

PROGRAMOVÁNÍ VE SCRATCH

pro 2. stupeň základní školy

Jiří Vaníček
Ingrid Nagyová
Monika Tomcsányiová



EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání



MINISTERSTVO ŠKOLSTVÍ,
MLÁDEŽE A TĚLOVÝCHOVY

Poděkování

Eliška Abrahámová, Adam Burdák, Radka Duží, Markéta Brázdová, Eva Ehlerová, Jiří Havela, Drahomíra Hanzlíková, Pavlína Horáčková, Karel Janda, Jan Konvalina, Radovan Mikeš, Lenka Papežová, Ondřej Sekera, Václav Vávra, Olga Vilímová.

Schválilo MŠMT č. j.: MSMT-11717/2021-1 dne 28. 4. 2021 k zařazení do seznamu učebnic pro základní vzdělávání jako součást ucelené řady učebnic pro vzdělávací obor Informatika s dobou platnosti šest let.



Jihočeská univerzita
v Českých Budějovicích
University of South Bohemia
in České Budějovice

PROGRAMOVÁNÍ VE SCRATCH pro 2. stupeň základní školy

doc. PaedDr. Jiří Vaníček, Ph.D.; RNDr. Ingrid Nagyová, PhD.;
doc. PaedDr. Monika Tomcsányiová, PhD.

Recenzent:

doc. RNDr. Ľubomír Šnajder, PhD.

Vydavatel:

Jihočeská univerzita v Českých Budějovicích, Pedagogická fakulta

Obálka:

Mgr. Pavel Pfauser

Rok vydání: 2020



Podléhá licenci Creative Commons
Uveďte původ - Zachovejte licenci 4.0



9 788073 947835 >

CO NAJDETE V TOMTO DOKUMENTU

- základní informace o učebnici
- obsah a harmonogram výuky
- popis navigace v učebnici
- jak učit podle těchto materiálů
- jak se na výuku připravit


Učitelé, přečtěte si prosím tento dokument pečlivě. Usnadní Vám orientaci v učebnici, dostanete návod, jak instalovat software, jak učebnici používat, jak vést výuku.

ZÁKLADNÍ INFORMACE O VZDĚLÁVACÍM OBSAHU

- Učebnice je připravena pro věk žáka 7. ročníku ZŠ (sekunda osmiletého gymnázia). Je možné je použít i v sousedních ročnících nebo látku do dvou ročníků rozdělit.
- Učebnice se skládá z materiálů pro žáka a z materiálů pro učitele:
 - **žákovské listy** (zadání úloh k promítnutí na projekci),
 - **pracovní soubory** s projekty (žáci si je spouští v počítači a pracují s nimi),
 - **metodická příručka** pro učitele.
- Rozsah učebnice je 32 vyučovacích hodin. Zaplní tedy celý rok (nebo dvě pololetí následujících ročníků). S rozšiřujícími aktivitami a rozvolněním tempa může pokrýt i více než 40 vyučovacích hodin.
- Materiály jsou členěny na kapitoly. Výuka je rozložena do deseti tematických celků. Každý tematický celek je věnován jednomu základnímu konceptu (viz názvy kapitol na stranách 3 – 4).
- Každá kapitola má základní a rozšiřující učivo.
- K učebnici jsou přidány vzorové ukázky testů programátorských dovedností.

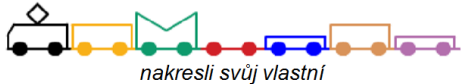
Poznámka: Máte-li možnost výuky informatiky ve dvou hodinách týdně, nedoporučujeme tzv. „dvouhodinovky“. Programování je náročná činnost a děti se po jedné hodině unaví. Fyziku také neučíme ve dvouhodinovkách. Jinou možností je jednu hodinu programovat a ve druhé hodině se věnovat jiné činnosti.

ŽÁKOVSKÉ LISTY



Barvy vagónů

1. Sestav scénář pro vlak, v němž bude lokomotiva zelená a vagóny oranžové.
2. Sestav scénář pro vlak, v němž každý vagón bude mít jinou barvu.



3. Smaž scénu a rychle vykresli pět úplně stejných barevných vlaků z předchozí úlohy.

Snímky prezentace se zadáními úloh pro žáky. Jeden list obsahuje dvě až tři úlohy pro práci zhruba na 3 – 8 minut.

Žákovské listy jsou určeny k projekci před třídou.

Učitel může rychlejšími žákům tyto listy zpřístupnit, žák si potom zadání promítá na svém monitoru. Listy není třeba tisknout.

Ikony v záhlaví snímků vysvětlíme na str. 8 - 10.

METODICKÁ PŘÍRUČKA

PRIM KAPITOLA 1 – SESTAVUJEME SCÉNÁŘE

snímek 7

Kreslíme podle scénáře

1. Jaký vlak vznikne podle tohoto scénáře? Nejřív nakresli na papír, jak bude vlak vypadat, a pak ověř.
2. Napiš svému sousedovi na papír scénář, k němuž bude mít za úkol nakreslit vlak tužkou. Zkontroluj, jestli vlak nakreslil správně. Pokud se neshodnete, scénář sestavte a v počítači ověřte.

METODICKÉ POZNÁMKY

1. Žáci se učí číst scénář, porozumět tomu, co scénář vykoná. Trénují si schopnost předvídat činnost počítače. Čtení scénáře je jedna z klíčových dovedností programování. Jde o aktivitu bez počítače – žáci pracují s promítaným zadáním úlohy a s papírem. Ověřování může probíhat společně u projekce. Žáci musí přijmout, že v informatice nestráví celou dobu výuky přímou práci s počítačem.
2. Aktivita pro dvojice žáků, kdy si žáci své zadání vymění a na konci úlohy také navzájem zkontrolují.

ŘEŠENÍ ÚLOH

1. Správné řešení:
2. Zde není „správné řešení“. Dvojice, která vymyslí originální scénář, může své řešení představit celé třídě.

MOŽNÉ POTÍŽE A JEJICH ŘEŠENÍ

- Pokud učitel povolí žákům, aby si sami ověřovali ve svém počítači, hrozí **riziko**, že žáci sklouznou k hraní si bez přemýšlení. V 1. úloze je třeba důsledně vyžadovat, aby žáci v úloze počítač nepoužívali.
- Ověřování na počítači ve 2. úloze by měl učitel výslovně povolit pouze na žádost jednotlivých dvojic žáků (pokud se spolu na správné odpovědi neshodnou).

Ke každému snímku žákovských listů existuje jedna nebo dvě stránky v metodice.

Na začátku takové stránky (viz obr.) je zobrazen **snímek žákovského listu** – učitel tak k přípravě stačí otevřít pouze metodiku, nepotřebuje otevírat žákovské listy.

Následují **metodické poznámky** ke každé úloze: co je cílem, na co učitel má dát důraz, jak kontrolovat práci žáka, jak organizovat výuku, jaké otázky pokládat. Někdy jsou vysvětlovány nové pojmy.

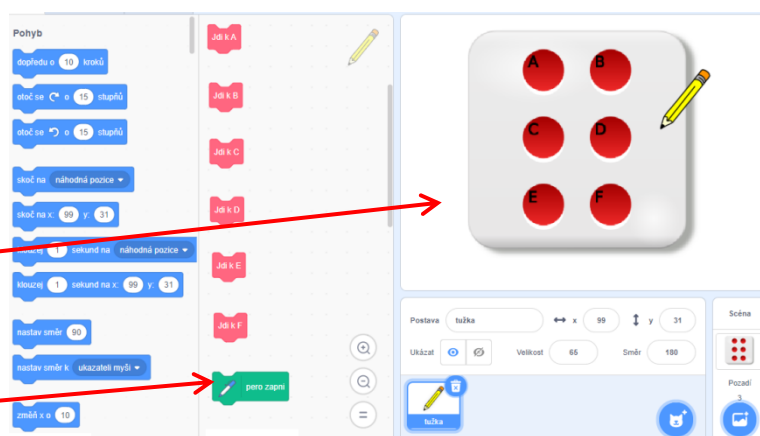
Ke každé úloze se zobrazuje **správné řešení** (správný scénář, vytvořený obrázek, správná odpověď na otázky).

Pokud je to potřeba, je stránka zakončena informacemi o **možných potížích**, které se mohou znenadání vyskytnout, a **radami**, jak v nastalé situaci reagovat a potíže odstranit.

PROGRAMOVACÍ PROJEKTY

Pod slovem projekty jsou ve Scratchi chápána různá prostředí, v nichž jsou předem připravené postavy a bloky. Žáci si je spustí a mohou hned začít programovat.

Na obrázku vidíte otevřený projekt. Na **scéně** je postava tužka, která se bude pohybovat na pozadí hrací kostky. Uprostřed plochy jsou předem připravené vybrané **bloky** pro práci žáků.



Projekty si žáci spouští na internetu ze **studia iMyšlení**. (viz str. 5). Upravené projekty se přihlášenému žákovi budou automaticky ukládat do cloudu, podobně jako např. Google dokumenty. Žák je bude moci opětovně spouštět např. i z domácího počítače.

Při **offline** použití Scratche budou žáci projekty otevírat jako pracovní soubory ze školního síťového disku (viz str. 5) a upravené budou ukládat do souboru, podobně jako např. obrázky.

Při **online** použití se stažená složka s pracovními soubory nevyužije.

ČASOVÝ PLÁN A SLED AKTIVIT

Každá kapitola je sledem několika aktivit na jedno téma. Každou aktivitu představuje v časovém plánu jedna kartička.

Celá aktivita probíhá v jednom tzv. projektu (pracovním souboru), který žáci otevřou a v němž pracují. Ikona pod názvem aktivity na kartičce symbolizuje použitý projekt ze studia iMyšlení. Ikona kočky znamená začít nový, prázdný projekt (*Soubor/Nový*).

Každá aktivita se skládá z několika úloh. Zadání těchto úloh najdete v žákovských listech na snímcích uvedených pod ikonou projektu (nebo též v metodice).


Celá učebnice představuje 32 hodin výuky.

Doporučujeme **dodržet předpokládanou dobu trvání** kapitol především na začátku učebnice.


1 – ÚVOD DO PROGRAMOVÁNÍ, SESTAVENÍ SCÉNÁŘE

3 HODINY


Stavíme vlak


snímky 4 – 14
45 minut


Kreslíme číslice


snímky 15 – 20
25 minut

Scratch - registrace


snímek 21
20 minut


Tiskneme písmena


snímky 22 – 27
45 minut


2 – OPAKOVÁNÍ BLOKŮ

3 HODINY


Stejně vagóny


snímky 4 – 9
45 minut

Kreslíme obrazce


snímky 10 – 14
45 minut


Písmena a slova


snímky 15 – 20
45 minut


3 – VLASTNÍ BLOKY

5 HODIN

Navigujeme cestu


snímky 3 – 7
45 minut


Kreslíme ornament


snímky 8 – 11
45 minut


Vlastní vagóny


snímky 12 – 14
45 minut

Kreslíme sluníčko


snímky 15 – 21
45 minut


Domek


snímky 22 – 27
45 minut


4 – OPAKOVÁNÍ S PODMÍNKOU

2 HODINY

Chytáme písmenka


snímky 3 – 6
20 minut


Tancujeme


snímky 7 – 10
25 minut

Stavíme mosty


snímky 11 – 12
15 minut


Balónek na poušti


snímky 13 – 19
30 minut


5 – MYŠ A KLÁVESNICE

3 HODINY


Kočíí procházka


snímky 3 – 6
25 minut

Dva tanečníci


snímky 7 – 11
35 minut

Otázky a odpovědi


snímky 12 – 14
30 minut

Akvárium


snímky 15 – 19
45 minut

6 – POSÍLÁNÍ ZPRÁV

3 HODINY

Žádost o tanec



snímky 3 – 7
20 minut

Čarujeme



snímky 8 – 9
25 minut

Oblékáme Edu



snímky 10 – 12
25 minut

Brýle a čepice



snímky 13 – 16
(rozšiřující aktivita)

Kreslený vtíp



snímky 17 – 20
45 minut

Setkání



snímky 21 – 22
20 minut

7 – ROZHODOVÁNÍ

3 HODINY

V bludišti



snímky 3 – 5
20 minut

Přistání na Marsu



snímky 6 – 10
45 minut

Moucha v láhvi



snímky 11 – 14
25 minut

Kvízová otázka



snímky 15 – 19
45 minut

8 – SOUŘADNICE

2 HODINY

Chtáme balónek



snímky 3 – 4
20 minut

Kreslicí aplikace



snímky 5 – 7
25 minut

Diskutujeme o souřadnicích



snímky 8 – 11
15 minut

Náhodná procházka



snímky 12 – 17
30 minut

Dvě podmínky



snímky 18 – 21
(rozšiřující aktivita)

9 – PARAMETRY

4 HODINY

Animace



snímky 3 – 4
20 minut

Obrazce, parametry



snímky 5 – 14
60 minut

Slova z klávesnice



snímky 15 – 21
30 minut

Zuby na pile



snímek 22
25 minut

Domek s parametry



snímky 23 – 27
45 minut

Vědomostní kvíz



snímek 28
(rozšiřující aktivita)

10 – PROMĚNNÉ

4 HODINY

Chtáme jablka



snímky 3 – 4
20 minut

Počítáme do 100



snímky 5 – 8
25 minut

Měníme rychlost



snímky 9 – 10
20 minut

Myslím si číslo



snímky 11 – 13
25 minut

Hra v kostky



snímky 14 – 16
30 minut

Hra Žralok



snímky 17 – 23
60 minut

Autodráha



snímky 24 – 27
(rozšiřující aktivita)

VZDĚLÁVACÍ CÍLE A RVP

Vzdělávací cíle, stanovené pro tuto učebnici, vycházejí z revidovaného RVP pro vzdělávací oblast informatika a ICT podle jejich verze z r. 2020. Tato verze RVP je zveřejněna na webu Národního ústavu pro vzdělávání na <http://www.nuv.cz/t/revize-rvp-ict> pod odkazem *Informatika – rámec očekávaných výstupů*.

Pokud žák zvládne základní učivo, získá níže vyjmenované kompetence z revidovaného RVP pro informatiku na 2. stupni ZŠ. V tabulce uvádíme, které výstupy učebnice naplňuje.

Tematický okruh Algoritmizace a programování
Očekávaný výstup
po přečtení jednotlivých kroků algoritmu nebo programu vysvětlí celý postup; určí problém, který je daným algoritmem řešen
rozdělí problém na jednotlivě řešitelné části a navrhne a popíše kroky k jejich řešení
upraví daný algoritmus pro jiné problémy, ověří správnost postupu navrženého i někým jiným, najde a opraví v něm případnou chybu
navrhne různé algoritmy pro řešení problému; vybere z více možností vhodný algoritmus pro řešení problém a svůj výběr zdůvodní *)
v blokově orientovaném programovacím jazyce sestaví přehledný program pro vyřešení zadaného problému; program otestuje a opraví v něm případné běhové a logické chyby
používá opakování, větvení programu, proměnné, podprogramy s parametry; používá události k paralelnímu spouštění podprogramů

*) K naplnění tohoto výstupu musí přispět ještě jiné úlohy, než pouze programovací (např. algoritmické úlohy ze soutěže Bobřík informatiky <http://www.ibobr.cz>), jinak hrozí riziko, že bude algoritmizace redukována pouze na programování.

NÁVAZNOST NA DALŠÍ UČEBNICE PROGRAMOVÁNÍ PRO ZŠ

Učebnice je připravena tak, aby se s ní programování učilo od úplného začátku. Pokud třída absolvuje **v 5. ročníku** učebnici [Kalaš, Miková: Základy programování ve Scratch](#), lze **první tři kapitoly využít k opakování**. Potom lze vypustit některé z aktivit, které ukazují nový koncept v dalším prostředí, např. v kap. 1 Tiskneme písmenka, v kap. 2 Písmena a slova, v kap. 3 Navigujeme cestu, Kreslíme sluníčko, v kap. 4 Chytáme písmenka.

Na tuto učebnici navazuje učebnice [Černochová, Vaňková, Štípek: Programování ve Scratch pro pokročilé](#) pro 8. či 9. ročník ZŠ. Podle ní žáci vytváří celé programovací projekty na základě dovedností získaných v této učebnici.

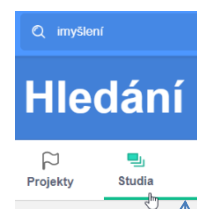
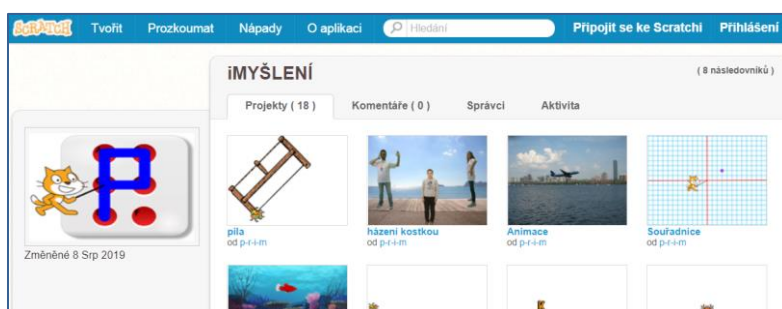
ZÁKLADNÍ INFORMACE O POUŽITÉM SOFTWARE

V učebnici používáme verzi Scratch 3.0 (platnou od ledna 2019). Tato verze je k dispozici online (s internetovým připojením) i offline (pro práci bez připojení nebo se špatným připojením k internetu).

PRÁCE S ONLINE VERZÍ SCRATCH

Scratch spustíte na adrese <https://scratch.mit.edu>. Potřebujete webový prohlížeč (z běžných prohlížečů Scratch nefunguje v Internet Exploreru, funguje v Chrome, Firefoxu, Edge). Pokud zde Scratch nefunguje, zkontrolujte, jestli nemáte starou verzi prohlížeče.

Žáci pracují s pracovními soubory (programovacími projekty) umístěnými na internetu v tzv. **studiu iMyšlení** (<https://scratch.mit.edu/studios/5715013/>). Adresu studia si žáci uloží do oblíbených.



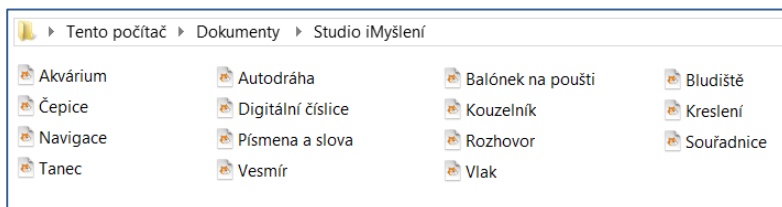
obr. – Studio iMyšlení

Studio iMyšlení lze ve Scratchi vyhledat - v nabídce do pole **Hledání** vložíme text *imyšlení* (s diakritikou) a mezi vyhledanými přepneme na Studia.

PRÁCE S OFFLINE VERZÍ SCRATCH


Tato verze se musí stáhnout a nainstalovat na žákovských počítačích z adresy <https://scratch.mit.edu/download>. V počítačové učebně svěřte tuto činnost jejímu správci.

Pracovní soubory studia iMyšlení najdete ve staženém kompletu učebnice ve složce *Pracovní soubory*. Složku jednoduše zkopírujte do umístění, odkud budou žáci soubory otevírat, ale do níž nemají právo soubory ukládat, protože by původní zadání přepisovali. Žáci budou projekty otevírat v menu *Soubor / Načti z tvého počítače*.



obr. – složka Studio iMyšlení pro práci bez připojení na internet

NASTAVENÍ ČESKÉHO JAZYKA

Pokud se Scratch sám nespustí v češtině, klikněte v nabídce na první položku zleva (Create, Entwickeln apod.) a na další stránce klikněte v nabídce na ikonu globusu . Vyberte si pak jazyk, ve kterém chcete pracovat.

TERMINOLOGIE

Ve Scratch používáme odlišnou terminologii, než v profesionálních programovacích prostředích. Protože žáci snadno pochopí programování jako ovládání postav, záměrně připodobňuje programování k režírování divadelní hry. Scéna má pozadí, tedy jakési kulisy, postavy mají kostýmy. Žák – programátor je režisérem, přiděluje jednotlivým postavám scénáře, které budou postavy na scéně realizovat.

Tuto terminologii zavádíme proto, že další nové pojmy (program, objekt, příkaz atd.), pod nimiž žák nemá konkrétní představu, by jej v úvodu do programování zbytečně zatěžovaly (stejně jako třeba syntaxe v textově orientovaných jazycích, kterou Scratch svým blokovým uspořádáním marginalizuje).

Tuto terminologii také lépe přijímají dívky.

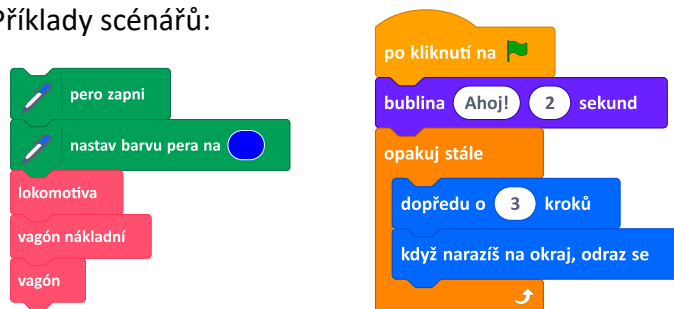


Základem prostředí Scratch je **scéna**, na které se nachází **postava** (později i více postav). Činnost postavy se řídí **scénářem**, který se skládá z **bloků**.

Scénáře se skládají z bloků různých barev, bloky podobné povahy mají stejnou barvu. Bloky nachází v prostředním sloupci a jsou uspořádány pod barevnými záložkami – barva bloku odpovídá barvě záložky. Bloky lze přetahovat do **plochy pro scénáře** a tam scénáře sestavovat.

Porovnání terminologie Scratch se standardními programovacími prostředími:

Příklady scénářů:



blok = příkaz

scénář = program, skript

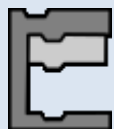
postava = objekt

vlastní blok = procedura

METODY PRÁCE - DRUHY ŽÁKOVSKÝCH AKTIVIT

Žákovské aktivity jsme rozdělili do čtyř druhů. V žákovských listech jsme každý snímek označili ikonou, na jednom snímku jsou vesměs úlohy stejného druhu. Nyní je vysvětlíme.

Doporučujeme, abyste si současně při čtení následující dvojstránky otevřeli žákovské listy pro kapitolu 1 a pro lepší názornost si vybrali konkrétní úlohu, označenou takovou ikonou.



SAMOSTATNÁ PRÁCE - PROGRAMOVÁNÍ

V těchto úlohách mají žáci **něco naprogramovat, vytvářet scénáře**. Pracují ve dvojici nebo sami individuálně, svým tempem. Čtou zadání z žákovských listů a snaží se je splnit. Úlohy na sebe často navazují, žáci by měli předchozí úlohu vždy dodělat, aby mohli pokračovat.

Jsou-li žáci trvale úspěšní, mohou po dohodě s učitelem přecházet na další snímky samostatně, pracovat třeba o dva snímky napřed před ostatními. K tomu slouží úlohy pro pokročilé (viz dále). **Není žádný důvod, aby bystří žáci museli pracovat stejným tempem** jako ti nejpomalejší.

Učitel by měl během samostatné práce procházet třídu a především **kontrolovat**, jestli žáci úkoly plní. Může si nechat některý scénář (programový kód) spustit znovu, nechat si vysvětlit, jak žáci k řešení došli. **Toto řeší individuálně** s danou skupinou nebo žákem, nikoliv před celou třídou. Učitel také pomáhá a radí žákům, kteří se ocitli v nesnázích (podle naší zkušenosti často stačí, aby si znovu přečetli zadání, třeba vícekrát nebo pod dohledem).

Učitel by měl požadovat, aby každý vytvořený scénář žáci ponechali na ploše, aby si jej mohl kdykoliv později prohlédnout.



OBJEVOVÁNÍ – ZKOUMÁNÍ

Aktivity pro celou třídu, **pracuje se společně**. Žáci např. u svých počítačů něco zkoušejí, experimentují, pak o tom referují, **povídají si před celou třídou**. Všichni pak poslouchají, co kdo říká, na co přišel, přidávají své zkušenosti.

Odpovědi v těchto aktivitách jsou vesměs jednoduché (co kdo vidí, jak mu to funguje) a je vhodné **dotazovat se slabších žáků**.

Tyto aktivity se používají v úvodu tématu, při otevření nového projektu nebo při zavádění nového pojmu či termínu. Oproti samostatné práci zde naopak **nesmí žáci pracovat napřed**. Učitel celé zkoumání řídí – uděluje pokyny, kdy a co se má vyzkoušet, kdy si o tom spolu popovídají. Měl by dát k téže otázce prostor k vyjádření více žákům.



ČTENÍ SCÉNÁŘŮ

V těchto úlohách mají žáci dán programový kód (scénář) a mají sami, bez použití počítače, **přijít na to, „co program dělá“**. Mohou takový obrázek nakreslit na papír, zapsat, mohou říci slovy, co postava bude dělat. Mohou také přiřazovat jednotlivé scénáře k nakresleným obrázkům apod. Cílem je, aby se žáci naučili nejen scénáře sestavovat, ale též číst, tedy představit si, co se bude podle takového kódu dít.

Jde opět o **společnou aktivitu celé třídy**. V první části žáci pracují sami nebo ve skupinkách, odpovídají pak před celou třídou. Vždy by měl odpovídat více než jeden žák, více než jedna skupina, aby zaznělo více (třeba i více správných) řešení.



DISKUSE

Opět jde o hromadnou aktivitu pro celou třídu. Často jde o modelovou úlohu, v níž žáci **přemýšlí nad konkrétní situací a mají ji vysvětlit** (např. proč někdo v programu udělal chybu).

Zde je opět výhodné, když jsou žáci organizováni ve skupinách. Pokud totiž diskutují jen za sebe, často nenajdou odvahu a říkají pouze odpovědi, které se naučili nebo kterými jsou si jisti. Pak se aktivita degraduje na otázky učitele s vyvoláváním. Diskuse by měla být spontánní, žáci by se měli přit, navzájem vysvětlovat. **Učitel by měl diskusi moderovat**, nikoliv stát na jedné straně, zastávat nějaký názor.

Je hodně těžké diskusi vést, zvláště tehdy, když nevede k výsledku, který si učitel přeje. **Nikdy by ale neměl diskusi tlačit do nějakého (správného) výsledku**. Spíše používá věty jako „A co si myslíš ty? Souhlasíte s tímto názorem nebo výpovědí? Jak bychom si to ověřili?“

Časem učitel zjistí, že **důležitější než výsledek diskuse** je diskuse jako taková, možnost argumentovat. Pro nejlepší žáky je příležitostí **vysvětlit něco** slabým žákům (což pro ně nebude snadné, protože slabý žák to od nich musí pochopit). Pro všechny je pak příležitostí k rozvíjení svého přemýšlení a to je důležitějším cílem než naučit se programovat.

Pokud se v diskusi třída nad nějakým řešením neshodne, lze před celou třídou příslušný scénář ve Scratchi sestavit a ověřit. Pak ale znovu musí přijít diskuse s vysvětlováním, proč počítač udělal to, co udělal. Ověření „na vlastní oči“ nezajistí porozumění.

NAVIGACE V UČEBNICI

ZAČÁTEK KAPITOL

Každá kapitola má své cíle, svůj hlavní koncept, který učí a k jehož pochopení směřují aktivity, seřazené jako korálky na niti.

V žákovském listu je na začátku každé kapitoly uvedeno, jaké **nové dovednosti** z programování žáci v kapitole zvládnou a také na jakých projektech budou pracovat, **jaké činnosti budou postavy vykonávat**. Slouží k motivaci žáků nebo jako pomůcka učitele pro vytvoření motivace.

Na konci kapitoly je pak shrnutí dovedností, které se žáci v kapitole naučili. Učitel může s žáky jednotlivé vypsané dovednosti procházet a hledat, při kterých úlohách se danou věc naučili.

V **metodické příručce** jsou na začátku každé kapitoly kromě výše uvedeného vyjmenovány **projekty**, se kterými se žáci setkají, a také **nové bloky**, které žáci budou používat.

IKONY PROJEKTŮ



Na každém snímku žákovských listů je v pravém horním rohu ikona **projektu, který má žák otevřen** a na němž pracuje. Ikona dává přehled, ke kterému projektu se činnost vztahuje.

Ikona kočky znamená, že se začíná s prázdnou scénou (nabídka *Soubor/Nový*).

STARTOVNÍ IKONA



Tato ikona v levém horním rohu snímku znamená, že **v tomto místě se má celá třída sejít a pracovat spolu**, od začátku. Ikona odstartuje hromadnou činnost celé třídy.

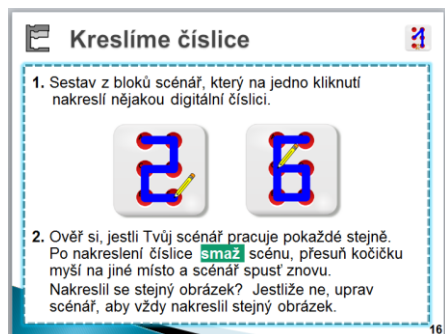
Tato ikona pomáhá učiteli, aby měl situaci stále pod kontrolou i tehdy, kdy žáci pracují individuálně a někteří jsou o několik snímků napřed. Žákům, kteří pracují rychle, stačí říci, aby došli až k této ikoně, nikoliv dále. Mezitím slabší žáci splní základní učivo a na pokyn učitele pak přeskočí úlohy pro pokročilé až k této ikoně. Třída bude opět pracovat společně.

Tato ikona se často **vykazuje u diskuse nebo u čtení scénáře**, kdy je třeba, aby třída pracovala spolu.

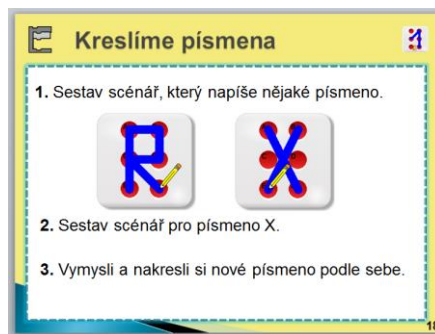
ZÁKLADNÍ A ROZŠIŘUJÍCÍ UČIVO

V žákovských listech je rozlišeno základní a rozšiřující učivo podbarvením snímku.

základní:



rozšiřující:



Pokud žák **zvládne základní učivo**, má předpoklad získat všechny vyjmenované kompetence a výstupy z RVP pro informatiku na 2. stupni ZŠ.

Předpokládáme, že na víceletém gymnáziu projdou všichni žáci prakticky všechny úlohy.

V žákovských listech jsme k vyznačení obtížnější úlohy záměrně nepoužili název „úloha pro pokročilé“ nebo standardní hvězdičku, protože podle zkušeností učitelů žáci nechtěli takové úlohy řešit; měli obavy, že pro ně budou těžké.

SBÍRKA ÚLOH

Pro procvičování je k učebnici přidaná **sbírka úloh** jako zvláštní nečíslovaná kapitola. Jde o sadu žákovských listů napříč kapitolami, které obsahují úlohy podobné úlohám z učebnice nebo jejich varianty. K úlohám ve sbírce nejsou vytvořeny metodické poznámky, pro některé z úloh poskytujeme vzorová řešení.

JAK UČIT PROGRAMOVÁNÍ

Programování je praktická činnost, jejíž výsledek je na počítači ihned po spuštění programu (scénáře) vidět. **Počítač žákovi poskytuje zpětnou vazbu**, tedy zda programoval správně nebo s chybou. **Učitel se musí naučit této výhody využít.**

Zde uvádíme **základní doporučení**, jak učit, aby se **žák učil aktivně si vytvářet své znalosti**. V dalším textu je vysvětlíme.

1. Nesdělujeme žákům správná řešení.
2. Neučme definice a pamětní poznatky.
3. Obrňme se trpělivostí.
4. Očekávejme živé a rušné prostředí (a podporujme jej).
5. Nechejme žáky pracovat ve dvojicích.
6. Hodnoťme praktické dovednosti.

1. Učitel by **neměl dělat tu chybu, že on sám říká správné řešení** nebo říká, že je program napsán správně. Měl by ponechat na počítači, aby to žákovi ukázal. Pokud učitel prochází lavicemi a narazí na chybný programový kód, lépe než „Toto máš špatně“ je říci „Spusť mi tento scénář“. Příště si žák scénář zkontroluje sám a naučí se samostatnosti.

Většinou **neexistuje jedno správné řešení**. Učitel by neměl bazírovat na tom, že jeho řešení nebo řešení z učebnice je jediné správné. Naopak oceňujme žáky, kteří přicházejí s novými, netradičními řešeními (i když vždy správná nebudou). Pokud je žákovo řešení málo obecné, nastavme situaci, v níž nebude fungovat, a zadejme žákovi, aby své řešení předvedl.

Ovšem pozor! Především na začátku **nepodporujme**, aby žák řešil úlohu tím, že **najde nějaký nový blok**, který mu řešení usnadní. Většinou se hned v následující úloze dostane do potíží, protože jeho objevený blok najednou nebude fungovat, jak žák potřebuje (příkladem je *Nastav směr* namísto *Otoč se o*). K řešení všech úloh stačí znalost bloků, které již žák zná z dřívější výuky; úlohy jsou tak záměrně vytvářeny. Nahrazování přemýšlení hledáním nástrojů, kterými počítač úkol vyřeší za žáka, nerozvíjí inženýrské myšlení.

2. Nemá valného smyslu učit děti definice, aby uměly pěkně odpovídat na otázky. To z nich mladé programátory neudělá a jejich myšlení to nijak nerozvine. Je také **zbytečné na začátku hodiny vše znovu vysvětlovat**, opakovat, žáky vyvolávat. Oni chtějí pracovat ve Scratchi, programovat, tvořit, tak je nechejme. Praktickou činností si látku zopakují rychleji.

3. Obrňme se trpělivostí. Trvá to třeba dva až tři měsíce, než se žáci dokážou v prostředí Scratche dobře orientovat a využívat jej. Pak ale jakoby najednou celý ten systém pochopí a začnou pracovat více automaticky, efektivněji a samostatněji. Žáci nemusí věci ihned pečlivě zvládnout. Není třeba být v tomto směru úzkostlivý; žáci pochopí věci postupně.

4. Očekávejme živé a rušné prostředí. Při programování a při práci ve skupině je to přirozené. Pokud je učitel zvyklý, že při psaní a kreslení na počítači je ve třídě ticho, pak může být překvapen, že žáci si chtějí a potřebují něco vysvětlovat, ukazovat, projevují nahlas emoce. Učitel by měl vědět, že tyto projevy jsou znakem kvalitní výuky. Je na učiteli, aby si ohlídal, aby žáci nediskutovali mimo téma výuky nebo zadané úlohy.

VE SKUPINÁCH, ANEBO 1 ŽÁK NA 1 POČÍTAČ?

Programování je velmi vhodné téma pro práci v týmech, ve dvojicích. Žáci se mohou navzájem radit, argumentovat, spolupracovat a tím se učit komplexněji. **Dovolme žákům, aby ve skupinách programovali.** Mimo jiné, v jejich budoucím zaměstnání tomu tak většinou bude, že budou kooperovat.

Učitelé mají **obavy**, že slabý žák se ve skupině jen „poveze“. Je ovšem otázkou, zda se sám namísto v týmu opravdu naučí více, nebo se bude pouze více trápit. Ve skupině bude moci alespoň něco odkoukat, bude moci spolupracovat. I když se programovat nenaučí, tak poznává sociální interakce v pracovní skupině, a to není málo.

Žáky, kteří chtějí pracovat sami, nechejte po nějakou dobu pracovat individuálně. Sami často přijdou na výhodnou možnost s někým své nápady konzultovat. Až i ti nejlepší narazí na pro ně obtížný úkol, budou rádi spolupracovat s někým, s kým se poradí.

Vyzkoušejte, je-li výhodnější dát k sobě intelektově podobné žáky, aby si lépe porozuměli.

JAK HODNOTIT, ZNÁMKOVAT ŽÁKY

Vždy hodnotíme praktické dovednosti: žák umí sestavit správný program, umí jej přečíst a vysvětlit, co program dělá, umí najít v programu chybu a opravit ji. Kromě tradičních metod (samostatná práce na naprogramování nějaké úlohy nebo test z úlohy na čtení programu) se nabízí objektivnější metody. Velmi vhodné je **pozorování** žáka při práci ve skupině či **individuální konzultování** jeho rozpracované práce. Klasifikace je pak snadná, protože je podobná slovnímu hodnocení.

Příklad takového hodnocení žáka, které učitel provede **individuálně** (zatímco ostatní žáci pracují): „Tomáši, vidím, že se snažíš, mám radost z toho, jak znovu a znovu zkoušíš úlohu opravit. Postavit domek se ti opravdu povedlo. Ještě ti nejde dobře poznat, jak mají být bloky ve scénáři uspořádány, nevíš, jak přijít na chybu ve scénáři. Tady se potřebuješ zlepšit. Za dosavadní práci ti dávám dvojku (do žákovské knížky).“

Další možností je **analyzovat práci žáka** (nechat si soubor na konci aktivity uložit a prohlédnout si, jak žák sestavil scénáře, jestli používá nějaké pokročilejší příkazy jako opakování, podmínky atd.).

Nikomu tento způsob hodnocení nevnučujeme. Co bychom ovšem **neradi**, aby učitelé ze zkoušení dělali nástroj k udržení své autority, aby zkoušeli, jestli žáci rychle odpovídají na jednoduché otázky nebo jestli něco umí z paměti – tím jistě schopnost programovat neověří.

Jako zvláštní kapitolu k metodice přidáváme vzorové ukázky testů či programovacích „písemek“, na nichž ukazujeme, jaké znalosti považujeme za důležité zkoušet. Tyto testy jsou připraveny k zařazení po absolvování 5. a 10. kapitoly. Jsou vytvořeny ve více variantách, které ovšem nejsou srovnatelné úrovně a nelze je použít jako „skupina A, B“. Testovací projekty jsou umístěny ve studiu *Testy pro opakování*, pro offline práci ve složce *Pracovní soubory*, pro online práci na adrese <https://scratch.mit.edu/studios/7811069/>.

Učitel tyto testy vůbec použít nemusí; raději doporučíme jednu z výše uvedených metod hodnocení. Ukázky testů slouží k získání představy, jaké dovednosti z oblasti sestavení a čtení scénáře vidí autoři jako důležité.

JAK VÉST VÝUKU

Kromě **respektování druhu žákovské aktivity** (dané ikonou na žákovském listu) je užitečné dodržovat několik pravidel, která vám usnadní práci a pomohou zajistit, aby žáci „uměli“.

Aby žáci programování zvládli:

- **Každý nový blok je třeba nejprve spustit samostatně**, spustit jej třeba vícekrát a zkoumat, co dělá. Teprve pak je vhodné jej zařadit do scénáře.
- **Nesestavujte dlouhé scénáře**. Kratší jsou srozumitelnější.
- Pokud program nedělá to, co žák chce, **není** dobrou cestou vše smazat a začít skládat nový scénář od začátku. Je vhodné s žákem **projít řádek po řádku, co program dělá**.
- **Nechvátejte**. Vše chce čas. V prvních deseti hodinách se Vám bude zdát, že tempo je pomalé, žáci si nic nepamatují a nevědí, jak pracovat. Teprve časem žáci najednou začnou sami opravovat, spouštět scénáře, měnit bloky, experimentovat.
- Situací, v nichž si nebudete vědět rady, se hlavně na začátku vyvarujete tím, že **nedovolíte žákům dělat si, co je napadne**. Pokud se žákovi „něco stalo“ s programem, často je to proto, že se nedržel pokynů úlohy, zkoušel něco jiného podle sebe. V takovém případě je vhodné nechat žáka pracovat od začátku podle pokynů.
- Nejen aktivity, v nichž žáci programují samostatně, jsou důležité. **Nepřeskakujte úlohy na čtení programu**, na diskusi, objevování. Žáci se tak učí dívat se na problémy komplexněji, z více úhlů, a lépe rozumí programovému kódu.
- Je vhodné občas **přiznat, že něco nevíte**, např. že nevíte, čím je chyba způsobena. Nic se neděje, žák aspoň příště bude více spoléhat na sebe, bude experimentovat rozumně.

JAK SE PŘIPRAVOVAT NA VÝUKU

- Pokud učíte podle učebnice poprvé, je třeba **vyhradit si dostatek času**. Počítejte s tím, že Vám **příprava vezme více času, než výuka** samotná. Toto hrozí především po několika kapitolách, když už látce „v globálu“ budete rozumět, ale bez precizní přípravy bude riziko, že při výuce budete dělat chyby, kterých si ani nebudete vědomi.
- **Nestačí, když budete mít před žáky „náskok“ jednu kapitolu**. Potřebujete vědět, k čemu zvládnuté dovednosti směřují a jak budou naučené pojmy a bloky používány v dalších kapitolách.
- Projděte si úlohy v žákovských listech a **zkuste je nejprve vyřešit sami, bez nápovědy**. Teprve pak se podívejte do metodiky. Projdete si tak chybami, které vaši mohou žáci dělat také. „Kouknutí se“ na výsledky nikoho programovat nenaučilo.
- Některé úlohy se zdají na pohled snadné a člověka to **svádí k tomu**, že jejich vyzkoušení při přípravě na hodinu **přeskočí**. Tím si může zadělat na problém při samotné hodině, protože se mu scénář může chovat jinak, než očekával.
- Je dobré vědět, jaké **cíle** se vlastně výukou sledují. Jsou uvedeny v metodice u úloh nebo na začátku kapitol.

Zkušenému učiteli netřeba radit. Až budete učit druhým rokem, řadu problémů nebudete jako problém vůbec vnímat. Až budete učit potřetí, možná učebnici nebudete ani potřebovat.

ZÁVĚREM

Hlavním naším cílem není naučit děti programovat, ale pomocí programování rozvíjet jejich schopnost přemýšlet, plánovat činnosti a rozumět světu počítačů. Programování je „hřiště“, na kterém žáci trénují své mentální schopnosti. Je to další způsob, jak žákům umožnit se sebezdokonalovat a porozumět tomu, jak funguje dnešní svět.

Nechejte žáky, aby se mohli učit aktivně, přicházet na nové věci sami.

Budeme Vám držet palce.

červen 2020

autoři

