

Робота з файловими потоками

Щоб програма могла взаємодіяти з файлом, необхідно використовувати змінну типу файловий потік. Така змінна задається ключовим словом `fstream`.

Для роботи з файловим потоком необхідно:

1. Підключити бібліотеку `fstream`:

`#include <fstream>`

2. Оголосити змінну типу файловий потік:

`fstream f;`

3. Відкрити файл:

– для запису у файл:

`f.open("1.txt", ios::out);`

– для читання з файлу:

`f.open("1.txt", ios::in);`

4. Провести запис у файл або читання з файлу:

– для запису у файл:

`f << "x=" << x;`

– для читання из файлу:

`f >> x;`

5. Закрити файл: **`f.close();`**

Примітка. При відкриванні файлу для запису файл створюється в папці з проектом. Якщо файл уже існує, то весь зміст стирається. Якщо потрібно додати текст в кінець уже існуючого файлу, то при відкритті файлу потрібно використати рядок:

`f.open("1.txt", ios::app);`

Приклад 1. Зчитати з файлу число та вивести його на екран.

```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    double x;
    fstream f;
    f.open("1.txt", ios::in);
    f >> x;
    f.close();
    cout << "x = " << x << endl;
    return 0;
}
```

Якщо файл лежить в каталозі, тоді `f.open("FILES/1.txt", ios::in);`

Приклад 2. Скласти програму, що виконує запис двох чисел в файли *.txt та *.xls.

Оголошуємо два файлових потоки ftxt - для запису в текстовий файл 2.txt, fxls - для запису в табличний файл 2.xls.

Зробимо так, щоб у файлі 2.txt появилася надпис:

a=7.2 b=-10.89

У файлі 2.xls виведемо кожне значення повідомлення в різні комірки. Для цього потрібно використовувати символ табуляції “\t”

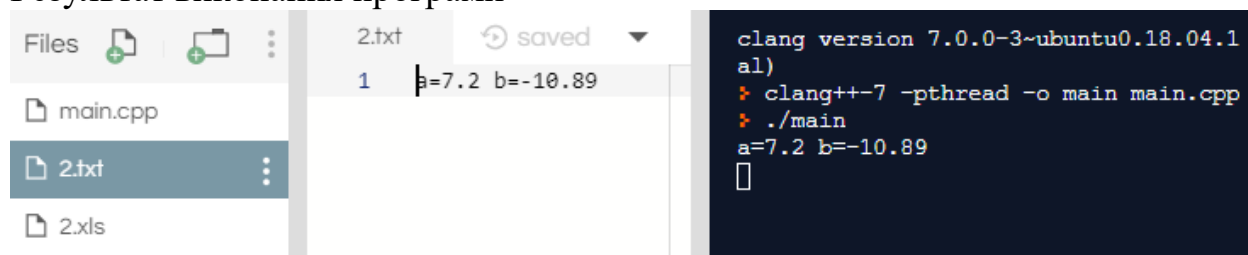
Код програми для прикладу 2:

```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    double a, b;
    fstream ftxt, fxls;
    a=7.2;
    b=-10.89;
    cout<<"a="<<a<<" b="<<b<<endl;
    ftxt.open("2.txt", ios::out);
    fxls.open("2.xls", ios::out);
    ftxt<<"a="<<a<<" b="<<b;
    fxls<<"a=\t"<<a<<"\tb=\t"<<b;
    ftxt.close();
    fxls.close();

    return 0;
}
```

Результат виконання програми



	A	B	C	D	E
1	a=	7,2	b=	-10,89	
2					

Приклад 3. Побудувати таблицю значень функції $y = \sin(x)$ при $0 \leq x \leq 2$ з кроком 0.1.

Даний приклад розв'язується за допомогою циклічного алгоритму. Будемо використовувати цикл `for`. Побудуємо таблицю значень функції на екрані, а також у файлі `1.xls` та побудуємо графік в MS Excel.

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <iomanip>

using namespace std;

int main()
{
    double x, y;
    fstream f;
    f.open("1.xls", ios::out);
    cout << setw(10) << "x" << setw(10) << "y" << endl;
    f << "x" << "\t" << "y" << endl;
    for (x=0; x<=2; x+=0.1){
        y=sin(x);
        cout << setw(10) << x << setw(10) << y << endl;
        f << x << "\t" << y << endl;
    }
    f.close();

    return 0;
}
```

```
main.cpp  saved
1  #include <iostream>
2  #include <fstream>
3  #include <math.h>
4  #include <iomanip>
5
6  using namespace std;
7
8  int main()
9  {
10     double x, y;
11     fstream f;
12     f.open("1.xls", ios::out);
13     cout << setw(10) << "x" << setw(10) << "y" << endl;
14     f << "x" << "\t" << "y" << endl;
15     for (x=0; x<=2; x+=x+0.1){
16         y=sin(x);
17         cout << setw(10) << x << setw(10) << y << endl;
18         f << x << "\t" << y << endl;
19     }
20     f.close();
21
22     return 0;
23 }
24
```

```
clang version 7.0.0-3~ubuntu0
RELEASE_700/final)
❏ clang++-7 -pthread -o main
❏ ./main
      x          y
      0          0
    0.1 0.0998334
    0.2 0.198669
    0.3 0.29552
    0.4 0.389418
    0.5 0.479426
    0.6 0.564642
    0.7 0.644218
    0.8 0.717356
    0.9 0.783327
    1 0.841471
    1.1 0.891207
    1.2 0.932039
    1.3 0.963558
    1.4 0.98545
    1.5 0.997495
    1.6 0.999574
    1.7 0.991665
    1.8 0.973848
    1.9 0.9463
```

Для реалізації файлового введення/виведення потрібно підключити заголовок **<fstream>**. В ньому визначено декілька класів:

ifstream

ofstream

fstream

Перший використовується для вводу, другий - для виводу, третій - для вводу та виводу. Перед відкриттям файлу, необхідно створити потік. Далі створюються потоки вводу, виводу та вводу/виводу:

ifstream in;

ofstream out;

fstream io;

Після цього потрібно використати функцію **open()** безпосередньо для відкриття файлу. Цій функції необхідно передати ім'я файлу та режим відкриття:

void ifstream::open(const char *имя_файла, openmode режим);

void ofstream::open(const char *имя_файла, openmode режим);

void fstream::open(const char *имя_файла, openmode режим);

ios::app - використовується для відкриття файлу з метою додавання інформації в його кінець. Застосовується тільки до файлів, відкритих для виводу.

ios::ale - викликає пошук кінця файлу, но операції вводу/виводу можуть бути виконані в будь-якій частині файлу.

ios::binary - двійковий режим. По замовченню усі файли відкриваються в

текстовому режимі, в якому має місце перетворення деяких символів, наприклад, послідовність символів `\r\n` перетворюється в `\n`. Якщо файл відкривається в двійковому режимі, такого перетворення немає. Будь-який файл може бути відкритий, як в текстовому, так й в двійковому режимі. Різниця тільки в наявності або відсутності перетворення.

`ios::in, ios::out`; - задає режим для вводу або режим для виводу.

`ios::trunk` - приводить до видалення змісту файлу при його відкритті та приведення його до нулевої довжини.

//Листинг 12.1. Запис в файл

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    // створемо потік та зразу відкриваємо файл
    ofstream fout("test.txt");
    if (!fout) {
        cout << "Не можу відкрити файл для запису return 1";
    }
    // Записуємо в потік два рядки
    fout << "Hello, world!\n";
    fout << "Yet another line\n";
    fout.close();
    cout << "Готово!" << endl;
    return 0;
}
```

//Лістинг 12.2. Читання інформації

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
    // відкриваємо потік для вводу
    ifstream fin ("test1.txt") ;
    // не пропускати пробіли
    fin.unsetf (ios:: skipws) ;
    if (!fin) {
        cout << "Не можу відкрити файл для читання return 1";
    }
    char ch;
    // читаємо файл посимвольно
    while (!fin.eof ()) {
        fin >> ch;
        cout << ch;
    }
    cout << endl;
    fin. close () ;
    return 0;}
```

Продemonструємо роботу наступного оператора: **fin.unself (ios::skipws)** (не пропускати пробіли)

```
den@den-pc:~/cpp-ex$ cat test.txt
Hello, world!
Yet another line
den@den-pc:~/cpp-ex$
```

Попередня програма читає файл посимвольно. Це не завжди зручно. На практиці часто потрібно читати файл порядком. Для цього використовується функція **getline()**. Приклад порядкового читання файлу приведений в листингу 12.3. В циклі **while** виконується порядкове читання - з файлу. Читається рядок та присвоюється змінній **s**, зміст якої виводиться на екран. Далі до рядка додається попередній рядок та модифікований рядок також виводиться на екран (консоль).

```
//Лістинг 12.3. Порядкове читання файлу на C++
#include <iostream> // консольний ввід/вивід
#include <string> // операції з рядками
#include <fstream> // файловий ввід/вивід
using namespace std;

int main () {
    string s, str=""; // сюди будемо читати рядки з файлу
    ifstream file ("test.txt") ;
    // пока не кінець файлу читати рядок з потіку
    // file в рядкову змінну s
    while (getline (file, s)) {
        cout << s << endl;
        str = str + " " + s;
        cout << str << endl;
        // виводимо s на екран
        // що небудь робимо з рядком
        // знов виводимо
    }
    file.close ();
    // закриваємо потік
    return 0;
}
```

1. Алгоритмизация. Введение в язык программирования C++/ И.Е. Белоцерковская, Н.В. Галина, Л.Ю. Катаева - М.: Национальный Открытый Университет "ИНТУИТ", 2016
2. Орленко П. А., Евдокимов П. В. C++ на примерах. Практика, практика и только практика.СПб.: Наука и Техника, 2019. 288 с., ил.