МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «КубГУ»)

Факультет компьютерных технологий и прикладной математики  
Кафедра информационных технологий

Отчёт по лабораторной работе №3

**«Конструкторы, поля только для чтения, вызов конструкторов»**

Работу выполнил  
студент 42 группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Чан Д. Х.

Направление подготовки 01.03.02 Прикладная математика и информатика

Проверил  
канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А.Полупанов

Краснодар 2020

**Задание.**

1. Ввести и выполнить приложение из примеров 1-4. Проанализировать и объяснить работу приложений по примерам.
2. Создать приложение, в котором в цикле создается 10 объектов класса. Поля только для чтения каждого экземпляра равны порядковому номеру, отражающего очередность создания объектов.
3. Изменить тело конструктора с аргументами таким образом, чтобы производился анализ диапазона значения передаваемого аргумента. При выходе из некоторого диапазона - печаталось соответствующее сообщение, а значение аргумента приравнивалось, в зависимости от ситуации, минимально или максимально допустимому значению.

**Выполнение задания**

1. Выполним приложения из примеров 1- 4 и проанализируем их.

В примере 1 показывается как влияет явное задание конструктора на результат. В случае, когда конструктор задается компилятор автоматически по умолчанию результат программы является таким как на рисунке 1. В случае добавления конструктора в класс СА результат работы приложения меняется на тот, который показан на рисунке 2.

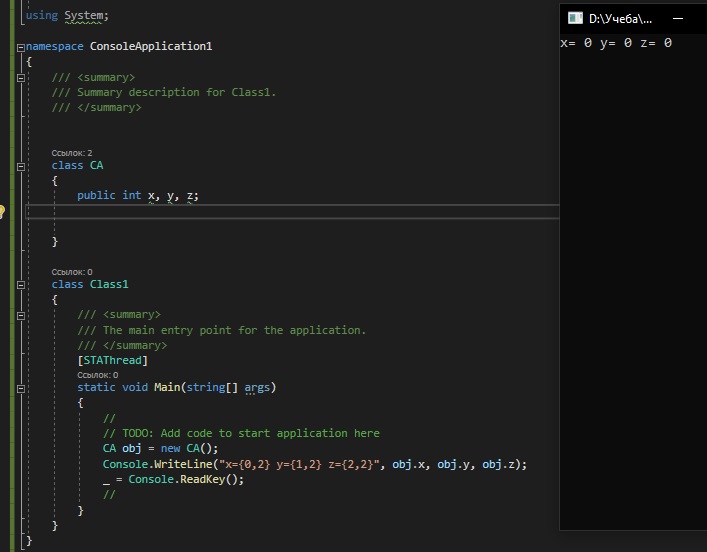


Рисунок 1 – результат работы программы из примера 1

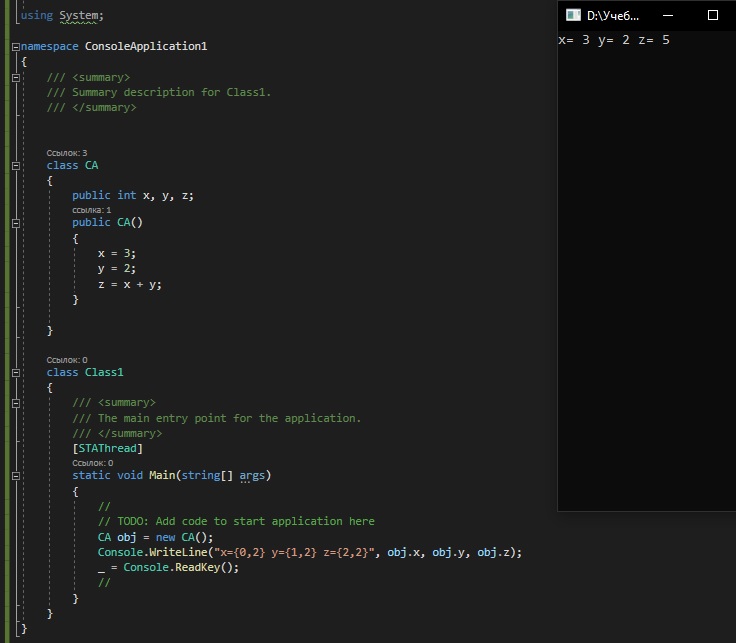


Рисунок 2 – результат работы программы из примера 1 после добавления явного конструктора

В программе из примера 2 можно увидеть, что Большинство конструкторов в **С#** принимают аргументы, с помощью которых разные объекты данного класса могут быть по-разному инициализированы. Наглядно это представлено на рисунке 3.

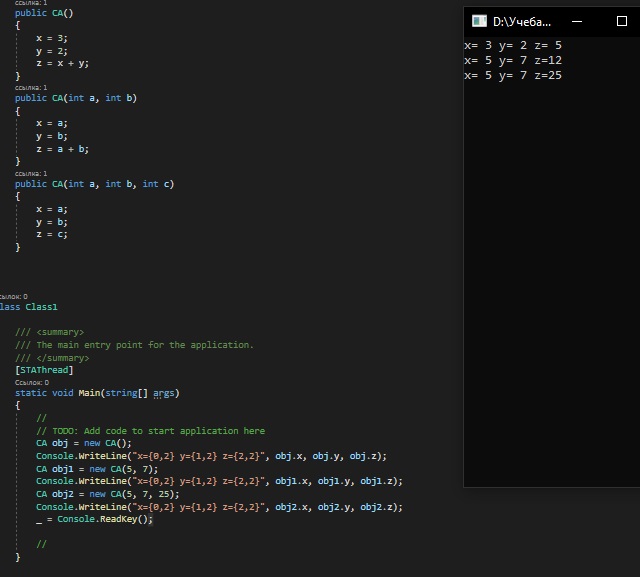


Рисунок 3 – результат работы программы из примера 2

В примере 3 показана работа с полями для чтения (в коде задается как public readonly int rd ) (рис.4)

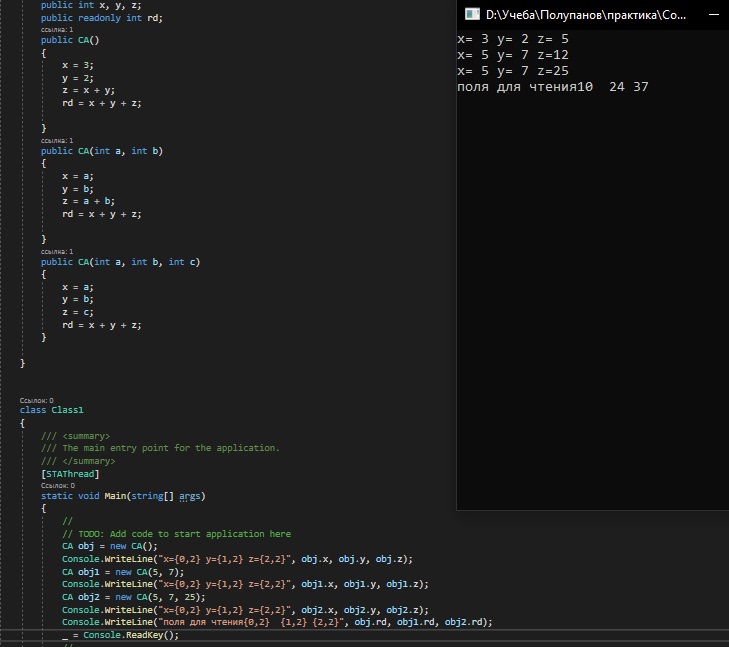


Рисунок 4 – результат работы программы из примера 3

Пример 4 описывает вызов конструктора из другого конструктора. При вызове конструктора из другого конструктора, может быть вызван только один конструктор, который всегда выполняется первым. (рис 5).

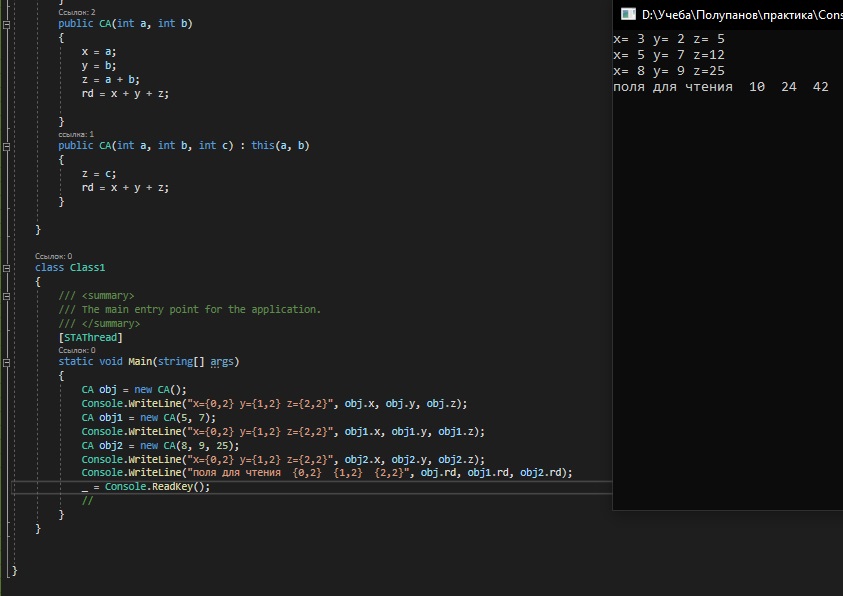


Рисунок 5 – результат работы программы из примера 4

2. Создать приложение, в котором в цикле создается 10 объектов класса. Поля только для чтения каждого экземпляра равны порядковому номеру, отражающего очередность создания объектов.

Код программы:

class CA

{

public int x, y, z;

public readonly int rd;

public CA()

{

x = 3;

y = 2;

z = x + y;

rd = x + y + z;

}

public CA(int a)

{

x = a;

y = b;

z = a + b;

rd = x + y + z;

}

class Third

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main(string[] args)

{

First[] obj = new First[10];

for (int i = 0; i < 10; i++)

{

obj[i] = new First(i);

Console.WriteLine("obj[{0}] = {1}", i, obj[i].GetP());

}

Console.WriteLine();

}

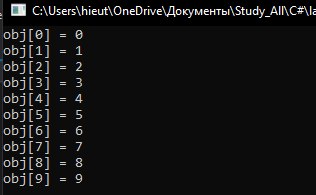


Рисунок 5 – результат работы программы

3. Изменить тело конструктора с аргументами таким образом, чтобы производился анализ диапазона значения передаваемого аргумента. При выходе из некоторого диапазона - печаталось соответствующее сообщение, а значение аргумента приравнивалось, в зависимости от ситуации, минимально или максимально допустимому значению.

Код программы:

namespace lab3

{

class Second

{

private readonly int l;

public int GetL(){

return l;

}

public Second(int i){

if (i > 99 && i < 1000)

this.l = i;

else

{

Console.WriteLine("Введено неправильное число ");

if (i < 99)

this.l = 100;

else

this.l = 999;

}

}

}

class Third

{

[STAThread]

static void Main(string[] args)

{

Second[] obj1 = new Second[3];

obj1[0] = new Second(66);

Console.WriteLine(obj1[0].GetL());

Console.WriteLine();

obj1[1] = new Second(777);

Console.WriteLine(obj1[1].GetL());

Console.WriteLine();

obj1[2] = new Second(1001);

Console.WriteLine(obj1[2].GetL());

Console.WriteLine();

Console.ReadKey();

}

}

}

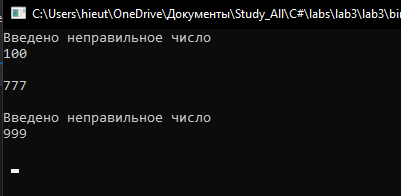


Рисунок 6 – результат работы программы

**Вывод**

В ходе лабораторной работы были изучены принципы работы с конструкторами классов. Рассмотрены основные возможности и ограничения.