**Лекция: Работа с файлами**

Большинство компьютерных программ работают с файлами, и поэтому возникает необходимость создавать, удалять, записывать читать, открывать файлы.

Файл – именованный набор  байтов, который может быть сохранен на некотором накопителе, которая имеет своё, уникальное имя, например файл.txt. В одной директории  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 адрес к директории файла с указанием имени файла, например: D:\docs\file.txt.

Для организации работы с файлами необходимо усвоить такое понятие как *поток – абстрактный канал связи, который создается в программе для обмена данными с файлами.*

*Текстовый поток* — это последовательность символов. При передаче символов из потока на экран, часть из них не выводится (например, символ возврата каретки, перевода строки).

*Двоичный поток* — это последовательность байтов, которые однозначно соответствуют тому, что находится на внешнем устройстве.

Рассмотрим работу с текстовыми файлами.

В C++ отсутствуют операторы для работы с файлами. Все необходимые действия выполняются с помощью функций, включенных в стандартную библиотеку. Они позволяют работать с различными устройствами, такими, как диски, принтер, коммуникационные каналы и т.д. Эти устройства сильно отличаются друг от друга. Однако файловая система преобразует их в единое абстрактное логическое устройство, называемое потоком.

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.

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 операторам cin иcout.

Например, необходимо создать текстовый файл и записать в него строку Работа с файлами в С++. 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.

Почему необходимо создавать объект класса ofstream, а не класса ifstream? Потому, что нужно сделать запись в файл, а если бы нужно было считать данные из файла, то создавался бы объект класса ifstream.

|  |  |
| --- | --- |
|  | // создаём объект для записи в файл  ofstream /\*имя объекта\*/; // объект класса ofstream |

Назовём объект – fout, Вот что получится:

|  |  |
| --- | --- |
|  | ofstream fout; |

Для чего нам 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("cppstudio.txt"); // связываем объект с файлом |

Через операцию точка получаем доступ к методу класса open(), в круглых скобочках 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. Итак, файл открыт, осталось записать в него нужную строку. Делается это так:

|  |  |
| --- | --- |
|  | fout << "Работа с файлами в С++"; // запись строки в файл |

Используя операцию передачи в поток совместно с объектом fout строка Работа с файлами в С++ записывается в файл. 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 С++.

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

|  |  |
| --- | --- |
|  | ofstream fout("cppstudio.txt"); // создаём объект класса ofstream  и связываем его с файлом cppstudio.txt |

Объединим весь код и получим следующую программу.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | #include <fstream>  using namespace std;    int main()  {      ofstream fout("cppstudio.txt"); // создаём объект класса  // ofstream для записи и связываем его с файлом cppstudio.txt      fout << "Работа с файлами в С++"; // запись строки в файл      fout.close(); // закрываем файл      return 0;  } |

Осталось проверить правильность работы программы, а для этого открываем файл cppstudio.txt и смотрим его содержимое, должно быть — Работа с файлами в С++.

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

1. создать объект класса ifstream и связать его с файлом, из которого будет производиться считывание;
2. прочитать файл;
3. закрыть файл.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | #include <iostream>  #include <fstream>  #include <cstring>  using namespace std;    int main()  {      setlocale(LC\_ALL, "rus"); // корректное отображение Кириллицы      char buff[50]; // буфер промежуточного хранения считываемого из //файла текста      ifstream fin("cppstudio.txt"); // открыли файл для чтения       fin >> buff; // считали первое слово из файла      cout << buff << endl; // напечатали это слово       fin.getline(buff, 50); // считали строку из файла      fin.close(); // закрываем файл      cout << buff << endl; // напечатали эту строку        return 0;  } |

В программе показаны два способа чтения из файла, первый – используя операцию передачи в поток, второй – используя функцию getline(). В первом случае считывается только первое слово, а во втором случае считывается строка, длинной 50 символов. Но так как в файле осталось меньше 50 символов, то считываются символы включительно до последнего. Обратите внимание на то, что считывание во второй раз (**строка 16**) продолжилось, после первого слова, а не с начала, так как первое слово было прочитано в **строке 14**.

**Общая схема организации ввода-вывода такова:**

1. Создать объект для организации ввода-вывода.

2. Убедиться в том, что объект создан успешно (с помощью метода is\_open()).

3. Выполнить необходимые операции по вводу данных (с помощью оператора >>) или по выводу данных (с помощью оператора <<).

4. Закончить работу с объектом (с помощью метода close()).

**Пример 1. Вывод данных в файл.**

// 1. Создать объект для вывода данных в файл

|  |  |  |
| --- | --- | --- |
| ofstream out("student1.txt"); | ofstream OutFile("out.txt"); | ofstream OutFile("function.txt"); |

// 2. Убедиться, что объект создан успешно

|  |  |  |
| --- | --- | --- |
| if (out.is\_open()) | if (OutFile.is\_open()) | if (OutFile.is\_open()) |

{

// 3. Выполнить операции по выводу данных

|  |  |  |
| --- | --- | --- |
| out <<“Good morning!<< endl;  out << “ Be happy!!!” << endl; | for ( i = 0; i < 10; i++ )  {  OutFile << i <<"\n"; // Запись  } | for ( int i = 0; i <= 10; i++ )  {  // Записываем в файл очередное //значение аргумента и функции  OutFile << dA + i \* dDeltaX << ‘\t’  << pow(dA + i \* dDeltaX, 2 ) <<"\n";  } |

// 4. Закончить работу с объектом

|  |  |  |
| --- | --- | --- |
| out.close(); | OutFile.close(); // Закрыть поток | OutFile.close();// Закрыть поток |

}

else

{

cerr << "Ошибка вывода в файл";

}

**Пример 2. Ввод данных из файла.**

// 1. Создать объект для ввода данных из файла

|  |  |  |
| --- | --- | --- |
| ifstream in("student.txt"); | ifstream InFile("out.txt"); | ifstream InFile("function.txt"); |

// 2. Убедиться, что объект создан успешно

|  |  |  |
| --- | --- | --- |
| if (in.is\_open()) | if (InFile.is\_open()) | if (InFile.is\_open()) |

{

// 3. Выполнить операции по вводу данных

|  |  |  |
| --- | --- | --- |
| char str[50];  in >> str;  in.getline(str,50); | for ( i = 0; i < 10; i++ )  {  // Чтение данных из файла в //массив nArray  InFile >> nArray[i];  // Вывод прочитанных данных на //экран  cout << nArray[i] << '\n';  } | for ( int i = 0; i <= 10; i++ )  {  // Читаем и в массив значение //аргумента  InFile >> nArray[i][0];  // Читаем и в массив значение //функции  InFile >> nArray[i][1];  // Выводим данные на экран  cout << nArray[i][0] << ‘\t’  << nArray[i][1] << "\n";  } |

// 4. Закончить работу с объектом

|  |  |  |
| --- | --- | --- |
| in.close(); | InFile.close(); // Закрыть поток | InFile.close(); |

in.close();

}

else

{

cerr << "Ошибка ввода из файла";

}

Задача 1. Дана функция: *y* = *x*2. Провести табуляцию функцию на интервале [*a*,*b*] с шагом Δ*x* = (b - a)/10. Занести результат в текстовый файл. Прочитать из файла значения табулированной функции и вывести их на экран.

#include <iostream>

#include<iomanip>

#include <fstream>

#include <math.h>

int main()

{

// Переменные для границ интервала и шага аргумента

double dA, dB, dDeltaX;

int i;

// Ввод данных

cout << "Please input a, b (b > a)\n";

cin >> dA >> dB;

// Проверка корректности введенных данных

if ( dB <= dA )

{

cout << "Wrong inteval\n";

return;

}

// Расчет шага аргумента

dDeltaX = (dB - dA)/10.;

// Открываем поток

ofstream OutFile("function.txt");

for ( i = 0; i <= 10; i++ )

{

// Записываем в файл очередное значение аргумента и функции

OutFile << setw(5)<< dA + i \* dDeltaX

<< setw(5)<< pow(dA + i \* dDeltaX, 2 ) <<endl;

}

// Закрываем поток

OutFile.close();

/\*Пример чтения информации из файла\*/

// Массив для хранения значений аргумента и функции

double nArray[11][2];

// Открываем поток

ifstream InFile("function.txt");

for ( i = 0; i <= 10; i++ )

{

// Читаем и в массив значение аргумента

InFile >> nArray[i][0];

// Читаем и в массив значение функции

InFile >> nArray[i][1];

// Выводим данные на экран

cout << setw(5)<< nArray[i][0] << setw(5)<< nArray[i][1] << endl;

}

// Закрываем поток

InFile.close();

return 0;

}