МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ»

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

ФАКУЛЬТЕТ №8 «ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ПРИКЛАДНАЯ МАТЕМАТИКА»

КАФЕДРА 806 «ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА И ПРОГРАММИРОВАНИЕ»

**Курсовой проект**

На тему

«Сортировка и поиск»

|  |  |
| --- | --- |
| Курс / Семестр: | 1 / 2 |
| Группа: | М8О-108Б-19 |
| ФИО студента: | Горохов М.А. |
| ФИО преподавателя: | Поповкин А.В. |
| Подпись: |  |
| Оценка: |  |
| Дата сдачи: |  |
| Дата проверки: |  |

Москва

2020

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc19304886)

[1. ЦЕЛИ ПРОЕКТА](#_Toc19304887) 4

[2. ПОСТАНОВКА ЗАДАЧИ 5](#_Toc19304890)

[3. СТРУКТУРА ПРОЕКТА 6](#_Toc19304894)

[4. ПРОГРАММНЫЙ КОД И ТЕСТОВЫЕ ДАННЫЕ 1](#_Toc19304894)1

[5. ЗАКЛЮЧЕНИЕ](#_Toc19304891) 18

[6. ИСПОЛЬЗОВАННЫЕ ИСТОЧНИКИ](#_Toc19304898) 19

# 

# ВВЕДЕНИЕ

В этом курсовом проекте я реализовал таблицу, хранящую элементы, содержащую ключ и строку, реализовал сортировку, перемешивание и поиск по таблице.

Реализация была на статических и динамичечких структурах, то есть с применением массивов, указателей, выделением и очищением памяти.

# ЦЕЛИ ПРОЕКТА

Изучить способы хранения и обработки таблицы в компьютере. Научиться некоторым алгоритмам сортировки, поиска и перемешивания.

# ПОСТАНОВКА ЗАДАЧИ

Мой ToDo list по данному курсовому проекту, который является структурой проекта:

7, 5

Сортировка Методом Шелла

Тип ключа вещественный, длина ключа 16 байт, хранение данных и ключа вместе

Todo list:

[+] 1. Подготовить тестовые данные

[+] 2. Реализовать структуру Item (ключ, данные) и структуру Table (размер, массив Item)

[+] 3. Реализовать базовые функции по работе со структурами: создание, добавление, изменение, удаление; печать только ключей, только данных или вместе

[+] 4. Реализовать перемешивание

[+] 5. Реализовать сортировку Шелла

[+] 6. Реализовать бинарный и линейный поиски

[+] 7. Реализовать программу-интерфейс по обработке таблицы

[+] 8. Раскидать проект по нескольким файлам для удобства пользователя

[+] 9. Подготовить отчет по курсовому проекту

# СТРУКТУРА ПРОЕКТА

Проект состоит из 7 файлов: main.c - главный файл, types.c - содержит переопределенны типы, item.c - содержит структуру item и функции по его обработке, table.c - содержит структуру table и функции по его обработке, sort\_search\_shuffle.c - содержит функции по сортировке, поиску и перемешиванию элементов (Item) таблицы, data1.txt и data2.txt - тестовые данные.

Пример работы программы:

[revammark@Raft kp9]$ date

Вт 12 мая 2020 18:43:06 MSK

[revammark@Raft kp9]$ ./main

1 filepath - Create table from file <filepath>

2 - print Keys

3 - print Data

4 - print size

5 - erase Table

6 shuffles - Shuffle <shuffles> times

7 - Shell sort Table

8 key accuracy - Binary Search(only for sorted Table)

9 key accuracy - Linear Search

0 - Exit programm

Action: 1 data1.txt

Table has been loaded

Action: 2

0.132300

0.232434

0.654655

1.453450

2.423535

5.546465

98765.424350

4567897654.987655

Action: 3

\_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_.---------------.

.'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_\_,` . \_\_\_\_ \_\_\_ \

|\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_:\ |`. \ \\_\_\_ \

\\'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_`.\_\_|""`. \ \\_\_\_ \

\\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_\_: \

\\'\\_\_\'\\_\_\'\\_\_\ \\_\_\'\\_;-----------------`

\\/ \/ \/ \/ \/ : hh|

\|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

Action: 4

Size 8

Action: 6 1

Action: 2

2.423535

0.232434

0.654655

1.453450

0.132300

5.546465

98765.424350

4567897654.987655

Action: 3

\\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_\_: \

.'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_\_,` . \_\_\_\_ \_\_\_ \

|\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_:\ |`. \ \\_\_\_ \

\\'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_`.\_\_|""`. \ \\_\_\_ \

\_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_.---------------.

\\'\\_\_\'\\_\_\'\\_\_\ \\_\_\'\\_;-----------------`

\\/ \/ \/ \/ \/ : hh|

\|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

Action: 6 666

Action: 2

0.232434

98765.424350

1.453450

0.132300

4567897654.987655

2.423535

0.654655

5.546465

Action: 3

.'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_\_,` . \_\_\_\_ \_\_\_ \

\\/ \/ \/ \/ \/ : hh|

\\'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_`.\_\_|""`. \ \\_\_\_ \

\_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_.---------------.

\|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

\\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_\_: \

|\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_:\ |`. \ \\_\_\_ \

\\'\\_\_\'\\_\_\'\\_\_\ \\_\_\'\\_;-----------------`

Action: 9 0.7 0.1

Key 0.654655 in Index: 6

Action: 8 0.7 0.1

Item haven't found

Action: 7

Action: 2

0.132300

0.232434

0.654655

1.453450

2.423535

5.546465

98765.424350

4567897654.987655

Action: 3

\_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_.---------------.

.'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_\_,` . \_\_\_\_ \_\_\_ \

|\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_:\ |`. \ \\_\_\_ \

\\'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_`.\_\_|""`. \ \\_\_\_ \

\\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_\_: \

\\'\\_\_\'\\_\_\'\\_\_\ \\_\_\'\\_;-----------------`

\\/ \/ \/ \/ \/ : hh|

\|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

Action: 9 0.7 0.1

Key 0.654655 in Index: 2

Action: 8 0.2 0.1

Key 0.232434 in Index: 1

Action: 8 100000 1000

Item haven't found

Action: 8 100000 2000

Key 98765.424350 in Index: 6

Action: 5

Erase table

Action: 4

Size 0

Action: 1

data2.txt

Table has been loaded

Action: 2

-345.345345

-123.235343

-34.987600

-12.876400

-3.098420

0.100234

12.098765

26.987600

4567.876540

45663365.235345

72435893832547576.765625

Action: 3

d888888b d888888b

d888 8888b d888888 888b

d88 88 898888b d8888 888 88b

d8P 88888888b d88888888888 b8b

98b 88888888888P 988888888 d8P

988 888 8888P \_=\_ 9888898 88 88P

9888 888888P q(-\_-)p 98888 888P

9888888P '\_) (\_` 9888888P

88 /\_\_/ \ 88

88 \_(<\_ / )\_ 88

d88b (\_\_\\_\\_|\_/\_\_) d88b

Action: 6 98763

Action: 2

72435893832547576.765625

26.987600

45663365.235345

-3.098420

12.098765

-12.876400

4567.876540

0.100234

-34.987600

-345.345345

-123.235343

Action: 3

d88b (\_\_\\_\\_|\_/\_\_) d88b

9888888P '\_) (\_` 9888888P

88 \_(<\_ / )\_ 88

98b 88888888888P 988888888 d8P

9888 888888P q(-\_-)p 98888 888P

d8P 88888888b d88888888888 b8b

88 /\_\_/ \ 88

988 888 8888P \_=\_ 9888898 88 88P

d88 88 898888b d8888 888 88b

d888888b d888888b

d888 8888b d888888 888b

Action: 8 0 100

Key -12.876400 in Index: 5

Action: 8 0 4

Item haven't found

Action: 9 0 4

Key -3.098420 in Index: 3

Action: 7

Action: 2

-345.345345

-123.235343

-34.987600

-12.876400

-3.098420

0.100234

12.098765

26.987600

4567.876540

45663365.235345

72435893832547576.765625

Action: 3

d888888b d888888b

d888 8888b d888888 888b

d88 88 898888b d8888 888 88b

d8P 88888888b d88888888888 b8b

98b 88888888888P 988888888 d8P

988 888 8888P \_=\_ 9888898 88 88P

9888 888888P q(-\_-)p 98888 888P

9888888P '\_) (\_` 9888888P

88 /\_\_/ \ 88

88 \_(<\_ / )\_ 88

d88b (\_\_\\_\\_|\_/\_\_) d88b

Action: 9 0 4

Key -3.098420 in Index: 4

Action: 8 0 4

Key 0.100234 in Index: 5

Action: 5

Erase table

Action: 0

# ПРОГРАММНЫЙ КОД И ТЕСТОВЫЕ ДАННЫЕ

Тестовый файл ***data1.txt***:

8

0.1323

\_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_.---------------.

0.232434

.'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_\_,` . \_\_\_\_ \_\_\_ \

0.6546546

|\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_:\ |`. \ \\_\_\_ \

1.45345

\\'\\_\_\'\\_\_\'\\_\_\'\\_\_\'\\_`.\_\_|""`. \ \\_\_\_ \

2.4235345345

\\/ \_\_\/ \_\_\/ \_\_\/ \_\_\/ \_\_: \

5.54646456

\\'\\_\_\'\\_\_\'\\_\_\ \\_\_\'\\_;-----------------`

98765.42435

\\/ \/ \/ \/ \/ : hh|

4567897654.987654678

\|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|

Тестовый файл ***data2.txt***:

11

-345.345345

d888888b d888888b

-123.2353426346

d888 8888b d888888 888b

-34.9876

d88 88 898888b d8888 888 88b

-12.8764

d8P 88888888b d88888888888 b8b

-3.09842

98b 88888888888P 988888888 d8P

0.100234234

988 888 8888P \_=\_ 9888898 88 88P

12.09876545

9888 888888P q(-\_-)p 98888 888P

26.9876

9888888P '\_) (\_` 9888888P

4567.87654

88 /\_\_/ \ 88

45663365.23534534543

88 \_(<\_ / )\_ 88

72435893832547576.763435342434834634636546546735

d88b (\_\_\\_\\_|\_/\_\_) d88b

Код программы ***main.c***:

#include <stdio.h>

#include <stdlib.h>

#include "types.c"

#include "item.c"

#include "table.c"

#include "sort\_search\_shuffle.c"

int main() {

printf("1 filepath - Create table from file <filepath>\n");

printf("2 - print Keys\n");

printf("3 - print Data\n");

printf("4 - print size\n");

printf("5 - erase Table\n");

printf("6 shuffles - Shuffle <shuffles> times\n");

printf("7 - Shell sort Table\n");

printf("8 key accuracy - Binary Search(only for sorted Table)\n");

printf("9 key accuracy - Linear Search\n");

printf("0 - Exit programm\n\n");

printf("Action: ");

short action;

char filepath[255];

Table table;

do {

// Я не понимаю как, но scanf заменяет size на 0 (??!!)

int size = table.size;

scanf("%d", &action);

if(table.size == 0) table.size = size;

// Конец какой-то магии

// Теперь локальная переменная size берет на себя удар.

// Я долго разбирался с этим багом. Я вставил костыль, но остался в недоумении

if(action == 0) break;

unsigned int shuffles;

int index;

float128 key, accuracy;

switch (action) {

case 1:

scanf("%s", filepath);;

createTable(&table, filepath);

break;

case 2:

printKeysTable(&table);

break;

case 3:

printDataTable(&table);

break;

case 4:

printf("Size %i\n", table.size);

break;

case 5:

printf("Erase table\n");

eraseTable(&table);

break;

case 6:

scanf("%u", &shuffles);

shuffleTable(&table, shuffles);

break;

case 7:

shellSortTable(&table);

break;

case 8:

scanf("%Lf", &key);

scanf("%Lf", &accuracy);

index = binarySearch(&table, key, accuracy);

if(index != -1)

printf("Key %LF in Index: %i\n", table.items[index]->key, index);

else

printf("Item haven't found\n");

break;

case 9:

scanf("%Lf", &key);

scanf("%Lf", &accuracy);

index = linearSearch(&table, key, accuracy);

if(index != -1)

printf("Key %LF in Index: %i\n", table.items[index]->key, index);

else

printf("Item haven't found\n");

break;

case 0:

break;

default:

printf("Getted: %d\nIt is strange..\n", action);

break;

}

printf("Action: ");

} while(1);

return 0;

}

Код библиотеки ***types.c***:

#ifndef \_\_types\_c\_\_

#define \_\_types\_c\_\_

typedef long double float128;

#endif

Код библиотеки ***item.c***:

#ifndef \_\_item\_c\_\_

#define \_\_item\_c\_\_

typedef struct {

float128 key;

char\* data;

} Item;

Item\* makeItem(float128 key, char\* string) {

Item\* item = (Item\*)malloc(sizeof(Item));

item->key = key;

item->data = string;

return item;

}

void destroyItem(Item\* item) {

free(item->data);

free(item);

}

#endif

Код библиотеки ***table.c***:

#ifndef \_\_table\_c\_\_

#define \_\_table\_c\_\_

#include <stdio.h>

#include <stdlib.h>

#include "types.c"

#include "item.c"

typedef struct {

unsigned int size;

// Массив указателей

Item\*\* items;

} Table;

void eraseTable(Table\* table) {

if(table->size != 0) {

for(int i=0; i<table->size; i++) {

destroyItem(table->items[i]);

}

table->size = 0;

}

}

void createTable(Table\* table, char\* filepath) {

FILE\* inputfile;

inputfile = fopen(filepath, "r");

int size;

fscanf(inputfile, "%i", &size);

table->size = size;

table->items = malloc(sizeof(Item)\*size);

// Заполняю массив указателей указателями на Item

for(int i=0; i < size; i++) {

float128 key;

char\* inputString = NULL;

fscanf(inputfile, "%Lf", &key);

// Ниже считываю посимвольно строку до конца строки

char buffer;

unsigned long size = 0;

buffer = fgetc(inputfile);

if (buffer == EOF) break;

do {

buffer = fgetc(inputfile);

if(buffer == '\n' || buffer == EOF) break;

size++;

if (inputString == NULL)

inputString = (char \*) malloc(sizeof(char));

else

inputString = (char \*) realloc(inputString, size \* sizeof(char));

inputString[size - 1] = buffer;

} while (buffer != '\n' && buffer != EOF);

inputString = realloc(inputString, sizeof(char)\*size);

// Считал строку

table->items[i] = makeItem(key, inputString);

}

printf("Table has been loaded\n");

return;

}

void printKeysTable(Table\* table) {

for(int i=0; i < table->size; i++) {

// Обращаюсь к таблице к i элементу массива типа Item\* к его ключу

printf("%Lf\n", table->items[i]->key);

}

}

void printDataTable(Table\* table) {

for(int i=0; i < table->size; i++) {

// Обращаюсь к таблице к i элементу массива типа Item\* к его строке

printf("%s\n", table->items[i]->data);

}

}

#endif

Код библиотеки ***sort\_search\_shuffle.c***:

#ifndef \_\_sort\_search\_shuffle\_c\_\_

#define \_\_sort\_search\_shuffle\_c\_\_

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include "types.c"

#include "item.c"

#include "table.c"

void shuffleTable(Table\* table, unsigned long shuffles) {

time\_t t;

time(&t);

srand(t + clock());

unsigned int randnum1, randnum2;

for(unsigned long i=0; i < shuffles; i++) {

randnum1 = (unsigned)(rand()\*rand())%table->size;

randnum2 = (unsigned)(rand()\*rand())%table->size;

Item\* tmp = table->items[randnum1];

table->items[randnum1] = table->items[randnum2];

table->items[randnum2] = tmp;

}

}

// Сортировка Шелла - улучшенная сортировка вставками

// Сортируем сначала элементы на расстоянии step

// С каждой новой итерацией уменьшаем step в 2 раза

void shellSortTable(Table\* table) {

// Проверка отсортированности Таблицы

short isSorted = 1, isReverseSorted = 1;

for(int i=1; i < table->size; i++) {

if(table->items[i-1] > table->items[i]) {

isSorted = 0;

break;

}

}

for(int i=1; i < table->size; i++) {

if(table->items[i-1] < table->items[i]) {

isReverseSorted = 0;

break;

}

}

if(isSorted) return;

if(isReverseSorted) {

Item\* tmp;

for(int i=0; i < table->size/2; i++) {

tmp = table->items[i];

table->items[i] = table->items[table->size-i-1];

table->items[table->size-i-1] = tmp;

}

return;

}

unsigned int step;

Item\* tmp;

for(step = table->size/2; step > 0; step /= 2) {

for(int i=step; i < table->size; i++) {

for(int j=i-step; j>=0 && (table->items[j]->key > table->items[j+step]->key); j -= step) {

tmp = table->items[j];

table->items[j] = table->items[j+step];

table->items[j+step] = tmp;

}

}

}

}

int linearSearch(Table\* table, float128 key, float128 accuracy) {

for(int i=0; i < table->size; i++) {

if(table->items[i]->key - accuracy <= key && key <= table->items[i]->key + accuracy) {

return i;

}

}

}

int binarySearch(Table\* table, float128 key, float128 accuracy) {

int low = 0;

int high = table->size-1;

int mid;

while(low <= high) {

mid = (high+low)/2;

if(key < table->items[mid]->key - accuracy) {

high = mid-1;

}

else if(key > table->items[mid]->key + accuracy) {

low = mid+1;

}

else if(table->items[mid]->key - accuracy <= key && key <= table->items[mid]->key + accuracy) {

return mid;

}

}

return -1;

}

#endif

# ЗАКЛЮЧЕНИЕ

Я составил таблицу в СП Си. Реализовал программу, работающую с этой таблицей. Реализовал библиотеку по поиску элемента в таблице (и бинарный поиск, и линейный), а также другие вспомогательные библиотеки

Реализованная программа умеет: считывать таблицу из файла, выводить ключи элементов, выводить значения элементов (строки), выводить количество элементов в таблице, очищать таблицу, перемешивать элементы в таблице n число раз, сортировать элменты в таблице методом Shell’а, искать элемент по ключу бинарным поиском, линейным поиском и успешно оканчивать работы программы.

Я полность выполнил поставленное задание

Я освоил работу со структурами в Си, с массивами, указателями и управлением памяти в СИ.

# ИСПОЛЬЗОВАННЫЕ ИСТОЧНИКИ

1. Керниган Б., Ритчи Д. Язык программирования Си = The C programming language. — 2-е изд. — М.: Вильямс, 2007. — С. 304. — ISBN 0-13-110362-8.
2. Эндрю Таненбаум*, Structured Computer Organization,* [ISBN 0-13-148521-0](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/0131485210)
3. В.Е. Зайцев, Конспект Лекций по курсу “Фундаментальная Информатика и Языки и методы программирования”.