



Семинар 9

Работа с файлами в языке Си++



Файлы в Си++

Для работы с файлами в Си++:

```
#include <fstream>
```

В этом файле описаны следующие классы:

- `ifstream` – класс входных файловых потоков
- `ofstream` – класс выходных файловых потоков
- `fstream` – класс двунаправленных файловых потоков



Файлы в Си++

В классе `base_ios` имеется группа флагов для задания режима работы с файлами:

app – запись данных в конец файла

ate – после открытия файла выполняется позиционирование в конец файла

binary – обмен в бинарном режиме

in – открытие для чтения (по умолчанию для потоков `ifstream`)

out – открытие для записи (по умолчанию для потоков `ofstream`)

trunc – удалить предыдущее содержимое



Файлы в Си++

Соответствие режимов работы с файлами в Си и Си++

w	std::ios::out std::ios::trunc Режим записи. Если файл существовал, содержимое стирается.
r	std::ios::in Режим чтения.
a	std::ios::out std::ios::app Режим записи в конец файла.
w+	std::ios::out std::ios::in std::ios::trunc Режим записи и чтения. Если файл существовал, содержимое стирается.
r+	std::ios::out std::ios::in
a+	std::ios::in std::ios::out std::ios::app



Файлы в Си++

При создании файлового потока вызывается конструктор:

```
ofstream имя_потока("Имя_файла", режим);
```

или

```
ifstream имя_потока("Имя_файла", режим);
```

например,

```
ofstream file1("myfile.txt",  
std::ios::out|std::ios::app);
```



Файлы в Си++

Методы классов файловых потоков:

`void close();` //Заккрытие файла

// Два метода для открытия файла:

`void open(char * filename, режим);`

`void open(char * filename);`

Во втором случае режим работы с файлом задаётся по умолчанию в соответствии с именем файла.

`bool is_open();`



Файлы в Си++

Методы классов файловых потоков
(продолжение):

// Проверка открытия файла (открыт ли файл
для потока, для которого метод вызван)
`bool is_open();`

// Проверка достижения конца файла
`bool eof();`



Файлы в Си++

Методы классов файловых потоков
(продолжение):

// Сброс флагов состояния

```
void clear(iostate state)
```

Возможные аргументы для этого метода:
badbit, eofbit, failbit, goodbit. Возможно
применение этого метода без параметров.



Файлы в Си++

Если содержимое файла было прочитано, и возникла необходимость прочитать содержимое файла повторно, то необходимо применить следующие методы:

```
file2.clear();
```

```
file2.close();
```

```
file2.open("Имя файла");
```

```
// далее – произвести чтение из файла
```



Файлы в Си++

Пример. Запись данных в файл.

```
#include "stdafx.h"
```

```
#include <iostream>
```

```
#include <fstream>
```

```
using namespace std;
```

```
struct person{
```

```
char firstName[30];
```

```
char secondName[30];
```

```
char phone[30];
```

```
unsigned age;
```

```
};
```



Файлы в Си++

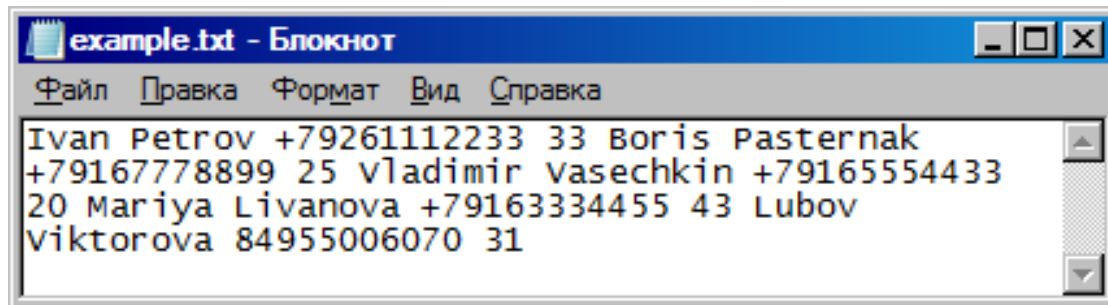
Пример. Запись данных в файл.

```
int main()
{
    person P;
    ofstream outFile("example.txt");
    for(int i=0; i<5; i++){
        cout<<"Enter new data:";
        cin>>P.firstName>>P.secondName>>P.phone>>P.age;
        outFile << P.firstName << ' ' << P.secondName << ' ' <<
        P.phone << ' ' << P.age << ' ';
    }
    outFile.close(); return 0;
}
```



Файлы в Си++

Пример. Запись данных в файл.



```
example.txt - Блокнот
Файл  Правка  Формат  Вид  Справка
Ivan Petrov +79261112233 33 Boris Pasternak
+79167778899 25 vladimir Vasechkin +79165554433
20 Mariya Livanova +79163334455 43 Lubov
viktorova 84955006070 31
```



Файлы в Си++

Пример. Чтение данных из файла.

```
person P;  
ifstream inFile("example.txt");  
for(int i=0; i<5; i++){  
    inFile>>P.firstName>>P.secondName>>P.phone>>P.age;  
    cout<<"\nData: ";  
    cout << P.firstName << ' ' << P.secondName << ' ' <<  
    P.phone << ' ' << P.age << ' ' << '\n';  
}  
getchar();  
inFile.close();
```



Файлы в Си++

Пример. Чтение данных до конца файла.

```
person P;  
ifstream inFile("example.txt");  
while(!inFile.eof()){  
    inFile>>P.firstName>>P.secondName>>P.phone>>P.age;  
    cout<<"\nData: ";  
    cout << P.firstName << ' ' << P.secondName << ' ' <<  
    P.phone << ' ' << P.age << ' ';&br/>}  
inFile.close();
```

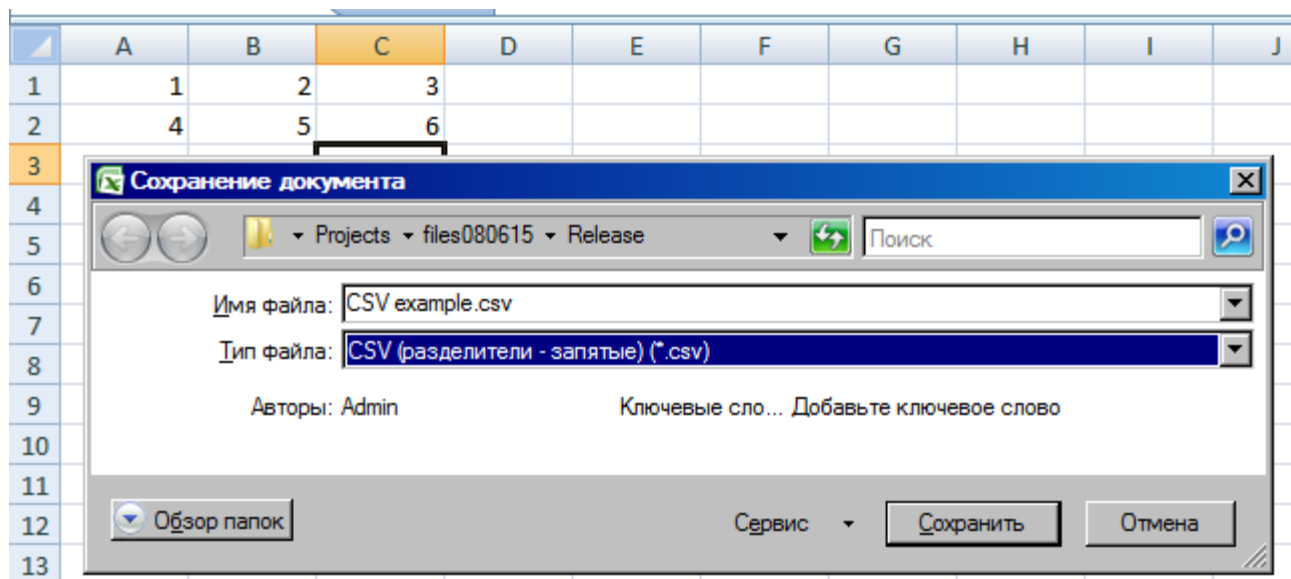
Для корректной работы этого примера из файла нужно удалить последний пробел.



CSV-файлы

CSV (comma separated values) – файлы, содержащие данные, разделённые некоторыми символами.

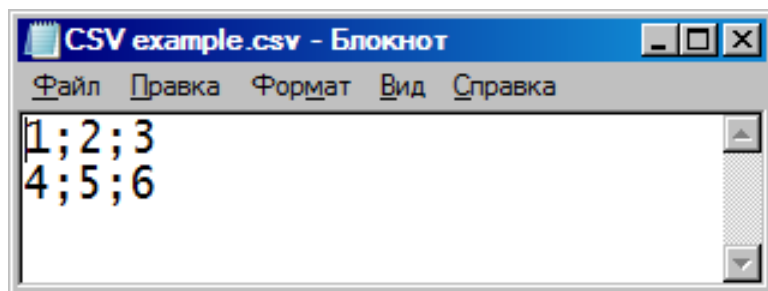
Пример данных и их сохранения в формате CSV:





CSV-файлы

... а вот как сохранены эти данные:



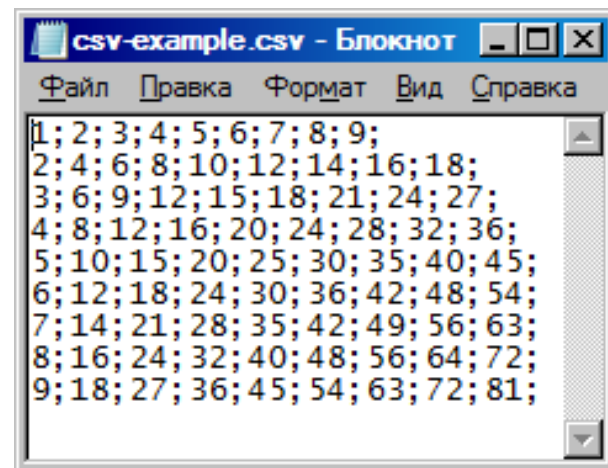
Резюме: если нужно, чтобы таблицу с данными можно было открыть в Excel (или в другой программе для работы с электронными таблицами), сохраните эти данные с расширением .csv и разделите их символом «точка с запятой»



CSV-файлы

Пример. Сохранение таблицы умножения в формате CSV.

```
ofstream inFile("csv-example.csv");  
for(int i=1; i<=9; i++){  
    for(int j=1; j<=9; j++){  
        inFile<<i*j<<',';  
        inFile<<'\n';  
    }  
}
```



	A	B	C	D	E	F	G	H	I
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

CSV-файлы



Задания:

1. Записать в CSV-файл таблицу степеней двойки до 20-й степени.
2. Ввести и вывести из файла данные о книгах (имя, фамилия автора, издательство, год издания, количество страниц, стоимость).