**1 Принципы объектно-ориентированного программирования.**

Задание 1. Дано расстояние в сантиметрах. Найти число полных метров в нем.

Листинг программы:

try

{

Console.WriteLine("Введите количество сантиметров (целочисленное число)");

var data = int.Parse(Console.ReadLine());

var result = data / 100;

Console.WriteLine($"result = {result}");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 20 | result = 0 |

Анализ результатов:



Рисунок 1.1 – Результаты работы программы

Источник: собственная разработка

Задание 2. Проверить истинность высказывания: «Квадратное уравнение A·x2 + B·x + C = 0 с данными коэффициентами A, B, C имеет вещественные корни».

Листинг программы:

try

{

Console.WriteLine("Введите A:");

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

Console.WriteLine("Введите B:");

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

Console.WriteLine("Введите C:");

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

int D = B \* B - 4 \* A \* C;

if (D >= 0)

Console.WriteLine("Вещественные корни: D=" + D);

else

Console.WriteLine("Комплексные корни: D=" + D);

Console.ReadKey(true);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| A =10  B = 203  C = 32210 | Комплексные корни: D=-1247191 |

Анализ результатов:

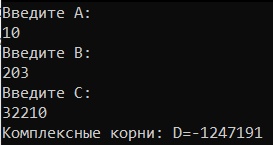


Рисунок 1.2 – Результаты работы программы

Источник: собственная разработка

Задание 3. Даны целые числа K и N. Вывести N раз число K.

Листинг программы:

try

{

Console.Write("Введите k = ");

var k = int.Parse(Console.ReadLine());

Console.Write("Введите n = ");

var n = int.Parse(Console.ReadLine());

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

{

Console.WriteLine(k);

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| k = 5  n = 5 | 5  5  5  5  5 |

Анализ результатов:

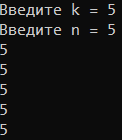
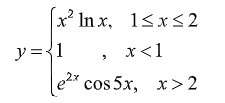


Рисунок 1.3 – Результаты работы программы

Источник: собственная разработка

Задание 4. Написать программу, которая вычисляет значение функции у:



Листинг программы:

try

{

Console.WriteLine("Введите х = ");

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

if (x <= 2 && x >= 1)

{

var y = Math.Pow(x, 2) \* Math.Log10(x);

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

}

else if (x < 1)

{

var y = 1;

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

}

else

{

var y = Math.Pow(Math.E, 2 \* x) \* Math.Cos(5 \* x);

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

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| x = 5 | y = 21832,694831227025 |

Анализ результатов:

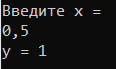


Рисунок 1.4 – Результаты работы программы

Источник: собственная разработка

Задание 5. Написать программу, которая определяет: максимальное значение для двух различных вещественных чисел.

Листинг программы:

try

{

Console.Write("Введите первое число = ");

var firstNumber = double.Parse(Console.ReadLine());

Console.Write("Введите второе число = ");

var secondNumber = double.Parse(Console.ReadLine());

if (firstNumber > secondNumber)

{

Console.WriteLine("Первое больше");

}

else if (firstNumber < secondNumber)

{

Console.WriteLine("Второе больше");

}

else

{

Console.WriteLine("Числа равны");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.5 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Первое число = 10  Второе число = 20 | Второе больше |

Анализ результатов:

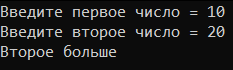


Рисунок 1.5 – Результаты работы программы

Источник: собственная разработка

Задание 6. Составить программу (при решении данных задач использовать оператор switch или вложенные операторы if). Проведен тест, оцениваемый в целочисленных баллах от нуля до ста. Вывести на экран оценку тестируемого в зависимости от набранного количества баллов: от 90 до 100 – «отлично», от 70 до 89 – «хорошо», от 50 до 69 – «удовлетворительно», менее 50 – «неудовлетворительно».

Листинг программы:

try

{

Console.Write("Введите оценку = ");

var mark = int.Parse(Console.ReadLine());

if (mark <= 100 && mark >= 90)

{

Console.WriteLine("Отлично");

}

else if (mark <= 89 && mark >= 70)

{

Console.WriteLine("Хорошо");

}

else if (mark <= 69 && mark >= 50)

{

Console.WriteLine("Удовлетворительно");

}

else if (mark <= 50)

{

Console.WriteLine("Неудовлетворительно");

}

else

{

Console.WriteLine("Error");

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.6 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Введите оценку = 71 | Хорошо |

Анализ результатов:



Рисунок 1.6 – Результаты работы программы

Источник: собственная разработка

Задание 7. Вывести на экран (задачу решите тремя способами – используя операторы цикла while, do while и for): целые числа 1, 3, 5, …, 101 в строчку через пробел;

Листинг программы:

var result = 0;

int i = 0;

int j = 0;

int k = 0;

Console.WriteLine("Цикл For");

for (i = 0; i <= 101; i++)

{

if (i % 2 != 0)

{

Console.WriteLine(i);

}

}

Console.WriteLine("\nЦикл while");

while (j <= 101)

{

if (j % 2 != 0)

{

Console.WriteLine(j);

}

j++;

}

Console.WriteLine("\nЦикл do while");

do

{

if (k % 2 != 0)

{

Console.WriteLine(k);

}

k++;

}

while (k <= 101);

Таблица 1.7 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | 1..101 |

Анализ результатов:



Рисунок 1.7 – Результаты работы программы

Источник: собственная разработка

Задание 8. Дано целое число N (N&gt;0). Найти сумму N^2  + (N + 1)^ ^2  + (N + 2)^2  + … + (2·N)^2.

Листинг программы:

try

{

Console.Write("Введите n = ");

var n = int.Parse(Console.ReadLine());

double result = Math.Pow(n, 2);

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

{

result += Math.Pow(n + i, 2);

}

result += Math.Pow(2 \* n, 2);

Console.WriteLine($"result = {result}");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.8 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| n = 5 | result = 480 |

Анализ результатов:



Рисунок 1.8 – Результаты работы программы

Источник: собственная разработка

Задание 9. Получить 20 значений функции при изменении аргумента x в диапазоне [0,1; 2,1] с шагом 0,1.

Листинг программы:

try

{

const double A = 0;

const double B = Math.PI / 2;

const double M = 10;

double h = (B - A) / M;

double result = 0;

double x = 0;

for (double i = 0.1; i < 2.1; i += 0.1)

{

x = i + h;

result = x - Math.Sin(x);

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

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.9 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | x = 0,0028223867475715703  x = 0,007540060210964761  x = 0,01575021754004252  x = 0,02836999250108474  x = 0,04627245941861846  x = 0,07027790970536751  x = 0,10114565578160806  x = 0,1395664442764618  x = 0,1861555543159068  x = 0,24144664985713638  x = 0,3058864474947033  x = 0,3798302530159732  x = 0,46353841130314466  x = 0,5571737050532454  x = 0,6607997283072322  x = 0,7743802510393553  x = 0,8977795811555991  x = 1,0307639202842325  x = 1,1730036998118545  x = 1,3240768738240407 |

Анализ результатов:

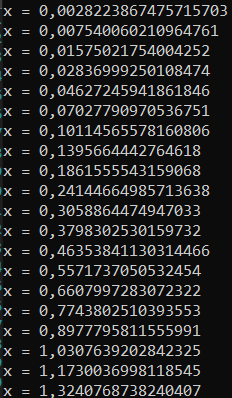


Рисунок 1.9 – Результаты работы программы

Источник: собственная разработка

Задание 10. Написать программу, удаляющую из записи числа все четные цифры. Например, из числа «12345» должно получиться «135».

Листинг программы:

using System.Text;

try

{

Console.Write("Введите число: ");

string str = Console.ReadLine();

var result =new string(str.Where(c => char.GetNumericValue(c) % 2!=0).ToArray());

Console.WriteLine(result);

var strBuilder = new StringBuilder();

foreach (var symbol in str)

{

var digit=char.GetNumericValue(symbol);

if (digit % 2 !=0)

{

strBuilder.Append(symbol);

}

}

Console.WriteLine(strBuilder.ToString());

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

Таблица 1.10 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 12345 | 135 |

Анализ результатов:



Рисунок 1.10 – Результаты работы программы

Источник: собственная разработка