Základy programování v C++ 17. cvičení
Soubory

Zuzana Petříčková

20. listopadu 2019

Přehled

Datový proud (stream)

Základní myšlenka

- data proudí ze zdroje do spotřebiče
  1. program → soubor, konzole, tiskárna
  2. soubor, klávesnice → program

Datový proud

- nástroj pro přenos dat mezi zdrojem a spotřebičem (např. souborem a programem)

Standardní datové proudy v C++ (už známe)

- **cin** ... standardní vstup (obvykle klávesnice), proměnná typu (instance třídy) **istream**
- **cout** ... standardní výstup (obvykle konzole), proměnná typu (instance třídy) **ostream**
- **cerr** ... standardní chybový výstup (obvykle konzole), proměnná typu (instance třídy)**ostream**

Práce se soubory

- se soubory můžeme pracovat podobně jako s datovými proudy **cin, cout**
- datové proudy pro soubory (v ASCII kódování) jsou deklarované v hlavičkovém souboru **fstream**:
  - **ifstream** ... proud pouze pro vstup
  - **ofstream** ... proud pouze pro výstup
  - **fstream** ... pokud chceme střídat vstup a výstup
- datové proudy pro soubory s širokými (Unicode) znaky jsou deklarované v hlavičkovém souboru **wfstream**:
  - **wifstream**, **wofstream**, **wfstream**
- sekvenční přístup k souboru
- rozlišujeme textové a binární soubory

Zápis do souboru

- **Ukázka 1:** funkce (vytvoří a) otevře soubor, zapíše do něj "Ahoj!"a soubor zase zavře.

```
#include<fstream>
...
void zapisDoSouboru(string nazev)
{
    ofstream soubor;     // vystupni datovy proud
    soubor.open(nazev); // otevru soubor
    if (soubor.is_open()) // if (soubor)
    {
        soubor << "Ahoj!" << endl;
        soubor.close();    // zavru soubor
    }
    else
    {
        cout << "chyba: soubor nelze otevrit";
        //throw (string)"soubor nelze otevrit!";
    }
}
```

Práce se soubory

Datové proudy (třídy) ifstream, ofstream, fstream

- metoda **open()** ... otevření datového proudu (souboru)
- metoda **is_open()** ... test, zda je datový proud (tj. soubor) otevřen
- metoda **close()** ... uzavření datového proudu
- operátory <<, >> a další funkce a metody, které známe pro **cin** a **cout**
  (jsou zděděné od společného předka: **ios, istream, ostream***)*
  - **getline()** ... načte řádek/kus textu ze vstupního datového proudu
  - **ignore()** ... načte a zahodí kus textu ze vstupního datového proudu
  - **clear()** ... změní/zruší chybový stav datového proudu

  ...

Název (cesta) k souboru

```
ofstream soubor1, soubor2, soubor3;

soubor1.open("text.txt"); // relativni cesta

soubor1.open(".\\slozka\\text.txt");
                              // relativni cesta (MS Windows)
soubor3.open("D:\\tmp\\text.txt");
                             // absolutni cesta (MS Windows)
...
```

Zápis do souboru

- **Ukázka 2**: funkce zapíše do souboru čísla od 1 do 10

```
void zapisDoSouboru1(string nazev)
{
    ofstream soubor;
    soubor.open(nazev);
    if (soubor.is_open()) // if (soubor)
    {
        soubor << "Zapisuji cisla: " << endl;
        for (int i = 1; i <= 10; i++)
        {
            soubor << i << endl;
        }
        soubor.close();
    }
    else {
        ... // chyba (soubor se nepodarilo otevrit)
    }
}
```

Čtení ze souboru

- **Ukázka 3**: funkce načte číslo ze souboru

```
int nactiZeSouboru(string nazev)
{
    int i=-1;
    ifstream soubor;
    soubor.open(nazev);
    if (soubor.is_open()) // if (soubor)
    {
        soubor >> i;
        if (soubor.fail())
        // if (!soubor)
            cout << "chyba: cislo se nepodarilo nacist";

        soubor.close();
    }
    else
        ... // chyba (soubor se nepodarilo otevrit)
    return i;
}
```

Čtení ze souboru

- **Ukázka 4**: funkce vypíše obsah souboru na konzoli

```
void nactiZeSouboru(string nazev)
{
    ifstream soubor;
    soubor.open(nazev);
    if (soubor.is_open()) // if (soubor)
    {
        string radek;
        while(!soubor.eof()) // neni konec souboru
        {
            getline(soubor, radek);
            cout << radek << endl;
        }
        soubor.close();
    }
    else {
    ... // chyba (soubor se nepodarilo otevrit)
    }
}
```

Načítání ze vstupního datového proudu

```
string s;
soubor >> s;                    // jedno slovo
// vsechny znaky ' ', '\t', '\n' pred slovem nacte a zahodi

getline(soubor, s);             // cely radek
// znak '\n' na konci nacte a zahodi

getline(soubor,s,'.');  // text az po znak '.'
// znak '.' na konci nacte a zahodi
```

Chybové stavy datových proudů

- datové proudy obsahují příznaky naznačující chybu
  - vyčištění chybových příznaků metodou **clear()**
- metody pro testování stavu:
  - přetížené chování jako **bool**, přetížený operátor **!**
  - good() ... načtení hodnoty se povedlo
  - fail() ... načtení hodnoty se nepovedlo
  - eof() ... je konec souboru

```
while (soubor.good()) // while (soubor) { ... }
{
    soubor >> x;
    ...
}
if(soubor.eof()) { ... }
else if(soubor.fail()) { ... }
```

Chybové stavy datových proudů

```
int x
soubor >> x;
if (!soubor) /*nacteni se nepodarilo*/
// if (soubor.fail())
{
      soubor.clear();
      // soubor.ignore(256,'\n');
}

string s;
getline(soubor, s);
if (!soubor) /*nacteni se nepodarilo
                (napr. konec souboru)*/
    ...
if (!getline(soubor, s)) /*nacteni se nepodarilo
                           (napr. konec souboru)*/
    ...
```

Metoda open()

```
ofstream soubor1;
ifstream soubor2;
fstream soubor3, soubor4;

soubor1.open("text.txt");
soubor3.open("text.txt", ios::out); // totez (alternativa)

soubor2.open("text.txt");
soubor4.open("text.txt", ios::in); // totez (alternativa)
```

- režimy otevření:
  - ios::in ... otevřít pro vstup (implicitní pro ifstream)
  - ios::out ... otevřít pro výstup (implicitní pro ofstream), přepisuje aktuální obsah souboru
  - ios::app ... výstup, za aktuální obsah souboru
  - ios::trunc ... výstup, na začátku vymaže původní obsah souboru
  - ios::in | ios::out ... otevřít pro vstup i výstup (impl. pro fstream)
  - ios::binary ... otevřít binárně (implicitní je textově)

Datový proud jako parametr funkce

```
// datovy proud predavame jako referenci!
void zapis(ostream &s, int x)
{
    s << x << " ";
}
int main()
{
    zapis(cout, 3);

    ofstream soubor;
    soubor.open("text.txt");
    if (soubor)
    {
        zapis(soubor, 3);
        soubor.close();
    }
    return 0;
}
```

Manipulátory

- speciální objekty, které je možno přidávat k operátorům <<a >>
- knihovna **iomanip**
  - manipulátory bez parametrů: **endl**, **left**, **right** (mění zarovnání),...
  - manipulátory s parametry: **setw()**, **setprecision()**, **setfill()**,...

```
const double pi = 3.1415;
soubor << pi << endl;
soubor << endl << right << setw(10) << setfill('_')
       << setprecision(3) << pi << endl;
soubor  << endl << left << setw(10) << setfill('_')
        << setprecision(3) << pi << endl;
```

Příklady

Cílem je načíst řádky vstupního souboru jeden po druhém a vypsat je do druhého souboru, ale v opačném pořadí.

1. Napište a zavolejte funkci **int pocetRadku(string nazevSouboru1)**, která spočítá a vrátí počet řádků v souboru.
2. Napište a zavolejte funkci **string* nactiRadky(string nazevSouboru1, int n)**, která načte prvních n řádků souboru do dynamického pole, pole vrátí.
3. Napište a zavolejte funkci **void vypisRadkyOpacne(string *radky, int n, string nazevSouboru2)**, která vypíše jednotlivé řádky z dynamického pole do souboru **v opačném pořadí**.

Lepší řešení: využít např. ,,ChytrePole"

Příklady ... na procvičení

1. Napište a zavolejte funkci **string* nactiSlova(string nazevSouboru, int &n)**, která načte jednotlivá slova souboru do dynamického pole, pole vrátí (soubor obsahuje slova oddělená mezerami).
2. Napište a zavolejte funkci **void vypisSlovaOpacne(string *slova,int n, string nazevSouboru)**, která vypíše jednotlivá slova z dynamického pole do souboru **v opačném pořadí**.