**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

**Институт среднего профессионального образования**

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

по дисциплине: «Основы алгоритмизации и программирования»

**Студент группы № 219/2**

**Клопова Ульяна**

**Вариант 15**

**Преподаватель: Рябова В.В.**

Санкт-Петербург

2024 г.

***1 часть Сортировка и бинарный поиск в одномерных массивах.***

Дан **одномерный** массив размерностью [15] элементов. Заполнить этот массив случайными числами (диапазон выбрать произвольный, но чтоб включал и положительные и отрицательные числа). Отсортировать данные в массиве по возрастанию. Выбор способа сортировки осуществить в зависимости от варианта.

Вариант 15

Метод сортировки – расческой

Используя **бинарный** поиск, реализовать поиск значения (выбрать любое, на свое усмотрение) в отсортированном массиве

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <algorithm>

using namespace std;

void combSort(int a[], int n) {

float shrink = 1.3;

int step = n;

bool swapped = true;

while (step > 1 || swapped) {

step = (step > 1) ? step / shrink : 1;

swapped = false;

for (int i = 0; i + step < n; i++) {

if (a[i] > a[i + step]) {

swap(a[i], a[i + step]);

swapped = true;

}

}

}

}

int Search\_Binary(int a[], int left, int right, int key) {

int mid = 0;

while (left <= right) {

mid = left + (right - left) / 2;

if (key < a[mid]) {

right = mid - 1;

}

else if (key > a[mid]) {

left = mid + 1;

}

else return mid;

}

return -1;

}

int main() {

setlocale(0, "");

const int size = 15;

int a[size];

srand(time(0));

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

a[i] = rand() % 101 - 50;

}

cout << "Массив до сортировки:" << endl;

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

cout << a[i] << " ";

}

cout << endl;

combSort(a, size);

cout << endl;

cout << "Массив после сортировки:" << endl;

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

cout << a[i] << " ";

}

int key = 0, index =0;

cout << endl;

cout << "Введите любое число: " << endl;

cin >> key;

index = Search\_Binary(a, 0, size - 1, key);

if (index >= 0) {

cout << "Указаное число храниться в ячейке с индексом: " << index;

}

else

cout << "В массиве нет такого число!" << endl;

return 0;

}

**Запуск 1.**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Результат при бинарном поиске числа 44 в массиве отсортированном методом расчески

**Запуск 2.**

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Результат при бинарном поиске числа 12 в массиве отсортированном методом расчески

**Запуск 3.**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Результат при бинарном поиске числа 7в массиве отсортированном методом расчески

**Указание:** для каждой задачи создайте программу с тремя запусками.

Заполните двумерный числовой массив М(7,7) случайными целыми числами в интервале [-16; 16]. Выведите значения элементов массива в виде матрицы, причём, на каждый элемент отведите 4 позиции. Ответ каждой задачи следует выводить с новой строки и с пояснительным текстом. Кроме того, следует каждый раз производить вывод массива на экран, если в нём произошли изменения: перестановка, замена, изменение значения элементов и т.д.

**Задача 6.1.**

Найдите произведение отрицательных элементов четвёртого столбца.

#include <iostream>

#include <iomanip>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

setlocale(0, "");

srand(time(0));

int m[7][7], p =1;

for (int i = 0; i < 7; i++) {

for (int j = 0; j < 7; j++) {

m[i][j] = rand() % 32 - 16;

cout << setw(4) << m[i][j];

}

cout << endl;

}

for (int i = 0; i < 7; i++) {

if (m[i][3] < 0) {

p \*= m[i][3];

}

}

cout << "Произведение отрицательных элементов= " << p << endl;

}

**Запуск 1.**

**Изображение выглядит как текст, снимок экрана, Шрифт, типография

Автоматически созданное описание**

Результат первого запуска

**Запуск 2.**

**Изображение выглядит как текст, снимок экрана, Шрифт, типография

Автоматически созданное описание**

Результат при втором запуске

**Запуск 3.**

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание**

Результат при третьем запуске

**Задача 6.2.**

Найдите количество положительных элементов под главной диагональю массива.

#include <iostream>

#include <iomanip>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

setlocale(0, "");

srand(time(0));

int m[7][7], c=0;

for (int i = 0; i < 7; i++) {

for (int j = 0; j < 7; j++) {

m[i][j] = rand() % 32 - 16;

cout << setw(4) << m[i][j];

}

cout << endl;

}

for (int i = 0; i < 7; i++) {

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

if (m[i][j] > 0) {

c+=m[i][j];

}

}

}

cout << "Количество положительных элементов под главной диагональю: " << c << endl;

}

**Запуск 1.**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Результат первого запуска

**Запуск 2.**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Результат при втором запуске

**Запуск 3.**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Результат при третьем запуске

**Задача 6.3.**

Найдите минимальный элемент в каждом нечётном столбце массива.

#include <iostream>

#include <iomanip>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

setlocale(0, "");

srand(time(0));

int m[7][7], min = m[0][0];

for (int i = 0; i < 7; i++) {

for (int j = 0; j < 7; j++) {

m[i][j] = rand() % 32 - 16;

cout << setw(4) << m[i][j];

}

cout << endl;

}

for (int j = 1; j < 7; j += 2) {

min = m[0][j];

for (int i = 1; i < 7; i++) {

if (m[i][j] < min) {

min = m[i][j];

}

}

cout << "Минимальный элемент в столбце " << j << ": " << min << endl;

}

}

**Запуск 1.**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

Результат первого запуска

**Запуск 2.**

**Изображение выглядит как текст, снимок экрана, Шрифт, типография

Автоматически созданное описание**

Результат при втором запуске

**Запуск 3.**

**Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание**

Результат при третьем запуске