**ЛАБОРАТОРНАЯ РАБОТА №4**

**«ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ОБРАБОТКИ ОДНОМЕРНЫХ СТАТИЧЕСКИХ МАССИВОВ»**

**Цель работы**

Изучить способы представления массивов в памяти ЭВМ, получить практические навыки реализации алгоритмов обработки одномерных массивов.

**Задания**

1. Разработать структурную схему решения задачи (вариант 12):

«В одномерном массиве, состоящем из N вещественных элементов, вычислить: 1) номер максимального по модулю элемента массива; 2) сумму элементов массива, расположенных после первого положительного элемента.

Преобразовать массив таким образом, чтобы сначала располагались все элементы, целая часть которых лежит внутри отрезка [а, b], а потом – все остальные. Упорядочить каждую часть массива по возрастанию, используя алгоритм сортировки методом прямого обмена. Значение а и b вводит пользователь».

1. Написать программу для решения задачи;
2. Разработать тестовые примеры и выполнить отладку программы;

**Текст программы**

#include<stdio.h>

#include<math.h>

main()

{

**const int** N=10; //длина массива

**int** i,j,t; //индексы

printf("N = %d\n", N); //вывод длины массива

**float** mas[N]; //массив

printf("Enter array elements:\n"); //заполнение массива

**for** (i=0;i<N;i++)

scanf("%f",&mas[i]);

//поиск позиции макс. по модулю элемента

**int** m=0; //позиция максимального по модулю числа

**for** (i=1;i<N;i++)

**if** (abs(mas[i])>abs(mas[m]))

m=i;

printf("\nPosition of the maximum modulo element of the array - %d\n\n",m);

//сумма элементов после первого положительного

**float** sum=0; //сумма

**int** fst=-1; //позиция первого положительного элемента

**for** (i=0;i<N;i++) //поиск первого положительного числа

**if** (mas[i]>=0) {

fst=i;

**break**;}

**if** (fst!=-1) //проверка на наличие положительного элемента

{ **for** (i=fst+1;i<N;i++) //суммирование

sum+=mas[i];

printf("Sum of elements after first positive one - %3.2f\n\n",sum); }

**else**

printf("No positive element\n\n"); //сообщение об отсутствии элементов >=0

//преобразование массива

**float** a,b;

printf("Enter a in [a..b]: "); //ввод a

scanf("%f",&a);

printf("Enter b in [a..b]: "); //ввод b

scanf("%f",&b);

**if** (a<=b) {

**for** (i=N;i>1;i--) //сортировка элементов из [a..b] прямым обменом

**for** (j=N-1;j>(N-i);j--)

{

**if** ( (mas[j]>=a) && (mas[j]<=b) && ((mas[j-1]>b) || (mas[j-1]<a) || (mas[j-1]>mas[j])) )

{

t=mas[j];

mas[j]=mas[j-1];

mas[j-1]=t;

}

}

**for** (i=N;i>1;i--) //сортировка элементов не из [a..b] прямым обменом

**for** (j=N-1;j>(N-i);j--)

{

**if** ( ((mas[j-1]>b) || (mas[j-1]<a)) && ((mas[j]>b) || (mas[j]<a)) && (mas[j-1]>mas[j]) )

{

t=mas[j];

mas[j]=mas[j-1];

mas[j-1]=t;

}

}

printf("\nSorted array:\n"); //вывод отсортированного массива

**for** (i=0;i<N;i++)

printf("%3.2f ",mas[i]); }

**else**

printf("\nError: a>b"); //сообщение об ошибке (a>b)

}

**Структурная схема программы**

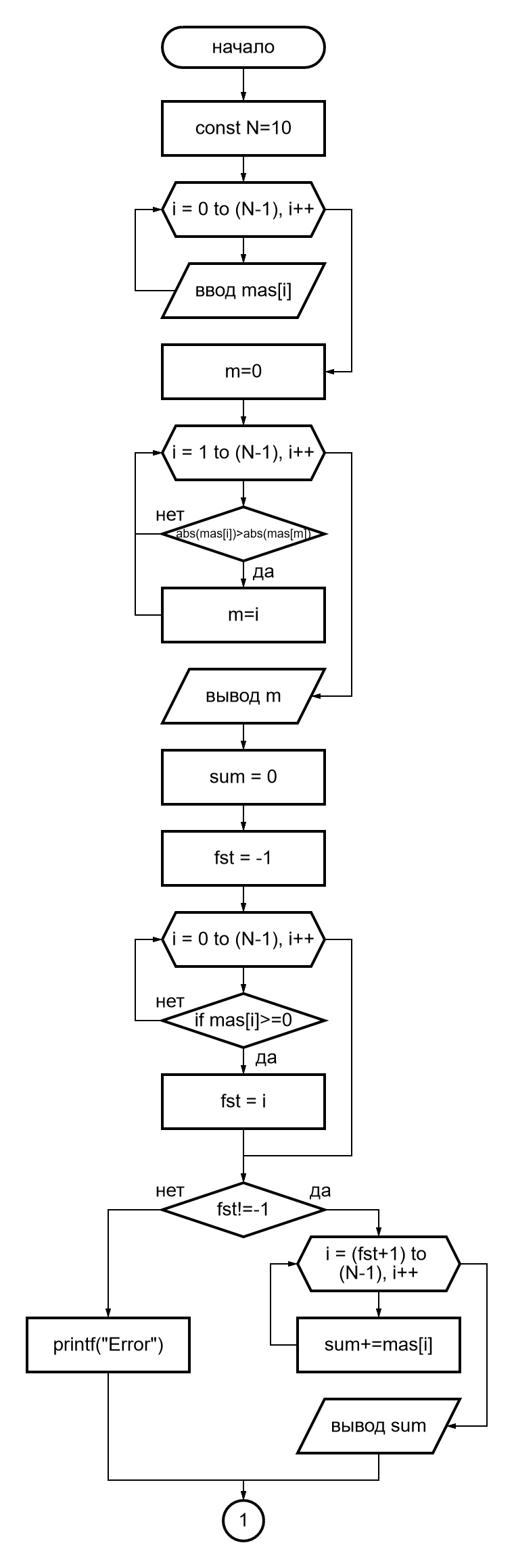


Рисунок 1 – Структурная схема

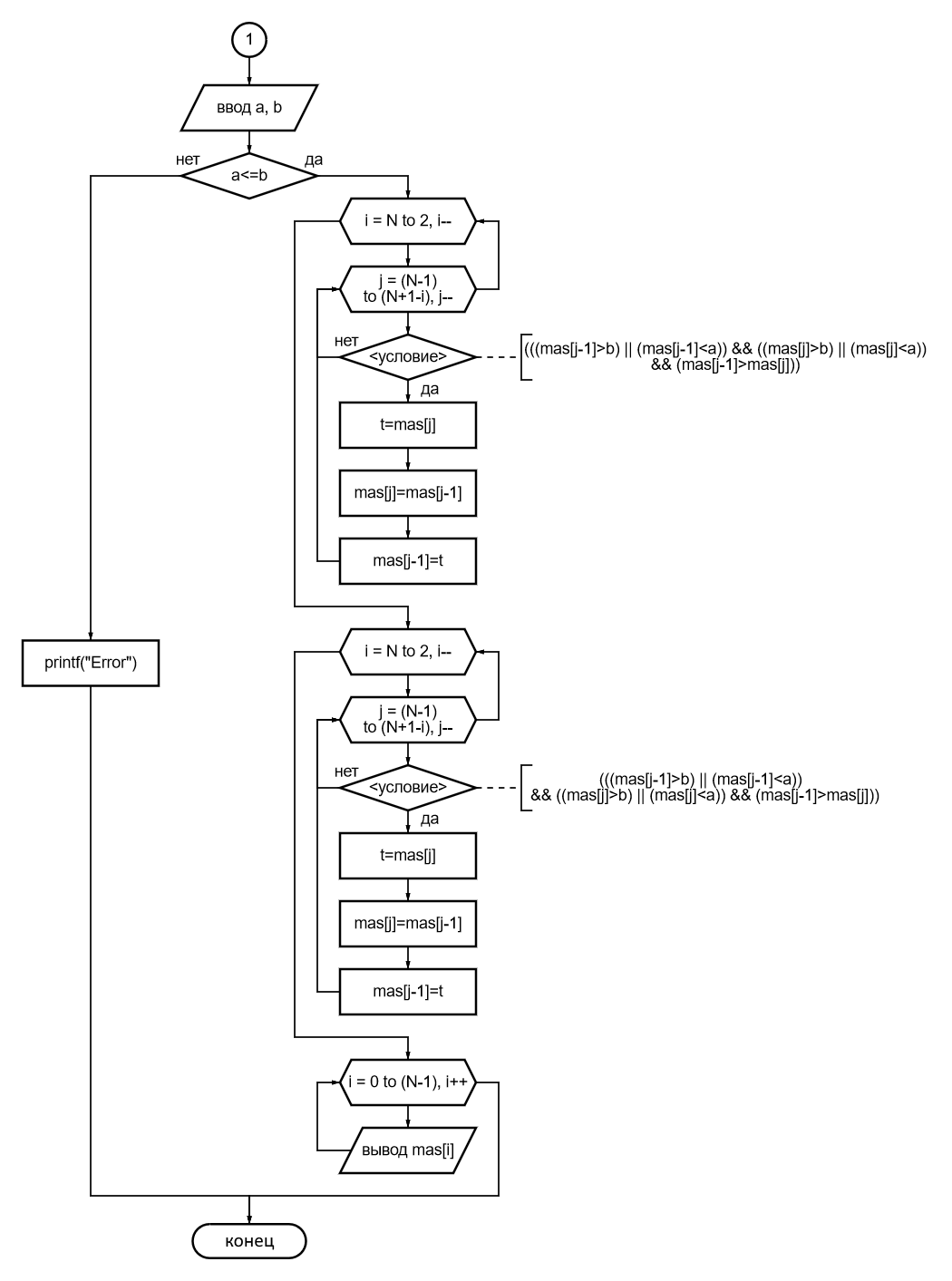


Рисунок 1 – Лист 2

**Тестовые примеры**

1. Массив: -5.0; -5.0; -5.0; -5.0; -5.0; -5.0; -5.0; -5.0; -5.0; -5.0; a=-4.0; b=-6.0. Программа должна вывести: позиция максимального по модулю элемента – 0, положительных чисел нет (сообщение об ошибке), a>b (сообщение об ошибке).

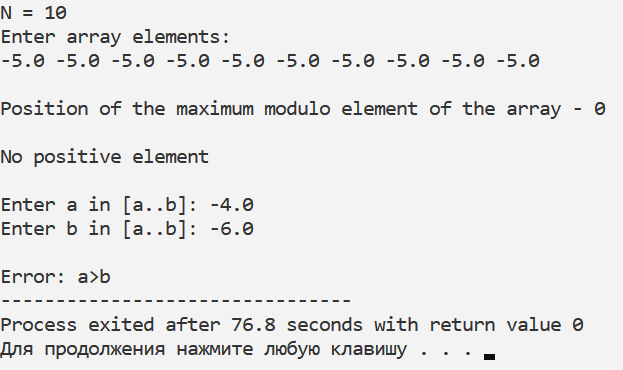


Рисунок 2 – Тестирование

1. Массив: -9.0; -8.0 -7.0; -6.0; -5.0; -4.0; -3.0; -2.0; -1.0; 0; a=-2.5; b=-0.5. Программа должна вывести: позиция максимального по модулю элемента – 0, сумма после первого положительного – 0, массив на выходе: -2.0; -1.0; -9.0; -8.0; -7.0; -6.0; -5.0; -4.0; -3.0; 0.

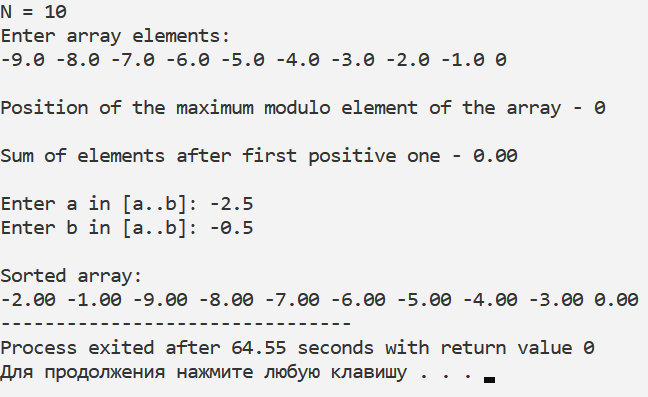


Рисунок 3 – Тестирование

1. Массив: -1.0; -2.0; -3.0; -4.0; -5.0; 0; 4.0; 3.0; 2.0; 1.0; a=-4.0; b=-4.0. Программа должна вывести: позиция максимального по модулю элемента – 4, сумма после первого положительного – 10.0, массив на выходе: -4.0; -5.0; -3.0; -2.0; -1.0; 0; 1.0; 2.0; 3.0; 4.0.

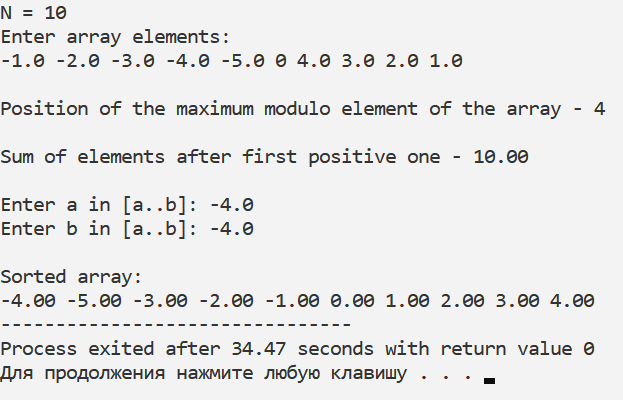


Рисунок 4 – Тестирование

1. Массив: 9.0; 8.0; 7.0; 6.0; 5.0; 4.0; 3.0; 2.0; 1.0; 0; a=-10.0; b=10.0. Программа должна вывести: позиция максимального по модулю элемента – 0, сумма после первого положительного – 36.0, массив на выходе: 0.0; 1.0; 2.0; 3.0; 4.0; 5.0; 6.0; 7.0; 8.0; 9.0.

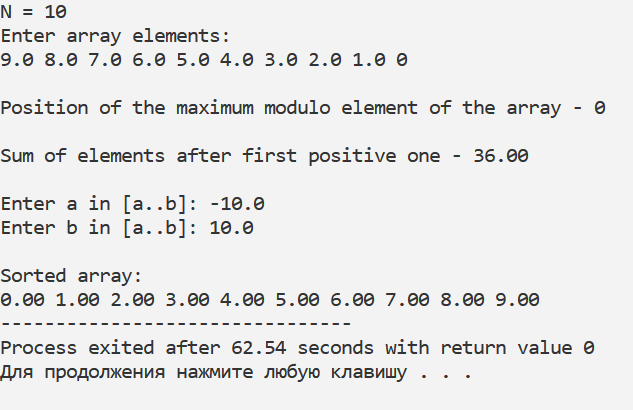


Рисунок 5 – Тестирование

1. Массив: -4.0; -9.0; -3.0; -8.0; -2.0; -7.0; -1.0; 0; -6.0; -5.0; a=1.0; b=2.0. Программа должна вывести: позиция максимального по модулю элемента – 1, сумма после первого положительного – -11, массив на выходе: -9.0; -8.0; -7.0; -6.0; -5.0; -4.0; -3.0; -2.0; -1.0; 0.

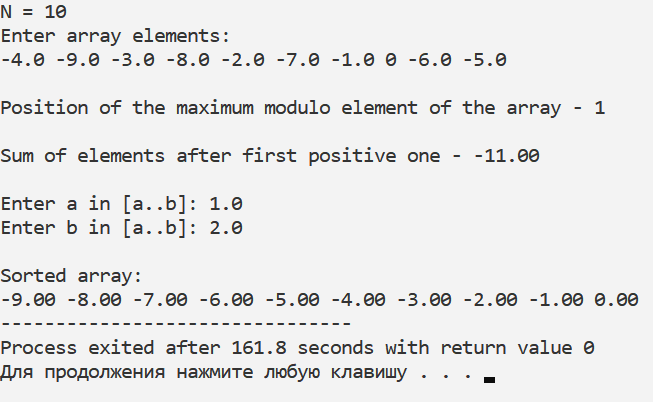


Рисунок 6 – Тестирование

По результатам тестирования программа вывела верные ответы для всех заданных тестовых примеров, на основании чего можно судить том, что алгоритм решения был написан верно.

**Вывод**

В ходе работы были изучены основные способы задания и обработки одномерных массивов в языке Си. Были получены практические навыки реализации алгоритмов обработки одномерных статических массивов.

Результатом работы стала программа, выполняющая поиск внутри одномерного массива, выборочное суммирование его элементов и его сортировку методом прямого обмена.