Федеральное государственное образовательное бюджетное учреждение

высшего профессионального образования

«Нижегородский государственный университет им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

Отчёт по лабораторной работе

Упорядочивание массивов и методы поиска данных

Выполнил:

студент ф-та ИТММ гр. 3821Б1ПМ3

Попов Д. Д.

Проверил:

Заведующий лабораторией СК и ВВ

Лебедев И. Г.

Нижний Новгород

2021 г.

Содержание

[1. Введение 3](#_Toc86228936)

[2. Постановка задачи 4](#_Toc86228937)

[3. Руководство пользователя 5](#_Toc86228938)

[4. Руководство программиста 6](#_Toc86228939)

[1) Описание структуры программы 6](#_Toc86228940)

[2) Описание структур данных 7](#_Toc86228941)

[3) Описание алгоритмов 8](#_Toc86228942)

[5. Эксперименты 12](#_Toc86228943)

[6. Заключение 14](#_Toc86228944)

[7. Литература 15](#_Toc86228945)

[8. Приложение 1 16](#_Toc86228946)

# Введение

Программирование — неотъемлемая часть нашей жизни. это возможность взаимодействия человека с машиной, возможность общения друг с другом на понятном языке. Всё, что мы делаем на компьютере, так или иначе связано с ними.  
Данная лабораторная работа поможет разобраться с тем, как устроены динамические массивы и как с ними работать, и как создавать случайные числа.

# Постановка задачи

Необходимо сделать программу, которая будет генерировать множество случайных чисел размера n в диапазоне (min, max), где n, min, max вводятся с клавиатуры. После чего подсчитывает и выводит сумму, которая получается следующим образом: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются.

# Руководство пользователя

Данная программа предусматривает ввод количества элементов, которое будет обрабатывается в работе. (рис. 1)

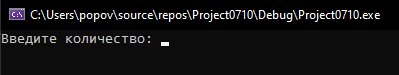


Рисунок 1. Ввод количества элементов

Также возможен ввод минимального (рис. 2) и максимального (рис. 3) значения.

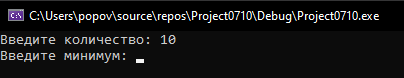


Рисунок 2. Ввод минимального значения

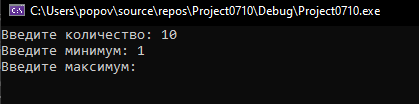


Рисунок 3. Ввод максимального значения

В конце своей работы программа выведет свой результат, т.е сумму. (рис, 4)

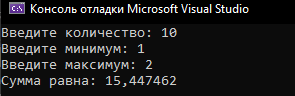


Рисунок 4. Вывод суммы

# Руководство программиста

## Описание структуры программы

Программа состоит из функции main() (рис. 5). В ней, через функции scanf\_s и printf, реализован ввод переменных, используемые для работы, а также проверка их на работоспособность в условиях этой программы, через условный оператор if (рис. 6)



Рисунок 5. Функция main()

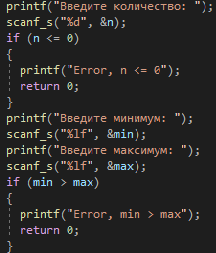


Рисунок 6.Ввод данных, а также их проверка

Через цикл for происходит инициализация массива, подсчет суммы всех элементов, а также сразу же производится отделение дробной части с помощью функции modf. (рис. 7)

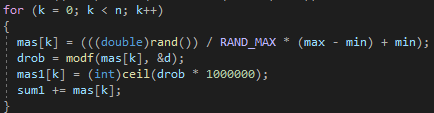


Рисунок 7. Цикл for для инициализации массива

Также имеется цикл for, который содержит внутри себя еще один цикл for. Он предназначен для сравнения дробной части и индексов элементов. (рис. 8)

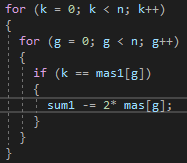


Рисунок 8. Цикл for для сравнения дробной части и индексов

Вывод суммы происходит через функцию printf. (рис. 9)



Рисунок 9. Вывод суммы через printf

В конце своей работы программа освобождает место в оперативной памяти с помощью функций free. (рис. 10)



Рисунок 10. Освобождение места в памяти

## Описание структур данных

В работе программы используются переменные типа int (рис. 11), double (рис. 12) и double\* и int\* для массивов (рис. 13).



Рисунок 11. Объявление переменных типа int



Рисунок 12. Объявление переменных типа double



Рисунок 13. Объявление переменных типов int\* и double\*

Также для работы используются следующие библиотеки. (рис. 14)

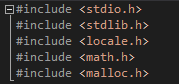


Рисунок 14. Библиотеки

## Описание алгоритмов

Представим алгоритмы, которые используются в программе с помощью блок схем.

Проверка min и max на то, не больше ли min, чем max. (рис. 15)



Рисунок 15. Блок схема на проверку max>min

Проверка количества элементов на натуральное значение, т.е n>0. (рис. 16)



Рисунок 16. Блок схема на проверку n >0

Работа первого цикла for. (рис. 17)

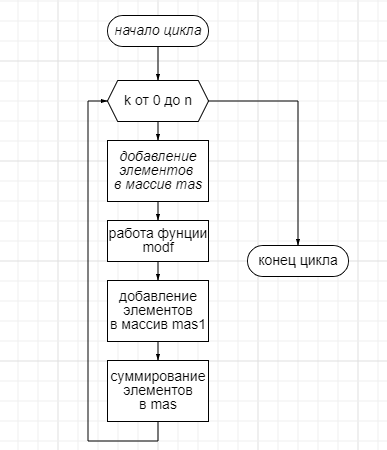


Рисунок 17. Работа первого цикла for

Алгоритм функции modf работает следующим образом. В нее поставляется число, которое необходимо разложить на целую часть и дробную. После чего она отделяет их друг от друга, и записывает значение целой части в переменную d, а дробную в drob.

(рис. 18 и рис. 19)



Рисунок 18. Пример функции modf

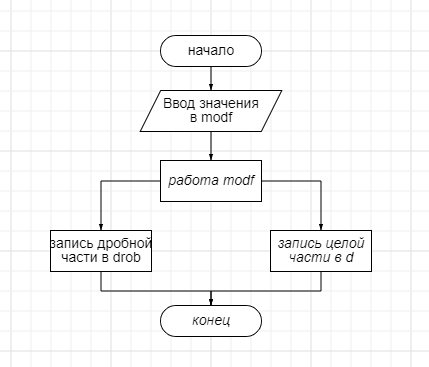


Рисунок 19. Работа функции modf

После того, как функция modf выделит дробную часть в переменную drob, программа умножит полученное значение на 1000000 и с помощью функции ceil, которая округляет число к наибольшему целому, а также приведения число к типу int, мы получим число целого типа, которое будет записано в массив mas1. (рис. 20)



Рисунок 20. Запись полученного целого числа в mas1

Алгоритм второго цикла for и его подцикла for сводится к тому, что каждый k сравнивается с каждым значением в mas1, если они совпадают то из суммы sum1 вычитается двойное значение mas1[k], так как до этого мы добавляли все значения в sum1. (рис. 21)

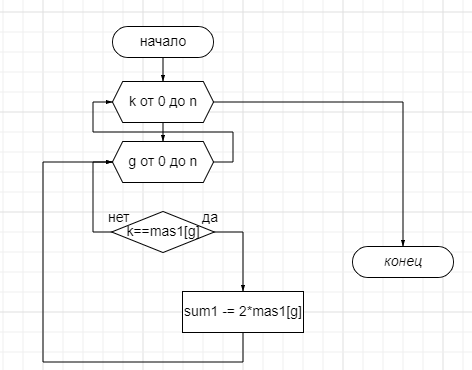


Рисунок 21. Принцип работы цикла for и подцикла for

# Эксперименты

Проведем некоторые эксперименты. Возьмем 10 элементов от 1 до 2 и посмотрим, что выведет программа. Также введем переменную sum2, которая будет считать просто сумму всех элементов для того, чтобы можно было сравнить с sum1. (рис. 22)

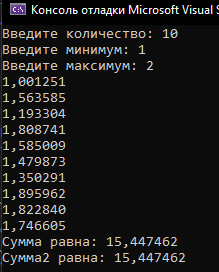


Рисунок 22. Работа программы при заданных условиях

Можно заметить, что программа получила 10 случайных чисел в диапазоне от 1 до 2, как и было сказано. Мы видим, что сумма, после выполнения всех действий, и сумма всех элементов не отличается. Это можно объяснить слишком малым количеством элементов.

Теперь попробуем взять 10000 элементов от 0 до 89, и посмотрим, что выведет программа в данном случае. (рис. 23)

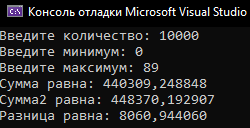


Рисунок 23. Эксперимент номер два

Заметим, что появилась разница между вычислениями. Это показывает, что программа работает.

Также можно проверить, работает ли проверка на правильный ввод данных.

(рис. 24 и рис. 25)



Рисунок 24. Error при n <= 0

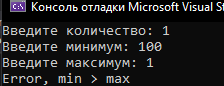


Рисунок 25. Error при max < min

Наглядно показано, что система защиты работает и полностью выполняет свою функцию.

# Заключение

Это была первая лабораторная работа. Она научила создавать случайно сгенерированные числа, работать с динамическими массивами в языке программирования C. Также по ней необходимо было написать отчет, который до этого никогда не писали.

# Литература

Керниган. Б. Язык программирования Си / Керниган Б., Ритчи Д. — «Вильямс»:

Издательский дом «Вильямс», 2015. — 304 с.

# Приложение 1

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <locale.h>
4. #include <math.h>
5. #include <malloc.h>
6. int main()
7. {
8. setlocale(LC\_ALL, "Rus");
9. double min, max, drob, d, sum1 = 0;
10. int k, g, n;
11. printf("Введите количество: ");
12. scanf\_s("%d", &n);
13. if (n <= 0)
14. {
15. printf("Error, n <= 0");
16. return 0;
17. }
18. printf("Введите минимум: ");
19. scanf\_s("%lf", &min);
20. printf("Введите максимум: ");
21. scanf\_s("%lf", &max);
22. if (min > max)
23. {
24. printf("Error, min > max");
25. return 0;
26. }
27. double\* mas = NULL;
28. int\* mas1 = NULL;
29. mas = ((double\*)malloc(n \* sizeof(double) + 1));
30. mas1 = ((int\*)malloc(n \* sizeof(double) + 1));
31. for (k = 0; k < n; k++)
32. {
33. mas[k] = (((double)rand()) / RAND\_MAX \* (max - min) + min);
34. drob = modf(mas[k], &d);
35. mas1[k] = (int)ceil(drob \* 1000000);
36. sum1 += mas[k];
37. }
38. for (k = 0; k < n; k++)
39. {
40. for (g = 0; g < n; g++)
41. {
42. if (k == mas1[g])
43. {
44. sum1 -= 2\* mas[g];
45. }
46. }
47. }
48. printf("Сумма равна: %lf\n", sum1);
49. free(mas);
50. free(mas1);
51. return 0;
52. }