

Programování 2 (NMIN102)

Soubory

RNDr. Michal Žemlička, Ph.D.

Soubor

- abstrakce vstupního, výstupního či vstupně–výstupního zařízení
- textová, typovaná a netyповaná varianta
- základní operace: otevření, čtení/zápis, zavření, kontrola konce

Textový soubor

- obdoba práce se standardním vstupem či výstupem – můžeme do něj zapisovat řetězce, čísla, ...
- používáme příkazy `write`, `writeln`, jen předřadíme (jako první parametr) odkaz na patřičný soubor, do něhož chceme zapisovat

Textový soubor – příklady

VAR

```
soubor : Text;
```

```
i, j, k : Integer;
```

```
s : String;
```

```
c : Char;
```

```
...
```

```
WriteLn(soubor, 'Ahoj ', s, '!');
```

```
Write(soubor, i:4, ' - ', j:6); WriteLn(soubor, '...', c);
```

Textový soubor – příklady (2)

```
BEGIN
```

```
Assign(soubor, 'jmeno_souboru'); {přiřadíme jméno souboru}
```

```
Reset(soubor); (* otevřeme soubor na čtení *)
```

```
j:=0; k:=0;
```

```
WHILE NOT EOF(soubor) DO
```

```
    BEGIN Read(soubor,i); j:=j+i; k:=k+1 END;
```

```
Close(soubor);
```

```
WriteLn('Soucet: ',j,', pocet: ',k);
```

```
END.
```

Textový soubor – podprogramy

assign(f,fn)	přiřazení jména souboru
reset(f)	otevření souboru pro čtení
rewrite(f)	otevření souboru pro zápis
append(f)	otevření souboru pro dopsání
close(f)	zavření souboru

Textový soubor – podprogramy (2)

read, readln	čtení ze souboru
write, writeln	zápis do souboru
eof()	už jsme na konci souboru?
eoln()	jsme na konci řádku?
seekeof()	až na bílé znaky konec souboru
seekeoln()	až na bílé znaky konec řádku

Typovaný soubor

- FILE OF *základní_typ*;
- kolekce položek daného typu
- záznamy stejné velikosti → můžeme si najít k . záznam, pokud potřebujeme

Typované soubory – příklad

TYPE

```
zaznam = RECORD
```

```
  jmeno, prijmeni: String;
```

```
END;
```

```
soubory = FILE OF zaznam;
```

VAR

```
soubor : soubory;
```

Typované soubory – odlišnosti

- `Read` a `Write` mají právě 2 parametry: soubor a záznam
- Varianty s odřádkováním pro tento typ souborů nejsou k dispozici (pouze pro `Text`)
- Jsou k dispozici nové funkce pracující s pozicí v souboru: `FilePos` (získá aktuální pozici v souboru) a `Seek` (nastaví aktuální pozici v souboru)

Typované soubory – pozice

Pozice v souboru udává, s kolikátým záznamem pracujeme nebo chceme pracovat

```
FUNCTION FilePos(VAR F:TypovanySoubor):pozice;
```

```
PROCEDURE Seek(VAR F:TypovanySoubor; p:pozice);
```

pozice je celočíselný údaj (longint u BP, int64 u FP)

Netyповaný soubor

Někdy potřebujeme pracovat se souborem velmi specificky a přitom efektivně – čtení byte po byte je příliš pomalé, čtení záznam po záznamu příliš restriktivní, a tak je zde i možnost pracovat po blocích.

TYPE

```
f : FILE;
```

Netypovaný soubor – odlišnosti

- čtení a psaní probíhá po blocích
- nefungují běžné souborové operace (ani `Read` a `Write`)
- k dispozici nové operace *BlockRead* a *BlockWrite*
- `Assign`, `Reset` a `Rewrite` fungují, byť poslední 2 s odlišností – může být deklarována (jako další parametr) délka bloku (pokud nezadáme, je to 128)

Netypovaný soubor – BlockRead

```
PROCEDURE BlockRead(VAR f: file; VAR Buf;  
                    count: pocet; VAR Result: pocet);
```

pocet je zpravidla největší celočíselný typ – u FP Int64 a u BP longint

Pozor! Musíme zajistit, že Buf je dostatečně veliký, aby se do něj patřičný počet záznamů vešel!

Nelze načíst jen část záznamu.

Netypovaný soubor – BlockWrite

```
PROCEDURE BlockWrite(VAR f: file; VAR Buf;  
                    count: pocet; VAR Result: pocet);
```

pocet je zpravidla největší celočíselný typ – u FP Int64 a u BP longint

Nelze zapsat jen část záznamu.

count říká, kolik záznamů chceme zapsat, result kolik se zapsat povedlo.

Přepínače

Chování námi vytvářeného programu můžeme ovlivnit nastavením parametrů při překladu, případně vhodným nastavením přepínačů specifickými direktivami.

syntax: `{$X+}`, `{$X-}`, kde X je příslušný přepínač

Přepínače (2)

Můžeme ovlivňovat:

kontrolu mezí \$R (range check) – hlídání mezí intervalů
(tedy i indexu polí)

kontrolu zásobníku \$S (stack check) – hlídá se, zda nepřetekl zásobník (to je při rekurzi či rozsáhlých parametrech či lokálních parametrech podprogramů možné)

kontrolu vstupu \$I (input) – hlídání práce se soubory během podporou/programem

A spustu dalších nastavení

Podmíněný překlad

```
{ $\$$ define ladime}
```

```
...
```

```
{ $\$$ ifdef ladime}
```

```
    tohle je pro ladění
```

```
{ $\$$ else}
```

```
    tohle je pro produkční verzi
```

```
{ $\$$ endif}
```

Podmíněný překlad (2)

```
{$ifopt R+}
```

```
    tady kontroluje systém
```

```
{$else}
```

```
    a tady musíme kontrolovat sami
```

```
{$endif}
```

Přepínače a direktivy

- přepínače a direktivy bývají specifické pro daný překladač
- některé direktivy se liší i dle platformy
- některé symboly definovány jen na určité platformě (jen při překladu pro ni), jiné zas pro určitou variantu jazyka či verzi prostředí