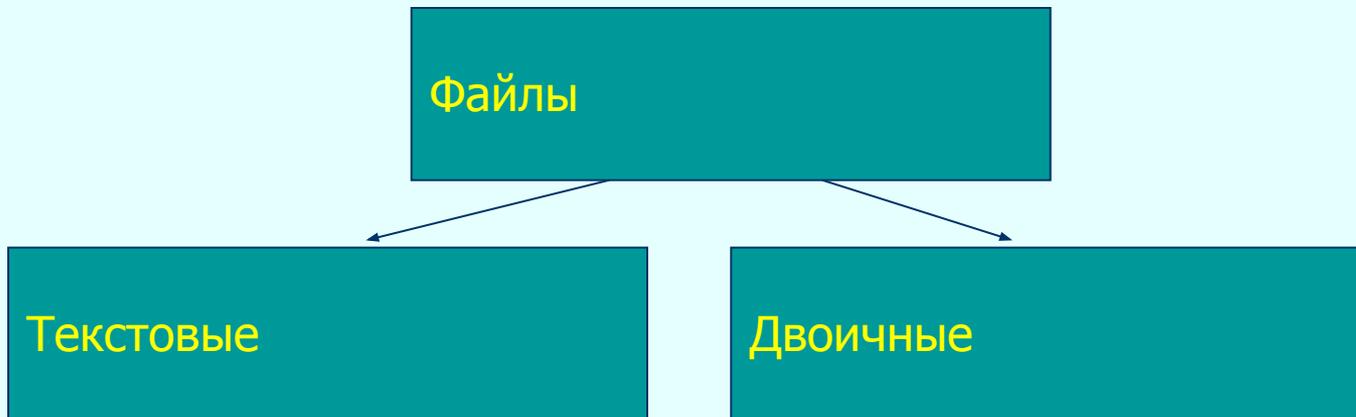


# Файлы в Си

## Тип доступа к файлам



```
редактирование SL9_1.CPP - Far
H:\BORLANDC\BIN\SL9_1.CPP      DOS   Строка      1/47   Кол 1      35
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>

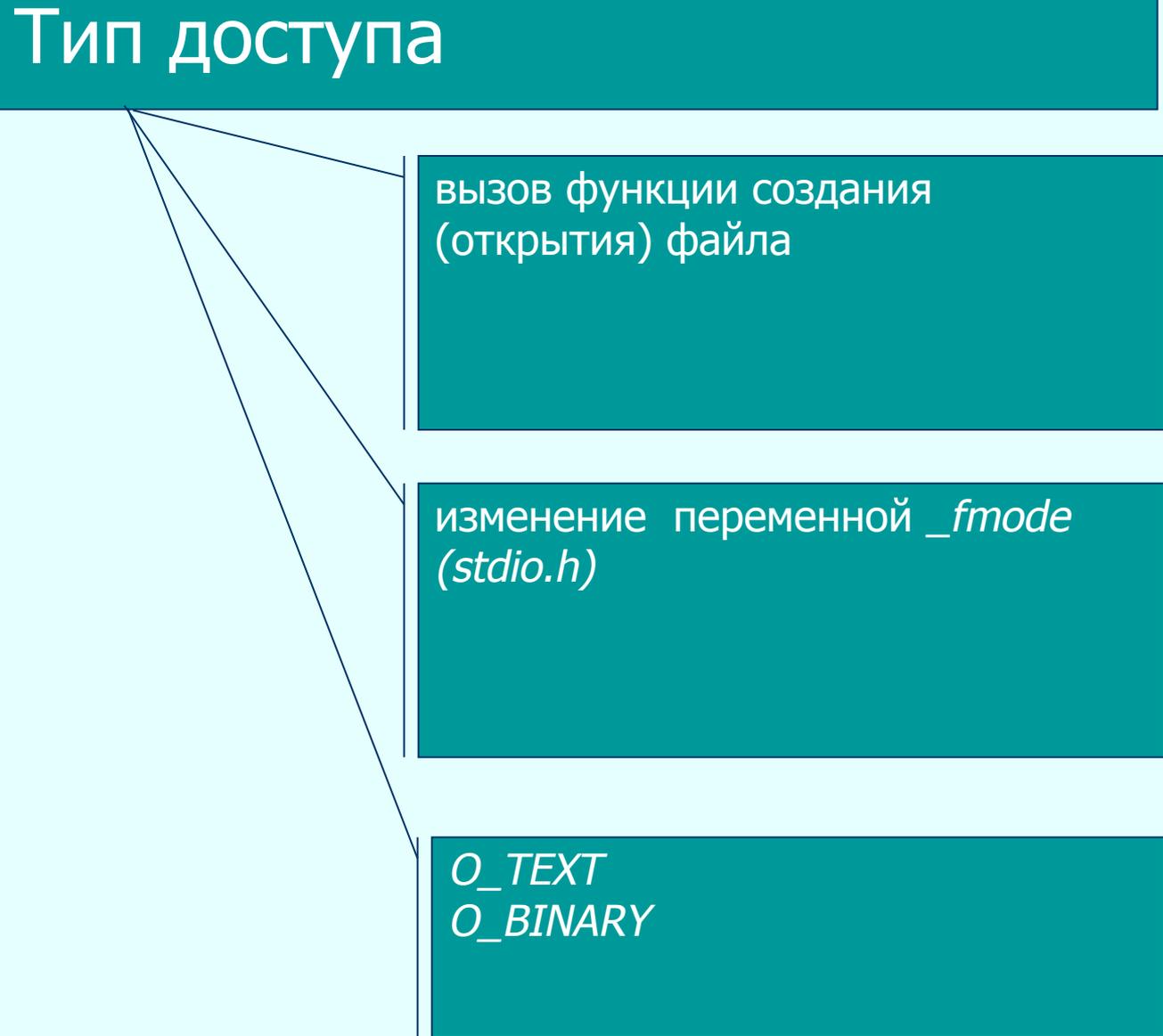
void Show(int *x, int n, char* t){
    printf("\n%s\n",t);
    for(int i=0;i<n;i++)
        printf("%4d",x[i]);
}

int *Create(int *x, int n){
    x = new int[n];
    for(int i=0;i<n;i++)
        x[i] = random(20);
    return x;
}

int Zero (int *x, int n)
{
    int z = 0;
    for(int i=0;i<n;i++)
        if(x[i]==0) z++;
    return z;
}
1 2 3 4 5 Печать 6 7 Назад 8 Строка 9 Видео 10
```

	DOS	903	Кол	0	0%
00000000000:	23 69 6E 63 6C 75 64 65	20 3C 63 6F 6E 69 6F 2E	#include <conio.		
00000000010:	68 3E 0D 0A 23 69 6E 63	6C 75 64 65 20 3C 73 74	h>#include <st		
00000000020:	64 69 6F 2E 68 3E 0D 0A	23 69 6E 63 6C 75 64 65	dio.h>#include		
00000000030:	20 3C 73 74 64 6C 69 62	2E 68 3E 0D 0A 0D 0A 76	<stdlib.h>#v		
00000000040:	6F 69 64 20 53 68 6F 77	28 69 6E 74 20 2A 78 2C	oid Show(int *x,		
00000000050:	20 69 6E 74 20 6E 2C 20	63 68 61 72 2A 20 74 29	int n, char* t)		
00000000060:	7B 0D 0A 20 20 70 72 69	6E 74 66 28 22 5C 6E 25	< printf("\n%z		
00000000070:	73 5C 6E 22 2C 74 29 3B	0D 0A 20 20 66 6F 72 28	s\n",t); for<		
00000000080:	69 6E 74 20 69 3D 30 3B	69 3C 6E 3B 69 2B 2B 29	int i=0;i<n;i++>		
00000000090:	0D 0A 20 20 70 72 69 6E	74 66 28 22 25 34 64 22	printf("%4d"		
000000000A0:	2C 78 5B 69 5D 29 3B 0D	0A 7D 0D 0A 0D 0A 69 6E	,x[i]); int		
000000000B0:	74 20 2A 43 72 65 61 74	65 28 69 6E 74 20 2A 78	*Create(int *x		
000000000C0:	2C 20 69 6E 74 20 6E 29	7B 0D 0A 20 78 20 3D 20	, int n)< x =		
000000000D0:	6E 65 77 20 69 6E 74 5B	6E 5D 3B 0D 0A 20 66 6F	new int[n]; fo		
000000000E0:	72 28 69 6E 74 20 69 3D	30 3B 69 3C 6E 3B 69 2B	r<int i=0;i<n;i+		
000000000F0:	2B 29 0D 0A 20 78 5B 69	5D 20 3D 20 72 61 6E 64	> x[i] = rand		
00000000100:	6F 6D 28 32 30 29 3B 0D	0A 20 72 65 74 75 72 6E	om(20); return		
00000000110:	20 78 3B 0D 0A 7D 0D 0A	0D 0A 69 6E 74 20 5A 65	x; int Ze		
00000000120:	72 6F 20 28 69 6E 74 20	2A 78 2C 20 69 6E 74 20	ro (int *x, int		
00000000130:	6E 29 0D 0A 7B 0D 0A 20	69 6E 74 20 7A 20 3D 20	n)< int z =		
00000000140:	30 3B 0D 0A 20 66 6F 72	28 69 6E 74 20 69 3D 30	0; for<int i=0		
00000000150:	3B 69 3C 6E 3B 69 2B 2B	29 0D 0A 20 20 69 66 28	;i<n;i++> if<		
00000000160:	78 5B 69 5D 3D 3D 30 29	20 7A 2B 2B 3B 0D 0A 20	x[i]==0) z++;		

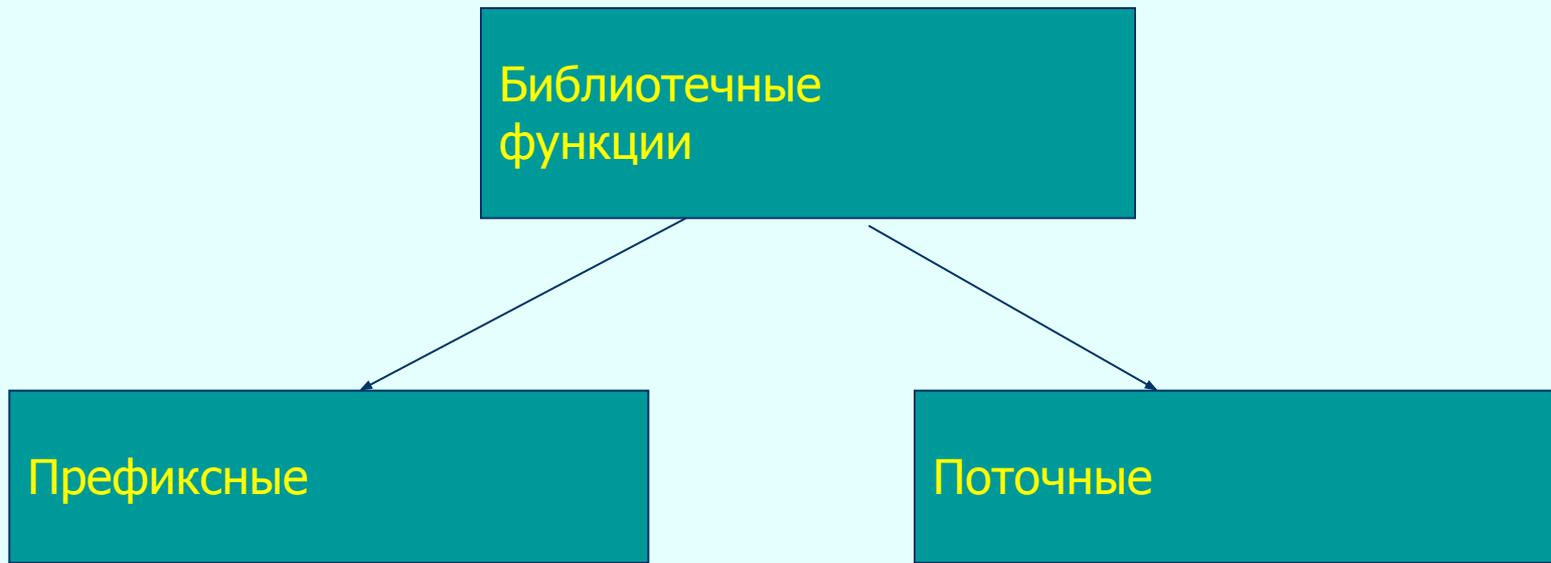
# Тип доступа



вызов функции создания  
(открытия) файла

изменение переменной *\_fmode*  
(*stdio.h*)

*O\_TEXT*  
*O\_BINARY*



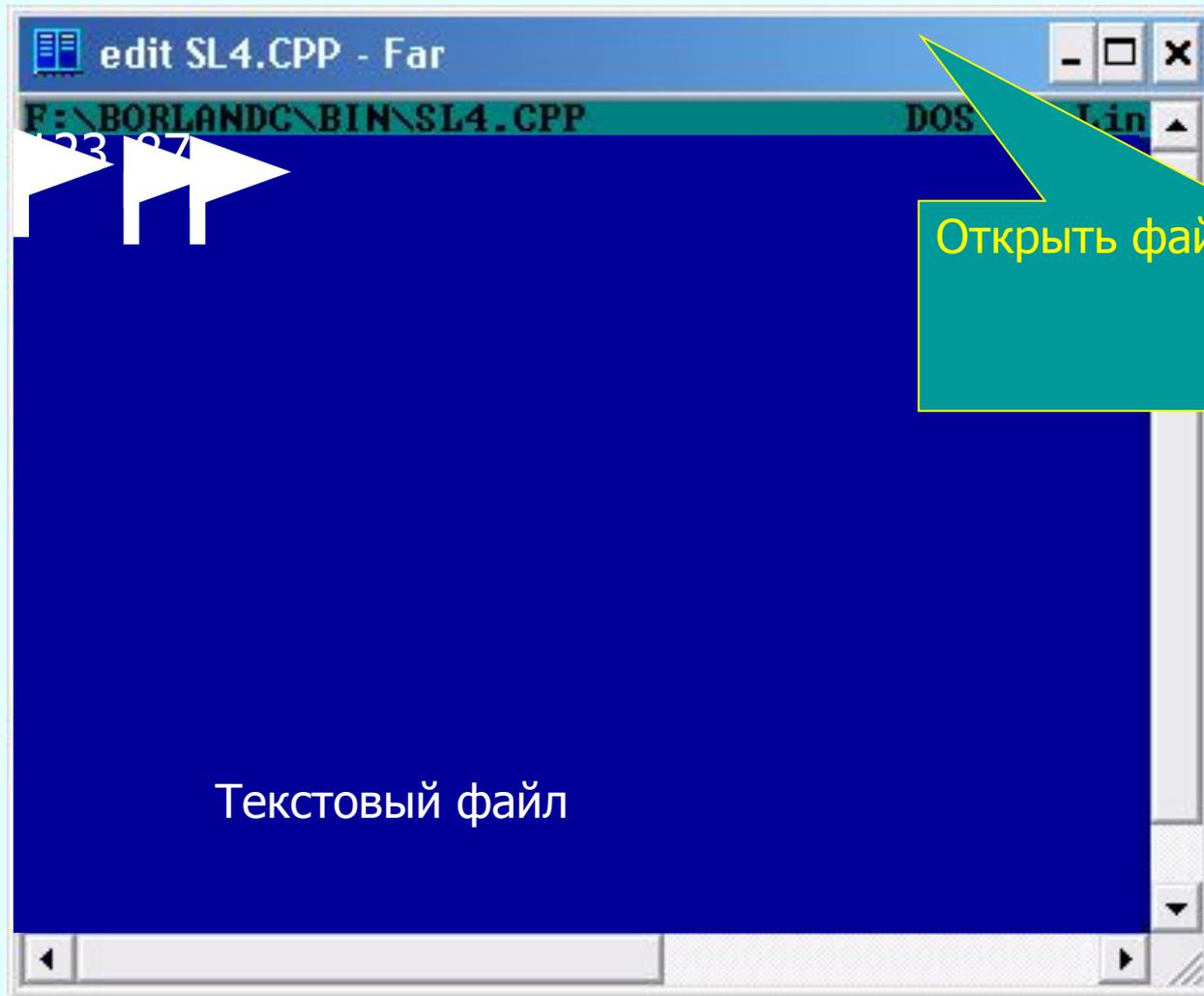
## Открытые файлы операционной системы

Префикс	Файл
0	<i>stdin</i>
1	<i>stdout</i>
2	<i>stderr</i>
3	<i>stdaux</i>
4	<i>stdprn</i>

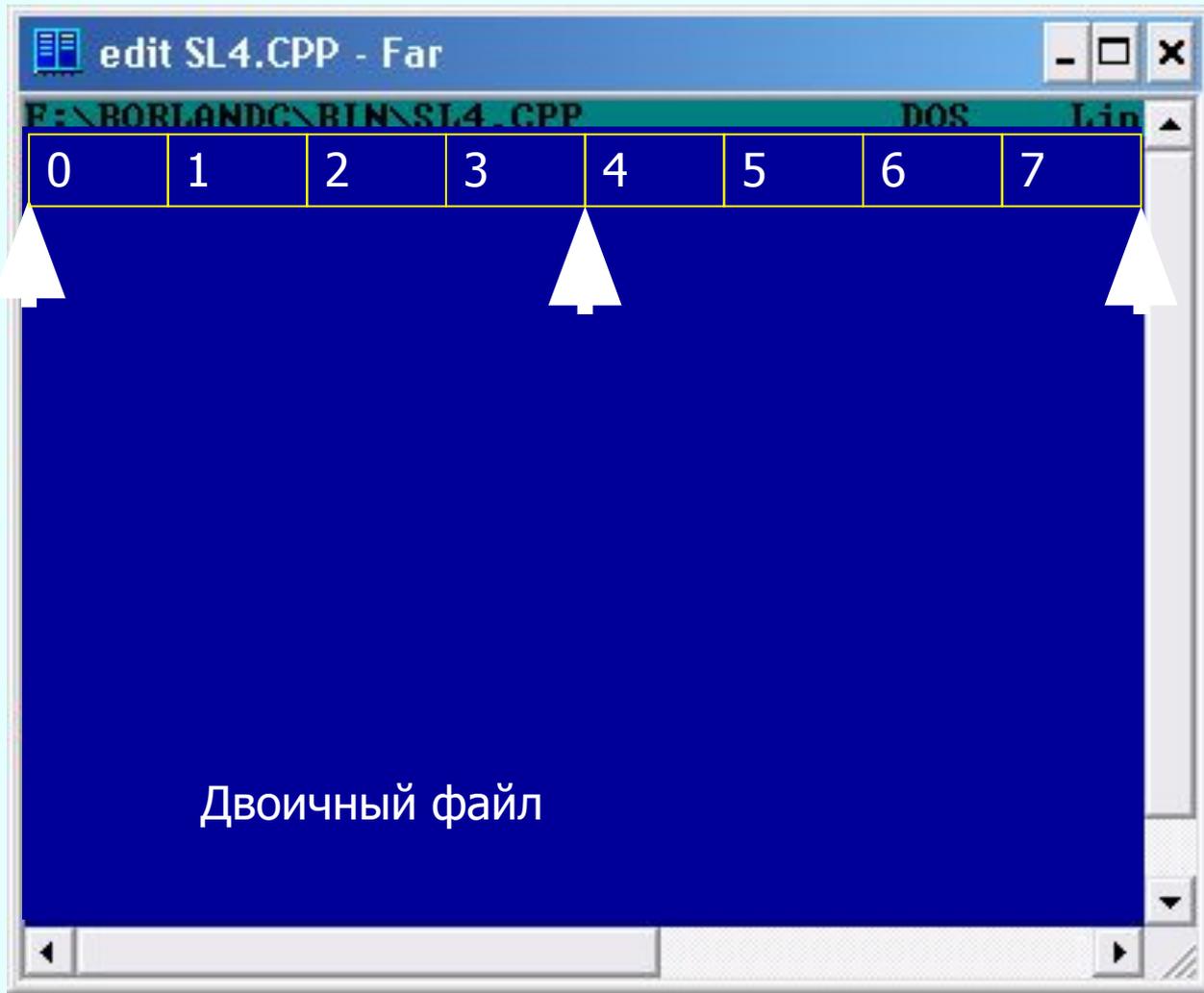
# Механизм чтения из файла

```
edit SL4.CPP - Far
F:\BORLANDC\BIN\SL4.CPP  DOS  Lin
#include <stdio.h>
#include <conio.h>
void main() {
  clrscr();
  int kol=0;
  int data;
  int sum=0;
  printf("Enter data: ");
  while (scanf("%d",&data))
  {
    sum+=data;
    kol++;
  }
  printf(" Summa %d, kol-vo %d ",sum, kol);
  getch();
}
```

# Механизм записи в файл



Открыть файл для записи



# Функции для поточного доступа к файлам

- Функции поточного ввода-вывода называют стандартными функциями ввода-вывода.
- Си создает внутреннюю структурную переменную по шаблону *FILE (stdio.h)*.

```
FILE* fopen (const char *filename, const char * mode)
```

константная строка - имя файла

В случае неуспеха функция возвращает значение **NULL**

режим открытия файла

## Режимы открытия файла

- *r* - открыть для чтения.
- *w* - создать для записи.

- *a* – открыть файл для обновления, открывает файл для записи в конец файла или создает файл для записи, если файла не существует.
- *r+* - открыть существующий файл для корректировки (чтения и записи).
- *w+* - создать новый файл для корректировки (чтения и записи).
- *a+* - открыть для обновления; открывает для корректировки (чтения и записи) в конец файла, или создает, если файла не существует.
- *t* – *открыть файл в текстовом режиме.*

- *b* – открыть файл в двоичном режиме.
- По умолчанию установлен **ТЕКСТОВЫЙ** режим доступа к файлу.
- Режимы *t* и *b* указываются вторыми символами в строковой переменной *mode*, например "*r+b*".

## Проверка корректности открытия файла:

```
int main(int argc, char *argv[])  
{ system("chcp 1251");  
  FILE *f;  
  char name [] = "prim.txt";  
  if ((f = fopen(name,"rb"))==NULL) {  
    printf( "File not found ");  
    system("PAUSE"); }  
  else printf( "Ok... ");  
  system("PAUSE");  
  return EXIT_SUCCESS;}
```

- *int fclose(FILE \*fp)* - закрывает файл *fp*, при успешной работе возвращает 0, при неуспешной *EOF*.
- *int closeall(void)* – закрывает все файлы, открытые в программе, при успешной работе возвращает число закрытых потоков, при неуспешной - *EOF*.
- *FILE \*freopen(const char \*filename, const char\* mode, FILE \*stream)* – закрывает поток *stream*, открывает поток *filename* с новыми правами доступа установленными в *mode*. Если потоки разные, то происходит переадресация потока *stream* в поток *filename*.

```
...char name[] = "prim.txt";  
int n;  
FILE *f;  
printf( "Введите целое число n: ");  
scanf("%d", &n);  
freopen(name,"wt",stdout);  
for (int i=0;i<n;i++)  
    printf("%d\n",i);  
fclose(f);  
system("PAUSE");  
return EXIT_SUCCESS; ...
```

- *ch = fgetc( <указатель на файл> )* – возвращает символ *ch* из файла, с которым связан указатель.
- *ch = getc( <указатель на файл> )* – возвращает символ *ch* из файла, с которым связан указатель.
- *fputc(ch, <указатель на файл> )* – записать символ *ch* в указанный файл.
- *putc(ch, <указатель на файл> )* – записать символ *ch* в файл.
- *fgets(str, n, <указатель на файл> )* – прочитать строку *str*, длиной *n* символов, или до первого встреченного `\n` из указанного файла.

- *fputs(str, <указатель на файл>)* – записать строку *str*, в файл. Символ перевода на другую строку в файл не записывается.
- *fscanf( <указатель на файл>, управляющая строка, ссылка)* – универсальная функция считывания из текстового файла.

*fscanf(f, "%d", &n)*

- *fread(ptr, size, n, <указатель на файл>)* – считывает *n* элементов размером *size* в область памяти, начиная с *ptr*. В случае успеха возвращает количество считанных элементов, в случае неуспеха – *EOF*.

- *fwrite(ptr, size, n, <указатель на файл>)* – записывает *n* элементов размером *size* из памяти, начиная с *ptr* в файл– в случае успеха возвращает количество записанных элементов, в случае неуспеха – *EOF*.
- *fprintf( <указатель на файл>, управляющая строка, [список аргументов] )* – форматированный вывод в файл.

## функции прямого доступа к файлу

- *rewind( <указатель файла>)* – установить указатель файла на начало файла.

- *int fseek( <указатель файла>, offset, fromwhere)* - установить указатель чтения-записи файла на позицию *offset*, относительно позиции *fromwhere*.

значения *fromwhere*

SEEK\_END – от конца файла,

SEEK\_SET – от начала файла,

SEEK\_CUR – от текущей позиции.

- *long int n = ftell ( <указатель на файл> )* – в переменную *n* передать номер текущей позиции в файле.
- *int z = fgetpos ( <указатель файла>, pros)* - в динамической памяти по адресу *pros* записать номер текущей позиции в файле, в случае успеха функция возвращает 0; в противном случае – любое ненулевое число.
- *int unlink ( <имя файла> )* – удаление файла, при успехе функция возвращает 0, при неуспехе - -1.
- *int rename( <старое имя>, <новое имя> )* – переименование файла, при успехе функция возвращает 0, при неуспехе - -1.
- *int feof ( <указатель на файл> )* возвращает 0, если конец файла не достигнут, любое ненулевое число, если достигнут.

- *int ferror( <указатель на файл> )*  
возвращает ненулевое значение, если при работе с файлом возникла ошибка, 0 – в противном случае.

# Примеры работы с файлами

## Запись данных в текстовый файл

*Пример 1.* Создать вещественный массив случайным образом и сохранить его в текстовом файле.

```
#include <conio.h>  
int main(int argc, char *argv[])  
{ int n;  
FILE *f;  
int flag = 1;  
char name[25];
```

```
system("chcp 1251");  
// Создадим цикл, позволяющий корректировать ввод  
// имени файла  
do  
{  
printf( "Введите имя создаваемого файла: ");  
scanf("%s",name);  
// Попытка открыть файл для чтения. Если такой файл  
// уже существует, то задать вопрос пользователю  
if ((f = fopen(name,"r"))!=NULL)  
{  
    // Заменить существующий файл?
```

```
printf("Файл уже существует. Заменить? (y/n)");  
char ch = getch();  
// Если пользователь нажал кнопку «n», очистить  
// экран, вернуться к началу цикла  
if (ch == 'n') { system("cls"); continue; }  
}  
// В этот блок программы управление попадет только  
// если пользователь подтвердил замену или задал имя  
// несуществующего файла.  
// Создать файл.  
if ((f=fopen(name, "w"))==NULL)  
{  
printf("Ошибка создания файла");  
system("PAUSE");
```

```
break;
}
printf("\nВведите размерность массива: ");
scanf("%d",&n);
for(int i=0;i<n;i++ )
{ float y = rand()
  %100/(rand()%50+1.)-rand()%30;
// На одной строке файла печатать только 10
// элементов.
  if (i>=10&& i%10==0) fprintf(f,"\n");
  fprintf(f,"%8.3f",y);
}
// Закреть файл.
fclose(f);
```

```
// Закончить цикл
    flag = 0;
} while (flag);
printf("Файл создан. ");
system("PAUSE");
return EXIT_SUCCESS;
}
```

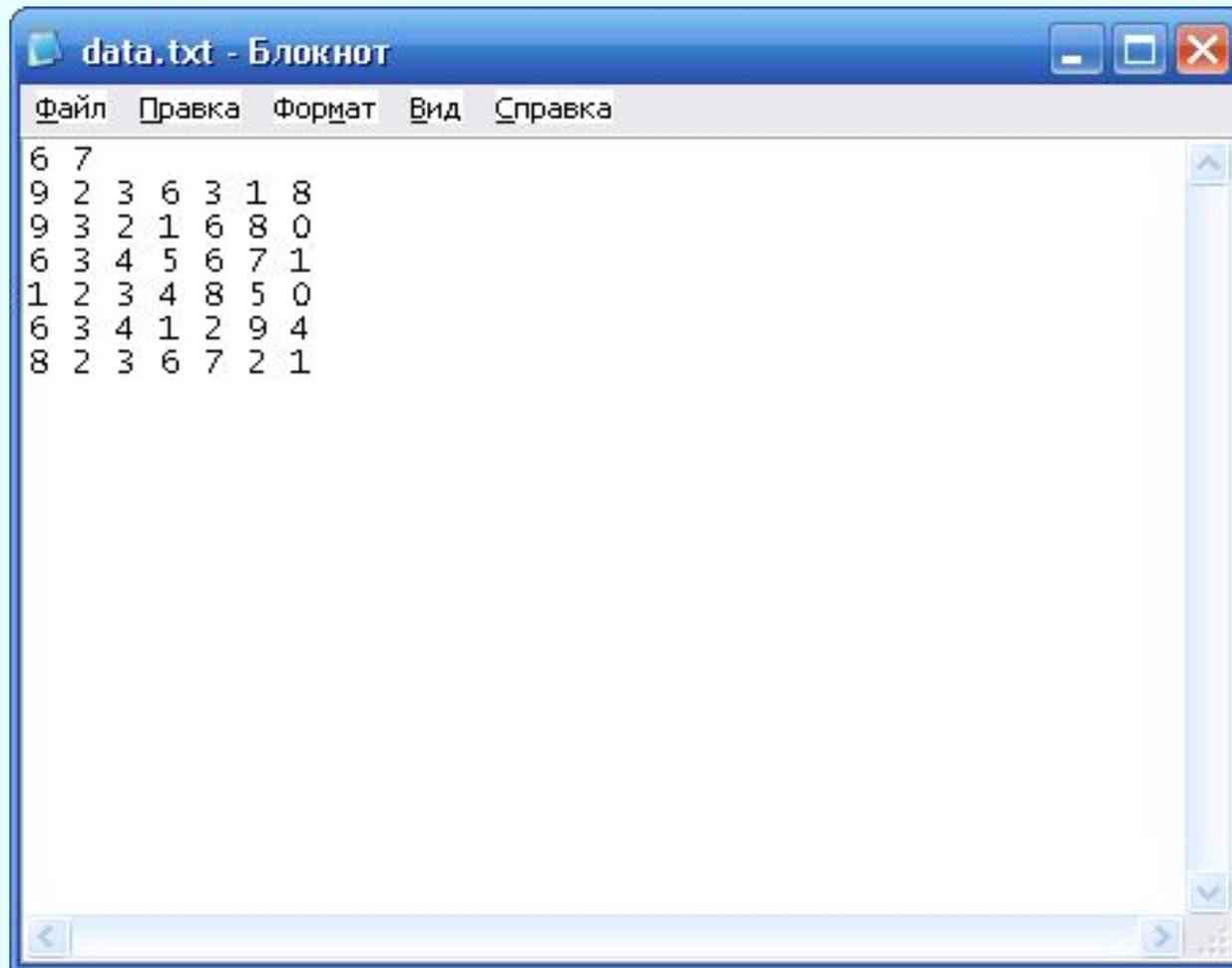
Data - Notepad

File Edit Format View Help

-1.722	-4.000	-13.333	5.667	1.893	-0.022	-20.270	-1.667	-18.212	-4.158
-9.259	-20.100	-23.660	2.500	-2.353	0.524	-14.211	-23.615	-19.300	-26.633
-19.556	-14.923	-3.070	-13.979	-7.000	-13.857	-7.102	-3.343	-14.634	-25.625
-24.900	-27.395	-8.455	-3.794	-19.667	-24.929	-9.027	-20.200	-14.667	-10.479
-8.902	-11.000	-8.800	42.000	-14.360	-7.250	-6.816	18.750	-14.500	-15.364
-25.532	-5.857	-12.000	-24.000	-7.368	-19.196	1.000	-8.560	-13.636	1.211
-20.686	-2.789	-7.000	-10.690	-15.588	-15.091	-28.133	-20.556	-19.000	-1.857
-21.333	-1.775	-0.361	-25.208	-4.500	-18.000	-6.298	-5.975	-10.875	-21.436
-8.733	-4.667	-15.615	-20.333	-6.000	-22.462	-16.794	-5.067	-0.347	-26.682
-19.889	1.423	-3.000	-12.250	-14.975	-17.333	-25.722	25.556	-28.030	-26.545

Ln 10, Col 90

# Чтение данных из текстового файла



The image shows a screenshot of a Notepad window titled "data.txt - Блокнот". The window contains a text file with the following content:

```
6 7
9 2 3 6 3 1 8
9 3 2 1 6 8 0
6 3 4 5 6 7 1
1 2 3 4 8 5 0
6 3 4 1 2 9 4
8 2 3 6 7 2 1
```

*Пример 2.* На первой строке в файле записана размерность целочисленной матрицы. Далее – сама матрица. Считать матрицу в память и вывести ее на экран. Данные записаны в файле *data.txt*.

```
int main(int argc, char *argv[])  
{  
    system("chcp 1251");  
    FILE *f;
```

```
f = fopen("data.txt", "r");  
// Проверка ошибки открытия файла  
if (f==NULL) {  
    printf("Файл не найден... /n Для окончания  
        работы нажмите любую клавишу...");  
    system("pause");  
    exit(0);  
}  
int n,m;  
// Чтение размерности матрицы  
fscanf(f,"%d",&n);  
fscanf(f,"%d",&m);  
int **a;  
// Выделение памяти под матрицу
```

```
a = new int* [n];
for(int i=0;i<n;i++)
    a[i] = new int [m];
// Чтение матрицы
for(int i=0;i<n;i++)
    for(int j=0;j<m;j++)
        fscanf(f,"%d",&a[i][j]);
printf("Прочитана матрица: \n");
// Печать элементов матрицы
for( int i=0;i<n;i++) {
    for(int j=0;j<m;j++)
        printf("%5d",a[i][j]);
    printf("\n"); }
system("pause");
}
```

*Пример 3.* В текстовом файле записано произвольное количество чисел. Считать данные из файла в массив и вывести на экран.

```
int main(int argc, char *argv[])  
{  
    system("chcp 1251");  
    FILE *f;
```

```
f = fopen("my.txt", "r");  
// Проверка ошибки открытия файла  
if (f==NULL) {  
printf("Файл не найден... /n Для окончания  
работы нажмите любую клавишу...");  
system("PAUSE");  
exit(0);  
}  
int n=0,y;  
int *a;  
// пока не конец файла f  
while (!feof(f))  
{
```

```
// читать элемент и
  int z = fscanf(f,"%d",&y);
  if (!z) continue;
// увеличивать счетчик.
  n++;
}
// После окончания цикла в переменной n хранится
// количество целых чисел, записанных в файле.
// Выделить память под массив.
a = new int [n];
// Указатель чтения-записи файла передвинуть в
// начало.
fseek(f,0,SEEK_SET);
// Читать n целых чисел из файла в массив.
for(int i=0;i<n;i++)
```

```
fscanf(f, "%d", &a[i]);  
printf("Прочитан массив: \n");  
for(int i=0; i<n; i++)  
    printf("%5d", a[i]);  
printf("\n");  
system("PAUSE");  
}
```

## Изменение текстового файла

*Пример 4.* В текстовом файле расположен произвольный текст. Не считывая весь текст в память изменить все первые буквы слов на прописные.

```
int main(int argc, char *argv[])
{
// Откроем файл для чтения с дополнением.
// Символ окончания файла в этом случае
// автоматически удаляется.
FILE *f = fopen("text.txt", "r+");
```

```
FILE *f1 = fopen("text.txt", "r+");  
if (f==NULL) {  
    printf("Файл не найден. \n");  
    system("PAUSE");  
    return EXIT_SUCCESS;  
}  
char word[100];  
long pos1;  
// Организуем бесконечный цикл для  
чтения файла.
```

```
while (1) {
```

```
// Если чтение прошло неуспешно, значит  
// достигнут конец
```

```
// файла,
```

```
// в этом случае нужно закончить выполнение  
// цикла
```

```
if ( fscanf(f,"%s",word)!=1) break;
```

```
// В переменную pos1 получить текущую  
// позицию указателя чтения-записи.
```

```
pos1 = ftell(f);
```

```
// Установить указатель на позицию, с которой  
// было
```

```
// считано слово.
```

```
fseek(f1,pos1-strlen(word),SEEK_SET);
word[0] = toupper(word[0]);
printf(" %s\n",word);
// Записать в файл измененное слово.
fprintf(f1,"%s",word);
}
fclose(f1);
// Вывод измененного файла на экран
fseek(f,0,SEEK_SET);
while (1) {
```

```
if ( fscanf(f,"%s",word)!=1) break;  
    puts(word);  
    }  
fclose(f);  
    system("PAUSE");  
    return EXIT_SUCCESS;  
}
```