

# Алгоритмизация и программирование

Программирование на C/C++  
(ч.11 – файлы)



Беркунский Е.Ю., кафедра ИУСТ, НУК  
eugeny.berkunsky@gmail.com  
<http://www.berkut.mk.ua>

# Файлы

- Что такое файл?
- Файл – именованный набор байтов, который может быть сохранен на некотором накопителе.
- Другими словами, под файлом понимается последовательность байтов, записанных на диск, которая имеет своё, уникальное имя, например **file.txt**.

# Файлы

- В одном каталоге не могут находиться файлы с одинаковыми именами.
- Под именем файла понимается не только его название, но и расширение, например: `file.txt` и `file.dat` - разные файлы, хоть и имеют одинаковые названия.
- Существует такое понятие, как полное имя файлов – это полный путь к каталогу файла с указанием имени файла, например:  
`D:\docs\file.txt`.

# Файлы

- Для работы с файлами необходимо подключить заголовочный файл `<fstream>`.
- В `<fstream>` определены несколько классов и подключены заголовочные файлы
  - `<ifstream>` - файловый ввод и
  - `<ofstream>` - файловый вывод.

# Файлы

- Файловый ввод/вывод аналогичен стандартному вводу/выводу, единственное отличие – это то, что ввод/вывод выполнятся не на экран, а в файл.
- Если ввод/вывод на стандартные устройства выполняется с помощью объектов `cin` и `cout`, то для организации файлового ввода/вывода достаточно создать собственные объекты, которые можно использовать аналогично тому, как использовались `cin` и `cout`.



- Например, необходимо создать текстовый файл и записать в него строку "Работа с файлами в C++".
- Для этого необходимо проделать следующие шаги:
  1. создать объект класса ofstream;
  2. связать объект класса с файлом, в который будет производиться запись;
  3. записать строку в файл;
  4. закрыть файл.

# Файлы

```
// создаём объект для записи в файл  
ofstream /*имя объекта*/; // объект класса ofstream
```

Назовём объект – fout:

```
ofstream fout;
```

Для чего нужен объект?

- Объект необходим, чтобы можно было выполнять запись в файл.
- Уже объект создан, но не связан с файлом, в который нужно записать строку.

```
fout.open("example.txt");  
// связываем объект с файлом
```

```
ofstream fout;  
fout.open("example.txt");
```

- Через операцию **точка** получаем доступ к методу класса `open()`, в круглых скобках которого указываем имя файла.
- Указанный файл будет создан в текущей директории с программой.
- Если файл с таким именем существует, то существующий файл будет заменен новым.

```
fout << "Работа с файлами в C++";  
// запись строки в файл
```

Используя операцию `<<` с объектом `fout`, строка “Работа с файлами в C++” записывается в файл.



```
ofstream fout;  
fout.open("example.txt");  
fout << "Работа с файлами в C++";
```

Так как больше нет необходимости изменять содержимое файла, его нужно закрыть, то есть отделить объект от файла.

```
fout.close(); // закрываем файл
```

Итог – создан файл со строкой “Работа с файлами в C++”.

Шаги 1 и 2 можно объединить, то есть в одной строке создать объект и связать его с файлом:

```
ofstream fout("example.txt");
```

В итоге получим такую программу:

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

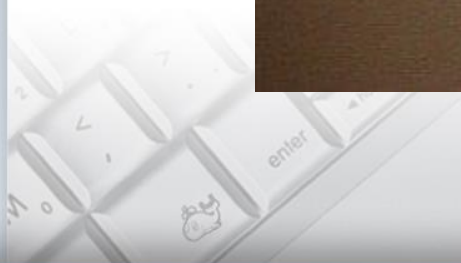
int main()
{
    ofstream fout("example.txt");
    fout << "Работа с файлами в C++";
    fout.close();
    return 0;
}
```

Осталось проверить правильность работы программы, а для этого открываем файл example.txt:

Работа с файлами в C++



# Демонстрація



# Файлы

Для того чтобы прочитать файл понадобится выполнить те же шаги, что и при записи в файл с небольшими изменениями:

- создать объект класса **ifstream** и связать его с файлом, в который будет производиться запись;
- прочитать файл;
- закрыть файл.

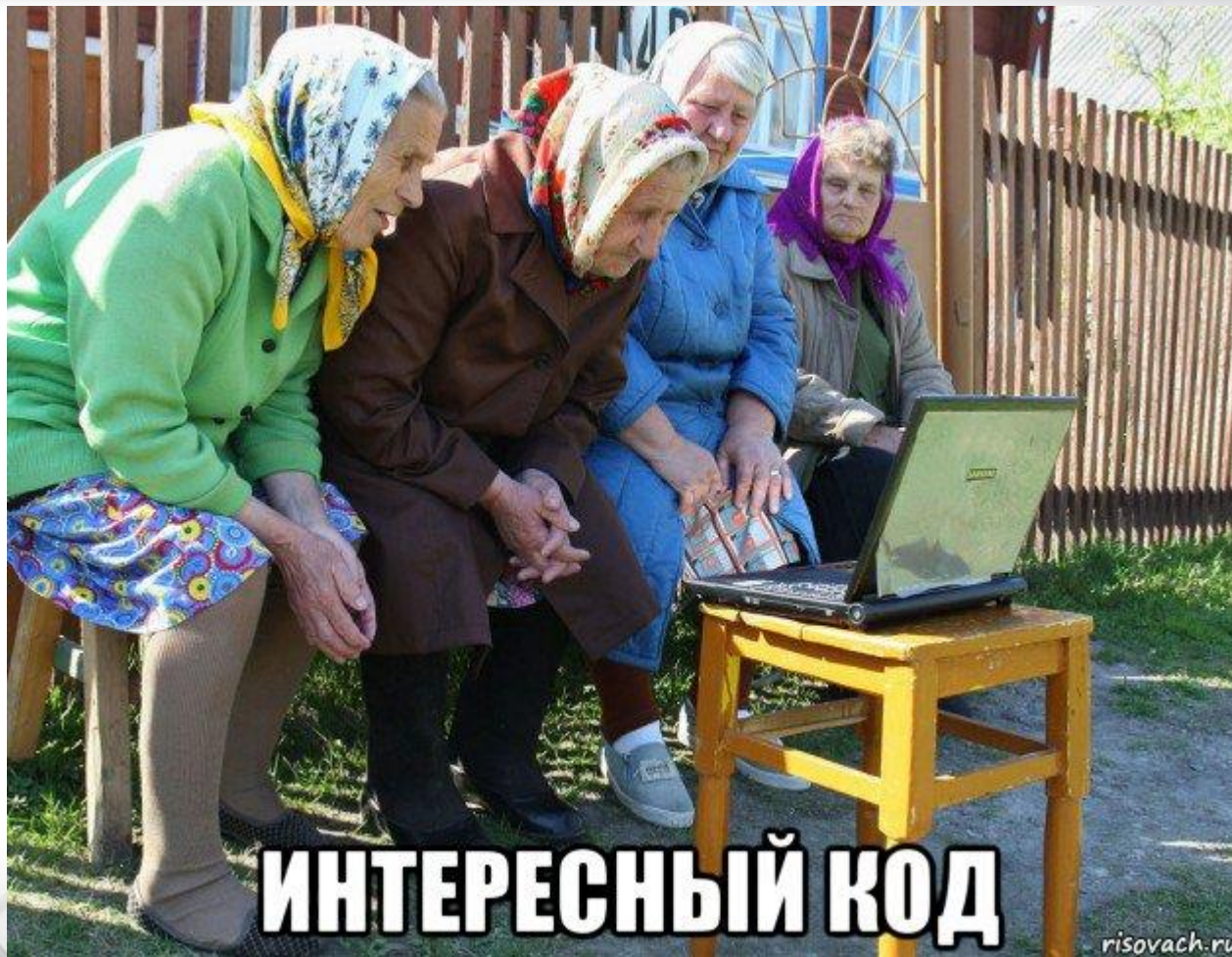
```
#include <fstream>
#include <iostream>
#include <windows.h>

using namespace std;

int main()
{
    SetConsoleCP(1251); SetConsoleOutputCP(1251);
    char buff[50];
    ifstream fin("example.txt");
    fin >> buff;
    cout << buff << endl;
    fin.getline(buff, 50);
    fin.close();
    cout << buff << endl;
    return 0;
}
```



# Демонстрація



**ИНТЕРЕСНЫЙ КОД**



# Файлы

- Програма сработала правильно, но не всегда так бывает, даже в том случае, если с кодом всё в порядке.
- Например, в программу передано имя несуществующего файла или в имени допущена ошибка.
- В этом случае ничего не произойдёт вообще.
- Файл не будет найден, а значит и прочитать его не возможно.
- Поэтому компилятор проигнорирует строки, где выполняется работа с файлом.
- В результате корректно завершится работа программы, но на экране ничего показано не будет.

# Файлы

- Простому пользователю не будет понятно, в чём дело и почему на экране не появилась строка из файла.
- Чтобы отреагировать в такой ситуации, в C++ предусмотрена специальная функция - `is_open()`, которая возвращает целые значения:
  - 1 — если файл был успешно открыт,
  - 0 — если файл открыт не был.
- Доработаем программу с открытием файла, таким образом, что если файл не открыт выводилось соответствующее сообщение.





```
#include <fstream>
#include <iostream>

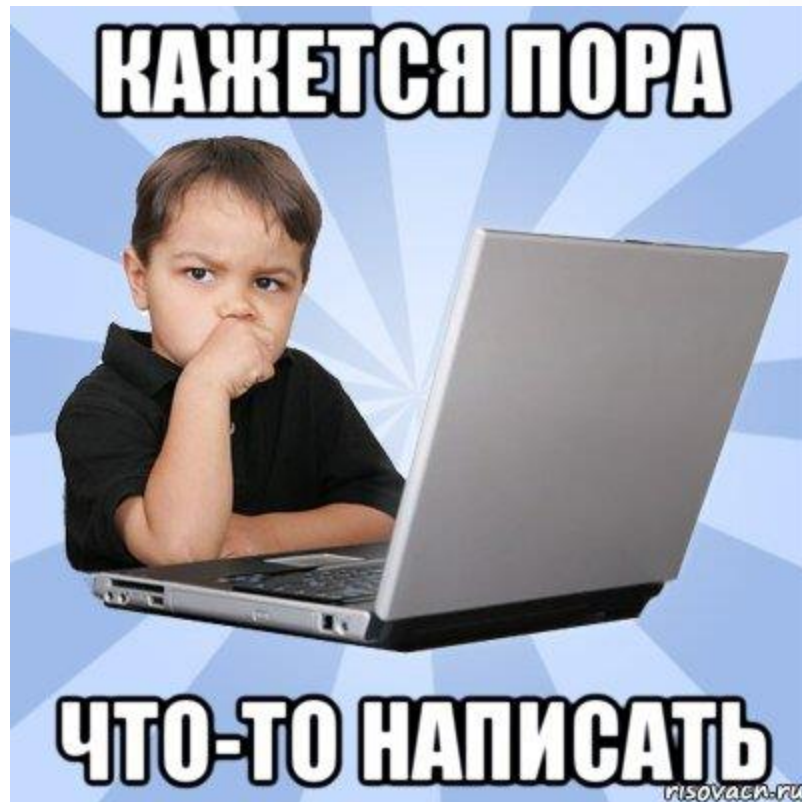
using namespace std;

int main()
{
    char buff[50];
    ifstream fin("example.txt");
    if (!fin.is_open()) // если файл не открыт
        cout << "Файл не может быть открыт!\n";
    else {
        fin >> buff;
        cout << buff << endl;
        fin.getline(buff, 50);
        fin.close();
        cout << buff << endl;
    }
    return 0;
}
```



НАЦІОНАЛЬНИЙ  
УНІВЕРСИТЕТ  
КОРАБЛЕБУДУВАННЯ  
ІМЕНІ АДМІРАЛА МАКАРОВА

# Демонстрація



# Режимы открытия файлов

Константа	Описание
<code>ios_base::in</code>	открыть файл для чтения
<code>ios_base::out</code>	открыть файл для записи
<code>ios_base::ate</code>	при открытии переместить указатель в конец файла
<code>ios_base::app</code>	открыть файл для записи в конец файла
<code>ios_base::trunc</code>	удалить содержимое файла, если он существует
<code>ios_base::binary</code>	открытие файла в двоичном режиме

# Режимы открытия файлов

Режимы открытия файлов можно устанавливать при создании объекта или при вызове функции `open()`

```
// открываем файл для добавления информации
// в конец файла
ofstream fout("example.txt", ios_base::app);
fout.open("example.txt", ios_base::app);
```

Режимы открытия файлов можно комбинировать с помощью поразрядной логической операции «или» - `|`, например: `ios_base::out | ios_base::trunc` - открытие файла для записи, предварительно очистив его.

# Режимы по умолчанию

- Объекты класса **ofstream**, при связке с файлами по умолчанию содержат режимы открытия файлов **ios\_base::out | ios\_base::trunc**.
- То есть файл будет создан, если не существует.
- Если же файл существует, то его содержимое будет удалено, а сам файл будет готов к записи.
- Объекты класса **ifstream** связываясь с файлом, имеют по умолчанию режим открытия файла **ios\_base::in** - файл открыт только для чтения.
- Режим открытия файла ещё называют — “флаг”.



НАЦІОНАЛЬНИЙ  
УНІВЕРСИТЕТ  
КОРАБЛЕБУДУВАННЯ  
ІМЕНІ АДМІРАЛА МАКАРОВА



# Алгоритмизация и программирование

Программирование на C/C++  
(ч.11 – файлы)



Беркунский Е.Ю., кафедра ИУСТ, НУК  
eugeny.berkunsky@gmail.com  
<http://www.berkut.mk.ua>