Střední průmyslová škola a Vyšší odborná škola technická Brno, Sokolská 1

Šablona: **Inovace a zkvalitnění výuky prostřednictvím ICT**

Název: **Programování se strukturovanými údaji**

Téma: **Globální a lokální proměnné**

Autor: **Ing. Hodál Jaroslav, Ph.D.**

Číslo: **VY_32_INOVACE_26–15**

Anotace: *Materiál popisuje globální a lokální prostředky podprogramů, především pak proměnné a jejich použití v jazyce Pascal.*
Materiál je určen pro 3. a 4. ročník oboru strojírenství a technické lyceum. Vytvořeno: duben 2013.

15. Globální a lokální proměnné

- protože podprogramy (viz 26-14) představují programy vnořené v programu a protože mohou definovat své vlastní prostředky (proměnné, konstanty ...) mohlo by docházet k mnoha problémům a zmatkům při jejich používání
- proto jsou veškeré prostředky (nejen proměnné) definované a deklarované kdekoliv v kódu programu (včetně kódu podprogramů) formálně rozděleny na dva druhy
  - **globální** (všeobecné, všude platné)
  - **lokální** (místně omezené, platné jen někde)

Globální prostředky

- všem prostředkům, které jsou uvedeny přímo v hlavičce hlavního programu, se říká **globální**, těm které si definuje nějaký podprogram, se pak říká **lokální**
- **globální** prostředky lze použít **kdekoliv** v kódu hlavního programu
- to znamená, že je možné používat je i **ve všech podprogramech**, které jsou součástí hlavního programu, stejně jako ve všech podprogramech do nich vnořených v libovolné úrovni
- existují ale rozumné důvody, proč používání globálních prostředků v podprogramech co nejvíce omezit (viz 26-19 – Programové jednotky)

Lokální prostředky

- **lokální prostředky** jsou takové, které jsou definovány či deklarovány v libovolném podprogramu
- jejich existence a tedy i použitelnost je **omezena** jen na kód podprogramu, v němž jsou umístěny
- mimo kód podprogramu je **nelze** použít, protože se vytvářejí a opět zanikají jen v okamžiku, kdy se kód podprogramu vykonává
- typicky se jedná o **pomocné proměnné**, které potřebuje podprogram pro provádění svých příkazů (pro ukládání mezivýsledků, řídící proměnné cyklů, ...), ale jejichž hodnota je po skončení podprogramu nepotřebná
- méně často se vyskytují **lokální konstanty**, ale i ty je možné v podprogramech deklarovat a používat

Vnořené podprogramy

- poměrně zajímavá situace nastává u vnořených podprogramů (podprogram jiného podprogramu)
- vlastní prostředky vnořeného podprogramu jsou samozřejmě vůči němu **lokální** (může je používat jen on)
- lokální prostředky podprogramu, ve kterém je vnořen, jsou však pro něj prostředky **globálními**
- pokud je vnoření víceúrovňové, jsou pro něj globálními všechny lokální prostředky všech podprogramů, ve kterých je postupně zanořen
- zároveň jsou pro něj dostupné i globální prostředky hlavního programu

Další zvláštnosti

- jako lokální proměnné se chovají i tzv. **parametry** (viz 26-18 – Parametry podprogramů)
- narozdíl od běžné lokální proměnné ale budou mít nastavenou **výchozí hodnotu**
- velkou pozornost bude nově vyžadovat také nazývání různých prostředků
- syntaxe jazyka Pascal totiž **připouští stejné názvy** globálních a lokálních prostředků v jednom programu
- může se tedy např. stát, že některý podprogram bude mít k dispozici stejně pojmenovanou lokální i globální proměnnou
- v takovém případě má vždy **přednost lokální** prostředek

Příklad globální a lokální proměnné

```
program texty;
var s:string; mz:char;  //globální
procedure maxznak;
var i:byte;             //lokální
begin
  mz := s[1];           //zde můžeme používat
  for i := 2 to length(s) do      //lokální i
    if mz < s[i] then mz := s[i]; //globální
end;

begin
  readln(s);            //pouze globální
  maxznak;              //použití lokálních
  writeln(mz);          //by vedlo k chybě
end.
```