Procedúry

Procedúra bez parametrov

S pojmom procedúra sa stretávame od začiatku programovania v Lazaruse. Dvojklikom na tlačidlo vo formulári sa na to automaticky pripravila konštrukcia **procedúry**. Túto konštrukciu sme mohli ďalej rozpracovávať: pridávali sme deklarácie premenných a do tela procedúry sme zapisovali postupnosť nejakých príkazov.

Okrem vytvárania nových procedúr sme pracovali aj s niektorými hotovými procedúrami: všetky grafické príkazy (napr. Rectangle, MoveTo, TextOut, ...) sú tiež procedúry, ktoré pre nás pripravili nejakí programátori. Nás už nemusí zaujímať ako sú naprogramované, len musíme vedieť ako sa používajú. Pre väčšinu týchto príkazov sme ešte museli zadávať aj súradnice bodov, prípadne nejaký text – hovoríme im **parametre** procedúry. Pre nás je pri používaní procedúry dôležitý počet parametrov a typ parametra. Okrem grafických príkazov sme používali už hotové podprogramy aj pre textovú plochu Memo (Clear, Append, ...).

Podprogramom nazývame pomenovanie nejakého algoritmu, nejakej časti programu s tým, že túto časť môžeme volať pomocou tohto mena.

V Pascale môžu byť podprogramy dvoch typov:

- **procedúry** – postupnosť príkazov, ktoré riešia nejakú podúlohu
- **funkcie** – postupnosť príkazov, ktoré majú za úlohu vypočítať nejaký výsledok – nejakú jednu hodnotu

Keď už raz zadefinujeme nejaký podprogram, tak potom ho môžeme volať aj na viacerých miestach. Najprv sa naučíme pracovať iba s procedúrami, t.j. podprogramami, ktoré pomenúvajú nejakú postupnosť príkazov. Takéto vlastné podprogramy môžeme umiestniť napr. do časti implementation nad všetky procedúry, ktoré vytvára Lazarus. Pozrime sa, ako budeme definovať svoju vlastnú procedúru **Zmaz**:

```
implementation

{$R *.lfm}

{ TForm1 }

procedure Zmaz;

begin

  Form1.Image1.Canvas.Brush.Color:=clWhite;

  Form1.Image1.Canvas.FillRect(Form1.Image1.ClientRect);

end;
```

meno procedúry

pretože ju vytvárame sami, musíme povedať aj pre ktorý formulár platí, preto každý príkaz začína `Form1.`

Teda keď chceme definovať vlastnú procedúru, musíme sa rozhodnúť na aké miesto ju vložíme a tiež meno procedúry musíme vymyslieť my.

Zadefinujte jednoduchú procedúru s menom **VypisPascal**, ktorá na náhodnú pozíciu v grafickej ploche napíše slovo **Pascal**.

Zadefinujte procedúru, ktorá na náhodnú pozíciu nakreslí 10 sústredných kruhov náhodnej farby.

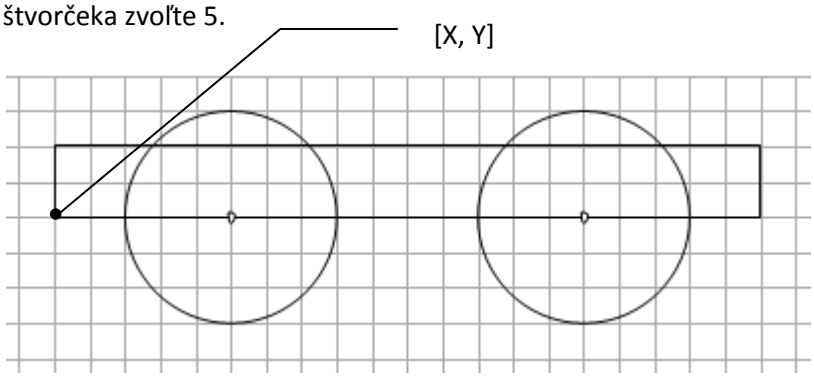
Zatlačenie tlačidla zmaže plochu na modro a nakreslí nočnú oblohu s 10 malými žltými hviezdičkami na náhodných pozíciách. Hviezdičky kreslite pomocou procedúry **Hviezda** (krúžok žltej farby na náhodnej pozícii s náhodným polomerom od 5 do 10).

Mechanizmus volania procedúry

Vieme, že **volanie procedúry** znamená, že sa pri vykonávaní príde na riadok s menom nejakého podprogramu. Vtedy sa postupuje nasledovne:

- **zapamätá sa návratové miesto**, t.j. riadok, kam sa treba vrátiť po skončení procedúry (vo vnútri počítača sa toto realizuje tak, že sa zapamätá adresa príkazu v pamäti, kde sa bude pokračovať vo vykonávaní po návrate z procedúry);
- **vytvoria sa lokálne premenné** procedúry (so zatiaľ nedefinovanou hodnotou);
- prenesie sa **riadenie** programu **do tela podprogramu** (za príslušný **begin**);
- vykonajú sa všetky príkazy podprogramu (až po koncový **end**);
- **zrušia sa lokálne premenné**;
- **riadenie sa vráti** za miesto v programe, odkiaľ bol podprogram volaný – t.j. **na návratové miesto**.

Zadefinujte procedúru **Vozik**, ktorá je zložená z pomocných procedúr **Doska** a **Koleso**. Tieto dve pomocné procedúry musia byť zadefinované ešte pre procedúrou Vozik, ktorá ich bude volať, a teda tieto dva identifikátory musia byť pre ňu viditeľné. Pri definovaní procedúr využite mriežku – veľkosť štvorčeka zvoľte 5.



Takto nakreslený vozík rozhýbte pomocou Timeru. Nezabudnite, že procedúra **Vozik** pokazila pri kreslení koliesok premennú X a my ju budeme potrebovať. Chceme, aby sa X zvyšovalo o hodnotu 10.

Ako naprogramujeme preteky dvoch vozíkov? Musíme mať okrem globálnych premenných X, Y pre nakreslenie vozíka ďalšie štyri globálne premenné X1, Y1, X2, Y2, v ktorých budeme uchovávať informácie o dvoch vozíkoch – tieto sa budú pohybovať paralelne.

Zopakujeme si ako je to s **viditeľnosťou premenných**. Hovoríme, že identifikátor (premennej alebo podprogramu) je definovaný v nejakej úrovni:

- všetky štandardné identifikátory (napr. preddefinované grafické príkazy, matematické funkcie a pod.) sú definované v **0. úrovni**;
- procedúry, ktoré spracovávajú akcie od udalostí (napr. kliknutie tlačidlá, ...), ale aj naše vlastné procedúry, ktoré sme umiestnili do časti implementation sú na **1. úrovni**;
- všetky premenné a procedúry definované v procedúrach na **1. úrovni** (tzv. vnorené procedúry) sú na **2. úrovni**;
- atď.
- každý identifikátor sa môže používať, až potom, keď bol zadeklarovaný, resp. zadefinovaný;
- identifikátor je viditeľný v tej procedúre (na tej úrovni), v ktorej bol zadefinovaný, ale aj vo všetkých v nej vnorených procedúrach.

O lokálnych premenných procedúry už vieme, že vznikajú pri volaní procedúry, existujú počas behu procedúry a zanikajú pri skončení procedúry.

Premenné, ktoré zadefinujeme na 1. úrovni sa nazývajú **globálne premenné** a majú tieto špeciálne vlastnosti:

- vznikajú pri štarte programu
- existujú so svojimi hodnotami počas celého behu programu
- môžeme im nastaviť ich počiatočné hodnoty už pri deklarácii – ak to nespravíme, majú nulové hodnoty

Tieto globálne premenné sú viditeľné vo všetkých procedúrach, ktoré sú v programe definované neskôr ako tieto premenné.

Procedúra s parametrami

Procedúry poskytujú elegantnejší spôsob tzv. posielanie parametrov. Procedúru môžeme zadefinovať tak, že pri jej volaní zadáme parametre, ktoré zapíšeme do okrúhlych zátvoriek. Aj všetky grafické príkazy sú definované s parametrami, napr. **Rectangle** má štyri celočíselné parametre. Keby sme túto procedúru chceli zavolať s iným počtom parametrov ako 4, alebo jej namiesto celého čísla poslali text, Pascal by nás upozornil na chybu v programe.

Keď budeme definovať nejakú svoju procedúru, počet parametrov, ich mená a aj typy závisia len od nás.

Prepíšme procedúry **Vozik**, **Doska**, **Koleso** na procedúry s parametrami a skúsme preteky vozíkov vyriešiť pomocou týchto nových procedúr.

Mechanizmus volania procedúry s parametrami

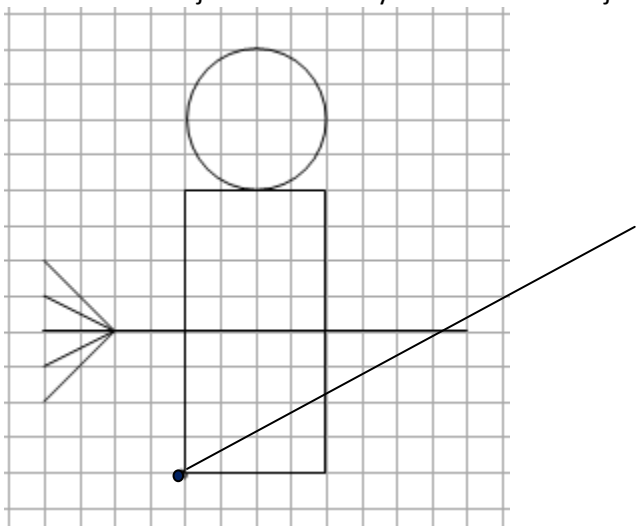
V predchádzajúcom texte sme popísali mechanizmus volania procedúry bez parametrov. Teraz, keď procedúra môže obsahovať aj parametre, tento mechanizmus treba doplniť. Teda pri **volaní procedúry** sa postupne:

- **zapamätá návratové miesto**, t.j. riadok, kam sa bude treba vrátiť po skončení procedúry;
- **vytvoria sa všetky zadeklarované lokálne premenné procedúry** – tieto majú nedefinovanú hodnotu – aj parametre sú lokálne premenné, preto sa na tomto mieste **vytvoria aj všetky parametre** a do nich sa priradia príslušné vstupné hodnoty;
- prenesie sa **riadenie** programu **do tela podprogramu** (za príslušný **begin**);
- vykonajú sa všetky príkazy podprogramu (až po koncový **end**);
- **zrušia sa lokálne premenné** – a teda sa **zrušia aj parametre**;
- **riadenie sa vráti** za miesto v programe, odkiaľ bol podprogram volaný – t.j. **na návratové miesto**.

Pristávanie bosoriek

Vytvorte program, ktorý bude simulovať pristávanie dvoch bosoriek vedľa seba. Pri pristávaní sa bosorky pohybujú smerom dole. Postupujte podľa zadania:

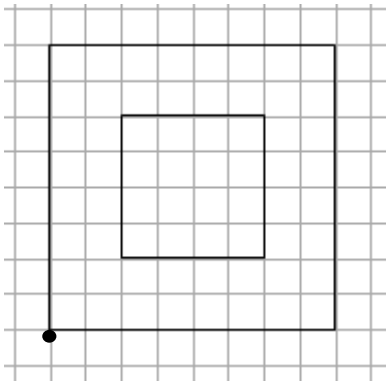
a) Vytvorte procedúru bosorka s troma parametrami, prvé dva nech určujú polohu bosorky a tretí nech určuje farbu bosorky. Nech štvorček tejto mriežky má veľkosť 5.



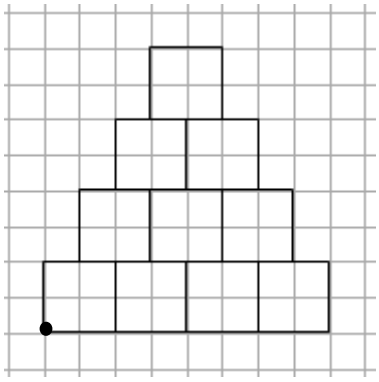
b) Na začiatku nech sú bosorky červená a modrá.
c) V každom kroku animácie sa každá bosorka pohybuje náhodnou rýchlosťou od 0 do 10 bodov (v jednom kroku). Program nám v závere vypíše na obrazovku informáciu o poradí pristátia bosoriek.
d) Zabezpečte, aby užívateľ mohol simuláciu bosoriek neustále spúšťať, bez toho, aby si stále spúšťal program. Pri každej simulácii nech je farba bosoriek iná. Výpis o poradí pristátia nech je tou farbou akú má víťazná bosorka.

Úlohy na riešenie:

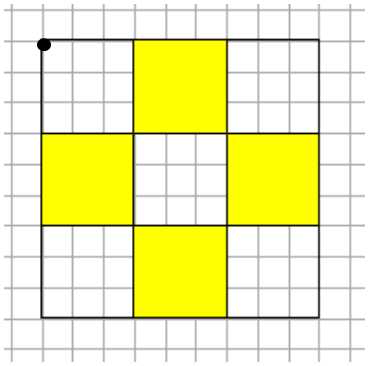
1. Procedúra **Panel** nakreslí šedý štvorec 40x40, v ktorom je bledomodrý menší štvorec (okno). Procedúra **Panelak** má dva parametre: počet panelov na jednom poschodí (šírka) a počet poschodí (výška). Nakreslite vedľa seba tri rôzne vysoké paneláky.



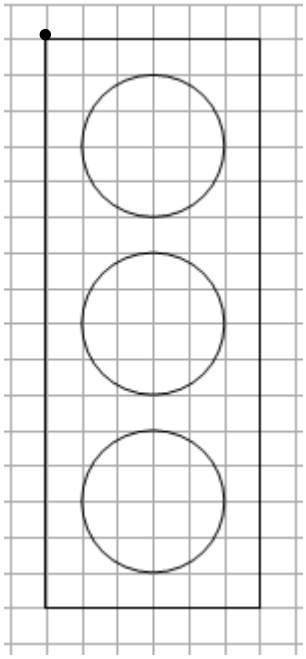
2. Zadefinujte procedúru **RadStvorcov(X, Y, N)**, ktorá nakreslí vedľa seba **N** štvorcov veľkosti 20, pričom prvý z nich začína na súradnici **[X, Y]**. Pomocou tejto procedúry zadefinujte procedúru **Pyramida(N)**, ktorá nakreslí pyramídu výšky **N** poschodí. Na obrázku je Pyramida(4)

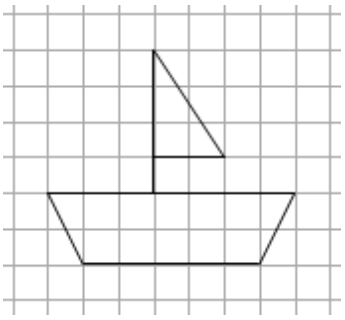


3. Procedúra **Stvorec(X, Y, Farba)** nakreslí farebný štvorec veľkosti 30. Napíšte procedúru **Sachovnica(N)**, ktorá nakreslí šachovnicu veľkosti NxN štvorcov (pomocou procedúry **Stvorec**), v ktorej sa striedajú dve farby štvorcov. Na obrázku je Sachovnica(3).



4. Vytvorte procedúru **Kruh(X, Y, Farba)** s polomerom 20 a pomocou tejto procedúry procedúru **Semafor(X, Y, Farba1, Farba2, Farba3)** – šírka semaforu nech je 60 a výška 160. Na náhodnú pozíciu nakreslite 3 semafory.





5. Vytvorte procedúru **Lodicka(X, Y)**. Potom naprogramujte preteky dvoch lodičiek. V časovači sa posunie o náhodnú hodnotu X – ová súradnica oboch lodičiek, plocha sa zmaže a nakreslia sa obe lodičky na nových pozíciách. Nech štvorček má hodnotu 5. Toto môžete vyriešiť aj pomocou konštanty – nech štvorček je konštanta A.