Федеральное государственное образовательное бюджетное учреждение

высшего профессионального образования

«Нижегородский Государственный Университет им.

Н.И.Лобачевского» (ННГУ)

Институт Информационных Технологий Математики и Механики

Отчёт по лабораторной работе

Работа со случайно сгенерированными числами

Выполнил:

студент группы 3821Б1ФИ3

Казанцев Е. А.

Проверил:

заведующий лабораторией суперкомпьютерных технологий и высокопроизводительных вычислений

Лебедев И.Г

Нижний Новгород

2021г.

1. Содержание

[Введение 3](#_Toc85900685)

[Постановка задачи 4](#_Toc85900686)

[Руководство пользователя 5](#_Toc85900687)

[Руководство программиста 6](#_Toc85900688)

[Описание структуры кода программы 6](#_Toc85900689)

[Описание структуры данных 8](#_Toc85900690)

[Описание алгоритмов 9](#_Toc85900691)

[Эксперименты 11](#_Toc85900692)

[Список литературы 14](#_Toc85900693)

# Введение

**Программирование** — это процесс превращения алгоритма в нотацию, написанную на языке программирования, которая может быть выполнена компьютером. Одной из ключевых задач компьютера является работа с данными. В том числе и со случайно (псевдослучайно) генерируемыми наборами данных, о которых пойдет речь в настоящей работе.

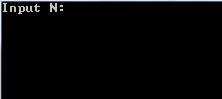
Случайные числа — это одна из основных составляющих любого языка программирования, на них строятся многие алгоритмы. Случайные числа широко используются в различных областях науки и техники, например, при вычислении многомерных интегралов, моделировании различных процессов методом Монте-Карло. Наиболее широкое применение, случайные числа находят в криптографии. Случайные последовательности используются для секретных ключей в системах симметричного шифрования, генерации паролей, PIN кодов для различных типов пластиковых карт, кодов аутентификации, вероятностных алгоритмов и систем квантового распределения ключей.

В ходе выполнения лабораторный работы на языке программирования «С» будет написана программа, работающая со случайными числами.

# Постановка задачи

Цель лабораторной работы — при помощи функцией «rand» сгенерировать множество псевдослучайных чисел. Реализовать возможность ввода пользователем с клавиатуры размера и диапазона значений множества. Посчитать сумму сгенерированных чисел, соблюдая условие: числа, номера которых совпали с дробной частью исходных, должны быть вычтены из суммы.

# Руководство пользователя

После запуска программа выводит в консоль сообщение «Input N:»

означающее, что пользователю, для дальнейшей работы, требуется ввести число элементов массива случайных чисел N. (см. рис. 1).

Рисунок 1. Консоль после запуска программы.

Затем, программа попросит пользователя поочередно ввести нижнюю границу генерации чисел min и верхнюю max. После каждого введенного значения необходимо нажимать кнопку «Enter». (см. рис 2, 3).

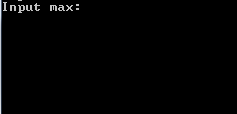
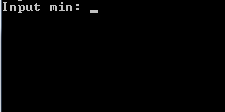


Рисунок 2. Ввод нижней границы. Рисунок 3. Ввод верхней границы.

Например, введем N = 1000; min = 0; max = 100.

После ввода всех значений и нажатия клавиши «Enter» программа выведет результат суммирования сгенерированных чисел, произведенного по правилам технического задания. На этом программа завершается. (см. рис. 4).

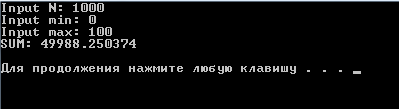


Рисунок 4. Результат суммирования.

# Руководство программиста

## Описание структуры кода программы

1. Подключение библиотек, с которыми предстоит работать.
2. Объявление переменных: 1) генерации счета, индекса счета, временная переменная, на случай неправильного ввода нижней и верхней границы генерации. 2) Переменная генерации, дробная переменная, переменные нижней и верхней границы. 3) Переменная целой части и суммы.
3. Объявление функции main(void), получение данных от пользователя, а также проверка полученных данных на корректность.

4.Происходит генерация случайных чисел и отделение дробной части для последующей проверки.

5. В случае, если номер какого-либо числа совпадет с дробной частью одного из чисел, то число с таким номером не идет в итоговую сумму. Для этого создаем циклы, в которых дробная часть проходит проверку на совпадение с номером. Проверка происходит по такому принципу: если N от 1 до 10, тогда наша дробная часть умножается на 10 и проверятся на совпадение с номером.

6. Результатом программы является сложение всех случайно сгенерированных чисел для получения требуемой техническим заданием суммы, вывод результата в терминал, завершение программы.

## Описание структуры данных

* В программе используются переменный счетчик типа int «i», количество генерируемых чисел «N» типа int.
* Переменные min, max и sum относятся к типу данных double
* Случайные числа, создаваемые в ходе работы программы, записываются в транзитную переменную «a»
* А также для функционирования программы требуются библиотеки: stdio.h; stdlib.h и math.h

## Описание алгоритмов

В программе реализованы два важных алгоритма

1. Проверяет на подлинность вводимых данных. Имеется в виду, что если минимальное число по ошибке пользователя будет больше максимального числа, тогда алгоритм запросит пользователя ввести данные вновь.

2. Во втором алгоритме производится отделение дробной части от сгенерированного числа и проверка этой дробной части на совпадение с номером этого числа. Далее происходит суммирование всех чисел, дробная часть которых не совпала с номером. Те, которые совпали, вычитаются из общей суммы. После всех операций происходит вывод суммы в консоль. (см. рис. 5).

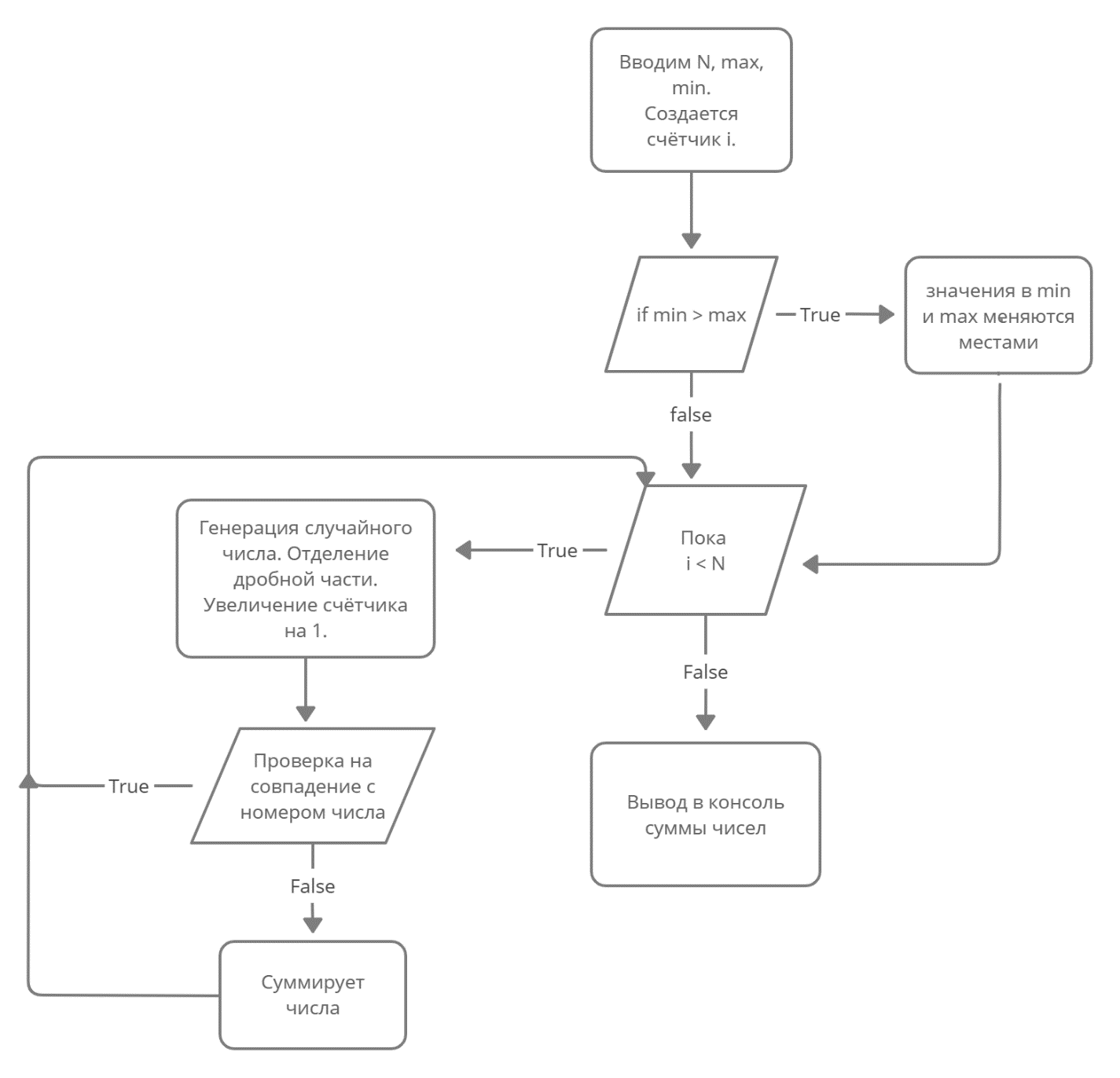


Рисунок 5. Схематическое представление программы.

# Эксперименты

Написанную программу протестируем на различных входных данных, чтобы убедится в её работоспособности в различных условиях.

1. В случае ввода некорректных данных, таких как отрицательное количество элементов (см. рис.6).

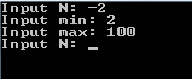


Рисунок 6. Неправильный ввод N.

Программа будет запрашивать ввести значение N до тех пор, пока оно не будет больше 0.

1. В случае неправильного ввода нижней границы генерации чисел min и верхней max (когда min > max) программа просто поменяет эти значения местами. (см. рис. 7)



Рисунок 7. Смена значений min и max и вывод результата.

Такой же результат мы бы получили при корректном вводе данных.

(см. рис. 8)



Рисунок 8. Корректный ввод.

При тестировании программы с различными входными данными в среде разработки Visual Studio никаких ошибок или предупреждений не выявлено. Программа даже для больших входных данных выполнялась за малое время.

1. **Заключение**

В ходе лабораторной работы мною была написана программа на языке «С», которая генерирует множество случайных чисел размера N в диапазоне (min, max), где N, min, max вводятся с клавиатуры. После чего подсчитывает, выводит сумму, которая получается следующим образом: все числа, номера которых совпадают с дробной частью одного из исходных чисел - вычитаются, все остальные прибавляются.

Выполнение данной лабораторной работы позволило мне понять принципы работы c циклами в языке «С», повысить уровень понимания при работе с различными алгоритмами, принципы работы со случайными числами, принципы работы с вводимыми пользователем данными их реализации внутри программы, их интерпретацию и проверку на корректность.

Выполнение подобных работ позволяет значительно улучшить свои навыки программирования, а также улучшить понимание работы с компьютером и средой разработки, в тоже время позволяет закрепить получаемые знания в области программирования и отработать их применения на практике.

1. Список литературы
2. «Язык программирования Си» — книга Брайана Кернигана и Денниса Ритчи.
3. C/C++. Программирование на языке высокого уровня / Т. А. Павловская. — СПб.: Питер, 2003. —461 с: ил.
4. Сборник задач по программированию/Д.М. Златопольский — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2011. — 304 с.
5. **Приложение**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | #include <stdio.h> | |  | #include <stdlib.h> | |  | #include<math.h> | |  |  | |  | int main() | |  | { | |  | int i; | |  | double max, min; | |  | double a = 0.0, fractional\_part; | |  | double num, sum = 0.0; | |  | int N = 0; | |  | double\* mas, \* mas1; | |  |  | |  | do | |  | { | |  | printf("Input N: "); | |  | scanf("%d", &N); | |  | printf("Input min: "); | |  | scanf("%lf", &min); | |  | printf("Input max: "); | |  | scanf("%lf", &max); | |  | } | |  |  | |  | while (min > max || N <= 0); | |  |  | |  |  | |  | mas1 = (double\*)malloc(N \* sizeof(double)); | |  | mas = (double\*)malloc(N \* sizeof(double)); | |  |  | |  | for (i = 0; i < N; i++) | |  | { | |  | mas[i] = (double)rand() / RAND\_MAX \* (max - min) + min; | |  |  | |  | fractional\_part = modf(a, &num); | |  |  | |  | mas1[i] = fractional\_part; | |  | } | |  |  | |  | for (i = 0; i < N; i++) | |  | { | |  | if (i < 10 != fractional\_part \* 10) | |  | sum += mas[i]; | |  |  | |  | if (i < 100 != fractional\_part \* 100) | |  | sum += mas[i]; | |  |  | |  | if (i < 1000 != fractional\_part \* 1000) | |  | sum += mas[i]; | |  |  | |  | if (i < 10000 != fractional\_part \* 10000) | |  | sum += mas[i]; | |  |  | |  | if (i < 100000 != fractional\_part \* 100000) | |  | sum += mas[i]; | |  | } | |  |  | |  | { | |  | if (i < 10 == fractional\_part \* 10) | |  | sum -= mas[i]; | |  |  | |  | if (i < 100 == fractional\_part \* 100) | |  | sum -= mas[i]; | |  |  | |  | if (i < 1000 == fractional\_part \* 1000) | |  | sum -= mas[i]; | |  |  | |  | if (i < 10000 == fractional\_part \* 10000) | |  | sum -= mas[i]; | |  |  | |  | if (i < 100000 == fractional\_part \* 100000) | |  | sum -= mas[i]; | |  | } | |  |  | |  | printf("SUM:%lf\n", sum); | |  |  | |  | free(mas); | |  | free(mas1); | |  |  | |  |  | |  | return 0; | |  | } | |