ЛАБОРАТОРНАЯ РАБОТА №8

ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ С ИСПОЛЬЗОВАНИЕМ ФАЙЛОВ

***Цель работы**:* изучить особенности способов создания и работы с файлами в языке Си.

Краткие теоретические сведения

Файл – это именованная область внешней памяти. Файлы, в отличии от переменных, предназначены для постоянного хранения больших объемов данных. Файлы располагаются не в оперативной памяти, а на жестких дисках и на внешних носителях.

Любой файл рассматривается как последовательный поток байтов. Такой подход понятия файла дает возможность работать с различными устройствами так же, как и с файлами.

Различаются *текстовые* и *бинарные* файлы.

Прежде чем работать с файлом, его нужно открыть для доступа, т.е. создать и инициализировать область данных, которая содержит информацию о файле: имя, путь и т.д.

ФУНКЦИИ ОТКРЫТИЯ И ЗАКРЫТИЯ ФАЙЛОВ

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

Первый параметр функции **указывает место**, в которое мы собираемся поместить файл, например “mas.txt” – файл с именем mas.txt будет находиться в вашем проекте, “d:\\work\\sved.txt” – файл с именем sved.txt будет находиться на d:, в каталоге work.

Второй параметр функции – это **режим доступа** и **режим вида** файла, который может принимать следующие значения:

режим доступа

r	Открыть файл для чтения
w	Открыть файл для записи, если файл существует, то его содержание теряется
a	Открыть файл для добавления информации в конец файла, если файл существует. Если файл не существует, то он создается
r+	Открыть файл для записи и чтения. Файл должен существовать.
w+	Открыть файл для записи и чтения. Если файл существует, то его содержание теряется.
a+	Открыть файл для записи в конец файла и для чтения, если файл существует. Если файл не существует, то он создается.

режим вида

t текстовый
b бинарный

По умолчанию файл открывается в текстовом режиме.

Если при открытии файла произошла ошибка, функция **fopen** возвращает значение **NULL**.

int fclose (FILE *);

Параметром функции является указатель на файл, который был получен при открытии файла. При завершении программы все незакрытые файлы автоматически закрываются системой.

Для закрытия нескольких файлов введена функция, объявленная следующим образом: **void** fcloseall(**void**);

Пример последовательности операторов, необходимых для корректной работы с файлом:

```
#include <stdio.h>
    . . .
    FILE *f_my;
        if(!(f_my = fopen("rez.txt", "r+t")))
        {
        puts("\n Ошибка при открытии файла!");
        getch();  return;
        }
        //      Работа с файлом
    fclose(f_my);
    . . .
```

ФУНКЦИИ ЗАПИСИ И ЧТЕНИЯ

Текстовый файл – это последовательность символов, которая может быть разбита на строки с помощью символа '\n'. *Текстовый файл* может быть просмотрен и отредактирован любым текстовым редактором и имеет простую структуру.

Функции работы с текстовыми файлами удобны при создании результирующих файлов для отчетов по лабораторным и курсовым работам.

Функции записи в текстовый файл

int fprintf(FILE*, **const char***,…..);

данная функция работает аналогично функции printf, только добавлен параметр – указатель на файл, к которому применяется данная функция.

int fputs(**char** *, FILE *);

данная функция записывает только строки в файл. Первый параметр – это имя строки, а второй параметр – указатель на файл. Строка записывается в файл до тех пор, пока не встретится '\0', который в файл не записывается.

Функции чтения из текстового файла

int fscanf(FILE*, **const char***,…..);

данная функция работает аналогично функции scanf, только добавлен параметр – указатель на файл, к которому применяется данная функция.

Функция возвращает:

положительное число, т.е. количество прочитанной информации;

0, если не удалось прочитать информацию;

-1, если достигли конца файла (EOF).

int fgets(**char** *,**int** , FILE *);

данная функция выполняет чтение только строк из файла. Первый параметр имя строки, второй параметр размер строки, а третий параметр указатель на файл. Чтение выполняется до тех пор, пока не встретит символ '\n' или пока не будет считано m символов.

Бинарный файл – это набор двоичной информации. Каждая программа структуру бинарного файла определяет сама. Просмотреть и отредактировать бинарный файл можно только с помощью кода программы.

int fread(**void** *ptr, **int** size, **int** n, FILE *f);

считывает **n** блоков по **size** байт каждый из файла **f** в область памяти, на которую указывает **ptr** (необходимо заранее отвести память под считываемый блок).

int fwrite(**void** *ptr, **int** size, **int** n, FILE *f);

записывает **n** блоков по **size** байт каждый из области памяти, на которую указывает **ptr** в файл f.

Для создания баз данных удобнее пользоваться функциями работы с бинарными файлами.

НЕКОТОРЫЕ ФУНКЦИИ ДЛЯ РАБОТЫ С ФАЙЛАМИ

1. *Функция определения достижения конца файла.*

int feof (FILE*);

Функция возвращает: **0** пока указатель(курсор) не достиг конца файла;

-1 указатель(курсор) достиг конца файла .

2.*Установка указателя (курсора) на заданную позицию*

int fseek(FILE*, **long** offset, **int** whence);

Параметр **long** offset – задает количество байт, на которое необходимо сместить указатель (курсор) в направлении, указанном параметром **int** whence.

Третий параметр **int** whence (смещение**)** может принимать следующие три значения:

SEEK_SET или **0** Смещение от начала файла.

SEEK_CUR или **1** Смещение от текущей позиции указателя (курсора).

SEEK_END или **2** Смещение от конца файла.

Второй параметр **long** offset (позиция**)** величина смещения может быть как положительной, так и отрицательной.

3. *Определение позиции указателя (курсора).*

long ftell(FILE*);

Возвращает *значение указателя* (курсора) на текущую позицию файла. В случае ошибки возвращает число **-1**.

ПРИМЕРЫ РАБОТЫ С ФАЙЛАМИ:

1. **Записываем в текстовый файл номер имя и баланс клиента. Ввод данных прекращается, когда введем сочетание клавиш <Ctrl+z>, что в системе IBM PC означает конец стандартного потока клавиатуры, т.е. EOF.**

```
#include<stdio.h>

void main(){
int account;
char name[30];
double balance;
FILE *cfPtr;
if((cfPtr=fopen("clients.dat","w"))== NULL)
{       printf("File could not be opened\n");
        return;
}
printf("Enter the account, name and balance.\n");
//в системе IBM PC EOF–это сочетание клавиш //<ctrl+z>
printf("Enter EOF to end input.\n");
printf("?  ");
scanf("%d%s%lf",&account,name,&balance);

while(!feof(stdin)){
fprintf(cfPtr,"%-10d%-20s%.2lf\n",account,name,
                balance);
printf("?  ");
scanf("%d%s%lf",&account,name,&balance);
}

fclose(cfPtr);
}
```

2. **Чтение из текстового файла, номер, имя и баланс клиента.**

```
#include <stdio.h>

void main(){
int account;
char name[30];
double balance;
FILE *cfPtr;
if((cfPtr=fopen("clients.txt","r"))== NULL)
{
        printf("File could not be opened\n");
        return;
}
```

```
printf("%-10s%-10s%s\n","Account","Name",
                "Balance");
fscanf(cfPtr,"%d%s%lf",&account,name,&balance);

while(!feof(cfPtr)){
printf("%-10d%-10s %.2lf\n",account,name,balance);
fscanf(cfPtr,"%d%s%lf",&account,name,&balance);
}

fclose(cfPtr);
}
```

3. **В текстовом файле записаны двузначные числа. Считать данные числа с конца файла.**

```
#include <stdio.h>
#include <windows.h>

char str[100];
char* RUS( char*);

void main(){
char name[12];
int x;
FILE *k;
printf("%s\t",RUS("Введите имя файла  имя.txt:")); scanf("%s",name);
if( !( k = fopen (name, "r") ) )
{
        puts(RUS("Файл не найден"));
        return;
}

fseek ( k ,  0* sizeof( char ), SEEK_END);
int n = ftell(k)/ sizeof ( char );

for ( int i = 2 ; i <= n ; i += 3)
{
        fseek (k, (-i)* sizeof ( char ), SEEK_END);
        if ( fscanf( k, "%d" , &x) != 1)
        {
                puts(RUS("Не соответствие типа"));
                break;
        }
        printf("%-5d",x);
}

fclose(k);
```

```
}

char* RUS( char*s)
{
	CharToOem(s,str);
	return str;
}
```

4. **Создать бинарный файл chisla.bin, куда будут записываться числа, введенные с клавиатуры, ввод завершается при вводе сочетания клавиш <Ctrl+z>.**

```
#include <stdio.h>

void main()
{
	FILE *k;
	if( !( k = fopen ("chisla.bin", "wb") ) )
	{
		puts("Errors!!!!");
		return;
	}
	int x;
	scanf("%d", &x);
	while( !feof(stdin))
	{
		fwrite (&x, sizeof ( int ), 1 , k);
		scanf("%d", &x);
	}
	fclose(k);
}
```

5. **Работа с бинарным файлом chisla.bin с использованием функций**.

```
#include <stdio.h>

//Считает количество чисел в файле
int getFileSize(FILE *f);

//Возвращает число записанное на i-ой позиции
int getElementAtPosition(int i,FILE*f);

//Заменяем число на позиции pos на число element
void setFileElements(int pos,int element,FILE *f);

//выводит содержание файла
void print_file (FILE *f);

void main()
```

```
{
FILE *f1;
int n,n1;
if( ! (f1=fopen("chisla.bin","r+b") ) )
{
	puts("FILE ne otkrit!!!!!!");
	return;
}

puts("\nSoderzanie faila:");
print_file(f1);

puts("\nKolichestvo elementov:");
printf("%d\n",getFileSize(f1));

puts("\nVvedite element konori xotite pomestit v file: ");
scanf("%d",&n);

puts("\nVvedite kuda vstavit element: ");
scanf("%d",&n1);

setFileElements(n1,n,f1);
puts("\nSoderzanie faila:");

print_file(f1);
fclose(f1);
}
int getFileSize(FILE *f)
{
	fseek(f, 0* sizeof ( int ),SEEK_END);
	int n = ftell(f) / sizeof ( int );
	return n;
}

int getElementAtPosition( int i, FILE*f)
{
	int buf;
	fseek(f, i* sizeof ( int ) ,SEEK_SET);
	fread(&buf, sizeof ( int ),1,f);

	return buf;
}

void print_file (FILE *f)
{
```

```
int n = getFileSize(f);
fseek(f,0 * sizeof ( int),SEEK_SET);

for (int i=0; i < n; i++)
{
        int x;
        if ( ! fread (&x , sizeof (int) , 1 , f) )      break;

        printf ("element # %d = %d\n" , i , x);
}
}

void setFileElements (int pos,int element,FILE*f)
{
fseek (f ,  pos * sizeof (int)  , SEEK_SET );
fwrite (&element ,sizeof (int) , 1 , f);
}
```

ЗАДАЧИ ДЛЯ ВЫПОЛНЕНИЯ:

1. В двух текстовых файлах хранятся упорядоченные числа, количество чисел не одинаковое. Переписать эти числа в третий файл, так чтобы числа были также упорядочены.

2. В двух текстовых файлах хранятся положительные трехзначные числа. В первом файле числа упорядочены в порядке возрастания, а во втором по убыванию. Переписать эти числа в третий файл, так, чтобы числа были упорядочены по возрастанию.

3. В текстовом файле записаны как символы, так и цифры. Переписать символы в один файл, а цифры в другой.

4. В файле text.txt хранится текст, причем, между словами могут быть лишние пробелы. Посчитать:
   1) количество лишних пробелов;
   2) количество символов в файле;
   3) количество знаков препинания.

Переписать содержимое файла text.txt в файл text1.txt без лишних пробелов.

5. В текстовом файле хранится текст. Ввести слово и посчитать, сколько раз данное слово встречается в тексте. Если слова нет в файле, то вывести сообщение об этом.

6. В текстовом файле заменить все строки, начинающиеся с буквы ‘A’ на строки, начинающиеся с буквы ‘C’.

7. Имеются два файла, в которых хранятся двузначные числа, отсортированные по возрастанию. Переписать их в третий файл, упорядочивая по убыванию. Все файлы текстовые. Дополнительных файлов и массивов не использовать.

8. В текстовом файле содержатся даты типа Date..
struct Date{char Month[15];int Day;}
Заменить все даты, у которых поле Month равно ”Июнь” или ”Июль” на “Август”.

9.. Имеется текстовый файл. Найти количество слов, которые начинаются и заканчиваются на одну и ту же букву и вывести их на экран.

10. Отсортировать числа в бинарном файле chisla.bin, не используя дополнительных файлов и массивов.

11. Написать программу, которая удаляет из бинарного файла все нечетные числа. Программа должна содержать следующие функции:

1. функция, которая заполняет, созданный в функции main, бинарный файл целыми числами;
2. функция, которая выводит содержимое бинарного файла;
3. функция, которая удаляет из бинарного файла все четные числа.

(Дополнительных массивов и файлов не использовать).

12. Написать программу, которая определяет минимальный из четных элементов целочисленной последовательности, которая хранится в бинарном файле. Программа должна содержать следующие функции:

1. функция, которая заполняет, созданный в функции main, бинарный файл целыми числами;
2. функция, которая определяет, минимальный из четных элементов последовательности содержащих в бинарном файле.

13. Написать программу, которая заменяет все числа в бинарном файле на их квадраты. Программа должна содержать следующие функции:

1. функция, которая заполняет, созданный в функции main, бинарный файл целыми числами;
2. функция, которая выводит содержимое бинарного файла;
3. функция, заменяет все числа в бинарном файле на их квадраты.

(Дополнительных массивов и файлов не использовать).