Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

Электротехнический факультет Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т по лабораторной работе**

Дисциплина: «Основы алгоритмизации и программирования»

Выполнил студент группы ИВТ21-1Б Лебедев Т.А.

Проверил Яруллин Д.В.

Пермь, 2021

**Постановка задачи:**

Для линейного и интерполяционного поиска: Дан массив целых чисел (или символов, если это дано по заданию), выполнить задание из своего варианта, используя указанный метод поиска. Массив можно заполнить вручную или случайными числами, размерность массива любая (не меньше 5). Массив для интерполяционного поиска предварительно отсортировать.

**Анализ решения задачи:**

создаем функцию для поиска индекса интересующего нас элемента

пока мы не нашли нужный нам элемент

формула для нахождения индекса

если элемент меньше искомого элемента, то сдвигаем левую границу на один вправо

если элемент больше искомого, то сдвигаем правую границу на один влево

если мы попали, то возвращаем индекс как результат функции

возвращаем -1 если нет такого элемента

функция для вывода массива

просим ввести размер массива

просим ввести искомый элемент

объявляем динамический массив типа int

рандомно его заполняем

выводим получившийся массив

сортируем его

выводим отсортированный массив

вызываем функцию для нахождения индекса искомого элемента

если не нашёлся элемент, то сообщаем об этом

если нашёлся, то меняем этот элемент местами с соседним

освобождаем память компьютера от динамического массива

**Описание переменных:**

n-количество элементов массива

templ-вспомогательная переменная, в которую мы будем записывать соседний элемент

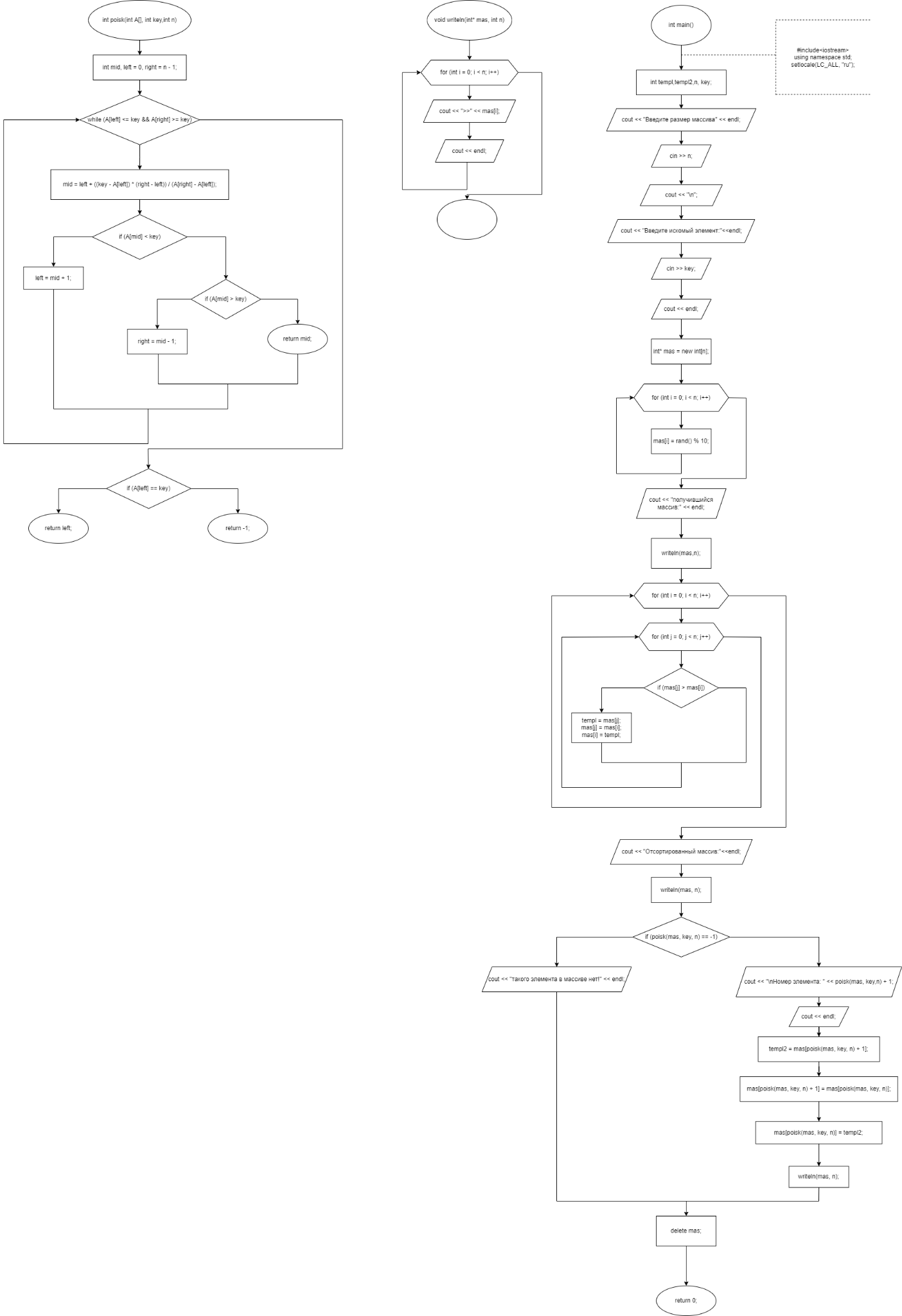
key-искомый элемент

left-левая граница массива

right-правая граница массива

mid-индекса искомого элемента

**Блок схема:**

****

**Код программы:**

#include<iostream>

using namespace std;

int poisk(int A[], int key,int n)//создаём функцию для поиска индекса интересующего нас элемента

{

int mid, left = 0, right = n - 1;

while (A[left] <= key && A[right] >= key)//пока мы не нашли нужынй нам элемент

{

mid = left + ((key - A[left]) \* (right - left)) / (A[right] - A[left]);//формула для нахождения индекса

if (A[mid] < key) left = mid + 1;//если элемент меньше искомого элемента, то сдвигаем левую границу на один вправо

else if (A[mid] > key) right = mid - 1;//если элемент больше искомого, то сдвигаем правую границу на один влево

else return mid;//если мы попали, то возвращаем индекс как результат функции

}

if (A[left] == key) return left;

else return -1;//возвращаем -1 если нет такого элемента

}

void writeln(int\* mas, int n)//функция для вывода массива

{

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

{

cout << ">>" << mas[i];

cout << endl;

}

}

int main()

{

int templ,templ2,n, key;

setlocale(LC\_ALL, "ru");

cout << "Введите размер массива" << endl;//просим ввести размер массива

cin >> n;

cout << "\n";

cout << "Введите искомый элемент:"<<endl;//просим ввести искомый элемент

cin >> key;

cout << endl;

int\* mas = new int[n];//объявляем динамический массив типа int

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

{

mas[i] = rand() % 10;//рандомно его заполняем

}

cout << "получившийся массив:" << endl;

writeln(mas,n);//выводим получившийся массив

for (int i = 0; i < n; i++)//сортируем его

{

for (int j = 0; j < n; j++)

{

if (mas[j] > mas[i])

{

templ = mas[j];

mas[j] = mas[i];

mas[i] = templ;

}

}

}

cout << "Отсортированный массив:"<<endl;

writeln(mas, n);//выводим отсортированный массив

if (poisk(mas, key, n) == -1)//вызываем функцию для нахождения индекса искомого элемента

{

cout << "такого элемента в массиве нет!" << endl;//если не нашёлся элемент, то сообщаем об этом

}

else

{

cout << "\nНомер элемента: " << poisk(mas, key,n) + 1;//если нашёлся, то меняем этот элемент местами с соседним

cout << endl;

templ2 = mas[poisk(mas, key, n) + 1];

mas[poisk(mas, key, n) + 1] = mas[poisk(mas, key, n)];

mas[poisk(mas, key, n)] = templ2;

writeln(mas, n);

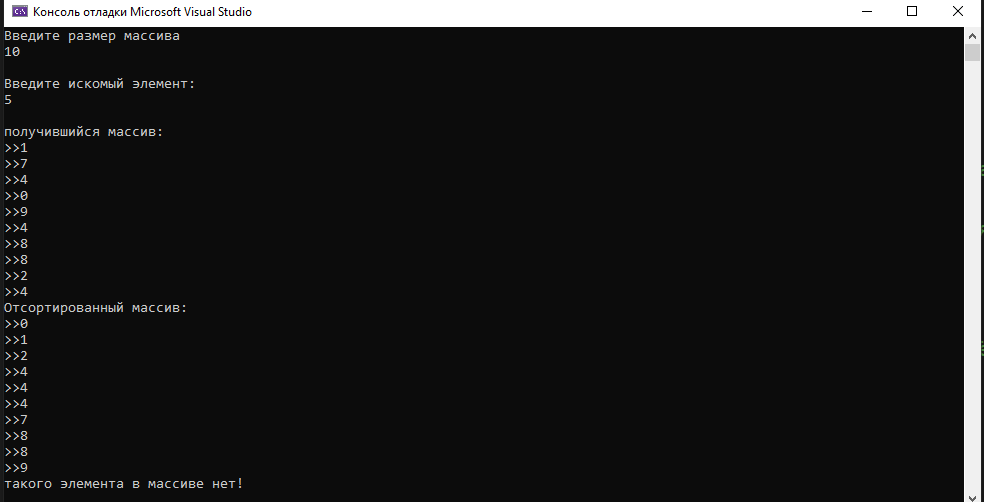
}

delete mas;//освобождаем память компьютера от динамического массива

return 0;

}

**Скриншоты результатов:**

****