**Шумилов Лев, РИС-20-1б, Вариант №25**

**Отчёт по практической работа №1 «Выполнение программы простой структуры. Вычисление выражений с использованием стандартных функций»**

**Таблица с задачами:**

|  |  |  |
| --- | --- | --- |
| Задача №1 | Задача №2 | Задача №3 |
| *1) m - ++n*  *2) m++ > --n*  *3) m-- < ++n*  *4) arcsin(|x+1|)* |  | *a = 100, b = 0.001* |

**Задание** состоит их трёх задач:

1. Вычисление значений выражений для данных выражений;
   1. Определить тип заданных данных и найти их значения;
   2. Составить систему тестов и вычислить полученное выражение для нескольких значений X, при каких X выражение не может быть вычислено;
2. Определить принадлежность точки к заштрихованной области;
   1. Вывести TRUE если принадлежит и FALSE в противном случае;
   2. Составить систему тестов и вычислить полученное выражение для нескольких точек, принадлежащих и не принадлежащих заштрихованной области;
3. Вычислить значение выражения, используя различные вещественные типы данных: float и double;
   1. Получить значения выражений и сравнить их;

**Дополнительно:**

1. Результаты всех вычислений выводить на печать;
2. Реализовать меню для выполнения всех вычислений за 1 сеанс выполнения программы;
3. Объяснить полученные результаты;

**Анализ задач:**

**Задача №1:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Описание функции | Входные данные | Классы входных данных | Выходные данные |
| 1 | Ввод числа с клавиатуры (для целого и вещественных чисел) | Строка | Строку можно перевести в число | Новое число |
| Строку нельзя перевести в число | Сообщение об ошибке и повторный ввод |
| 2 | Вызов функции по номеру из задачи | Целое число | Число входит в заданный диапазон | Выполнение нужной функции |
| Число не входит в заданный диапазон | Сообщение об ошибке |
| 0 | ничего |
| 3 | Перевод на русский значений TRUE и FALSE | Логическое выражение | True | Строка «Истина» |
| False | Строка «Ложь» |
| 4 | Проверка выражения №1 | Целые числа n, m | Любое число | Значение выражения |
| 5 | Проверка выражения №2 | Целые числа n, m | n = m | Ложь |
| n > m + 2 | Истина |
| n < m | Ложь |
| 6 | Проверка выражения №3 | Целые числа n, m | m = n | Ложь |
| m > n + 2 | Истина |
| m < n | Ложь |
| 7 | Проверка выражения №4 | Вещественное число x | x < -1 || x > 1 | Сообщение об ошибке (выход из ограничения) |
| x >= -1 && x <= 1 | Значение выражения |
| 8 | Работа меню | ничего | - | Работа консольного меню |

**Проектирование:**

Взаимодействие пользователя с консольным приложением будет производиться посредством специального, меню, позволяющим выбрать номер выражения, с которым пользователь желает произвести работу.

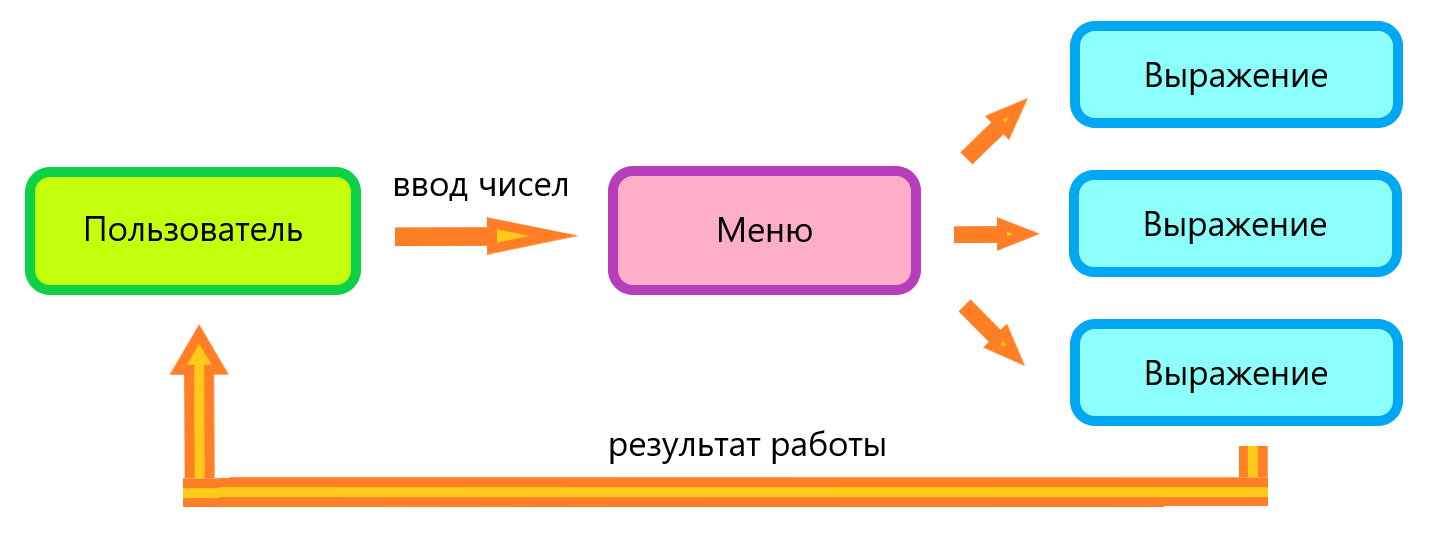
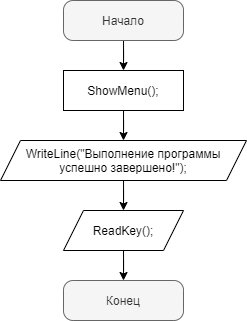


Рис. 1 - Схема работа пользователя с приложением

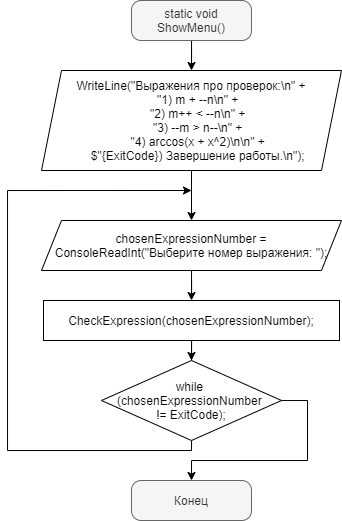
|  |  |  |  |
| --- | --- | --- | --- |
| № | Метод | Входные данные | Выходные данные |
| 1 | ConsoleReadInt | string inputMessage, int inputNumber | int – введённое число,  Сообщение об ошибке и повтор ввода |
| 2 | CheckExpression | int expressionNumber | Invoke method |
| void |
| Сообщение об ошибке |
| 3 | ToRussian | bool state | string: Истина |
| string: Ложь |
| 4 | GetExpressonValue | int n, m | string outputMessage |
| 5 | CompareFirstBoolExpression | int n, m | string outputMessage |
| 6 | CompareSecondBoolExpression | int n, m | string outputMessage |
| 7 | GetMathFunctionValue | double x | Сообщение об ошибке (выход из ограничения) |
| string outputMessage |
| 8 | ShowMenu | ничего | Работа консольного меню |

**Блок-схемы:**

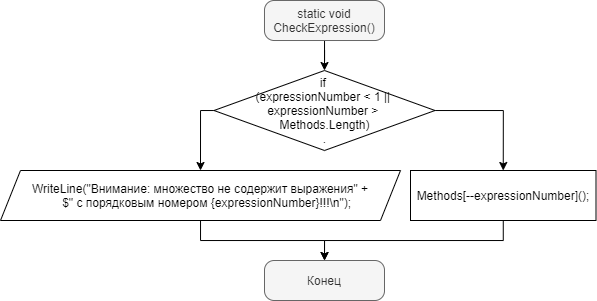
Блок-схема функции Main:



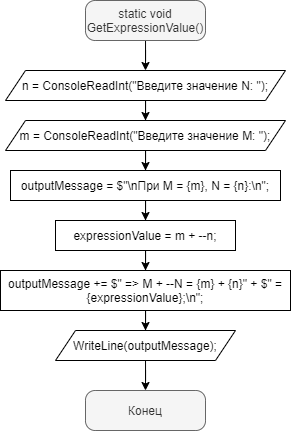
Блок-схема функции ShowMenu:



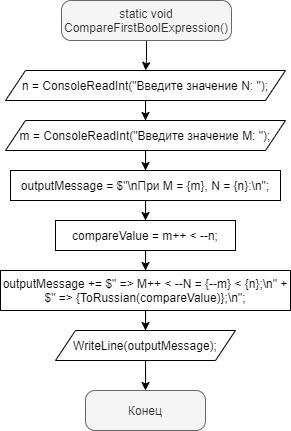
Блок-схема функции CheckExpression:



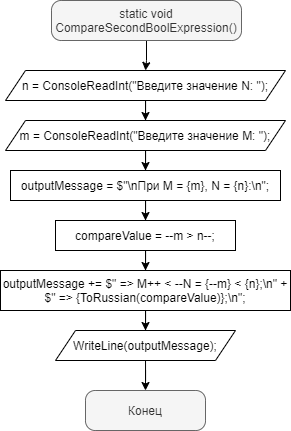
Блок-схема функции GetExpresionValue():



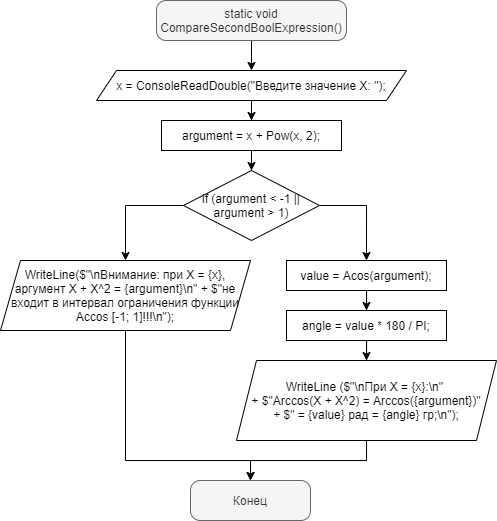
Блок-схема функции CompareFirstBoolExpression:



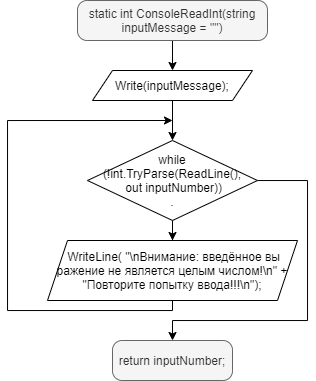
Блок-схема функции CompareSecondBoolExpression:



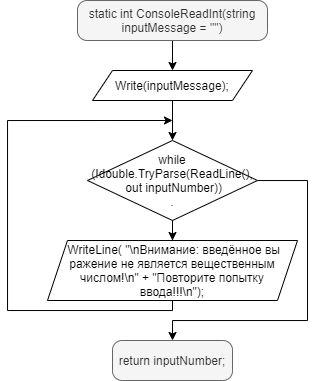
Блок-схема функции GetMathFunctionValue:



Блок-схема функции ConsoleReadInt:



Блок-схема функции ConsoleReadDouble:



**Полный исходный код программы на C#**

/\*

\* 09.10.2021

\* Variant 25

\*

\* Shumilov Lev, ISD-20-1b

\* Шумилов Лев, РИС-20-1б

\*

\* Laboratory work #1

\* Task #1: Expressions processing

\*/

using System;

using static System.Console;

using static System.Math;

namespace TaskNo1

{

class Program

{

const int ExitCode = 0;

static void Main(string[] args)

{

ShowMenu();

WriteLine("Выполнение программы успешно завершено!");

ReadKey();

}

static void ShowMenu()

{

WriteLine("Выражения про проверок:\n" +

"1) m + --n\n" +

"2) m++ < --n\n" +

"3) --m > n--\n" +

"4) arccos(x + x^2)\n\n" +

$"{ExitCode}) Завершение работы.\n");

int chosenExpressionNumber;

do

{

chosenExpressionNumber = ConsoleReadInt("Выберите номер выражения: ");

CheckExpression(chosenExpressionNumber);

}

while (chosenExpressionNumber != ExitCode);

}

static int ConsoleReadInt(string inputMessage = "")

{

Write(inputMessage);

int inputNumber;

while (!int.TryParse(ReadLine(), out inputNumber))

WriteLine(

"\nВнимание: введённое выражение не является целым числом!\n" +

"Повторите попытку ввода!!!\n");

return inputNumber;

}

static void CheckExpression(int expressionNumber)

{

if (expressionNumber < 1 || expressionNumber > Methods.Length)

{

if (expressionNumber != ExitCode)

WriteLine("Внимание: множество не содержит выражения" +

$" с порядковым номером {expressionNumber}!!!\n");

return;

}

Methods[--expressionNumber]();

}

delegate void Method();

static Method[] Methods =

{

new Method(GetExpressionValue),

new Method(CompareFirstBoolExpression),

new Method(CompareSecondBoolExpression),

new Method(GetMathFunctionValue),

};

static string ToRussian(bool state) => state ? "Истина" : "Ложь";

static void GetExpressionValue()

{

int n = ConsoleReadInt("Введите значение N: ");

int m = ConsoleReadInt("Введите значение M: ");

var outputMessage = $"\nПри M = {m}, N = {n}:\n";

int expressionValue = m + --n;

outputMessage +=

$" => M + --N = {m} + {n}" +

$" = {expressionValue};\n";

WriteLine(outputMessage);

}

static void CompareFirstBoolExpression()

{

int n = ConsoleReadInt("Введите значение N: ");

int m = ConsoleReadInt("Введите значение M: ");

var outputMessage = $"\nПри N = {n}, M = {m}:\n";

var compareValue = m++ < --n;

outputMessage +=

$" => M++ < --N = {--m} < {n};\n" +

$" => {ToRussian(compareValue)};\n";

WriteLine(outputMessage);

}

static void CompareSecondBoolExpression()

{

int n = ConsoleReadInt("Введите значение N: ");

int m = ConsoleReadInt("Введите значение M: ");

var outputMessage = $"\nПри N = {n}, M = {m}:\n";

var compareValue = --m > n--;

outputMessage +=

$" => --M > N-- = {m} > {++n};\n" +

$" => {ToRussian(compareValue)};\n";

WriteLine(outputMessage);

}

static void GetMathFunctionValue()

{

double x = ConsoleReadDouble("Введите значение X: ");

double argument = x + Pow(x, 2);

if (argument < -1 || argument > 1)

{

WriteLine($"\nВнимание: при X = {x}, аргумент X + X^2 = {argument}\n" +

$"не входит в интервал ограничения функции Accos [-1; 1]!!!\n");

return;

}

double value = Acos(argument);

double angle = value \* 180 / PI;

WriteLine($"\nПри X = {x}:\n" +

$"Arccos(X + X^2) = Arccos({argument})" +

$" = {value} рад = {angle} гр;\n");

}

static double ConsoleReadDouble(string inputMessage = "")

{

Write(inputMessage);

double inputNumber;

while (!double.TryParse(ReadLine(), out inputNumber))

WriteLine(

"\nВнимание: введённое выражение не является вещественным числом!!!\n" +

"Повторите попытку ввода!!!\n");

return inputNumber;

}

}

}

**Задача №2:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Описание функции | Входные данные | Классы входных данных | Выходные данные |
| 1 | Ввод вещественного числа с клавиатуры | Строка | Строку можно перевести в число | Новое число |
| Строку нельзя перевести в число | Сообщение об ошибке и повторный ввод |
| 2 | Площадь треугольника | 3 точки координат вершин треугольника | Любая точка | Вещественное число - площадь |
| 3 | Проверка на включение точки в треугольник | Точка с координатами | Точка входит | true |
| Точки не входит | false |
| 4 | Проверка на включение точки в круг | Точка с координатами | Точка входит | true |
| Точки не входит | false |
| 5 | Задание радиуса круга | Вещественное число | > 0 | Радиус задаётся |
| < 0 | Сброс исключения |

**Проектирование:**

Приложение решает лишь одну задачу, причём входные данные постоянны. Радиус круга больше нуля, точки не имеют никаких ограничений, дополнительное тестирование не требуется.

Для большего удобства и дополнительной абстракции, для точки, круга и треугольника спроектированы отдельные классы:

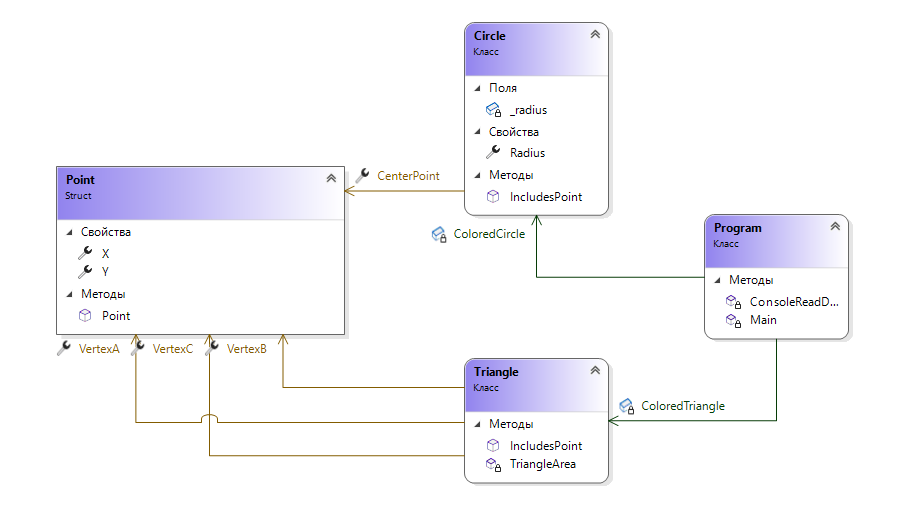
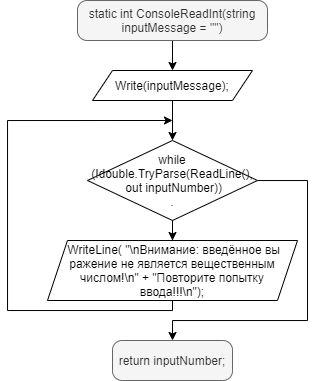


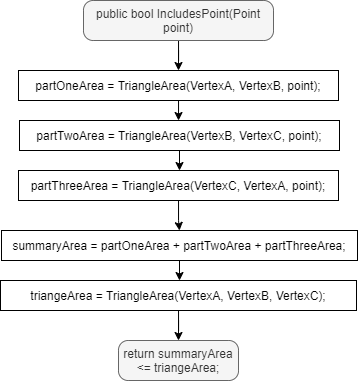
Рис. 2 - UML-диаграмма классов приложения

|  |  |  |  |
| --- | --- | --- | --- |
| № | Метод | Входные данные | Выходные данные |
| 1 | ConsoleReadDouble | Строка | double inputNumber |
| 2 | TriangleArea | Point A, B, C | double |
| 3 | Triangle.IncludesPoint | Point point | bool |
| 4 | Circle.IncludesPoint | Point point | bool |
| 5 | Circle.Radius | Double value | void |

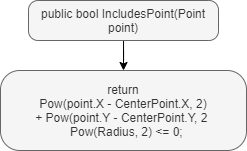
Блок-схема функции ConsoleReadDouble:



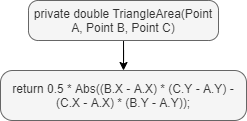
Блок-схема метода Triangle.IncludesPoint:



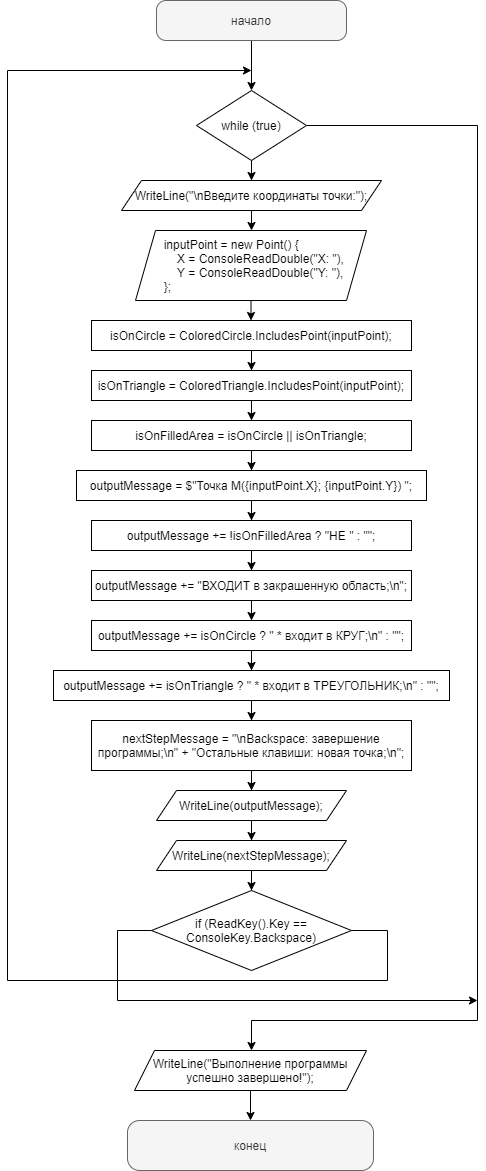
Блок-схема метода Circle.IncludesPoint:



Блок-схема метода TriangleArea:



Блок-схема функции Main:



**Полный исходный код программы на C#**

Program.cs

\* 09.10.2021

\* Variant 25

\*

\* Shumilov Lev, ISD-20-1b

\* Шумилов Лев, РИС-20-1б

\*

\* Laboratory work #1

\* Task #2: 2D Geometry belonging

\*/

using System;

using static System.Console;

using static System.Math;

namespace TaskNo2

{

class Program

{

static Triangle ColoredTriangle = new Triangle()

{

VertexA = new Point(-10, 0),

VertexB = new Point(0, 5),

VertexC = new Point(0, -5),

};

static Circle ColoredCircle = new Circle()

{

Radius = 5,

CenterPoint = new Point(5, -1),

};

static double ConsoleReadDouble(string inputMessage = "")

{

Write(inputMessage);

double inputNumber;

while (!double.TryParse(ReadLine(), out inputNumber))

WriteLine(

"\nВнимание: введённое выражение не является вещественным числом!!!\n" +

"Повторите попытку ввода!!!\n");

return inputNumber;

}

static void Main(string[] args)

{

while (true)

{

WriteLine("\nВведите координаты точки:");

var inputPoint = new Point()

{

X = ConsoleReadDouble("X: "),

Y = ConsoleReadDouble("Y: "),

};

var isOnCircle = ColoredCircle.IncludesPoint(inputPoint);

var isOnTriangle = ColoredTriangle.IncludesPoint(inputPoint);

var isOnFilledArea = isOnCircle || isOnTriangle;

var outputMessage = $"Точка M({inputPoint.X}; {inputPoint.Y}) ";

outputMessage += !isOnFilledArea ? "НЕ " : "";

outputMessage += "ВХОДИТ в закрашенную область;\n";

outputMessage += isOnCircle ? " \* входит в КРУГ;\n" : "";

outputMessage += isOnTriangle ? " \* входит в ТРЕУГОЛЬНИК;\n" : "";

var nextStepMessage =

"\nBackspace: завершение программы;\n" +

"Остальные клавиши: новая точка;\n";

WriteLine(outputMessage);

WriteLine(nextStepMessage);

if (ReadKey().Key == ConsoleKey.Backspace) break;

}

WriteLine("Выполнение программы успешно завершено!");

ReadKey();

}

}

}

Point.cs

namespace TaskNo2

{

struct Point

{

public double X { get; set; }

public double Y { get; set; }

public Point(double x, double y)

{

X = x;

Y = y;

}

}

}

Triangle.cs

using System;

using static System.Math;

namespace TaskNo2

{

class Triangle

{

public Point VertexA { get; set; }

public Point VertexB { get; set; }

public Point VertexC { get; set; }

public bool IncludesPoint(Point point)

{

double partOneArea = TriangleArea(VertexA, VertexB, point);

double partTwoArea = TriangleArea(VertexB, VertexC, point);

double partThreeArea = TriangleArea(VertexC, VertexA, point);

double summaryArea = partOneArea + partTwoArea + partThreeArea;

double triangeArea = TriangleArea(VertexA, VertexB, VertexC);

return summaryArea <= triangeArea;

}

private double TriangleArea(Point A, Point B, Point C) =>

0.5 \* Abs((B.X - A.X) \* (C.Y - A.Y) - (C.X - A.X) \* (B.Y - A.Y));

}

}

Circle.cs

using System;

using static System.Math;

namespace TaskNo2

{

class Circle

{

private double \_radius = 0;

public double Radius

{

get => \_radius;

set

{

if (value < 0)

throw new ArgumentOutOfRangeException();

\_radius = value;

}

}

public Point CenterPoint { get; set; }

public bool IncludesPoint(Point point) =>

Pow(point.X - CenterPoint.X, 2) +

Pow(point.Y - CenterPoint.Y, 2) -

Pow(Radius, 2) <= 0;

}

}

**Задача №3:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Описание функции | Входные данные | Классы входных данных | Выходные данные |
| 1 | Получение значения выражения через float | нет | нет | Строка с ответом |
| 2 | Получение значения выражения через double | нет | нет | Строка с ответом |

**Проектирование:**

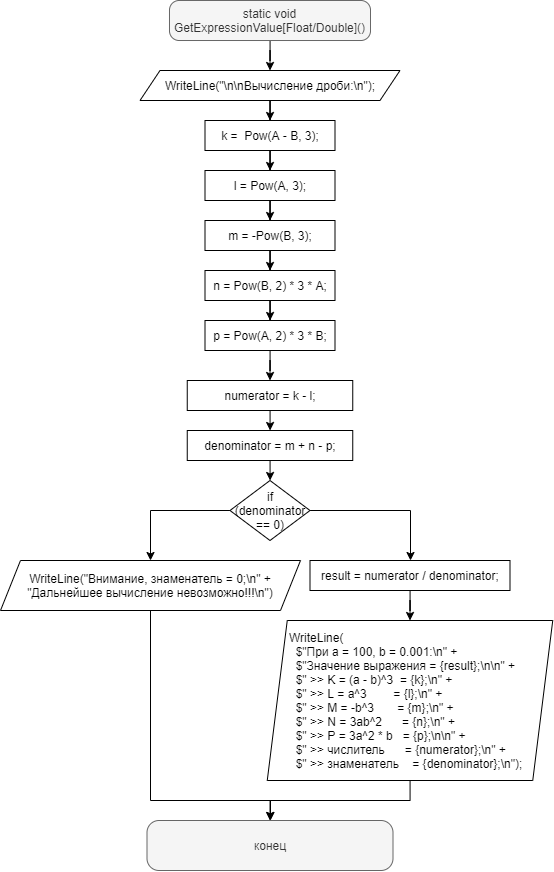
Для типов float и double выделяется разный объём памяти: 4 и 8 байт, соответственно. Возможны ситуации, когда значения выражений, полученные через эти два типа значительно различаются (разница более 1%).

Обе функции выполняют одно и тоже действие, вычисляют значение выражения:

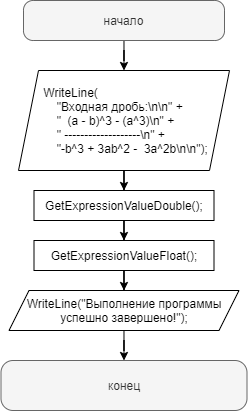


|  |  |  |  |
| --- | --- | --- | --- |
| № | Функция | Входные данные | Выходные данные |
| 1 | GetExpressionValueFloat | нет | Строка с ответом |
| 2 | GetExpressionValueFloat | нет | Строка с ответом |

Блок-схема обеих функций:



Блок-схема функции Main:



**Полный исходный код программы на C#**

/\*

\* 09.10.2021

\* Variant 25

\*

\* Shumilov Lev, ISD-20-1b

\* Шумилов Лев, РИС-20-1б

\*

\* Laboratory work #1

\* Task #3: Computing complex expressions

\*/

using System;

using static System.Console;

using static System.Math;

namespace TaskNo3

{

class Program

{

const float A = 100f;

const float B = 0.001f;

static void GetExpressionValueDouble()

{

WriteLine("Вычисление дроби через тип DOUBLE:\n");

double

k = Pow(A - B, 3),

l = Pow(A, 3);

double

m = -Pow(B, 3),

n = 3 \* A \* Pow(B, 2),

p = 3 \* Pow(A, 2) \* B;

double

numerator = k - l,

denominator = m + n - p;

if (denominator == 0)

{

var errorMessage =

"Внимание, знаменатель = 0;\n" +

"Дальнейшее вычисление невозможно!!!\n";

WriteLine(errorMessage);

return;

}

double result = numerator / denominator;

var outputMessage =

$"При a = 100, b = 0.001:\n" +

$"Значение выражения = {result};\n\n" +

$" >> K = (a - b)^3 = {k};\n" +

$" >> L = a^3 = {l};\n" +

$" >> M = -b^3 = {m};\n" +

$" >> N = 3ab^2 = {n};\n" +

$" >> P = 3a^2 \* b = {p};\n\n" +

$" >> числитель = {numerator};\n" +

$" >> знаменатель = {denominator};\n";

WriteLine(outputMessage);

}

static void GetExpressionValueFloat()

{

WriteLine("\n\nВычисление дроби через тип FLOAT:\n");

float

k = (float) Pow(A - B, 3),

l = (float) Pow(A, 3);

float

m = (float) -Pow(B, 3),

n = (float) Pow(B, 2) \* 3 \* A,

p = (float) Pow(A, 2) \* 3 \* B;

float

numerator = k - l,

denominator = m + n - p;

if (denominator == 0)

{

var errorMessage =

"Внимание, знаменатель = 0;\n" +

"Дальнейшее вычисление невозможно!!!\n";

WriteLine(errorMessage);

return;

}

float result = numerator / denominator;

WriteLine(

$"При a = 100, b = 0.001:\n" +

$"Значение выражения = {result};\n\n" +

$" >> K = (a - b)^3 = {k};\n" +

$" >> L = a^3 = {l};\n" +

$" >> M = -b^3 = {m};\n" +

$" >> N = 3ab^2 = {n};\n" +

$" >> P = 3a^2 \* b = {p};\n\n" +

$" >> числитель = {numerator};\n" +

$" >> знаменатель = {denominator};\n");

}

static void Main(string[] args)

{

WriteLine(

"Входная дробь:\n\n" +

" (a - b)^3 - (a^3)\n" +

" -------------------\n" +

"-b^3 + 3ab^2 - 3a^2b\n\n");

GetExpressionValueDouble();

GetExpressionValueFloat();

WriteLine("Выполнение программы успешно завершено!");

ReadKey();

}

}

}