Федеральное агентство по образованию Российской Федерации

Государственное образовательное учреждение

высшего профессионального образования

Нижегородский государственный университет им. Н. И. Лобачевского

Институт информационных технологий математики и механики

Отчёт по лабораторной работе

Работа с массивами и генерация случайных чисел

Выполнил:

Студент гр. 3821Б1ПМ3

Мартынов А.Ю.

Проверил:

Заведующий лабораторией

суперкомпьютерных технологий и

высокопроизводительных

вычислений

Лебедев И. Г.

Нижний Новгород

2021 г.

Оглавление

[1. Введение 3](#_Toc90372440)

[2. Постановка задачи 4](#_Toc90372441)

[3. Руководство пользователя. 4](#_Toc90372442)

[4. Руководство программиста 4](#_Toc90372444)

[5.Эксперименты: 8](#_Toc90372445)

[6. Заключение 10](#_Toc90372446)

# 

# 1. Введение

На сегодняшний день программирование играет ведущую роль в жизни.

Программирование – создание оптимизированных алгоритмов для решения различного рода задач. Программы создаются с помощью специальных языков программирования, имеющих некоторое сходство с естественными языками, хотя это не всегда. Они созданы для того, чтобы интерпретировать команды человека в машинный код. Язык C является фундаментом для любого программиста, так как нём реализано огромное количество возможностей для разработчика, из - за он является одним из фундаментальных языков программирования. На нём можно написать огромное количество различных программ. Я также написал свою лабораторную на языке C.

# 2. Постановка задачи

Написать программу, которая генерирует множество случайных чисел размера n в диапазоне (min, max), где n, min, max вводятся с клавиатуры. После этого она должна посчитать сумму по следующему условию: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются. Результатом работы программы должно одно число – полученная сумма.

# 3. Руководство пользователя.

# Руководство пользователя

На вход программы подаются следующие значения: минимальное значение диапазона случайных чисел, затем максимальное число из диапазона случайных чисел, количество случайных чисел(Рис.1). Пользователь должен вводить числа в строгом порядке : минимальное, максимальное(Рис.2), количество, также он должен следить за тем, чтобы минимальное число не было больше максимального, количество числе не должно быть равно нулю.

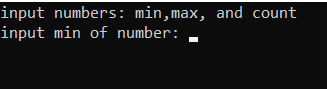


Рис.1- строка, подсказывающая о количестве и назначении переменных

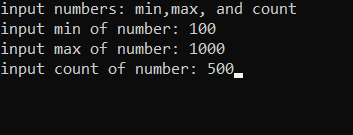


Рис.2 – ввод переменных

# 4. Руководство программиста

**4.1 Описание структуры программы.**

1) Первым идёт блок кода, в котором идет подключение библиотек.(Рис.2). <stdlib.h> для определения функции rand(); <stdio.h> -ввод/вывод данных и другие стандартные возможности языка C



Рис 2 Фрагмент кода(1) – подключение библиотек.

2) Блок функций: remains и main – тело программы.

3) В функцию main можно разделить на некоторые блоки. Первый из них – это объявление необходимых переменных(Рис.3)

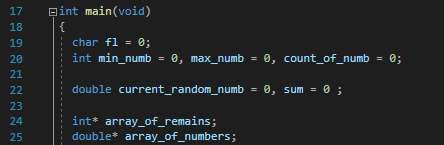


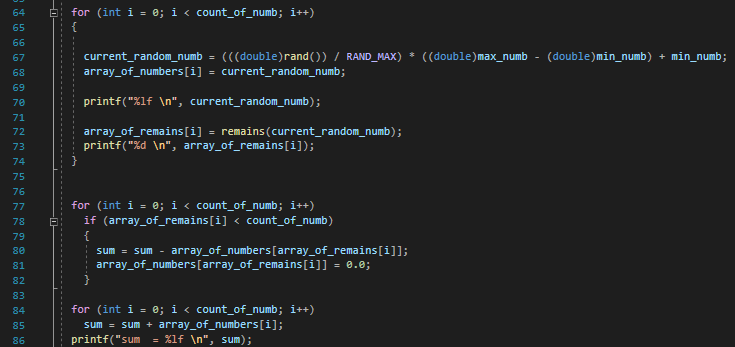
Рис. 3 Фрагмент кода(2) – блок определения переменных.

4) блок ввода данных и их проверки (Рис. 4)



Рис.4 Фрагмент кода(3) – блок ввода и проверки данных

5) блок заполнения массива и подсчёта суммы (Рис. 5)

Рис. 5. Фрагмент кода(4) - заполнение массива и подсчёт суммы

**4.2 Описание структуры данных.**

В программе использовалось несколько типов данных.

1)В данной программе используются 2 переменные типа double current\_random\_numb и sum. current\_random\_numb – эта переменная хранит дробное случайное число которое генерируется на n шаге в цикле for.

2)Используются 3 переменные типа int для хранения данных, вводимых пользователем: min\_numb, max\_numb, count\_of\_numb.

3)Переменная fl типа char в качестве флага.

4)Для хранения остатков и чисел используется динамический массив array\_of\_remains типа int и динамический массив array\_of\_numbers типа double.

**4.3 Описание алгоритмов**

В данной программе имеются несколько ключевых алгоритмов:

1) функция выделения остатка. Функция int remains(double double\_number) – функция выделения остатка из дробного числа. Принимает на вход случайное число типа double. Далее мы из дробного числа вычитаем его целую часть – число без остатка. Полученный остаток мы умножаем на 1000, чтобы число получилось не таким большим. Потом мы проверяем число на следующее условие: если остаток от деления переменной rem равен 0, то мы “отрезаем” этот 0, получается, что проверяем число на то, что если справа после умножения оказались нули, то мы их удаляем(Рис.6)

8

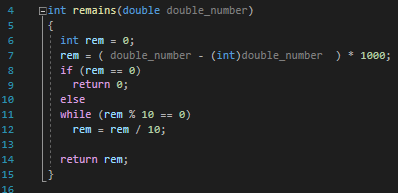


Рис. 6 Фрагмент кода(5) – реализация функции по выделению остатка.

2) Алгоритм подсчёта суммы (Рис.7), блок-схема №1 состоит из двух частей -. В первом цикле сначала идет проверка на то, что элемент массива array\_of\_remains меньше переменной, хранящей в себе количество чисел, после чего элемент массива array\_of\_numbers с индексом array\_of\_remains вычитается из значения итоговой суммы и зануляется. После этого мы идём по всем остаткам и перебираем их. После завершения цикла мы получаем то, что значение суммы остаётся отрицательным. Затем мыы идём но новому циклы и суммируем array\_of\_numbers[i] и переменную sum. В итоге получаем итоговое значение суммы.

(Рис.7) , блок-схема №1

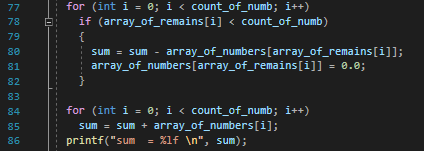
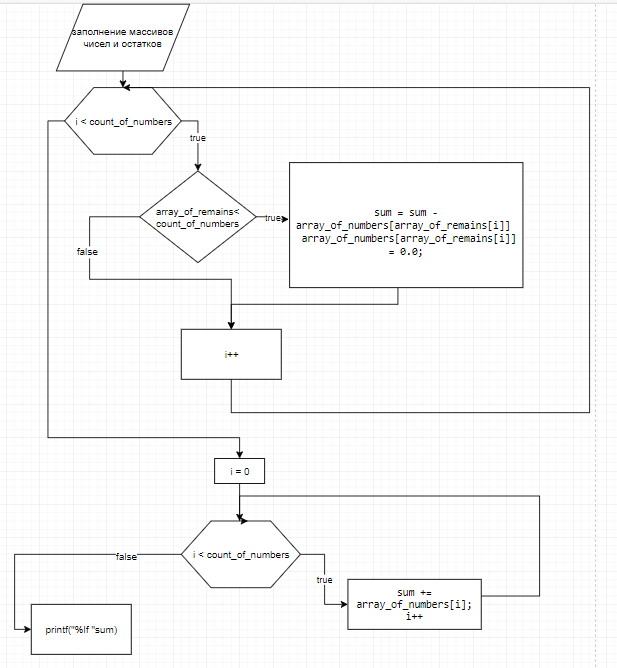


Рис.7 Фрагмент кода(6) – реализация суммы

**Блок-схема №1.**

****

# 5.Эксперименты:

Для того, чтобы удостовериться в работоспособности программы проведем ряд тестов.

Для упрощения анализа, будем задавать параметры, основываясь на которых, программа будет генерировать маленькие числа, также количество из будет мало, это нужно для того, чтобы мы могли вручную сосчитать итоговую сумму и сравнить её с получившейся. Изменим код программы, чтобы в цикле мы могли видеть полученные остатки. На первом рисунке(Рис.8) предоставлены результаты работы программы для диапазона [7;9] и количества в 10 чисел.

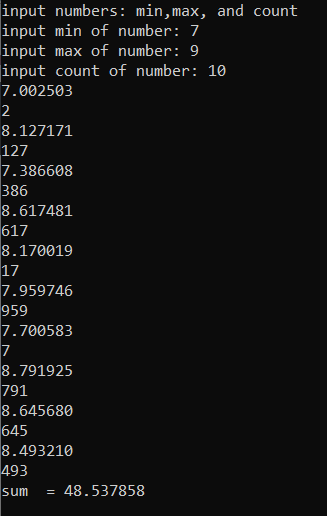


Рис. 8. Пример работы программы с изменениями.

%3CmxGraphModel%3E%3Croot%3E%3CmxCell%20id%3D%220%22%2F%3E%3CmxCell%20id

11

# 6. Заключение

В ходе выполнения лабораторной работы №1, мною написана программа на языке я С, которая находит значения суммы случайных чисел, при заданных условиях. Научился создавать случайные числа в выбранном диапазоне, считать их сумму при определённых условиях. Научились использовать динамические в качестве ячейки хранения информации и обращаться к ним в цикле for.

**7. Литература**

1. Бьерн Страуструп. Язык программирования С++ - Режим доступа: [..\Downloads\Straustrup-Yazyk\_programmirovaniya\_c.pdf](file:///C:\Users\Alex\Downloads\Straustrup-Yazyk_programmirovaniya_c.pdf)

**8. Приложение 1**

#include<stdio.h>

#include<stdlib.h>

int remains(double double\_number)

{

int rem = 0;

rem = ( double\_number - (int)double\_number ) \* 1000;

if (rem == 0)

return 0;

else

while (rem % 10 == 0)

rem = rem / 10;

return rem;

}

int main(void)

{

char fl = 0;

int min\_numb = 0, max\_numb = 0, count\_of\_numb = 0;

double current\_random\_numb = 0, sum = 0 ;

int\* array\_of\_remains;

double\* array\_of\_numbers;

printf("input numbers: min,max, and count \n");

printf("input min of number: ");

scanf\_s("%d", &min\_numb);

printf("input max of number: ");

scanf\_s("%d", &max\_numb);

printf("input count of number: ");

scanf\_s("%d", &count\_of\_numb);

array\_of\_remains = (int\*)malloc(count\_of\_numb \* sizeof(int));

array\_of\_numbers = (double\*)malloc(count\_of\_numb \* sizeof(double));

if (array\_of\_remains == 0)

{

printf("Error: memory is not allocated \n");

return 0;

}

if (array\_of\_numbers == 0)

{

printf("Error: memory is not allocated \n");

return 0;

}

if (max\_numb < min\_numb)

{

printf("Error: min number > max number");

return 0;

}

if (count\_of\_numb == 0)

{

printf("Error: count of number = 0");

return 0;

}

for (int i = 0; i < count\_of\_numb; i++)

{

current\_random\_numb = (((double)rand()) / RAND\_MAX) \* ((double)max\_numb - (double)min\_numb) + min\_numb;

array\_of\_numbers[i] = current\_random\_numb;

printf("%lf \n", current\_random\_numb);

array\_of\_remains[i] = remains(current\_random\_numb);

printf("%d \n", array\_of\_remains[i]);

}

for (int i = 0; i < count\_of\_numb; i++)

if (array\_of\_remains[i] < count\_of\_numb)

{

sum = sum - array\_of\_numbers[array\_of\_remains[i]];

array\_of\_numbers[array\_of\_remains[i]] = 0.0;

}

for (int i = 0; i < count\_of\_numb; i++)

sum = sum + array\_of\_numbers[i];

printf("sum = %lf \n", sum);

free(array\_of\_remains);

free(array\_of\_numbers);

return 0;

}