Московский авиационный институт

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

КУРСОВОЙ ПРОЕКТ

по курсу “Практикум на ЭВМ”

II семестр

Тема “Сортировка и поиск”

Работу выполнила:

Студентка группы М8О-106Б-21

Деревянко Екатерина Андреевна

Работу проверил:

Старший преподаватель

Дубинин Алексей Владимирович

Москва 2022

**Оглавление**

[Введение 3](#_Toc32618)

[Теоретическая часть 4](#_Toc29037)

[Сортировка простой вставкой 4](#_Toc1727)

[Практическая часть 6](#_Toc17131)

[Об алгоритме 6](#_Toc8046)

[Протокол 6](#_Toc1661)

[Заключение 12](#_Toc3194)

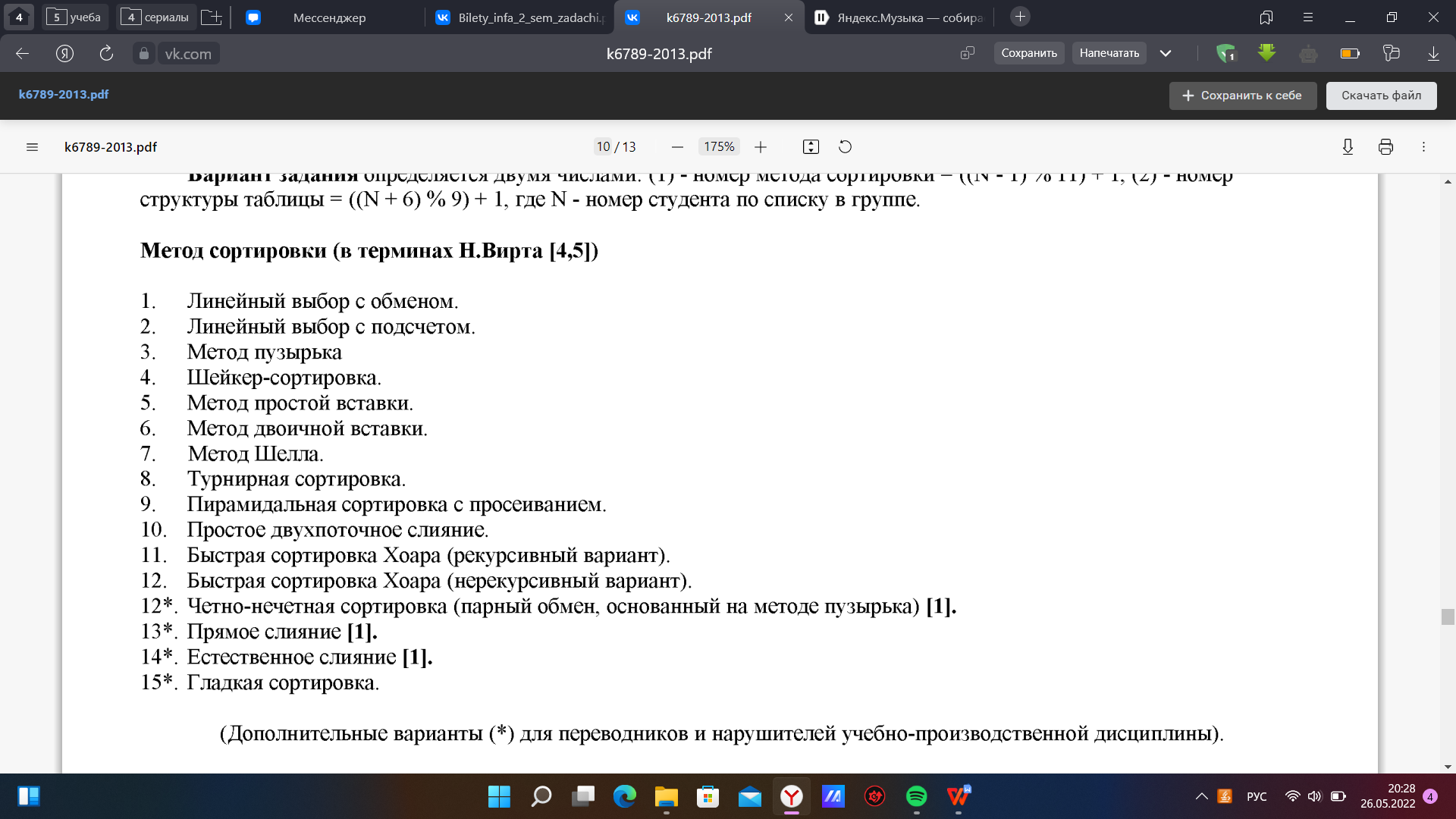
[Список источников 13](#_Toc30262)

# Введение

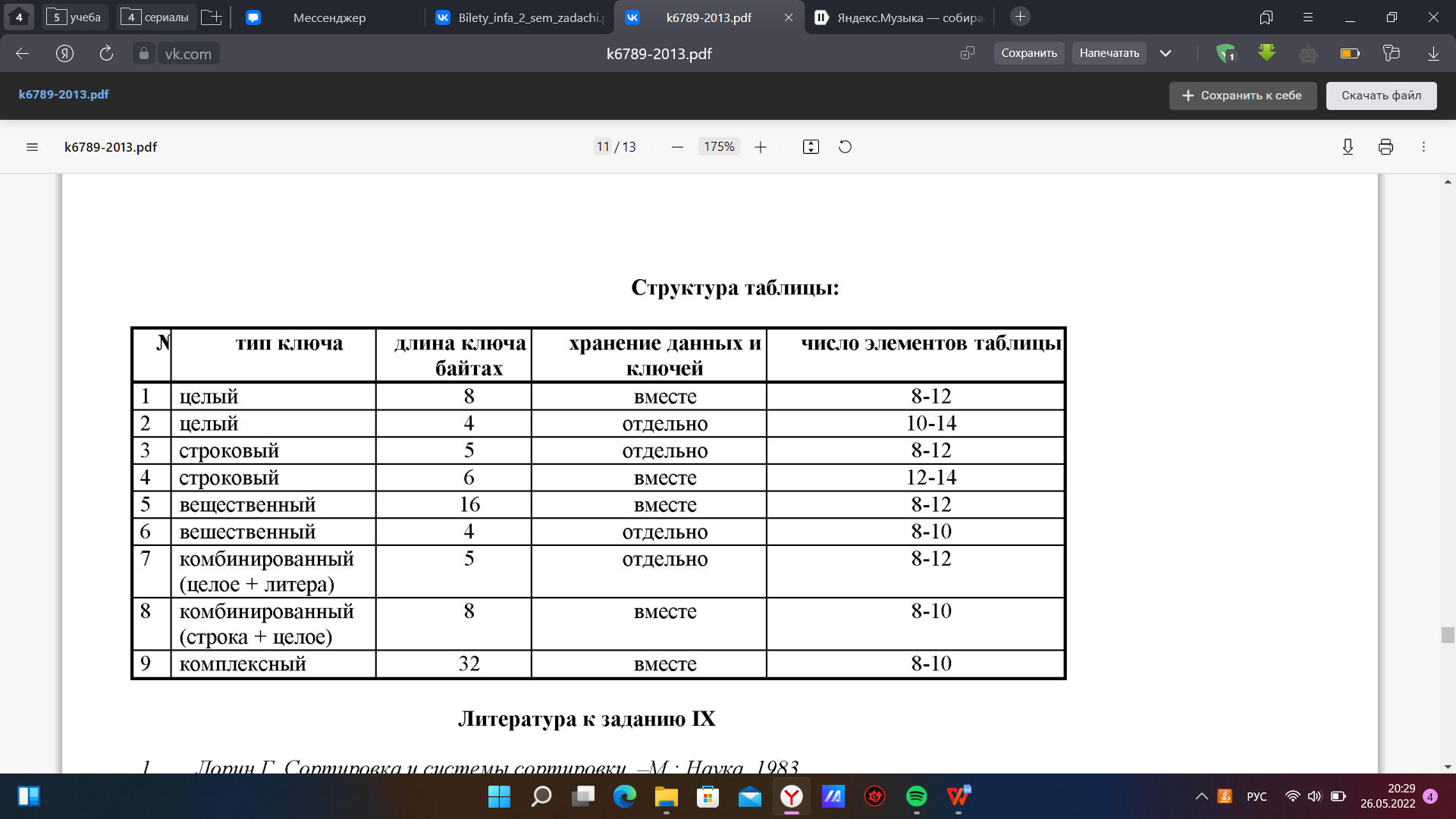
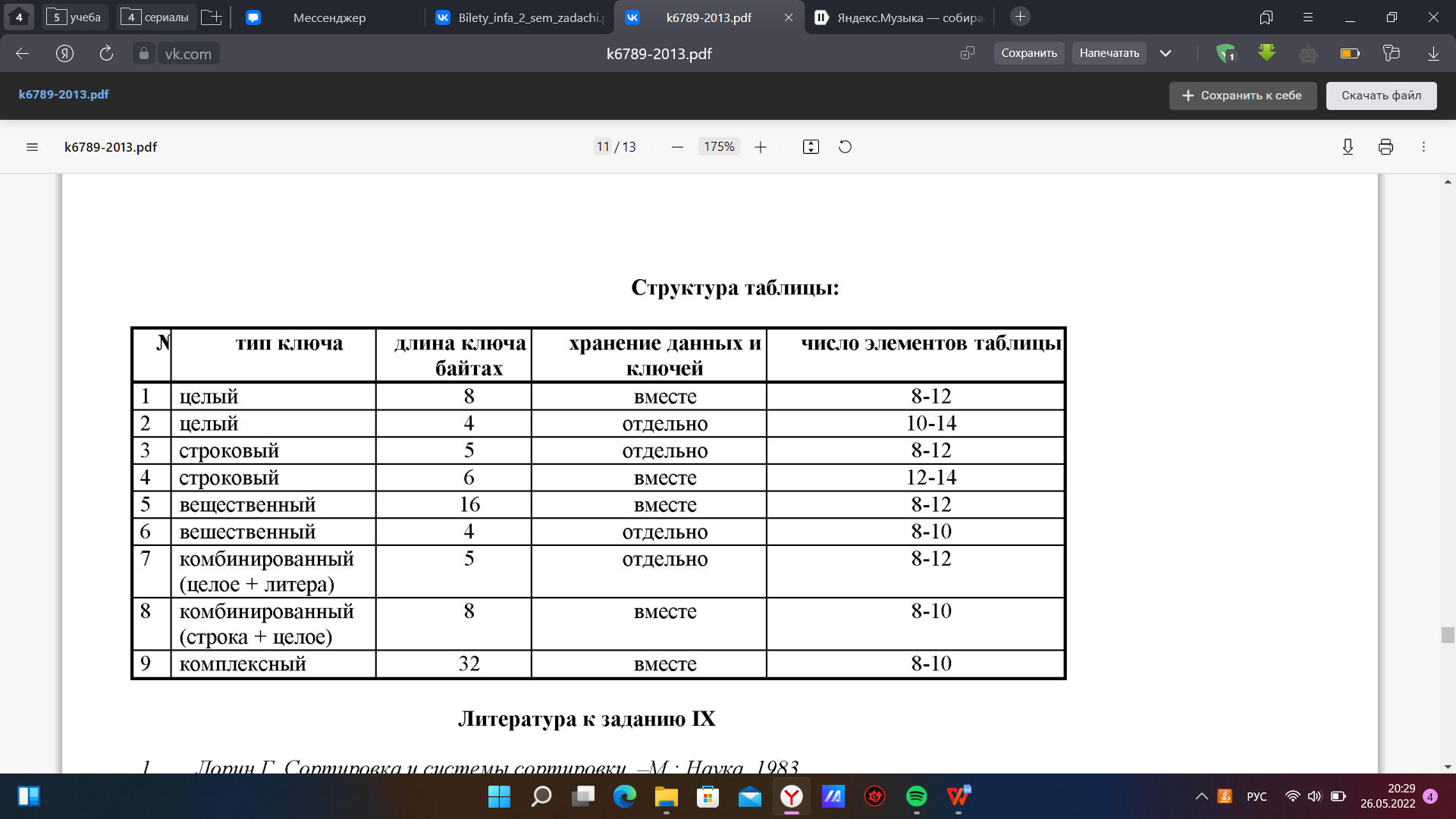
При работе с данными может потребоваться их отсортировать, тогда встает вопрос о том, как это сделать. Существует множество методов сортировки, каждый из которых имеет разную сложность. В этой работе будет рассмотрена сортировка таблицы простой вставкой, а так же бинарный поиск по отсортированным данным.

*Вариант:*

Метод сортировки



Структура таблицы



# Теоретическая часть

## Сортировка простой вставкой

Алгоритм сортировки - алгоритм по упорядочиванию элементов. В случае, когда элемент в массиве имеет несколько полей, поле, служащее критерием порядка, называется ключом сортировки. На практике в качестве ключа часто выступает число, а в остальных полях хранятся какие-либо данные, никак не влияющие на работу алгоритма.

Сортировка вставками (Insertion sort) – простой алгоритм сортировки, на каждом шаге которого мы выбираем один из элементов входных данных и вставляем его на нужную позицию в уже отсортированном массиве до тех пор, пока набор входных данных не будет исчерпан. Метод выбора очередного элемента из исходного массива произволен, может использоваться практически любой алгоритм выбора. Обычно (и с целью получения устойчивого алгоритма сортировки), элементы вставляются по порядку их появления во входном массиве.

Асимптотика в среднем и худшем случае – O(n2), в лучшем – O(n).

Устойчивая сортировка - сортировка, которая не меняет взаимного расположения равных элементов. Если элемент состоит только из одного ключа (т. е. значения, по которому и производится сортировка) то устойчивость сортировки не важна, но при сортировке списка, состоящего из кортежей (например, возраст и фамилия) в поддержании исходного порядка может быть смысл, так как равные по ключу элементы различны.

Сортировка простой вставкой устойчива.

# Практическая часть

## Об алгоритме

На вход подаются поля таблицы с соответствующими ключами. Ключи имеют тип данных long double и хранятся вместе с данными поля. С помощью сортировки простой вставкой таблица сортируется по ключам, после чего можно ввести какой-либо ключ, и программа выведет все поля с соответствующим ключом. Для поиска по ключу применяется бинарный поиск. Сложность бинарного поиска O(), сложность сортировки простой вставкой O().

## Протокол

*Реализация вектора*

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include "vector.h"

void print(vector \*v)

{

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

printf("%Lf %s\n", v->begin[i].key, v->begin[i].value);

}

printf("\n");

}

bool is\_empty(vector \*v)

{

return v->size == 0;

}

bool is\_full(vector \*v)

{

return v->size == v->allocated;

}

size\_t size(vector \*v)

{

return v->size;

}

void create(vector \*v)

{

v->size = 0;

v->allocated = 0;

v->begin = NULL;

}

void destroy(vector \*v)

{

free(v->begin);

v->size = 0;

v->allocated = 0;

v->begin = NULL;

}

void push(vector \*v, value\_type value)

{

if (is\_full(v)) {

v->allocated += 10;

v->begin = realloc(v->begin, v->allocated \* sizeof(value\_type));

}

v->begin[v->size] = value;

v->size++;

}

value\_type pop(vector \*v)

{

value\_type res = v->begin[v->size - 1];

v->size--;

return res;

}

int comparator(long double x, long double y)

{

if (x > y) {

return 1;

} else if (x < y) {

return -1;

} else {

return 0;

}

}

void insertion\_sort(vector \*v)

{

value\_type tmp;

int j;

for (int i = 1; i < size(v); i++) {

tmp = v->begin[i];

j = i - 1;

while ((j >= 0) && (comparator(v->begin[j].key, tmp.key) == 1)) {

v->begin[j + 1] = v->begin[j];

j--;

}

v->begin[j + 1] = tmp;

}

}

int bin\_search(vector \*v, long double key)

{

int left = -1, right = size(v), mid;

while (right > left + 1) {

mid = (left + right) / 2;

if (comparator(key, v->begin[mid].key) == -1 || comparator(key, v->begin[mid].key) == 0) {

right = mid;

} else if (comparator(key, v->begin[mid].key) == 1) {

left = mid;

}

}

if (comparator(key, v->begin[right].key) == 0) {

return right;

} else {

return -1;

}

}

void searching(vector \*v, long double key)

{

int it = bin\_search(v, key);

if (it != -1) {

while(comparator(key, v->begin[it].key) == 0) {

printf("%Lf %s\n", v->begin[it].key, v->begin[it].value);

it++;

}

printf("\n");

} else {

printf("This key no exists\n");

}

}

*Основная программа*

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include "vector.h"

int main(int argc, char const \*argv[])

{

vector v;

create(&v);

value\_type t;

while (!feof(stdin)) {

printf("Enter key & value | other to break: ");

if (scanf("%Lf%s", &t.key, t.value) == 2) {

push(&v, t);

} else {

break;

}

}

printf("\n");

insertion\_sort(&v);

print(&v);

printf("\n");

fflush(stdin);

long double find;

int err;

char c;

while ((c = getchar()) != '\n' && c != EOF) { };

printf("\nSearch in table:\n");

while (1) {

if (!is\_empty(&v)) {

printf("Enter key | ctrl + d to break: ");

err = scanf("%Lf", &find);

if (err == 0) {

printf("Wrong input\n");

while ((c = getchar()) != '\n' && c != EOF) { };

continue;

}

if (err == EOF) {

break;

}

searching(&v, find);

} else {

printf("The table is empty\n");

break;

}

}

printf("\n");

destroy(&v);

return 0;

}

# Заключение

Сортировка простой вставкой не самый оптимальный вариант сортировки, так как имеет квадратичную сложность, но зато не требует выделения дополнительной памяти.

# Список источников

1. Описание алгоритмов сортировки

<https://habr.com/ru/post/335920/>

1. Сортировка вставками

<https://forkettle.ru/vidioteka/programmirovanie-i-set/108-algoritmy-i-struktury-dannykh/sortirovka-i-poisk-dlya-chajnikov/1005-sortirovka-vstavkami-insertion-sort>

1. Устойчивые сортировки

<https://foxford.ru/wiki/informatika/ustoychivost-sortirovok>