|  |  |  |
| --- | --- | --- |
| Защищено:  Гапанюк Ю. Е.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |  | Демонстрация:  Гапанюк Ю. Е.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**Отчет по лабораторной работе №1 по курсу**

**«Парадигмы и конструкции языков программирования»**

#### Тема работы: "Изучение языка C#"

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: | Агапова Анна Денисовна |
| студент группы  ИУ5Ц-52Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Гапанюк Ю.Е. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Москва, МГТУ 2024

**СОДЕРЖАНИЕ**

[Цель лабораторной работы 3](#_Toc176887611)

[Практическое задание 3](#_Toc176887612)

[Листинг программы 4](#_Toc176887613)

[Результат работы программы 7](#_Toc176887614)

## Цель лабораторной работы

Получение практических навыков по C#

## Практическое задание

Разработать программу для решения биквадратного уравнения.

1. ﻿﻿﻿Программа должна быть разработана в виде консольного приложения на языке С#.
2. ﻿﻿﻿Программа осуществляет ввод с клавиатуры коэффициентов А, В, С, вычисляет дискриминант и корни уравнения (в зависимости от дискриминанта).
3. ﻿﻿﻿Если коэффициент А, В, С введен некорректно (не приводится к действительному числу), то необходимо проигнорировать некорректное значение и ввести коэффициент повторно.
4. ﻿﻿﻿Корни уравнения выводятся зеленым цветом. Если корней нет, то сообщение выводится красным цветом.
5. ﻿﻿﻿Коэффициенты А, В, С задаются в виде параметров командной строки.
6. Если они не указаны, то вводятся с клавиатуры в соответствии с пунктом 2. Проверка из пункта 3 в этом случае производится для параметров командной строки без повторного ввода с клавиатуры.

## Листинг программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace LR\_1

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Данная программа предназначена для решения биквадратного уравнения\n");

// Предполагаем, что коэффициенты A, B, C инициализируются нулевыми значениями

double A = 0, B = 0, C = 0;

// Пытаемся получить коэффициенты из параметров командной строки

if (args.Length == 3)

{

//присваивем коэффициентам A, B, C значения из аргументов командной строки

double.TryParse(args[0], out A);

double.TryParse(args[1], out B);

double.TryParse(args[2], out C);

Console.WriteLine("А В С получены из командной строки\n");

Console.WriteLine("(" + args[0] + ")x^4 + (" + args[1] + ")x^2 + (" + args[2] + ") =0\n");

//если таковые заданы (проверим и если нет - попросим пользователя ввести их вручную)

if (!double.TryParse(args[0], out A) || !double.TryParse(args[1], out B) || !double.TryParse(args[2], out C))

{

Console.WriteLine("Необходимо ввести вещественное число. Вводите коэффициенты заново.\n");

}

}

else //если аргументы не заданы, их вводит пользователь

{

A = ReadDouble("Введите коэффициент A: ");

B = ReadDouble("Введите коэффициент B: ");

C = ReadDouble("Введите коэффициент C: ");

}

double discriminant = (B \* B) - (4 \* A \* C); //вычислим дискриминант

Console.WriteLine("Дискриминант = " + discriminant + "\n");

// Рассмотрим возможные значения дискриминанта

if (discriminant > 0)

{

double TempVal1 = (-B + Math.Sqrt(discriminant)) / (2 \* A);

double TempVal2 = (-B - Math.Sqrt(discriminant)) / (2 \* A);

Console.ForegroundColor = ConsoleColor.Green;

if (TempVal1 >= 0)

{

double root1 = Math.Sqrt(TempVal1);

double root2 = -Math.Sqrt(TempVal1);

if (root1 == 0)

{

Console.WriteLine("Корень 1: {0:F2}", -root1);

}

else if (root1 == root2)

{

Console.WriteLine("Корень 1: {0:F2}", root1);

}

else

{

Console.WriteLine("Корень 1: {0:F2}", root1);

Console.WriteLine("Корень 2: {0:F2}", root2);

}

}

if (TempVal2 >= 0)

{

double root3 = Math.Sqrt(TempVal2);

double root4 = -Math.Sqrt(TempVal2);

if (root3 == -0)

{

Console.WriteLine("Корень 4: {0:F2}", -root3);

}

else if (root3 == root4)

{

Console.WriteLine("Корень 3: {0:F2}", root3);

}

else

{

Console.WriteLine("Корень 3: {0:F2}", root3);

Console.WriteLine("Корень 4: {0:F2}", root4);

}

}

Console.ResetColor();

}

else if (discriminant == 0)

{

double TempVal = (-B) / (2 \* A);

ProcessSingleRoot(TempVal);

}

else

{

// Нет корней => выводим сообщение красным цветом

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Корней нет\n");

Console.ResetColor();

}

}

/// <summary>

/// Ввод вещественного числа с проверкой корректности ввода

/// </summary>

/// <param name="message">Подсказка при вводе</param>

static double ReadDouble(string message)

{

string resultString;

double resultDouble;

bool flag;

do

{

Console.Write(message);

resultString = Console.ReadLine();

flag = double.TryParse(resultString, out resultDouble);

if (!flag)

{

Console.WriteLine("Необходимо ввести вещественное число");

}

}

while (!flag);

return resultDouble;

}

static void ProcessSingleRoot(double TempVal)

{

Console.ForegroundColor = ConsoleColor.Green;

if (TempVal >= 0)

{

double root1 = Math.Sqrt(TempVal);

double root2 = -Math.Sqrt(TempVal);

Console.WriteLine("Корень 1: {0:F2}", root1);

Console.WriteLine("Корень 2: {0:F2}\n", root2);

}

Console.ResetColor();

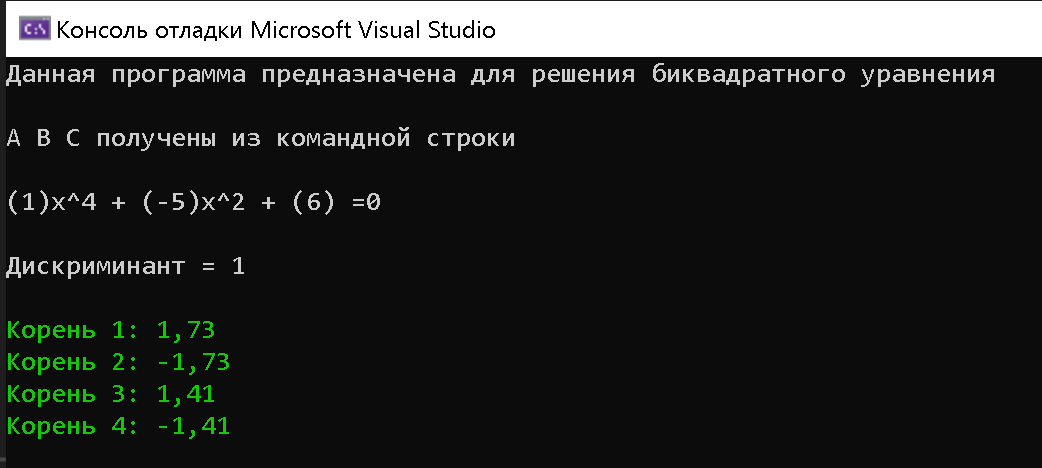
}

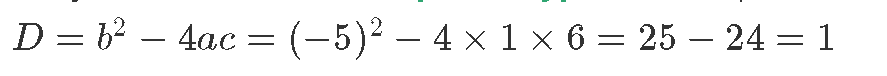
}

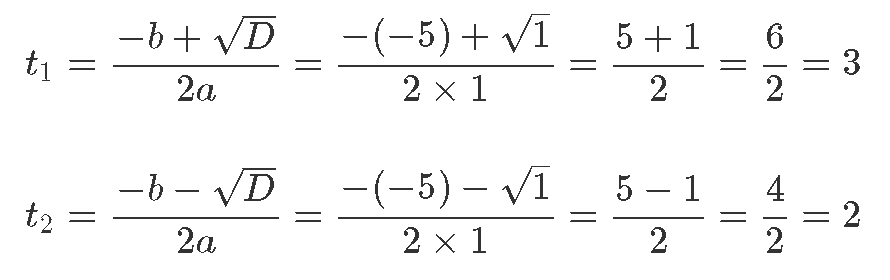
}

## Результат работы программы

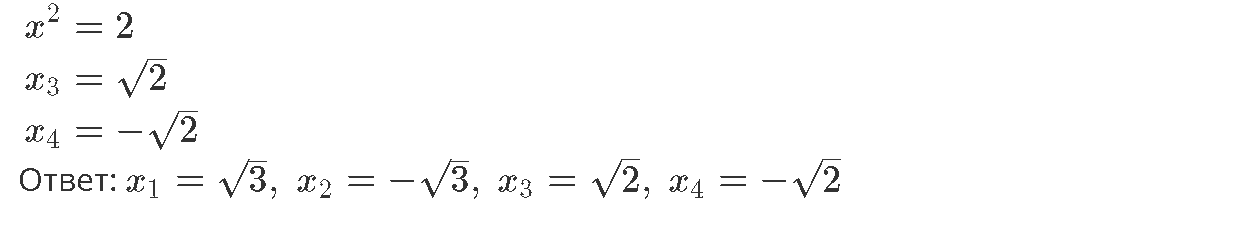




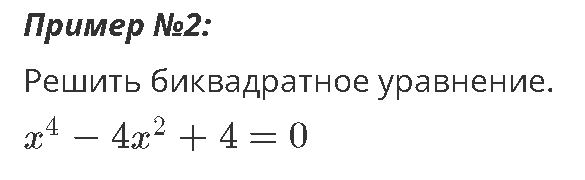


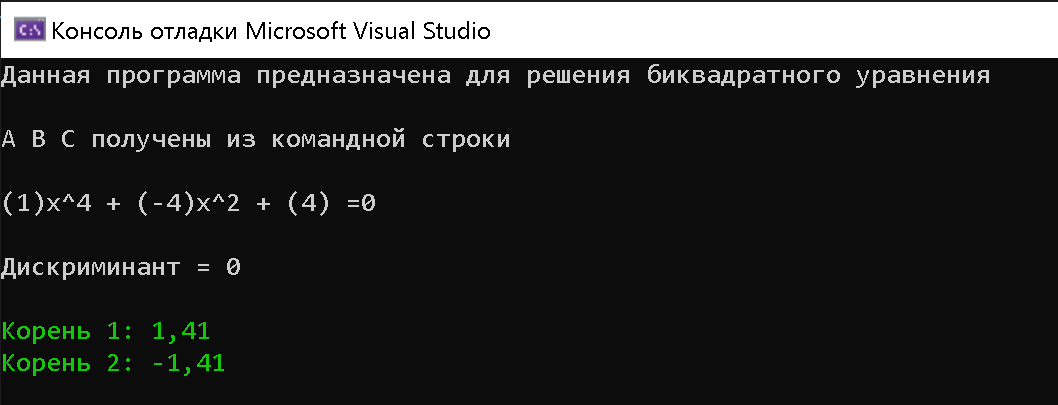


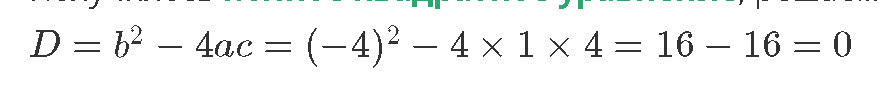
Вернемся к старой переменной:



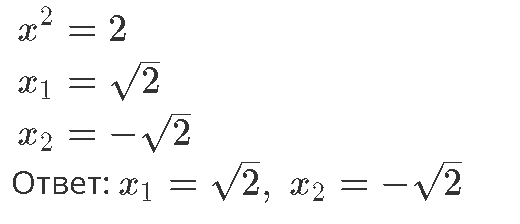
 



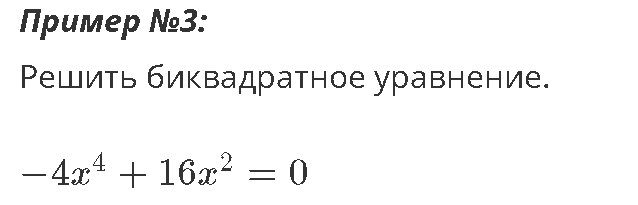


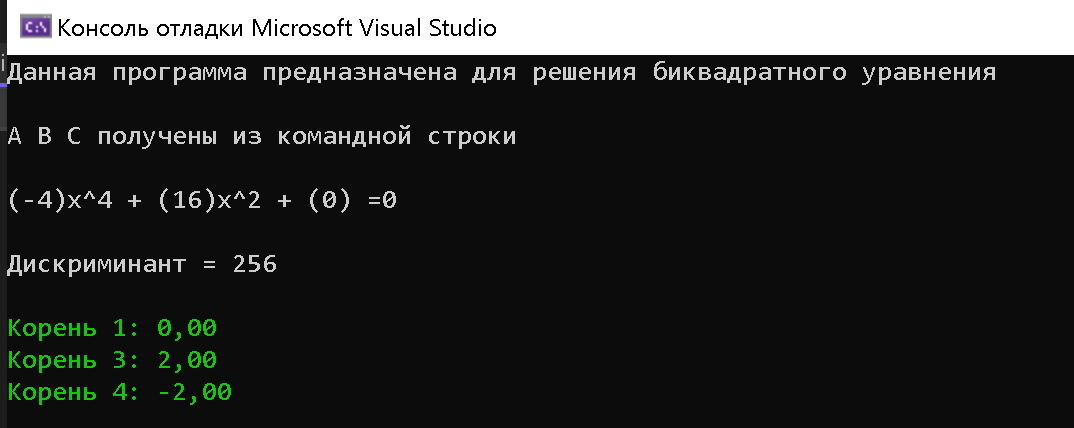


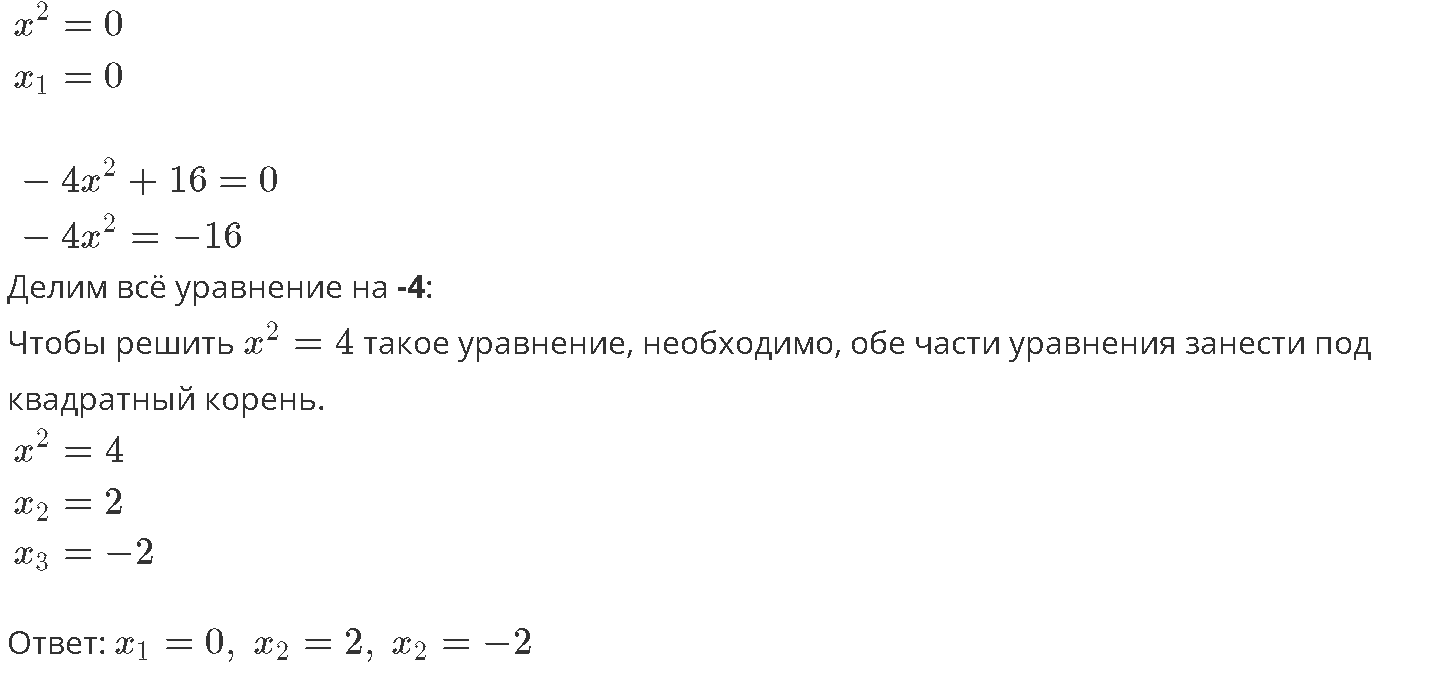
Вернемся к старой переменной:

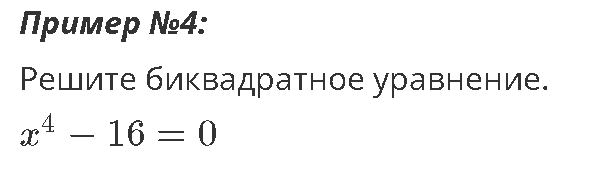


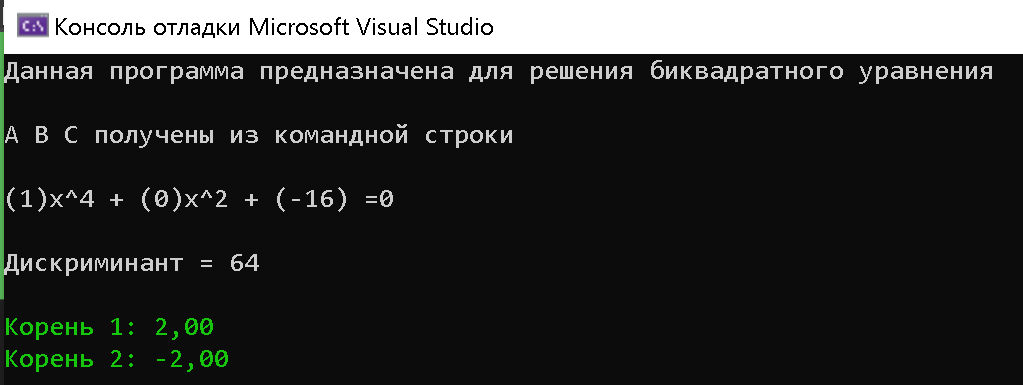


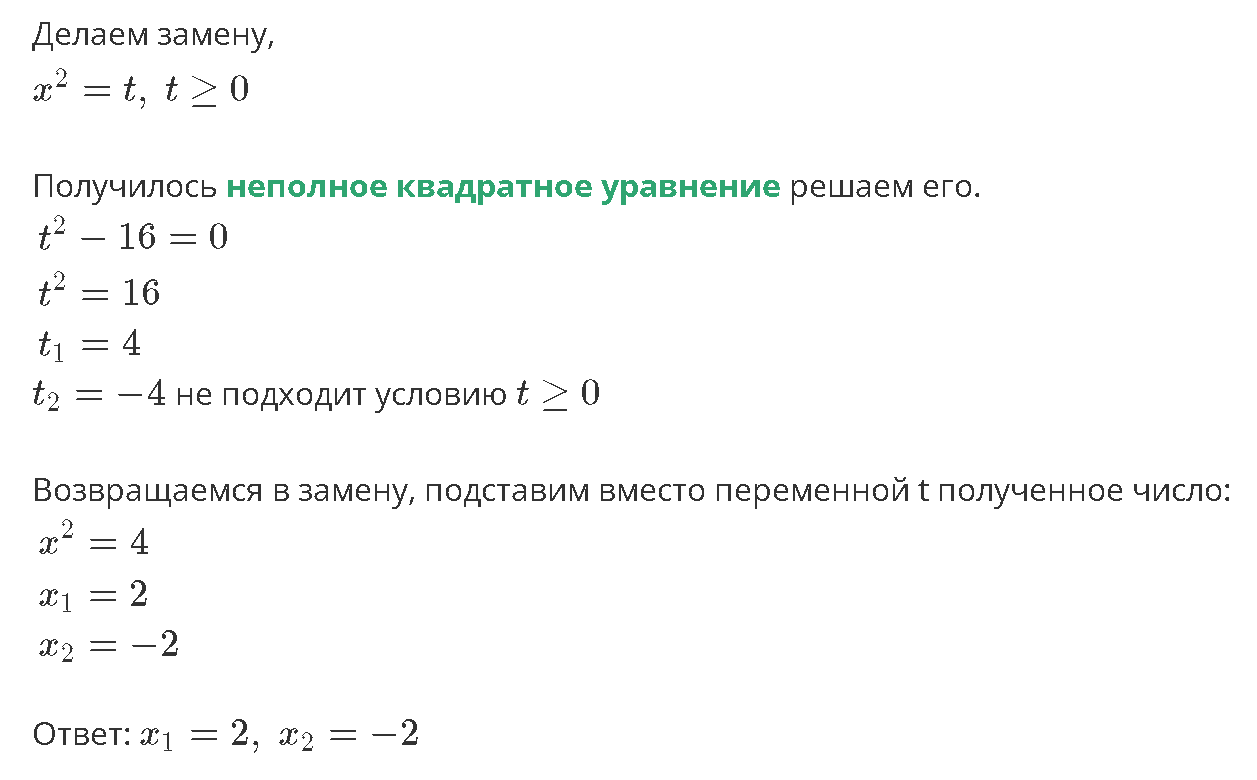


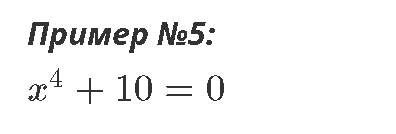


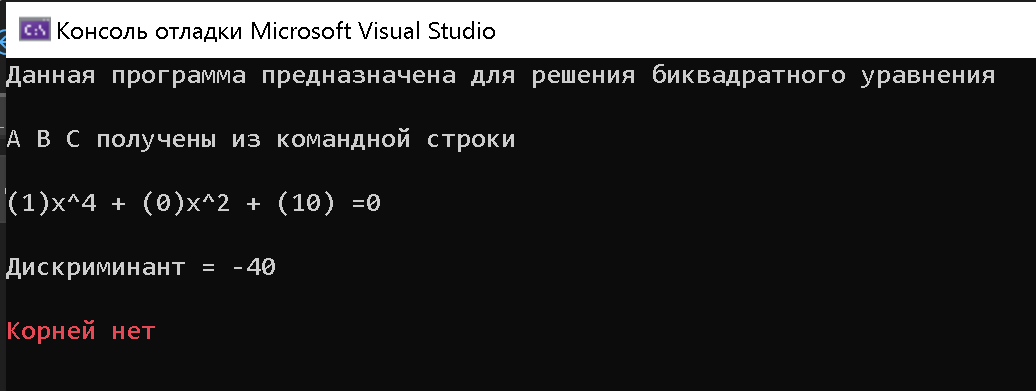


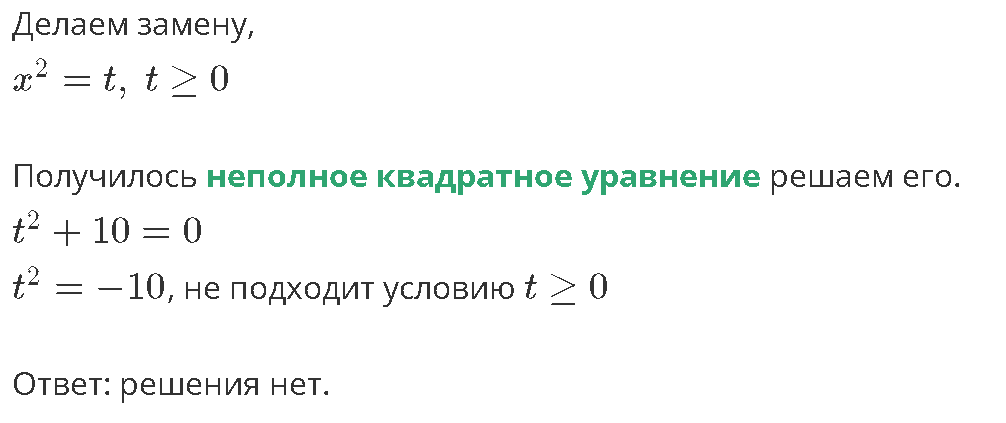






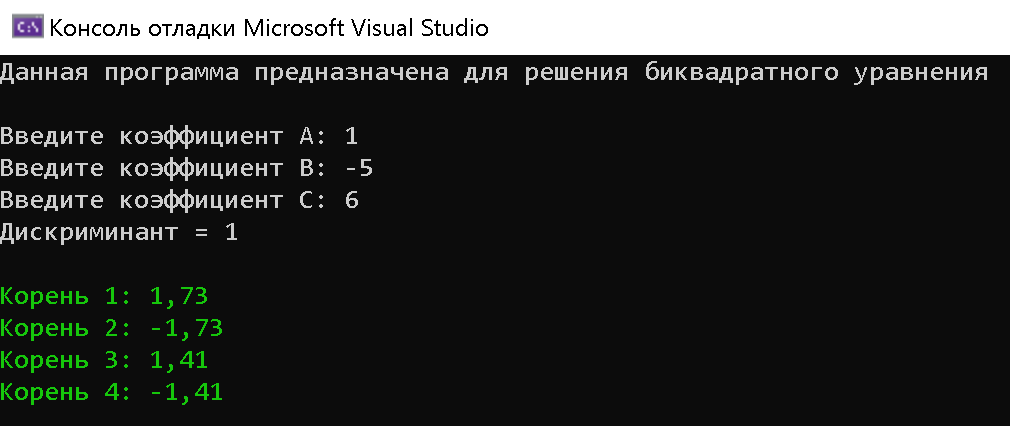






Проверка пункта (6):

Если они (параметры командной строки) не указаны, то вводятся с клавиатуры в соответствии с пунктом 2. Проверка из пункта 3 в этом случае производится для параметров командной строки без повторного ввода с клавиатуры.

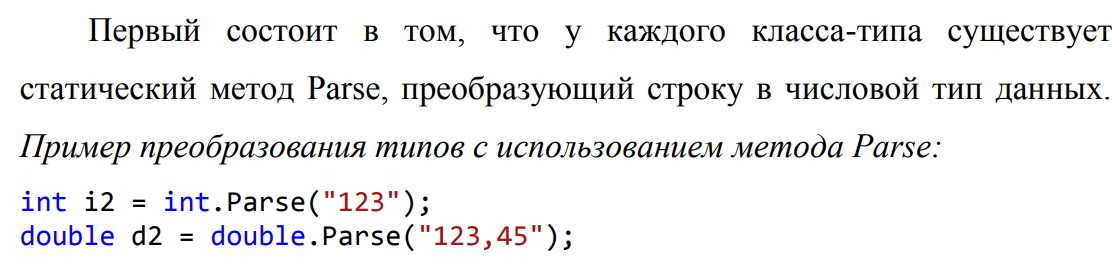


## Ответы на контрольные вопросы

**1) Какими способами можно преобразовать значение строкового типа в значение числового типа?**

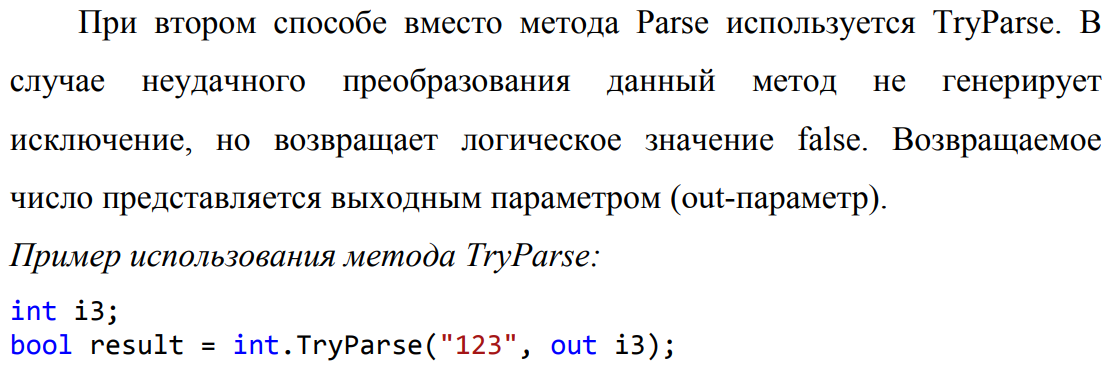
1.методом Parse, преобразующий строку в числовой тип данных.

Пример:



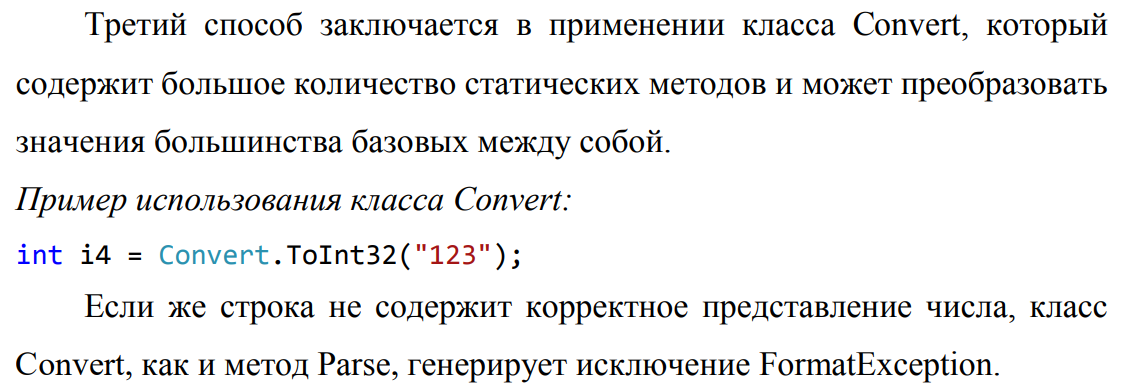
2. TryParse. В случае неудачного преобразования данный метод не генерирует исключение, но возвращает логическое значение false. Возвращаемое число представляется выходным параметром (out-параметр)

Пример:

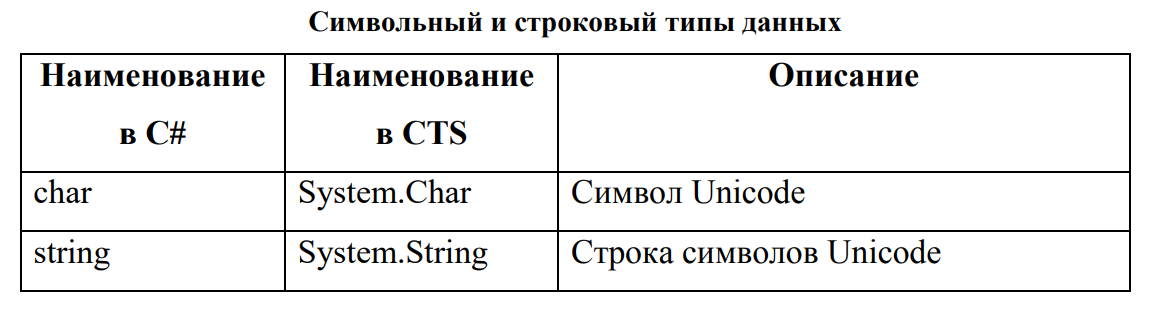


3.С помощью класса Convert

Пример:



**Какие символьные и строковые типы данных существуют в С#?**



**Какие средства консольного ввода/вывода существуют в С#?**

Ввод-вывод в консоль осуществляется посредством класса Console, у которого предусмотрены статические методы ввода-вывода данных:

• Console.WriteLine – вывод данных с переводом строки;

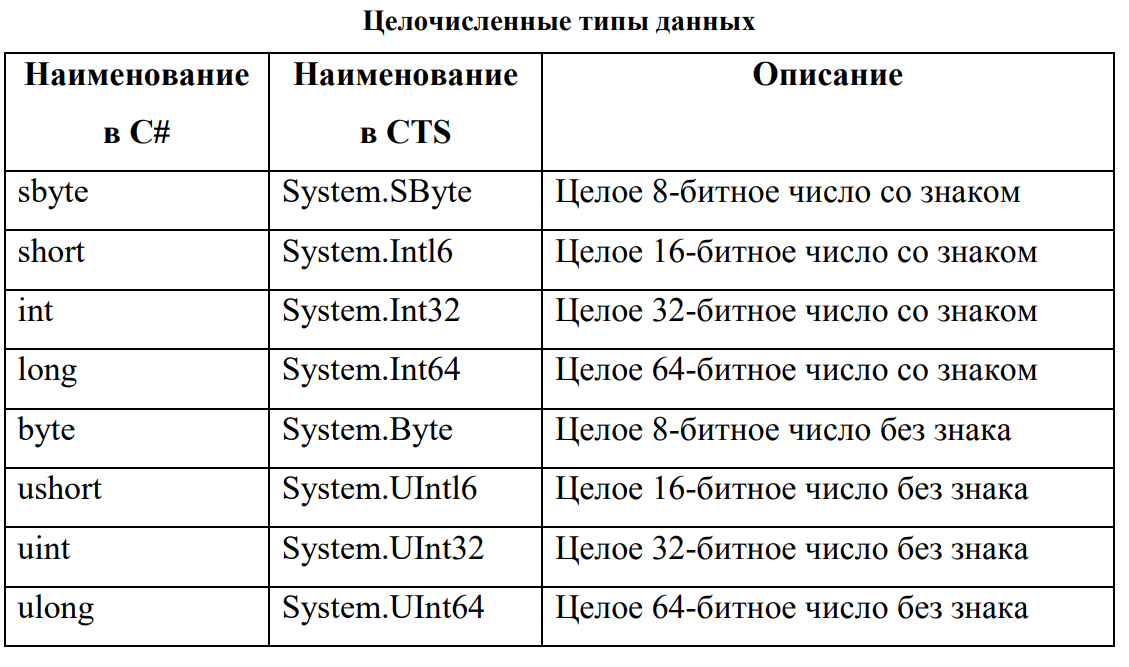
• Console.Write – вывод данных без перевода строки;

• Console.Read – чтение текущего символа;

• Console.ReadKey – чтение текущего символа или функциональной клавиши;

• Console.ReadLine – чтение строки до нажатия ввода.

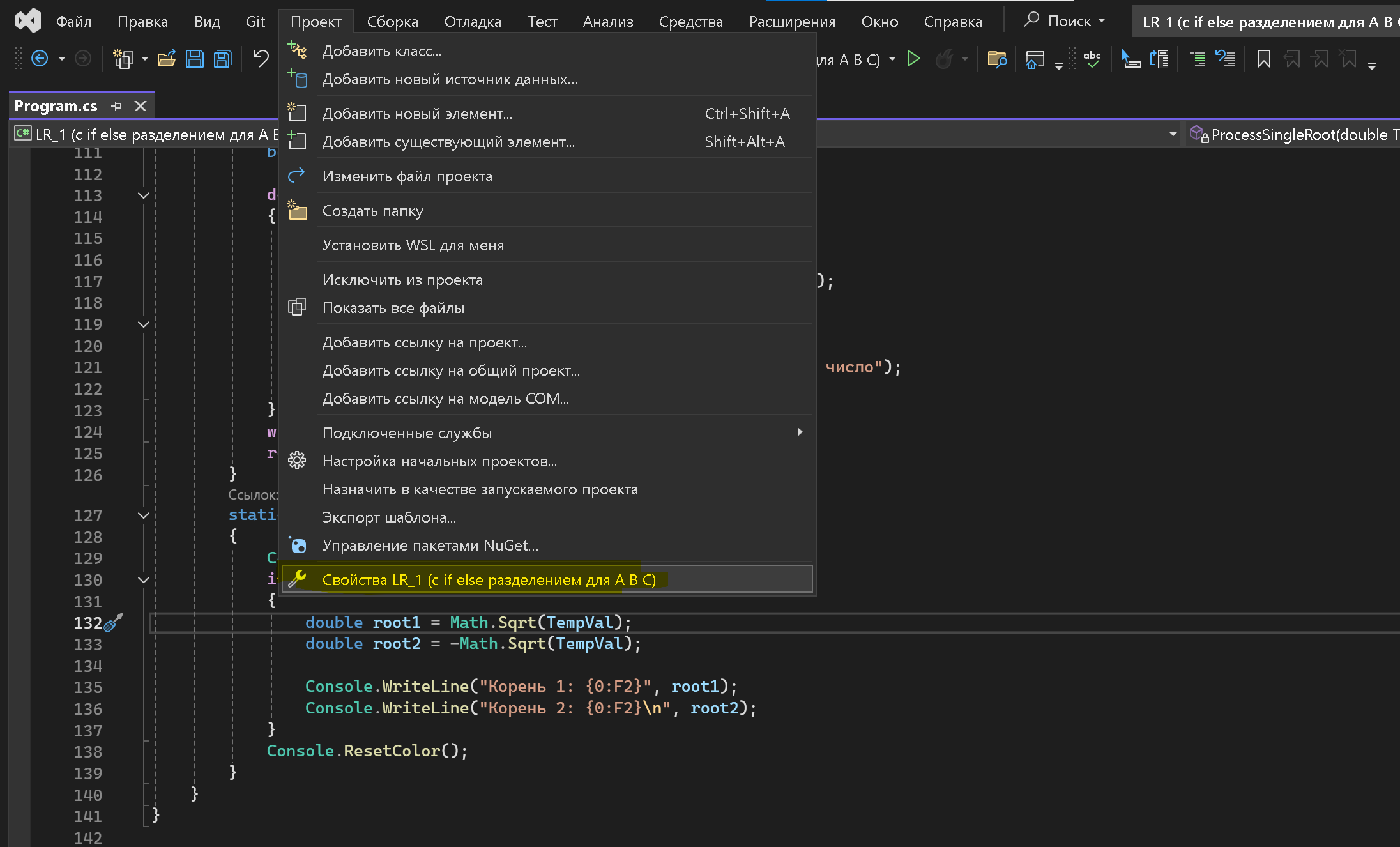
**﻿﻿﻿Какие целочисленные типы данных существуют в С#?**

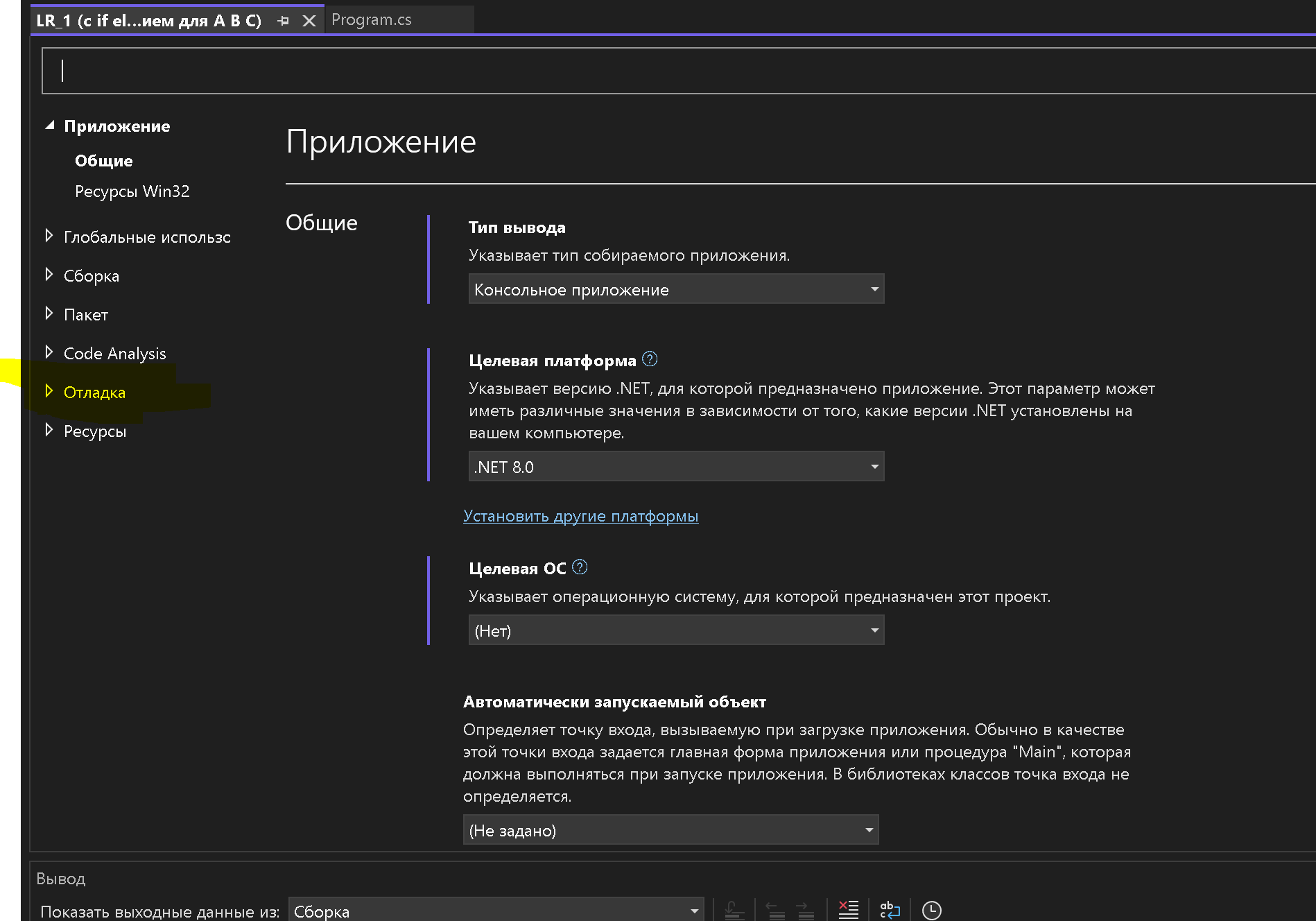


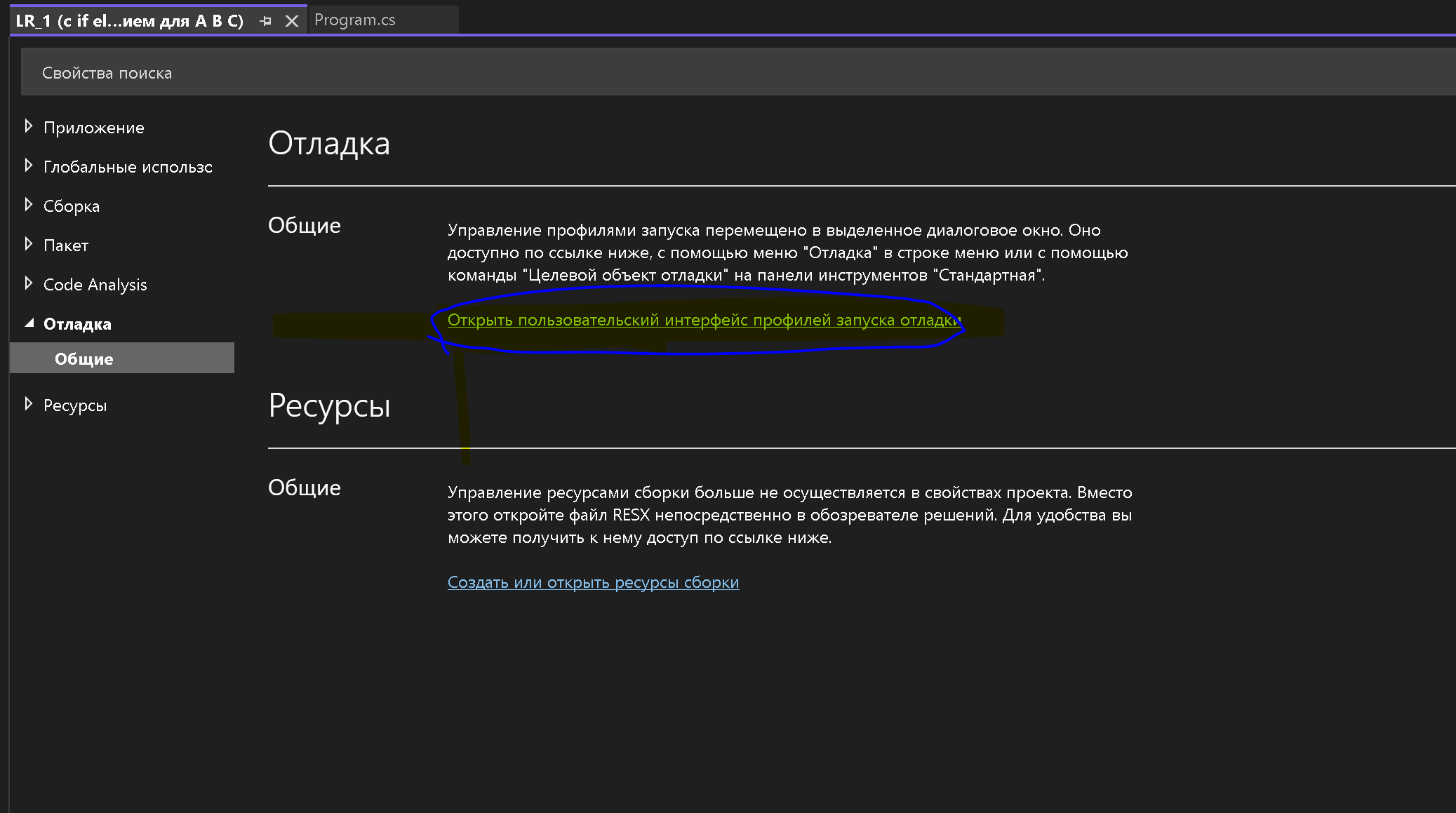
**﻿﻿﻿**

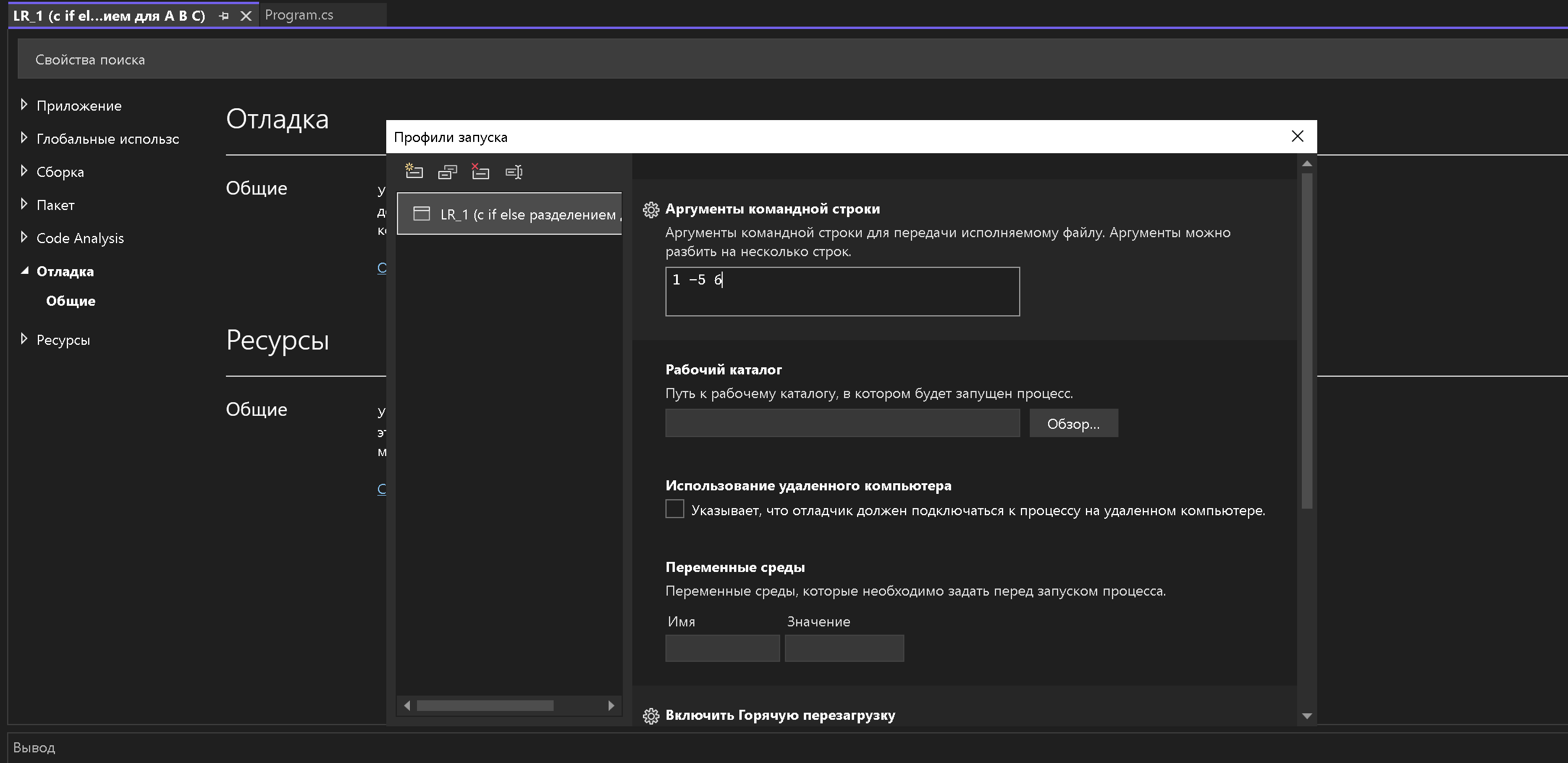
**Как задаются и обрабатываются аргументы командной строки в консольном приложении?**

Проще пояснить на практике:

****







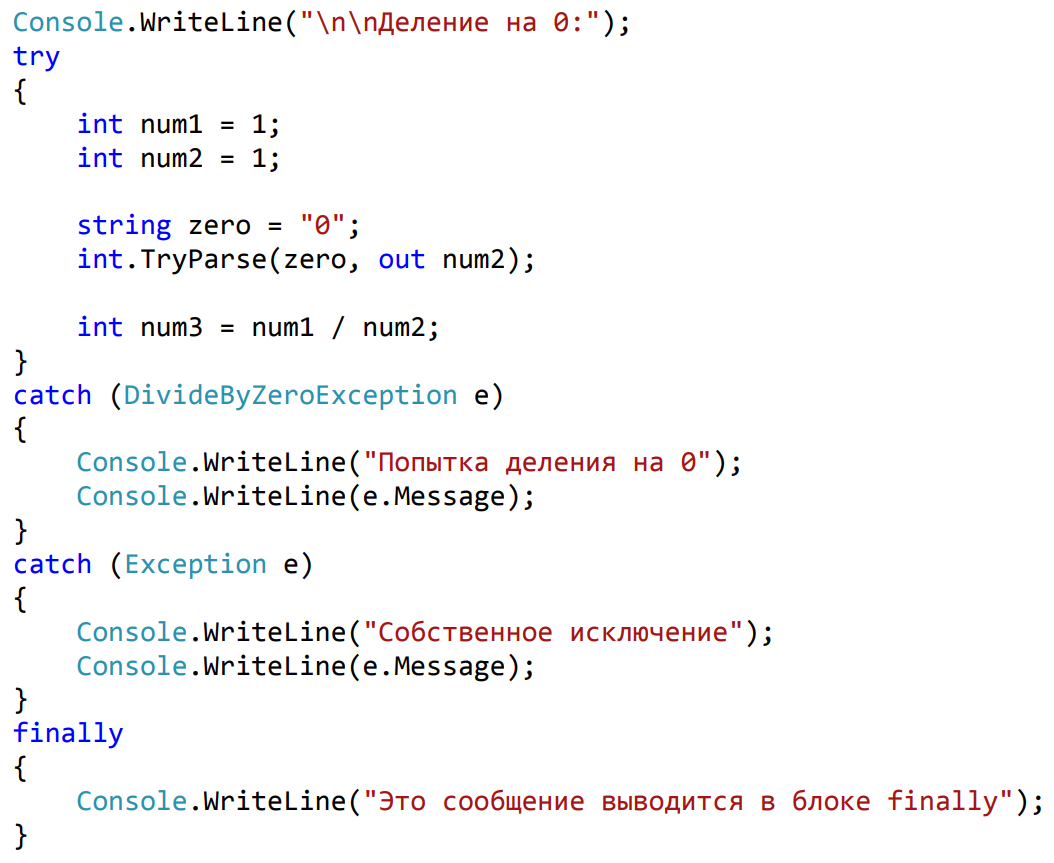
**﻿﻿﻿Как работает механизм обработки исключений в С#?**

В блоке try записываются операторы, которые могут привести к возникновению ошибок.

Блоков catch может быть несколько, каждый из них осуществляет перехват исключения определенного вида. Необходимо, чтобы первыми располагались наиболее детальные исключения, те, которые размещаются на наиболее детальном уровне в дереве наследования от класса Exception.

Необязательный блок finally располагается после блоков catch. Действия в блоке finally выполняются всегда, вне зависимости от того произошло исключение в блоке try или нет.

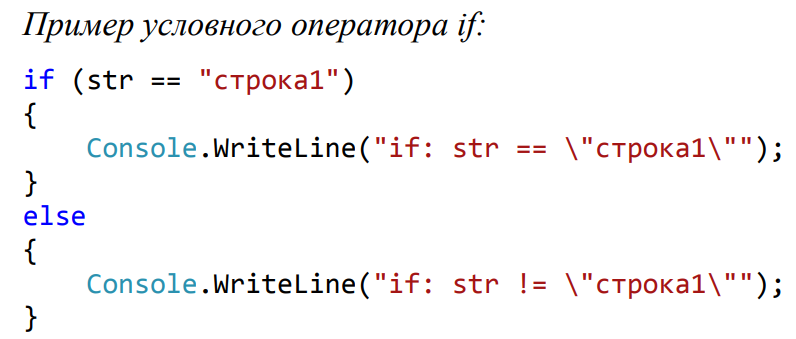
Пример:

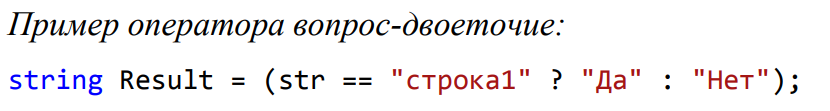


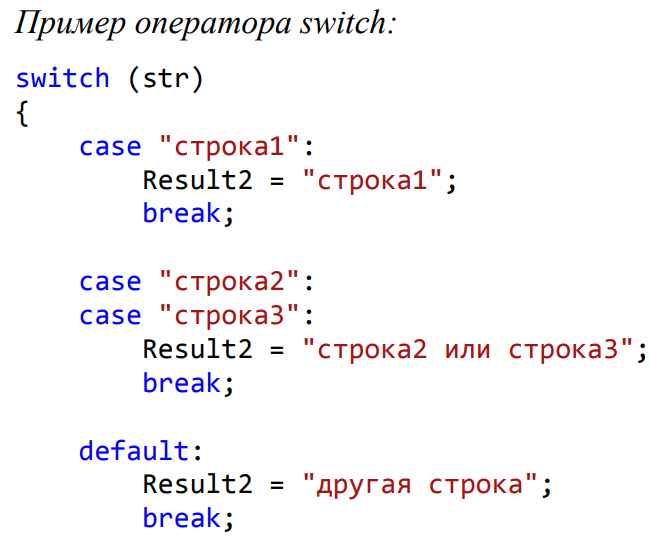
**﻿﻿﻿**

**Какие условные операторы существуют в С#?**

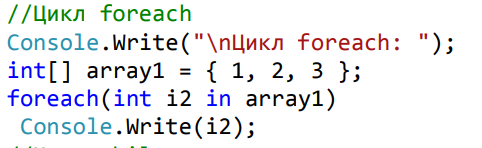
Операторы if-else, вопрос-двоеточие, switch-case.





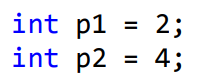


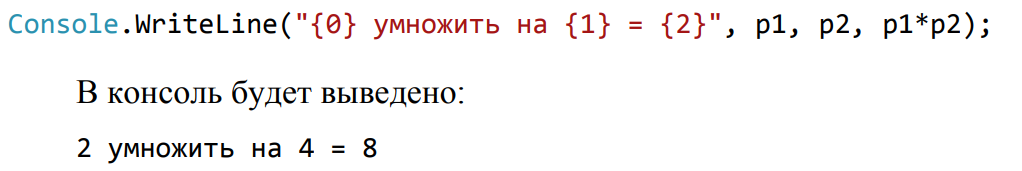
**﻿﻿﻿Как работает цикл foreach?**



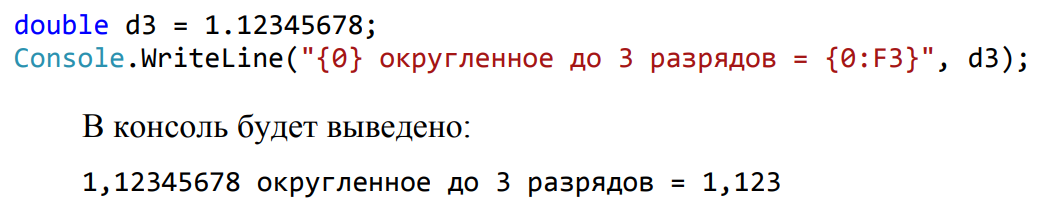
**﻿﻿﻿Как работает форматированный вывод в консоль?**

Пример вывода с обычным форматированием:



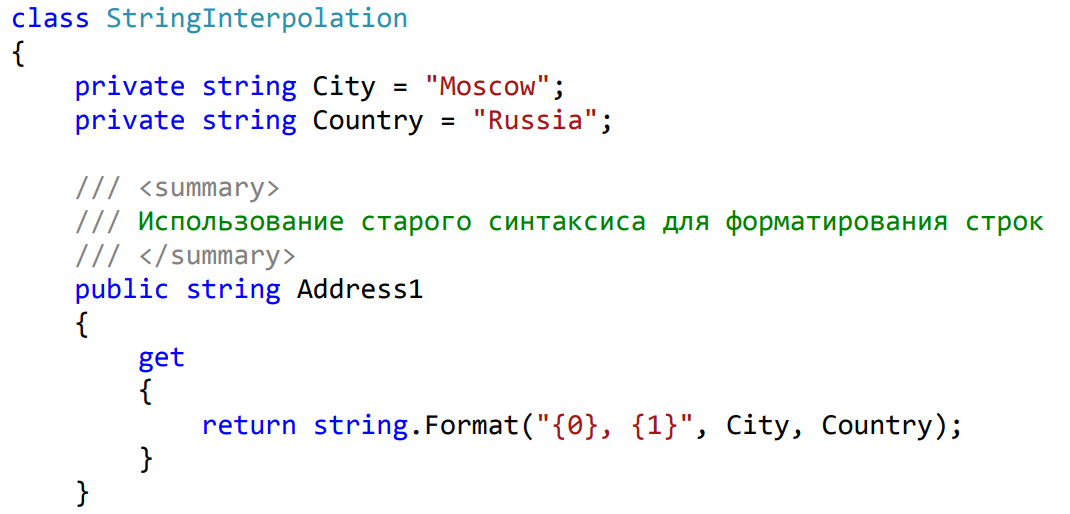


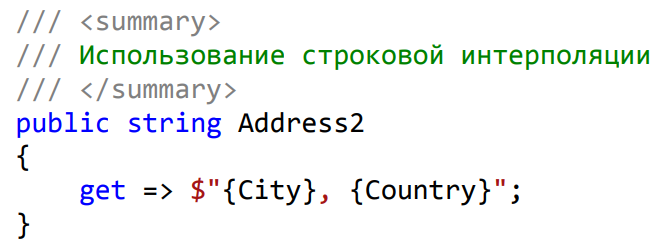
Пример вывода с округлением разрядов:

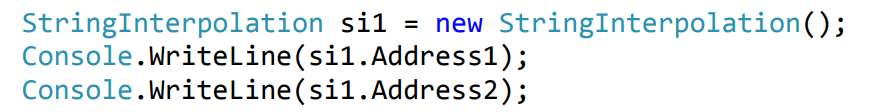


В данном примере выражение {0:F3} означает, что нулевой параметр нужно вывести в виде числа с плавающей точкой, округлив до 3 знаков после разделителя разрядов.

В C# также можно организовать строковую интерполяцию:







В консоль будет выведено:

**Moscow, Russia**

**Moscow, Russia**