**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Вариант №12

Курс «Основы информатики»

Отчет по лабораторной работе №3

«Программирование циклических алгоритмов. Расчёт по формулам»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-13Б |  | преподаватель каф. ИУ5 |
| Поляков Данила |  | Аксёнова М.В. |
| Подпись и дата 19.09.2021 |  | Подпись и дата: |

Москва, 2021 г.

**Постановка задачи**

1. Найдите сумму натуральных чисел, которые делятся на 5 и не делятся на m (m<n). Количество натуральных чисел *n* и значение m введите с клавиатуры.
2. Составьте программу для вычисления:



Значение *а* введите с клавиатуры**.**

1. Составьте программу вычисления значения суммы и функции *Y(x)* = *cos(2х-1) ,* где 0≤ *х* ≤1, с шагом *h=0.2.* Вычисление суммы ряда Тейлора производите с погрешностью, не превышающей 0.000001.



Результат представить в виде таблицы (без рамок), которая содержит четыре строки со значениями *x*, *Y(x), S(x) и N,* где *N* - номер последнего слагаемого.

4. Напишите программу для вычисления *у* по формуле:



Натуральное значение n введите с клавиатуры. Значение *х* также введите с клавиатуры. Обеспечьте возможность, не завершая программу, вычислить *y* для нескольких значений *n* и выведите на экран значения промежуточных (частичных) сумм при количестве слагаемых 3, 5 и 10.

**Разработка алгоритма**

**Описание используемых переменных**

**Для задачи №1:**

int n,m,sum;

n и m – вводимые значения, прописанные в условии

sum – переменная для хранения суммы чисел

**Для задачи №2:**

double a, product;

a – вводимое значение, необходимое в условии

product – переменная для хранения искомого произведения

**Для задачи №3:**

double y, s, s1, eps = 0.000001;

int n;

y – значение, заданное функцией

s – сумма ряда Тейлора

s1 – текущий член ряда

eps – эпсилон, использующийся для точности

n – номер члена ряда

**Для задачи №4:**

int n

double x,y,y1

n – номер члена расчётного ряда

y – сумма ряда

y1 – текущий член ряда

**Определение расчётного соотношения**

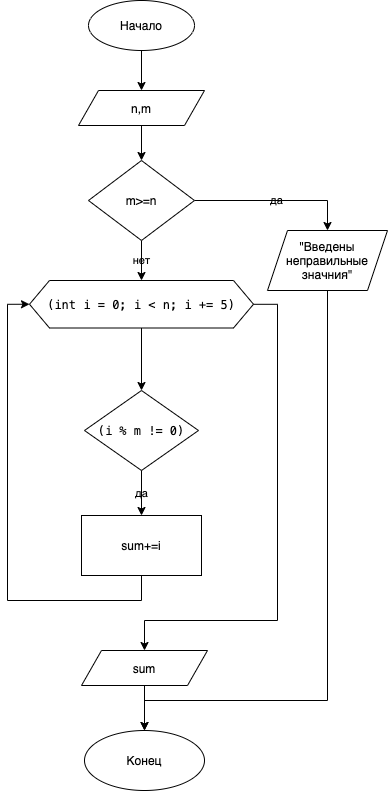
Для того, чтобы не считать каждый новый член ряда в 3 и 4 задании исходя из формулы, данной в условии, мною был выведен коэффициент (расчётное соотношение) для нахождения следующего члена ряда, позволяющее сократить нагрузку на процессор во время работы.

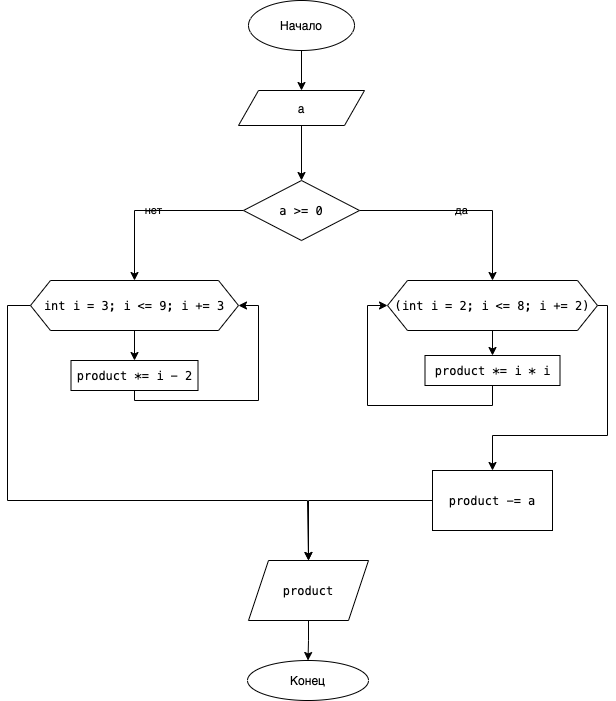
Чтобы вывести формулу этого соотношения, я вручную посчитал для каждого ряда 1,2 и 3 его члены, после чего выписал их соотношения. Полученные выражения можно было описать одной формулой, которая и будет являться коэффициентом, благодаря которому зная предыдущий член ряда можно было найти следующий, а сама эта формула регулировалась значением n (номер члена), так же в ней фигурировало значение n.

Для третьей задачи формула сводится к , а для четвёртой –

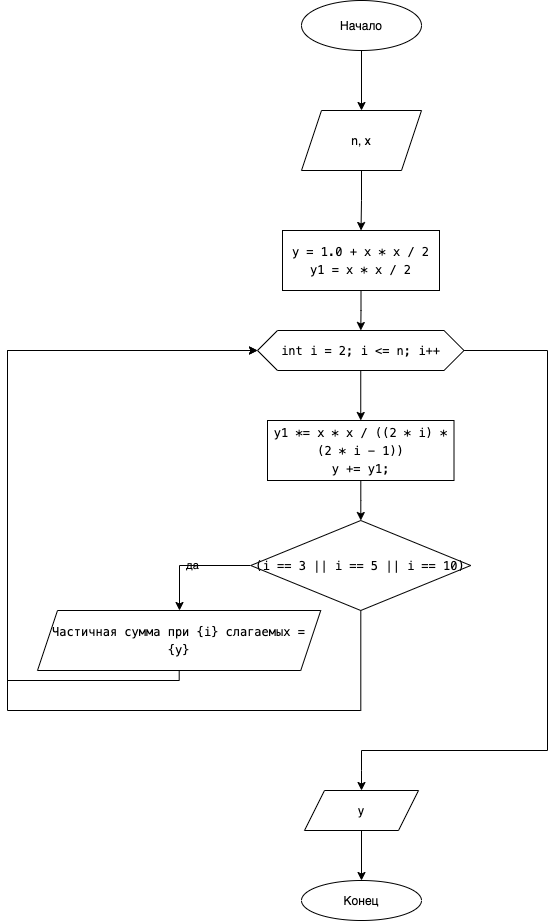
.

**Блок-схемы для каждой из задач:**

****

****

****



**Текст программы**

**Листинг кода программы:**

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

int main() {

int progNumber;

cout << "Enter needed programm number: ";

cin >> progNumber;

switch (progNumber) {

case 1: //task 1

{

bool resume = true;

int n, m, sum = 0;

while (resume) {

sum = 0;

//n - count of numbers

cout << "n: m: \n";

cin >> n >> m;

if (m >= n) {

cout << "incorrect values!" << endl;

break;

}

for (int i = 0; i < n; i += 5) {

if (i % m != 0) {

sum += i;

}

}

cout << sum;

cout << "\nDo you want to resume?(1/0): ";

cin >> resume;

}

}

break;

case 2: //task 2

{

double a, product = 1.0;

cout << "a: ";

cin >> a;

if (a >= 0) {

for (int i = 2; i <= 8; i += 2) {

product \*= i \* i;

}

product -= a;

} else {

for (int i = 3; i <= 9; i += 3) {

product \*= i - 2;

}

}

cout << fixed << setprecision(3) << product;

}

break;

// x: 0 k: 1

// x: 0.2 k: 1.05565

// x: 0.4 k: 2.20471

// x: 0.6 k: 15.8114

// x: 0.8 k: -45.8114

// x: 1 k: -31

case 3: {

{

cout << setw(3) << 'x' << setw(9) << 'y' << setw(8) << 's' << setw(4) << 'n' << endl;

double y, s, s1, eps = 0.000001;

int n;

for (double x = 0; x <= 1; x += 0.2) {

y = cos(2 \* x - 1);

s = 1.0 - pow(2 \* x, 2) / 2;

n = 1;

s1 = pow(2 \* x, 2) / 2;

while (abs(s1) > eps) {

n++;

s1 = s1 \* (-(4 \* x \* x / (2 \* n \* (2 \* n - 1))));

s += s1;

}

cout << setw(3) << x << '|' << setw(9) << y << '|' << setw(8) << s << '|' << setw(1) << n << endl;

}

}

}

break;

case 4: {

bool resume = true;

while (resume) {

int n = 0;

double x;

cout << "n: ";

cin >> n;

cout << '\n' << "x:";

cin >> x;

double y = 1.0 + x \* x / 2, y1 = x \* x / 2;

for (int i = 2; i <= n; i++) {

y1 \*= x \* x / ((2 \* i) \* (2 \* i - 1));

y += y1;

if (i == 3 || i == 5 || i == 10) {

cout << "Частичная сумма при " << i << " слагаемых = " << y << endl;

}

}

cout << "y: " << y << endl;

cout << "do you want to continue?(1/0)";

cin >> resume;

}

}

break;

default:

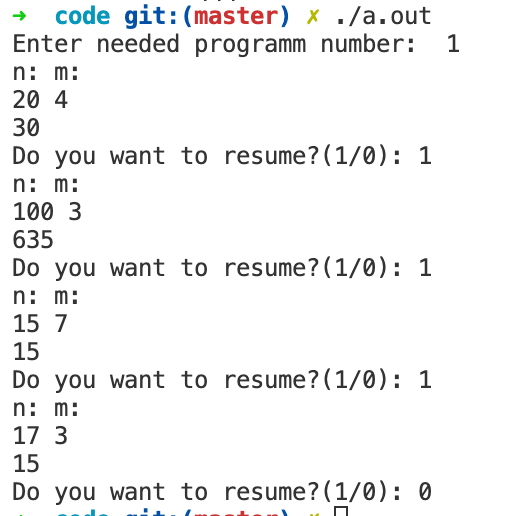
break;

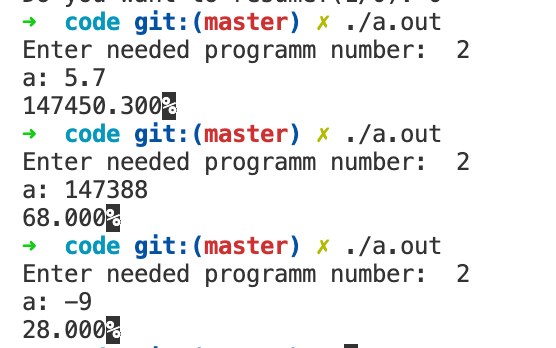
}

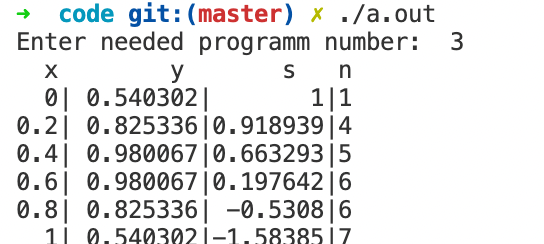
return 0;

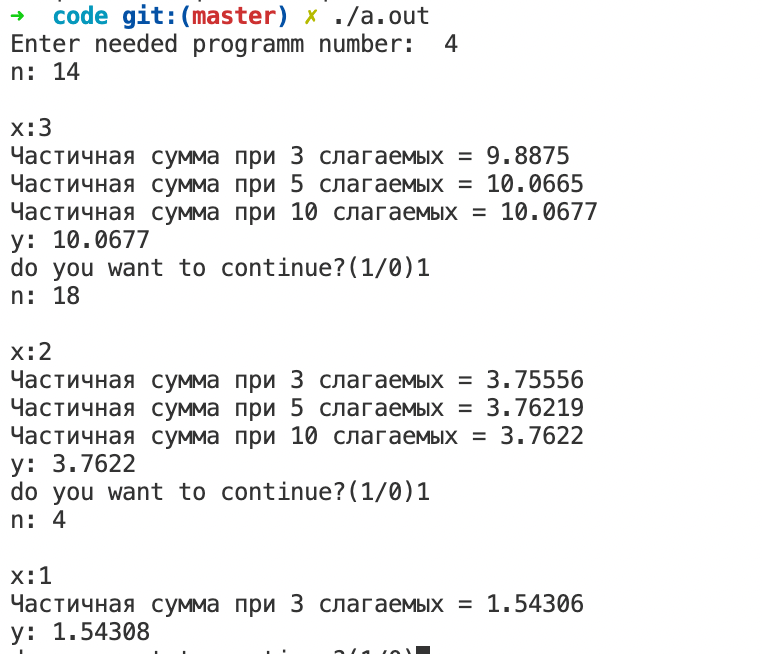
}

**Анализ результатов**

****

****

****

****