Федеральное Государственное Автономное Образовательное Учреждение Высшего Образования «Национальный Исследовательский Нижегородский Государственный Университет им. Н. И. Лобачевского».

Институт информационных технологий математики и механики.

Фундаментальная информатика и информационные технологии.

Отчёт по лабораторной работе

Генерация случайных чисел в массиве.

Суммирование членов массива.

Выполнила:

студентка ф-та ИИТММ ФИИТ 2 группа

Тюлькина О. В.

Проверил:

ассистент кафедры математического обеспечения и суперкомпьютерных технологий ИИТММ

Лебедев И. Г.

Нижний Новгород

2019 г.

Содержание

[1. Введение 2](#_Toc25058630)

[2. Постановка задачи 3](#_Toc25058631)

[3. Руководство пользователя 4](#_Toc25058632)

[4. Руководство программиста 5](#_Toc25058633)

[4.1. Описание структур данных 6](#_Toc25058634)

[4.2. Описание алгоритмов 6](#_Toc25058635)

[4.3. Описание структуры программы 6](#_Toc25058636)

[5. Эксперименты 7](#_Toc25058637)

[6. Заключение 10](#_Toc25058638)

[7. Литература 11](#_Toc25058639)

[8. Приложение 12](#_Toc25058640)

## 1. Введение

Всем известно, что программирование невозможно представить без математики. Математика развивает абстрактное мышление, применять разные действия, учит понимать и ставить задачу, анализировать возможные решения, решать задачи. В нашей жизни математика присутствует везде. Ежедневно мы производим огромное количество математических операций. Например, такое математическое действие как суммирование, мы используем постоянно. Оно помогает нам объединять разные элементы и рассматривать их как совокупность объектов. Как и в жизни, в программировании невозможно обойтись без операции суммирования. Актуальность этого математического действия никогда не исчезнет, ведь без него невозможно производить даже простейшие вычисления, не говоря уже о создании каких-либо сложных математических моделей. В моей работе также встретится операция суммирования, которая поможет нам получить верное решение.

## 2. Постановка задачи

Выполнить задание:

«Программа генерирует множество случайных чисел размера n в диапазоне (min, max), где n, min и 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.»

Вывести все данные и результат на экран.

## 3. Руководство пользователя

1) Ввести целое количество случайных чисел – n.

2) Задать определённый диапазон случайных чисел, для этого нужно ввести max и min значения с клавиатуры. Значения должны быть целыми числами. (max>min).



Рисунок 1. Ввод данных.

3) Нажать кнопку Enter и посмотреть результат.

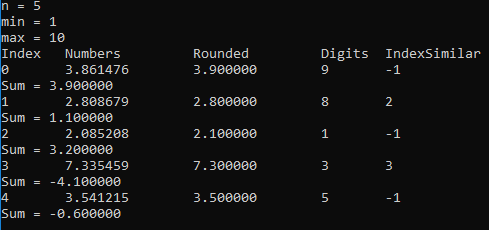


Рисунок 2. Вывод результата на экран

## 4. Руководство программиста

## 4.1. Описание структур данных

*Целочисленный тип данных int* **-** служит для представления целых чисел. Все целочисленные типы поддерживают арифметические операторы, побитовые логические операторы, операторы сравнения и равенства**.**

*Вещественный тип данных double* — это тип данных, который используется для переменных, содержащих числа с дробной частью. Может представлять дробные и целые значения длинной до 15 знаков.

## 

## 4.2. Описание алгоритмов

Суммирование – сложение переменных.

*Округление до ближайшего целого:*

Функции округляют аргумент x до ближайшего целого числа. Если округляемый аргумент отстоит от наибольшего и наименьшего целого на одну и ту же величину, то округления произведется до ближайшего большего целого числа. Аргумент и возвращаемое значение функций являются значениями с плавающей точкой.

*Округление до ближайшего дробного числа* - округление значения с плавающей запятой двойной точности до указанного числа знаков после запятой.

*Умножение*  - Умножить одно целое число на другое значит повторить одно число столько раз, сколько в другом содержится единиц.

*Логическое* И - возвращает истину тогда и только тогда, когда оба аргумента являются истиной.

*Логическое* ИЛИ - истинен тогда, когда истиной является хотя бы один его аргумент.

*Возведение в степень* - бинарная операция, результат многократного умножения числа на самого себя.

## 4.3. Описание структуры программы

*Функция main* вызывается при старте программы после инициализации нелокальных объектов со статической длительностью хранения. Это точка входа в программу.

*Функция pow* возводит аргумент x в степень y.

Алгоритм работы: ввод целых чисел, создание массивов, генерация случайных чисел, округление до десятичной части, сравнение порядкового номера с цифрой после запятой, вывод суммы.

## 5. Эксперименты

Эксперимент №1.

Введем данные следующим образом: min <n <max. В результате мы получили 6 чисел в диапазоне от 0 до 12. После округления их дробные части сравниваются с порядковым номером. В трех числах из 10 совпадающие порядковые номера не были найдены, следовательно, появился индекс «-1». В итоге, после последовательного сложения 3 положительных и 3 отрицательных чисел мы получили положительную сумму, равную 9,3.

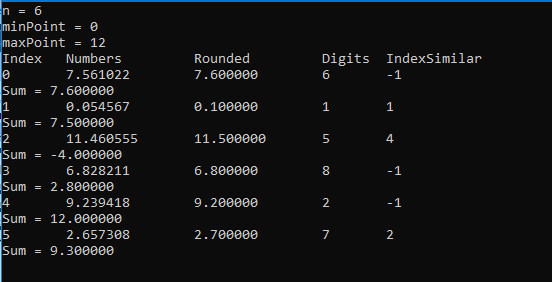


Рисунок 3. Эксперимент №1.

Эксперимент №2.

Введем данные следующим образом: min <max <n.

Получили 7 чисел в диапазоне от 0 до 1. После округления их дробные части сравниваются с порядковым номером. В 4 числах из 7 совпадающие порядковые номера не были найдены, следовательно, появился индекс «-1». В итоге, после последовательного сложения 4 положительных и 3 отрицательных чисел мы получили положительную сумму, равную 0.8.

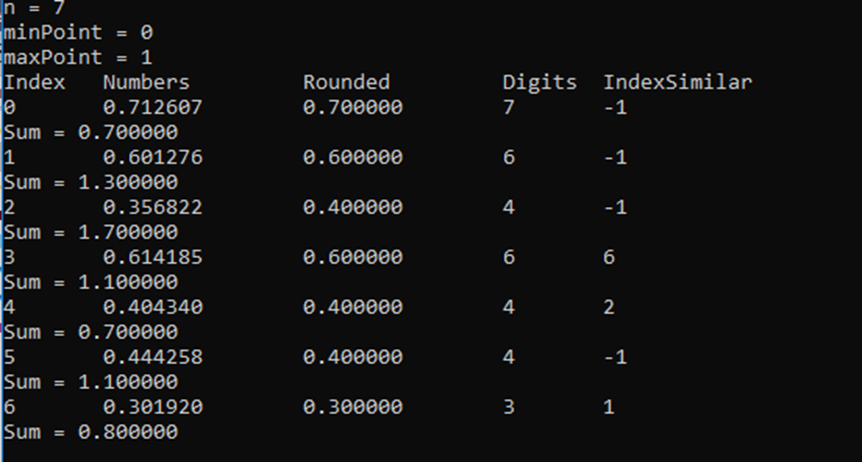


Рисунок 4. Эксперимент № 2.

## 6. Заключение

Я написала программу, позволяющую вычислить сумму случайных чисел в зависимости от соответствия их порядковых номеров с дробными частями. Эта программа содержит в себе множество математических действий, включая суммирование, без которого эту программу невозможно было бы выполнить. Это еще раз доказывает важность математики и математических операций в программировании.

## 7. Литература

1. Брайан У. Керниган. Язык программирования Си / Деннис М. Ритчи – М.: Москва: Вильямс, 2015. — 304 с.

2. М. Бен-Ари. Языки программирования. Практический сравнительный анализ. / М. Бен-Ари. — М.: Москва: Мир, 2000. — 366 с.

3. Подбельский, В. В. Курс программирования на языке Си: учебник/ С. С. Фомин, В. В. Подбельский, - М.: ДМК Пресс, 2012. — 318 с.

4. Прата С. Язык программирования С: Лекции и упражнения / С. Прата — М.: Вильямс, 2006. — С. 960.

## 8. Приложение

код программы

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <time.h>

int main()

{

int n, minPoint, maxPoint;

double S = 0.;

srand(time(NULL));

printf("n = ");

scanf\_s("%d", &n);

printf("minPoint = ");

scanf\_s("%d", &minPoint);

printf("maxPoint = ");

scanf\_s("%d", &maxPoint);

double\* r1 = new double[n];

double\* r2 = new double[n];

int\* digits = new int[n];

int\* fountNum = new int[n];

int numDigitsAfterComa;

numDigitsAfterComa = int(round(log10(double(n))));

if (numDigitsAfterComa > 3)

{

numDigitsAfterComa = 3;

}

printf("Index\tNumbers\t\tRounded\t\tDigits\tIndexSimilar\n");

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

{

r1[i] = double(rand()) / double(RAND\_MAX) \* double(maxPoint - minPoint) + double(minPoint);

r2[i] = round(r1[i] \* pow(10., double(numDigitsAfterComa))) / pow(10., double(numDigitsAfterComa));

digits[i] = int((r2[i] - double(int(r2[i])) + pow(10., -double(numDigitsAfterComa + 1))) \* pow(10, double(numDigitsAfterComa)));

}

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

{

printf("%d\t", i);

printf("%lf\t", r1[i]);

printf("%lf\t", r2[i]);

printf("%d\t", digits[i]);

bool isFound = false;

j = 0;

while (j < n && isFound == false)

{

isFound = isFound || (i == digits[j]);

j++;

}

if (isFound)

{

fountNum[i] = j - 1;

S = S - r2[i];

}

else

{

fountNum[i] = -1;

S = S + r2[i];

}

printf("%d\n", fountNum[i]);

printf("Sum = %lf\n", S);

}

delete r1, r2, digits, fountNum;

getchar();

getchar();

}