Stavebné prvky jazyka Pascal

3.1 Procedúry (bez parametrov)

S pojmom procedúra sa stretávame od začiatku programovania v prostredí Delphi. Začali sme s ňou pracovať hneď vtedy, keď sme definovali akcie, ktoré sa majú vykonať pri stlačení nejakého tlačidla. Dvojklikom na tlačidlo vo formulári sa na to automaticky pripravila konštrukcia procedúry (v tomto prípade je to procedúra na spracovanie nejakej udalosti – kliknutie na tlačidlo). Túto konštrukciu môžeme ďalej rozpracovať: pridať deklarácie premenných a do tela procedúry zapísať postupnosť nejakých príkazov.

Okrem vytvárania takýchto 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 vo firme Borland. Vďaka tomu nás nemusí zaujímať, ako sú naprogramované. Pre nás je dôležité vedieť, ako sa používajú. Pre väčšinu týchto príkazov sme ešte museli zadávať aj súradnice bodov, prípadne nejaký text – hovoríme im **parametre** príkazu (procedúry). Pre nás je pri používaní procedúry dôležitý počet parametrov a či je príslušný parameter číslo alebo text.

Podprogramy zvykneme vytvárať aj z iných dôvodov, ako sme uviedli. Pri tvorbe väčších programov je vhodné si celú úlohu najprv rozdeliť na menšie podúlohy. Potom riešime každú takúto podúlohu zvlášť a pritom dbáme na to, aby to dokopy dávalo riešenie našej pôvodnej úlohy. Často pre každú takúto podúlohu vytvoríme samostatný podprogram – **zadefinujeme podprogram**. Podprogram pritom dostáva meno, aby sme sa pri riešení úlohy mohli na tento podprogram odvolávať práve týmto menom – budeme tomu hovoriť **volanie podprogramu**. Znamená to, že ak na nejakom mieste programu uvedieme meno podprogramu, na tomto mieste sa použije (zavolá) príslušný podprogram.

V Pascale poznáme podprogramy dvoch typov:
- **procedúra** – postupnosť príkazov, ktoré riešia nejakú podúlohu,
- **funkcia** – postupnosť príkazov, ktoré majú za úlohu vypočítať nejaký výsledok – nejakú hodnotu.

Keď už raz zadefinujeme nejaký podprogram, tak ho potom už môžeme volať na viacerých miestach. Definovali sme ho raz, ale používame ho veľakrát. Ak pri ladení programu v ňom nájdeme nejakú chybu, tak ju opravíme len raz. Programátori často, po napísaní podprogramu, ho najprv samostatne odladia a až potom ho začnú používať (volať) v svojom hlavnom programe. Rozdelenie úlohy na podprogramy má aj tú výhodu, že na týchto rozdelených podúlohách môže naraz pracovať viac programátorov. Niektorí programátori môžu vytvárať podprogramy a iní ich vo svojich programoch (alebo aj podprogramoch) používajú.

Najprv sa naučíme pracovať iba s procedúrami, t. j. podprogramami, ktoré pomenúvajú nejakú postupnosť príkazov. Pozrime sa, ako budeme definovať svoju vlastnú procedúru:

```
procedure meno;
   // deklarácie premenných pre danú procedúru
begin
   // telo procedúry
end;
```

Pozrime sa na mechanizmus, ktorý prebehne pri volaní každého podprogramu (t. j. ak sa pri vykonávaní programu príde na riadok s menom nejakého podprogramu):

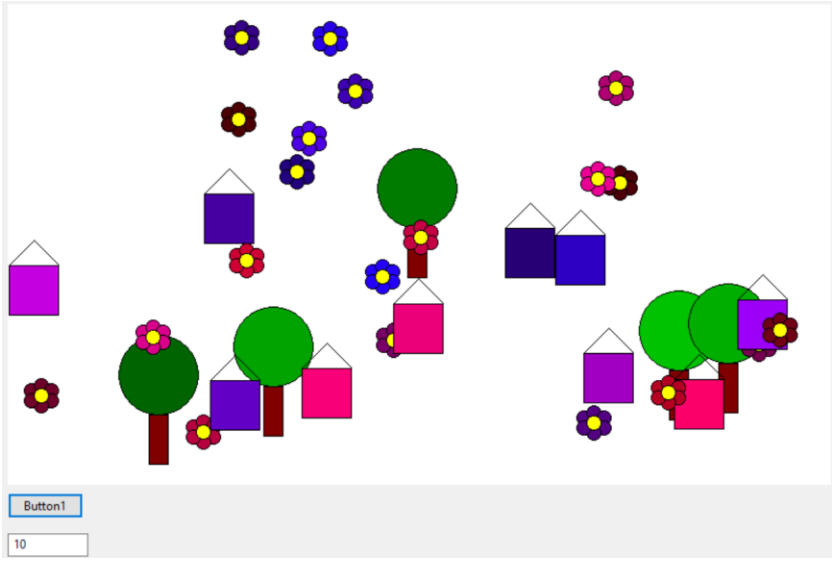
1. Zapamätá sa návratová adresa (kam sa treba vrátiť).
2. Vytvoria sa lokálne premenné procedúry (so zatiaľ nedefinovanou hodnotou) – v našom prípade sú to premenné **x**, **y**, **i**.
3. Prenesie sa riadenie programu do tela podprogramu (za príslušný **begin**).
4. Vykonajú sa všetky príkazy podprogramu (až po koncový **end**).
5. Zrušia sa lokálne premenné – „zabudnú" sa premenné **x**, **y**, **i**.
6. Riadenie sa vráti za miesto v programe, odkiaľ bol podprogram volaný, t. j. na návratovú adresu.

Teraz uvedieme pravidlá, ktoré v Pascale hovoria o identifikátoroch premenných a podprogramov:

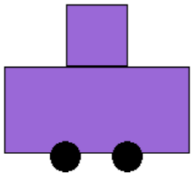
- každý identifikátor sa môže používať až potom, keď bol zadeklarovaný, resp. zadefinovaný,
  - — každá premenná musí byť zadeklarovaná ešte pred **begin** (telom procedúry),
  - — každá procedúra musí byť v programe zadefinovaná skôr ako ju budeme volať,
- štandardné deklarácie sú definované, akoby ešte pred samotným programom – hovoríme, že sú na nultej úrovni,
  - — napr. grafické podprogramy, matematické funkcie a pod.
- procedúry, ktoré spracúvajú akcie od udalostí, napr. kliknutie tlačidla, sú na prvej úrovni,
  - — procedúra **Button1Click** je na prvej úrovni
- hovoríme, že všetky premenné a procedúry definované v procedúrach na prvej úrovni sú na druhej úrovni,
  - — napr. **nahodnyTerc**, **vypisDelphi** sú na druhej úrovni
  - — lokálne premenné procedúry **Button1Click** sú na druhej úrovni
  - — lokálne premenné **x**, **y**, **i** deklarované v procedúre **nahodnyTerc** sú na tretej úrovni
- identifikátor nesmieme v rámci jednej deklarácie definovať viackrát, ale môžeme ho prekryť deklaráciou vo vyššej úrovni,
- identifikátor „vidí" tá procedúra, ktorá ho definovala ale aj všetky v nej vnorené procedúry, ak tento identifikátor neprekryli nejakou svojou definíciou,
- identifikátor je viditeľný len na svojej úrovni (len od tohto miesta v programe nižšie) a na všetkých pre neho vnorených úrovniach, kde nie je predefinovaný.

```
procedure TForm1.Button1Click(Sender: TObject);

var //globálna premenná pre MojaProcedúra, ale lokálna premenná pre Button1Click

    procedure mojaProcedura;
    var //lokálna premenná procedúry MojaProcedúra
    begin

      //telo procedúry mojaProcedúra

    end;


var //lokálna premenná procedúry Button1Click
begin

  //telo Button1Click-u

  mojaProcedura;  //volanie procedúry mojaProcedúra

end;
```

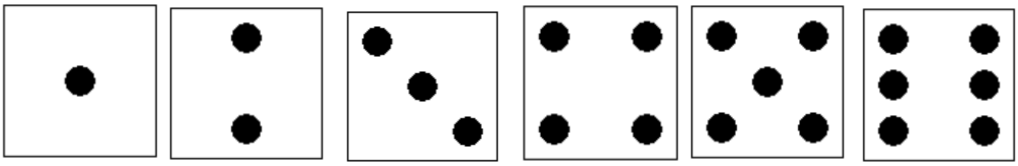
1. Napíšte procedúry: `strom` (koruna je v odtieňoch zelenej), `kvet` (každý kvet je náhodnej farby), `dom` (je v odtieňoch fialovej), ktoré kreslia príslušné tvary na náhodných pozíciách po celom Image (žiadny z nich netrčí von z Image). Tlačidlo si od vás vypýta číslo a vykreslí zadaný počet domov, dvojnásobok kvetov a polovicu čísla stromov.



2. Napíšte procedúry `koleso`, ktorá kreslí koleso auta  , `karoseria`, ktorá nakreslí karosériu auta  a `auto`, ktorá z predchádzajúcich dvoch procedúr poskladá auto.



3. Napíšte procedúry `jednotka`, `dvojka`, `trojka`, `stvorka`, `patka`, `sestka`, ktoré budú vykresľovať príslušnú hodnotu na hracej kocke.

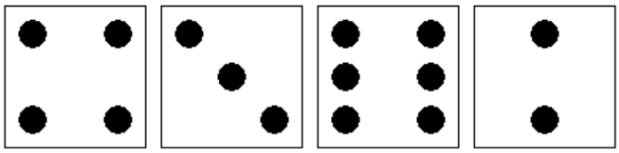


Napíšte program, ktorý simuluje hru s kockami. Pravidlá hrania: hrá sa na 10 hodov, hádže sa 4 kockami (stav jednotlivých hodov vykreslite pomocou for cyklu):

a. ak padnú 4 rovnaké čísla skóre sa zväčší o 10násobok padnutej hodnoty,
b. ak padnú 3 rovnaké čísla skóre sa zväčší o 30,
c. ak padnú 2 rovnaké čísla skóre sa zväčší o 2 body (neriešime prípad ak padnú 2 rovnaké dvojice)
d. ak padnú 4 rôzne čísla odráta sa jeden bod.

4ty ťah

nahral si: 2 bodov



koniec hry, tvoje nahrané skóre je: 115bodov
pre novú hru stlač tlačidlo

4. Napíšte procedúru `clovek`, ktorá bude kresliť postavu. Program bude vykresľovať rad ľudí, pričom každý tretí bude väčší ako ostatný.