# ФАЙЛЫ

## Классы ifstream, ofstream, fstream

## basic_ifstream, basic_ofstream, basic_fstream

```
<Класс> <Переменная>;
```

```
<Класс> <Переменная> (<Путь к файлу> [, <Режим>
                                      [, <Блокировка>]]);
```

```
open()
```

```
std::ofstream file;
file.open("data.txt");
```

```
std::ofstream file("C:\\Task_file\\data.txt");
```

```
std::ofstream file("data.txt");
```

- ifstream – ios::in (ios::failbit);
- ofstream – ios::out
- fstream – ios::in | ois::out (ios::failbit)

- ios::in
- ios::out

```
std::fstream file("data_.txt", std::ios::out); // 123456789
file << "***"; // ***
```

```
std::fstream file("data_.txt",std::ios::out | std::ios::in); // 123456789
file.seekg(3, std::ios::cur);
file << "***"; // 123***789
```

- ios::trunc (ios::out, ios::in, ios::app, ios::ate)
- ios::app
- ios::ate (seek());
- ios::binary

**share.h**

- _SH_DENYRW – 0x10
- _SH_DENYWR – 0x20
- _SH_DENYRD – 0x30
- _SH_DENYNO – 0x40

## Открытие и закрытие файла

Прототип метода:

```
void open(<Путь к файлу>[, <Режим>[, <Блокировка>]]);
```

*Пример* открытия файла на запись:

```
std::ofstream file;
file.open("data.txt");
```

Этот код эквивалентен следующему коду:

```
std::ofstream file("data.txt");


if (!file) cout << "Файл не удалось открыть\n";
if (!file.is_open()) cout << "Файл не удалось открыть\n";
```

## Запись в файл и чтение из файла

Методы, предназначенные для **записи**:

**<<**

**put()**

```
char ch='A';
file.put(ch);
```

**write()**

Прототип метода:

```
ostream &write(const char *Str, streamsize Count);
```

*Пример*. Запись структуры в файл, открытый на запись в бинарном режиме.

```
std::ofstream file;
file.open("data_binary.txt", std::ios::out | std::ios::binary);

struct Point{
     int x,y;
} point;
point.x = 10;
point.y = 20;

file.write((char *)&point, sizeof(Point));
file.flush();

file.close();
```

Для ускорения работы производится буферизация записываемых данных. Информация из буфера записывается в файл полностью только в момент закрытия файла. Сбросить буфер в файл явным образом можно с помощью метода `flush()`.

Методы, предназначенные для **чтения**:

**>>**

**get()**

Первый прототип:

```
int get();
```

```
char ch=file.get();
```

Второй прототип:

```
istream &get(char &ch);
```

Посимвольное считывание из файла.

```
char ch;
while (!file.get(ch).eof())
{
	cout << ch;
}
```

```
char ch;
while (file.get(ch))
{
	cout << ch;
}
```

Третий прототип:

```
istream &get(char *Str, streamsize Count);
```

```
char str[100] = {0};
file.get(str,50);
```

Четверый прототип:

```
istream &get(char *Str, streamsize Count, char Delim);
```

```
char str[100] = {0};
file.get(str,50,' ');
```

### peek()

Прототип метода:

```
int_type peek();
```

### getline

Прототипы метода:

```
istream &getline(char *Str, streamsize Count);
istream &getline(char *Str, streamsize Count,
                 char Delim);
```

### ignore()

Прототип метода:

```
istream &ignore(streamsize Count=1,
                                int_type Metadelim=EOF);
```

```
ignore(91, '\n');
```

**read()**

Прототип метода:

```
istream &read(char *Str, streamsize Count);


streamsize gcount() const;
```

Чтение структуры из файла, открытого на чтение в бинарном режиме.

```
std::ifstream file;
file.open("data_binary.txt", std::ios::in | std::ios::binary);

struct Point{
     int x,y;
} point;

file.read((char *)&point, sizeof(Point));
cout << point.x << ' ' << point.y << endl;

file.close();
```

## Файлы произвольного доступа

**tellg()** и **tellp()** – возвращают позицию соответствующего курсора относительно начала файла. При ошибке возвращается значение –1. Метод tellg() возвращает позицию курсора чтения, а tellp() – позицию курсора записи.

Прототипы методов:

```
pos_type tellg();
pos_type tellp();
```

*Пример*:

```
std::ifstream file("data.txt", std::ios::in | std::ios::binary);

if (!file) return 1;

cout << file.tellg() << endl; // 0
char ch = file.get();
cout << file.tellg() << endl; // 1
```

**seekg()** и **seekp()** – устанавливают соответствующие курсоры в позицию, имеющую смещение Offset относительно позиции Origin, или в позицию Pos относительно начала файла. Метод seekg() возвращает позицию курсора чтения, а seekp() – позицию курсора записи.

Прототипы методов:

```
istream &seekg(pos_type Pos);
istream &seekg(off_type Offset, ios::seekdir Origin);
istream &seekp(pos_type Pos);
istream &seekp(off_type Offset, ios::seekdir Origin);
```

В параметре Origin могут быть указаны следующие значения:

- **ios::beg** – начало файла;
- **ios::cur** – текущая позиция курсора;
- **ios::end** – конец файла.

*Пример*:

```
std::ifstream file("data.txt", std::ios::in | std::ios::binary);
// в файле находится слово "Строка"

char ch;
file.seekg(4, std::ios::cur); // Перемещаем курсор
                              // относительно текущей позиции
ch = file.get();
cout << ch << endl; // к

file.seekg(0, std::ios::beg); // Перемещаем курсор в начало
ch = file.get();
cout << ch << endl; // С

file.seekg(-1, std::ios::end); // Перемещаем курсор в конец
ch = file.get();
cout << ch << endl; // а
```

## Проверка состояния потока

**ios::goodbit**
**ios::eofbit**
**ios::failbit**
**ios::badbit**

**rdstate** (good(), eof(), fail() и bad())

Прототип метода:

```
iostate rdstate() const;
```

```
if (file.rdstate() == std::ios::goodbit)
{
     std::cout << "goodbit" << endl;
}
```

```
if ((int)file.rdstate() & (int)std::ios::eofbit)
{
     std::cout << "eofbit" << endl;
}
```

**good()**
**eof()**
**fail()**

```
if (!file)
{
	std::cout << "failbit" << endl;
}
```

**bad()**

**clear()**

Прототип метода:

```
void clear(iostate State = goobit, bool Reraise = false);
```

*Пример:*

```
file.clear(); // Сбрасываем флаг
file.clear(std::ios::failbit | std::ios::eofbit);
// Устанавливаем флаги failbit и eofbit
cout << file.good() << endl; // 0
cout << file.eof() << endl;  // 1
cout << file.fail() << endl; // 1
cout << file.bad() << endl;  // 0
```