**2 Классы. Библиотеки FCL. Классы как типы и объекты этих типов.**

Задние 1. Создать класс А с целочисленными полями а и b и двумя методами согласно варианту. Внутри класса реализовать конструктор для инициализации a и b. Создать объект класса и продемонстрировать работу со всеми элементами класса. Метод вычисления суммы a и b, метод вычисления значения выражения .

Листинг программы:

try

{

Console.Write("Введите a: ");

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

Console.Write("Введите b: ");

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

A clas = new A(a, b);

clas.SummMethod(a, b);

clas.MathMethod(a, b);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

class A

{

public int first;

public int second;

public A(int first, int second)

{

this.first = first;

this.second = second;

}

public void SummMethod(int a, int b)

{

Console.WriteLine($"Результат первого выражения: {a + b}");

}

public void MathMethod(double a, double b)

{

Console.WriteLine($"Результат первого выражения: {(Math.Sin(b)) / (3 \* a)}");

}

}

Таблица 2.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| a = 10  b = 20 | Результат первого выражения: 30  Результат первого выражения: 0,030431508357587587 |

Анализ результатов:

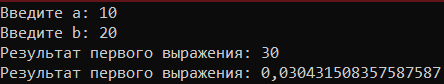


Рисунок 2.1 – Результаты работы программы

Источник: собственная разработка

Задние 2. Построить иерархию классов в соответствии с вариантом задания: cтудент, преподаватель, персона, заведующий кафедрой.

Листинг программы:

try

{

Student st = new Student() {

FirstName = "Anton",

LastName = "Markiv",

Course = 1,

Age = 18,

Sex = "M"

};

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

class Person

{

public string FirstName { get; set; }

public string LastName { get; set; }

public int Age { get; set; }

public string Sex { get; set; }

public void Print()

{

Console.WriteLine("Персона создана");

}

}

class HeadOfDepartment : Person

{

public string Faculty { get; set; }

public string Subjects { get; set; }

public string Timetable { get; set; }

public void Print()

{

Console.WriteLine("Заведующий создан");

}

}

class Teacher : HeadOfDepartment

{

public string Marks { get; set; }

public string Speciality { get; set; }

public void Print()

{

Console.WriteLine("Учитель создан");

}

}

class Student : Person

{

public int Course { get; set; }

public void Print()

{

Console.WriteLine("Студент создан");

}

}

Таблица 2.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| FirstName, LastName, Course, Age, Sex | Print() |

Анализ результатов:



Рисунок 2.2 – Результаты работы программы

Источник: собственная разработка

Задание 3. Описать класс, реализующий десятичный счетчик, который может увеличивать или уменьшать свое значение на единицу в заданном диапазоне. Предусмотреть инициализацию счетчика значениями по .умолчанию и произвольными значениями. Счетчик имеет два метода: увеличения и уменьшения, — и свойство, позволяющее получить его текущее состояние.

Написать программу, демонстрирующую все разработанные элементы класса.

Листинг программы:

try

{

Counter counter = new Counter();

var stop = true;

while (stop)

{

Console.WriteLine("\nНажмите '1', чтобы вывести значение счетчика");

Console.WriteLine("Нажмите '2', чтобы увеличить значение счетчика");

Console.WriteLine("Нажмите '3', чтобы уменьшить значение счетчика");

Console.WriteLine("Нажмите '0', чтобы завершить программу");

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

if (personStop == 0)

{

stop = false;

}

else if (personStop == 1)

{

Console.WriteLine("--------------------------------");

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

Console.WriteLine("--------------------------------");

}

else if (personStop == 2)

{

Console.WriteLine("--------------------------------");

counter.increase();

Console.WriteLine("Счетчик увеличил свое значение");

Console.WriteLine("--------------------------------");

}

else if (personStop == 3)

{

Console.WriteLine("--------------------------------");

counter.low();

Console.WriteLine("Счетчик уменьшил свое значение");

Console.WriteLine("--------------------------------");

}

}

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

class Counter

{

public int value;

public int min;

public int max;

public string Result { get { return value.ToString(); } }

public Counter(int value = 0, int min = 0, int max = 10)

{

this.value = value;

this.min = min;

this.max = max;

}

public void low()

{

value -= 1;

if (value < min) { value = max;}

}

public void increase()

{

value += 1;

if (value > max) { value = min;}

}

}

Таблица 2.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1 | result = 0 |

Анализ результатов:

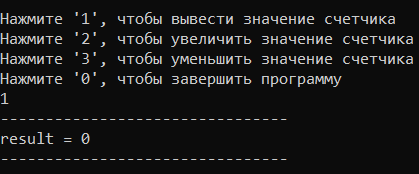


Рисунок 2.3 – Результаты работы программы

Источник: собственная разработка