Национальный исследовательский

Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

Отчёт по лабораторной работе

Суммирование по определенному принципу n элементов

Выполнил:

студент ф-та ИИТММ гр. 381906-2

Сорокина Е.В.

Проверил:

преподаватель каф. МОСТ, ИИТММ

Лебедев И.Г.

Нижний Новгород

2019 г.

**Содержание**

[Введение 3](#_Toc25040516)

[Постановка задачи 4](#_Toc25040517)

[Руководство пользователя 5](#_Toc25040518)

[Руководство программиста 6](#_Toc25040519)

[Описание структуры программы 6](#_Toc25040520)

[Описание структур данных 6](#_Toc25040521)

[Описание алгоритмов 6](#_Toc25040522)

[Эксперименты 7](#_Toc25040523)

[Заключение 8](#_Toc25040524)

[Литература 9](#_Toc25040525)

[Приложения 10](#_Toc25040526)

[Приложение 1 10](#_Toc25040527)

[Приложение 2 14](#_Toc25040528)

# Введение

Суммирование является одной из основных математических операций, которой мы пользуемся в практически любых отраслях и разделах наук.

Это алгебраическое действие встречается нам во многих разнообразных задачах: в постройке домов, при проектировании и постройки ракеты, машины, при пошиве одежды, для приготовления блюд, для выращивания животных, для изготовления лекарств и во многих других сферах деятельности. Суммирование (или как его ещё называют – сложение) сопровождает нас буквально всюду. Эта повседневность и распространённость и приводят к осознанию того, что математика, включающая суммирование, – важная часть человеческой культуры.

Значение суммирования нельзя не оценить, так как эта операция возникла ещё в древних времен, заложив начало развития математики и всех остальных наук, перекликающихся с ней. Однако по мере развития человечества способы суммирования совершенствовались и упрощались, возникли специальные устройства – калькуляторы – основное назначение которых вычисления различных величин. В настоящее время с появлением ЭВМ калькуляторы есть в любом компьютере, ноутбуке, смартфоне, планшете и т.д. А любой человек, хотя бы немного знакомый с программированием, способен написать код, производящий арифметические операции по заранее заданному алгоритму. Именно подобную задачу, а именно суммирование n элементов по строго определённому принципу, мы и рассмотрим в данной работе.

# Постановка задачи

Программа генерирует 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 предполагается для использования, в первую очередь, в рамках основной задачи, описанной в предыдущем разделе отчёта. Однако помимо этого данную программу можно использовать также для выведения n элементов в заданном диапазоне, для выделения дробной части числа равной (по длине) количеству элементов массива, для выявления совпадений между индексом (номером) элемента и дробной части каждого из всех чисел массива. Руководство программиста

## Описание структуры программы

Данная программа состоит из единственного модуля, основное и единственное назначение которого – это выполнение поставленной задачи. Код программы написан на языке C, что исключает использование классов. Модуль программы включает в себя такие функции, как void main, содержащую код программы, pow, возводящую число в степень, и modf, отделяющую целую часть числа от дробной.

Алгоритм работы программы построен следующим образом: сначала идёт ввод запрашиваемых данных (min, max, n) и проверка их достоверности (например, из соображений логики n не может иметь отрицательных значений); далее следует определение размера (длины) числа n, на основе чего потом будет происходить выделение дробной части числа; далее следуют несколько циклов, во время которых происходит отделение дробной части числа каждого из элементов, с которой после идёт поиск совпадений с номером каждого элемента; в конце на основе данных о совпадении (или несовпадении) номера элемента с дробной частью каждого числа происходит суммирование этого элемента по принципу, описанному в поставленной задаче.

## Описание структур данных

Код данной программы, во-первых, как написанный на языке C, не может содержать классы, а , во-вторых, написан таким образом, чтобы обойтись без использования структур и глобальных переменных.

## Описание алгоритмов

В данной программе применяются такие алгоритмы, состоящие из циклов, как : подсчёт количества цифр (определение длины) числа n, выделение дробной части каждого элемента параллельно записи её в соответствующую ячейку массива для дробных частей и записи номера каждого элемента в соответствующую ячейку массива для индексов элементов массива, поиск совпадений между индексом и дробной частью каждого элемента и запись их количества в соответствующую ячейку массива для количества совпадений, суммирование элементов на основе данных из массива о совпадении.

# Эксперименты

Эксперимент 1 : проверка работы программы при отрицательных значениях n. Результат : «neverno vvedeni znachenia - ispravte i povtorite popitky» . Программа определила, что вывод отрицательного количества элементов является логически неверной операцией.

Эксперимент 2: проверка работы программы при значениях max < min. Результат: «neverno vvedeni znachenia - ispravte i povtorite popitky». Программа определила, что диапазон данных задан логически неверно.

Эксперимент 3: проверка работы программы при значения n =0. Результат: «Summa=0». Программа определила, что при выводе нулевого количества элементов нечего суммировать, то есть их сумма равна 0.

Эксперимент 4: проверка работы программы при диапазоне min=max. Результат: все элементы имеют одинаковое значение равное ведённым концам заданного диапазона.

Эксперимент 5: время работы программы при количестве элементов n=10000. Результат: 8,21с. Большая часть времени тратится на вывод элементов на экран.

# Заключение

Нам удалось достичь таких результатов, как выведения n элементов в заданном диапазоне, для выделения дробной части числа равной (по длине) количеству элементов массива, для выявления совпадений между индексом (номером) элемента и дробной части каждого из всех чисел массива. И, наконец, самое главное – нам удалось написать код, выполняющий поставленную задачу.

# Литература

1. Function printf. Parameters - [http://www.cplusplus.com/reference/cstdio/printf/]
2. Описание функций языка СИ. Pow, powf, powl – возведение в степень - [http://all-ht.ru/inf/prog/c/func/pow,powf,powl.html]
3. Описание функций языка СИ. Modf, modff, modfl – разделение числа на дробную и целую части - [http://all-ht.ru/inf/prog/c/func/modf,modff,modfl.html]

# Приложения

## Приложение 1

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <malloc.h>

void main()

{

double min, max;

int elem1;

double elem2;

int i=0,j=0;

int n=0,N=0;

int kolvo\_cifr = 0;

double drob\_elem, cel\_elem, drob\_elem1, drob\_elem2, cel\_elem2;

int prov = 0;

int lishnii\_nol = 0;

double\* massiv;

double\* massiv\_drob;

double\* ind\_mas;

int\* sovp;

double Summa = 0;

printf("vvedite min znachenie dlya diapozona :");

scanf\_s("%lf", &min);

printf("vvedite max znachenie dlya diapozona :");

scanf\_s("%lf", &max);

printf("vvedite kol-vo elementov :");

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

if ((n < 0) || (max < min))

{

printf("neverno vvedeni znachenia - ispravte i povtorite popitky\n");

}

else

{

N = n;

while (N > 0)

{

N = N / 10;

kolvo\_cifr = kolvo\_cifr + 1;

}

massiv = (double\*)malloc(n \* sizeof(double));

massiv\_drob = (double\*)malloc(n \* sizeof(double));

ind\_mas = (double\*)malloc(n \* sizeof(double));

sovp = (int\*)malloc(n \* sizeof(int));

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

{

elem1 = rand();

elem2 = (((double)elem1) / RAND\_MAX) \* (max - min) + min;

ind\_mas[i] = i;

if (elem2 < 0)

{

elem2 = elem2 \* (-1);

drob\_elem = modf(elem2, &cel\_elem);

drob\_elem1 = drob\_elem \* pow(10, kolvo\_cifr);

drob\_elem2 = modf(drob\_elem1, &cel\_elem2);

prov = cel\_elem2;

if (prov != 0)

{

if (prov % 10 == 0)

{

while (prov % 10 == 0)

{

lishnii\_nol = lishnii\_nol + 1;

prov = prov / 10;

}

}

}

massiv[i] = (-1) \* (cel\_elem + prov \* (1 / pow(10, kolvo\_cifr)));

}

else

{

drob\_elem = modf(elem2, &cel\_elem);

drob\_elem1 = drob\_elem \* pow(10, kolvo\_cifr);

drob\_elem2 = modf(drob\_elem1, &cel\_elem2);

prov = cel\_elem2;

if (prov != 0)

{

if (prov % 10 == 0)

{

while (prov % 10 == 0)

{

lishnii\_nol = lishnii\_nol + 1;

prov = prov / 10;

}

}

}

massiv[i] = cel\_elem + prov \* (1 / pow(10, kolvo\_cifr));

}

if (prov == 0)

{

printf("massive[%d] = %0.1lf\n", i, massiv[i]);//вместо, например, 17 или 17.000000 пишет 17.0

}

else

{

printf("massive[%d] = %g\n", i, massiv[i]);//%g отбрасывает лишние нули в дробной части числа, например, 17.123000 -> 17.123

}

massiv\_drob[i] = prov;

}

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

{

sovp[i] = 0;

for (j = 0; j < n; j++)

{

if (ind\_mas[i] == massiv\_drob[j])

{

sovp[i] = sovp[i] + 1;

}

}

printf("sovp[%d] - %d\n", i, sovp[i]);//показывает число совпадений индексов элементов с их дробной частью

}

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

{

if (sovp[i] != 0)

Summa = Summa - massiv[i];

else

Summa = Summa + massiv[i];

}

printf("Summa = %g\n", Summa);

}

};

## Приложение 2