Programové jednotky TPU

Programová jednotka - knihovna procedur a funkcí

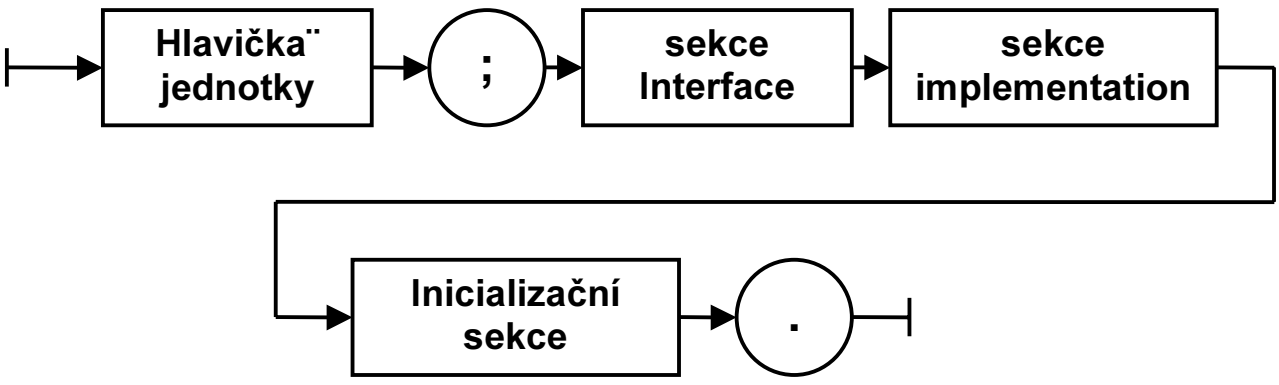
Důvody zavedení TPU :

- **umožňuje vytvářet knihovny**
- **u starších verzí Pascalu - maximální velikost programu = velikost segmentu (64 KB), větší programy - rozdělit na jednotky, každá jednotka je umístěna v samostatném segmentu tj. maximální velikost každé jednotky je 64 KB.**

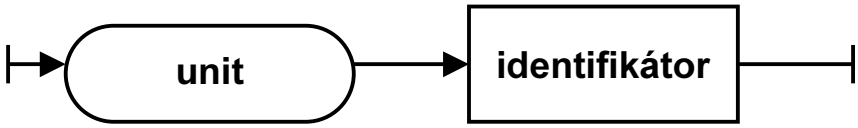
Jednotky v Turbo Pascalu :¨

- **standardní - jsou součástí distribuce TP**
- **uživatelsky definované**

Uživatelsky definované jednotky



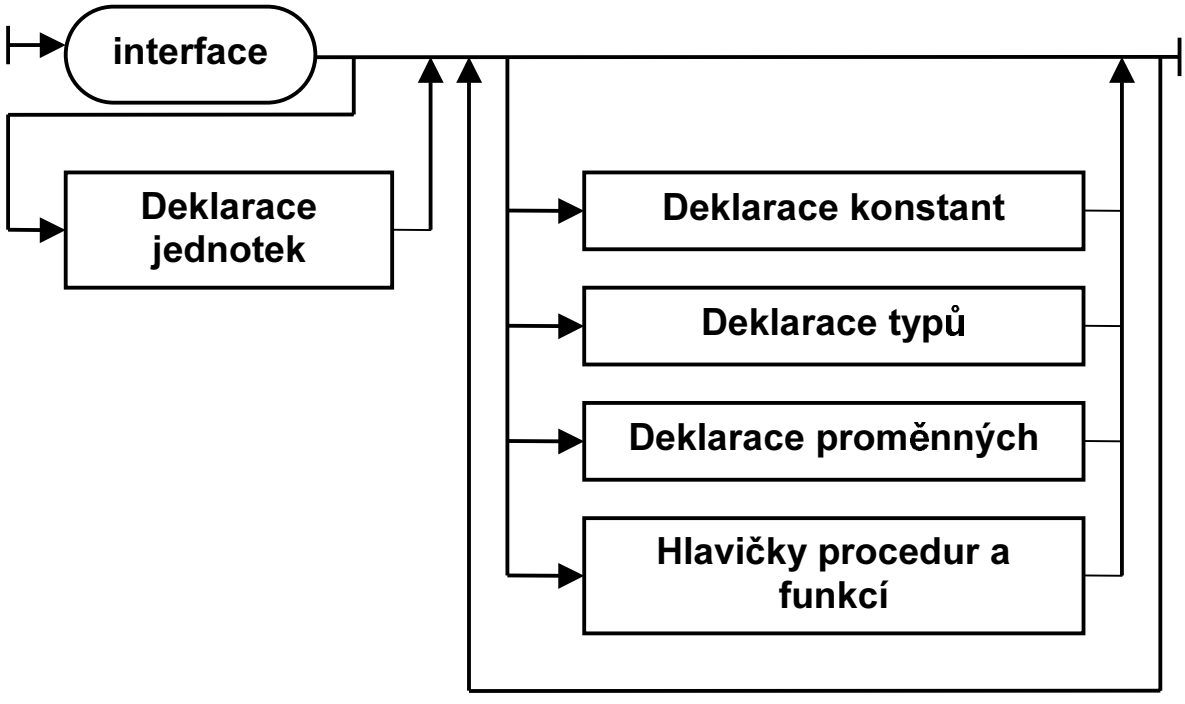
Hlavička jednotky



Identifikátor jednotky musí být shodný se jménem souboru (bez přípony) ve kterém je uložen zdrojový text jdnotky, jinak dojde k chybě.

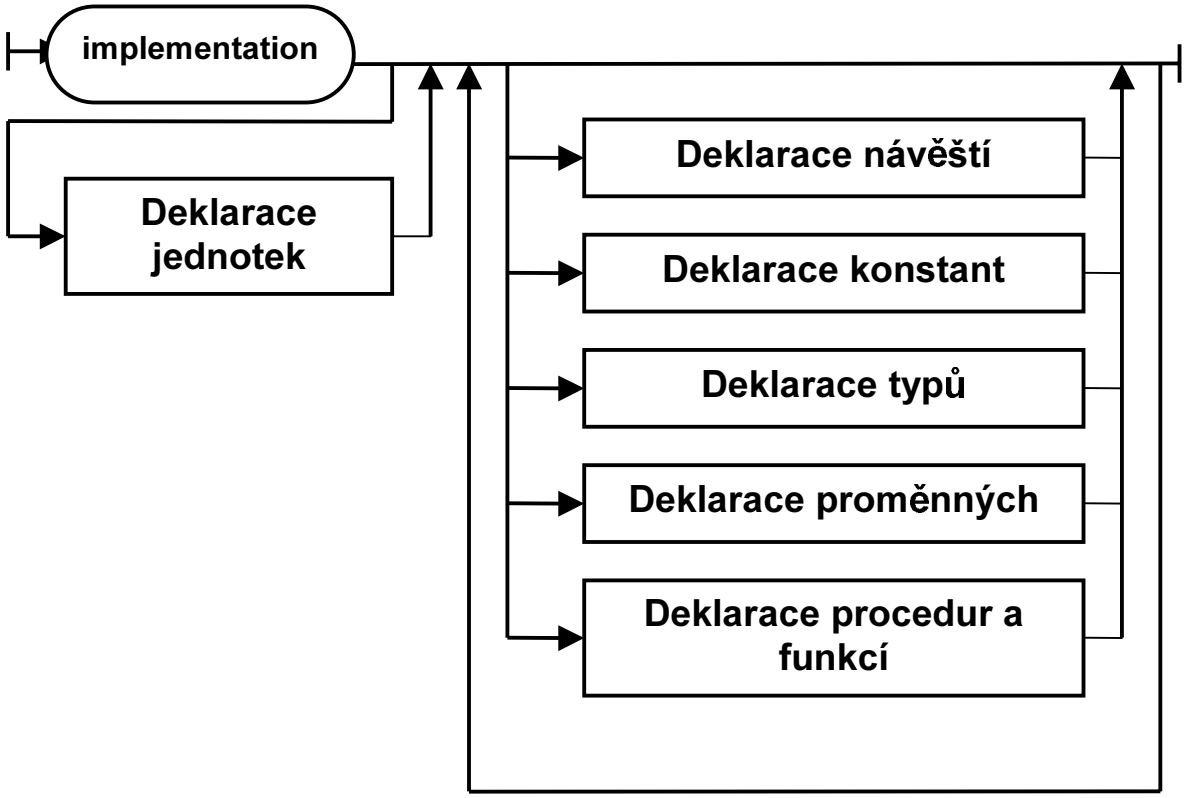
Sekce interface

- **obsahuje deklarace konstant, typů, proměnných a hlavičky procedur a funkcí, které jsou veřejné tj. jsou dostupné v programu (v jednotce), ke kterému je jednotka připojena. Sekce interface končí uvedením rezervovaného slova *implementation***



Sekce implementation

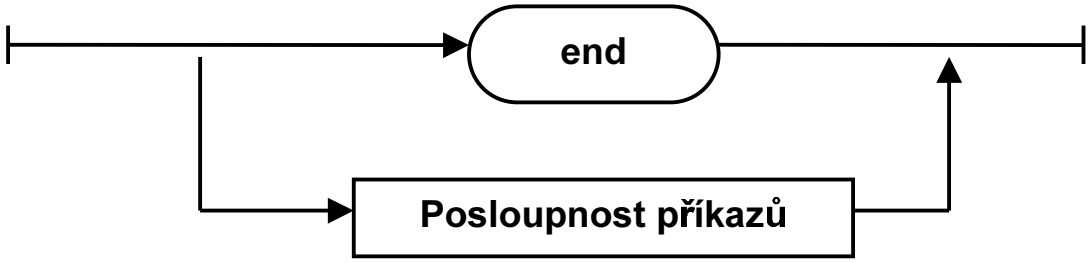
- **obsahuje těla veřejných procedur a funkcí deklarovaných v části interface,**
- **obsahuje deklaraci pomocných návěští, konstant, typů, proměnných, procedur a funkcí, které jsou lokální v jednotce (uživatel jednotky nemá k těmto datům přístup).**



Hlavičky procedur a funkcí definovaných v sekci *interface* mohou být v sekci *implementation* zopakovány, musí se však shodovat s hlavičkami ze sekce *interface*.

Inicializační sekce

- **obsahuje buďto rezervované slovo *end* (v případě, že jednotka nemá inicializační kód) nebo posloupnost příkazů, které se provedou před první instrukcí hlavního programu, který jednotku využívá. Používá-li hlavní program více programových jednotek, jsou inicializační sekce vykonávány postupně v pořadí v jakém jsou uvedeny v příkazu *uses.***



Použití programových jednotek

Programové jednotky nejsou používány ve zdrojové formě, ale jsou samostatně překládány – přeložením vznikne soubor s příponou TPU (Turbo **P**ascal **U**nit **).**

Přeložený kód je sestaven společně s hlavním programem, který jednotku používá. Použití programových jednotek s hlavním programu vyžádá příkazem

uses unit1, unit2, … ;

umístěným za hlavičkou programu

Př. ***Program pokus;***
uses Crt, Dos, Graph;

Pořadí jednotek v seznamu za uses hraje roli, pokud nějaká jednotka používá další jednotku (nejprve musí být uvedena jednotka, která je využívána).

Př. Jednotka pro práci s maticemi

```
unit matrix;
interface
const  max=10;
type matice = array [ 1..max, 1.. max] of real;
var IDENT : matice;
procedure cti_matici(var A: matice; var N : integer);
procedure souc(A,B: matice; var C : matice; N:integer);
…
implementation
 var I,J: integer;
 procedure cti_matici;
   var I,J: integer;
 begin
  writeln('Zadej rad matice'); readln(N);
  for I:=1 to N do
   for J:=1 to N  do read(A[I,J]);
 end;
  …
begin
 for I := 1 to N do
   for J := 1 to  N do
     if I=J then IDENT[I,J] := 1 else IDENT[I,J]:=0;
 end.
```

```
Program pokus;
 uses matrix;
var A,B,C: matice;
      N1,N2: integer;
begin
 cti_mat(A,N1);
 cti_mat(B,N2);
 if N1=N2 then souc(A,B,C,N1);
 …
end.
```

Standardní programové jednotky

- neliší se od programových jednotek definovaných uživatelem. Hlavním účelem je rozšířit vlastnosti Turbo Pascalu a doplnit vestavěné procedury, funkce a konstanty.

Předdefinované jednotky :

CRT – Umožňuje lepší práci s klávesnicí a obrazovkou, tj. řízení režimu obrazovky, rozšířené kódy klávesnice, barvy, práce s okénky, řízení zvuku.

DOS – Zpřístupňuje nejdůležitější funkce MS-DOSu, především funkce data a času, funkce řízení adresářů a funkce spouštění programů.

OVERLAY – Dovoluje používat metodu překrývání modulů.

GRAPH – Programová podpora různých grafických karet, implementuje jednoduchý grafický systém složený z funkcí, které nastavují grafické prostředí, kreslí grafické objekty, vrací parametry nastaveného prostředí .

PRINTER – Zjednošuje práci s tiskárnou.

SYSTEM – Knihovna pomocných funkcí Turbo Pascalu.

Standardní programové jednotky se používají pomocí klauzule *uses,* např.

```
Uses Dos,Crt,Graph;
```

Jednotka SYSTEM se přisestavuje automaticky (nemusí být uvedena za *uses*).

Jednotka PRINTER

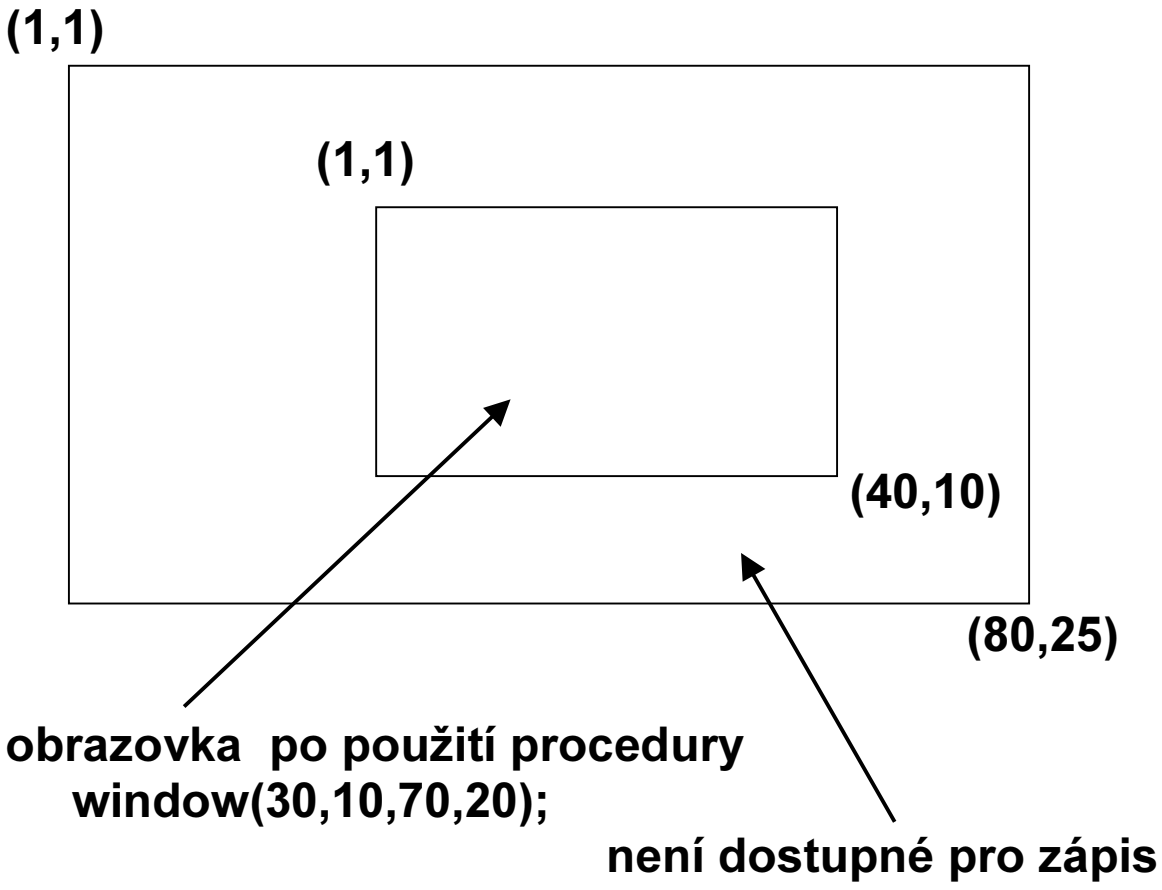
- **deklaruje textový soubor Lst a přiřadí jej k zařízení LPT1.**

```
program Test_printer;
 uses Printer;
begin
  Writeln(Lst, 'Toto je test tiskárny');
end.
```

Jednotka CRT

- **jednotka obsahuje standardní prostředky pro snadnou práci s obrazovkou (v textovém režimu), klávesnicí a zvukovým generátorem. Umožňuje řízení režimů obrazovky, rozšířeného kódu klávesnice, barev, oken a zvuku.**
- **Jednotka využívá standardních služeb BIOSu, popř. zapisuje přímo do obrazové paměti ⟶ rychlost (proměnná *DirectVideo* musí být nastavena na *true*, jinak jsou výstupy na obrazovku přes standardní rutiny MSDOS).**

Obrazovka v textové režimu



Všechny obrazovkové souřadnice (kromě souřadnic v proceduře window) jsou vztaženy relativně k pozici levého horního rohu obrazovky (okna) (1,1)

1. **Procedury pro změnu pozice kurzoru**
   GotoXY(X,Y),
   X:=WhereX,
   Y:=WhereY } **vrací pozici kurzoru**

2. **Práce s obrazovkou (změna barev, atributů textu, okna …)**
   Window(X1,Y1,X2,Y2) **vytvoří okno**
   TextColor(Color) **nastaví barvu textu**
   TextBackground(Color) **nastaví barvu pozadí**
   ClrScr **maže obrazovku**
   …
3. **Práce s klávesnicí**
   Keypressed **- funkce, vrací true pokud byla stisknuta klávesa**

Typické použití:
repeat until keypressed;

Readkey – funkce, čte znak z klávesnice bez výpisu na obrazovku. Po stisknutí funkční klávesy (F1 – F12) popř. rozšiřující klávesy (šipky, editační klávesy) vrací funkce nejprve znak 0 a potom tzv. scan kód klávesnice.
Ctrl, Alt, Shift, Pause, Num Lock, … negenerují žádný kód

4. Práce se zvukem

Sound(Hz) **spustí tonový generátor**
NoSound **vypne tónový generátor**

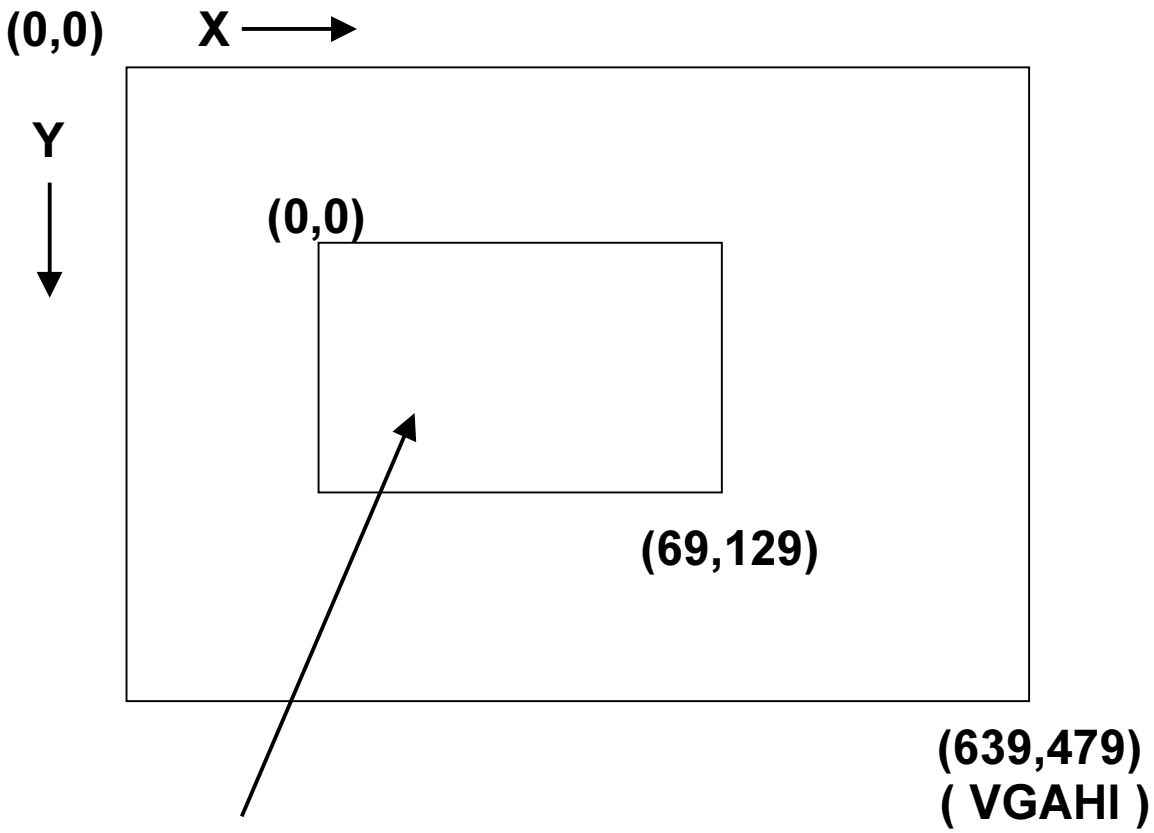
Programová jednotka GRAPH

- **Programová jednotka GRAPH je samostatný grafický systém, který je potřeba na začátku práce s grafikou inicializovat a po ukončení práce s grafikou řádně ukončit**

Pokud chceme pracovat s grafikou musíme mít k dispozici:

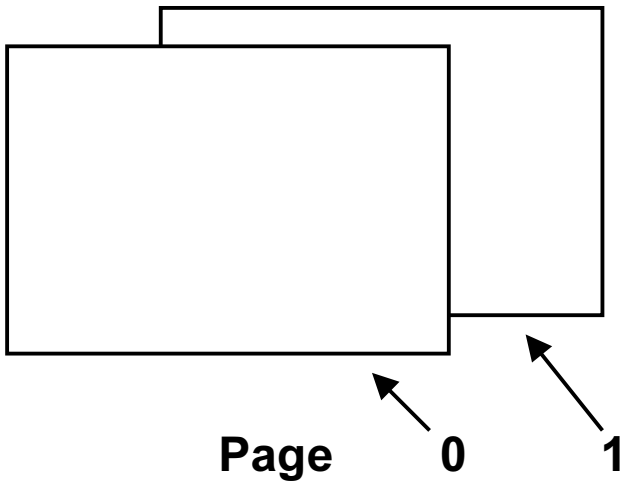
a) **Ovladače grafické obrazovky (soubory s příponou .BGI). Základní drivery jsou součástí TP. Driver musí odpovídat grafickému adaptéru umístěnému v počítači (v současné době nejčastěji SVGA grafické karty ⟶ použít ovladač *egavga.bgi*).**
b) **Soubory sazeb písma (soubory .CHR), které obsahují fonty které se používají k psaní textu v grafickém režimu.**

Obrazovka v grafickém režimu



okno v grafickém režimu
Set*viewport(X1,Y1,X2,Y2,ClipOn)*

Některé adaptéry umožňují vytvářet více grafických stránek (vhodné pro animace)



Každá grafická operace vrací kód, který indikuje způsob jejího ukončení, popř. druh vzniklé chyby.
Návratový kód operace ⟶ funkce *Graphresult*, lze volat po ukončení grafické operace reprezentované podprogramem.
Slovní popis chyby (v angličtině) - funkce *GraphErrorMsg(Errcode), ErrCode* - kód chyby vrácený funkcí *GraphResult.*

Před zahájením práce s grafikou je nutná inicializace grafiky tj. použití procedury

Initgraph(var Driver, Mode: integer; Path: string)

***Driver* - ovladač grafické karty,**
předdefinované hodnoty
***Detect* - automatická detekce karty**
EGA
VGA
Hercmono

***Mode* - odpovídající mód zobrazení (určuje rozlišení)**
předdefinované hodnoty

EGALO	**(640 x 200)**
EGAHI	**(640 x 350)**
VGALO	**(640 x 200)**
VGAMED	**(640 x 350)**
VGAHI	**(640 x 480)**

***Path* - cesta k souboru s grafickým ovladačem**

Základní struktura grafického programu

```
program grafika;
 uses graph;
var Driver, Mode, Err : integer;
 begin
   Driver:=Detect;
   Initgraph(Driver, Mode, 'E:\TP\BGI');
   Err := GraphResult;
   If Err <> GrOk then
     begin
     writeln('Chyba pri inicializaci:',GraphErrorMsg(Err));
       …
    { ošetření chyby, popř. ukončení programu }
       …
     end
else  begin
            …
        {   použití grafických procedur     }
            …
        RestoreCrtMode;  { přepnutí do textového módu }
            …
         {   práce v textovém módu    }
            …
         SetGraphMode(Mode); { návrat do grafiky }
            …
          {   použití grafických procedur     }
            …

          CloseGraph; { ukončení práce s grafiko }

end.
```

POZOR !!!
Pokud chceme zapisovat text na grafickou obrazovku není možné použít procedury *write*, popř. *writeln* .

Nastavení textového stylu :

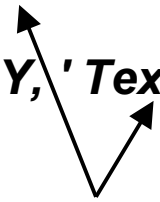
SetTextStyle(Font, Direction, Charsize)

DefaultFont ***HorizDir***
TriplexFont ***VertDir***
SmallFont
GothicFont
SansSerifFont

Výstup textu do grafiky :

OutText(' Text ') **- tisk v aktuální pozici kurzoru**

OutTextXY(X, Y, ' Text ') **- tisk v zadané pozici**



je možné použít i proměnnou typu *string*

Přepínání grafických stránek

- **lze použít pouze pokud to umožňuje ovladač**

SetActivePage(Page) **- stránka do které se zapisuje**

SetVisualPage(Page) **- stránka, která je viditelná**