Pracovní list 3: Práce se soubory

Co už máme znát

- přetypování, reference (odkaz);
- práce se standardními soubory: `cin`, `cout`, `cerr`;
- textový soubor, binární soubor;
- objekty `fstream`, `ifstream`, `ofstream` z knihovny `fstream`;
- otevření souboru, režimy souboru;
- metody `get`, `getline`, `put`, `read`, `write`;
- pozice čtení, pozice zápisu v souboru.

Kontrolní otázky

3.1 S čím je spojen proces přenosu dat ze standardních (obecně všech textových) souborů?

3.2 Proč se používají netextové (binární) soubory?

3.3 V jakých režimech lze otevřít soubor?

3.4 Jaký je rozdíl mezi `soubor.getline(x)` a `getline(soubor, x)`?

3.5 Jak se zjistí a nastaví pozice čtení v souboru?

3.6 Jak se zjistí a nastaví pozice zápisu do souboru?

Příprava na cvičení

Ve cvičení budeme potřebovat přístup k serveru akela, kde využijeme překladač jazyka C++, editor (joe, případně jiný) a schopnosti příkazového řádku. Pro analýzu obsahu binárního souboru použijeme příkaz `od -ctx1 soubor`. Zpracovávaná data jsou na serveru akela v souboru `/home/rybicka/vyuka/progt/cecko/cviceni/cv03/data.txt`, zkopírujte si je do svého adresáře, kde budete zpracovávat úlohy tohoto pracovního listu.

Řešené příklady

Příklad 3.1 Na standardním vstupu se na každém řádku nachází trojice údajů o zaměstnanci: jméno (max. 13 znaků), číslo pracoviště (celočíselná hodnota) a výkon (desetinné číslo typu `double`). Vytvořte na disku soubor záznamů s těmito údaji.

Řešení: Definujeme záznam se třemi požadovanými složkami: znakové pole o délce 13, jedna celočíselná a jedna desetinná hodnota. Z těchto záznamů vytvoříme pole, naplníme je daty ze vstupu a vypíšeme do výstupního binárního souboru. Do souboru na začátek ještě vypíšeme počet záznamů, abychom mohli při čtení tento údaj použít pro správné naplnění paměťové struktury.

```
89  #include <iostream>
90  #include <fstream>
91  using namespace std;
92
93  int main(){
94    typedef struct {
95      char jmeno[13]; //požadované složky záznamu
96      int dilna;
97      double vykon;
98    } TypZam;
99    const int MaxPrac = 50; //zvolíme podle uvážení
100   typedef TypZam TypZavod[MaxPrac];
101
102   fstream zapis("zavod.dat", ios::out | ios::binary);
103   TypZavod Zavod;
104   int Obsazeno=0;
105
106   while (cin>>Zavod[Obsazeno].jmeno>>Zavod[Obsazeno].dilna
107              >>Zavod[Obsazeno].vykon) Obsazeno++; //přečtení vstupu
108   zapis.write((char*)&Obsazeno, sizeof(Obsazeno));
109       //výpis počtu záznamů
110   zapis.write((char*)Zavod, sizeof(TypZam)*Obsazeno);
111       //výpis celého obsahu pole
112   zapis.close();
113   return 0;
114 }
```

Příklad 3.2 Na disku existuje binární soubor s uloženými řetězci ve formátu jazyka C (posloupnost znaků zakončená bajtem s hodnotou 0). Přečtěte tento soubor a vypište délku nejdelšího řetězce souboru.

Řešení: Řetězce je potřebné číst z tohoto binárního souboru bajt po bajtu. Přečtené bajty budou interpretovány jako znaky, když je budeme skládat do proměnné typu `string`. Pro přečtení jednoho bajtu ze souboru můžeme použít operaci `get` nebo obecný způsob čtení pomocí operace `read`.

```
115 #include <iostream>
116 #include <fstream>
117 using namespace std;
118
119 int main(){
120    ifstream Retezce ("retezce.dat", ios::binary);
```

```
121     string Retez;
122     char znak;
123     int Max=0;
124
125     while (not Retezce.eof()) {
126        Retez="";
127        znak=Retezce.get(); //čtení řetězce zakončeného nulou
128        while (znak!=0 and not retezce.eof()) {
129            Retez+=znak;     //přidání přečteného znaku k řetězci
130            znak=Retezce.get();
131        }
132     if (Retez.length()>Max) Max=Retez.length(); //zjištění maximální délky
133     }
134     cout << "Nejdelší řetězec má " << Max << " bajtů." << endl;
135     return 0;
136 }
```

Příklady

Příklad 3.3 Analyzujte soubor, který vznikl řešením příkladu 3.1. Jakou velikost zabírá v souboru jeden záznam? Odpovídá tato délka součtu délek jednotlivých složek záznamu? Zdůvodněte výsledky zjištění.

Příklad 3.4 Napište program, který ze zadaných dat (soubor `data.txt`) vytvoří soubor vhodný pro zpracování v příkladu 3.2. Realizací příkladu 3.2 zjistěte, jaká je maximální délka řetězce.

Příklad 3.5 Napište program, který vypíše na standardní výstup N-tý řetězec ze souboru, jehož formát odpovídá vstupu příkladu 3.2. Číslo N čtěte ze standardního vstupu.

Příklad 3.6 Napište program, který ze zadaných dat (soubor `data.txt`) vytvoří binární soubor, kde bude každý řetězec vypisován na délku 170 bajtů.

Příklad 3.7 Napište program, který vypíše na standardní výstup N-tý řetězec ze souboru, jehož formát odpovídá vstupu příkladu 3.6. Hodnota N je přečtena ze standardního vstupu. Porovnejte charakter tohoto algoritmu s identickým příkladem 3.5.

Příklad 3.8 Napište program, který opět vytvoří soubor ze zadaných dat `data.txt`, řetězce ukládá ve formátu příkladu 3.2, přitom ale zároveň umožňuje vybrat N-tý řetězec bez nutnosti přečíst předchozích $N - 1$ řetězců. Do souboru pro tento účel vložte tabulku (celočíselné pole), v níž budou uloženy indexy počátků všech řetězců. Varianta 1: tabulka o pevné délce 20 000 položek bude uložena na začátku souboru.

Příklad 3.9 Předchozí příklad proveďte ve variantě 2 – tabulka bude uložena na konci souboru a bude mít jen tolik položek, kolik řetězců je skutečně v souboru.

Příklad 3.10 Napište program, který vypíše na standardní výstup *N*-tý řetězec ze souboru, jehož formát odpovídá výstupu příkladu 3.8. Hodnota *N* je přečtena ze standardního vstupu. Porovnejte charakter tohoto algoritmu s předchozími příklady stejného zaměření.

Příklad 3.11 Napište program, který vypíše na standardní výstup *N*-tý řetězec ze souboru, jehož formát odpovídá výstupu příkladu 3.9. Hodnota *N* je přečtena ze standardního vstupu.

Co máme po cvičení umět

- Ovládání uživatelských souborů.
- Práce s binárními soubory.
- Práce s řetězci a jejich přenos do/ze souboru.
- Zacházení s pozicemi v souboru (get pointer, put pointer).

Kontrolní otázky

3.7 Jak se efektivně přenášejí data z paměti do souboru a opačně?

3.8 Jak se může projevit memory aligning?

3.9 Jak se ukládají v operační paměti řetězce ve formátu jazyka C?

3.10 Jaké jsou možnosti ukládání řetězců do souborů?

3.11 K čemu slouží get pointer a put pointer?

3.12 Jakou výhodu a nevýhodu má možnost přímého přístupu do určitého místa souboru?

3.13 Jaké výhody a nevýhody má ukládání dat ve formátu podobném jako v posledním příkladu?