# Упражнения: Логически изрази и проверки

## Празно Visual Studio решение (Blank Solution)

Създайте празно решение (**Blank Solution**) във Visual Studio за да организирате решенията на задачите от упражненията – всяка задача ще бъде в отделен проект и всички проекти ще бъдат в общ solution.

1. Стартирайте Visual Studio.
2. Създайте нов **Blank Solution**: [File]🡪 [New] 🡪 [Project].



1. Изберете от диалоговия прозорец [Templates] 🡪 [Other Project Types] 🡪 [Visual Studio Solutions] 🡪 [**Blank Solution**] и дайте подходящо име на проекта, например “Simple-Conditions”:



Сега имате създаден **празен Visual Studio Solution** (без проекти в него):



Целта на този **blank solution** e 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)* |

1. Създайте **нов проект** в съществуващото Visual Studio решение. В Solution Explorer кликнете с десен бутон на мишката върху **Solution 'Simple-Conditions'**. Изберете [Add] 🡪 [New Project…]:



1. Ще се отвори диалогов прозорец за избор на тип проект за създаване. Изберете C# конзолно приложение и задайте име “Excellent-Result”:



Вече имате solution с едно конзолно приложение в него. Остава да напишете кода за решаване на задачата.

1. Отидете в тялото на метода Main(string[] args) и напишете решението на задачата. Можете да си помогнете с кода от картинката по-долу:



1. **Стартирайте** програмата с [Ctrl+F5] и я **тествайте** с различни входни стойности:

 

## Отлична оценка или не

Следващата 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, или “**Not excellent.**” в противен случай.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 6 | Excellent! | 5 | Not excellent. | 5.50 | Excellent! | 5.49 | Not excellent. |

1. Първо създайте **нов C# конзолен проект** в решението “**Simple-Conditions**”.
   * Кликнете с мишката върху решението в Solution Explorer и изберете [Add] 🡪 [**New Project**…].
   * Изберете [Visual C#] 🡪 [Windows] 🡪 [**Console Application**] и задайте име “Excellent-or-Not”.
2. **Напишете кода** на програмата. Може да си помогнете с примерния код от картинката:



1. Включете режим на **автоматично превключване към текущия проект** като кликнете върху главния solution с десния бутон на мишката и изберете **[Set StartUp Projects…]**:



Ще се появи диалогов прозорец, от който трябва да се избере **[Startup Project]** 🡪 [**Current selection**]:



1. Сега **стартирайте програмата**, както обикновено с [Ctrl+F5] и я тествайте:

 

## Четно или нечетно число

Да се напише програма, която чете **цяло число**, въведено от потребителя, и печата дали е **четно** или **нечетно**. Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 2 | even | 3 | odd | 25 | odd | 1024 | even |

1. Първо добавете **нов C# конзолен проект** в съществуващия solution.
2. **Напишете кода** на програмата. Проверката за четност може да се реализира чрез проверка на **остатъка при деление на 2** по следния начин: var even = (num % 2 == 0).
3. **Стартирайте** програмата с **[Ctrl+F5]** и я тествайте:



## Намиране на по-голямото число

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

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 5  3 | 5 | 3  5 | 5 | 10  10 | 10 | -5  5 | 5 |

1. Първо добавете **нов C# конзолен проект** в съществуващия solution.
2. **Напишете кода** на програмата. Необходима е единична if-else конструкция.
3. **Стартирайте** програмата с **[Ctrl+F5]** и я тествайте:



**Подсказка**: може да си помогнете частично с кода от картинката, който е нарочно замъглен, за да помислите как да си го напишете сами:



## 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!**”. Примери:

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

**Подсказка**: използвайте if-else конструкцията.

## Еднакви думи

Да се напише програма, която **чете две думи**, въведени от потребителя, и проверява дали са еднакви. Да не се прави разлика между главни и малки думи. Да се изведе “yes” или “no”. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| Hello  Hello | yes | SoftUni  softuni | yes | Soft  Uni | no | beer  vodka | no | HeLlO  hELLo | yes |

**Подсказка**: използвайте if-else конструкция. Преди сравняване на думите ги обърнете в долен регистър: word = word.ToLower().

## Време плюс 15 минути

Да се напише програма, която **чете час и минути** от 24-часово денонощие, въведени от потребителя, и изчислява колко ще е **часът след 15 минути**. Резултатът да се отпечата във формат hh:mm. Часовете винаги са между 0 и 23, а минутите винаги са между 0 и 59. Часовете се изписват с една или две цифри. Минутите се изписват винаги с по две цифри, с **водеща нула** когато е необходимо. Примери:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 1  46 | 2:01 | 0  01 | 0:16 | 23  59 | 0:14 | 11  08 | 11:23 | 12  49 | 13:04 |

**Подсказка**: добавете 15 минути и направете няколко проверки. Ако минутите надвишат 59, увеличете часовете с 1 и намалете минутите със 60. По аналогичен начин разгледайте случая, когато часовете надвишат 23. При печатането на минутите проверете за водеща нула.

## \* Три еднакви числа

Да се напише програма, която чете 3 числа и отпечатва дали са еднакви (yes / no).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |
| 5 6 7 | no | 5 5 5 | yes |

## Министерство на образованието и науката (МОН)

* Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист".



* Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под **свободен** **лиценз CC-BY-NC-SA** (Creative Commons Attribution-Non-Commercial-Share-Alike 4.0 International).

