

## Лаб: Условни конструкции

Задачи за упражнение в клас и за домашно към курса ["Основи на програмирането" @ СофтУни](#).

Тествайте решенията си в Judge системата: <https://judge.softuni.bg/Contests/2369>

### • Празно Visual Studio решение (Blank Project)

Създайте празно решение (**Blank Solution**) във Visual Studio. Решенията (solutions) във Visual Studio обединяват **група проекти**. Тази възможност е изключително удобна, когато искаме да работим по няколко проекта и бързо да превключваме между тях или искаме да обединим логически няколко взаимосвързани проекта.

В настоящото практическо занимание ще използваме **Blank Solution с няколко проекта**, за да организираме решенията на задачите от упражненията – всяка задача в отделен проект и всички проекти в общ solution.

- Стартирайте Visual Studio.
- Създайте нов проект: [Create a new project].

# Visual Studio 2019

## Open recent

As you use Visual Studio, any projects, folders, or files that you open will show up here for quick access.

You can pin anything that you open frequently so that it's always at the top of the list.

## Get started



### Clone or check out code

Get code from an online repository like GitHub or Azure DevOps



### Open a project or solution

Open a local Visual Studio project or .sln file



### Open a local folder

Navigate and edit code within any folder



### Create a new project

Choose a project template with code scaffolding to get started

[Continue without code →](#)

- Изберете [Black solution], ако не го виждате, в търсачката изпишете ["Blank solution"].


## Create a new project

### Recent project templates

 Console App (.NET Core)


 Blank Solution

C#

blank solution 

Language 

Platform 

Project type 



Blank Solution

Create an empty solution containing no projects

Other

Not finding what you're looking for?  
[Install more tools and features](#)

Back

Next

- Задайте подходящо име на проекта, например **"Conditional-Statements"**:

## Configure your new project

Blank Solution Other

Project name

Conditional-Statements

Location

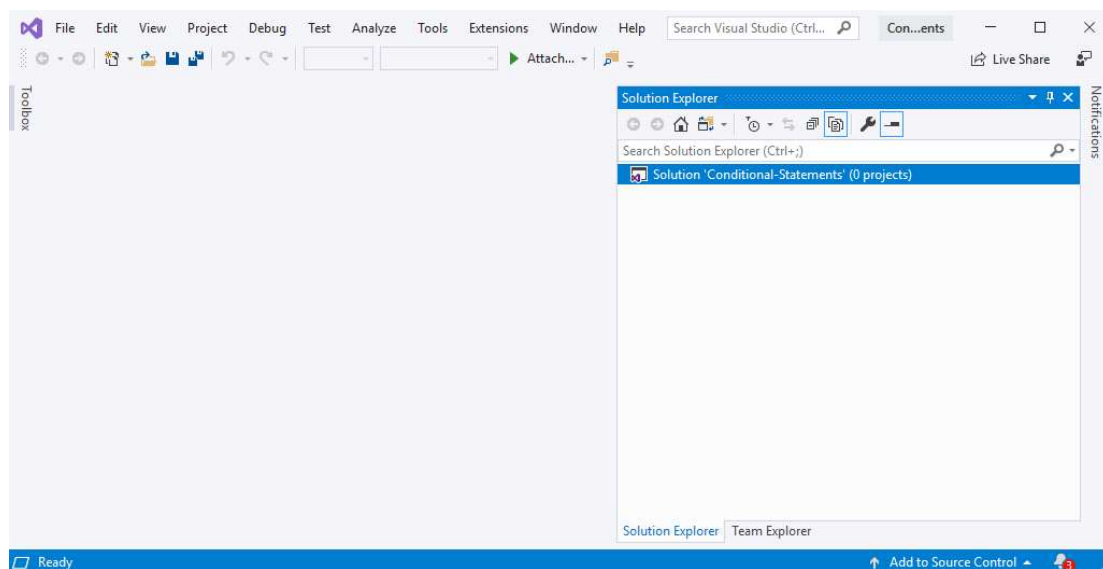
C:\Users\Computer\source\repos

Solution name ⓘ

Conditional-Statements

Back

Сега имате създаден **празен Visual Studio Solution** (с 0 проекта в него):



Целта на този blank solution е да добавяте в него **по един проект за всяка задача** от упражненията.

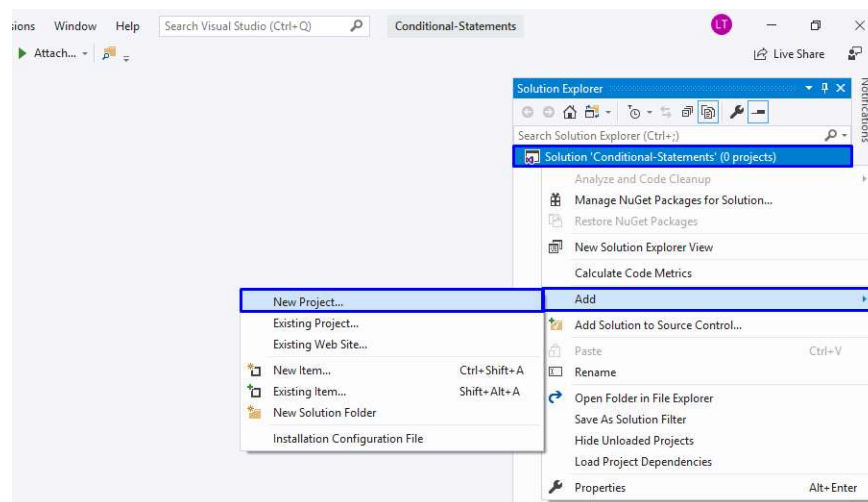
## • Проверка за отлична оценка

Първата задача от тази тема е да се напише **конзолна програма**, която **чете оценка** (десетично число), въведена от потребителя и отпечатва **"Excellent!"**, ако оценката е **5.50** или по-висока.

вход	изход	вход	изход	вход	изход	вход	изход
6	Excellent!	5	(няма изход)	5.50	Excellent!	5.49	(няма изход)

## Насоки:

- Създайте нов **C# конзолен проект** с име **"ExcellentResult"** в **Blank Project** с име **" Conditional-Statements "**, като натиснем с десен бутон на мишката в/у **Solution Conditional-Statements-> add -> New Project....**:



## Add a new project

### Recent project templates

Console App (.NET Core)


C#


Search for templates (Alt+S)


Language


Platform


Project type


**Console App (.NET Core)**  
A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.  
C# Linux macOS Windows Console

**ASP.NET Core Web Application**  
Project templates for creating ASP.NET Core applications for Windows, Linux and macOS using .NET Core or .NET Framework. Create web apps with Razor Pages, MVC, and Blazor, or Single Page Apps (SPA) using Angular, React, or React + Redux. Also create Web APIs, gRPC Services, and Worker Services.  
C# Windows Linux macOS Web

**WPF App (.NET Framework)**  
Windows Presentation Foundation client application  
C# Windows Desktop

**Class Library (.NET Standard)**  
A project for creating a class library that targets .NET Standard.  
C# Android iOS Linux macOS Windows Library

**Azure Functions**  
A template to create an Azure Function project.  
C# Azure Cloud

**Empty Project**

Next

## Configure your new project

Console App (.NET Core) C# Linux macOS Windows Console

Project name

ExcellentResult

Location

C:\Users\Computer\source\repos

Back

Create

- Вече имате създаден клас със **Main** метод

```

namespace ExcellentResult
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
        }
    }
}

```

- Отидете в тялото на метода **Main(string[] args)** (между къдравите скоби). Създайте една променлива, в която да запазите **реално число** – оценката, което ще прочетете от конзолата:

```

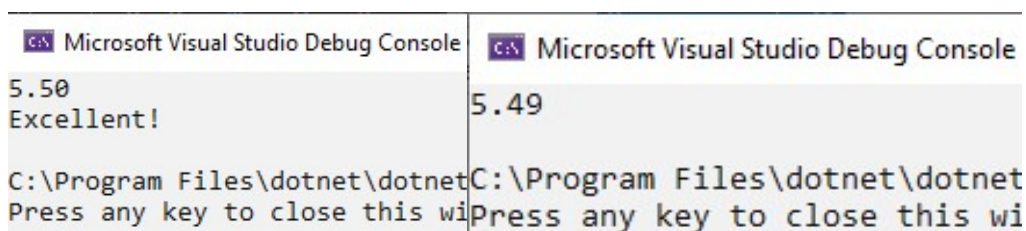
namespace ExcellentResult
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            double grade = double.Parse(Console.ReadLine());
        }
    }
}

```

- Направете проверка за стойността на оценката. Ако тя е по-голяма или равна на 5.50 отпечатайте изхода по условие:

```
static void Main(string[] args)
{
    double grade = double.Parse(Console.ReadLine());
    if (grade >= 5.50)
    {
        Console.WriteLine("Excellent!");
    }
}
```

- Стартирайте програмата с **Ctrl + F5** и я **тествайте** с различни входни стойности:



## • Намиране на по-голямото число

Да се напише програма, която чете **две цели числа** въведени от потребителя и отпечатва **по-голямото** от двете.

### Примерен вход и изход

ВХОД	ИЗХОД	ВХОД	ИЗХОД	ВХОД	ИЗХОД	ВХОД	ИЗХОД
5	5	3	5	10	10	-5	5
3		5		10		5	

### Насоки:

- Прочетете две цели числа от конзолата:

```
static void Main(string[] args)
{
    int num1 = int.Parse(Console.ReadLine());
    int num2 = int.Parse(Console.ReadLine());
}
```

- Сравнете, дали първото число **num1** е по-голямо от второто **num2**. Отпечатайте по-голямото число.



```

if (num1 > num2)
{
    Console.WriteLine(num1);
}
else
{
    Console.WriteLine(num2);
}

```

## • Четно или нечетно

Да се напише програма, която чете **цяло число** въведено от потребителя и отпечатва на конзолата, дали е **четно** или **нечетно**.

### Примерен вход и изход

вход	изход		вход	изход		вход	изход		вход	изход
2	even		3	odd		25	odd		1024	even

### Насоки:

- Първо добавете **нов конзолен проект** към съществуващия проект
- Прочетете едно цяло число от конзолата:

```

class Program
{
    0 references
    static void Main(string[] args)
    {
        int num = int.Parse(Console.ReadLine());
    }
}

```

- Проверете, дали числото е четно, като използвате модуло оператора с 2 и проверите, дали има остатък от целочисленото деление. Отпечатайте изхода по условие – текста **"even"**:

```
static void Main(string[] args)
{
    int num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("even");
    }
}
```

- В противен случай отпечатайте "odd":

```
static void Main(string[] args)
{
    int num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("even");
    }
    else
    {
        Console.WriteLine("odd");
    }
}
```

## • Число от 100 до 200

Да се напише програма, която **чете цяло число**, въведено от потребителя и проверява, дали е **под 100**, **между 100 и 200** или **над 200**. Да се отпечата съответно съобщения, като в примерите по-долу:

### Примерен вход и изход

вход	изход	вход	изход	вход	изход
95	Less than 100	120	Between 100 and 200	210	Greater than 200

## • Познай паролата

Да се напише програма, която **чете парола** (един ред с произволен текст), въведена от потребителя и проверява, дали въведеното **съвпада** с фразата "s3cr3t!P@ssw0rd".

При съвпадение да се изведе "Welcome". При несъвпадение да се изведе "Wrong password!".

## Примерен вход и изход

вход	изход	вход	изход	вход	изход
qwerty	Wrong password!	s3cr3t!P@ssw0rd	Welcome	s3cr3t!p@ss	Wrong password!

## • Лица на фигури

Да се напише програма, в която потребителят **въвежда вида и размерите на геометрична** фигура и пресмята лицето ѝ. Фигурите са четири вида: квадрат (**square**), правоъгълник (**rectangle**), кръг (**circle**) и триъгълник (**triangle**). На първия ред на входа се чете вида на фигурата (**square, rectangle, circle** или **triangle**).

- Ако фигурата е **квадрат**, на следващия ред се чете едно число - дължина на страната му.
- Ако фигурата е **правоъгълник**, на следващите два реда четат две числа - дължините на страните му.
- Ако фигурата е **кръг**, на следващия ред се чете едно число - радиусът на кръга.
- Ако фигурата е **триъгълник**, на следващите два реда четат две числа - дължината на страната му и дължината на височината към нея.

Резултатът да се закръгли до **3 цифри след десетичната точка**.

## Примерен вход и изход

вход	изход	вход	изход	вход	изход	вход	изход
square	25.000	rectangle	17.500	circle	113.097	triangle	45.000
5		7		6		4.5	
		2.5				20	

## Примерна изпитна задача

## • Магазин за детски играчки

Петя има магазин за детски играчки. Тя получава голяма поръчка, която трябва да изпълни. С парите, които ще спечели иска да отиде на екскурзия. Да се напише програма, която пресмята печалбата от поръчката.

Цени на играчките:

- Пъзел - 2.60 лв.
- Говореща кукла - 3 лв.
- Плюшено мече - 4.10 лв.

- Миньон - 8.20 лв.
- Камионче - 2 лв.

Ако поръчаните играчки са **50 или повече** магазинът прави **отстъпка 25% от общата цена**. От спечелените пари Петя трябва да даде **10% за наема** на магазина. Да се пресметне дали парите ще ѝ стигнат да отиде на екскурзия.

От конзолата се четат **6 реда**:

- Цена на екскурзията - реално число в интервала [1.00 ... 10000.00]
- Брой пъзели - цяло число в интервала [0... 1000]
- Брой говорещи кукли - цяло число в интервала [0 ... 1000]
- Брой плюшени мечета - цяло число в интервала [0 ... 1000]
- Брой миньони - цяло число в интервала [0 ... 1000]
- Брой камиончета - цяло число в интервала [0 ... 1000]

На конзолата се отпечатва:

- Ако парите са достатъчни се отпечатва:
  - "Yes! {оставащите пари} lv left."
- Ако парите НЕ са достатъчни се отпечатва:
  - "Not enough money! {недостигащите пари} lv needed."

Резултатът трябва да се форматира до втория знак след десетичната запетая.

## Примерен вход и изход

Вход	Изход	Обяснения
40.8 20 25 30 50 10	Yes! 418.20 lv left.	<p>Сума: <math>20 * 2.60 + 25 * 3 + 30 * 4.10 + 50 * 8.20 + 10 * 2 = 680</math> лв.</p> <p>Брой на играчките: <math>20 + 25 + 30 + 50 + 10 = 135</math></p> <p><math>135 &gt; 50 \Rightarrow</math> 25% отстъпка; 25% от 680 = 170 лв. отстъпка</p> <p>Крайна цена: <math>680 - 170 = 510</math> лв.</p> <p>Наем: 10% от 510 лв. = 51 лв.</p> <p>Печалба: <math>510 - 51 = 459</math> лв.</p> <p><math>459 &gt; 40.8 \Rightarrow 459 - 40.8 = 418.20</math> лв. остават</p>
Вход	Изход	Обяснения
320 8 2 5 5 1	Not enough money! 238.73 lv needed.	<p>Сума: 90.3 лв.</p> <p>Брой на играчките: 21</p> <p><math>21 &lt; 50 \Rightarrow</math> няма отстъпка</p> <p>Наем: 10% от 90.3 = 9.03 лв.</p> <p>Печалба: <math>90.3 - 9.03 = 81.27</math> лв.</p> <p><math>81.27 &lt; 320 \Rightarrow 320 - 81.27 = 238.73</math> лв. не достигат</p>

