МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Физико-технический факультет

Кафедра вычислительной техники и электроники (ВТиЭ)

Лабораторная работа № 003

Выполнил студент 595 гр.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Лаптев

Проверил: к.т.н,, доцент каф. ВТиЭ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ю.Г. Скурыдин

Лабораторная работа защищена

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Барнаул 2019

**ЗАДАНИЕ №1.**

**1. Формулировка задачи**

Дано 20 вещественных чисел. Создать программу для вычисления разности между максимальным и минимальным числами.

**2. Постановка задачи**

На вход программы поступают 20 вещественных чисел. Среди этих чисел находится наибольшее и наименьшее, а затем вычисляется их разность. На выход программы поступает вещественное число, которое является разностью наибольшего и наименьшего чисел.

**3. Математическая модель**

Математическую модель данной программы можно рассмотреть на следующем примере (пример реальной работы программы): генерируется первое случайное число: -128.778, значение первого числа принимается одновременно за максимальное и минимальное. Затем происходит последовательное попарное сравнение полученных в дальнейшем чисел: 390.712, -60.0502, -264.676, -138.859, 191.666, 27.7769, -392.678, 258.397, -257.768, -423.181, -10.2465, -302.654, 404.153, -52.5042, -335.427, -164.144, 264.736, -402.043, 432.193- с первым числом. Если находится число, меньшее данного, то этому числу присваивается значение min, а если находится число большее первого, то такому числу присваивается значение max и потом последовательное сравнение оставшихся чисел происходит уже не с первым числовым значением, а с тем, которое его заменило и стало наибольшим/наименьшим. Далее, если, вдруг, опять находится числовое значение, которое меньше, чем полученное в ходе предыдущего сравнения, то уже последнее значение становится минимальным и таким же образом для максимального значения. В данном примере: -128.778 сравнивается с 390.712, второе значение больше первого, поэтому ему присваивается значение max. Следующее значение: -60.0502 сравнивается как с -128.778 (min), так и с 390.712 (max). Сравниваемое число не превосходит 390.712 и не является меньшим, чем -128.778, поэтому оно пропускается. Потом -264.676 также сравнивается с теми же числами, которые были использованы в предыдущем сравнении, и оно является меньшим, чем -128.778 (которому присвоено значение min), поэтому ему присваивается значение min и т.д. И в ответе будет вычислена разность конечных чисел (в данном случае: 432.193 – (-423.181) = 855.374).

**4. Описание алгоритма**

Начало

4.1 Генерация случайного числа, которому присваивается значение max

4.2 Присвоение переменной min значения, равного max

4.3 Вывод на экран значения max

4.4 Присвоение переменной-счётчику (i) значения 2 (для генерации случайных чисел, начиная со второго до двадцатого)

4.5 Проверка условия (i <= 20). Если условие не выполняется, то осуществляется выход из цикла и переход к пункту 4.13

4.6 i++

4.7 Генерация (i-1) числа

4.8 Вывод (i-1) числа на экран

4.9 Проверка условия, при котором будет осуществлён поиск наибольшего значения из данных чисел, путём сравнения каждого из чисел с числом, имеющим максимальное значение. Если условие не выполняется, то происходит переход к пункту 4.11

4.10 Наибольшему числу присваивается значение max

4.11 Проверка условия, при котором будет найдено наименьшее значение из данных, путём сравнения каждого из чисел с числом, имеющим минимальное значение. Если условие не выполняется, то происходит переход к пункту 4.5

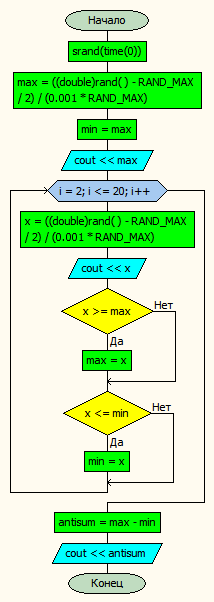
4.12 Наименьшему числу присваивается значение min. Осуществляется переход к пункту 4.5

4.13 Вычисление разности между максимальным и минимальным значениями

4.14 Вывод получившейся разности в качестве ответа

Конец

**5. Опорный граф (блок-схема) алгоритма**



**6. Проект программы с определением замкнутых программных единиц и необходимых структур данных**

В данном случае программа должна состоять из трех частей – блок ввода, блок вывода и блок преобразований, который включает в себя цикл по генерации 20 случайных вещественных чисел и проверку условий, для нахождения максимального и минимального из этих чисел.

**7. Текст программы:**

**на языке С++**

#include <iostream>;

#include <math.h>;

#include <ctime>;

#include <stdlib.h>;

using namespace std;

int main()

{

int i;

double x, min, max, antisum = 0;

setlocale(0, "");

srand(time(0));

cout << "Введите 20 вещественных чисел" << endl;

max = ((double)rand() - RAND\_MAX / 2) / (0.001 \* RAND\_MAX); // Генерация первого случайного числа, которому будет присвоено значение max

min = max;

cout << max << endl;

for (i = 2; i <= 20; i++) // Цикл для ввода оставшихся 18 случайных вещественных чисел

{

x = ((double)rand() - RAND\_MAX / 2) / (0.001 \* RAND\_MAX);

cout << x << endl; // Вывод этих чисел на экран

if (x >= max) // Условие для определения наибольшего числа из данных

max = x;

if (x <= min) // Условие для определения наименьшего из данных чисел

min = x;

}

antisum = max - min; // Вычисление разности между этими числами

cout << "Разность наибольшего и наименьшего чисел: " << antisum << endl; // Вывод разности на экран

return 0;

}

**на языке Pascal**

**Program** lab\_3\_1;

**Var** i : integer;

a, b, x, max, min, antisum : real;

**Begin**

Writeln ('Введите двадцать вещественных чисел');

randomize;

a := -9999999999;

b := 9999999999;

antisum := 0;

max := random (a, b);

min := max;

Writeln(max);

**for** i := 2 **to** 20 **do** // Цикл для ввода оставшихся 18 случайных вещественных чисел

**begin**

x := random (a, b);

Writeln (x); // Вывод этих чисел на экран

**if** x >= max **then** // Условие, при котором среди этих чисел находится наибольшее

max := x;

**if** x <= min **then** // Условие, при котором находится наименьшее из данных чисел

min := x;

**end**;

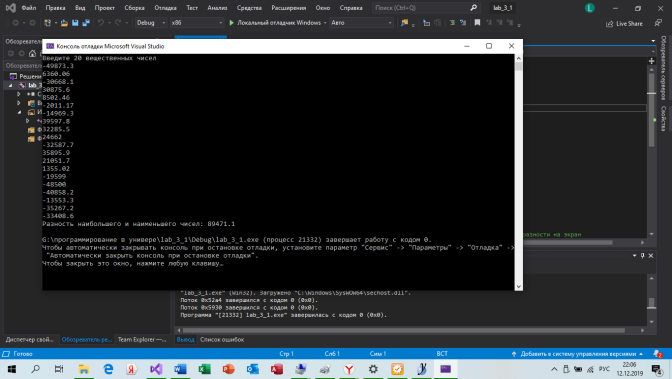
antisum := max - min; // Разница между наибольшим и наименьшим числами

Writeln('Разность между наибольшим и наименьшим числами: ',antisum); // Вывод разности этих чисел на экран

**End**.

**8. Проверка работоспособности (тестирование) программы**

Для проверки работоспособности программы приведён скриншот после её выполнения (на С++).



**9. Сравнительный анализ и оценка эффективности работы программ на разных языках программирования**

Ввиду своей простоты программа одинаково быстро выполняется на Pascal и C++.

**ЗАДАНИЕ №2.**

**1. Формулировка задачи**

Для числового ряда, общий член которого имеет вид , найти сумму тех членов ряда, для которых , где значение вводится с клавиатуры.

**2. Постановка задачи**

На вход программы поступает вещественное положительное число < 1, которым задаётся степень точности суммы членов числового ряда. Далее, программа вычисляет сумму элементов сходящегося числового ряда с заданной точностью. На выходе программы вещественным положительным числом выводится конечная сумма членов числового ряда, с учётом степени точности.

**3. Математическая модель**

Математическую модель данной задачи можно рассмотреть на следующем примере. На вход программы поступает число = 0.01, которое определяет степень точности конечной суммы членов сходящегося числового ряда. Далее вручную вычисляется значение первого члена последовательности. В данном случае: a1 = 1, которое больше заданной степени точности. Сумма членов становится равна 1. Затем создаётся цикл, в котором вычисляется n-ые члены последовательности согласно формуле: . Потом найденное число прибавляется к ранее найденной сумме членов последовательности и происходит увеличение порядкового номера члена, который будет найден во время следующей итерации цикла, на 1. В данном примере: а2 = -1/4, оно больше по модулю, чем , значит значение а2 прибавляется к общей сумме. Сумма становится равной 0,75. Происходит вычисление члена а3: а3 = 1/27, это значение по модулю больше, чем , значит значение а3 также прибавляется к общей сумме. И сумма становится равной 0.787037037. Происходит вычисление члена а4: а4 = -1/256, это значение уже меньше по модулю, чем , поэтому оно не прибавляется к сумме членов ряда. После того как найдена сумма всех членов последовательности, она округляется до степени точности, которая введена с клавиатуры и равна . В данном примере: 0.787037037 округляется до 2 знаков после запятой: 0.78. И на выходе программы полученное число выводится в качестве ответа.

**4. Описание алгоритма**

Начало

4.1 Ввод вещественного числа ( < 1)

4.2 Присвоение порядковому номеру элемента значения 1 (n = 1)

4.3 Присвоение первому члену последовательности значения 1 (an = 1), а начальной сумме значения 1 (sum = 1)

Начало цикла

4.4 Проверка условия . Если условие не выполняется, то переход к пункту 4.8

4.5 Увеличение порядкового номера члена данной последовательности (n++)

4.6 Вычисление n-ого члена данной последовательности по формуле:

4.7 Суммирование вычисленного значения с общей суммой членов последовательности, полученной ранее и переход к пункту 4.4

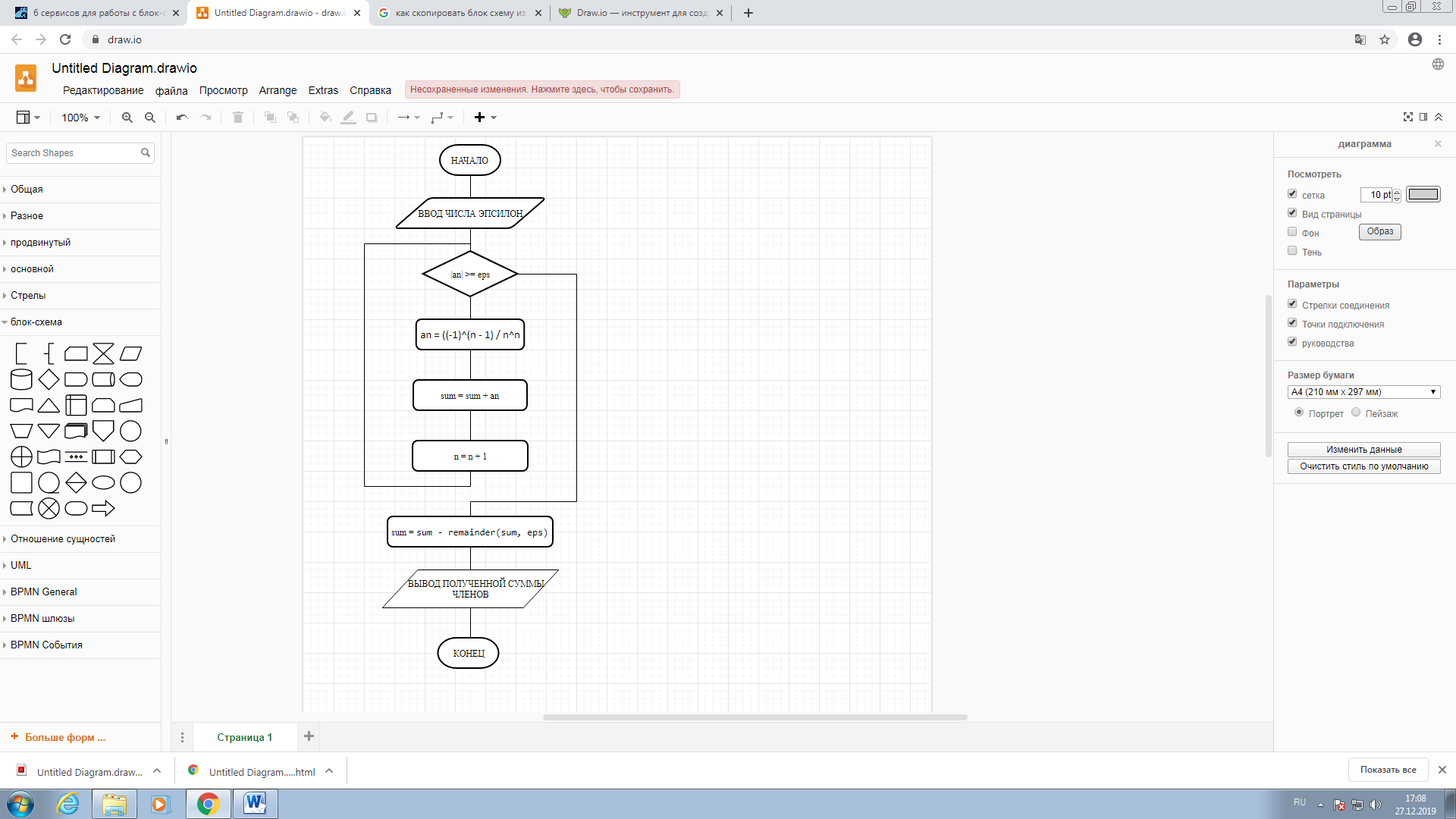
Окончание работы цикла

4.8 Процесс округления полученной суммы до степени точности, равной

4.9 Вывод полученной суммы на экран

Конец

**5. Опорный граф (блок-схема) алгоритма**



**6. Проект программы с определением замкнутых программных единиц и необходимых структур данных**

В данном случае программа должна состоять из трех частей – блок ввода, блок вывода и блок преобразований, включающий цикл. В качестве структурных данных могут быть введены переменные для хранения исходного числа и остальных необходимых переменных.

**7. Текст программы:**

**на языке С++**

#include<iostream>;

#include<math.h>;

using namespace std;

int main()

{

int n = 1;

double eps, an = 1, sum = 0;

cout << "Введите значение эпсилон" << endl;

cin >> eps; // Ввод вещественного числа эпсилон

while (abs(an) >= eps) // Цикл, который отвечает за вычисление общей суммы всех членов последовательности

{

an = pow((-1), n - 1) / pow(n, n); // Вычисление n-ого члена последовательности

sum = sum + an; // Прибавление найденного члена к общей сумме членов, найденной ранее

n = n + 1; // Увеличение порядкового номера члена последовательности, который будет найден в следующей итерации

}

cout << "sum = " << (sum - remainder(sum, eps)) << endl; // Округление полученной суммы и вывод её на экран

return 0;

}

**на языке Pascal**

**Var** an, sum, eps, k: real;

n, i:integer;

**Begin**

Writeln ('Введите значение эпсилон');

Readln (eps); // Ввод числа эпсилон

sum := 0; // Инициализвация переменных

an := 1;

n := 1;

**for** i := 1 **to** 40 **do** // Цикл, который отвечает за определение количества знаков после запятой в данном числе эпсилон, чтобы в дальнейшем,

**begin** // чтобы в дальнейшем округлить полученную сумму членов последовательности до необходимой точности

k := round(eps, i); // Определение количества знаков после запятой в числе k, путём добавления к числу 1 знака после запятой в каждой итерации цикла

**if** (k = eps) **then** // Условие, при котором будет найдено точное количество знаков после запятой

**break**; // (т.е. когда число знаков после запятой в числе k будет равно количеству знаков после запятой в числе эпсилон)и осуществится выход из цикла

**end**;

**while** abs(an) >= eps **do** // Цикл, который отвечает за вычисление общей суммы всех членов последовательности

**begin**

an := (power((-1), n-1)) / (power(n, n)); // Вычисление n-ого члена последовательности

sum := sum + an; // Прибавление найденного члена к общей сумме членов, найденной ранее

n := n + 1; // Увеличение порядкового номера члена последовательности, который будет найден в следующей итерации цикла

**end**;

Writeln ('sum = ', round(sum, i)); // Вывод округлённой суммы на экран

**End**.

**8. Проверка работоспособности (тестирование) программы**

Для проверки работоспособности по запросу (после запуска программы) введем число: = 0,0001. Тогда результатом выполнения программы будет число: 0.7834.

**9. Сравнительный анализ и оценка эффективности работы программ на разных языках программирования**

В данной задаче, программы написанные на разных языках программирования несколько отличны по структуре друг от друга. Например, в программе на С++ округление значения происходит уже в конце работы программы, а в Паскале нужен дополнительный цикл, для определения количества знаков после запятой, для округления полученной далее суммы, который прописывается ранее. Благодаря чему программа написанная на языке С++ будет выполняться быстрее и гораздо эффективнее, чем на Паскале.