МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ»)

Факультет информационных технологий и компьютерной безопасности

Кафедра Систем управления и информационных технологий в строительстве

КУРСОВОЙ ПРОЕКТ

по дисциплине Основы программирования и алгоритмизации

Тема: Разработка программы для работы с файловой базой данных «IT-компании».

Расчетно-пояснительная записка

Разработал студент М. Р. Черниченко

Подпись, дата Инициалы, фамилия

Руководитель О.В. Минакова

Подпись, дата Инициалы, фамилия

Нормоконтроллер О.В. Минакова

Подпись, дата Инициалы, фамилия

Защищена Оценка

дата

2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРТСВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ»)

Кафедра Систем управления и информационных технологий в строительстве

ЗАДАНИЕ

на курсовой проект

по дисциплине Основы программирования и алгоритмизации

Тема: Разработка программы для работы с файловой базой данных «IT-компании».

Студент группы бИСТ-232 Черниченко Максим Русланович

Фамилия, имя, отчество

База данных «IT-компании», Признак поиска: Название курса, Вариант сортировки: По рейтингу (возрастание)

Технические условия Windows 11, MicrosoftVisualStudio2022, язык программирования C

Содержание и объем проекта (графические работы, расчеты и прочее):  
 стр, рисунков, таб, приложение

Сроки выполнения этапов анализ и постановка задачи (10.9-5,10.21); разработка пошаговой детализации программы (6.10 -11.11.21); реализация программы (11.11-5,12.21); тестирование программы (6.12-11.12.21); оформление пояснительной записки (11.12-14.12.21).

Срок защиты курсового проекта

Руководитель О.В. Минакова

Подпись, дата Инициалы, фамилия

Задание принял студент М. Р. Черниченко

Подпись, дата Инициалы, фамилия

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 5](#_Toc154460062)

[**ОСНОВНАЯ ЧАСТЬ** 6](#_Toc154460063)

[**1. Постановка задач** 6](#_Toc154460064)

[**2. Пошаговая детализация решения** 9](#_Toc154460065)

[**3. Тестирование и отладка** 16](#_Toc154460066)

[**ЗАКЛЮЧЕНИЕ** 20](#_Toc154460067)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 21](#_Toc154460068)

[**ПРИЛОЖЕНИЕ** 22](#_Toc154460069)

# **ВВЕДЕНИЕ**

Цель настоящей работы заключается в разработке программы, предоставляющей доступ к файловой базе данных "IT-компании", осуществляющей заданный функционал. Для достижения этой цели предстоит решить несколько ключевых задач:

1. Выбор структуры данных и формата файла. Необходимо обосновать выбор конкретной структуры данных для хранения записей в файле, а также определить оптимальный формат файловой базы данных.
2. Реализация пользовательского интерфейса. Требуется создание интуитивно понятного интерфейса для взаимодействия пользователя с программой. Этот интерфейс должен предоставлять полный набор функций и оставаться активным до момента, когда пользователь решит завершить работу.
3. Обеспечение основных функций: Программа должна гарантировать возможность создания, добавления, сортировки и поиска записей по заданным критериям, обеспечивая тем самым полный цикл управления информацией в файловой базе данных.

Реализация этих задач требует не только точного понимания структуры и функций базы данных, но и эффективной программной реализации, что станет ключевым аспектом данного курсового проекта

Программа разрабатывается в виде консольного приложения на языке Си в среде Microsoft Visual Studio 2022.

### **1. Постановка задач**

Для создания файловой базы данных необходимо разработать структуру для хранения информации о компаниях, включая поля для имени компании, её описания, года основания, количества сотрудников, рыночной капитализации и рейтинга.

Для работы с файловой базой данных нужно реализовать следующие функции:

1. Добавление компании в файл;
2. сохранение введенной информации о компании пользователем в файл;
3. загрузка данных из файла с последующим выводом в консоль;
4. сортировка данных в файле по нескольким критериям;
5. поиск данных в файле по нескольким критериям.

Хранение информации осуществляется в файле с расширением «.txt». В данной программе используется указатель на массив структур. Это решение стало оптимальным в контексте языка программирования C, так как в нем отсутствуют встроенные динамические массивы, а база данных предполагает постоянное изменение количества записей.

Использование указателя на массив структур обеспечивает возможность изменения выделенной памяти для него с помощью функций «malloc» и «realloc». Это дает гибкость в управлении памятью и количеством записей, что является альтернативой использованию динамических массивов в данном контексте.

В программе должен быть реализован понятный пользовательский интерфейс программы, которое должно представлять собой интуитивно понятное меню, предоставляющее доступ к основным функциям работы с файловой базой данных «IT-компании». Принцип работы меню должен быть следующий:

* Пользователь выбирает нужную функцию, указав соответствующую цифру из меню;
* После выполнения выбранной операции, пользователь возвращается к меню для возможности выбора следующего действия или завершения работы.

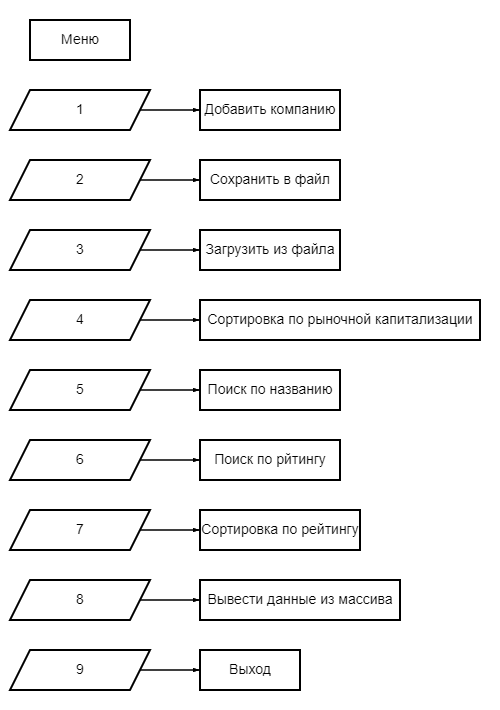
Схема работы пользовательского интерфейса (меню) представлена на рисунке 1.

Рисунок 1 – Схема работы меню

Контрольный пример заполненной структуры показан в таблице 1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Описание | Год основания | Количество сотрудников | Рейтинг | Рыночная капитализация |
| Intel | Отличная | 1990 | 5000 | 80 | 10 |
| Nvidia | Прогрессирующая | 1995 | 2000 | 70 | 50 |
| Ростелеком | Стабильная | 2010 | 7000 | 40 | 10 |
| Softline | Плохая | 2005 | 1000 | 5 | 5 |

### **2. Пошаговая детализация решения**

Разрабатываемая программа должна предоставить пользователю возможность ввести данные, потом предложить ему выбор определенных действий, которые будут производить работу с предметной областью.

Для использования встроенных функций подключены заголовочный файлы с помощью директивы #include:

* <stdio.h> - содержит стандартные функции, которые реализуют основные возможности ввода и вывода;
* <string.h> - содержит стандартные функции необходимые для работы с различными видами строк;
* <stdlib.h> - содержит вспомогательные функции, которые могут быть использованы как для ввода и вывода, так и для других задач;
* <locale.h> - используется для задач, связанных с локализацией программы.

При помощи директивы препроцессора #define объявлено \_CRT\_SECURE\_NO\_DEPRECATE, что позволит пользоваться функциями ввода.

Для создания массива структур необходимо реализовать структурный тип comp\_t. В который будут входить такие поля как:

char name[100] – символьный массив, в котором хранится название компании;

char description[500] – символьный массив, в котором хранится описание компании;

int foundationYear – целочисленная переменная, в которой хранится год основания компании;

int numEmployees – целочисленная переменная, в которой хранится количество сотрудников компании;

double markertCap – целочисленная переменная, в которой хранится рыночная капитализация компании;

int rating – целочисленная переменная, в которой хранится рейтинг компании.

Программа, написанная на языке программирования Си должна содержать основную функцию main(), с данной функции должна начинаться программа. Из нее вызываются другие функции программы, по завершению работы функции программа возвращается в main(), в то место откуда функция была вызвана.

В начале программа вызывает функцию главного меню, в которой вызываются функции, отвечающие за указанные в меню действия.

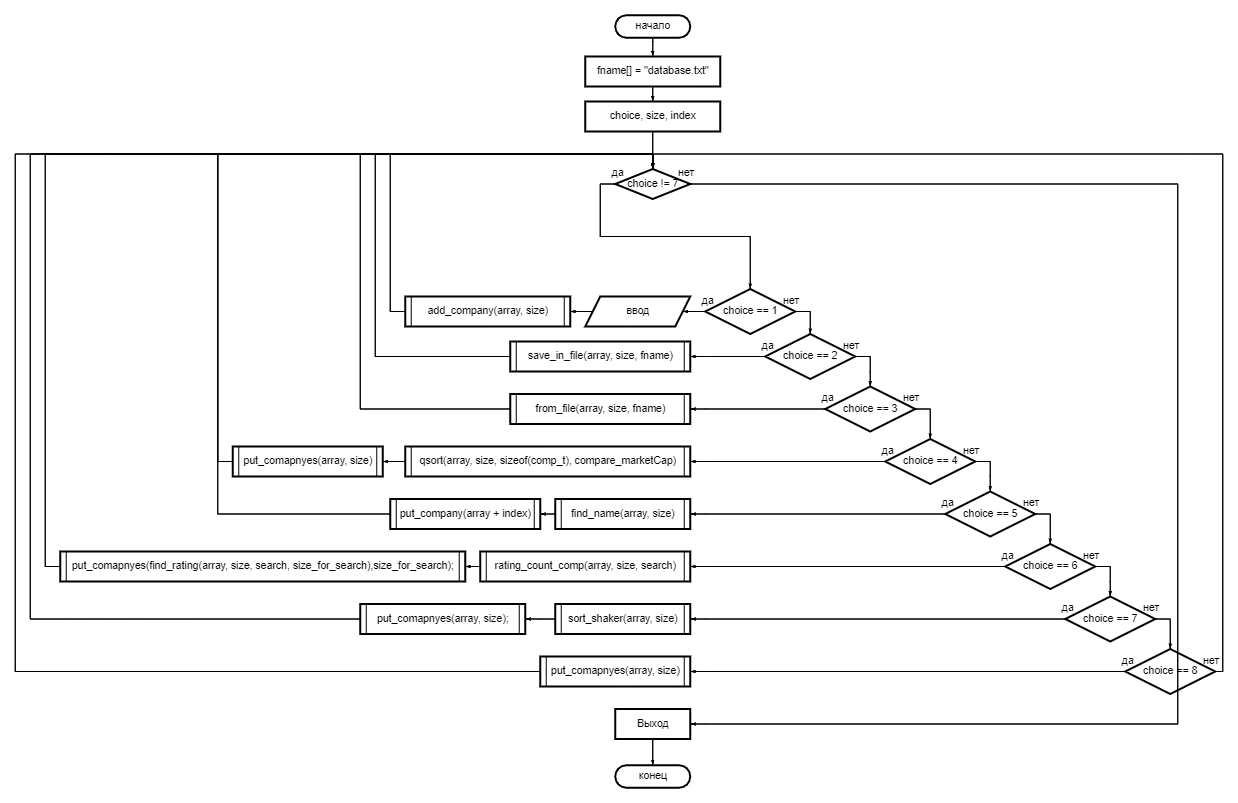
Многоальтернативный выбор, реализован с помощью оператора switch. Блок-схема данной функции (Рис. 2).

Рисунок 2 – Блок-схема функции main ()

Рисунок 1 – Блок-схема функции main()

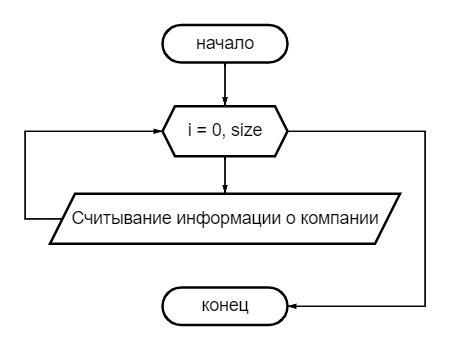
Функция int add\_company(comp\_t\* array, int size) предназначена для добавления компании, принимает массив структур, адресуемый параметром comp\_t, размером size структур. Функция необходима для ручного ввода полей структур (Рис. 3).

Рисунок 3 – Блок-схема функции добавления компании

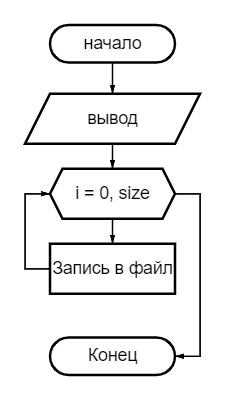
Функция int save\_in\_file(comp\_t\* array, int size, char\* file\_name) предназначена для сохранения в файл, принимает указатель на массив структур comp\_t, размером size структур и имя файла задающееся параметром file\_name. Функция нужна для сохранения в файл (Рис. 4).

Рисунок 4 – Блок-схема функции сохранения в файл

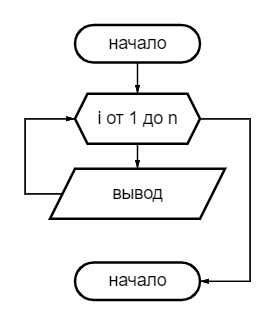
Функция int from\_file(comp\_t\* array, int size, char\* file\_name) предназначена для вывода в консоль массива структур адресуемым параметром array, размером size и названием файла (Рис. 5).

Рисунок 5 – Блок-схема функции загрузки из файла

Функция int compare\_marketCap(const void\* av, const void\* bv) используется в сравнении для qsort(), принимает параметр av и bv (Рис. 6). Возвращаемые значения:

* если, av больше bv, то -1;
* если, av меньше bv, то 1;
* если, av равен bv, то 0.

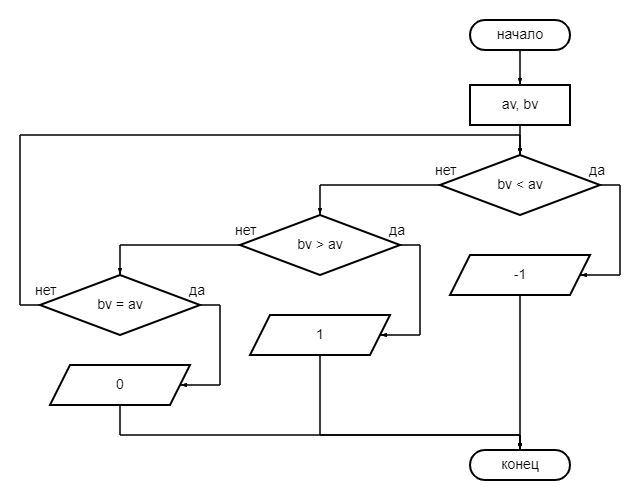
После сортировки массива структур, его необходимо вывести в консоли, за это отвечает функция int put\_companyes(comp\_t\* array, int size) предназначенная для вывода всего массива структур адресуемым параметром array, размером size.

Рисунок 6 – Блок-схема функции сравнения для qsort()

Функция int find\_name(comp\_t\* array, int size) предназначена для поиска компании по названию, принимает указатель на массив структур comp\_t, размером size и возвращает индекс найденного элемента в соответствии с введенным признаком поиска (Рис. 7).

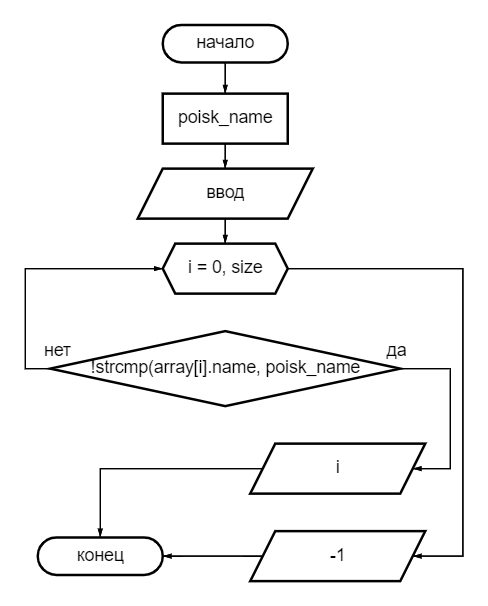
После нахождения необходимого названия, его нужно вывести в консоли, для этого используется функция void put\_company(comp\_t\* array) используемая для вывода в консоль одной структуры из массива array.

Рисунок 7 – Блок-схема функции поиска по названию

Функция int rating\_count\_comp(comp\_t\* array, int size,int search) предназначена для подсчета количества структур с полем рейтинг совпадающим с параметром search, которые хранятся в массиве структур адресуемым параметром array, размером size (Рис. 8).

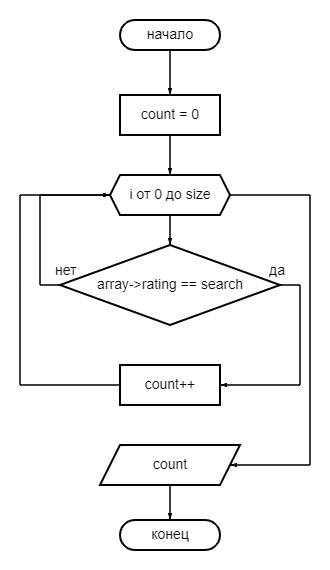


Рисунок 8 – Блок-схема для подсчета количества структур с полем рейтинг

comp\_t\* find\_rating(comp\_t\* array, int size, int search,int count) – функция для поиска по рейтингу, принимает указатель на массив структур comp\_t, размером size структур и возвращает новый массив структур размером count (Рис. 9). Параметр count задаётся функцией rating\_count\_comp.

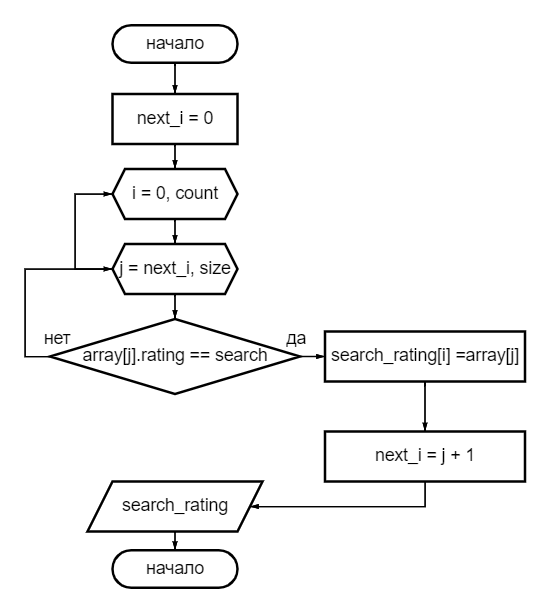
Функция void sort\_shaker(comp\_t\* array, int n) предназначена для сортировки массива структур методом шейкерной сортировки по полю rating (рейтинг). Принимает указатель на массив структур и размер массива (Рис.10)

Рисунок 9 – Блок-схема функции поиска по рейтингу

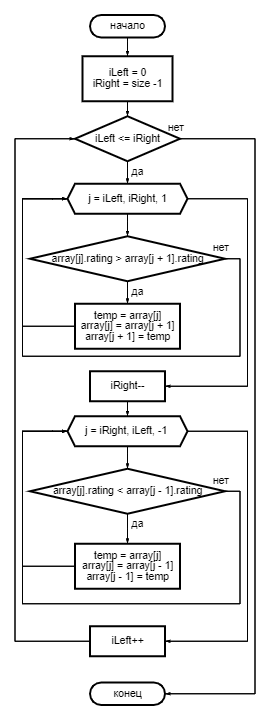


Рисунок 10 – Блок-схема шейкерной сортировки

### **3. Тестирование и отладка**

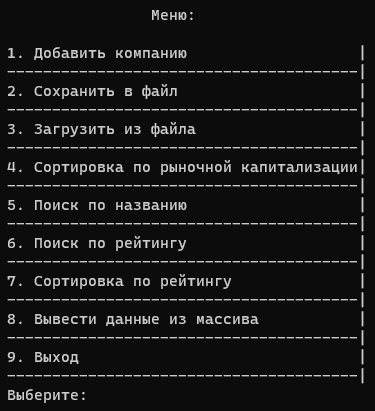
При запуске программы перед пользователем появляется меняю для взаимодействия с базой данных (Рис. 11).

Рисунок 11 – Главное меню

После выбора пункта 1 «Добавить компанию», пользователю будет предложено ввести количество записей и данные о компании (Рис. 12).

После добавления записи, перед пользователем вновь откроется меню. предположим пользователь захотел сохранить данные в файл, для этого он выберет пункт 2 «Сохранить в файл». Данные будут сохранены в файл

После добавления записи, перед пользователем вновь откроется меню. предположим пользователь захотел сохранить данные в файл, для этого он выберет пункт 2 «Сохранить в файл». Данные будут сохранены в файл

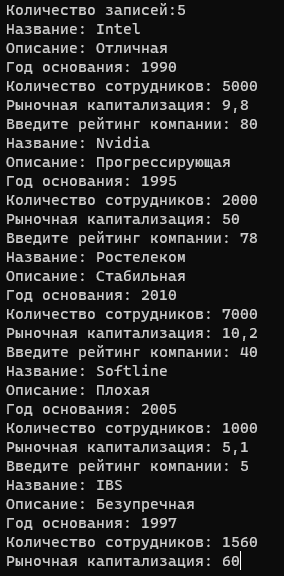
После добавления записи, перед пользователем вновь откроется меню. предположим пользователь захотел сохранить данные в файл, для этого он выберет пункт 2 «Сохранить в файл». Данные будут сохранены в файл.

Рисунок 12 – Пример заполнения базы данных

Далее для того, чтобы совершать какие-либо операции над базой данных, её необходимо загрузить из файла, для этого пользователю необходимо будет выбрать пункт 3 «Загрузить из файла». Данные будут загружены в файл.

Предположим, что пользователь захотел просмотреть данные, которые были записаны в файл, для этого действия он может воспользоваться пунктом 8 «Вывести данные из массива» (Рис. 13).

После пользователю необходимо отсортировать данные по рыночной капитализации (по убыванию), для этого он выберет пункт 4 «Сортировка по рыночной капитализации» (Рис. 14).

Рисунок 14 – Пример сортировки по рыночной капитализации

Рисунок 13 - Пример чтения данных

Если же пользователю потребуется отсортировать данные по рейтингу, он может воспользоваться пунктом 7 «Сортировка по рейтингу» (Рис. 15).

Рисунок 15 – Пример сортировки по рейтингу

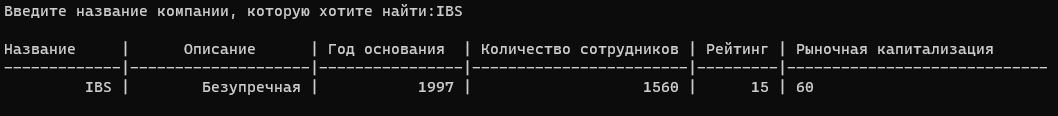
Далее пользователю необходимо найти компанию по названию, для этого он воспользуется пунктом 5 «Поиск по названию» (Рис. 16).

Рисунок 16 – Поиск компании по названию

А если пользователю потребуется найти компанию по рейтингу, в этом ему поможет пункт 6 «Поиск по рейтингу» (Рис. 17).

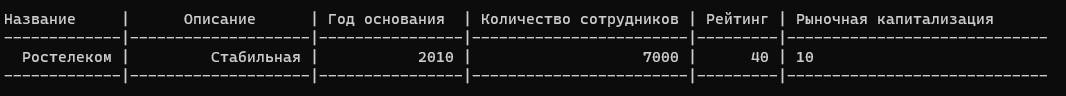
Для того чтобы выйти из программы пользователь может воспользоваться пунктом 8 «Выход».

Рисунок 17 – Поиск компании по рейтингу

# **ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсового проекта на языке программирования Си с использованием среды Microsoft Visual Studio 2022 была разработана структура для заполнения предметной области – «IT-компании».

При решении поставленной задачи были выполнены следующие действия:

* Составлен алгоритм решения;
* Реализована работа со структурами;
* Созданы и описаны пользовательские функции;
* Реализован диалог с использованием;
* Проведена работа с научно-технической литературой.

Отсюда следует, что пользователь с помощью структур может работать с данными в базе данных, выполняя различные действия с информацией, представленной в данной предметной области.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Курипта О.В. Основы программирования и алгоритмизации: практикум / О.В.Курипта, О.В. Минакова, Д.К. Проскурин; Воронехский ГАСУ. – Воронех, 2015.
2. Подбельский В.В., Фомин С.С. Программирование на языке Си. М.: ФиС.
3. Неземский В.И. Процедуры и функции [Электронный ресурс]: методические указания/ Неземский В.И., Орешкина О.А.— Электрон. текстовые данные. — М.: Московский государственный технический университет имени Н.Э. Баумана, 2009.
4. Бахенова И.Ю. Введение в программирование [Электронный ресурс]: учебное пособие/ Бахенова И.Ю., Сухомлин В.А.— Электрон. Текстовые данные. — М.: БИНОМ. Лаборатория знаний, Интернет- Университет Информационных Технологий (ИНТУИТ), 2007.
5. Шишкин А.Д. Программирование на языке Си [Электронный ре-сурс]: учебное пособие/ Шишкин А.Д.— Электрон. текстовые данные. — СПб.: Российский государственный гидрометеорологический университет, 2003
6. Далека В.Д., Деревянко А.С., Кравец О.Г., Тимановская Л.Е. Модели и структуры данных [Текст] / Харьков: ХГПУ, 2000. – 241 с.
7. Структуры [электронный ресурс] / http://www.c-cpp.ru/ – сайт по программированию на С и С++ // http://www.ccpp.ru/books/struktury – статья о структурах.
8. В.В. Подбельский, С.С.Фомин. Программирование на языке Си [Текст] / Москва, «Финансы и статистика», 2003. – 590с
9. Керниган Б., Ритчи Д. Язык программирования Си [Текст] / Пер. с англ., 3е изд., испр. – СПб.: «Невский Диалект», 2001. – 352 с
10. Солдатенко И.С. Основы программирования на языке Си: [Текст] учеб. пособие. Тверь: Твер. гос. ун-т, 2017. – 159 с.

# ПРИЛОЖЕНИЕ

#define \_CRT\_SECURE\_NO\_DEPRECATE

#include <locale.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct Company {

char name[100];

char description[500];

int foundationYear;

int numEmployees;

double marketCap;

int rating;

};

typedef struct Company comp\_t;

int add\_company(comp\_t\*, int);

int save\_in\_file(comp\_t\*, int, char\*);

int from\_file(comp\_t\*, int, char\*);

int find\_name(comp\_t\*, int);

void put\_company(comp\_t\*);

int compare\_marketCap(const void\*, const void\*);

int put\_comapnyes(comp\_t\*, int);

comp\_t\* find\_rating(comp\_t\*, int, int,int);

int rating\_count\_comp(comp\_t\*, int, int);

void sort\_shaker(comp\_t\*, int);

void put\_name\_struct();

int main() {

setlocale(LC\_ALL, "RUS");

system("chcp 1251");

char fname[] = "database.txt";

FILE\* file;

if ((file = fopen(fname, "rb")) == NULL) {

printf("Ошибка открытия файла для записи");

return 0;

}

int choice;

comp\_t\* array = 0;

int size;

fscanf(file, "%d\n", &size);

fclose(file);

array = malloc(sizeof(comp\_t) \* size);

int index, flag = 0;

do {

printf("\n\t\tМеню:\n\n");

printf("1. Добавить компанию |\n");

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

printf("2. Сохранить в файл |\n");

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

printf("3. Загрузить из файла |\n");

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

printf("4. Сортировка по рыночной капитализации|\n");

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

printf("5. Поиск по названию |\n");

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

printf("6. Поиск по рейтингу |\n");

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

printf("7. Сортировка по рейтингу |\n");

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

printf("8. Чтение записанных данных |\n");

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

printf("9. Выход |\n");

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

printf("Выберите: ");

scanf("%d", &choice);

switch (choice) {

case 1: {

printf("Количество записей:");

scanf("%d", &size);

array = malloc(sizeof(comp\_t) \* size);

add\_company(array, size);

flag++;

}break;

case 2: {

save\_in\_file(array, size, fname);

}break;

case 3: {

from\_file(array, size, fname);

flag++;

}break;

case 4: {

qsort(array, size, sizeof(comp\_t), compare\_marketCap);

put\_comapnyes(array, size);

}break;

case 5: {

index = find\_name(array, size);

if (index == -1) {

printf("Данного названия нет в базе данных");

break;

}

put\_name\_struct();

put\_company(array + index);

}break;

case 6: {

int size\_for\_search, search;

printf("Введите рейтинг компании, которую хотите найти: ");

scanf("%d", &search);

size\_for\_search = rating\_count\_comp(array, size, search);

if (size\_for\_search == 0) {

printf("Нет компаний с таким рейтингом\n");

break;

}

put\_comapnyes(find\_rating(array, size, search, size\_for\_search),size\_for\_search);

}break;

case 7: {

sort\_shaker(array, size);

put\_comapnyes(array, size);

}break;

case 8: {

if (!flag) {

printf("Для начала заполните массив данными");

break;

}

put\_comapnyes(array, size);

}break;

case 9:

printf("Выход...\n");

};

} while (choice != 9);

}

int add\_company(comp\_t\* array, int size) {

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

printf("Название: ");

scanf("%s", array->name);

printf("Описание: ");

scanf("%s", array->description);

printf("Год основания: ");

scanf("%d", &array->foundationYear);

printf("Количество сотрудников: ");

scanf("%d", &array->numEmployees);

printf("Рыночная капитализация: ");

scanf("%lf", &array->marketCap);

printf("Введите рейтинг компании: ");

scanf("%d", &array->rating);

array++;

}

}

int save\_in\_file(comp\_t\* array, int size, char\* file\_name) {

FILE\* file = fopen(file\_name, "wb");

if (file == NULL) {

printf("Ошибка открытия файла для записи\n");

return;

}

fprintf(file,"%d\n", size);

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

fprintf(file, "%s %s %d %d %d %lf \n", array->name, array->description, array->foundationYear, array->numEmployees, array->rating, array->marketCap);

array++;

}

fclose(file);

printf("Данные сохранены в файл.\n");

}

int from\_file(comp\_t\* array, int size, char\* file\_name) {

FILE\* file = fopen(file\_name, "rb");

if (file == NULL) {

printf("Ошибка открытия файла для записи\n");

return;

}

fscanf(file,"%d\n", &size);

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

fscanf(file, "%s %s %d %d %d %lf\n", array->name, array->description, &array->foundationYear, &array->numEmployees, &array->rating, &array->marketCap);

array++;

}

fclose(file);

printf("Данные загружены из файла.\n");

}

void put\_company(comp\_t\* array) {

fprintf(stdout, "%12s | %18s | %14d | %22d | %7d | %2.lf \n", array->name, array->description, array->foundationYear, array->numEmployees, array->rating, array->marketCap);

}

int compare\_marketCap(const void\* av, const void\* bv) {

const comp\_t\* b = av, \* a = bv;

if (a->marketCap < b->marketCap) return -1;

if (a->marketCap > b->marketCap) return 1;

return 0;

}

int find\_name(comp\_t\* array, int size) {

char poisk\_name[30];

printf("Введите название компании, которую хотите найти:");

scanf("%s", poisk\_name);

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

if (!strcmp(array[i].name, poisk\_name)) {

return i;

}

}

return -1;

}

comp\_t\* find\_rating(comp\_t\* array, int size, int search,int count) {

comp\_t\* search\_rating;

int next\_i = 0;

search\_rating = malloc(sizeof(comp\_t) \* count);

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

for (int j = next\_i; j < size; j++) {

if (array[j].rating == search) {

search\_rating[i] = array[j];

next\_i = j + 1;

break;

}

}

}

return search\_rating;

}

int rating\_count\_comp(comp\_t\* array, int size,int search) {

int count = 0;

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

if (array->rating == search) {

count++;

}

array++;

}

return count;

}

int put\_comapnyes(comp\_t\* array, int size) {

printf("\nНазвание | Описание | Год основания | Количество сотрудников | Рейтинг | Рыночная капитализация \n");

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

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

put\_company(array);

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

array++;

}

return size;

}

void sort\_shaker(comp\_t\* array, int size) {

int iLeft = 0;

int iRight = size - 1;

comp\_t temp;

while (iLeft <= iRight) {

for (int j = iLeft; j < iRight; j++) {

if (array[j].rating > array[j + 1].rating) {

temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

}

}

iRight--;

for (int j = iRight; j > iLeft; j--) {

if (array[j].rating < array[j - 1].rating) {

temp = array[j];

array[j] = array[j - 1];

array[j - 1] = temp;

}

}

iLeft++;

}

}

void put\_name\_struct() {

printf("\nНазвание | Описание | Год основания | Количество сотрудников | Рейтинг | Рыночная капитализация \n");

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

}