Потоки и файлы

Часть 2

Работа с файлами

Классы для работы с дисковыми файлами

- ifstream
- ofstream
- fstream

Форматированный файловый вывод

```
#include<fstream>
int main ()
{
int i=99;
float f=5.375;
char c='*';
double d=3.2;
string s="something";
ofstream tofile("myfile.txt");
tofile<<i<<' '<<f<<c<<d<<s;
return 0;
}
```

Форматированный файловый ввод

```
#include<fstream>
int main ()
{
char c; double d;
int i; float f;
string s;
ifstream fromfile("myfile.txt");
fromfile>>i>>f>>c>>d>>s;
cout<<i<<endl<<f<<endl<<c<<endl<<d<<endl<<s;
return 0;
}
```

Файловый ввод\вывод строк

```
#include<fstream>
int main ()
{
ofstream
   tofile("myfile.txt");
tofile<<"First string\n";
tofile<<"Second string\n";
tofile<<"Third string\n";
return 0;
}
```

```
#include<fstream>
int main ()
{
char buff[100];
ifstream
   fromfile("myfile.txt");
while (!fromfile.eof()){
fromfile.getline(buff,100);
cout<<buff<<endl;
}
return 0;
}
```

```
if (fromfile.good())…
if (fromfile)…
```

Посимвольный файловый ввод\вывод

```
#include<fstream>
int main ()
{
string s = "Some string";
ofstream tofile("myfile.txt");
for (int i=0; i<s.size(); i++)
   tofile.put(s[i]);
return 0;

}
```

```
#include<fstream>
int main ()
{
char c;
ifstream
   fromfile("myfile.txt");
while (fromfile){
   fromfile.get(c);
   cout<<c;
   }
return 0;
}
```

```
ifstream fromfile("myfile.txt");
cout << fromfile.rdbuf();
```

static_cast

```
int i = 123;
float f = static_cast<float>(i);
cout << f/10;
```

reinterpret_cast

```
int i = 68;
//char *pi =(char*)&i;
char
  *pi=reinterpret_cast<char*>(&i);
cout<<*pi<<endl;// D
```

Двоичный файловый ввод\вывод

```
#include<fstream.h>
int main ()
{
int imas[100];
//заполнение imas
ofstream tofile("myfile.data", ios::binary);
tofile.write(reinterpret_cast<char*>(imas),
  100*sizeof(int));
tofile.close();
//очистить imas
ifstream fromfile("myfile.data",
  ios::binary);
fromfile.read(reinterpret_cast<char*>(imas),
  100*sizeof(int));
return 0;
}
```

Использование OpenDialog, SaveDialog

```
int imas[100];
for (int i=0; i<100; i++)imas[i]=i;
if (SaveDialog1->Execute()){
  ofstream tofile (SaveDialog1
   ->FileName.c_str(), ios::binary);
tofile.write(reinterpret_cast<char*>(imas),
  100*sizeof(int));
tofile.close();
}
for (int i=0; i<100; i++)imas[i]=0;
if (OpenDialog1->Execute()){
ifstream fromfile(OpenDialog1
->FileName.c_str(), ios::binary);
fromfile.read(reinterpret_cast<char*>(imas),
  100*sizeof(int));
}
```

Файловый ввод\вывод структурированных данных

```
struct Student{
   unsigned short Course;
   char FirstName[15];
   char LastName[15];
   char Speciality[10];
   bool PaymentForm;
};
int main ()
{
Student Petrov={1,"Petrov","Petr","MOAIS",false};
ofstream tofile("myfile.stud", ios::binary);
tofile.write(reinterpret_cast<char*>(&Petrov),
   sizeof(Student));
tofile.close();
Student PetrovsCopy;
ifstream fromfile("myfile.stud", ios::binary);
fromfile.read(reinterpret_cast<char*>(&PetrovsCopy),
   sizeof(Student));
return 0;
}
```

Файловый ввод\вывод массива структурированных данных

```
struct Student;
int main (){
const int n=24;
//ввод n
Student group113[n]; //статический
Student *Group113=new Student[n];//динамический
//заполнение массива group113
ofstream tofile("myfile.stud", ios::binary);
tofile.write(reinterpret_cast<char*>(group113),
  n*sizeof(Student));
tofile.close();
ifstream fromfile("myfile.stud", ios::binary);
fromfile.read(reinterpret_cast<char*>
  (Group113), n*sizeof(Student));
return 0;
}
```

Биты режимов открытия файлов

- in
- out
- ate
- app
- trunc
- nocreate
- noreplace
- binary

Комбинации битов режима:

- **ios::out | ios::trunc** удаляется существующий файл и (или) создается для записи;
- **ios::in | ios::out | ios::trunc** существующий файл удаляется и (или) создается для для чтения и записи;
- **ios::in | ios::out** открывается существующий файл для чтения и записи:;
- **ios::out | ios::app** открывается существующий файл для дозаписи в конец файла.;
- **ios::in | ios::out | ios::app** открывается существующий файл для чтения и дозаписи в конец файла.

Метод open

Имя_файловой переменной.open(имя_файла, режим_открытия)

```
ifstream file;
file.open("C:\\temp\\1.data",ios::bin
  ary);
```

Указатель файла

seekg(int);

seekg(int, pos);

tellg();

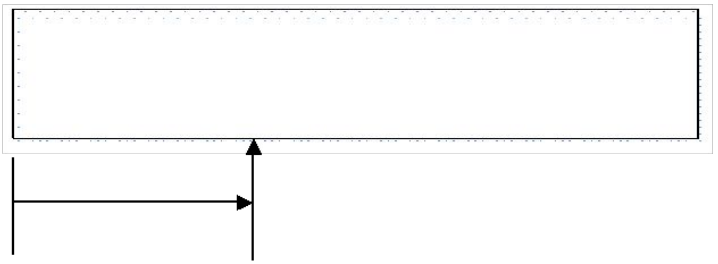
seekp(int);

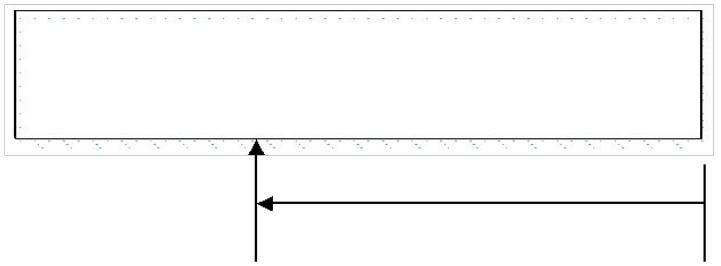
seekp(int, pos);

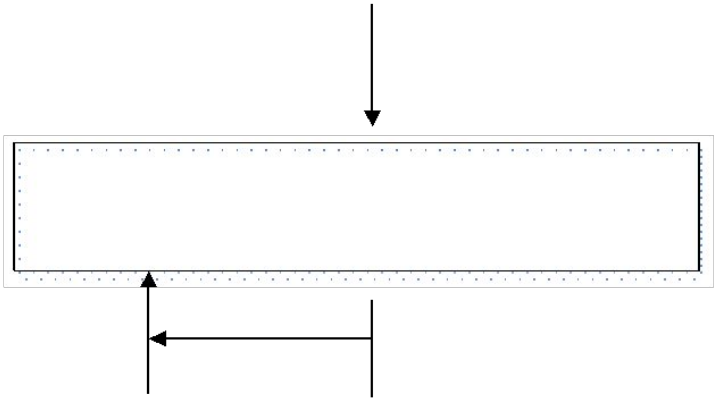
tellp();

pos: ios::beg,

ios::cur, ios::end







Пример записи в конец файла

```
struct Student;
int main (){
Student student;
//ввод данных о студенте
ofstream ofile("myfile.stud", ios::binary |
   ios::app);
ofile.write(reinterpret_cast<char*>(&student),
   sizeof(Student));
ofile.close();
ifstream ifile;
ifile.open("myfile.stud", ios::binary);
ifile.read(reinterpret_cast<char*>(&student),
   sizeof(Student));
while(!file.eof()){
   //вывод информации о студенте student
   file.read(reinterpret_cast<char*>(&student),
   sizeof(Student));
   }
return 0;
}
```

Пример чтения из произвольной позиции в файле

```
struct Student;
int main (){
Student student;
ifstream fromfile;
fromfile.open("myfile.stud", ios::binary | ios::in);
fromfile.seekg(0, ios::end);
int endpos = fromfile.tellg();
int count = endpos/sizeof(Student);
int number;
cin>>number; //ввести номер студента
fromfile.seekg((number-1)*sizeof(Student));
file.read(reinterpret_cast<char*>(&student),
   sizeof(Student));
//вывести информацию о студенте student
return 0;
}
```

Проверка ошибок при файловом вводе\выводе

```
int main (){
ifstream fromfile;
ofstream tofile;
fromfile.open("myfile1.data",
  ios::binary);
if (!fromfile)…//обработка ошибки
tofile.open("myfile2.data",
  ios::binary);
if (!tofile)…//обработка ошибки
```

Аргументы командной строки

```
#include <iostream>
using namespace std;
int main(int argc, char* argv[])
{
cout<<"args count = "<<argc<<endl;
for (int j=0; j<argc; j++) //output all args
cout<<"arg "<<j<<" = "<<argv[j]<<endl;
return 0;
}

/*Например, следующая командная строка:
D:\temp>Project1.exe arg1 12.6 1234567 lastarg
Выведет на экран:
args count = 4
arg 0 = arg1
arg 1 = 12.6
arg 2 = 1234567
arg 3 = lastarg
*/
```

Функции языка С для работы с файлами

stdio.h

- **FILE** *указатель на файл;
- FILE ***fopen**(const char **имя_файла*, const char **режим_открытия*); //(a, r, w, a+, r+…)
- int **fclose**(FILE **указатель_на _файл*);
- int **putc**(int символ, FILE * указатель_на _файл);
- int **getc**(FILE * указатель_на _файл);
- int **feof**(FILE * указатель_на _файл);
- int **fputs**(const char * *строка*, FILE * *указатель_на _файл*);
- char ***fgets**(char **строка*, int *длина*, FILE * *указатель_на _файл*);
- size_t **fwrite**(const void * *записываемое_данное*, size_t *размер_элемента*, size_t *число_элементов*, FILE *указатель_на _файл);
- size_t **fread**(void * *считываемое_данное*, size_t *размер_элемента*, size_t *число_элементов*, FILE *указатель_на _файл);
- long **ftell**(FILE * *указатель_на _файл*);
- int **fseek**(FILE * *указатель_на _файл*, long int *число_байт*, int *точка_отсчета*); //(SEEK_SET, SEEK_CUR, SEEK_END)
- …

Системный ввод\вывод

io.h

- *chsize* - изменяет размер файла
  filelength - контролирует длину файла
  locking - защищает области файла от несанкционированного доступа.
- *mktemp* - создает уникальное имя файла
- *remove* - удаляет файл
- *rename* - переименовывает файл
- *setmode* - устанавливает режим обработки файла
- *stat* - получает по имени файла информацию о статусе файла
- …