik** vzájemně se doplňujících **forem procvičování a mikrosvětů**, díky čemuž si žáci mohou každé téma procvičit různými způsoby,
- velmi **rozsáhlá sbírka příkladů** (stovky zadání),
- důraz na práci s obtížností, systematický přístup ke **stupňování obtížnosti** pomocí postupného odebírání nápovědných prvků – žáci přechází od jednoduchého výběru ze dvou možností, přes drobné úpravy programů až ke zcela samostatné tvorbě.



Formy procvičování

Umíme informatiku nabízí několik forem procvičování, které se vzájemně doplňují. Tyto formy procvičování jsou do velké míry nezávislé na konkrétních algoritmických konceptech (posloupnosti příkazů, cykly, podmínky, proměnné); všechny lze využít při procvičování několika konceptů.

Následující tabulka nabízí základní přehled cvičení s jejich stručným popisem a přehled běžných případů užití. Kolonka „typický čas“ udává čas řešení jednoho zadání. U cvičení Rozhodovačka a Přesouvání žáci řeší vždy celou sadu úloh podobného typu, splnění celé sady většinou trvá alespoň 3 minuty. U každého typu cvičení jsou k dispozici desítky nebo i stovky zadání.

	interaktivita	typický čas		rozcvička	cvičení v hodině	domácí práce	bonus
Rozhodovačka	☐	3–15 s	výběr z možností	✓	✓	✓	☐
Přesouvání	✓	20–60 s	doplňování chybějících částí	✓	✓	✓	☐
Želví grafika	✓	1–7 min	kreslení obrázků	☐	✓	✓	✓
ProgMalování	✓	1–7 min	kreslení obrázků	☐	✓	✓	✓
Plošinovka	✓	1–7 min	ovládání postavy ve hře	☐	✓	✓	✓
Stavitel	✓	2–8 min	ovládání postavy ve hře	☐	✓	✓	✓
RoboAréna	✓	5–10 min	souboj robotů	☐	✓	☐	✓
Robotanik	✓	5–10 min	zjednodušené programy	✓	✓	✓	✓
Šipkovaná	✓	0,5–5 min	logická úloha na mřížce	✓	✓	✓	✓

Rozhodovačka a Přesouvání

V těchto cvičeních mají zadání podobu statických obrázků, žáci vybírají odpověď z připravených možností (kliknutím nebo přesouváním). Otázky mají typicky formu *„Co udělá program?“* nebo *„Jaký příkaz musíme doplnit do programu?“*. Pokročilejší otázky se pak ptají na úpravy či shodnost programů.

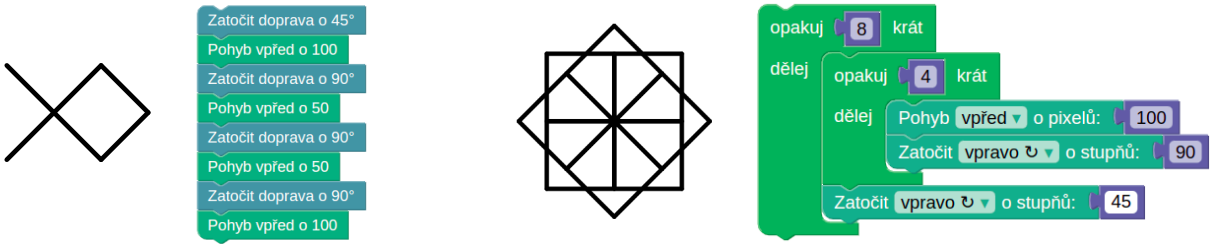
Tato cvičení mají dva základní způsoby využití:

- **rozcvička** před vlastní tvorbou programů,
- **kontrola** znalostí, **ujasnění miskoncepcí** – identifikace konceptů, které žákům stále dělají problémy a které potřebují ujasnit (buď dalším procvičováním v systému nebo diskuzí s učitelem).

Želví grafika

Želví grafika umožňuje vykreslovat **elegantní obrázky** pomocí virtuální želvy ovládané jednoduchými příkazy pro pohyb (posun vpřed, otočení). Želví grafika je jedním z nejtypičtějších přístupů k výuce základů programování a její základy jdou velmi dobře zpracovat pomocí blokového programování. Cílem úlohy je napsat program, který vykreslí zadaný obrázek. Tvorba programu je **interaktivní**, žáci si mohou svoje programy průběžně zkoušet a ladit.

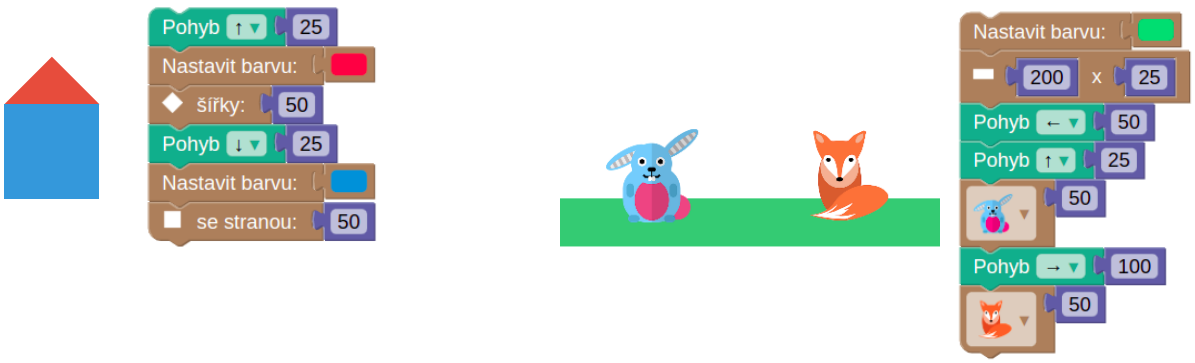
Želví grafika má přirozený **přesah do geometrie**, především do práce s úhly. Základní princip želví grafiky je velmi intuitivní a názorný. Cvičení tak může posloužit jako dobrá pomůcka i pro žáky, kteří úhly ještě důkladně neovládají – v želví grafice lze do velké míry použít k práci s úhly přístup „pokus & omyl“, kterým si žáci postupně budují intuici.



ProgMalování

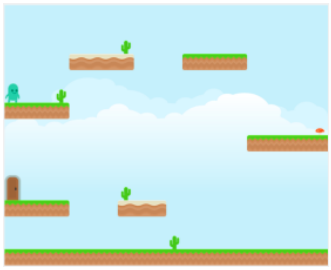
ProgMalování je podobný typ úlohy jako želví grafika. Jde o variaci méně známou a používanou než želví grafika. V této úloze máme malíře, který se pohybuje po plátně a dělá „razítka“ (základní tvary jako čtverec, kruh, obdélník a několik obrázků postav). Pohyb je oproti želví grafice zjednodušený, malíř se pohybuje pouze ve čtyřech směrech, bez použití úhlů. Důraz úlohy je na správné **pořadí akcí** – díky překrývání útvarů lze ze základních tvarů vytvářet složitější obrazce.

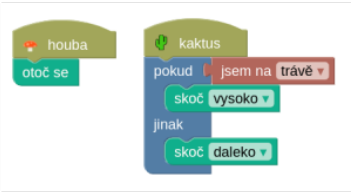
I u této úlohy je přirozený přesah do geometrie, tentokrát především k tématu souřadnic a vlastností základních útvarů.



Plošinovka

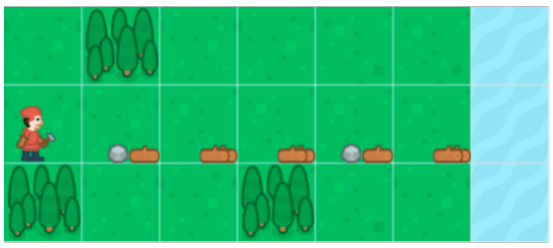
Úloha Plošinovka vychází ze stejnojmenného klasického žánru počítačových her, ve kterém se postava pohybuje pomocí skoků ve světě vertikálně uspořádaných plošin, přičemž má za úkol vyhýbat se překážkám, sbírat mince a dojít k východu. V naší variantě je postava ovládaná programem. Postava se automaticky posunuje vpřed, důraz úlohy je na **podmínky**.

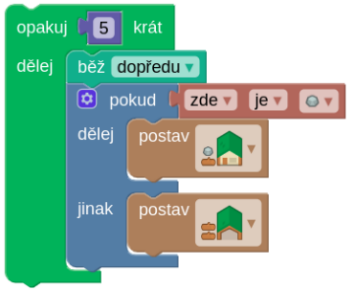




Stavitel

Základ této úlohy tvoří „postava na mřížce“, což je jedno z často využívaných témat pro procvičování blokového programování. Úkolem je napsat program pro stavitele, který se pohybuje po lese, sbírá materiál a staví stavby. Úloha je zaměřena především na procvičení **opakování**, **podmínek** a jejich vzájemné kombinování (cyklus „opakuj dokud“, použití podmínek v rámci cyklu).





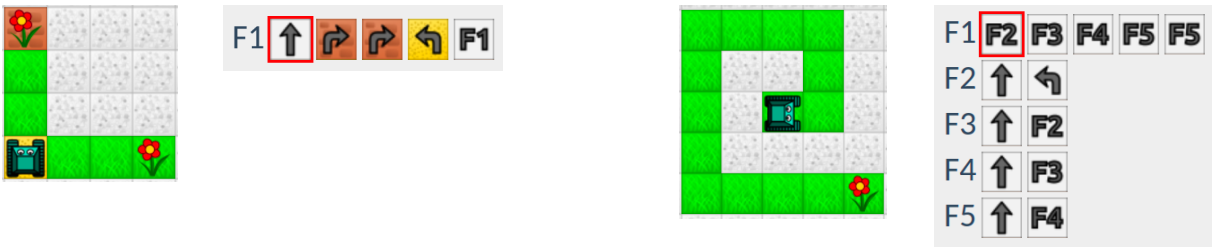
RoboAréna

V této úloze jde také o ovládání postavy na mřížce. Úloha je netypická v tom, že nejde o splnění zadaného úkolu, ale o poražení soupeřících robotů. RoboAréna je tedy vhodná především jako **hra více hráčů** přímo v hodině. Lze ji nicméně použít i pro individuální řešení, kdy hráč hraje proti „umělé inteligenci“ (tu představují předpřipravené strategie, které jsou přiměřeně inteligentní, aby bylo reálné je porazit).

Robotanik

Robotanik nepoužívá standardní blokové programování, ale zjednodušenou verzi, ve které se program tvoří jako **sekvence šipek**, přičemž pomocí podbarvení lze vyjádřit podmínky. Hlavní důraz je u této úlohy na **dekompozici programu** na dílčí funkce. Funkce se mohou vzájemně volat, čímž lze simulovat cykly a realizovat i složitou funkcionalitu pomocí rekurze.

Základní princip úlohy je v různých podobách hojně využívaný. Známá je například hra LightBot, další variace lze najít např. v rámci Hour of Code. Zpracování na Umíme informatiku klade důraz na dostatečně bohatou sbírku příkladů odstupňovanou podle obtížnosti. V úvodní zadáních se tvoří jednoduché sekvence kroků, které jsou zvládnutelné i pro mladší žáky. Poslední úlohy pak představují náročný oříšek i pro talentované programátory.



Šipkovaná

Šipkovaná je jednoduchá logická úloha, která však procvičuje programátorský styl myšlení a může posloužit jako dobrá rozcvička před blokovým programováním. Podobně jako v Robotanikovi i zde se program tvoří pomocí šipek, tentokrát se však **šipky umisťují přímo do herního plánu**. Díky tomu je význam příkazů a tvorba programu velmi intuitivní. Hra je použitelná i pro děti, které ještě ani neumí číst. Hra nicméně obsahuje i náročnější prvky (souběžnost, změna stavové informace), takže nabízí zajímavé úlohy i pro starší řešitele.

Koncepty algoritmického myšlení

Základní pojetí výuky je vhodné organizovat podle konceptů algoritmického myšlení a k nim vybírat dílčí cvičení. Žádné cvičení není úplně „univerzální“, různé formy se vzájemně doplňují a je výhodné je kombinovat. Následující tabulky udává zjednodušeně základní přehled, na co jsou jednotlivá cvičení vhodná.

	Rozhodovačka	Přesouvání	Želví grafika	ProgMalování	Plošinovka	Stavitel	RoboAréna	Robotanik	Šipkovaná
Význam a pořadí příkazů	✓	✓	✓	✓	✓	✓	✓	✓	✓
Opakování	✓	✓	✓	✓		✓	✓	✓	
Podmínky	✓	✓			✓	✓	✓	✓	✓
Proměnné	✓		✓	✓	✓				
Funkce, rozklad na podproblémy	✓		✓	✓		✓		✓	
Ladění programů			✓	✓	✓	✓	✓	✓	✓

Význam a pořadí příkazů

Úplné základy tvorby programů představují elementární příkazy a jejich řazení. Zde dobře poslouží jako rozcvička Šipkovaná a Rozhodovačka. Důležitým aspektem k procvičení je význam pořadí akcí, ten pěkně ilustruje zejména ProgMalování.

`https://www.umimeinformatiku.cz/cviceni-alg-mysleni-vyznam-prikazu`
`https://www.umimeinformatiku.cz/cviceni-posloupnost-akci`

Opakování, cykly

Opakování a cykly tvoří **těžiště procvičování pomocí blokového programování**. Cykly lze pomocí bloků velmi snadno a přirozeně vyjádřit. Současně jde o téma, které již není úplně jednoduché a má tedy smysl jej s žáky důkladně procvičovat. Typické problematické partie:

- rozpoznání a zápis vzorů – abychom využili cyklus, musíme najít vzor, který se pravidelně opakuje a umět jej zapsat,
- zápis a interpretace opakování – především aspekt „co je součást cyklu, co patří mimo cyklus“,
- vnořené cykly – zejména pomocí želví grafiky lze pomocí vnořených cyklů vytvářet elegantní obrazce, základní použití vnořených cyklů je zde intuitivní, ale jejich důkladné ovládnutí vyžaduje trénink.

Rozhodovačka se zaměřuje cíleně na ujasnění častých miskoncepcí, např. ohledně úrovní zanoření.

`https://www.umimeinformatiku.cz/cviceni-algoritmicke-mysleni-opakovani`

Podmínky

Na podmínky je speciálně orientované cvičení Plošinovka. Z dalších interaktivních úloh se podmínky vyskytují v RoboAréně, Robotanikovi a Šipkované. Základní procvičení pak nabízí Rozhodovačka a Přesouvání.

https://www.umimeinformatiku.cz/cviceni-algoritmicke-mysleni-podminky

Proměnné

Pomocí základních cvičení s výběrem možností je k dispozici procvičování sloužící k ujasnění základního mentálního modelu proměnné. U interaktivních úloh se použití proměnných nachází na hraně toho, co je rozumné zpracovávat formou blokového programování – složitější manipulace s proměnnými je formou bloků mírně nepraktická. V našich příkladech se soustředíme na základní použití proměnných pomocí několik elementárních příkazů (`nastavit hodnotu`, `zvýšit hodnotu`). Ve spojení s kreslením se i pomocí těchto základních příkazů dají tvořit zajímavé obrázky.

https://www.umimeinformatiku.cz/cviceni-algoritmicke-mysleni-promenne

Funkce, rozklad na podproblémy

Podobně jako u proměnných i u tématu funkcí není blokové programování pro zpracování úplně ideální – tvorba a editace programů s funkcemi pomocí bloků je občas nepřehledná. Proto se u tohoto tématu soustředíme především na to, aby žáci pracovali s předpřipravenými materiály.

- Rozhodovačka – uvažování o připravených funkcích, doplňování částí kódu.
- ProgMalování – předpřipravené funkce z bloků, které žáci dále upravují či využívají ve svých programech (nemusí řešit samotnou definici funkce pomocí bloků).
- Želví grafika – silný aspekt „rozklad na podproblémy“, který je však typicky zapisován pomocí vnořených cyklů (bez funkcí).
- Robotanik – cvičení je založeno na využití funkcí; ty jsou sice hodně jednoduché, cvičení však dobře procvičuje princip „rozklad na podproblémy“.

https://www.umimeinformatiku.cz/cviceni-algoritmicke-mysleni-funkce
https://www.umimeinformatiku.cz/cviceni-rozklad-programu

Ladění programů

Ladění programů a testování jsou procvičována samovolně při použití libovolného z interaktivních cvičení: žáci při tvorbě programů téměř vždy potřebují hledat dílčí chyby a opravovat je. Pro cílenější procvičení těchto dovedností máme připravené sady zadání, které jsou specificky cílené na ladění. V těchto úlohách dostanou žáci jako součást zadání i program, který zadanou úlohu téměř řeší. V programu jsou však dílčí drobné chyby, které je nutno nalézt a opravit.

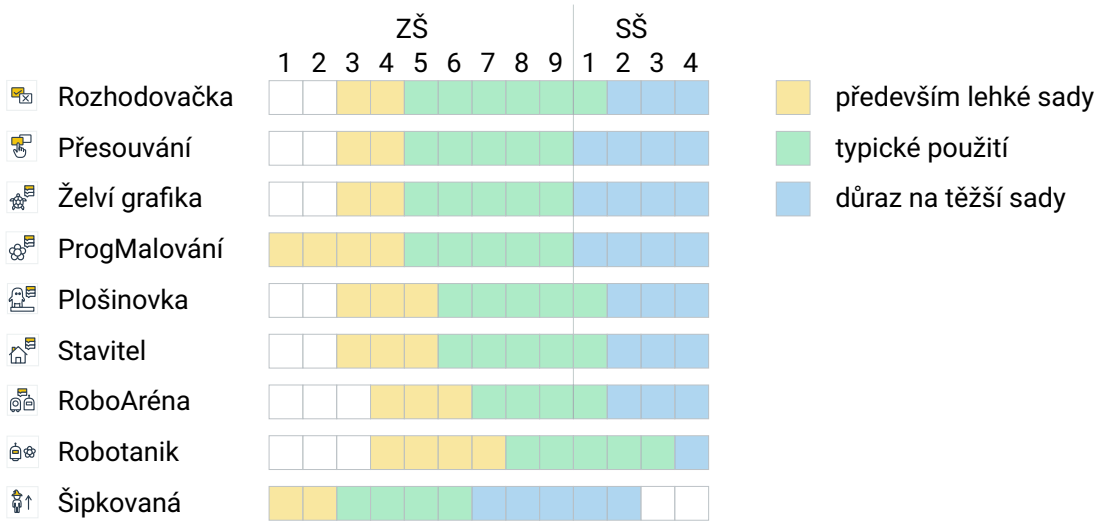
Mnoho žáků k ladění přistupuje metodou „pokus-omyl“ (to se ostatně týká nejen žáků, v programátorské hantýrce se toto chování označuje jako *shotgun debugging*). Je tedy vhodné aktivně vést žáky k alespoň trochu systematickému postupu, např. k využití krokování a k explicitnímu pojmenování zdrojů chyb.

https://www.umimeinformatiku.cz/cviceni-algoritmicke-mysleni-hledani-chyb

Doporučení podle věku

Následující tabulka udává orientační přehled vhodnosti jednotlivých cvičení podle ročníků studia. Blokové programování je ideální především pro 2. stupeň ZŠ. Pro žáky na 1. stupni lze doporučit především cvičení Šipkovaná; i u dalších cvičení jsou však úvodní sady pro mladší žáky většinou dobře řešitelné.

Blokové programování je vhodné i pro žáky středních škol, poslouží například jako dobrá rozcvička před programování v Pythonu nebo před řešením otevřených úloh s roboty. Všechna cvičení navíc obsahují i těžší zadání, u kterých se i zkušený programátor musí zamyslet. Takže i středoškolští žáci se rozhodně při řešení nudit nebudou.



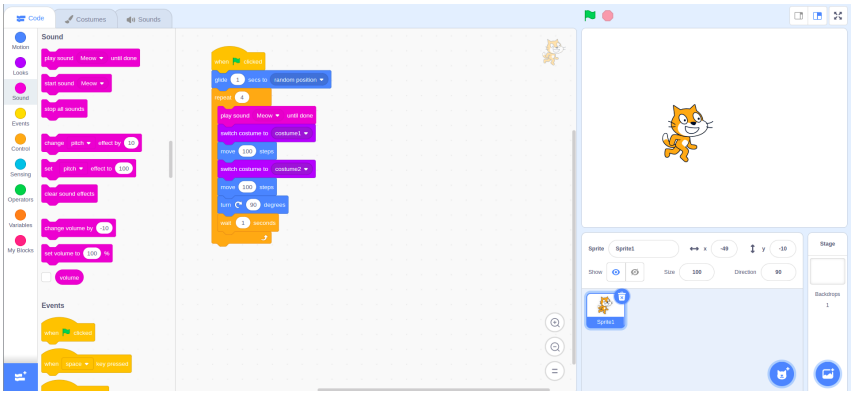
Návaznosti

Systém Umíme informatiku je vhodný pro úvodní seznámení s algoritmickým myšlením a blokovým programováním. K dispozici je dostatek velmi jednoduchých úloh, které jsou určeny úplným začátečníkům. I u složitějších konceptů jsou k dispozici úkoly s jasným cílem, který je současně pro žáky dosažitelný. Takto si mohou žáci vybudovat dobré základy, na které je pak možné následně navázat dalšími aktivitami, které dají žákům prostor řešit více otevřené a kreativní projekty.

Možnost, aby si žáci sami vymýšleli zadání je zajímavá a užitečná: vlastní projekty mohou být motivující, žáci při nich trénují další užitečné dovednosti (např. formulování zadání, organizování práce, vyhodnocování postupu). Vlastní otevřená zadání ale přináší i rizika: u programátorských úloh není snadné najít zadání adekvátní obtížnosti. Pokud si žáci sami vymyslí úkol, snadno se stane, že bude příliš ambiciózní a nepodaří se jim naplnit, což může vést k demotivaci. Přechod k otevřeným projektům je tedy dobré neuspěchat. Průprava na zadání s jasným zadáním žákům pomůže získat odhad toho, co je v dosahu jejich schopností.

Scratch

Mezi nejznámější prostředí pro blokové programování patří Scratch (a jeho zjednodušená verze ScratchJr použitelná i s malými dětmi). Scratch nabízí velmi širokou škálu příkazů a prostředí. Jeho silnou stránkou je především tvorba interaktivních animací, příběhů a jednoduchých her.



https://scratch.mit.edu/

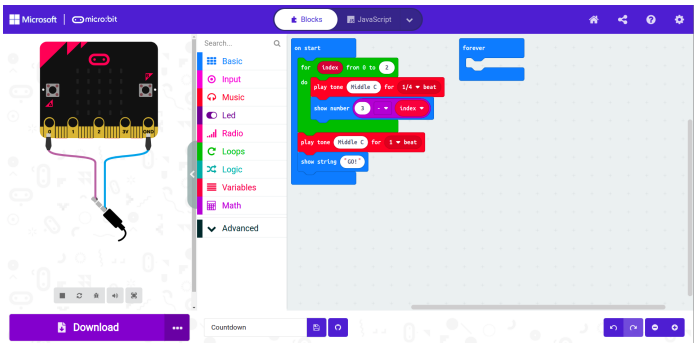
Robotika

Blokové programování lze využít i k programování fyzických robotů. Po koncepční stránce je programování velmi podobné programování virtuálních agentů v mikrosvětech (jak je použito ve cvičeních v Umíme informatiku) a nepřináší žádné zásadní nové prvky. Interakce s fyzickými roboty je však pro mnoho žáků motivující a slouží jako dobrý doplněk cvičení s virtuálními roboty.

Ozobot

https://ozobot.com/create/ozoblockly

Micro:bit



https://makecode.microbit.org/

Umíme informatiku
https://www.umimeinformatiku.cz/

Autorský kolektiv:

- Metodika a tvorba zadání příkladů na blokové programování: doc. Mgr. Radek Pelánek, Ph.D.; RNDr. Tomáš Effenberger, Ph.D.
- Návrh a realizace cvičení: Mgr. Petr Jarušek, Ph.D.; Mgr. Tomáš Vejpustek

Metodické postupy použité při přípravě zadání na blokové programování jsou zdokumentované v následujících odborných publikacích:

- Pelánek, Radek, and Tomáš Effenberger. Design and analysis of microworlds and puzzles for block-based programming. *Computer Science Education* (2022).
- Effenberger, Tomáš, Jaroslav Čechák, and Radek Pelánek. Measuring difficulty of introductory programming tasks. *Proceedings of ACM Conference on Learning@Scale* (2019).