

Jazyk C

Práce se soubory

- Přístup k souboru
 - Sekvenčně
 - Náhodně
- V/V operace **bufferované** = omezení počtu periferních operací (stream – proud dat).
- ANSI C nepřipouští přímo nebufferované V/V, ale **dá se nastavit bufferování blokové, řádkové nebo žádné**.
- ANSI C rozlišuje **binární** soubor a **textový** soubor.

Práce se souborem v C

FILE *f; - f je pointer na objekt typu FILE

Slovo **FILE** musí být psáno velkými písmeny (předdefinované MAKRO).

Identifikátor **f** se dá použít jak pro zápis tak čtení ze souboru.

Otevření souboru pro **čtení**:

```
f = fopen('pokus.dat', 'r');
```

Otevření souboru pro **zápis**:

```
f = fopen('pokus.dat', 'w');
```

Ukončení práce se souborem

- Zapiše na disk vše co bylo zapsáno bufferu a uzavře soubor.

```
fclose(f);
```

Operace s otevřeným souborem

- Čtení znaku ze souboru: **c = getc(f);**
- Zápis znaku do souboru: **putc(c, f);**
- Formátované čtení ze souboru:
fscanf(f, 'formát', argumenty);
- Formátovaný zápis do souboru:
fprintf(f, 'formát', argumenty);

Příklad

```
#include <stdio.h>
void main()
{
    FILE *fw;
    int i;

    fw = fopen('pokus.txt', 'w');
    for(i = 1; i <= 10; i++)
        fprintf(fw, '%d \n', i);
    fclose(fw);
}
```

Režimy otevření souboru

- **r** – textový soubor pro čtení
- **w** – textový soubor pro zápis nebo přepsání
- **a** – textový soubor pro připojení na konec
- **rb** – binární soubor pro čtení
- **wb** – binární soubor pro zápis nebo přepsání
- **ab** – binární soubor pro připojení na konec

Režimy otevření souboru

- **r+** – textový soubor pro čtení a zápis
- **w+** – textový soubor pro čtení, zápis nebo přepsání
- **a+** – textový soubor pro čtení a zápis na konec
- **rb+** – binární soubor pro čtení a zápis
- **wb+** – binární soubor pro čtení, zápis nebo přepsání
- **ab+** – binární soubor pro čtení a zápis na konec

Test správnosti otevření a uzavření souboru

Test otevření:

```
if((fr = fopen("test.txt","r"))== NULL)
    printf("Soubor se nepodarilo otevrit");
```

Test uzavření:

```
if (fclose(fr)== EOF)
    printf("Soubor se nepodarilo uzavrit");
```

Test konce řádky

Možné přístupy s označením konce řádky:

- Samotný znak **<CR>** (carriage return) – znakově **'\r'**, hexa: **0xD**, dekadicky: **13** – posun kurzoru na začátek řádky
- Samotný znak **<LF>** (linefeed) – znakově **'\n'**, hexa: **0xA**, dekadicky: **10** – posun kurzoru na novou řádku v tomtéž sloupci
- Znak **<CR>** bezprostředně následovaný znakem **<LF>**. Opačné pořadí znaků jen výjimečně.

Test konce řádky - 2

- Při zmáčknutí klávesy **<Enter>** z klávesnice dostáváme znak **<CR>**, ale na obrazovku se vypíše **<CR><LF>**.
- V **UNIXu není problém** – používá se **jen <LF>**.
- **'\n'** – interpretace je vnitřní záležitost překladače -> nemusíme se o to více starat, můžeme používat jak pro čtení tak zápis do souboru

Test čtení do konce řádky:

```
if((c = getc(fr))!= '\n')
```

Test konce souboru

- Pomocí symbolické konstanty **EOF**
 - Definováno v stdio.h s hodnotou (-1)
 - Test:

```
if((c = getc(fr)) != EOF)
```

- **proměnná c musí být typu int**

Test konce souboru

- Pomocí standardního makra **feof()**
 - Je pomalejší – volání tohoto makra
 - Vrací hodnotu **FALSE** (nula) – pokud nejsme na konci souboru
 - Vrací **TRUE** (nenulová hodnota) – pokud poslední čtení bylo již za koncem souboru
 - Je vhodnější pro testování u binárních souborů - může se vyskytnout byte s hodnotou **0xFF** – implicitní typová konverze převede na **EOF** - > čtení by skončilo uprostřed souboru!
 - Test:

```
while(c = getc(fr), feof(fr) == 0){}
```

Standardní vstup a výstup

- **stdin** – standardní vstup (klávesnice)
- **stdout** – standardní výstup (monitor)
- **stderr** – standardní chybový výstup

Práce s binárními soubory

- **fw = fopen("kopie.txt", "wb");**
- **fr = fopen("kopie.txt", "rb");**
- Zápis dat do souboru
- **fwrite(char *odkud, int velikost, int pocet, FILE *f);**
- Čtení dat ze souboru
- **fread(char *kam, int velikost, int pocet, FILE *f);**

Nastavení pozice v souboru

- **int fseek(FILE *fr, long posun, int odkud);**

odkud:

- **SEEK_SET** – od začátku souboru
- **SEEK_CUR** – od aktuální pozice
- **SEEK_END** – od konce souboru

Zjištění pozice v souboru

- **long ftell(FILE *f);**

Vynucený zápis obsahu bufferu na disk

- **int fflush(FILE *f);**