Национальный исследовательский университет «Московский авиационный институт»

Факультет №8 «Информационные технологии и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

КУРСОВОЙ ПРОЕКТ

**ПО КУРСУ “ПРАКТИКУМ НА ЭВМ”**

**1 СЕМЕСТР ЗАДАНИЕ №4**

**“СОРТИОВКА И ПОИСК”**

|  |  |
| --- | --- |
| Выполнил студент | Мохляков Павел Александрович |
| Группа | М80-108Б-19 |
| Преподаватель: | Поповкин Александр Викторович |
| Дата |  |
| Оценка |  |

Москва

2020

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 1](#_Toc40788308)

[ЗАДАНИЕ 2](#_Toc40788309)

[ОСНОВНОЙ МЕТОД РЕШЕНИЯ 3](#_Toc40788310)

[ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ 4](#_Toc40788311)

[пФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ 5](#_Toc40788312)

[ОПИСАНИЕ ПРОГРАММЫ 6](#_Toc40788313)

[АЛГОРИТМЫ РАБОТЫ 6](#_Toc40788314)

[ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ 6](#_Toc40788315)

[ИСПОЛЬЗУЕМЫЕ ПЕРЕМЕНЫЕ 8](#_Toc40788316)

[ПРОТОКОЛ 9](#_Toc40788317)

[ЗАКЛЮЧЕНИЕ 18](#_Toc40788318)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 19](#_Toc40788319)

# ЗАДАНИЕ

Составить программу на языке Си с использованием процедур и функций для сортировки таблицы заданным методом и двоичного поиска по ключу в таблице.

Программа должна вводить значения элементов неупорядоченной таблицы и проверять процедуры сортировки в трех случаях: элементы таблицы с самого начала упорядочены элементы; таблицы расставлены в обратном порядке; элементы таблицы не упорядочены. В последнем случае можно использовать встроенные процедуры генерации псевдослучайных чисел.

ВАРИАНТ 16

Метод сортировки: метод простой вставки. Структура таблицы: тип ключа вещественный; длина ключа в батах 16; хранение данных и ключей вместе; число элементов таблицы 8-12.

# ОСНОВНОЙ МЕТОД РЕШЕНИЯ

Программа создает пустой указатель на список, и указатель на массив. Далее запускается меню, где для начала мы выбираем чтение из файла. Чтение происходит посимвольно, отделяя ключ от строки пробелом, ключ и строка добавляются в список и так до конца файла. После чего нам дается возможность отсортировать список.

Сортировка происходит методом простой вставки, начиная с первого элемента элементы переносятся влево по списку пока не дойдут до конца списка или пока слева от него не окажется элемент меньше его. В списке это происходит путем удаления элемента и добавления его на нужную позицию.

После сортировки список копируется в массив. Поиск элемента в массиве происходит по алгоритму бинарного поиска. Берется средний элемент между началом и концом массива. Если выбранный элемент меньше ключа поиска, то начало перемещается на эту позицию, иначе перемещается конец массива. Поиск продолжается пока элемент не найден или расстояние между индексами элементов равно единице.

.

# ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

|  |  |
| --- | --- |
| Аппаратное обеспечение | Ноутбук на базу Intel Core i5 |
| Операционная система | Manjaro 5.4.36 |
| Язык и система программирования | GNU C |
| Число строк | 330 |
| Компиляция программы в терминале | Zsh 5.8 |

Таблица А.1 - Общие сведение о программе

# пФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

Программа предназначена для считывания из файла таблицы, сортировки и поиска в таблице с использованием динамический и статических структур данных.

# ОПИСАНИЕ ПРОГРАММЫ

## АЛГОРИТМЫ РАБОТЫ

1. Подключаем необходимые библиотеки
2. Создаем служебные функции
3. Создаем структуру данных
4. Считываем данные из файла
5. Сортируем таблицу
6. Производим поиск в таблице

## ОПИСАНИЕ ФУНКЦИЙ ПРОГРАММЫ

Таблица А.2 - Функции файла main.c

|  |  |  |
| --- | --- | --- |
| Название | Аргументы и их тип | Описание функции |
| int main() |  | Создает базовые указатели и вызывает интерфес |
| struct list\* rfile() | struct list \*top,FILE \*fl | Считывает данные из файла |
| void menu() | struct list \*top, struct mas\* m | Интерфейс |

Таблица А.3 - Функции файла list.c

|  |  |  |
| --- | --- | --- |
| Название | Аргументы и их тип | Описание функции |
| void Print\_list() | struct list \*top | Выводит список |
| struct list\* Create\_list() | struct list \*top | Создает список |
| int Size\_list() | struct list \*top | Возвращает длину списка |
| struct list\* Push\_list() | struct list\* top,long double key,char \*data, int pos | Добавляет элимент на заданную позицию в списке |
| struct list\* Delete\_list\_pos() | struct list \*top,int pos | Удаляет элимент в заданной позиции |
| void Pop\_list() | struct list \*top,long double \*key, char \*\*data, int pos | Возвращает данные элемента на заданной позиции |
| struct list\* Change\_pos() | struct list \*top,int pos1,int pos2 | Меняет позицию элимента |
| char\* Copy\_mass() | char \*data | Копирует массив |
| struct list\* Sort\_list() | struct list \*top | Сортирует список |
| struct list\* Clear\_list() | struct list \*top | Очищает список |
| struct mas\* List\_mas() | struct list\* top, int \*size | Копирует список в массив |
| void Search\_mas() | long double key,long double accur,struct mas \*m,int size | Бинарный поиск в массиве |
| struct list\* Mix\_list() | struct list\* top | Перемешивает список |

# ИСПОЛЬЗУЕМЫЕ ПЕРЕМЕНЫЕ

Таблица А.4 - Общие переменные

|  |  |  |  |
| --- | --- | --- | --- |
| Имя переменной | Начальное значение | Тип | Назначение |
| top |  | struct list\* | Указатель на вершину списка |
| data |  | int | Строка данных |
| key |  | int | Ключ |
| size |  | int | Длина списка |

Таблица А.5 - Переменные menu() main.c

|  |  |  |  |
| --- | --- | --- | --- |
| Имя переменной | Начальное значение | Тип | Назначение |
| sort | 0 | int | Флаг сортировки списка |
| read | 0 | int | Заполнен ли список |
| sw |  | int | Переменная выбора |
| fl |  | FILE\* | Указатель на файл |
| fname |  | char\* | Путь до фала |

# ПРОТОКОЛ

[pavel@lenovo kp9]$ cat main.c

#include <stdio.h>

#include <stdlib.h>

#include "list.h"

#include <math.h>

struct list\* rfile(struct list \*top,FILE \*fl)

{

top = Clear\_list(top);

int i=1;

long double key;

char \*data;

char ch;

ch = fgetc(fl);

while(ch!=EOF)

{

int j=0;

int flagm=0;

data = malloc(256\*sizeof(char));

key=0;

long double mn=0.1;

while((int)ch!=46 && (int)ch!=32)

{

if((int)ch == 45)

{

flagm=1;

ch = fgetc(fl);

continue;

}

key\*=10;

key+=(int)ch-48;

ch = fgetc(fl);

}

if((int)ch!=32)

{

ch = fgetc(fl);

while((int)ch!=32)

{

key+=(((int)ch-48)\*mn);

mn\*=0.1;

ch = fgetc(fl);

}

}

if(flagm == 1) key\*=-1;

ch = fgetc(fl);

while((int)ch!=10)

{

data[j]=ch;

j++;

ch = fgetc(fl);

}

top=Push\_list(top,key,data,i);

i++;

//ch = fgetc(fl);

ch = fgetc(fl);

}

return top;

}

void menu(struct list \*top, struct mas\* m)

{

int size = 0;

int sort = 0;

int read = 0;

while(1)

{

int sw;

printf("Список действий:\n");

printf("0 - Выход\n");

printf("1 - Прочитать файл\n");

if(read == 1) printf("2 - Сортировка таблицы\n");

else printf("2 - Сортировка таблицы(доступно после считывания файла)\n");

if(sort == 1) printf("3 - Поиск в таблице\n");

else printf("3 - Поиск в таблице (не доступно на не отсортированной таблице)\n");

if(read == 1) printf("4 - Перемешать таблицу\n");

else printf("4 - Перемешать таблицу(доступно после считывания файла)\n");

printf("Введите вариант: ");

scanf("%d",&sw);

if(sw==0)

{

break;

}

else if(sw ==1)

{

FILE \*fl;

char fname[80];

printf("Введите имя файла:");

scanf("%s",fname);

if ((fl = fopen(fname, "r")) == NULL)

{

printf("Не удалось открыть файл\n");

continue;

}

top = rfile(top,fl);

fclose(fl);

Print\_list(top);

sort = 0;

read = 1;

}

else if(sw == 2 && read == 1)

{

top = Sort\_list(top);

Print\_list(top);

m = List\_mas(top,&size);

sort = 1;

}

else if(sw == 3 && sort == 1)

{

printf("Введите ключ для поиска:");

long double key;

scanf("%Lf",&key);

long double accur;

printf("Введите точность:");

scanf("%Lf",&accur);

Search\_mas(key,accur,m,size);

}

else if(sw == 4 && read == 1)

{

top = Mix\_list(top);

sort = 0;

Print\_list(top);

}

}

}

int main()

{

struct list \*l\_tab = NULL;

struct mas \*m = NULL;

menu(l\_tab,m);

return 0;

}

[pavel@lenovo kp9]$ cat list.c

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include "list.h"

struct list\* Push\_list(struct list\* top,long double key,char \*data, int pos)

{

if(top==NULL)

{

top = Create\_list(top);

top->key = key;

top->data = data;

}

else

{

if(pos > Size\_list(top))

{

struct list \*q;

q=malloc(sizeof(struct list));

q->key = key;

q->data = data;

q->next = NULL;

struct list \*m = top;

while(m->next!=NULL)

{

m = m->next;

}

m->next = q;

q->prev = m;

}

else

{

struct list \*q;

q = malloc(sizeof(struct list));

q->key = key;

q->data = data;

if(pos == 1)

{

q->next = top;

top->prev = q;

q->prev = NULL;

top = q;

}

else

{

struct list \*m = top;

for(int i = 2;i<pos;i++) m = m->next;

q->next = m->next;

m->next = q;

q->prev = m;

if(q->next!=NULL)

{

q->next->prev = q;

}

}

}

}

return top;

}

struct list\* Create\_list(struct list\* top)

{

if(top == NULL)

{

top=malloc(sizeof(struct list));

top->next=NULL;

top->prev=NULL;

top->key=0;

top->data=NULL;

}

return top;

}

int Size\_list(struct list\* top)

{

int size=0;

while(top!=NULL)

{

top=top->next;

size++;

}

return size;

}

void Print\_list(struct list \*top)

{

while(top!=NULL)

{

printf("%Lf\t%s\n",top->key,top->data);

top = top->next;

}

}

struct list\* Delete\_list\_pos(struct list \*top,int pos)

{

if(top != NULL)

{

struct list \*q = top;

if(pos > Size\_list(top)) pos = Size\_list(top);

if(pos != 0)

{

if(pos == 1)

{

top = top->next;

top->prev = NULL;

free(q);

}

else

{

for(int i = 2;i < pos;i++) q = q->next;

struct list \*m = q->next;

q->next = m->next;

if(m->next!=NULL)

{

struct list \*t = m->next;

t = q->next;

t->prev = q;

}

free(m);

}

}

}

return top;

}

void Pop\_list(struct list \*top,long double \*key, char \*\*data, int pos)

{

int i;

for(i = 1;i<pos;i++) top = top->next;

\*key=top->key;

\*data=top->data;

}

struct list\* Change\_pos(struct list \*top,int pos1,int pos2)

{

long double key;

char \*data;

Pop\_list(top,&key,&data,pos1);

data = Copy\_mass(data);

top = Delete\_list\_pos(top,pos1);

top = Push\_list(top,key,data,pos2);

return top;

}

char\* Copy\_mass(char \*data)

{

char \*data\_copy = NULL;

data\_copy = malloc(256\*sizeof(char));

for(int i=0;i<256;i++)

{

data\_copy[i] = data[i];

}

return data\_copy;

}

struct list\* Sort\_list(struct list \*top)

{

for(int i=1;i<=Size\_list(top);i++)

{

struct list \*m = top;

for(int j = 1; j < i; j++) m = m->next;

struct list \*q = m;

int j;

for(j=i;j>=1;j--)

{

if(q->prev == NULL) break;

if(q->prev->key < m->key) break;

q = q->prev;

}

top = Change\_pos(top,i,j);

}

return top;

}

struct list\* Clear\_list(struct list \*top)

{

while(top!=NULL)

{

if(Size\_list(top) == 1)

{

free(top);

top = NULL;

break;

}

top = Delete\_list\_pos(top,1);

}

return top;

}

struct mas\* List\_mas(struct list\* top, int \*size)

{

struct mas \*m;

\*size = Size\_list(top);

m = malloc(sizeof(struct mas));

for(int i=0;i<\*size;i++)

{

m[i].key = top->key;

m[i].data = top->data;

top = top->next;

}

return m;

}

void Search\_mas(long double key,long double accur,struct mas \*m,int size)

{

int min\_ind = 0;

int max\_ind = size-1;

int ind;

while(max\_ind - min\_ind > 1)

{

ind = (min\_ind + max\_ind)/2;

if(key >= m[ind].key - accur && key <= m[ind].key + accur) break;

else if(m[ind].key < key) min\_ind = ind;

else max\_ind = ind;

}

if(key >= m[ind].key - accur && key <= m[ind].key + accur) printf("%Lf\t%s\n",m[ind].key,m[ind].data);

else if(key >= m[max\_ind].key - accur && key <= m[max\_ind].key + accur) printf("%Lf\t%s\n",m[max\_ind].key,m[max\_ind].data);

else if(key >= m[min\_ind].key - accur && key <= m[min\_ind].key + accur) printf("%Lf\t%s\n",m[min\_ind].key,m[min\_ind].data);

else printf("Элимент не найден\n");

}

struct list\* Mix\_list(struct list\* top)

{

int size = Size\_list(top);

for(int i=0;i<2;i++)

{

srand(time(NULL));

for(int j=1;j<=size;j++)

{

int r = rand()%size;

top = Change\_pos(top,j,r);

printf("-----------------------\n");

//Print\_list(top);

}

}

return top;

}

[pavel@lenovo kp9]$ cat list.h

#ifndef \_LIST\_

#define \_LIST\_

struct list{

struct list\* next;

struct list\* prev;

long double key;

char \*data;

};

struct mas

{

long double key;

char \*data;

};

void Print\_list(struct list \*top);

int Size\_list(struct list\* top);

struct list\* Create\_list(struct list\* top);

struct list\* Push\_list(struct list\* top,long double key,char \*data, int pos);

struct list\* Delete\_list\_pos(struct list \*top,int pos);

void Pop\_list(struct list \*top,long double \*key, char \*\*data, int pos);

struct list\* Change\_pos(struct list \*top,int pos1,int pos2);

char\* Copy\_mass(char \*data);

struct list\* Sort\_list(struct list \*top);

struct list\* Clear\_list(struct list \*top);

struct mas\* List\_mas(struct list\* top, int \*size);

void Search\_mas(long double key,long double accur,struct mas \*m,int size);

struct list\* Mix\_list(struct list\* top);

#endif

[pavel@lenovo kp9]$ cat Makefile

CC=gcc

CFLAGS= -c -Wall -g

LFLAGS= -g -lm

all: kp9

kp9: main.o list.o

$(CC) $(LFLAGS) main.o list.o -o prog

main.o: main.c

$(CC) $(CFLAGS) main.c

list.o: list.c

$(CC) $(CFLAGS) list.c

clean:

rm -rf \*.o prog

[pavel@lenovo kp9]$ ./prog

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы(доступно после считывания файла)

3 - Поиск в таблице (не доступно на не отсортированной таблице)

4 - Перемешать таблицу(доступно после считывания файла)

Введите вариант: 1

Введите имя файла:test

12.550000 ░░░░▄▄▄▄░░░░▄▄████████████▄▄░░░░▄▄▄▄░░░░

11.000000 ░▄▀▀░░░▀▀█▄██████████████████▄█▀▀░░░▀▀▄░

10.000000 ░░░▄▄▄▄▄▄██████████████████████▄▄▄▄▄▄░░░

9.000000 ░▄▀░░▄▄▄████████████████████████▄▄▄░░▀▄░

8.000000 ▀░░█▀░░▄████████████████████████▄░░▀█░░▀

7.000000 ░░█░░▄██▀▀████████████████████▀▀██▄░░█░░

6.000000 ░█░░▄██░░░▀▀████████████████▀▀░░░██▄░░█░

5.000000 ░░░░▀███▄░░░░░▀▀▀██████▀▀▀░░░░░▄███▀░░░░

4.000000 ░░░░░░▀██████████▄░░░░▄██████████▀░░░░░░

3.000000 ░░░░░░░░░█████████▄░░▄█████████░░░░░░░░░

2.000000 ░░░░░░░░░▀████░░░▀▀░░▀▀░░░████▀░░░░░░░░░

1.000000 ░░░░░░░░░░▀▀███▄░░░░░░░░▄███▀▀░░░░░░░░░░

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице (не доступно на не отсортированной таблице)

4 - Перемешать таблицу

Введите вариант: 4

6.000000 ░█░░▄██░░░▀▀████████████████▀▀░░░██▄░░█░

8.000000 ▀░░█▀░░▄████████████████████████▄░░▀█░░▀

2.000000 ░░░░░░░░░▀████░░░▀▀░░▀▀░░░████▀░░░░░░░░░

4.000000 ░░░░░░▀██████████▄░░░░▄██████████▀░░░░░░

3.000000 ░░░░░░░░░█████████▄░░▄█████████░░░░░░░░░

7.000000 ░░█░░▄██▀▀████████████████████▀▀██▄░░█░░

12.550000 ░░░░▄▄▄▄░░░░▄▄████████████▄▄░░░░▄▄▄▄░░░░

1.000000 ░░░░░░░░░░▀▀███▄░░░░░░░░▄███▀▀░░░░░░░░░░

11.000000 ░▄▀▀░░░▀▀█▄██████████████████▄█▀▀░░░▀▀▄░

10.000000 ░░░▄▄▄▄▄▄██████████████████████▄▄▄▄▄▄░░░

9.000000 ░▄▀░░▄▄▄████████████████████████▄▄▄░░▀▄░

5.000000 ░░░░▀███▄░░░░░▀▀▀██████▀▀▀░░░░░▄███▀░░░░

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице (не доступно на не отсортированной таблице)

4 - Перемешать таблицу

Введите вариант: 2

1.000000 ░░░░░░░░░░▀▀███▄░░░░░░░░▄███▀▀░░░░░░░░░░

2.000000 ░░░░░░░░░▀████░░░▀▀░░▀▀░░░████▀░░░░░░░░░

3.000000 ░░░░░░░░░█████████▄░░▄█████████░░░░░░░░░

4.000000 ░░░░░░▀██████████▄░░░░▄██████████▀░░░░░░

5.000000 ░░░░▀███▄░░░░░▀▀▀██████▀▀▀░░░░░▄███▀░░░░

6.000000 ░█░░▄██░░░▀▀████████████████▀▀░░░██▄░░█░

7.000000 ░░█░░▄██▀▀████████████████████▀▀██▄░░█░░

8.000000 ▀░░█▀░░▄████████████████████████▄░░▀█░░▀

9.000000 ░▄▀░░▄▄▄████████████████████████▄▄▄░░▀▄░

10.000000 ░░░▄▄▄▄▄▄██████████████████████▄▄▄▄▄▄░░░

11.000000 ░▄▀▀░░░▀▀█▄██████████████████▄█▀▀░░░▀▀▄░

12.550000 ░░░░▄▄▄▄░░░░▄▄████████████▄▄░░░░▄▄▄▄░░░░

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице

4 - Перемешать таблицу

Введите вариант: 3

Введите ключ для поиска:12.55

Введите точность:0.001

12.550000 ░░░░▄▄▄▄░░░░▄▄████████████▄▄░░░░▄▄▄▄░░░░

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице

4 - Перемешать таблицу

Введите вариант: 1

Введите имя файла:text.txt

1.000000 .............................8........8.....................

2.000000 ........................8..................8................

3.000000 ............................$$$$$$$$$$$$$...................

4.000000 .....................8......$$$$$$$$$$$$$.......8...........

5.000000 ............................$$$$$$$$$$$$$...................

6.000000 .................8..........$$$$$$$$$$$$$..........8........

7.000000 ............................$$$$$$$$$$$$$...................

8.000000 .........$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

9.000000 .........$$;;;.':$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:.;;;$

10.000000 .........$$$$$$:,.':$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:'...$$$$

11.000000 .........$$$$$$$$....':$$$$$$$$$$$$$$$$$$$$$$$$$..'..:'$$$$$

12.000000 ..................`.:....:..$$$$$$$:.'.$$...:.'....:'.......

13.000000 ................8....`:....:.:$$$$:..;:$$:'....:..'..8......

14.000000 .......................`.........'....'.'.....:.............

15.000000 ..................8......:.........:......:.'.......8.......

16.000000 ..........................`.:......:....,.........8.........

16.777778 .......................8....$:.........:$.....8.............

19.000000 ..........................8.$...........$.8.................

20.000000 ............................$$:.......:$$...................

21.000000 ............................$$$:......'$$...................

22.000000 ............................$$$..'.`..:$$...................

23.000000 ............................$$.........:$...................

24.000000 ............................$$......:...$...................

25.000000 ............................$$..........$...................

26.000000 ............................$$$........;$...................

27.000000 ............................$$$$$......;$...................

28.000000 ............................$$$$$$....,$$...................

29.000000 ............................$$$$$`....$$$...................

30.000000 ............................$$$$$:...`$$$...................

31.000000 ............................$$$$$,....$$$...................

32.000000 ............................$$$$$$....$$$...................

33.000000 ............................$$$$$$$...$$$...................

34.000000 ............................$$$$$$$...$$$...................

35.000000 ............................$$$$$$'....$$...................

36.000000 ............................$$$$$$....$$$...................

36.000000 ............................$$$$$$$..$$$$...................

37.000000 ............................$$$$$$$$$$$$$...................

38.000000 ............................$$$$$$$$$$$$$...................

39.000000 ............................$$$$$$$$$$$$$...................

40.000000 ............................$$$$$$$$$$$$$...................

41.000000 ............................$$$$$$$$$$$$$...................

42.000000 ............................$$$$$$$$$$$$$...................

43.000000 ...........................$$$$$$$$$$$$$$$..................

44.000000 ..........................$$$$$$$$$$$$$$$$$.................

45.000000 .........................$$$$$$$$$$$$$$$$$$$................

46.000000 .........................$$$$$$$$$$$$$$$$$$$................

47.000000 .........................$$$$$$$$$$$$$$$$$$$................

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице (не доступно на не отсортированной таблице)

4 - Перемешать таблицу

Введите вариант: 4

11.000000 .........$$$$$$$$....':$$$$$$$$$$$$$$$$$$$$$$$$$..'..:'$$$$$

19.000000 ..........................8.$...........$.8.................

37.000000 ............................$$$$$$$$$$$$$...................

29.000000 ............................$$$$$`....$$$...................

44.000000 ..........................$$$$$$$$$$$$$$$$$.................

43.000000 ...........................$$$$$$$$$$$$$$$..................

20.000000 ............................$$:.......:$$...................

8.000000 .........$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

14.000000 .......................`.........'....'.'.....:.............

6.000000 .................8..........$$$$$$$$$$$$$..........8........

35.000000 ............................$$$$$$'....$$...................

30.000000 ............................$$$$$:...`$$$...................

23.000000 ............................$$.........:$...................

28.000000 ............................$$$$$$....,$$...................

9.000000 .........$$;;;.':$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:.;;;$

3.000000 ............................$$$$$$$$$$$$$...................

42.000000 ............................$$$$$$$$$$$$$...................

46.000000 .........................$$$$$$$$$$$$$$$$$$$................

33.000000 ............................$$$$$$$...$$$...................

13.000000 ................8....`:....:.:$$$$:..;:$$:'....:..'..8......

34.000000 ............................$$$$$$$...$$$...................

12.000000 ..................`.:....:..$$$$$$$:.'.$$...:.'....:'.......

5.000000 ............................$$$$$$$$$$$$$...................

31.000000 ............................$$$$$,....$$$...................

15.000000 ..................8......:.........:......:.'.......8.......

38.000000 ............................$$$$$$$$$$$$$...................

2.000000 ........................8..................8................

39.000000 ............................$$$$$$$$$$$$$...................

21.000000 ............................$$$:......'$$...................

45.000000 .........................$$$$$$$$$$$$$$$$$$$................

10.000000 .........$$$$$$:,.':$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:'...$$$

41.000000 ............................$$$$$$$$$$$$$...................

47.000000 .........................$$$$$$$$$$$$$$$$$$$................

26.000000 ............................$$$........;$...................

27.000000 ............................$$$$$......;$...................

16.000000 ..........................`.:......:....,.........8.........

25.000000 ............................$$..........$...................

4.000000 .....................8......$$$$$$$$$$$$$.......8...........

16.777778 .......................8....$:.........:$.....8.............

22.000000 ............................$$$..'.`..:$$...................

24.000000 ............................$$......:...$...................

36.000000 ............................$$$$$$$..$$$$...................

1.000000 .............................8........8.....................

40.000000 ............................$$$$$$$$$$$$$...................

36.000000 ............................$$$$$$....$$$...................

32.000000 ............................$$$$$$....$$$...................

7.000000 ............................$$$$$$$$$$$$$...................

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице (не доступно на не отсортированной таблице)

4 - Перемешать таблицу

Введите вариант: 2

1.000000 .............................8........8.....................

2.000000 ........................8..................8................

3.000000 ............................$$$$$$$$$$$$$...................

4.000000 .....................8......$$$$$$$$$$$$$.......8...........

5.000000 ............................$$$$$$$$$$$$$...................

6.000000 .................8..........$$$$$$$$$$$$$..........8........

7.000000 ............................$$$$$$$$$$$$$...................

8.000000 .........$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

9.000000 .........$$;;;.':$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:.;;;$

10.000000 .........$$$$$$:,.':$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$:'...$$$$

11.000000 .........$$$$$$$$....':$$$$$$$$$$$$$$$$$$$$$$$$$..'..:'$$$$$

12.000000 ..................`.:....:..$$$$$$$:.'.$$...:.'....:'.......

13.000000 ................8....`:....:.:$$$$:..;:$$:'....:..'..8......

14.000000 .......................`.........'....'.'.....:.............

15.000000 ..................8......:.........:......:.'.......8.......

16.000000 ..........................`.:......:....,.........8.........

16.777778 .......................8....$:.........:$.....8.............

19.000000 ..........................8.$...........$.8.................

20.000000 ............................$$:.......:$$...................

21.000000 ............................$$$:......'$$...................

22.000000 ............................$$$..'.`..:$$...................

23.000000 ............................$$.........:$...................

24.000000 ............................$$......:...$...................

25.000000 ............................$$..........$...................

26.000000 ............................$$$........;$...................

27.000000 ............................$$$$$......;$...................

28.000000 ............................$$$$$$....,$$...................

29.000000 ............................$$$$$`....$$$...................

30.000000 ............................$$$$$:...`$$$...................

31.000000 ............................$$$$$,....$$$...................

32.000000 ............................$$$$$$....$$$...................

33.000000 ............................$$$$$$$...$$$...................

34.000000 ............................$$$$$$$...$$$...................

35.000000 ............................$$$$$$'....$$...................

36.000000 ............................$$$$$$....$$$...................

36.000000 ............................$$$$$$$..$$$$...................

37.000000 ............................$$$$$$$$$$$$$...................

38.000000 ............................$$$$$$$$$$$$$...................

39.000000 ............................$$$$$$$$$$$$$...................

40.000000 ............................$$$$$$$$$$$$$...................

41.000000 ............................$$$$$$$$$$$$$...................

42.000000 ............................$$$$$$$$$$$$$...................

43.000000 ...........................$$$$$$$$$$$$$$$..................

44.000000 ..........................$$$$$$$$$$$$$$$$$.................

45.000000 .........................$$$$$$$$$$$$$$$$$$$................

46.000000 .........................$$$$$$$$$$$$$$$$$$$................

47.000000 .........................$$$$$$$$$$$$$$$$$$$................

Список действий:

0 - Выход

1 - Прочитать файл

2 - Сортировка таблицы

3 - Поиск в таблице

4 - Перемешать таблицу

Введите вариант: 3

Введите ключ для поиска:5

Введите точность:0.1

5.000000 ............................$$$$$$$$$$$$$...................

ЗАКЛЮЧЕНИЕ

В данной работе я изучил работу с файлами, таблицами и списками. Изучил методы сортировки и поиска, и реализовал их на практике.

Эффективность программы можно увеличить, для повторного использования. Преобразовав таблицу после сортировки в бинарный файл. Таким образом при повторном использовании на не придется считывать данные из таблицы посимвольно, а можно считать сразу целиком структуру с отсортированной таблицей.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. РосДиплом, Оформление таблиц в дипломной работе, особенности и

требования ГОСТ / Электронный диплом / Режим доступа: <https://www.rosdiplom.ru/rd/pubdiplom/view.aspx?id=288>

1. Диплом Журнал, Оформление курсовой работы по ГОСТу

2019(образец) / Электронный диплом / Режим доступа: <https://journal.duplom.ru/kursovaya/oformlenie-kursov..>

1. Vyuchit.work – универсальная методичка / Электронный диплом / Режим доступа: <https://vyuchit.work/samorazvitie/sekretyi/oformlenie..>
2. Архив вопросов и ответов для программистов / Электронный диплом / Режим доступа: <https://qarchive.ru/320864_parametry_gcc__lm__lz__lrt..>
3. Компилятор GCC / Электронный диплом / Режим доступа: <http://parallel.uran.ru/book/export/html/25>
4. Керниган, Брайан У., Ритчи, Деннис М. Язык программирования С, 2-е издание. :Пер. с англ. – М. : Издательский дом «Вильямс», 2009. – 304 с. : ил. –