**4 ОБРАБОТКА ИСКЛЮЧИТЕЛЬНЫХ СИТУАЦИЙ**

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

Листинг программы:

while (true)

{

try

{

Console.Write("Enter x: ");

double x = double.Parse(Console.ReadLine());

Console.WriteLine($"Formula 1 = {Formula(x)}");

Console.WriteLine($"Formula 2 = {Formula2(x)}");

break;

}

catch (FormatException e)

{

Console.WriteLine($"Sorry, it's format exception (Error: {e})");

}

catch (DivideByZeroException e)

{

Console.WriteLine($"Divide by zero (x / 0) (Error: {e})");

}

catch (Exception e)

{

Console.WriteLine($"Some error (Error: {e})");

}

}

static double Formula(double x)

{

try

{

return x + ((2 \* x) / (4 \* Math.Sin(x)));

}

catch (Exception)

{

throw new DivideByZeroException();

}

}

static double Formula2(double x)

{

try

{

return x - 3 +(1/ Math.Tan(x - 1));

}

catch (Exception)

{

throw new DivideByZeroException();

}

}

Таблица 4.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 78 | 153.8786, 74.9690 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 4.1.

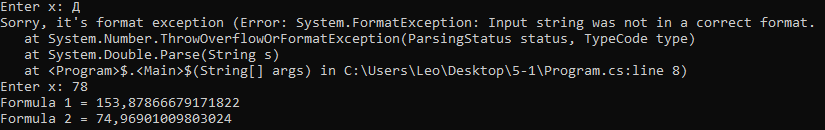


Рисунок 4.1 – Результат работы программы

Источник: собственная разработка

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.

Сгенерировать пользовательское исключение с помощью оператора throw при проверке условий выхода за диапазон.

Листинг программы:

while (true)

{

try

{

Console.Write("Enter x: ");

double x = double.Parse(Console.ReadLine());

Console.WriteLine($"Formula 1 = {Formula(x)}");

break;

}

catch (FormatException e)

{

Console.WriteLine($"Sorry, it's format exception (Error: {e})");

}

catch (DivideByZeroException e)

{

Console.WriteLine($"Divide by zero (x / 0) (Error: {e})");

}

catch (Exception e)

{

Console.WriteLine($"Some error (Error: {e})");

}

}

static double Formula(double x)

{

try

{

if ((x > -5) && (x < 1)) return x - 3;

else if (x >= 1) return Math.Pow(x, 2) / (3 \* x - 9);

else throw new Exception("X - Not in range!");

}

catch (DivideByZeroException)

{

throw new DivideByZeroException();

}

}

Таблица 4.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 10 | 4.761904 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 4.2.

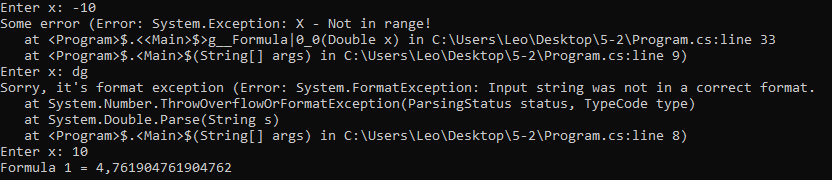


Рисунок 4.2 – Результат работы программы

Источник: собственная разработка

Задание №3. В каждой разработанной программе должна быть осуществлена обработка исключительных ситуаций. Отдельно обработаны исключения DivideByZeroException, FormatException. По возможности сгенерируйте пользовательское исключение с помощью оператора.

Описать метод AddLeftDigit(D, K ), добавляющий к целому положительному числу K слева цифру D (D — входной параметр целого типа, лежащий в диапазоне 1–9, K — параметр целого типа, являющийся одновременно входным и выходным). С помощью этого метода последовательно добавить к данному числу K слева данные цифры D1и D2, выводя результат каждого добавления.

Листинг программы:

while (true)

{

try

{

Console.Write("Enter K: ");

int K = int.Parse(Console.ReadLine());

Console.Write("Enter D1: ");

int D1 = int.Parse(Console.ReadLine());

AddLeftDigit(D1, ref K);

Console.WriteLine(K);

Console.Write("Enter D2: ");

int D2 = int.Parse(Console.ReadLine());

AddLeftDigit(D2, ref K);

Console.WriteLine(K);

break;

}

catch (FormatException e)

{

Console.WriteLine($"Sorry, it's format exception (Error: {e})");

}

catch (DivideByZeroException e)

{

Console.WriteLine($"Divide by zero! {e}");

}

catch (Exception e)

{

Console.WriteLine($"Some error (Error: {e})");

}

}

static void AddLeftDigit(int D, ref int K)

{

if (D <= 0 || D > 9 || K <= 0)

throw new ArgumentException("Неверное значение параметров");

int n = K;

while (n != 0)

{

n /= 10;

D \*= 10;

}

K += D; }

Таблица 4.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 28, 7, 6 | 728, 6728 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 4.3.

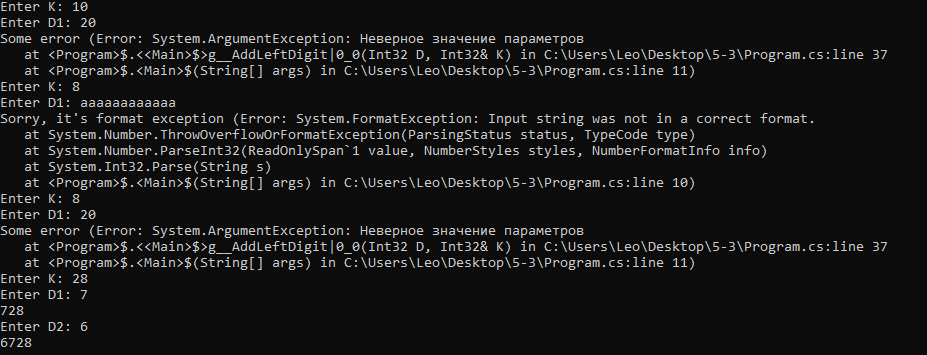


Рисунок 4.3 – Результат работы программы

Источник: собственная разработка