**3.** **Обработка исключительных ситуаций**

Задание №1. Вычислить значение выражений. В каждой разработанной программе должна быть осуществлена обработка исключительных ситуаций. Отдельно обработаны исключения DivideByZeroException, FormatException.

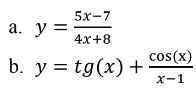


Рисунок 3.1 – Функции для задания №1

Листинг программы:

try

{

Console.Write("Введите x: ");

int x = Convert.ToInt32(Console.ReadLine());

double y = (5 \* x - 7) / (x - 5);

Console.WriteLine($"y: {y}");

y = Math.Tan(y) + Math.Cos(x) / (x - 1);

Console.WriteLine($"y: {y}");

}

catch (DivideByZeroException ex)

{

Console.WriteLine(ex.Message);

}

catch (FormatException ex)

{

Console.WriteLine(ex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 3.1 – Входные и выходные данные программы задание №1

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 4 | -13  -0,681 |

Анализ результатов:



Рисунок 3.2 – Результат выполнения программы задание №1

Задание №2. Для данного вещественного x найти значение следующей функции f, принимающей вещественные значения. В каждой разработанной программе должна быть осуществлена обработка исключительных ситуаций. Отдельно обработаны исключения DivideByZeroException, FormatException. Сгенерируйте пользовательское исключение с помощью оператора throw при проверке условий выхода за диапазон.

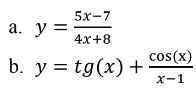


Рисунок 3.3 – Функции для задания №2

Листинг программы:

try

{

Console.Write("Введите x: ");

double x = Convert.ToInt32(Console.ReadLine());

Console.Write("Ответ: ");

if(-4 < x && x < 1 )

{

Console.WriteLine($"{3 \* x / (x - 2)}");

}

else if(x <= 1)

{

Console.WriteLine(3);

}

else

{

throw new Exception("x не входит в диапозон");

}

}

catch (DivideByZeroException ex)

{

Console.WriteLine(ex.Message);

}

catch (FormatException ex)

{

Console.WriteLine(ex.Message);

}

catch(Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 3.2 – Входные и выходные данные программы задание №2

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1 | 3 |

Анализ результатов:



Рисунок 3.4 – Результат выполнения программы задание №2

Задание №3. Описать метод RectPS(x1, y1, x2, y2, P, S), вычисляющий

периметр P и площадь S прямоугольника со сторонами, параллельными осям координат, по координатам (x1,y1), (x2, y2) его противоположных вершин(x1, y1, x2, y2— входные, P и S — выходные параметры вещественного типа). С помощью этого метода найти периметры и площади трех прямоугольников с данными противоположными вершинами.

Листинг программы:

for (int i = 0; i < 3; i++)

{

Console.Write("Введите x1:");

double x1 = Convert.ToDouble(Console.ReadLine());

Console.Write("Введите y1:");

double y1 = Convert.ToDouble(Console.ReadLine());

Console.Write("Введите x2:");

double x2 = Convert.ToDouble(Console.ReadLine());

Console.Write("Введите y2:");

double y2 = Convert.ToDouble(Console.ReadLine());

RectPS(x1, y1, x2, y2, out double P, out double S);

Console.WriteLine($"Периметр: {P}");

Console.WriteLine($"Площадь: {S}");

Console.WriteLine();

}

void RectPS(double x1, double y1, double x2, double y2,

out double P, out double S)

{

P = 0;

S = 0;

try

{

P = 2 \* (x2 - x1 + y2 - y1);

S = (x2 - x1) \* (y2 - y1);

}

catch (DivideByZeroException ex)

{

Console.WriteLine(ex.Message);

}

catch (FormatException ex)

{

Console.WriteLine(ex.Message);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

Таблица 3.3 – Входные и выходные данные программы задание №3

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1 итерация:  x1:1 ; y1:1 ; x2:5 ; y2:6 ;  2 итерация:  x1:2 ; y1:2 ; x2:4 ; y2:5 ;  3 итерация:  x1: 2; y1:3 ; x2: 5; y2:6 ; | 1 итерация:  P:18; S:20;  2 итерация:  P:10; S:6;  3 итерация:  P:12; S:9; |

Анализ результатов:

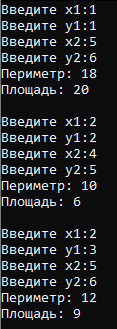


Рисунок 3.5 – Результат выполнения программы задание №3