***Открытие файла*****FILE\* fopen (const char\* fname, const char\* mode);  
fname** – имя файла:  
 **mode –** режим открытия.   
В случае удачи возвращает указатель на поток, иначе – 0.   
**mode** может принимать значения:  
**r** существующий файл открывается для чтения;  
**w** создается новый файл, который открывается для записи; если такой файл уже существует,

то предыдущее содержимое стирается;   
**a** существующий файл открывается для добавления информации в его конец;

**r+** существующий файл открывается для чтения и записи; но не разрешается запись в конец

файла, т.е. увеличение файла;   
**w+** создается новый файл, который открывается для чтения и записи;   
**a+** существующий файл открывается для чтения и добавления информации в его конец или

создается при его отсутствии.

***Закрытие файла*  
int fclose(FILE\* stream);**

***Таким образом, канва программы, работающей с файлами, имеет следующий вид:***

FILE \*f;

if ((f=fopen(”myfile.txt”,”rt”)) != NULL)

{

// работа с файлом

fclose(f);

}

***Построчное чтение из текстового файла***void ReadStringFile( FILE \*in,char\* filename )  
{char \*str=(char\*)malloc (81);  
 if ((in = fopen(filename,"r"))!= NULL)   
 {   
 fgets(str,80,in); // gets(str);   
 if (feof(in)) printf("Файл пустой\n");   
 while (!feof(in))  
 {   
 fputs(str,stdout);   
 fgets(str,80,in); // gets(str);  
 }  
 fclose(in);   
 }  
 else   
 printf("Файл не открывается\n");  
}

***Построчное создание текстового файла***void CreateStringFile(FILE \*out,  
 char\* filename)  
{  
 char str[81];  
 out = fopen(filename,"wt");  
 fgets(str,80,stdin); // gets(str);   
 while(!feof(stdin))  
 { fputs(str,out);  
 fgets(str,80,stdin); // gets(str);   
 }   
 fclose(out);  
}

***Текстовый файл input.txt содержит строки длиной не более 100 символов.***

***Найти число строк, начинающихся с пробела.***

int main()

{

int N = 0;

FILE\* f;

char \*S = new char[101];

if ((f = fopen("input.txt", "rt"))!= NULL)

{

while (fgets(S, 101, f)!= NULL)

if (S[0] == ' ') N++;

fclose(f);

cout << N;

}

else

printf("Файл не открывается\n");

return 0;

}

***Посимвольный ввод-вывод***    
**int fputc(int c, FILE\* stream);  
int putc(int c, FILE\* stream);**Записывает символ **c** в поток **stream** и продвигает индикатор позиции на следующий символ.

Перед записью в файл параметр **c** типа **int** преобразуется в тип **unsigned char**. При успехе возвращает **c**,

в случае неудачи **EOF** (-1).  
 Функции идентичны, но **putc** определена обычно как макрос.

**int fgetc(FILE\* stream);  
int getc(FILE\* stream);**Работают аналогично **fputc**, но читают символ. В случае достижения конца файла возвращают **EOF**

и устанавливают индикатор конца файла.   
В консольном режиме значению **EOF** соответствует нажатие клавиш **Ctrl+Z**.

***Пример 18.4.***void ReadChFile(FILE \*in, char\* filename )  
// Чтение текстового файла посимвольно  
{int ch;  
 if ((in = fopen(filename, "r")) != NULL)   
 {   
 ch = getc(in);  
 if(ch == EOF) printf("Файл пустой\n");  
 while (ch != EOF)   
 {   
 putc(ch,stdout);   
 ch = getc(in);  
 }  
 fclose(in); // закрываем файл   
 }  
 else   
 printf (" Файл не открывается\n");  
}

***Пример 18.5.***

void CreateChFile( FILE \*out, char\* filename )

//Создание текстового файла по одному символу

{

int ch;

out = fopen(filename, "w");

ch=getchar();

while (ch!=EOF)

{

putc(ch,out);

ch=getchar();

}

fclose(out);

}

***Форматированный ввод-вывод*  
  
int fprintf(FILE\* stream,   
 const char\* format,...);**   
Форматированный вывод в файл **stream**. Работает аналогично **sprintf**, в случае удачи

возвращает число выводимых символов, иначе возвращает -1.  
**int printf(const char \* формат, …)**

**int fscanf(FILE\* stream, const char\* format   
[ адреса\_переменных ],**...**);**   
 Форматированный ввод из файла **stream**, в случае успеха возвращает количество единиц

прочитанных данных, иначе возвращает **EOF**.  
  
**int scanf(const char \* формат,   
[ адреса\_переменных ],…)**

***Пример 18.6. Создание текст. файла, используя форматированный ввод строк***

void CreateFStringFile( FILE \*out, char\* filename )

{ int k;

char str[80];

out = fopen(filename, "w") ;

scanf("%s",str);

//считывает до первого пробела либо конца строки

while(!feof(stdin))

{

fprintf(out,"%s\n",str);

scanf("%s",str);

}

fclose(out);

}

***Пример 18.7.***

void ReadFStringFile( FILE \*in, char\* filename )

{ char \*str=(char\*)malloc (80);

if ((in = fopen(filename, "r") ) != NULL)

{ fscanf(in,"%s",str);

if (feof(in)) printf("Файл пустой\n");

else

{ do

{ printf("%s\n",str);

fscanf(in,"%s",str);

}

while (!feof(in));

}

fclose(in);

}

else printf (" Файл не открывается\n");

}

***Пример 18.6. Слияние двух упорядоченных по возрастанию текстовых файлов целых чисел***

#include <stdio.h>

#include <iostream>

#include <string.h>

#include <limits.h> //константа INT\_MAX

using namespace std;

int main()

{

FILE\* f1; FILE\* f2; FILE\* fout;

int n1,n2,nnnn;

if(!(f1 = fopen("f1.txt", "r")))

{ printf("Open f1 file failed.\n");

return 1;

}

if(!(f2 = fopen("f2.txt", "r")))

{ printf("Open f2 file failed.\n");

return 1;

}

if(!(fout = fopen("fout.txt", "w")))

{ printf("Open fout file failed.\n");

return 1;

}

fscanf(f1,"%d",&n1);

if (feof(f1)) n1=INT\_MAX;

fscanf(f2,"%d",&n2);

if (feof(f2)) n2=INT\_MAX;

while((n1!=INT\_MAX) || (n2!=INT\_MAX))

{ if (n1<n2)

{ fprintf(fout,"%d ",n1);

fscanf(f1,"%d",&n1);

if (feof(f1)) n1=INT\_MAX;

}

else

{ fprintf(fout,"%d ",n2);

fscanf(f2,"%d",&n2);

if (feof(f2)) n2=INT\_MAX;

}

}

fclose(f1);

fclose(f2);

fclose(fout);

return 0;

**}**

***Пример 18.7.***

*В первой строке текстового файла записаны кол-во*

*строк и столбцов матрицы. В каждой следующей – по одной строке матрицы. В выходной файл*

*записать матрицу и суммы элементов каждой строки в виде a11+a12+...+a1n=сумма1*

int main()

{ FILE\* in=NULL;

// Для избежания проверок в Debug

// В режиме Release NULL не нужен

FILE\* out=NULL;

int m, n, i, j, max;

if(!(in = fopen("in1.txt", "r")))

{ printf("Open in file failed.\n");

return 1;

}

fscanf(in, "%d%d%", &m, &n);

printf("m = %d n = %d \n", m,n);

int \*\*a; // матрица динамическая

a=new int \*[m]; //массив указателей

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

a[i]= new int [n];

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

for (j=0;j<n;j++)

if (!feof(in))

fscanf(in, "%d",&a[i][j]);

else

{

printf("error\n");

return 1;

}

fclose(in);

if(!(out = fopen("out2.txt", "w")))

{ printf("Open out file failed.\n");

return 1;

}

int sum;

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

{ sum=0;

for (j=0;j<n;j++)

{ sum+=a[i][j];

fprintf(out,"%d %s",a[i][j],

(j==n-1?” = ”:” + ”));

}

fprintf(out, "%d \n", sum);

}

fclose(out);

return 0;

}

**ПОТОКИ**

***Методы класса istream***

get() - извлекает из потока символ и возвращает его код или EOF.

getc(ch) - извлекает из потока символ и помещает его в ch.

***Особенности работы с файловыми потоками*  
*Создание файлового потока* :**  
fstream(const char \*имяфайла, *[* int *режим* =ios::in *]*)  
ofstream(const char \* имяфайла, *[* int *режим* =ios::out | ios::trunc*]*)  
fstream(const char \* имяфайла, *[* int *режим* =ios::out | ios::in *]*)

***Пример 19.1****. Файл in1.txt содержит строки длиной не более 100 символов.   
Требуется посчитать в этом файле число строк, начинающихся с пробела.* **#include <fstream>  
#include <iostream>   
using namespace std;  
  
int main()   
{int N = 0;  
 char \*S = new char[101];  
 ifstream f("in1.txt");  
 while (f.getline(S,101)!= NULL)  
 if (S[0] == ’ ’) N++;  
 cout << N << endl;  
 delete [] S;   
 return 0;}**

***Пример 19.2****.* **//***Переписать все непустые строки из файла* **in1.txt,** *описанного в предыдущем*

*примере, в выходной текстовый файл* **out1.txt   
int main() {  
 char \*S = new char[101];  
 ifstream f2("in1.txt");  
 ofstream f1("out1.txt");  
 while (f2.getline(S,101)!= NULL)   
 if (S[0] != '\0')  
 {  
 f1.write(S, strlen(S));  
 f1.put('\n');  
 //или иначе f1<<S<<”\n”;   
 }   
 delete [] S; return 0;}**