КР1 КОНСТРУИРОВАНИЕ ПРОГРАММ И ЯЗЫКИ ПРОГРАМИРОВАНИЯ

12.Перечислите этапы создания проекта консольной программы.

Запускаем Visual Studio

Создание нового проекта: File —» New —> Project ( В русской версии данное меню будет выглядеть так – Файл —» Создать —> Проект ).

В окне **New Project (Новый Проект)** на левой панели **(Project Types)( Типы Проектов)** выберите язык **(Visual С#)** и платформу **(Windows).** На центральной панели выберите вид приложения **Console Application (Консольное Приложение).**

В поле **Name(Имя)** вы можете напечатать вместо установленого по умолчанию имени свое. В поле **Location(Путь)** введите полное имя папки, в которой будет сохранено решение, например, С:Программы. По умолчанию решению приписывается стандартное имя, но его можно заменить на, например Program\_l (Стандартным является ConsoleApplication(Цифра начиная от 1 в зависимости от того сколько приложений вы создали со стандартным именем проекта). Кнопкой **ОК** запускаем процесс создания проекта (и решения).

Среда Visual Studio создает решение, проект приложения и открывает окно редактора с готовой заготовкой исходного кода программы следующего вида

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Program\_l

{

class Program

{

static void Main(string[] args)

{

}

}

}

52 . Опишите отношение клиенты – поставщики.

*Класс*, как не раз отмечалось, играет две роли: он является модулем - архитектурной единицей, и он имеет содержательный смысл, определяя некоторый *тип данных*.

Классы программной системы находятся в определенных отношениях друг с другом. Два основных типа отношений между классами определены в ОО-системах.

· **Клиенты и поставщики** – называется часто **клиентским отношением** или **отношением вложенности (встраивания)**.

**Определение** . Классы A и В находятся в отношении "клиент - поставщик", если одним из полей класса В является *объект* класса А. *Класс* А называется поставщиком класса В, *класс* В называется клиентом класса А.

Следуя этому определению, *объект* класса A "вложен" в *класс* B. *По* этой причине *отношение* "клиент - поставщик" называют также *отношением вложенности* или *встраивания*.

65 Напишите программу, которая определяет число отрицатель­ных чисел во введенной с клавиатуры последовательности (длина по­следовательности неограниченна).

using System;

using System.Collections.Generic;

using System.Text;

namespace ZADANIE65

{

class Program

{

static void Main(string[] args)

{

Console.Title = "Программа определяющая число отрицательных чисел во введенной с клавиатуры последовательности";

Console.WriteLine("Задание 65 - Шифр 29 - \n Второй Курс Могилёвского Государственного Политехнического Колледжа - Сделано Савичем Андреем Олеговичем");

Console.WriteLine("Пожалуйста Ввведите последовательность");

double max = +1;

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

double t = 0.0;

int i = 0;

do

{

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

i ++;

}

while(t < max);

Console.WriteLine("Количество Отрицательных чисел - {0}", i);

Console.ReadLine();

}

}

}

76 . Решите задачу. Дан двумерный массив. Заполните его по строкам с клавиатуры и определите:

количество строк, не содержащих ни одного нулевого элемента;

максимальное из чисел, в заданной строке массива.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

class Program

{

static void Main()

{

Console.Write("Введите количество строк: ");

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

Console.WriteLine();

Console.Write("Введите количество столбцов: ");

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

Console.WriteLine();

int[,] array = new int[firstDimension, secondDimension];

for (int i = 0; i < array.GetLength(0); i++)

{

string enterString = Console.ReadLine();

string[] massiveString = enterString.Split(new Char[] { ' ' });

for (int j = 0; j < massiveString.Length; j++)

{

array[i, j] = int.Parse(massiveString[j]);

}

}

// Перебираем каждый элемент матрицы и если он равен 0, тогда инкрементируем локальную переменную kolElem и

// выводим потом на экран в каждой строке. Если строка не содержит нулевые элементы матрицы, инкрементируем

// локальную переменную kolStr

int kolElem = 0;

int kolStr = 0;

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

{

for (int j = 0; j < secondDimension; ++j)

{

if (array[i, j] == 0)

{

++kolElem;

}

}

Console.WriteLine("Строка {0} содержит {1} нулевых элементов", (i+1), kolElem);

if (kolElem == 0)

{

++kolStr;

}

else

{

kolElem = 0;

}

}

// Выводим на экран локальную переменную kolStr

Console.WriteLine("Количество строк не содержащих нулевые элементы: " + kolStr);

Console.WriteLine();

//Находим максимальное значение в матрице

int maxElem = 0;

int kolVstrech = 0;

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

{

for (int j = 0; j < secondDimension; ++j)

{

if (array[i, j] == maxElem)

{

++kolVstrech;

}

if (array[i, j] > maxElem)

{

maxElem = array[i, j];

kolVstrech = 1;

}

}

}

while (kolVstrech < 2)

{

if (kolVstrech <= 1)

{

Console.WriteLine("Максимальное значение: {0} не повторяется в матрице, поэтому ищем новое максимальное значение, которое встречается два или более раз", maxElem);

int maxElemPrediduschiy = maxElem;

maxElem = 0;

kolVstrech = 0;

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

{

for (int j = 0; j < secondDimension; ++j)

{

if ((array[i, j] == maxElem) && (array[i, j] < maxElemPrediduschiy))

{

++kolVstrech;

}

if ((array[i, j] > maxElem) && (array[i, j] < maxElemPrediduschiy))

{

maxElem = array[i, j];

kolVstrech = 1;

}

}

}

}

}

// Находим максимальное

Console.WriteLine("Максимальный элемент: {0}, встречается {1} раз(а)", maxElem, kolVstrech);

Console.ReadKey();

}

}