Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

ŠTÁTNY PEDAGOGICKÝ ÚSTAV
NATIONAL INSTITUTE FOR EDUCATION

Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Didaktika programovania pre ZŠ 1

Predmet: Didaktika programovania pre ZŠ

Línia: Didaktika informatiky a informatickej výchovy

Operačný program VZDELÁVANIE

EURÓPSKA ÚNIA
Európsky sociálny fond

Európska únia
Európsky sociálny fond

Moderné vzdelávanie pre vedomostnú spoločnosť/Projekt je spolufinancovaný zo zdrojov ES

Didaktika programovania pre ZŠ 1

Garant predmetu:

RNDr. Gabriela Lovászová, PhD.
KI FPV UKF, Nitra
glovaszova@ukf.sk

Autori:

RNDr. Gabriela Lovászová, PhD.
KI FPV UKF Nitra
Mgr. Martin Cápay, PhD.
KI FPV UKF Nitra
PaedDr. Viera Palmárová, PhD.
KI FPV UKF Nitra

Identifikácia modulu

Aktivita projektu:	1.2 Vzdelávanie nekvalifikovaných učiteľov informatiky na 2. stupni ZŠ a na SŠ
Línia aktivity:	Didaktika informatiky a informatickej výchovy
Predmet:	Didaktika programovania pre ZŠ

Zaradenie modulu

Po úvodnom module Didaktika programovania 0 pokračuje predmet Didaktika programovania dvomi alternatívnymi modulmi pre základnú a strednú školu. Modul Didaktika programovanie pre ZŠ 1 je prvým z dvoch modulov, ktoré sa zaoberajú didaktikou programovania na 2. stupni základnej školy.

Abstrakt modulu

V module Didaktika programovania pre ZŠ 1 získa účastník vzdelávania ucelenú predstavu o cieľoch, obsahu a metodike vyučovania programovania na 2. stupni základnej školy.

Ako programovacie prostredie vhodné na vyučovanie programovania v nižšom sekundárnom vzdelávaní sa v module používa programovací jazyk Imagine Logo. Voľbu tohto prostredia podporuje aj existencia učebnice a ďalších materiálov o programovaní v Logu dostupných učiteľom informatiky v slovenskom jazyku. Inšpirácie na logovské aktivity sú čerpané aj z iných vyučovacích programovacích prostredí.

Obsah vyučovania programovania je v module rozdelený do troch tematických celkov: Korytnačia grafika, Objekty a udalosti, Tvary a animované tvary. V každom celku sa podrobne analyzuje učivo a navrhujú sa metodické postupy podporujúce konštruktívny poznávací proces. Dôraz sa kladie na motiváciu a zmysluplnosť aktivít.

Absolvovaním modulu získa účastník vzdelávania prehľad o obsahu vyučovania programovania na ZŠ a o metodických postupoch na vyučovanie jednotlivých tém. V nadväzujúcom module Didaktika programovania na ZŠ 2 využije tieto vedomosti pri príprave konkrétnych vyučovacích hodín.

Modul má e-learningovú podporu

Obsah

Úvod

Predmet Didaktika programovania na ZŠ je rozdelený do dvoch modulov. V tomto module sa budeme zaoberať obsahom a metodikou vyučovania programovania na úrovni *tematických celkov*. V druhom module sa zameriame na menšiu vyučovaciu jednotku a budeme pripravovať konkrétne *vyučovacie hodiny*.

V každom tematickom celku si najprv zhrnieme, čo by mal o téme vedieť *učiteľ*. V materiáli sú pripravené úlohy, riešenie ktorých vám pomôže „upratať" si svoje vedomosti a prípadne doplniť ich. Ak ste v programovaní v Imagine Logu zruční, môžete tieto úlohy vynechať. Potom analyzujeme tému z pohľadu *žiaka*. Uvádzame námety na prvé stretnutie žiakov s témou a vyučovanie ďalších podtém. Zdôrazňujeme úlohu motivácie v poznávacom procese. Okrem programovania začínajúceho „prázdnym projektom" využívame dokončovanie, úpravu projektov a prácu s malými didaktickými aplikáciami.

Okrem tohto študijného materiálu pracujeme v module s učebnicou programovania pre ZŠ [3]. Analyzujeme a vysvetľujeme metodické postupy z učebnice a dopĺňame ich ďalšími námetmi.

Cieľ modulu

Ciele modulu sú:

- Poznať ciele a obsah vyučovania programovania na ZŠ.
- Byť zručný v programovaní v prostredí Imagine Logo na úrovni štandardov pre ZŠ:
  - korytnačia grafika, cyklus, procedúry, parametre,
  - viac objektov v projekte, vlastnosti objektov, udalosti, podmienený príkaz,
  - tvary korytnačiek, procesy.
- Poznať učebnicu programovania v Imagine Logu na ZŠ.
- Vedieť analyzovať učivo a navrhovať metodické postupy podľa tematických celkov.

Vstupné vedomosti

Požadované prerekvizity

Moduly: Úvod do programovania, Malé programovacie jazyky a Didaktika programovania 0.

Predpokladané vstupné vedomosti, skúsenosti a zručnosti

Predpokladá sa, že:

- Účastníci vzdelávania vedia používať v programovacom jazyku Imagine Logo príkazy korytnačej grafiky, cyklus, vlastné príkazy bez parametra a s parametrom, programovať reakcie na udalosti, procesy.
- Vedia vymedziť postavenie programovania v rámci informatiky na základnej a strednej škole; poznajú historické súvislosti vzniku a vývoja programovacích jazykov; dokážu analyzovať a posúdiť vhodnosť programovacích jazykov a prostredí pre potreby vyučovania programovania; poznajú etapy konštruktívneho poznávacieho procesu.
- Poznajú niekoľko programovacích jazykov vhodných na vyučovanie základov programovania (Karel, Baltík, IzyLogo, Živý obraz, Scratch); vedia zhodnotiť prínos týchto jazykov pre vyučovanie programovania.

1 Postupy, riešenie problémov, algoritmizácia

V tomto module sa budeme zaoberať vyučovaním tematického okruhu *Postupy, riešenie problémov, algoritmické myslenie*. Vzdelávací štandard za celý stupeň vzdelávania nájdeme v Štátnom vzdelávacom programe (ŠVP) v dokumente pre kategóriu ISCED 2:

Obsahový štandard vymenúva, *čo má učiteľ učiť*, na aké pojmy, vlastnosti, vzťahy a postupy sa má vo vyučovaní zamerať:

- postup riešenia, formálny zápis riešenia, etapy riešenia problémov;
- programovací jazyk, elementárny príkaz, postupnosť, cyklus, procedúra, parametre, premenná, hodnota, priradenie;
- zložitosť riešenia problému.

Výkonový štandard konkretizuje, *čo má žiak* po absolvovaní nižšieho sekundárneho vzdelávania *vedieť*:

- zapisovať postupy riešenia formálnym spôsobom a interpretovať takého zápisy;
- v detskom programovacom prostredí riešiť úlohy s opakovaním nejakých činností, zoskupovaním častí riešenia do procedúr, zapamätávaním výsledkov výpočtov do premenných;
- porovnať čas trvania rôznych riešení problému.

O všeobecných cieľoch a obsahu predmetu informatika v nižšom sekundárnom vzdelávaní ste diskutovali aj v module *Didaktika programovania* [7].

Moderné programovacie prostredia vhodné pre deti ste bližšie spoznali v module *Malé programovacie jazyky* [5].

Učebnica o programovaní v prostredí *Imagine Logo* určená pre 2. stupeň ZŠ a nižší stupeň osemročných gymnázií [3] je rozdelená do 10 kapitol:

1. Pero, farba a hrúbka
2. Opakuj a pomenuj
3. Stavebnice príkazov
4. Dôležité udalosti v živote korytnačky
5. Viac korytnačiek, viac možností
6. Korytnačky a ich tvary
7. Animované tvary a procesy
8. Príkazy s premennými
9. Pohyby a preteky
10. Pokusy a ďalšie hry

Súčasťou učebnice je CD so žiackou licenciou prostredia *Imagine Logo*. Inštalácia obsahuje aj priečinok s hotovými projektmi, ktoré dopĺňajú úlohy v učebnici o ďalšie aktivity zamerané na zbieranie skúseností, experimentovanie a rozvíjanie tvorivosti.

Aktivita 1.1 **Diskusia**	Pracujte vo dvojiciach. Prelistujte učebnicu *Tvorivá informatika - 1. zošit z programovania*. V jednotlivých kapitolách učebnice identifikujte cieľové vedomosti, zručnosti a spôsobilosti naznačené vo vzdelávacom štandarde.
Aktivita 1.2 **Diskusia**	Ujasnime si niektoré formulácie zo štandardu: a) Uveďte rôzne príklady formálneho zápisu postupu riešenia (algoritmu), aj také, ktoré nesúvisia so žiadnym programovacím prostredím. b) Podľa akých kritérií sú schopní žiaci rozhodnúť o tom, ktoré z dvoch riešení úlohy je lepšie?
Aktivita 1.3 **Diskusia**	ŠVP nepredpisuje používanie žiadneho konkrétneho programovacieho prostredia. Učiteľ si môže vybrať sám. Kritériom výberu je dostupnosť prostredia a metodickej podpory (učebnica, komunita používateľov), vlastnosti a možnosti prostredia. Imagine Logo je jedna z odporúčaných alternatív. Ako by ste zdôvodnili jeho používanie vo svojom Školskom vzdelávacom programe?

Čo sme sa naučili

Preštudovali sme obsahový a výkonový štandard tematického okruhu *Postupy, riešenie problémov a algoritmické myslenie* v dokumente ISCED 2. Porovnali sme ho s odporúčanou učebnicou. Vieme zdôvodniť, prečo vo vyučovaní používame programovacie prostredie Imagine Logo.

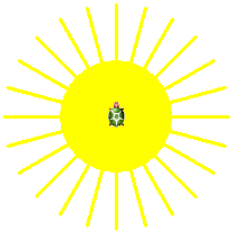
2 Preverenie vstupných vedomostí

S programovacím prostredím Imagine Logo ste sa zoznámili už v module *Úvod do programovania* v 1. semestri štúdia. Pripomeňme si obsah tohto modulu:

- posúvaním a otáčaním korytnačky ste na stránke kreslili obrázky z čiar a bodov, nastavovali ste rôzne farby a hrúbky pera,
- pri kreslení obrázkov ste príkazy na pohyb, otáčanie a nastavenie pera používali aj s náhodnými vstupmi,
- kreslenie obrázkov s opakujúcim sa vzorom (s opakujúcou sa postupnosťou príkazov) ste si zjednodušili použitím príkazu `opakuj` (cyklu s pevným počtom opakovaní),
- množinu príkazov, ktoré korytnačka pozná, ste sa naučili rozšíriť definovaním vlastných príkazov (bez parametrov i s parametrami),
- korytnačkám ste naprogramovali reakcie na udalosť `priKliknutí` alebo `priŤahaní`, zmazanie stránky alebo kreslenie obrázkov ste realizovali pri zapnutí tlačidla,
- stránke ste nastavili pozadie, umiestnili ste na ňu viac korytnačiek, nastavili ste im rôzne tvary (statické i animované), vytvorili ste z nich obraz, a oživili ho spustením nezávislých procesov (pomocou príkazu `každých`).

Aby sme sa dobre pripravili na ďalšie témy v tomto module, vyriešime sériu úloh zameraných na zopakovanie a doplnenie vyššie vymenovaných vedomostí a zručností.

Základy korytnačej grafiky a tvorba vlastných príkazov:



```
? nechFp "žltá
? nechHp 3

? opakuj 24 [
    do 100
    vz 100
    vp 15
  ]

? bod 100
```

Projekt *Farebný panelák*

Všetky úlohy, ktoré budeme v tejto kapitole riešiť, smerujú k vytvoreniu interaktívneho projektu s farebným panelákom:

Úloha 2.1

a) Do príkazového riadku napíšte postupnosť príkazov, ktorou nakreslíte rad troch štvorcových okien s dĺžkou strany 50.

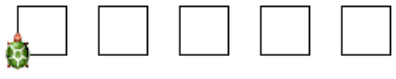
b) Vytvorte vlastný príkaz `okno` a nakreslite pomocou neho rad 5 okien.

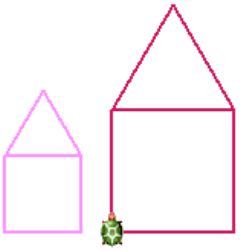
c) Upravte príkaz `okno` tak, aby ste ho mohli použiť na kreslenie okien s ľubovoľnou dĺžkou strany. Nakreslite viac radov rôzne veľkých okien.

Riešenie:

V príkaze `okno` pridáme jeden parameter, nazvime ho `d`, bude reprezentovať dĺžku strany kresleného štvorca.

```
viem okno :d
  opakuj 4 [do :d vp 90]
koniec

? opakuj 5 [okno 50 ph vp 90 do 70 vl 90 pd]
```

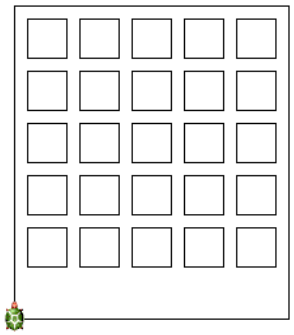




```
? uprav "domček
```

```
viem domček :a
  pd
  nechHp 2
  nechFp ?
  štvorec :a
  do :a
  vp 30
  trojuholník :a
  vl 30
  vz :a
  ph
koniec
```

```
? znovu
? domček 50
? vp 90 do 70 vl 90
? domček 80
```



Úloha 2.2

Napíšte príkaz `panelák` s tromi parametrami:

- počet radov okien,
- počet okien v rade,
- dĺžka strany štvorcového okna.

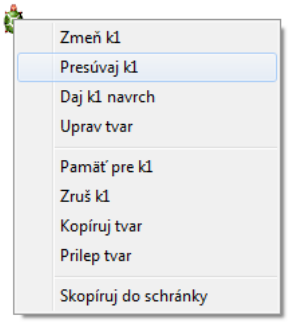
Otestujte ho s rôznymi vstupmi, napr.:

```
? panelák 5 5 50
? panelák 1 8 30
? panelák 7 2 40
```

Panelák je vhodné stavať po poschodiach. Pripravte si príkaz na nakreslenie jedného radu okien. Ten potom použite v príkaze `panelák`.

Úloha 2.3

Korytnačke `k1` dajte meno *Staviteľka*. Naprogramujte jej reakciu na udalosť `priKliknutí` tak, aby ste pomocou korytnačky-Staviteľky mohli pohodlne postaviť nový panelák. Nastavte jej vhodnú domovskú pozíciu.

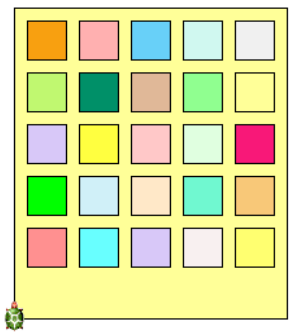


Korytnačku môžeme presunúť na iné miesto na stránke postupným zadávaním príkazov `dopredu`, `vzad`, `vľavo`, `vpravo` (obyčajne so zdvihnutým perom). Praktickejšie je korytnačku pomocou myši zdvihnúť a rovno položiť na cieľové miesto (príkazom *Presúvaj* z miestnej ponuky pre korytnačku). Novú pozíciu môžeme korytnačke nastaviť aj príkazom `nechPoz`. Napr. takto:

```
? nechPoz [0 0]
? nechPoz [100 50]
? nechPoz ?
? nechPoz pozMyši
```

V príkladoch vidíme ako nastaviť pozíciu uvedením súradníc konkrétneho bodu na stránke, určením náhodného bodu alebo bodu, v ktorom sa práve nachádza ukazovateľ myši.

Pomocnú korytnačku je vhodné skryť. Nezabudnime jej vypnúť pero!



Úloha 2.4

Aby bolo na sídlisku veselšie, zafarbite okná a fasádu paneláku náhodne volenými farbami. Farbenie na stránke realizujte pomocou myši - klikaním do vnútra okna, resp. fasády domu.

Riešenie:

Vložte na stránku novú pomocnú korytnačku, nazvite ju `Pomocná`. Objektu `Stránka1` naprogramujte reakciu na udalosť `priĽavomDolu`.

Pri stlačení ľavého tlačidla myši chceme, aby korytnačka skočila na pozíciu myši, nastavila si náhodnú farbu výplne a príkazom `vyplň` vyplnila uzavretú oblasť, v ktorej sa nachádza:

```
Pomocná'nechPoz pozMyši
Pomocná'nechFv ?
Pomocná'vyplň
```

V zložitejších projektoch sa nezaobídeme bez *príkazu vetvenia*. V jazyku Imagine Logo zapisujeme binárne vetvenie pomocou príkazov `ak` a `ak2`. Korytnačka sa môže rozhodovať napr. podľa farby bodu, na ktorom práve stojí:

```
ak farbaBodu = "sivá2 [ nechFv "žltá vyplň ]
```

Vysvetlenie: Ak korytnačka stojí na bode farby *sivá2* (t. j. ak je podmienka splnená), nastaví si farbu výplne na žltú a vyplní ňou príslušnú oblasť (t. j. vykoná postupnosť príkazov uvedenú v zátvorkách).

Pridajme do projektu tlačidlo, ktorým novej korytnačke s menom *Umývačka* nariadime umyť okno (ak práve na nejakom stojí). Naprogramujme reakciu tlačidla na udalosť `priZapnutí`:

```
ak farbaBodu <> "biela [ nechFv "biela vyplň ]
```

Ak bude pod korytnačkou bod inej ako bielej farby, korytnačka príslušnú oblasť (farebné okno) vyplní bielou farbou (umyje ho).

Čo ak budeme chcieť umyť okno čiernej farby? Spolu s vnútrom okna zafarbíme na bielo i jeho obrys ☹. Aj ten má totiž čiernu farbu. Problému sa zbavíme ľahko upravením reakcie stránky na udalosť `priKliknutí`. Po výbere náhodnej farby skontrolujeme, či sa nevybrala *čierna*. Ak áno, nahradíme ju farbou *sivá2*:

```
pre "Pomocná [
  nechPoz pozMyši
  nechFv ?
  ak farbaVýplne = "čierna [nechFv "sivá2]
  vyplň
]
```

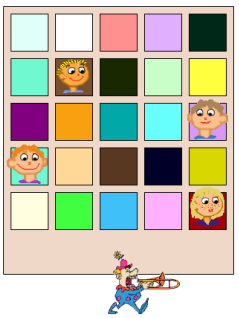
Úloha 2.5	Pridajte do projektu ďalšie tlačidlo, ktorým odštartujete pohyb korytnačky-Umývačky po stránke (spustíte pre ňu proces príkazom `každých`). Aby korytnačka našla okná, ktoré má umyť, musíme ju na ne navigovať. Otáčanie korytnačky vpravo a vľavo zabezpečte reakciami na udalosti `priĽavomDolu` a `priPravomDolu`. Poumývajte niekoľko okien.
Úloha 2.6*	Zotmelo sa, v niektorých oknách sa svieti, v iných je tma. Pridajte do projektu ďalšiu korytnačku (*Svetlušku*), ktorá sa bude pohybovať, v rozsvietených oknách zhasínať a v tmavých zase zažínať svetlo. Východiskovú situáciu si pripravte pomocou nástroja *Výplň* z *Panelu kreslenia*. Rozsvietené okná môžete reprezentovať farbou *žltá*, zhasnuté farbou *sivá2*. Pri rozlišovaní rozsvietených a tmavých okien použite príkaz `ak2`.
Úloha 2.7	Obrázok s farebným panelákom, resp. s panelákom s rozsvietenými oknami dotvorte pomocou *Panelu kreslenia* alebo nastavením vhodného pozadia zo súboru. Pridajte doň korytnačky s tvarom detí a umiestnite ich do okien. Pred panelákom nech pochoduje šašo, na ktorého sa deti pozerajú a pod.

V učebnici [3] sa príkaz `ak` prvýkrát použije v kapitole 9 pri tvorbe živého obrazu. Korytnačke nastavíme tvar loďky, domčeka alebo balóna podľa farby bodu, na ktorom ju pri ťahaní po stránke pustíme:

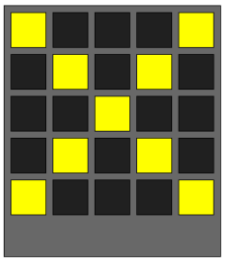
V príkaze `ak2` sú dve vetvy. V prvej napíšeme čo sa má vykonať ak je podmienka splnená, v druhej ak nie je splnená:

```
ak2 podmienka
    [ príkazy1 ]
    [ príkazy2 ]
```

Pri tvorbe zložených podmienok môžeme používať logické spojky `zároveň`, `alebo`, `nieJe`. Vyhľadajte informáciu o nich v *Pomocníkovi* (F1).



Okná na paneláku môžeme rozsvietiť tak, aby vytvárali pekný vzor:



3 Korytnačia grafika

Po zopakovaní a doplnení vedomostí z programovania v prostredí Imagine Logo sa vráťme na začiatok a venujme sa didaktike programovania. Budeme analyzovať obsah učiva po tematických celkoch a navrhovať metodické postupy vyučovania.

Korytnačia grafika je jadrom programovacieho jazyka Imagine Logo. Grafické pero je personifikované do postavy korytnačky, ktorá má svoju polohu a smer natočenia. Korytnačka je ovládaná príkazmi na zmenu polohy v aktuálnom smere natočenia a na zmenu smeru. Korytnačka sa môže pohybovať so zdvihnutým perom (nezanecháva stopu) alebo s položeným perom (zanecháva stopu). Atraktívnejšie obrázky nakreslíme rôznymi nastaveniami vlastností pera (farby a hrúbky).

Program na nakreslenie šípky v Pascale:

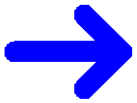
```
Image1.Canvas.Pen.Color:=clBlue;
Image1.Canvas.Pen.Width:=20;
Image1.Canvas.MoveTo(300,200);
Image1.Canvas.LineTo(400,200);
Image1.Canvas.LineTo(365,165);
Image1.Canvas.MoveTo(400,200);
Image1.Canvas.LineTo(365,235);
```

v Logu:

```
nechfp "modrá
nechhp 20
nechSmer 90
pd
do 100
vl 135
do 50
vz 50
vl 90
do 50
```

V rodnom liste korytnačky na karte *Pozícia* zmeňte typ oblasti: *Dokola*, *Bez hraníc*, *S hranicou*, *S odrazom*.

Aký vplyv má typ oblasti na pohyb korytnačky?

Úloha 3.1	Porovnajte spôsob kreslenia v programovacích jazykoch Pascal a Logo. Napíšte program, ktorý nakreslí šípku v Pascale a v Logu. 
Úloha 3.2	Vymenujte čo najviac príkazov jazyka Logo, ktoré slúžia na ovládanie pohybu korytnačky a kreslenie. Navrhnite minimálnu množinu príkazov, s ktorou sa dá začať s programovaním v Logu.
Úloha 3.3	Niektoré príkazy majú vstupy (parametre). Čo môže byť vstupom príkazov `dopredu`, `vľavo`, `nechFarbaPera`?

Parametrami príkazov môžu byť čísla (celé, desatinné s desatinnou bodkou), slová (začínajú úvodzovkou a končia medzerou alebo iným oddeľovačom) alebo zoznamy (v hranatých zátvorkách). Môžu byť zadané ako konštantné hodnoty alebo ako výraz.

Experimentujme s rôznymi vstupmi a sledujme správanie korytnačky:

```
dopredu 100
dopredu -100
dopredu 100.5
dopredu 5000
dopredu 20+40
dopredu ?
vľavo 270
vľavo -90
vľavo 1000
vľavo ?
vľavo 5
vľavo -90 + náhodne 181
nechFarbaPera "červená
nechFarbaPera "červená8
nechFarbaPera 12
nechFarbaPera [255 84 185]
nechFarbaPera [? 84 185]
nechFarbaPera ?
nechFarbaPera ?prvok [zelená4 zelená5 zelená6 zelená7]
```

Možné problémy so zadávaním príkazov a ich parametrov:

- prečo sa korytnačka posunula vzad, keď zadám `dopredu`?
- prečo sa korytnačka otočila vpravo, keď zadám `vľavo`?
- prečo sa korytnačka neotočila, keď zadám `vľavo`?
- prečo korytnačka nič nespraví, keď zadám príkaz `nechFarbaPera`?

Úloha 3.4

Ak zabudneme príkazu so vstupom zadať vstupný údaj, zobrazí sa pomôcka. Zistite, ktoré príkazy majú pomôcky a ktoré príkazy pomôcky nemajú.

Pomôcku môžeme vyvolať z príkazového riadka:

- vedome stlačením klávesu F9. Voľbou *Urob!* sa doplní vstup a príkaz sa vykoná. Voľbou *Napíš!* sa vstup len doplní za meno príkazu.
- keď zabudneme zadať vstup a po napísaní príkazu stlačíme ENTER (len ak je príkaz jediný v riadku). Vtedy je v pomôcke dostupná len voľba *Urob!*

Pomôcky sú dialógové okná, ktoré slúžia na uľahčenie zadávania vstupov príkazov. V dialógovom okne zadáme hodnotu vstupu, ktorý sa doplní do príkazového riadka za príkaz.

Používanie pomôcok je užitočné, keď:

- žiak nevie odhadnúť vzdialenosť, ktorú predstavuje vstup pre príkazy `dopredu` alebo `vzad`,
- žiak neovláda pojem uhol a jeho veľkosť v stupňoch alebo je preňho ťažké orientovať sa, keď korytnačka nie je v smere 0,
- je syntax vstupu pre žiaka náročná, napríklad zadávanie farby slovom s úvodzovkami,
- žiak zabudne zadať vstup, nevygeneruje sa chybové hlásenie.

Prvé stretnutie žiaka s korytnačou grafikou

Propedeutika korytnačieho pohybu

Relatívny pohyb korytnačky závislý od aktuálneho smeru natočenia môže byť pre žiakov náročný na orientáciu. Ako úvod do programovania v Logu môžeme precvičiť korytnačí pohyb v aktivitách s **interaktívnym ovládaním pohybu**.

Tlačidlá na relatívny pohyb Baltíka v režime Čarovať scénu:

V *IzyLogu* v *priamom režime* navigujeme korytnačku alebo objekt iného tvaru stláčaním tlačidiel so šípkami dopredu, vľavo a vpravo. Rovnakým spôsobom ovládame pohyb *Baltíka* v režime *Čarovať scénu*. V *Scratchi* môžeme naprogramovať aktivitu na riadenie pohybu klávesovými šípkami v relatívnych smeroch.

Tlačidlá na relatívny pohyb v IzyLogu v priamom režime.

Na prvom stupni základnej školy sa žiaci na hodinách informatickej výchovy stretli s **ikonickým programovaním**. Vedia zostavovať krátke postupnosti ikonických príkazov. Ikonické programovanie je pre deti mladšieho školského veku vhodné najmä preto, lebo eliminuje problémy s editovaním príkazov (malá zručnosť v práci s klávesnicou, preklepy, syntaktické chyby). Ikonické príkazy sa vyberajú z ponuky a zoraďujú sa do postupnosti väčšinou ťahaním alebo klikaním myšou.

Úloha 3.5

Vyskúšajte si ikonické programovanie v projekte *robot.imp*. Úlohou robota je rozsvietiť modré políčka v miestnosti. Robot sa riadi ikonickými príkazmi *dopredu*, *vľavo*, *vpravo*, *rozsvieť* a *skoč*. Príkazy sa ťahaním myšou umiestňujú do prázdnych okienok na spodnom okraji okna.

Vyriešte postupne všetky úlohy. Pripravte vlastné zadanie.

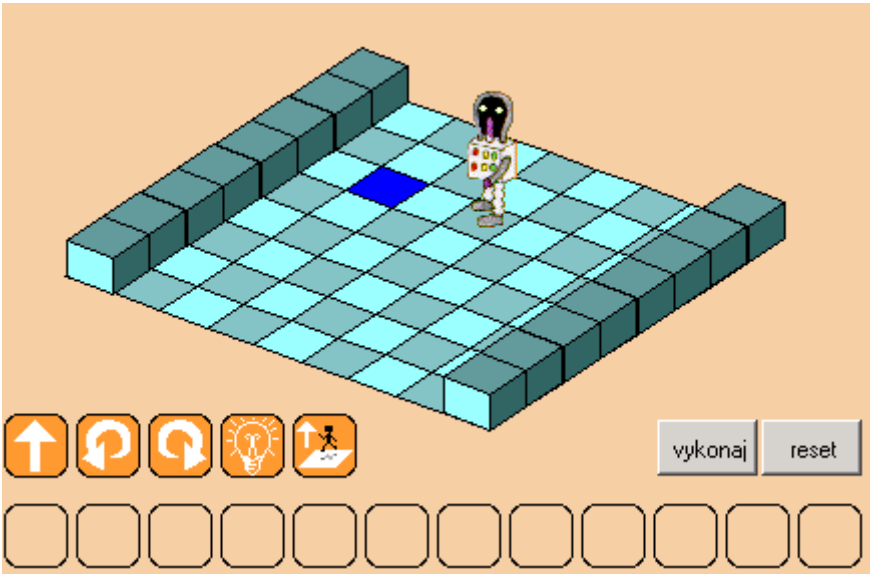
Projekt *robot.imp* je v e-learningovom kurze k tomuto modulu.

Zadania úloh v jednotlivých miestnostiach sa dajú upravovať v textových súboroch. Každá miestnosť je uložená v textovom súbore, ktorý obsahuje maticu čísiel. Každé číslo kóduje počet tehličiek na danom štvorčeku miestnosti. Štvorček, ktorý má byť rozsvietený, má k počtu tehličiek pridaný znak „x“. Umiestnenie robota v miestnosti je vyznačené znakom „r“ za počtom tehličiek na príslušnom mieste v matici. Číslo 1 až 4 za symbolom robota určuje jeho začiatočný smer (na sever, na východ, na juh, na západ). Pri vytváraní miestnosti dávajme pozor, aby bola úloha riešiteľná na 12 príkazov (počet okienok pre príkazy).

Kód miestnosti v programe *robot*:

```
1 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
1 0 0x 0 0r2 0 0 0 1
1 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
1 0 0 0 0 0 0 0 1
```

V projekte robot sa programy spracovávajú **dávkovo** – vytvorí sa ikonický program a spustí sa naraz tlačidlom *vykonaj*.



Obrázok 3.1: Projekt Robot

V *jednoduchom režime* prostredia IzyLogo skladáme program z kartičiek, na ktorých sú príkazy jazyka Logo. Skladanie programu z kartičiek je podobné ikonickému programovaniu, avšak obrázok je nahradený textom. Po vložení kartičky do programu sa príkaz v IzyLogu vykoná okamžite. Programovanie je **interaktívne**.

Obrázok 3.2: Interaktívne kartičkové programovanie v IzyLogu

Korytnačí pohyb v Imagine Logu

V Imagine Logu zadávame **textové** príkazy pomocou klávesnice. Z prvého stupňa základnej školy by mali žiaci vedieť používať klávesnicu a editovať krátke texty. Programujeme interaktívne, príkazy zadávame do príkazového riadka spočiatku po jednom. Po zadaní sa príkaz okamžite vykoná.

V propedeutických aktivitách v predchádzajúcej časti sa zadávali príkazy bez parametrov (pohyb dopredu o jednotkovú vzdialenosť, otočenie o uhol 90 stupňov) alebo so špecifickými parametrami pomocou číselníka v jednoduchom režime IzyLoga. V Imagine Logu môžeme precvičiť voľný pohyb korytnačky po celej stránke.

V námetoch pozadí môžeme uplatniť medzipredmetové vzťahy.

Úloha 3.6

Vymyslite motiváciu pre pohyb korytnačky. Pripravte pozadie, na ktorom sa korytnačka bude pohybovať, a sformulujte zadanie.

Práca na stránke s pozadím je viac motivujúca ako práca s prázdnou stránkou. Pozadie navyše slúži ako zadanie úlohy, pohyb korytnačky má svoj cieľ. Možnosti práce s pozadím stránky:

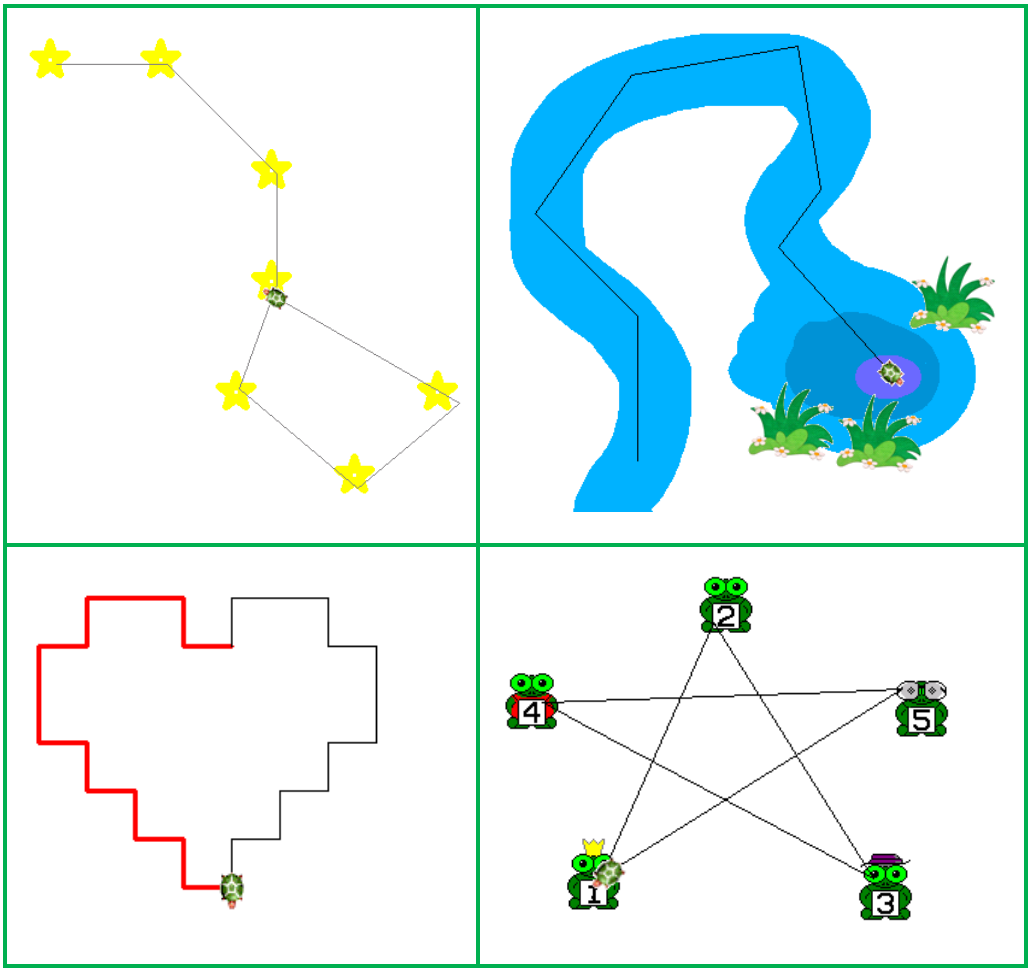
- Žiak si načíta pozadie zo súboru do prázdnej stránky. Pozadie pripraví učiteľ alebo si ho môžu pripraviť aj žiaci v grafickom editore. Pri zmazaní stránky

žiak opäť načíta pozadie zo súboru.

- Učiteľ pripraví projekt, v ktorom je pozadie načítané, prípadne je pripravených viac stránok s rôznymi pozadiami. Môže zadefinovať stránke reakciu na udalosť `priZmazaní`, v ktorej automaticky pri zmazaní načíta pozadie zo súboru.

V prvom prípade cieľom aktivity nie je len programovanie pohybu korytnačky, ale aj osvojovanie si práce vo vývojom prostredí Imagine Loga, čo je užitočné ako príprava na budúcu tvorbu komplexných projektov. Vytvárajú sa tiež väzby medzi tematickými oblasťami v predmete informatika – programovanie, práca s obrázkami, práca so súbormi a priečinkami.

V druhom prípade žiak dostáva *malú didaktickú aplikáciu* so špecifickým cieľom – experimentovať s príkazmi na pohyb korytnačky. Všetky problémy, ktoré nesúvisia s cieľom, sú v projekte vyriešené a žiak sústredí svoju pozornosť len na jednu vec.



Obrázok 3.3: Príklady aktivít pre nácvik pohybu korytnačky: Pospájaj hviezdy, doplávaj do jazierka, dokonči srdce symetricky, navštív žabky podľa poradia.

Editovanie textových príkazov uľahčujú **skratky** – kratší text znamená menšiu pravdepodobnosť preklepu. Skratky majú len často používané príkazy. Menej používané alebo krátke príkazy skratky nemajú.

Príkazy je možné zadávať s **diakritikou**, aj bez nej. Názory na používanie diakritiky v príkazoch pre korytnačku sa rôznia: príkazy bez diakritiky sa ľahšie editujú, používajú sa len základné klávesy klávesnice, ale písanie príkazov (slovenských slov) bez diakritiky môže podporiť nevhodný návyk písania bez diakritiky aj v inom kontexte.

Pri pokusoch s pohybom a otáčaním korytnačky používame pomôcky:

- *pravítko* pre príkazy `dopredu`, `vzad`,
- *terčík* pre príkazy `vľavo`, `vpravo`.

Príkazy zadávané do príkazového riadka sa neukladajú do projektu. Ako výsledok svojej práce si žiak môže na konci hodiny uložiť obrázok, ktorý nakreslil – uloží pozadie stránky.

Zamyslite sa:

Často počúvame argument, že žiaci nemôžu programovať v Logu, lebo ešte nepreberali na matematike pojem uhol. Nemôže byť informatika a prostredie Loga, v ktorom môžu experimentovať s uhlami a ich veľkosťami, miestom, kde sa prvýkrát stretnú s pojmom uhol? Nemôžu byť skúsenosti s uhlami získané na informatike dobrým východiskom na pochopenie pojmu uhol na matematike?

V publikáciách sa používa diakritika kvôli lepšej čitateľnosti.
Pri písaní treba zvážiť argument ľahšieho editovania.

Aký je váš názor na používanie diakritiky pri písaní príkazov v Logu?

Náročnejšie obrázky na kreslenie rôznymi hrúbkami pera pre trpezlivých – športové piktogramy:

Piktogramy sa používajú ako medzinárodná reč alebo reč adresovaná deťom, ktoré nevedia čítať.

Kreslenie, hrúbky, farby

Nastavenie hrúbky a farby pera korytnačky podstatne rozširuje možnosti kreslenia v Logu. Z čiar rôznych dĺžok, hrúbok a farieb vieme poskladať krásne obrázky, ktoré rozvíjajú estetické cítenie a vkus žiakov.

Pri prvých nastavovaniach hrúbky a farby pera použime pomôcky. V pomôcke pre príkaz `nechHrúbkaPera` žiak vidí, ako bude vyzerať čiara nastavenej hrúbky. Hrúbku pera vyberá myšou z ponuky alebo zadaním inej hrúbky do editovacieho políčka.

V pomôcke pre príkaz `nechFarbaPera` žiak myšou vyberá farbu a posúvačom jej odtieň. V editovacom políčku si všíma meno farby, ktoré sa doplní ako vstup pre príkaz `nechFarbaPera`. Miešanie farieb zo zložiek RGB nie je potrebné, 180 pomenovaných farieb a ich odtieňov je dostatočne veľa.

Zaujímavé obrázky dostaneme, keď farbu pera generujeme náhodne. Použiť môžeme `?` (otáznik) alebo `?prvok`. Náhodný výber z niekoľkých hodnôt operáciou `?prvok` má ťažkú syntax, napr. `nechfp ?prvok [žltá6 žltá7 žltá8]`, nepoužívajme ho na generovanie náhodných vstupov so začiatočníkmi. Náhodný vstup môžeme použiť aj pri iných príkazoch.

Obrázok 3.4: Topografické značky ako námety na jednoduché čiernobiele obrázky nakreslené rôznymi hrúbkami pera

Obrázok 3.5: Odtiene farby a náhodné farby

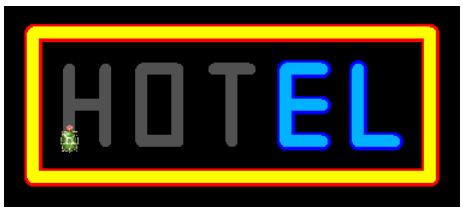
Niektorí žiaci kreslia vždy iný obrázok, ako učiteľ zadal. Odôvodňujú to tým, že „im sa to takto páči“. Môže to byť signál toho, že nevedia splniť zadanie.

Farebné obrázky kreslia žiaci do prázdnej stránky **podľa predlohy** z učebnice alebo z obrazu premietaného učiteľom. Je dôležité, aby vedeli vyriešiť úlohu (nakresliť obrázok) podľa zadania. Učiteľ môže tiež pripraviť pozadia na dokresľovanie. Kreslenie do pozadia umožňuje ľahko overiť správnosť riešenia.

Úloha 3.7

Pripravte aktivitu pre žiakov na dokresľovanie: V neónovej reklame nesvietia niektoré svetlá. Naprogramujte korytnačku, aby ich rozsvietila.

Efekt neónového svetla vytvorte tmavším okrajom čiar.

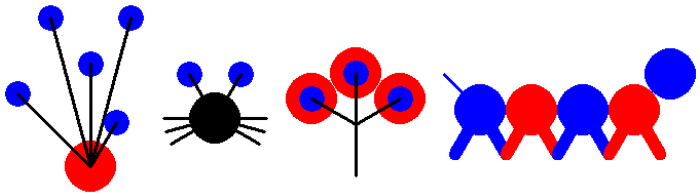


Úlohy na kreslenie podľa predlohy striedajme s **voľným kreslením**, pri ktorom môžu žiaci uplatniť svoju tvorivosť. Voľné kreslenie je relaxačná hrová aktivita – činnosť s nešpecifickým cieľom. Jej cieľom je zbieranie skúseností.

Zadávanie príkazov do príkazového riadka striedajme so zadávaním príkazov pomocou **tlačidiel.** Zadávanie príkazov stláčaním tlačidiel je rýchle, uľahčuje prácu, podporuje experimentovanie. Žiaci si môžu sami vytvoriť tlačidlá pre príkazy, ktoré často používajú, alebo použiť hotovú didaktickú aplikáciu s pripravenými tlačidlami.

Úloha 3.8

Vyskúšajte si prácu s projektom *1pero.imp*. Príkazy sa zadávajú stláčaním tlačidiel. Nakreslite obrázky z učebnice. Nájdite časti obrázkov, ktoré sa opakujú a použite tlačidlá *zapamätaj si* a *znovu vykonaj*. Nakreslite obrázky podľa vlastnej fantázie.



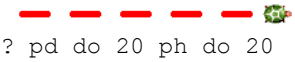
Projekt *1pero.imp* je na CD vloženom do učebnice.

Opakovanie príkazov

Ako propedeutiku príkazu `opakuj` použijeme zoskupovanie dvoch alebo viacerých príkazov buď v príkazovom riadku, ktorý opakovane vyvolávame stláčaním klávesu ŠÍPKA HORE, alebo na tlačidle, ktoré opakovane stláčame.

Námety na jednoduché vzory s opakovaním:

- prerušovaná čiara, bodkovaná čiara, bodkočiarkovaná čiara a iné „pásové“ vzory,
- štvorec, trojuholník, hviezdy, oblúky a iné „kruhové“ vzory.



```
? pd do 20 ph do 20
```

```
? bod 10 do 20
```

```
? bod 10 do 20 pd do 20
ph do 20
```

```
? do 50 vz 50 vl 30
```

```
? do 20 vp 90 do 30 vl
90
```

Príkaz `opakuj` má dva parametre: počet opakovaní a zoznam príkazov, ktoré sa majú opakovať. Skracuje program, teda v skutočnosti sa vykoná počet opakovaní krát viac príkazov, ako sme napísali v tele cyklu.

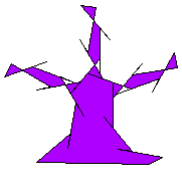
Úloha 3.9

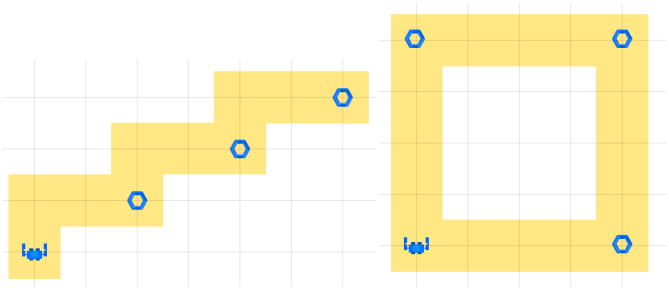
Odpovedzte na otázky:

a) Aká dlhá bodkovaná čiara sa nakreslí príkazom `opakuj 8 [ bod 10 do 20 ]` ?
b) Aká dlhá bodkočiarkovaná čiara sa nakreslí príkazom `opakuj 8 [ bod 10 do 20 pd do 20 ph do 20 ]`?
c) Aké vysoké schody sa nakreslia príkazom `opakuj 3 [ do 20 vp 90 do 30 vl 90 ]` ?
d) Aké široké schody sa nakreslia príkazom `opakuj 3 [ do 20 vp 90 do 30 vl 90 ]` ?
e) Ako by ste so žiakmi overili správnosť odpovedí?

Pri kreslení konkrétnych obrázkov pomocou cyklu treba nájsť v predlohe časti obrázka, ktoré sa opakujú. Okrem časti obrázka je dôležité správne určiť pozíciu a smer korytnačky, ktoré má mať korytnačka, keď sa začína opakovanie - tzv. **invariant**. Invariantné pozície korytnačky spočiatku učiteľ pomáha žiakom hľadať vyznačovaním v obrázku. Napríklad v IzyLogu v aktivitách pre pohyb robota sú vyznačené rozsypanými maticami (obr. 3.6). Matice označujú len invariantnú polohu, nie smer.

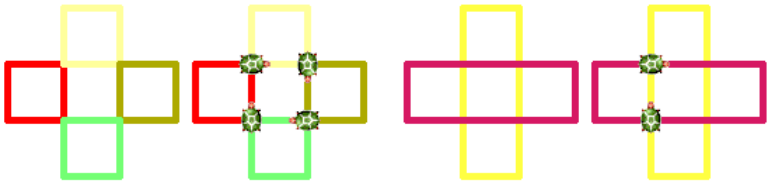
Zaujímavým projektom na experimentovanie s príkazmi v Logu bol *teleprojekt 20 príkazov*. Pozrite si stránku projektu a galériu obrázkov na adrese http://imagine.infovek.sk/projekty/prikaz20/index.php





Obrázok 3.6: Aktivity v IzyLogu s naznačenými invariantnými pozíciami

Nájsť v obrázku opakujúce sa časti žiakom uľahčíme tak, že ich farebne odlíšime. V cykle náhodne meníme farbu. Invariantné pozície korytnačky vyznačíme, keď na začiatku tela cyklu odtlačíme tvar korytnačky. Na obrázku 3.7 sú vyznačené dva spôsoby nakreslenia toho istého obrázka.



Obrázok 3.7: Farebné vyznačenie tela cyklu a invariantné pozície korytnačky

Programovanie v príkazovom riadku striedajme s prácou s hotovými didaktickými aplikáciami. Pestrosť aktivít zvyšuje motiváciu.

Projekty *2karticky.imp* a *2opakuj.imp* sú na CD vloženom do učebnice.

Úloha 3.10	Vyskúšajte projekt *2karticky.imp*. Uložte kartičky do správneho poradia tak, aby sa nakreslil obrázok podľa predlohy.
Úloha 3.11	Vyskúšajte projekt *2opakuj.imp*. Experimentujte s rôznymi postupnosťami príkazov v cykle. Výsledkom voľného experimentovania môžu byť zaujímavé obrázky. Pridajte do projektu tlačidlo, pomocou ktorého si žiak môže nakreslený obrázok uložiť do súboru.

V úlohe 3.11 podľa úrovne svojich programátorských zručností:

- vytvorte buď jednoduché štandardné tlačidlo, alebo tlačidlo podobné tlačidlám v projekte,
- naprogramujte buď ukladanie pozadia do súboru s pevným menom, alebo meno súboru zadávajte v pomôcke.

Vlastné príkazy

Rozšírme slovník korytnačky, „naučme korytnačku nový príkaz“. Písanie vlastného príkazu je náročnejšie ako interaktívne programovanie v príkazovom riadku, lebo žiak nevidí okamžitú reakciu na príkazy, ktoré napíše. Má však už skúsenosti s písaním viacerých príkazov do riadka a s cyklami, pri ktorých tiež vidí vykonanie celého riadka naraz.

Prvé podprogramy píšeme jednoduché s niekoľkými príkazmi s využitím čiar a bodov. Užitočnosť nového príkazu demonštrujeme v zadaniach, v ktorých je motivácia viackrát kresliť ten istý obrázok na rôznych miestach.

Pri tvorbe zadaní pre žiakov si zvoľme nejakú motivačnú tému, ktorá poskytuje dostatok námetov na definovanie nových príkazov a ich viacnásobné použitie. Námety tém na písanie vlastných príkazov:

- ovocný sad: jablko, hruška, čerešňa, iné,
- park: listnatý, ihličnatý strom, lavička, kvet, lampa,

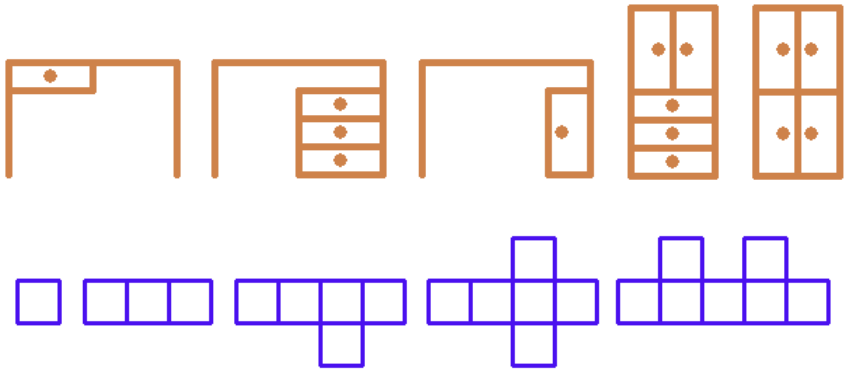
- Vianoce: prskavka, hviezda, snehuliak, zvonček, iné ozdoby,
- mapa pokladu: topografické značky.

Obrázok 3.8: Použitie vlastných príkazov na vhodnom pozadí: Vianoce, mapa pokladu

Iná motivácia pre písanie vlastných príkazov je používať ich ako **diely stavebnice**, ktoré môžeme použiť pri definovaní iných vlastných príkazov. Typickými stavebnými dielmi sú geometrické tvary-mnohouholníky: štvorec, obdĺžnik, trojuholník a iné.

Používaním vlastného príkazu je program kratší, prehľadnejší a jednoduchší. Mnohouholníky nakreslíme pomocou cyklu. Opakované kreslenie mnohouholníkov by preto znamenalo použiť cyklus v cykle. S vlastným príkazom sa program zjednoduší. Žiak rozmýšľa o probléme bez konkrétnych detailov, ktoré rieši podprogram.

V učebnici je veľa príkladov obrázkov, ktoré sa skladajú z mnohouholníkov. Na obrázku 3.9 uvádzame niektoré ďalšie námety.



Obrázok 3.9: Sektorový nábytok poskladaný z dielov a tvary do hry „Lodičky“ poskladané zo štvorcov: granát, puška, delo, lietadlo, krížnik

Úloha 3.12

Nakreslite rad stromov (príkaz `alej`) pomocou vlastného príkazu `strom` definovaného dvomi spôsobmi:

```
viem strom
 nechhp 12
 nechfp "hnedá5
 pd do 50
 nechfp "zelená5
 bod 50 ph
koniec
```

```
viem strom
 nechhp 12
 nechfp "hnedá5
 do 50
 nechfp "zelená5
 bod 50 ph vz 50
koniec
```

Vlastné príkazy editujeme pomocou príkazu `uprav`, skratka `up`.

Možné ***problémy s editovaním*** vlastných príkazov:

Žiak napíše do príkazového riadka hlavičku príkazu `viem` a meno príkazu. Príkazový riadok sa zmení na riadkový editor na zaznamenanie tela príkazu. Výzva na začiatku riadka sa zmení z `?` na `>`. Príkazy sa nevykonávajú, ale ukladajú do pamäte.

Riešenie
Editovanie príkazu ukončime posledným slovom v definícii nového príkazu: `koniec`. Potom otvorme editor príkazom `uprav` a dokončime program.

Žiak naprogramuje vlastný príkaz, ale nič sa nevykreslí.

Riešenie
Vysvetlíme, že korytnačka nakreslí obrázok, až keď zadáme meno príkazu v príkazovom riadku.

Možné ***problémy s používaním*** vlastných príkazov:

Žiak použije vlastný príkaz v inom príkaze, avšak stavebné diely nakreslené vlastným príkazom sú v obrázku rozhádzané alebo sa nevykreslia všetky.

Riešenie
V obrázku, ktorý používame ako stavebný diel, vyznačíme začiatočnú a koncovú pozíciu a smer korytnačky a stav jej pera na konci kreslenia. Žiakom odporučíme, aby pozícia a smer korytnačky na začiatku a na konci boli rovnaké.

Návod na riešenie	V príkaze `alej` použijeme `strom` ako stavebný diel. Všimajme si, v akej pozícii korytnačka začína a končí kreslenie stromu a ako sa mení stav pera v príkaze `strom`.

Vlastné príkazy sa na rozdiel od príkazov zadávaných interaktívne do príkazového riadka ukladajú do projektu. Ukladajú sa tiež pozadie stránky, tlačidlá, korytnačka (prípadne ďalšie objekty v projekte). Ukážeme žiakom, ako uložiť projekt.

Príkazy so vstupmi

Žiaci majú skúsenosti so zadávaním vstupov pre vstavané príkazy Loga. Vedia, že niektoré príkazy treba korytnačke „upresniť“ zadaním nejakej hodnoty (napr. `dopredu` koľko?) a niektoré upresňovať netreba (napr. `znovu`). Majú skúsenosti s písaním vlastných príkazov a sú motivovaní tiež napísať príkaz, ktorý nebude kresliť vždy to isté (podobne ako príkazy so vstupmi, ktoré poznajú).

Najviac skúseností majú žiaci s príkazom `dopredu` a s jeho parametrom počet krokov. Preto prvé príkazy s parametrom píšeme na nakreslenie obrázkov s premennou *veľkosťou*. Parametrizujeme jednoduché vlastné príkazy, ktoré už žiaci poznajú, majú s nimi dostatok skúseností a pociťujú potrebu zadávať im vstupy (napr. štvorec, trojuholník).

Najprv napíšeme príkaz bez parametra. Potom v ňom hľadáme údaj, ktorý treba zmeniť, aby sa vykreslil obrázok inej veľkosti, a nahradíme ho premennou. Premennú zapíšeme tiež do hlavičky za meno vlastného príkazu. Príkaz vyskúšame s rôznymi vstupmi. Príkaz použijeme s rôznymi vstupmi v zmysluplných úlohách. Na obrázku 3.10 je použitý príkaz `trojuholník` s premennou veľkosťou v rôznych obrázkoch.

```
viem trojuholník :vel
 opakuj 3 [
  do :vel vp 120
 ]
koniec

? trojuholník 40
```

```
viem figúrka :farba
 nechfp :farba
 pd
 vp 30
 trojuholník 40
 do 40 vl 30
 ph
 do 15 bod 30 vz 15
 vp 30 vz 40 vl 30
koniec

? figúrka "červená
```

```
viem radStromov :počet
 opakuj :počet [
  strom
  vp 90 do 60 vl 90
 ]
koniec

? radStromov 5
```

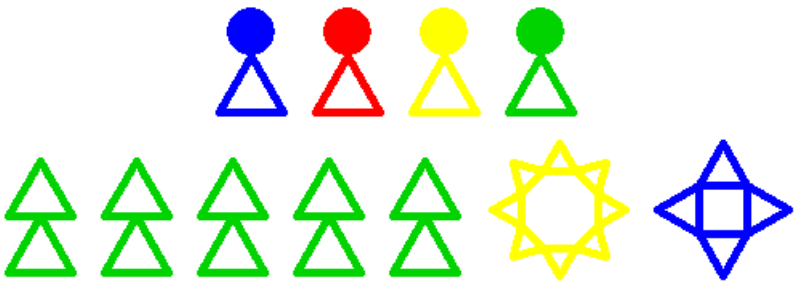
```
viem slnko :počet
 opakuj :počet [
  trojuholník 20
  do 20
  vl 360/:počet
 ]
koniec

? slnko 8
```

Možné *problémy so syntaxou* vlastných príkazov so vstupmi:

- žiak vynechá dvojbodku pred menom premennej (`do vel`),
- žiak napíše medzi dvojbodku a meno premennej medzeru (`do : vel`),
- žiak vynechá medzeru medzi príkazom a premennou (`do:vel`),
- žiak neuvedie parameter v hlavičke za menom príkazu.

Obrázok 3.10: Obrázky využívajú príkaz `trojuholník` s premennou veľkosťou

Ďalšími prirodzenými parametrami obrázkov sú *farba* a *počet* (obr. 3.11).



Obrázok 3.11: Hracie figúrky s premennou farbou a premenný počet stromov v rade a trojuholníkov v kruhu (slnko, kompas)

Pri písaní príkazov s parametrami začíname s jednoduchými úlohami, v ktorých treba premennou nahradiť jeden údaj. V zložitejších úlohách treba nahradiť údaje na viacerých miestach v programe alebo použiť premennú vo výraze (napríklad ako v príkaze `slnko` na okraji strany).

Úloha 3.13

Napíšte príkaz `kniha` s parametrom `:výška`, ktorý nakreslí čiaru náhodnej farby s hrúbkou 10 a premennou výškou. Začiatočná a koncová pozícia korytnačky nech je rovnaká. Nakreslite policu s knihami náhodnej výšky:

Riešenie

Napíšeme príkaz `kniha`:

```
viem kniha :výška
 nechfp ?
 pd
 do :výška
 vz :výška
 ph
koniec
```

Vyskúšame kreslenie kníh rôznych výšok:

```
? kniha 30
? kniha 50
? kniha ?
? kniha ?prvok [30 40 50]
```

Príkaz použijeme na nakreslenie police:

```
viem polica
 opakuj 20 [
  kniha ?prvok [30 40 50]
  vp 90 do 10 vl 90
 ]
 vl 90
 nechfp "hnedá
 pd do 210 ph
 vl 90
koniec
```

Žiaci sú zvyknutí zadávať náhodné vstupy základným príkazom Loga a budú ich zadávať aj vlastným príkazom.

Možné *problémy so zadávaním náhodných vstupov* pomocou `?`:

- `?` generuje vstupy, ktoré nám nevyhovujú (príliš veľké alebo malé číslo, nevhodná farba),
- vlastný príkaz nefunguje, ako sme očakávali: ak sa premenná vyskytuje na viacerých miestach v programe, jej hodnota sa generuje pri každom výskyte.

Riešenie

Namiesto `?` použijeme operáciu `?prvok`. V jej vstupnom zozname vymenujme hodnoty, z ktorých sa má náhodne vyberať (vylúčime tým nevhodné vstupy). Hodnota zo zoznamu sa náhodne vygeneruje a dosadí sa ako vstup nášho príkazu. V príkaze sa už hodnota vstupnej premennej viackrát negeneruje.

Čo sme sa naučili

Systemizovali sme svoje vedomosti o korytnačom kreslení.

Zoznámili sme sa s aktivitami na propedeutiku (úvod do) korytnačieho pohybu.

Analyzovali sme obsah tematického celku Korytnačia grafika.

Zoznámili sme sa s metodickými postupmi na vyučovanie korytnačieho pohybu, kreslenia, opakovania príkazov a písania a používania vlastných príkazov.

Vieme navrhovať aktivity pre žiakov, pripravovať alebo upravovať malé didaktické aplikácie.

Zoznámili sme sa s možnými problémami pri vyučovaní v tematickom celku Korytnačia grafika a ich riešeniami.

4 Objekty a udalosti

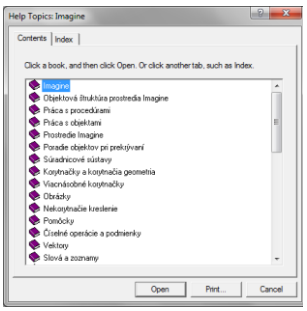
Keď chce cukrár z cesta vykrojiť koláčik v tvare srdiečka, použije príslušnú formičku.

Keď chce programátor v projekte použiť objekt nejakého typu, použije príslušnú triedu.

Imagine Logo je moderné programovacie prostredie, ktoré podporuje *objektovo-orientované a udalosťami riadené programovanie*. Ide o dôležité programovacie paradigmy uplatňované v profesionálnej programátorskej praxi. Už v rámci vyučovania informatiky v nižšom sekundárnom vzdelávaní môžeme názorne a atraktívne realizovať propedeutiku týchto tém.

Prvé stretnutie s objektmi žiaci nevedome zažijú hneď pri prvom kontakte s prostredím Imagine Logo. Vidia a používajú objekt `k1` triedy `Korytnačka`, ktorý je súčasťou objektu `Stránka1` triedy `Stránka`. Triedy `Korytnačka` a `Stránka` pripravili autori programovacieho prostredia Imagine Logo. Trieda `Korytnačka` obsahuje definíciu základných vlastností (nastavení) a správania, ktoré majú všetky korytnačky. Analogicky, trieda `Stránka` obsahuje definíciu základných vlastností (nastavení) a správania, ktoré majú všetky stránky. Vlastnosti objektov sú v pamäti reprezentované premennými, správanie príkazmi, ktorým objekty rozumejú a vedia ich vykonať. So správaním objektov súvisia aj udalosti, na ktoré sú schopné reagovať.

V objektovo-orientovanom programovaní chápeme triedu ako vzor na vytváranie objektov. Žiaci na základnej škole samozrejme neprogramujú vlastné triedy. Pracujú priamo *s konkrétnymi objektmi* (predovšetkým s korytnačkami, tlačidlami a stránkami). Vlastnosti objektov nastavujú príkazmi alebo prostredníctvom kariet dialógového okna *Zmeň*. Pre objekty definujú reakcie na udalosti.



Úloha 4.1

V *Pomocníkovi* (F1) vyhľadajte informácie o triede `Korytnačka`. Zistite, koľko základných nastavení táto trieda obsahuje.

Ktoré zo základných nastavení korytnačky už poznáte?

V triede `Korytnačka` sú definované tieto základné nastavenia:

- aktívnaFarba; nechAktívnaFarba
- aPoz; nechAPoz
- autoAnimovanie; nechAutoAnimovanie
- autoŤahanie; nechAutoŤahanie
- domovskýStav; nechDomovskýStav
- farbaPera; nechFarbaPera
- farbaVýplne; nechFarbaVýplne
- fáza; nechFáza
- hrúbkaPera; nechHrúbkaPera
- meno
- oblasť; nechOblasť
- oblasťKdeMaVidno; nechOblasťKdeVidno
- pero; nechPero
- písmo; nechPísmo
- ponukaKlávesov; nechPonukaKlávesov
- poz; nechPoz
- poznámka; nechPoznámka
- reagujeAjNaPriesvitné; nechReagujeAjNaPriesvitné
- reagujeNaZrážku; nechReagujeNaZrážku
- rýchlosťAnimácie; nechRýchlosťAnimácie
- smer; nechSmer
- tvar; nechTvar
- typOblasti; nechTypOblasti
- určovanieZáberu; nechUrčovanieZáberu
- vidno; nechVidno
- vzorPera; nechVzorPera
- vzorVýplne; nechVzorVýplne
- xSúr; nechXSúr
- ySúr; nechYSúr
- záber; nechZáber
- zamknutie; nechZamknutie
- zväčšenie; nechZväčšenie

Obrázok 4.1: Prehľad základných nastavení v triede Korytnačka v Pomocníkovi prostredia Imagine Logo

Všimnime si, že ku každému nastaveniu máme k dispozícii dva príkazy:

- **operáciu** na zistenie hodnoty nastavenia (napr. `farbaPera`, `písmo`, `poz`, `smer`, `tvar`, `xSúr`, `záber`, `zväčšenie` a pod.)
- **procedúru** na nastavenie novej hodnoty (napr. `nechFarbaPera`, `nechPísmo`, `nechPoz`, `nechSmer`, `nechTvar`, `nechXSúr`, `nechZáber` a pod.)

Úloha 4.2	Nastavte korytnačke farbu pera, hrúbku pera, automatické ťahanie, pero a viditeľnosť v okne *Zmeň* (v rodnom liste korytnačky). Zapíšte príkazy, ktorými by ste zmenili jednotlivé nastavenia korytnačky bez použitia rodného listu.

Riešenie	Jedným z možných riešení je postupnosť príkazov:
	
	`? nechFp "modrá`
	`? nechHp 5`
	`? nechAutoŤahanie "nie`
	`? nechPero "ph`
	`? nechVidno "nie`
	
	V prípade pera a viditeľnosti sme zvyknutí skôr na dvojice príkazov `ph` a `pd`, resp. `skry` a `ukáž`.

Aktuálne hodnoty nastavení môžeme zistiť aj tak, že ich vypíšeme do *Plochy výpisov*.

Výpis dosiahneme príkazom `zobraz` (skrátene `zo`):

```
? zo pero
ph
? zo fp
čierna
? zo hp
1
? zo autoŤahanie
nie
? zo poz
[260 -326]
? zo smer
0

?
```

Okrem objektov triedy `Korytnačka` používame v projektoch aj objekty iných tried (tlačidlá, texty, posúvače, stránky, papiere a i.). Voláme ich aj *súčiastky*, pretože z nich skladáme (zostavujeme) prostredie projektu.

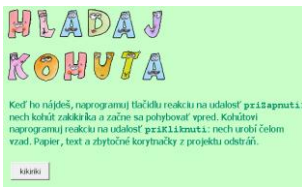
Aké vizuálne komponenty ste používali pri tvorbe aplikácií vo vývojovom prostredí *Lazarus/Delphi*?

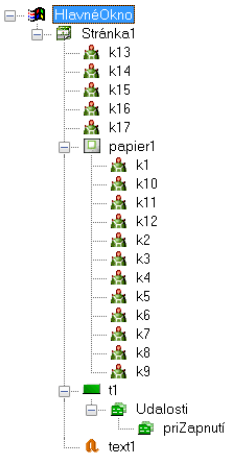
Obrázok 4.2: Hlavný panel s tlačidlami na vkladanie nových korytnačiek a ďalších súčiastok

Úloha 4.3	Otvorte projekt *kikiriki.imp*. Zistite, koľko objektov obsahuje. Nájdite korytnačku s tvarom kohúta a dokončite projekt podľa pokynov na stránke.

Riešenie	Po otvorení projektu vidíme na stránke *text* a *tlačidlo*. Ľahko overíme predpoklad, že písmenká z nápisu sú tiež *korytnačky*. Žiadneho kohúta ale nevidno. Kompletnú informáciu o štruktúre projektu získame zobrazením okna *Pamäť* (F4).
	
	Ak si uvedomíme, že 12 korytnačiek s tvarom písmen sa nachádza na papieri, zostane nám už len 6 podozrivých korytnačiek. Nájdeme tú s tvarom kohúta a zmeníme jej nastavenie `vidno` na `"áno` (v rodnom liste zaškrtneme políčko vidno). Keďže chceme s korytnačkou-kohútom ďalej pracovať, bude praktické ju premenovať.
	
	Tlačidlu do reakcie na udalosť `priZapnutí` doplníme príkaz, ktorým pre kohúta spustíme proces (motorček):
	
	`priZapnutí: hrajSúbor "cockerel.wav kohuťkaždých 250 [do 2]`
	
	Kohúta môžeme vypátrať aj rýchlejšie ☺. Oslovíme všetky korytnačky, ktoré stránka (pozor, nie papier!) obsahuje:
	`? pre všetky [ukáž]`

Projekt *kikiriki.imp* nájdete v e-learningovom kurze k tomuto modulu.





V reakcii na udalosť `priZmene` prečítame aktuálnu hodnotu nastavenú na posúvači operáciou `hodnota`.

Na inom mieste programu zistíme hodnotou posúvača `p1` takto:

```
p1'hodnota
```

Niekedy chceme objekt triedy `Text` použiť na zobrazovanie stavu hry (bodového skóre), Konkrétnu hodnotu nastavíme príkazom:

```
text1'nechHodnota 0
```

Zvýšiť hodnotu zobrazenú v `text1` znamená k aktuálnej hodnote, ktorú obsahuje, pripočítať 1:

```
pre "text1 [
  nechHodnota
          hodnota+1
]
```

Úloha 4.4	Do projektu *kikiriki.imp* vložte súčiastku *vodorovný posúvač*. Pomocou posúvača zväčšujte a zmenšujte tvar korytnačky-kohúta.
Riešenie	Veľkosť tvaru korytnačky môžeme zmeniť pomocou základného nastavenia `zväčšenie`. Nájdeme ho v rodnom liste na karte Tvar. Ak má hodnotu 1, tvar korytnačky vidíme v skutočnej veľkosti. Hodnotou 0.5 zabezpečíme zmenšenie tvaru o polovicu, hodnotou 2 dvojnásobné zväčšenie a pod. Nastavme na posúvači `minimum` na 1, `maximum` na 10. Posúvaču naprogramujme reakciu na udalosť `priZmene`: priZmene: kohut'nechZväčšenie hodnota / 5 Aké hodnoty získame vyhodnotením výrazu `hodnota/5` ?
Úloha 4.5	V projekte *kikiriki.imp* sa ešte chvíľu zdržíme. Preskúmajte rodné listy objektov `text1`, `papier1`, `t1` a `Stránka1`. Čo všetko im môžeme v rodnom liste nastaviť? Na ktoré udalosti sú tieto objekty schopné reagovať?

Projekty s viacerými stránkami

Viac stránok sa nám môže zísť napr. pri príprave série zadaní pre žiakov. Žiaci sa medzi stránkami prepínajú pomocou tlačidiel na *Hlavnom paneli*. Alebo im pripravíme tlačidlá priamo na stránkach. Prepnutie na nasledujúcu, predchádzajúcu, resp. ktorúkoľvek z existujúcich stránok dosiahneme príkazom `nechAktívnaStránka`, napr. `nechAktívnaStránka "Stránka2`.

Aj žiaci sa často domáhajú možnosti zachovať si pekný obrázok, ktorý práve pomocou korytnačky nakreslili. Ukážme im preto, ako jednoducho si pridajú do projektu novú stránku a ako na ňu položia novú korytnačku. Projekt s obrázkami zapamätanými na stránkach si môžu na konci hodiny uložiť.

Úloha 4.6 **Diskusia**	Vymyslite viacstránkový projekt s úlohami pre žiaka. O svoj nápad sa podeľte s kolegami. Každá stránka môže obsahovať krátky text s pokynmi pre žiaka, pekné pozadie, korytnačku umiestnenú na vhodnej pozícii, ďalšie potrebné objekty. Môže ísť o sériu na seba nadväzujúcich gradovaných úloh zameraných na trénovanie nových poznatkov alebo o úlohy rôznorodé, nezávislé, pri riešení ktorých si žiaci zopakujú a tvorivo aplikujú vedomosti a zručnosti z viacerých hodín. Úlohy pre žiaka môžeme formulovať v kontexte zaujímavého príbehu, jednoduchej hry, aktuálneho sviatku, ročného či školského obdobia, spoločenského diania, učiva z iného predmetu a pod.

Kopírovanie objektov pomocou schránky

Pri vytváraní väčšieho počtu objektov s rovnakými alebo podobnými nastaveniami a správaním si prácu významne uľahčíme kopírovaním (klonovaním) jedného vopred pripraveného vzorového objektu. Objekty kopírujeme pomocou schránky. Najčastejšie kopírujeme korytnačky:

1. V miestnej ponuke korytnačky-vzoru vyberieme príkaz *Skopíruj do schránky*.
2. Na stránke alebo na papieri (v mieste kde chceme vytvoriť jej kópiu) zvolíme príkaz *Prilep zo schránky*.

Rovnako kopírujeme aj iné typy objektov.

So schránkou ako dočasnou pomocnou pamäťou sa žiaci stretávajú bežne aj v iných aplikáciách, napr. v grafickom či textovom editore.

Úloha 4.7	Otvorte projekt *penazenka.imp*. Preskúmajte nastavenia a správanie pripravenej korytnačky. Vyrobte čo najrýchlejšie celú sadu euro mincí. Zoraďte ich podľa nominálnej hodnoty.
Riešenie	Prvú mincu môžeme 7 krát skopírovať. Každej novej korytnačke nastavíme iný tvar. Zoradíme ich podľa veľkosti. Každej korytnačke nastavíme aktuálnu pozíciu ako domovskú.
Úloha 4.8	Pridajte do projektu tlačidlo, pri zapnutí ktorého sa mince presunú na náhodné miesta na stránke. Jedna z nich (zvolená náhodne) sa okrem toho otočí na opačnú stranu. Druhým tlačidlom zabezpečte upratovanie (mince znovu „nastúpia“ do pôvodného radu, obrátené číselnou hodnotou nahor).

Riešenie	Rozhádzanie mincí po stránke je jednoduché. Oslovíme všetky korytnačky, ktoré máme na stránke a presunieme ich na náhodne zvolené miesto:
	
	`? pre všetky [nechPoz ?]`
	
	V zadaní sa požaduje, aby sa náhodne zvolená minca obrátila na opačnú stranu. Ak by sme chceli, aby to bola niektorá konkrétna minca (alebo skupina mincí), vieme ju (ich) osloviť priamo:
	
	`? pre "k3 [nechZáber záber+1]`
	
	alebo len
	
	`? k3'nechZáber k3'záber+1`
	`? pre [k3 k4 k7] [nechZáber záber+1]`
	
	My chceme obrátiť náhodne zvolenú mincu. Zo zoznamu korytnačiek `všetky` náhodne vyberieme niektorý z prvkov pomocou operácie `?prvok`.
	
	Oslovíme zvolenú korytnačku (necháme sa prekvapiť, ktorá to bude ☺):
	
	`? pre ?prvok všetky [nechZáber záber+1]`

Projekt *penazenka.imp* nájdete v e-learningovom kurze k tomuto modulu.

O tvaroch korytnačiek sa dozviete viac v kapitole 5.

Procedúrou `nechZáber` zmeníme záber tvaru na iný.

Operácia `záber` vráti číslo aktuálneho záberu tvaru korytnačky.

Vyhodnotením výrazu `záber+1` získame číslo nasledujúceho záberu. Za posledným záberom nasleduje znovu prvý.

Úloha 4.9	V projekte *penazenka.imp* otestujte uvedené dva príkazy a vysvetlite rozdiel medzi nimi:
	
	`? pre všetky [nechZáber záber+1 čakaj 500]`
	
	`? preKaždú všetky [nechZáber záber+1 čakaj 500]`

Riešenie	V prvom prípade oslovíme všetky korytnačky súčasne, mince sa obrátia naraz. V druhom prípade tiež oslovíme všetky korytnačky, ale postupne. V zozname `všetky` sú uvedené v tom poradí, v akom sa narodili.
Úloha 4.10	Vyhľadajte v učebnici [3] projekty, v ktorých je možné (a vhodné) uplatniť kopírovanie korytnačiek, príp. tlačidiel.

Prvé stretnutie žiaka s objektmi a udalosťami

S objektmi a udalosťami sa žiaci prvýkrát stretnú pri používaní tlačidiel. Tlačidlo si vytvoria na uľahčenie písania príkazov do príkazového riadka pri kreslení obrázkov. Naučia sa:

- vložiť na stránku nové tlačidlo z panela so súčiastkami,
- vyvolať miestnu ponuku pre vložené tlačidlo kliknutím pravého tlačidla myši a otvoriť dialógové okno *Zmeň*,
- zmeniť nápis na tlačidle v nastavení *Popis*,
- napísať príkaz do udalosti `priZapnutí` na karte *Základné* dialógového okna *Zmeň*,
- zmeniť veľkosť tlačidla pomocou myši,
- presunúť tlačidlo pomocou myši výberom voľby *Presúvaj* v miestnej ponuke tlačidla,
- zrušiť tlačidlo výberom voľby *Zruš* v miestnej ponuke tlačidla.

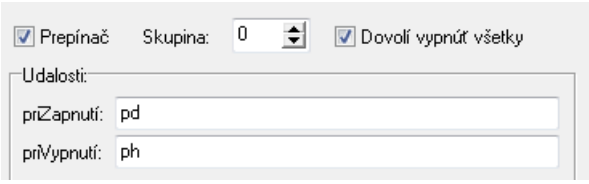
Stlačením tlačidla sa vykoná príkaz zapísaný ako reakcia na udalosť `priZapnutí`. Zadávaním príkazov pomocou tlačidiel je ovládanie korytnačky jednoduchšie a rýchlejšie.

Námety na jednoduché projekty obsahujúce tlačidlo:

- tlačidlo na zmazanie stránky,
- tlačidlá na nastavenie konkrétnej / náhodnej farby alebo hrúbky pera
  - kreslenie hracích plánov, vzorov,
- tlačidlo ako ovládač na riadenie pohybu korytnačky.

Projekt *auticko.imp* nájdete v e-learningovom kurze k tomuto modulu.

Úloha 4.11	V projekte *auticko.imp* nájdete štvorcovú sieť, v strede jedného políčka siete je umiestnená korytnačka-autíčko. Dokončite projekt tak, aby bolo možné pohyb autíčka v štvorcovej sieti ovládať tlačidlami: *Choď, Cúvaj, Odboč vpravo, Odboč vľavo, Späť do garáže*.
Riešenie	Rozmiestnenie objektov na stránke treba premyslieť, dbať pri tvorbe projektu aj na estetické kritériá: vhodný nápis na tlačidle, vhodné rozmery a umiestnenie tlačidla na stránke.

Úloha 4.12	Vložte do projektu *auticko.imp* ďalšie tlačidlo. Bude slúžiť na zapínanie a vypínanie záznamu prejdenej dráhy. Tlačidlo naprogramujte ako prepínač.
Riešenie	Zvolením políčka *Prepínač* na karte Základné rodného listu tlačidla zobrazíme aj riadok pre udalosť `priVypnutí`: 

Rozmery tlačidla vieme nastaviť v jeho rodnom liste. Veľkosť tlačidla však rýchlejšie prispôsobíme pomocou myši: ukážeme na jeho pravý dolný roh, stlačíme kláves CTRL a ťaháme myšou so stlačeným pravým tlačidlom myši v potrebnom smere.

Udalosti korytnačky priKliknutí a priŤahaní

Skúsenosti s používaním tlačidiel žiaci využijú pri práci s objektom korytnačka. Za základné udalosti korytnačky považujeme udalosti `priKliknutí` a `priŤahaní`. Kliknutie aj ťahanie sú činnosti, ktoré sú žiakovi zrozumiteľné, nie je potrebné ich vysvetľovať. V úlohách, ktoré so žiakmi riešime, je reakciou na udalosť `priKliknutí`:

- nakreslenie jednoduchého, farebného čiarového obrázka (s použitím príkazov `do`, `vz`, `vp`, `vl`, `bod`, `opakuj` a vlastných príkazov),
- posunutie korytnačky na iné miesto (v zvislom alebo vodorovnom smere, o pevnú alebo náhodnú dĺžku, s otočením o pevný alebo náhodný uhol)
  - neskôr môžeme presunutie realizovať s použitím príkazov `nechPoz`, `nechXSúr`, `nechYSúr`,
- odtlačenie (opečiatkovanie) obrázku na stránke
  - obrázku zo súboru príkazom `odtlačObrázok`,
  - neskôr aktuálneho tvaru korytnačky príkazom `odtlač`.

Tlačidlami vieme pripraviť pre korytnačku viac činností - každé tlačidlo vyvoláva inú reakciu korytnačky. V udalosti korytnačky `priKliknutí` môžeme definovať len jednu reakciu. To môže byť motiváciou pridať na stránku viac korytnačiek (rovnakým spôsobom ako tlačidlá) a každej naprogramovať inú reakciu na udalosť `priKliknutí`.

Možné ***problémy s editovaním*** reakcií na udalosti:

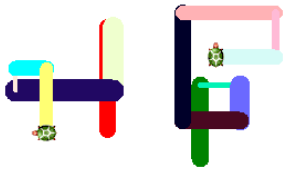
Reakcie na udalosti sa vkladajú do riadka vedľa mena udalosti v rodnom liste korytnačky. Riadok je krátky a písmo malé a proporcionálne. Keď chcú žiaci napísať dlhší program, ktorý presahuje dĺžku riadka, program je neprehľadný a zle čitateľný.

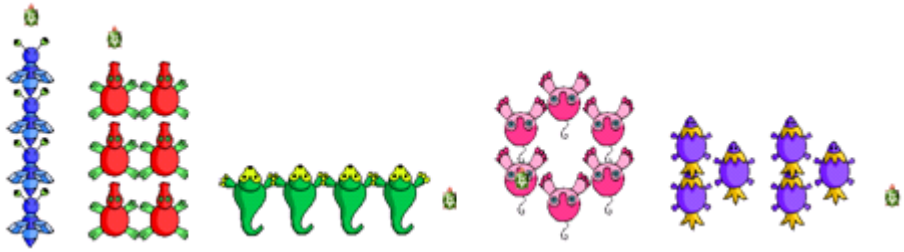
Riešenie

Naprogramujeme vlastný príkaz, ktorý realizuje časť alebo celú reakciu na udalosť. Zápis v rodnom liste sa tým skráti.

Úloha 4.13	Konkretizujte niektorý z vyššie uvedených všeobecných námetov vo forme úloh pre žiaka. Pracujte vo dvojiciach. Riešenia vymyslených úloh aj naprogramujte.
Príklad riešenia	Potvorky z rôznych krajín prišli na olympiádu a nacvičujú na otvárací ceremoniál. Potrebujú vedieť, kde má ktorá z nich stáť. Vložte na stránku viac korytnačiek, ktoré pomôžu potvorkám zoradiť sa. Naprogramujte každej korytnačke reakciu na udalosť `priKliknutí` a klikaním na ne zoraďte výpravy potvoriek do rovnakých alebo rôznych útvarov (odtláčajte obrázok potvorky príkazom `odtlačObrázok`).

Akými príkazmi reaguje korytnačka na kliknutie myšou, keď po viacnásobnom kliknutí kreslí takéto obrázky?





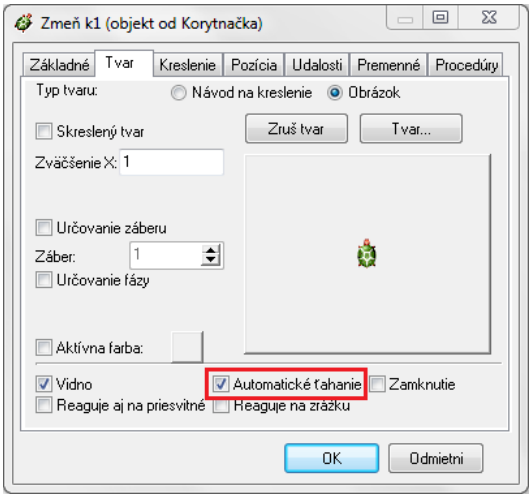
Obrázok 4.3: Rôzne zoradenia výprav potvoriek na olympiáde

Práca s viacerými korytnačkami ako s objektmi, ktoré reagujú na udalosti, je pre žiakov prirodzená. Keď sa však vrátime k príkazovému riadku, vznikne problém, ktorej korytnačke sú určené príkazy v riadku.

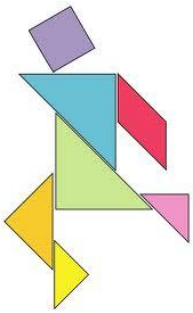
Pozrite si v učebnici, ako je motivované oslovovanie korytnačiek.

Príkazy v príkazovom riadku sú určené pre prvú aktívnu korytnačku. Ak chceme osloviť inú korytnačku, musíme uviesť jej meno pred príkazom alebo ju osloviť príkazom `pre`. Rovnako môžeme zadávať príkazy pre korytnačku v rodnom liste inej korytnačky alebo na tlačidle.

Programovaniu reakcie na udalosť `priŤahaní` by mala predchádzať skúsenosť so zapínaním *automatického ťahania*.



Obrázok 4.4: Zapnutie automatického ťahania v rodnom liste korytnačky



Automatické ťahanie potrebujeme povoliť, keď

- chceme korytnačku používať ako štetec či ceruzku
  - namaľovať ňou na stránke písaným písmom svoje meno, zahrať sa na pána učiteľa a opraviť chyby v diktáte, dokončiť osemsmerovku, naznačiť cestu z bludiska a pod.
- v pripravenom projekte s viacerými korytnačkami rôznych tvarov máme
  - zoradiť korytnačky-kartičky do správnej postupnosti,
  - zostaviť korytnačky-časti skladačky do jedného celku,
  - ťahať korytnačky je v tomto prípade oveľa praktickejšie ako presúvať ich iným spôsobom, žiaci môžu projekt opraviť a zahrať sa.

Pri programovaní reakcie na udalosť `priŤahaní` žiakov zaujme plynulé kreslenie (čarovanie) obrázkov. Reakcia na udalosť `priŤahaní` sa nevykoná len raz ako pri klikaní na korytnačku, ale vykonáva sa plynulo veľakrát, až kým neprestaneme korytnačku ťahať. Udalosť `priŤahaní` sa nenachádza na základnej karte dialógového okna *Zmeň* korytnačky, žiaci ju musia pridať na karte *Udalosti*.

Čarovanie obrázkov pri ťahaní korytnačky si môžu žiaci najprv vyskúšať na hotovom projekte.

V reakciách na udalosť `priKliknutí` sa korytnačka väčšinou presunula na inú pozíciu, aby pri opätovnom kliknutí nekreslila na to isté miesto.

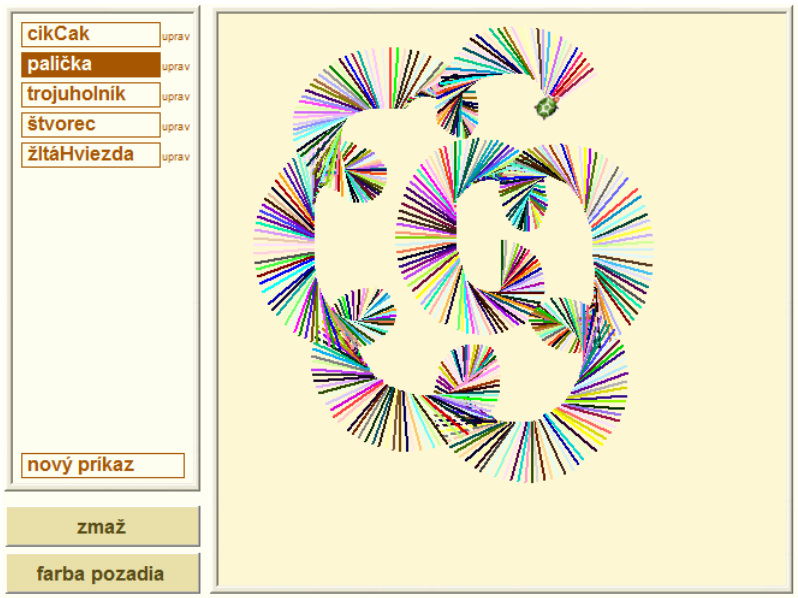
V reakcii na udalosť `priŤahaní` nemá zmysel korytnačku presúvať, ťaháme ju myšou.

Úloha 4.14

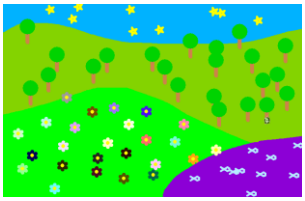
Otvorte projekt *4tahanie.imp*. Experimentuje s ním:

- vyskúšajte všetky pripravené príkazy,
- upravte niektorý z pripravených príkazov ,
- pridajte nový príkaz, pomocou ktorého budete vytvárať reťaz z rôznofarebných mašličiek,
- navrhnite a pridajte do projektu vlastný príkaz,
- porovnajte projekty *4tahanie.imp* a *5vzory.imp*.

Projekty *4tahanie.imp* a *5vzory.imp* sú na CD vloženom do učebnice.



Obrázok 4.5: Pohľad na prostredie projektu 4tahanie.imp z učebnice [3]



Zmeny pozície korytnačky pri ťahaní sú dobrou príležitosťou na použitie podmieneného príkazu. Ak korytnačku ťaháme po farebnom pozadí, mení sa tiež farba bodu, na ktorom sa aktuálne nachádza. Podľa meniacej sa farby pozadia alebo pozície korytnačky môžeme pri ťahaní vykonávať rôzne príkazy podmienene.

Ďalšie korytnačie udalosti a ďalšie objekty

Úloha 4.15

Nájdite v učebnici [3] projekty, v ktorých sa korytnačke programuje reakcia na udalosť `priĽavomDolu`, `priĽavomHore` alebo `priPravomDolu`.

Udalosti `priĽavomDolu` a `priĽavomHore` nastanú na začiatku ťahania, resp. jeho konci (pri pustení ťahanej korytnačky). Pred programovaním projektov s týmito udalosťami je vhodné, aby žiaci najprv fyzicky demonštrovali a slovne opísali, ako pomocou myši realizujú ťahanie.

Ďalšími objektmi, ktoré môžu obohatiť žiacke projekty, sú stránky a texty. Žiaci ich pridávajú do projektu z *Panela súčiastok* podobne ako tlačidlá a korytnačky. Vloženie a editovanie textu pripomína vkladanie textu v grafickom editore. Vloženie novej stránky je jednoduché – stlačí sa tlačidlo *Nová stránka*.

Na základnej škole stačí, keď žiaci používajú stránky a texty pasívne, teda neprogramujú reakcie na udalosti a nepracujú s procedúrami stránok a textov.

Práca s textom:
- vložiť z *Panela súčiastok*,
- formátovať písmo v *Paneli písma*,
- editovať dvojklikom,
- presúvať výberom voľby *Presúvaj* z miestnej ponuky.

Práca so stránkou:
- vložiť z *Panela nástrojov*,
- prepínať prepínačmi.

Možné ***problémy s vkladaním textov***:

Žiak vloží na stránku súčiastku text a zatvorí *Panel písma* ešte pred vložením textu. Hodnota textu zostane prázdna a prázdny text sa na stránke „stratí“.

Riešenie:

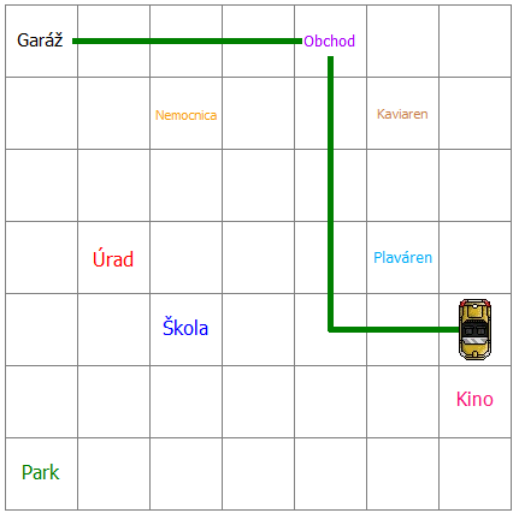
Text vyhľadáme v okne *Pamäť* (F4). Otvoríme dialógové okno *Zmeň* a zmeníme farbu pozadia. Text sa na stránke zviditeľní a môžeme ho editovať dvojklikom.

Obrázok 4.6: Tlačidlá na pridávanie textov, stránok a prepínače stránok

Úloha 4.16

Navhnite ďalšie rozšírenie projektu o autíčku. Vymyslite pravidlá, ktoré musíme pri navigovaní autíčka po hracej ploche dodržiavať.

Uvedieme jeden z možných námetov na rozšírenie: Do políčok štvorcovej siete môžu žiaci povkladať farebné texty symbolizujúce rôzne miesta v meste (obr. 4.7). Do projektu žiaci doplnia vlastné príkazy, ktorými autíčku rozkážu ísť najkratšou možnou cestou z políčka `Garáž` na políčko `Obchod`, `Škola`, `Kino`, `Park` a pod. Tiež môžu programovať príkazy `pondelok`, `utorok` atď. Každý deň podniknú s autíčkom inú okružnú jazdu.



Obrázok 4.7: Texty ako objekty v meste

Čo sme sa naučili

Sumarizovali sme svoje vedomosti o objektovo-orientovanom a udalosťami riadenom programovaní v Logu.

Analyzovali sme obsah vyučovania témy Objekty a udalosti na základnej škole: tlačidlo a jeho udalosť priZapnutí, priVypnutí, korytnačka a jej udalosti priKliknutí, priŤahaní, priĽavomHore, priĽavomDolu, priPravomDolu, viac korytnačiek v projekte a ich oslovovanie, objekty text a stránka v žiackych projektoch.

Vyskúšali sme si prácu s didaktickými aplikáciami na experimentovanie s ťahaním korytnačky.

Zoznámili sme sa s motiváciou na zavedenie príkazu s podmienkou.

Zoznámili sme sa s metodickými postupmi vyučovania témy Objekty a udalosti a s možnými problémami.

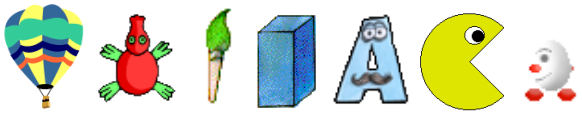
5 Tvary, animované tvary, procesy

Korytnačka v Imagine Logu nie je len personifikáciou grafického pera. V projektoch môže plniť rôzne iné úlohy, v ktorých vystupuje s rôznymi tvarmi. So zdvihnutým perom môže predstavovať pohybujúci sa objekt v príbehu alebo statický objekt čakajúci na nejaký podnet a pod.

Štandardný tvar grafického pera je zelená korytnačka.

Úloha 5.1

Zmenou tvaru pripravíme korytnačku na plnenie rôznych úloh. Uvažujte, v akom projekte by sa dali využiť nasledujúce tvary korytnačiek.



Tvary korytnačiek sú uložené v súboroch s príponou *lgf*.

Tvar môžeme zmeniť buď v rodnom liste korytnačky na karte *Základné* alebo *Tvar*, alebo príkazom `nechTvar menoObrázka`, napríklad `nechTvar "dusan`.

Meno obrázka v príkaze `nechTvar` začína úvodzovkami:

```
nechTvar "andula
```

Ak je v mene špeciálny znak (napr. medzera alebo `\`), uzavrieme ho medzi dvojicu znakov `||`.

```
nechTvar "|modre kocky.lgf|
```

```
nechTvar "|C:\Documents and Settings\Project\My Documents\My Pictures\robot.lgf|
```

Ak sa chceme vrátiť k základnému tvaru, použijeme príkaz

```
nechTvar []
```

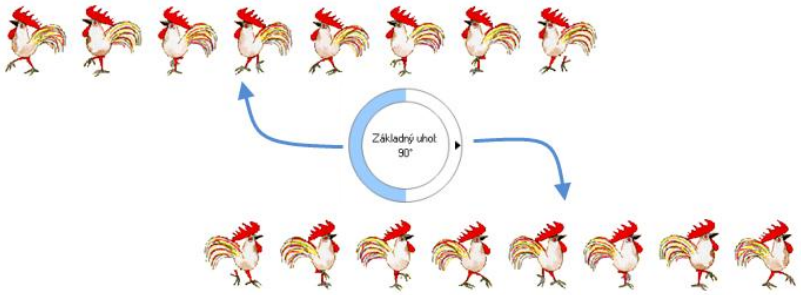
Meniť tvar pomocou rodného listu korytnačky si vyžaduje len základné zručnosti (v rodnom liste zvolíme ponuku *Tvar*, vyberieme vhodný tvar a potvrdíme tlačidlom OK).

Pri zmene tvaru pomocou príkazu je potrebné vedieť, že základný priečinok, v ktorom Imagine „hľadá“ tvary, je *..Imagine\Obrazky*. Ak sa obrázok nachádza v inom priečinku, musíme zadať cestu k nemu v štruktúre priečinkov.

Tvar korytnačky nemusí byť len jednoduchý obrázok. Môže obsahovať postupnosť viacerých obrázkov – **záberov**, ktoré korytnačka automaticky mení pri otáčaní sa do rôznych smerov. Každý záber reprezentuje tvar korytnačky pre istý interval natočenia korytnačky. Napríklad základný tvar zelenej korytnačky má 24 záberov. Každý z nich predstavuje tvar korytnačky v 15-stupňových intervaloch natočenia.

Tvar môže byť aj animovaný obrázok. Vtedy sa korytnačka „pohybuje“, aj keď nemení svoju polohu ani smer. Animácia vzniká usporiadaným prehrávaním statických obrázkov. Ak je frekvencia prehrávania vyššia ako 24 obrázkov za sekundu, ľudské oko ho už eviduje ako súvislý pohyb, v skutočnosti však ide len o dojem. Každý záber tvaru korytnačky môže obsahovať sériu obrázkov – **fáz**, ich cyklickým prehrávaním vzniká animácia.

Zábery základného tvaru korytnačky:



Obrázok 5.1: Zábery animovaného tvaru kohut kraca.lgf, každý záber má osem fáz

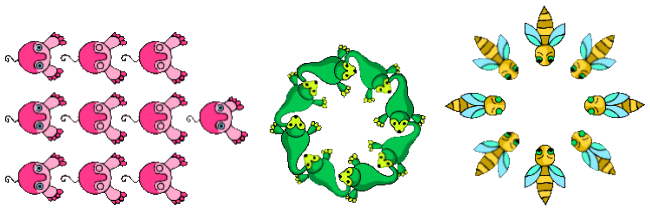
Úloha 5.2

Nájdite medzi obrázkami v priečinku *Obrazky* Imagine Loga tvary korytnačky:

- animovaný, ktorý nemení svoj tvar pri otáčaní,
- animovaný, ktorý mení svoj tvar pri otáčaní,
- statický, ktorý nemení svoj tvar pri otáčaní,
- statický, ktorý mení svoj tvar pri otáčaní .

Ak má obrázok viac záberov a fáz, príkazom `odtlačObrázok` sa odtlačí prvá fáza prvého záberu obrázka.

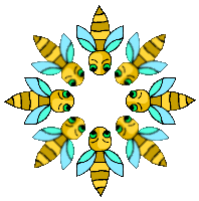
Príkaz `odtlač` odtlačí tvar korytnačky v tom zábere a fáze, v akých sa nachádzala pri odtláčaní.

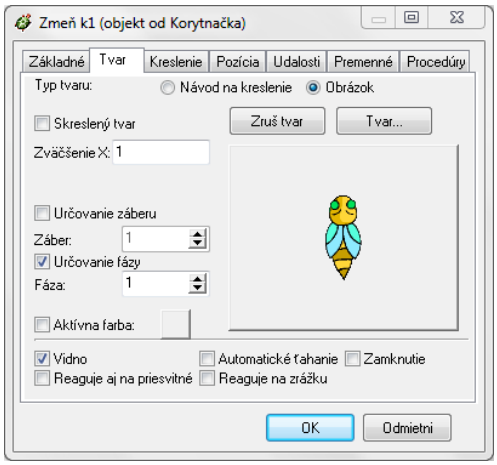
Úloha 5.3	Upravte riešenie úlohy 4.13 o olympiáde potvoriek zo str. 23-24 tak, aby sa potvorky zoradili takto: 
Riešenie	Korytnačku „oblečieme“ do dresu výpravy, ktorej pomáha rozostaviť sa - tvar zmeníme na tvar potvorky. Odtláčame tvar korytnačky v rôznych smeroch príkazom `odtlač`.

Zmeny záberu a fázy tvaru korytnačky môžu byť automatické alebo ich nastavujeme v rodnom liste korytnačky alebo príkazmi v programoch. Určovanie záberov a fáz zapneme v rodnom liste korytnačky na karte *Tvar* (obr. 5.2). Označením zaškrtávacieho políčka zároveň vypneme automatické zmeny. Číselníkom nastavíme číslo záberu alebo fázy.

Nastavením pevnej fázy tvaru korytnačky odtlačte potvorky takto:

alebo takto:





Obrázok 5.2: Určovanie fázy tvaru korytnačky vypne automatické prehrávanie fáz

V programoch nastavujeme režim zmien fáz a záberov príkazmi:

```
nechAutoAnimovanie "áno      nechUrčovanieZáberu "áno
nechAutoAnimovanie "nie      nechUrčovanieZáberu "nie
```

Príkazy na zmenu fázy:

```
nechFáza 1
nechFáza ?
```

Príkazy na zmenu záberu:

```
nechZáber 2
nechZáber ?
```

Číslo fázy a záberu zisťujeme operáciami `fáza` a `záber`:

```
zobraz fáza                zobraz záber
nechFáza fáza+1            nechZáber záber+1
```

Ďalšou vlastnosťou korytnačky, ktorá súvisí s tvarom, je zväčšenie. Nastaviť sa dá v rodnom liste na karte *Tvar* (obr. 5.2). V programoch príkazom `nechZväčšenie` nastavíme zväčšenie a operáciou `zväčšenie` zistíme jeho hodnotu.

Okrem korytnačky môžeme nastaviť obrázkový tvar aj tlačidlu v jeho rodnom liste na karte *Vzhľad* (obr. 5.3). Tvar tlačidla môže mať tri zábery: obrázok vypnutého tlačidla, obrázok vypnutého tlačidla, keď sa nad ním nachádza kurzor myši, a obrázok zapnutého tlačidla.

Obrázok 5.3: Tri zábery obrázkového tlačidla

Úloha 5.4

V projekte *auticko.imp* z úlohy 4.16 na strane 26 zmeňte tlačidlá na ovládanie autíčka na obrázkové.

Prvé stretnutie žiaka s tvarom korytnačky

Základný tvar zelenej korytnačky sa hodí na korytnačie kreslenie, lebo:

- má veľký počet záberov – dobre vizualizuje smer korytnačky,
- má primeranú veľkosť – nezakrýva príliš veľkú časť obrázka.

Keď začíname s programovaním v Logu (kreslíme obrázky), nie je dôvod, aby sme tvar korytnačky menili. Ak príliš skoro ukážeme žiakom možnosť zmeniť tvar korytnačky, nevyhneme sa tomu, že ho budú meniť nevhodne. Zvolia si tvar, ktorý je príliš veľký, odvádza pozornosť animáciou, má malý počet záberov, čím sa stráca informácia o natočení korytnačky.

Ak nechceme riešiť takéto problémy, odložme zmenu tvaru korytnačky na neskôr – v úlohách, v ktorých korytnačka neplní úlohu grafického pera, ale stáva sa „hercom“: dotvára nejaké prostredie (pozadie stránky), je aktérom deja, reaguje na udalosti.

V učebnici je nastavenie tvaru korytnačky motivované vytvorením „živého obrazu“. Do pozadia s chalupami, ktoré si žiaci pripravili pečiatkovaním obrázkov zo súboru a čarovaním obrázkov pri ťahaní korytnačky, pridávajú korytnačky s tvarom detí, ktoré vykúkajú z okien (obr. 5.4) a pri kliknutí reagujú rôznym pohybom.

Zo začiatku pracujme s jednoduchými neanimovanými tvarmi s jedným záberom.

Možné reakcie detí na kliknutie:

- drobné pohyby s návratom späť („mrvenie sa“),
- zmiznutie a objavenie sa znovu o nejaký čas,
- zmiznutie a objavenie sa v susednom okne.

Obr. 5.4: Živý obraz s korytnačkami-deťmi

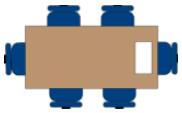
Ďalšou motiváciou na zmenu tvaru môže byť fakt, že korytnačky sa dajú presúvať ťahaním myšou. Môžeme tak vytvoriť projekty typu skladačka. Do prázdneho alebo neprázdneho pozadia si žiaci vložia niekoľko korytnačiek, ktorým nastavia tvary predstavujúce časti nejakého celku, a zapnú automatické ťahanie. Ťahaním presúvajú korytnačky na vhodné miesto. Potom všetky odtlačia a uložia si pozadie. Pri odtláčaní musia korytnačky oslovovať: spočiatku po jednej (napr. `k1 'odtlač`), neskôr naraz príkazom `pre všetky [odtlač]`. Na odtláčanie korytnačiek si môžu

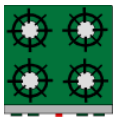
do projektu vložiť tlačidlo.

Námety na projekty typu skladačka:

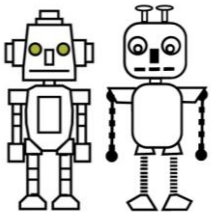
- Zariadiť izbu (pripraviť obrázky pre nábytok a iné zariadenie a pozadie s prázdnou izbou).
- Poskladať robota (pripraviť obrázky pre rôzne časti robota a nejaké futuristické pozadie).
- Puzzle (pripraviť obrázky, ktoré vzniknú rozstrihaním väčšieho obrázka na menšie štvorcové časti).
- Mapa Európy (pripraviť obrázky jednotlivých štátov a pozadie s obrysmi krajín celej Európy).

Námety na tvary prevzaté z prostredia *Revelation Natural Art*





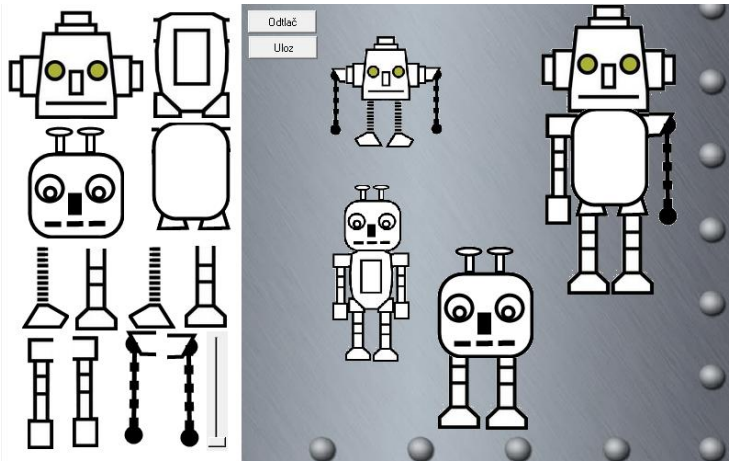
Robota si môžeme nakresliť v jednoduchom grafickom editore, napr. aj v *Skicári* a následne ho „rozstriháme“.



Dodržme proporcie ako i rovnaké časti v miestach budúceho strihania. Takto môžeme časti medzi sebou ľubovoľne kombinovať.

Úloha 5.5

Vytvorte projekt Robot. Cieľom projektu je z vopred pripravených častí poskladať ľubovoľného robota. Časti skladačky budú samostatné korytnačky s automatickým ťahaním. Po stlačení tlačidla Odtlač sa vyskladaný robot opečiatkuje na pozadie a časti sa znova uložia na svoje domovské miesto. Umožnite vytvorený obrázok uložiť a zároveň vyčistiť pozadie pre nové kreslenie.



V projekte s robotom si žiaci precvičia pridávanie korytnačiek a nastavovanie tvaru v rodnom liste korytnačky, ako aj ďalšie vlastnosti korytnačiek. Na karte *Tvar* môžu nastavovať zväčšenie. Na karte *Pozícia* nastavia domovskú pozíciu (tlačidlom *Prevezmi*) a príkazom `domov` vracajú časti robota späť na domovské miesto. Poskladaného robota odtlačia do pozadia pomocou tlačidla a uložia ho známym spôsobom voľbou *Ulož pozadie* z miestnej ponuky stránky alebo príkazom `uložPozadie` na ďalšom tlačidle. Pozadie pre nové kreslenie vyčistia opätovným načítaním pozadia stránky zo súboru voľbou *Pozadie zo súboru* z miestnej ponuky stránky alebo príkazom `pozadieZoSúboru` na tlačidle.

Tvary s viacerými zábermi

Tvary s viacerými zábermi slúžia na:

- vizualizáciu smeru korytnačky,
- uchovávanie rôznych tvarov, do ktorých korytnačku v projekte „prezliekame“ (steny hracej kocky, rôzne kocky stavebnice, rôzne variácie vybraného typu, napr. stromy, listy, zvieratá alebo časti nejakého celku).

Pri vizualizácii smeru korytnačky sa zábery tvaru menia automaticky pri zmene smeru korytnačky. Námety na prácu s automatickou zmenou záberov:

- odtláčať tvar korytnačky v rôznych natočeniach (napríklad ako v úlohe 5.3),

- zmeniť tvar korytnačky tak, aby sa hodila do prostredia (pozadia), v ktorom sa pohybuje:
  - navštíviť autíčkom rôzne miesta v meste,
  - raketou obletieť všetky hviezdy,
  - lietať motýľom z kvetu na kvet na lúke.

Úloha 5.6	Pripravte pre žiakov tvar vhodný na vizualizáciu smeru korytnačky.

Tvar korytnačky editujeme v grafickom editore *LogoMotion*, ktorý je súčasťou inštalácie Imagine Loga.

Návod na editovanie tvarov s viacerými zábermi je v Prílohe.

Korytnačky, ktorých zábery nepredstavujú smer, sa dajú využiť v projektoch-skladačkách. Rôzne obrázky, ktoré chceme použiť v projekte ako tvar korytnačky, vložíme do jedného súboru ako obrázok s viacerými zábermi. V programe potom môžeme tvar korytnačky meniť namiesto príkazu `nechTvar` s menom obrázka príkazom `nechZáber` s číslom záberu.

Výhodou záberov je, že sú zoradené a očíslované. Môžeme ich vyberať pomocou poradového čísla, meniť po poradí príkazom `nechZáber záber+1` alebo meniť náhodne príkazom `nechZáber ?`. Zoskupovanie obrázkov do postupností záberov v jednom súbore je príkladom štruktúrovania údajov, demonštruje výhody zoskupovania údajov pod jedným menom.

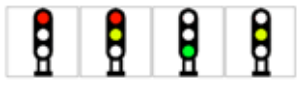
Úloha 5.7	Upravte projekt Robot z úlohy 5.5: Pre každú časť tela (hlava, trup, ľavá ruka, pravá ruka, ľavá noha, pravá noha) zoskupte obrázky do jedného súboru s viacerými zábermi – variáciami časti tela. Vytvorte korytnačky-časti tela. Generujte rôznych robotov náhodne - zmenou záberu všetkých korytnačiek.
Úloha 5.8	Preskúmajte projekt *10pismena.imp* z učebnice. Zistite, aké tvary majú korytnačky v tomto projekte. Zmeňte alebo doplňte obrázky do tvaru korytnačke-obrázkovej kartičke. 

V učebnici [3] je uvedená podobná aktivita: skladačka s kockami.

Vyskúšajte projekt *6kocky.imp* na CD vloženom do učebnice. Žiakom môže slúžiť ako motivácia na vytvorenie podobného projektu.

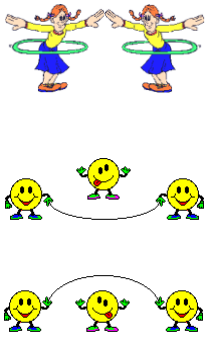
Animované tvary a procesy

V priečinku *Obrazky* Imagine Loga nájdeme viacero animovaných obrázkov, ktoré môžeme nastaviť ako tvar korytnačky. Ďalšie animované obrázky nájdeme na internete, databázy voľne dostupných animácií (väčšinou vo formáte *gif*) sú napr. na stránkach http://www.gifs.net/gif/ alebo http://www.amazing-animations.com/. Animovaný obrázok *gif* môžeme otvoriť v editore *LogoMotion* a uložiť ho ako tvar korytnačky vo formáte *lgf*.



Výučbe programovania animovaných korytnačiek by mala predchádzať na hodinách informatiky práca s animáciou v grafickom editore. Žiaci by mali poznať princíp animácie a vedieť vytvárať jednoduché animácie v grafickom editore LogoMotion alebo RNA.

Ďalšie dvojfázové animácie:



Motiváciou na použitie animovaného tvaru korytnačky môže byť vytvorenie pohyblivého obrázka do živého obrazu. Spočiatku vytvárame jednoduché dvojfázové animácie cyklického pohybu alebo blikajúce objekty – viacfázové animácie so zmenou farby. Automatická animácia cyklicky (donekonečna) strieda fázy obrázka v časových intervaloch nastavených v trvaní každej fázy. Netreba nič programovať, stačí vložiť korytnačku s animovaným tvarom do živého obrazu.

Obrázok 5.5: Ukážky jednoduchých dvojfázových animácií

Úloha 5.9	Vytvorte jednoduché dvojfázové animácie detí do živého obrazu na obr. 5.4, ktoré realizujú takéto opakujúce sa správanie detí: • dieťa v okne sa niekoľko sekúnd pozerá na jednu, potom niekoľko sekúnd na druhú stranu, • dieťa na niekoľko sekúnd zmizne, potom sa znovu objaví v okne.

Návod na vytvorenie animácie v grafickom editore LogoMotion je v Prílohe.

Cyklické striedanie fáz obrázka donekonečna je príkladom nekonečného *procesu*, ktorý sa v našom projekte deje automaticky. Podobné nekonečné a paralelné deje môžeme spustiť príkazom `každých`. Vstupom príkazu je počet milisekúnd a zoznam príkazov, ktorý sa má cyklicky donekonečna opakovať v zadanom časovom intervale.

Možné *problémy s nekonečnými procesmi*:

Žiak naprogramuje tlačidlo na spustenie nekonečného procesu. Tlačidlo potom stláča viackrát, procesy sa nekontrolovane množia, korytnačka vykonáva príkazy zo všetkých procesov. (Rovnako môže viackrát spustiť proces z príkazového riadka.)

Riešenie:
Tlačidlo naprogramujme ako prepínač. Pri zapnutí spustíme proces a pri vypnutí ho zastavíme.

Motiváciou na spustenie procesu (motorčeka) korytnačky je obyčajne posúvanie korytnačky dopredu, keď jej tvar animuje pohyb – animácia znázorňuje pohyb, ale korytnačka „prešľapuje“ na mieste. Príkladom rozpohybovania animovanej korytnačky je kohút v úlohe 4.3.

Korytnačka, ktorá v nekonečnom procese v určitých časových intervaloch niečo vykonáva, súčasne môže reagovať aj na ďalšie podnety a vykonávať príkazy z príkazového riadka, príkazy z tlačidiel, reakcie na udalosti (napr. kliknutie myšou), príkazy z iných procesov.

Úloha 5.10	Ako by ste zrealizovali scenár pre korytnačku-kohúta: • kohút sa pomaly prechádza, • každých 5 sekúnd sa otočí a prechádza sa opačným smerom, • pri kliknutí zakikiríka.
Úloha 5.11	Navrhnite ďalšie jednoduché úlohy s procesmi vhodné pre žiakov.

Čo sme sa naučili

Sumarizovali sme si vedomosti o tvaroch korytnačiek.

Analyzovali sme obsah učiva v tematickom celku Tvary, animované tvary, procesy.

Zoznámili sme sa s typmi aktivít na prácu s tvarmi korytnačky: živý obraz, skladačka, hry s ovládaním pohybu a smeru korytnačky, pečiatkovanie tvaru korytnačky.

Čo sme sa naučili v tomto module

Zhrnutie

Na vyučovanie programovania na ZŠ sa odporúča používať detský programovací jazyk.

Odporúčaná učebnica programovania je Tvorivá informatika: 1. zošit z programovania, ktorá používa jazyk Imagine Logo.

Vyučovanie programovania v Imagine Logu začíname programovaním kreslenia v korytnačej grafike.

Súčasne s procedurálnym programovaním zoznamujeme žiakov s princípmi objektovo-orientovaného a udalosťami riadeného programovania.

Volíme vyučovacie postupy, ktoré podporujú konštruktívny poznávací proces. Navrhujeme zmysluplné aktivity. Nové pojmy demonštrujeme na jednoduchých príkladoch. Uzavreté úlohy striedame s tvorivými zadaniami. Podporujeme experimentovanie a hry s malými didaktickými aplikáciami.

V programovaní využívame väzby na ostatné tematické okruhy informatiky a medzipredmetové vzťahy.

Preverenie výstupných vedomostí

Účastník vzdelávania preukáže požadované výstupné vedomosti podrobným vyriešením niektorej z tvorivých úloh podľa pokynov lektora.

Literatúra a použité zdroje

[1] Bezáková, D. a kol. (2009) Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika: Úvod do programovania. Bratislava : ŠPÚ, 2009. ISBN 978-80-89225-55-2

[2] Blaho, A., Kalaš, I. (1998) *Comenius Logo: Tvorivá informatika 1. časť*. 1. vyd. Bratislava : CL Group, 1998. ISBN 80-967999-0-8

[3] Blaho, A., Kalaš, I. (2007) Tvorivá informatika: 1. zošit z programovania. Bratislava : SPN – Mladé letá, 2007. ISBN 80-10-01223-7

[4] Hrušecká, A., Kalaš, I. (2006) Programovanie v prostredí Imagine. Bratislava : MPC, 2006. ISBN 80-8052-260-X

[5] Lovászová, G. a kol. (2010) Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika: Malé programovacie jazyky. Bratislava : ŠPÚ, 2010. ISBN 978-80-8118-066-8

[6] Salanci, Ľ. (2005) Tvorivá informatika: 1. zošit o obrázkoch. Bratislava : SPN – Mladé letá, 2007. ISBN 80-10-00649-1

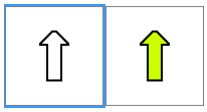
[7] Salanci, Ľ. a kol. (2010) Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika: Didaktika programovania. Bratislava : ŠPÚ, 2010. ISBN 978-80-8118-065-1

[8] Varga, M., Blaho, A., Zimanová, R. (1999) *Algoritmy s Logom*. Bratislava : SPN, 1999. ISBN 80-08-02965-X

Príloha – Tvorba animovaného obrázka

Kopírovanie fáz
a dokreslenie/úprava

Príklad 1 Vytvorme jednoduchú dvojfázovú animáciu blikajúcej smerovej šípky.



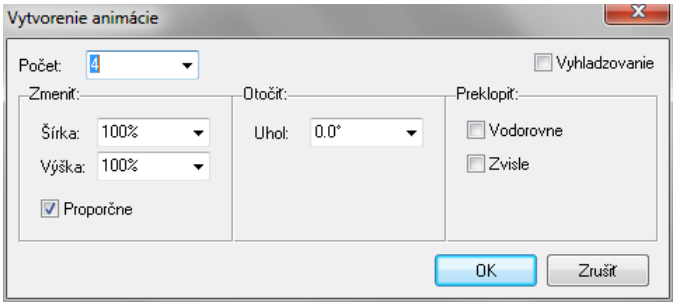
 tlačidlo *Ukážka*

Otvoríme LogoMotion a nastavíme si veľkosť papiera napr. na 30 x 50 (*Animácia/Nastaviť papier*). Pomocou štandardných kresliacich nástrojov v paneli *Kreslenie* nakreslíme šípku. Zobrazíme okno *Obsah* (*Zobraziť/Obsah*). Vyberieme myšou fázu s nakreslenou šípkou, skopírujeme ju do schránky a vložíme ako ďalšiu fázu. Šípku v druhej fáze vyfarbíme. Tlačidlom *Ukážka* si prezrieme vytvorenú animáciu.

Automatická animácia:
Otočiť

Príklad 2 Vytvorme animáciu otáčajúceho sa slniečka.

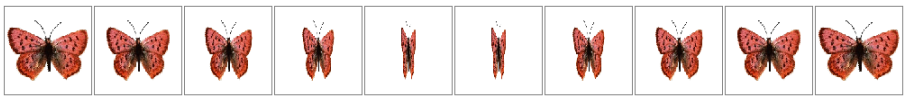
Do prvej fázy umiestnime obraz slniečka, označíme ho. Otvoríme dialógové okno Vytvorenie animácie (*Animácia/Vytvoriť animáciu*). Zvolíme *Otočiť* o uhol 360° a *Počet* fáz 6. Každá fáza sa pootočí o 60°. Pridá sa šesť fáz, posledná bude zhodná s prvou. Môžeme ju vymazať.



Obrázok 1: Dialóg pre automatické vytvorenie animácie

Automatická animácia:
Preklopiť

Príklad 3 Vytvorme animáciu motýľa mávajúceho krídlami.



Čím menší je počet fáz,
tým rýchlejšia je animácia.

Do prvej fázy umiestnime obraz motýľa, označíme ho, otvoríme okno *Vytvorenie animácie* (*Animácia/Vytvoriť animáciu*). Zvolíme *Počet* fáz 9 a *Preklopiť vodorovne*. Vytvorí sa dojem mávania krídlami.

Rovnako môžeme vytvoriť animácie otáčajúcich sa objektov.

Príklad 4

Vytvorme animáciu tlčúceho srdca.

Automatická animácia: **Zmeniť veľkosť**

Do prvej fázy umiestnime obraz srdca, označíme ho, otvoríme okno *Vytvorenie animácie* (*Animácia/Vytvoriť animáciu*). Zvolíme *Počet* fáz 4 a zmeníme *Šírku* a *Výšku Proporčne* na 75%. Vytvorí sa polovica animácie. V okne *Obsah* označíme všetky fázy, vytvoríme kópiu, v ktorej prevrátime poradie fáz (*Animácia/Prevrátiť poradie*). Poslednú fázu môžeme vymazať (je rovnaká ako prvá), tiež jednu z dvoch rovnakých prostredných fáz. Pulzujúci tlkot srdca dosiahneme nerovnakými dĺžkami trvania fáz (nastavujú sa v okne *Obsah* pod zobrazením fázy).

Časový interval predstavujúci dĺžku trvania fázy je vyjadrený v milisekundách. Čím vyšší je interval, tým dlhšie sa fáza zobrazí a animácia sa spomalí.

Príklad 5

Vytvorme tvar motýľa s viacerými zábermi znázorňujúcimi rôzne natočenia motýľa.

Zábery

Čím väčší je počet záberov, tým presnejšie korytnačka zobrazuje otáčanie.

Vytvoríme 12 fáz automaticky otáčaním tvaru motýľa rovnako ako v príklade 2. Fázy zmeníme na zábery voľbou *Animácia/Fázy/Zmeniť na zábery.*

Príklad 6

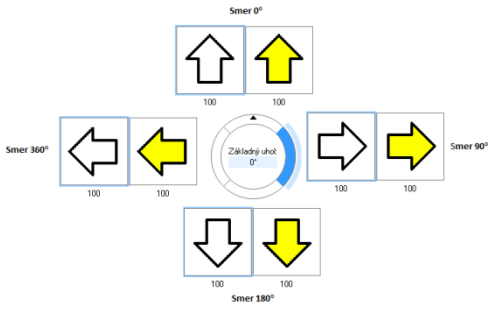
Nastavme základný bod blikajúcej šípky so štyrmi zábermi:

- do hrotu,
- do stredu.

Vyskúšajme otáčanie korytnačky s tvarom šípky vľavo, vpravo a kreslenie ťahaním myšou.

Základný bod

V každej fáze každého záberu je určený *základný bod*. Je to bod, kde je umiestnené kresliace pero a okolo ktorého sa otáča tvar korytnačky.



Základný bod nastavíme interaktívne pomocou myši po zvolení nástroja *Základný bod* v *Paneli nástrojov* alebo v dialógovom okne *Základný bod* (*Animácia/Nastaviť základný bod*) zadaním súradníc. Ak v okne *Obsah* označíme viac fáz, základný bod sa nastaví pre všetky označené fázy.

 **nástroj *Základný bod***

Častou chybou pri umiestňovaní základného bodu je jeho rozdielne nastavenie pre jednotlivé fázy. To spôsobuje „skákanie“ animácie do strán.

Fázy animácie môžeme vytvárať aj voľným kreslením. Môžeme tak vytvoriť *viacfázové animácie cyklického charakteru* – vizualizácia semafora, pohyb auta, skákanie lopty, alebo *animácie s príbehom* - zmena výzoru listnatého stromu počas roka, prezentácia rôznych fyzikálnych javov.

Príklad 7

Vytvorme animáciu štyri ročné obdobia.

Animácia deja

Zaujímavé animácie študentov nájdeme na http://www.gjar-po.sk/studenti/informatika/informatika_04_05.htm

Nakreslíme strom a spravíme niekoľko kópií. V kópiách dokresľujeme listy alebo sneh podľa ročných období. Užitočnou pomôckou sú priesvitky (*Zobrazenie/Priesvitky*) – zobrazujú predchádzajúce fázy na pozadí (nie sú súčasťou kresby).

Tento študijný materiál vznikol ako súčasť národného projektu Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika v rámci Aktivity „Vzdelávanie nekvalifikovaných učiteľov informatiky na 2. stupni ZŠ a na SŠ“.

Autori © RNDr. Gabriela Lovászová, PhD.
Mgr. Martin Cápay, PhD.
PaedDr. Viera Palmárová, PhD.

Názov Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika

Podnázov Didaktika programovania pre ZŠ 1

Študijný materiál prešiel recenzným pokračovaním.

Recenzenti PaedDr. Daniela Bezáková, PhD.
RNDr. Ľubomír Šnajder, PhD.

Počet strán 36

Náklad 300 ks

Prvé vydanie, Bratislava 2011

Vydal Štátny pedagogický ústav, Pluhová 8, 830 00 Bratislava, v súčinnosti s Univerzitou Pavla Jozefa Šafárika v Košiciach, Univerzitou Komenského v Bratislave, Univerzitou Konštantína Filozofa v Nitre, Univerzitou Mateja Bela v Banskej Bystrici a Žilinskou univerzitou v Žiline

Vytlačil BRATIA SABOVCI, s r.o., Zvolen

ISBN 978-80-8118-080-4