**Федеральное государственное автономное образовательное учреждение высшего образования   
"Национальный исследовательский университет   
"Высшая школа экономики"**

*Факультет социально-экономических и компьютерных наук*

Осока Варвара Максимовна

**ЛАБОРАТОРНАЯ РАБОТА № 1**

*Вариант №1*

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *38.03.05 Бизнес-информатика*

Приглашённый преподаватель  
кафедры ИТБ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

С.А. Волков

Пермь, 2023 год

**Оглавление**

[1 Задание № 1 3](#_Toc148311357)

[1.1 Постановка задачи 3](#_Toc148311358)

[1.2 Анализ 3](#_Toc148311359)

[1.3 Проектирование 4](#_Toc148311360)

[1.4 Разработка 7](#_Toc148311361)

[1.5 Тестирование 8](#_Toc148311362)

[2 Задание № 2 11](#_Toc148311363)

[2.1. Постановка задачи 11](#_Toc148311364)

[2.2. Анализ 11](#_Toc148311365)

[2.3. Проектирование 12](#_Toc148311366)

[2.4. Разработка 14](#_Toc148311367)

[2.5. Тестирование 14](#_Toc148311368)

[3 Задание №3 16](#_Toc148311369)

[3.1. Постановка задачи 16](#_Toc148311370)

[3.2. Анализ 16](#_Toc148311371)

[3.3. Проектирование 16](#_Toc148311372)

[3.4. Разработка 18](#_Toc148311373)

[Приложение А Код программы 19](#_Toc148311374)



# **Задание № 1**

Первое задание лабораторной работы направлено на то, чтобы научиться работать с операторами языка С#. Оно проверяет понимание операций инкремента и декремента, а также учит анализировать выражение для вывода правильной ошибки.

## **Постановка задачи**

Общая постановка задачи:

1. Найти значения выражений, обработав исходные данные.
2. Составить систему тестов и вычислить полученное выражение для разных X.
3. Результаты вывести на печать.

Частная постановка задачи.

Для выражений 1-3 посчитать их значения после ввода пользователем m и n. Область определения в 1-3 выражениях нас не интересует. Для выражений 2 и 3 вывести результат сравнения.

Конкретные задачи:

1. n+++m;
2. m--> n;
3. n--> m;

Для 4 выражения необходимо вычислить его значение. Область определения нас не интересует, так как область определения все числа.

1. .

## **Анализ**

В 1-3 выражениях пользователь вводит данные типа int, в 4 типа double.

Анализ классов входных и выходных данных приведён в таблицах 1, 2, 3. Для каждой переменной создана своя таблица.

Стандартные выходные данные для первых 3-х заданий (ожидаемый вывод):

1. int;
2. string;
3. string.

Таблица 1. Классы входных и выходных данных для переменной n

| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| --- | --- | --- |
| *n* - строка | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *n* – вещественное число или целое за пределами Int32 | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *n* – любое целое число в пределах типа Int32 | Программа выполняется | Стандартные выходные данные |

Похожая таблица, сделанная для переменной m, представлена ниже.

Таблица 2. Классы входных данных для переменной m

| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| --- | --- | --- |
| *m* - строка | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *m* – вещественное число или целое за пределами Int32 | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *m* – любое целое число в пределах типа Int32 | Программа выполняется | Стандартные выходные данные |

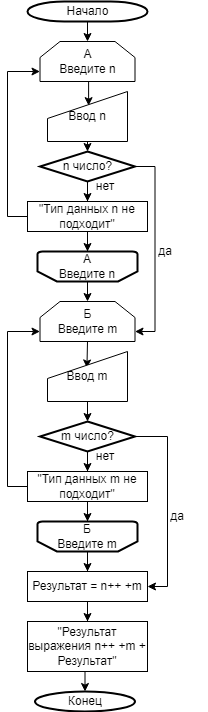
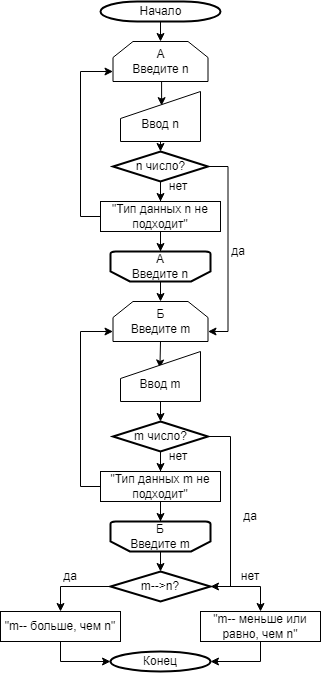
В 4-м задании вводится переменная x типа double. Ожидаемый стандартный вывод для 4-го задания – число типа double. Ниже приведена таблица анализа классов входных и выходных данных для переменной x. В качестве sin(x) используется число в радианах. Переменная x может быть любым вещественным числом, поэтому другие критерии рассматриваться не будут. В задании у нас есть дробь, но её знаменатель никогда не будет равен 0.

Таблица 3. Классы входных данных для переменной x

| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| --- | --- | --- |
| *x* – строка или вещественное число за пределами типа double | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *x* >= 0 | Программа выполняется | Стандартные выходные данные |
| *x* < 0 | Программа выполняется | Стандартные выходные данные |

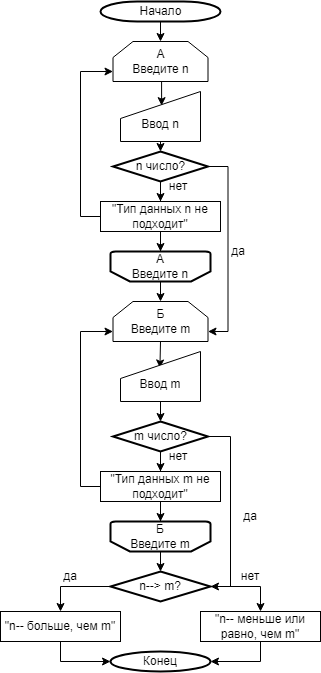
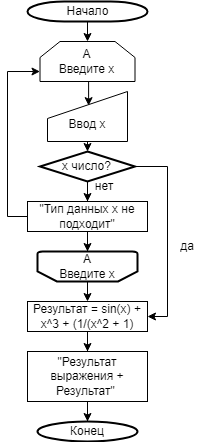
## **Проектирование**

Для реализации проектирования необходимо составить блок схемы, они представлены на рисунках 1.3.1., 1.3.2., 1.3.3., и 1.3.4. (по заданиям).



***Рис. 1.3.1. – схема алгоритма для выражения 1.***

***Рис 1.3.2. – схема алгоритма для выражения 2.***

******

***Рис 1.3.3 – схема алгоритма для выражения 3.***

***Рис 1.3.4 - схема алгоритма для выражения 4.***

## **Разработка**

Разработка осуществляется с помощью листинга программы. Код см. в приложении А.

В коде использована конструкция switch/case, которая оценивает выражение и сравнивает его значение с набором значений. При совпадении выполняется код.

С помощью данной конструкции осуществлено переключение между заданиями (1, 2 и 3), а также между выражениями (1-4) в 1 задании.

Для того, чтобы данная конструкция работала бесконечное количество раз, был создан цикл do…while, с условием, что флаг = false. Таким образом, мы заходим в case 1 – задание 1.

Выражение 1.

Здесь объявлены переменные int m, int n, которые будут использованы в выражении 1, 2 и 3. Далее следует цикл while, в котором методом Try.Parse мы пытаемся преобразовать строку в int. При исполнении метода флаг tmp становится true и мы выходим из цикла. Если это не удаётся, то пользователю выводится сообщение об ошибке, после чего требуется повторный ввод.

Далее идёт такой же цикл, но в отношении переменной m.

Затем вычисляется значение выражения n++ +m. Пользователю выводится результат выражения.

Выражение 2.

Здесь использованы такие же циклы, как и в прошлом выражении.

Далее идёт конструкция if, которая проверяет больше ли m--, чем n. Если больше, то пользователю выводится сообщение, сообщающее об этом, в противном случае тоже выводится сообщение.

Выражение 3.

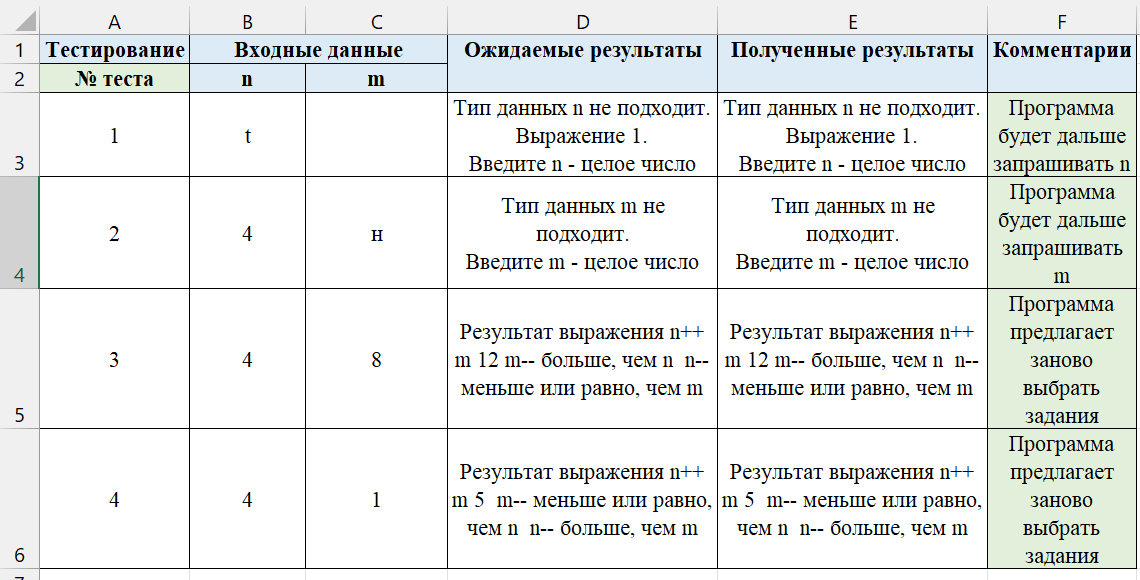
Здесь использованы такие же циклы, как и в предыдущих двух заданиях, а также та же конструкция if. Она проверяет больше ли n--, чем n. Если да, то пользователю выводит сообщение об этом, в противном случае, тоже выводится сообщение.

Выражение 4.

Переменной x присваивается значение double. Далее тем же циклом происходит преобразование строки в double. После чего с помощью Math происходит вычисление выражения и вывод его результата пользователю.

## **Тестирование**

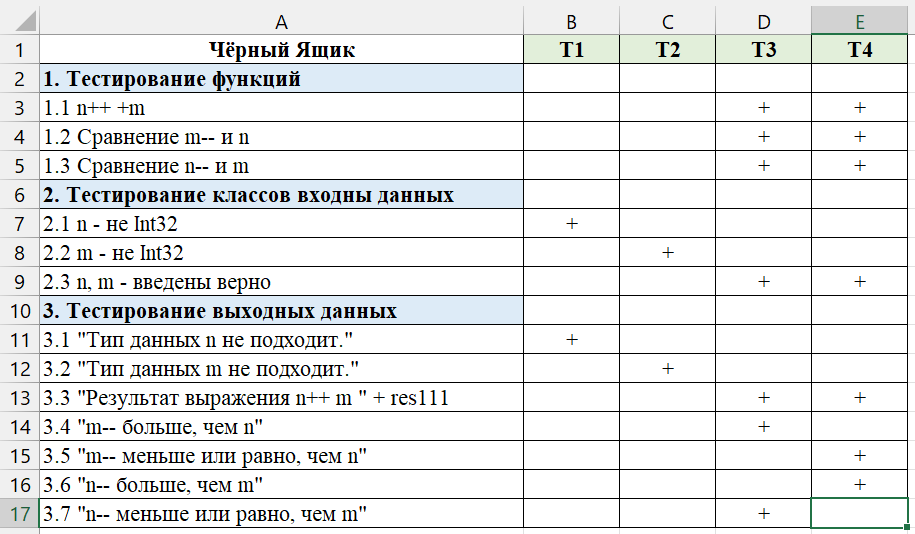
Результаты тестирования представлены на рисунке 1.5.1. На нём отображено тестирование 1-3 выражений.



***Рисунок 1.5.1 – Таблица тестов для задания 1 выражений 1-3***

Таблица тестов составлялась с помощью дополнительных таблиц по критериям черного и белого ящиков. Они представлены ниже на рисунках 1.5.2 и 1.5.3.

Рисунок 1.5.2 содержит таблицу с критериями черного ящика (см. ниже).



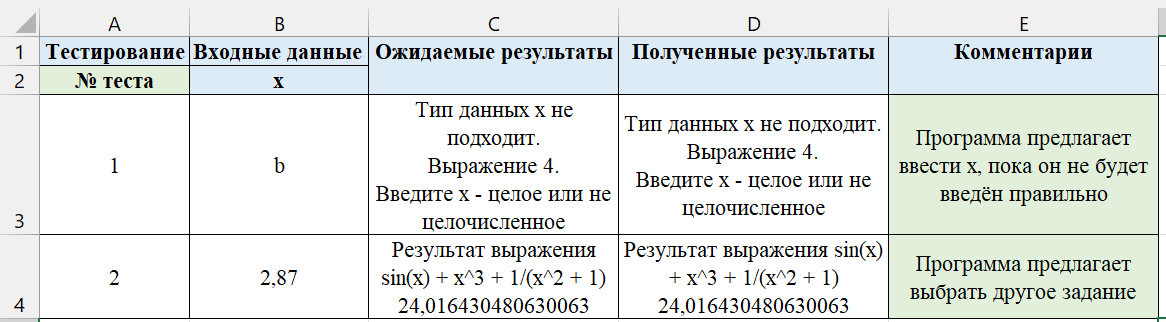
***Рисунок 1.5.2 – Таблица критериев черного ящика для задания 1 выражений 1-3***

Также тестирование проводилось по критериям белого ящика. Таблицу с критериями белого ящика можно заметить на рисунке 1.5.3 (см. ниже).



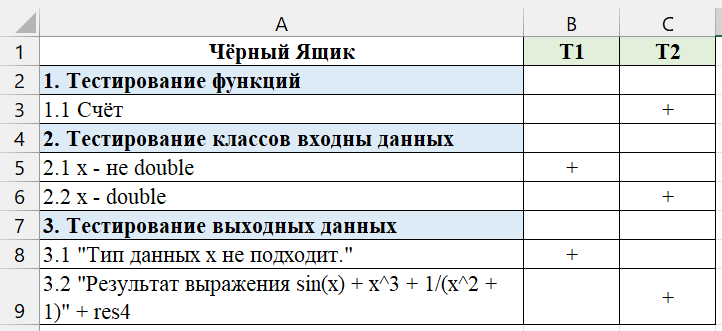
***Рисунок 1.5.3 – Таблица критериев белого ящика для задания 1 выражений 1-3***

Результаты тестирования представлены на рисунке 1.5.4. На нём предоставлены результаты тестирование четвёртого выражения.

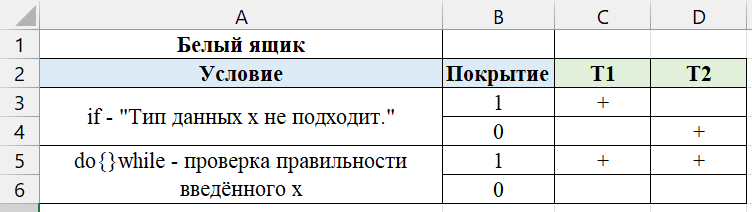


***Рисунок 1.5.4 – Таблица тестов для задания 1 выражения 4***

Рисунок 1.5.5 содержит таблицу с критериями черного ящика (см. ниже).



***Рисунок 1.5.5 – Таблица критериев черного ящика для задания 1 выражения 4***

Также тестирование проводилось по критериям белого ящика. Таблицу с критериями белого ящика можно заметить на рисунке 1.5.6 (см. ниже). 

***Рисунок 1.5.6 – Таблица критериев белого ящика для задания 1 выражения 4***

# **Задание № 2**

Второе задание посвящено получению навыков работы с операторами языка С# и работы со сложными условиями.

## **Постановка задачи**

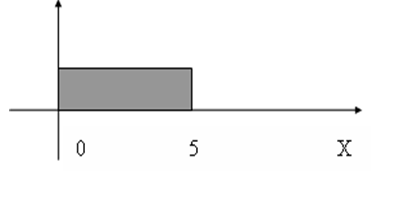
Общая постановка задачи:

1. Определить принадлежит ли точка области, обработав исходные данные.
2. Составить систему тестов и проверить на разных x и y.
3. Результаты вывести на печать.

Частная постановка задачи.

Сравнить координаты x и y с данной нам областью и вывести пользователю ответ, принадлежит точка этой области или нет.

Конкретная задача расположена на рисунке 2.1.1.



***Рис 2.1.1 – задание 2***

## **Анализ**

Во втором задании пользователь вводит числа в переменные x и y, которым присвоены значения double. Затем идёт сравнение с ограничением по x (>= 0; <= 5) и по y(>= 0 и <= 1).

В таблицах 4 и 5 предоставлены классы входных и выходных данных для переменных x и y.

Таблица 4 – Классы входных и выходных данных для переменной x

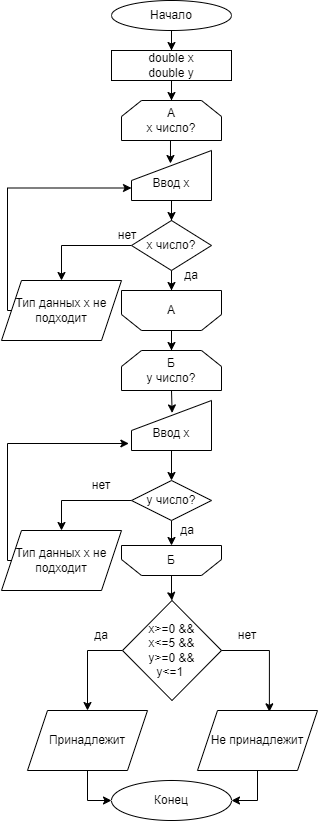
| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| --- | --- | --- |
| *x* - строка | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *x* – вещественное число за пределами double | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *x* – любое число в пределах типа double | Программа выполняется | Стандартные выходные данные |

Таблица 5 – Классы входных и выходных данных для переменной y

| **Классы входных данных** | **Условия** | **Классы выходных данных** |
| --- | --- | --- |
| *y* - строка | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *y* – вещественное число за пределами double | Невозможно вычислить | Сообщение об ошибке из-за неправильного типа введённых данных |
| *y* – любое число в пределах типа double | Программа выполняется | Стандартные выходные данные |

## **Проектирование**

Для реализации проектирования необходимо составить блок схему, она представлена на рисунке 2.3.1.



***Рис 2.3.1 - схема алгоритма для задания 2***

## **Разработка**

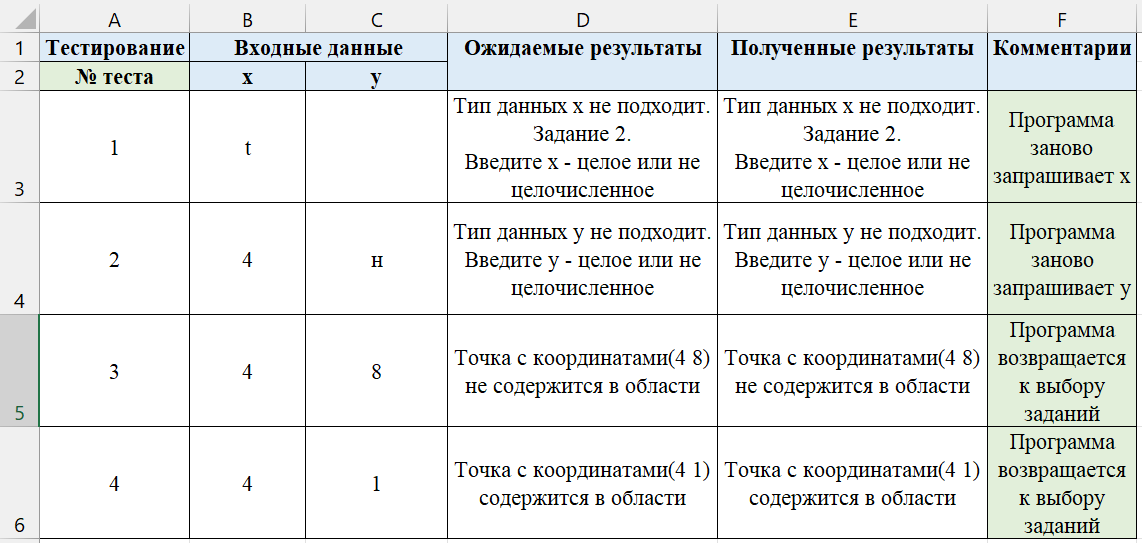
Разработка осуществляется с помощью листинга программы. Код см. в приложении А.

Создаём переменные x и y и присваиваем им значение double. Далее повторяем цикл по проверке подходят нам данные или нет, если нет, то выводим ошибку.

Далее с помощью условия if проверяем ограничение области с 4 сторон.

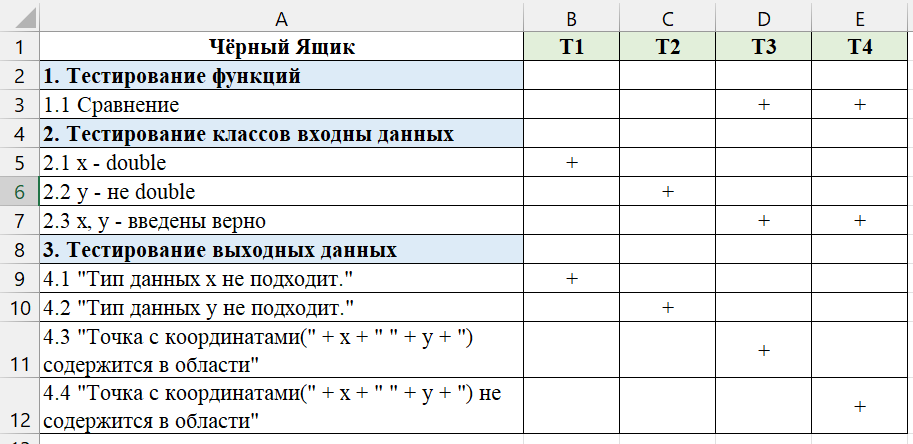
После чего на основе результата сравнения x и y выводим сообщение о том, находится точка в области или нет.

## **Тестирование**

Результаты тестирования представлены на рисунке 2.5.1 (см. ниже). 

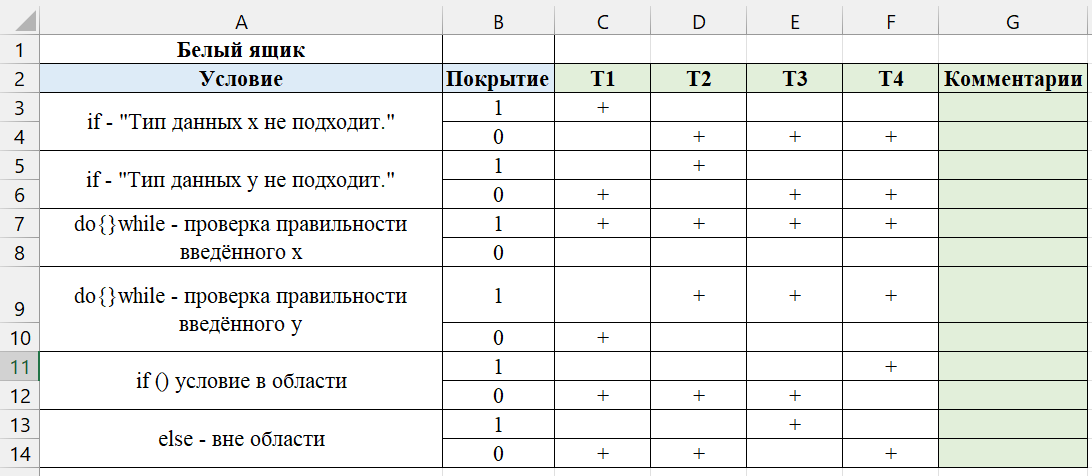
***Рисунок 2.5.1 – Таблица тестов для задания 2***

Таблица тестов составлялась с помощью дополнительных таблиц по критериям черного и белого ящиков. Они представлены ниже на рисунках 2.5.2 и 2.5.3.

Рисунок 2.5.2 содержит таблицу с критериями черного ящика (см. ниже). 

***Рисунок 2.5.2 – Таблица критериев тестирования черным ящиком для задания 2***

Также тестирование проводилось по критериям белого ящика. Таблицу с критериями белого ящика можно заметить на рисунке 2.5.3 (см. ниже).



***Рисунок 2.5.3 – Таблица критериев тестирования белым ящиком для задания 2***

# **Задание №3**

Задание 3 направлено на анализ типов данных, которыми мы оперируем и анализ результатов, которые мы получаем на основе тех типов данных, которые используем.

## **Постановка задачи**

Общая постановка задачи:

1. Посчитать значение выражения присвоив разные вещественные типы данных a и b.
2. Результат всех вычислений вывести на печать.
3. Объяснить полученные результаты.

Частная постановка задачи.

Посчитать значение выражения в разных типах данных при заданных a = 1000, b =0.0001.

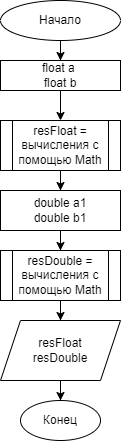
## **Анализ**

Входных данных нет, так как переменные объявлены в виде констант. В качестве выходных данных получится число вещественного типа. Результаты в типе float и double будут не одинаковыми, потому что в какой-то момент тип float (за счет 4 битов вместо 8 типа double) обрежет часть числа и результат получится совершенно другой.

Так как в задании содержаться ФСУ, данная формула после раскрытия скобок и сокращения принимает вид ,что равно единице.

## **Проектирование**

Для дальнейшей реализации проектирования необходимо составить блок схему, она представлена ниже на рисунке 3.3.1.



***Рис 3.3.1 - схема алгоритма для задания 3***

## **Разработка**

Разработка осуществляется с помощью листинга программы. Код см. в приложении А.

Создаём переменные a, b, resFoat типа float, с помощью Math вычисляем.

Создаём переменные a1, b1, resDoublet типа double, с помощью Math вычисляем.

* 1. Выводим результаты с объяснением почему они разные.

# **Приложение А Код программы**

namespace lab11

{

internal class Program

{

static void Main(string[] args)

{

Console.WriteLine( "Добро пожаловать в Лабораторную работу №1.");

var caseFlag = false; //создаём флаг, который отвечает за правильность ввода данных пользователем

do

{

var flagTmp = false;

caseWrite();

switch (Console.ReadLine()) //выбор задания

{

case "1":

var expFlag = false;

do

{

Console.WriteLine("Выберите выражение от 1 до 4");

switch (Console.ReadLine())

{

case "1":

int m;

int n;

do //вводим n пока оно не будет числом

{

Console.WriteLine(

"Выражение 1. \n" +

"Введите n - целое число");

var tmp = Console.ReadLine();

flagTmp = int.TryParse(tmp, out n);

if (!flagTmp)

{

Console.WriteLine("Тип данных n не подходит.");

}

} while (!flagTmp);

do //вводим m пока оно не будет числом

{

Console.WriteLine("Введите m - целое число");

var tmp = Console.ReadLine();

flagTmp = int.TryParse(tmp, out m);

if (!flagTmp)

{

Console.WriteLine("Тип данных m не подходит.");

}

} while (!flagTmp);

var res111 = n++ + m;

Console.WriteLine("Результат выражения n++ m \n" + res111);

expFlag = true;

break;

case "2":

do //вводим n пока оно не будет числом

{

Console.WriteLine(

"Выражение 2. \n" +

"Введите n - целое число");

var tmp = Console.ReadLine();

flagTmp = int.TryParse(tmp, out n);

if (!flagTmp)

{

Console.WriteLine("Тип данных n не подходит.");

}

} while (!flagTmp);

do //вводим m пока оно не будет числом

{

Console.WriteLine("Введите m - целое число");

var tmp = Console.ReadLine();

flagTmp = int.TryParse(tmp, out m);

if (!flagTmp)

{

Console.WriteLine("Тип данных m не подходит.");

}

} while (!flagTmp);

if (m-- > n)

{

Console.WriteLine("m-- больше, чем n");

}

else

{

Console.WriteLine("m-- меньше или равно, чем n");

}

expFlag = true;

continue;

case "3":

do //вводим n пока оно не будет числом

{

Console.WriteLine(

"Выражение 3. \n" +

"Введите n - целое число");

var tmp = Console.ReadLine();

flagTmp = int.TryParse(tmp, out n);

if (!flagTmp)

{

Console.WriteLine("Тип данных n не подходит.");

}

} while (!flagTmp);

do //вводим m пока оно не будет числом

{

Console.WriteLine("Введите m - целое число");

var tmp = Console.ReadLine();

flagTmp = int.TryParse(tmp, out m);

if (!flagTmp)

{

Console.WriteLine("Тип данных m не подходит.");

}

} while (!flagTmp);

if (n-- > m)

{

Console.WriteLine("n-- больше, чем m");

}

else

{

Console.WriteLine("n-- меньше или равно, чем m");

}

expFlag = true;

continue;

case "4":

double x4;

do //вводим x пока оно не будет числом

{

Console.WriteLine(

"Выражение 4. \n" +

"Введите x - целое или не целочисленное");

var tmp = Console.ReadLine();

flagTmp = double.TryParse(tmp, out x4);

if (!flagTmp)

{

Console.WriteLine("Тип данных x не подходит.");

}

} while (!flagTmp);

var res4 = Math.Sin(x4) + Math.Pow(x4, 3) + (1 / (Math.Pow(x4, 2) + 1));

Console.WriteLine("Результат выражения sin(x) + x^3 + 1/(x^2 + 1)\n" + res4);

expFlag = true;

continue;

}

} while (expFlag == false); // !expFlag

break;

case "2":

double x;

double y;

do //вводим x пока оно не будет числом

{

Console.WriteLine(

"Задание 2. \n" +

"Введите x - целое или не целочисленное");

var tmp = Console.ReadLine();

flagTmp = double.TryParse(tmp, out x);

if (!flagTmp)

{

Console.WriteLine("Тип данных x не подходит.");

}

} while (!flagTmp);

do //вводим y пока оно не будет числом

{

Console.WriteLine("Введите y - целое или не целочисленное");

var tmp = Console.ReadLine();

flagTmp = double.TryParse(tmp, out y);

if (!flagTmp)

{

Console.WriteLine("Тип данных y не подходит.");

}

} while (!flagTmp);

if ((x >= 0) && (x <= 5) && (y >= 0) && (y <= 1)) //проверяем наличие точки в диапазоне

{

Console.WriteLine("Точка с координатами(" + x + " " + y + ") содержится в области");

}

else

{

Console.WriteLine("Точка с координатами(" + x + " " + y + ") не содержится в области");

}

continue;

case "3":

float a = 1000;

float b = 0.0001f;

float resFloat = ((float)Math.Pow((a + b), 2) - ((float)Math.Pow(a, 2) + 2 \* a \* b)) / (float)Math.Pow(b, 2); //float перед Pow потому что Pow это double

double a1 = 1000;

double b1 = 0.0001;

double resDouble = (Math.Pow((a1 + b1), 2) - (Math.Pow(a1, 2) + 2 \* a1 \* b1)) / Math.Pow(b1, 2);

Console.WriteLine(

"Задание 3. \n" +

"Выражение ((a + b)^2 - (a^2 + 2ab))/b^2 имеет 2 ответа в зависимости от того, с каким типом данных (float или double) мы считаем. \n Ответ 1. Тип данных float. \n" + resFloat + "\n Ответ 2. Тип данных double \n" + resDouble);

break;

}

} while (caseFlag is false);

}

public static void caseWrite()

{

Console.WriteLine(

"\n Выберите задание от 1 до 3 \n" +

"Задание 1. \n " +

"Выражение 1.n++ m \n " +

"Выражение 2.m-- > n \n " +

"Выражение 3.n-- > m \n " +

"Выражение 4.sin(x) + x ^ 3 + 1 / (x ^ 2 + 1) \n " +

"Задание 2.\n " +

"Принадлежит ли заданная точка области ? \n " +

"Задание 3.\n " +

"Вычисление выражения((a + b) ^ 2 - (a ^ 2 + 2ab)) / b ^ 2");

}

}

}namespace lab11