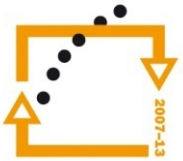

INVESTICE
DO ROZVOJE
VZDĚLÁVÁNÍ

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

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

Název: **Programování se strukturovanými údaji**

Téma: **Programové jednotky**

Autor: **Ing. Hodál Jaroslav, Ph.D.**

Číslo: **VY_32_INOVACE_26–19**

Anotace: *Materiál popisuje strukturu a zapojení programové jednotky v programech jazyka Pascal. Materiál je určen pro 3. a 4. ročník oboru strojírenství a technické lyceum. Vytvořeno: květen 2013.*

Podpora digitalizace a využití ICT na SPŠ CZ.1.07/1.5.00/34.0632

19. Programové jednotky

- programové jednotky jsou samostatné soubory obsahující části programového kódu, který lze připojit do libovolného programu
- díky tomuto mechanismu nemusíme často používané úseky kódu opakovaně psát do každého programu, ve kterém je potřebujeme použít
- programová jednotka tak funguje jako knihovna hotových řešení
- zdrojový soubor programové jednotka má stejně jako hlavní program opět příponu pas, jeho struktura je však značně odlišná

Připojení programových jednotek

- programová jednotka může být do hlavního programu připojena přímo (v jeho části *připojení programových jednotek*, viz 25-01 - Základní kostra programu v jazyce Pascal)
- lze ji ale také připojit prostřednictvím jiné programové jednotky, která ji potřebuje pro svůj chod (jednotka připojuje a používá jinou jednotku)
- připojení programových jednotek probíhá v bloku

```
uses název_1_jednotky,název_2_jednotky;
```

- i při použití programových jednotek vzniká překladem celého programu stále jediný výstupní spustitelný (*.exe) soubor

Obsah programových jednotek

- programová jednotka obsahuje nejčastěji následující fragmenty zdrojového kódu:
- konstanty
- definice datových typů
- podprogramy (procedury a funkce)

- v pascalovských prostředích také existují již hotové jednotky například
  - pro práci s grafikou (jednotka CRT)
  - pro komunikaci s operačním systémem disky apod. (jednotka DOS)
  - rozšiřující matematické funkce (jednotka Math) atd.

Struktura programové jednotky (1)

- programová jednotka má 4 části:
  - hlavička
  - rozhraní (interface)
  - výkonná část (implementační)
  - finalizační část (někdy také nazývaná inicializační)

- hlavička obvykle obsahuje pouze název jednotky, někdy v ní bývají připojovány další programové jednotky
- je uvozena klíčovým slovem **`unit`**
- uvedení hlavičky je povinné

```
unit název_jednotky;
```

Struktura programové jednotky (2)

- rozhraní je jakousi „výkladní skříní“ programové jednotky
- obsahuje všechny prostředky (konstanty, datové typy, podprogramy), které programová jednotka „nabízí“ k použití jakémukoliv programu či jiné jednotce, které si ji připojují

```
interface
   připojení programových jednotek
 A deklarace konstant
   definice datových typů
   podprogramy (pouze jejich deklarační
část)
```

Struktura programové jednotky (3)

- implementační (výkonná) část je místem, kde musí být zrealizovány všechny podprogramy přislíbené v části rozhraní
- dále zde mohou být i další konstanty, datové typy i podprogramy, které ale budou sloužit pouze pro potřeby této programové jednotky

```
implementation
    připojení programových jednotek
  A deklarace konstant
  B definice datových typů
    podprogramy (z interface)
    podprogramy (místní)
```

Struktura programové jednotky (4)

- finalizační/inicializační část není uvozena žádným klíčovým slovem
- zůstává obvykle prázdná
- v takovém případě je tvořena pouze klíčovým slůvkem **`end`** následovaným tečkou (stejně jako na konci programu)
- pokud by měla něco obsahovat, bude se jednat o sadu příkazů
- ty by pak byly uzavřeny do kompletních programových závorek **`begin`** - **`end.`**

Příklad programové jednotky (1)

```
unit vypocty;                                          hlavička

interface                                              rozhraní
  const
    MojePI=3.1415;
  function ObsahKruhu(polomer:byte):real;

implementation                                         výkonná část
  function ObsahKruhu(polomer:byte):real;
  begin
    ObsahKruhu := MojePI*sqr(polomer);
  end;

end.                                                   finalizační část
```

- konstanta **`MojePI`** i funkce **`ObsahKruhu`** bude k dispozici v celé jednotce i kdekoliv, kam je jednotka připojena

Připojení programové jednotky (1)

```
program kruhy;
uses vypocty;
var
  r: byte;

begin
  readln(r);
  writeln('Obsah kruhu o polomeru ',r,
    ' pri pouziti PI s presnosti ', MojePI,
    ' je ', ObsahKruhu(r):0:3);
end.
```

- konstantu **`MojePI`** zde lze v kódu použít proto, že byla uvedena v části interface
- kdyby byla deklarována v jiné části programové jednotky, mohla by ji využívat jen tato jednotka a to jen v kódu následujícím po deklaraci

Příklad programové jednotky (2)

```
unit vypocty;                                          hlavička

interface                                              rozhraní
  function ObsahKruhu(polomer:byte):real;

implementation                                         výkonná část
  const
    MojePI=3.1415;
  function ObsahKruhu(polomer:byte):real;
  begin
    ObsahKruhu := MojePI*sqr(polomer);
  end;

end.                                                   finalizační část
```

- konstanta **`MojePI`** je k dispozici pouze v kódu této programové jednotky
- funkce **`ObsahKruhu`** bude k dispozici v celé jednotce i kdekoliv, kam je jednotka připojena

Příklad programové jednotky (3)

```
unit vypocty;                                                    hlavička

interface                                                        rozhraní
  procedure Kruh(polomer:byte; var obsah, obvod:real);

implementation                                                   výkonná část
  function ObsahKruhu(polomer:byte):real;
  begin
    ObsahKruhu := 3.14*sqr(polomer);
  end;
  function ObvodKruhu(polomer:byte):real;
  begin
    ObvodKruhu := 2*3.14*polomer;
  end;
  procedure Kruh(r:byte; var obsah, obvod:real);
  begin
    obsah := ObsahKruhu(r);
    obvod := ObvodKruhu(r);
  end;

end.                                                             finalizační část
```

Připojení programové jednotky (3)

```
program kruhy;
uses vypocty;
var
  r: byte;
  s, o : real;

begin
  readln(r);
  Kruh(r, s, o);

  writeln('Obsah kruhu o polomeru ',r, ' je ', s:0:3);
  writeln('Obvod kruhu o polomeru ',r, ' je ', o:0:3);
end.
```

- přímé volání funkcí **ObsahKruhu** a **ObvodKruhu** by vedlo k chybě, protože předchozí programová jednotka je nenabízí v části interface a program se k nim tedy nemůže dostat protože o jejich existenci „neví“