# Упражнения: Четене и печатане на конзолата

## Празно Visual Studio решение (Blank Solution)

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.

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-Calculations”:



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



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

## Пресмятане на лице на квадрат

Първата задача от тази тема е следната: да се напише **конзолна програма**, която **прочита цяло число** „a“,въведено от потребителя, и **пресмята лицето на квадрат** **със страна** „a“. Задачата е тривиално лесна: въвеждате число от конзолата, умножавате го само по себе си и печатате получения резултат на конзолата.

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



1. Ще се отвори диалогов прозорец за избор на тип проект за създаване. Изберете C# конзолно приложение с име “Square-Area”:



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 кода от картинката по-долу:



Кодът прочита цяло число с a = int.Parse(Console.ReadLine()), след това изчислява area = a \* a и накрая печата стойността на променливата area.

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



## От инчове към сантиметри

Да се напише програма, която **чете от конзолата число** (не непременно цяло), въведено от потребителя, и преобразува числото **от инчове в сантиметри**. За целта **умножава инчовете по 2.54** (защото 1 инч = 2.54 сантиметра).

1. Първо създайте **нов C# конзолен проект** в решението “**Simple-Calculations**”. Кликнете с мишката върху решението в Solution Explorer и изберете [Add] 🡪 [New Project…]:



Изберете [Visual C#] 🡪 [Windows] 🡪 [Console Application] и задайте име “Inches-to-Centimeters”:



1. **Напишете кода** на програмата. Може да си помогнете с примерния код от картинката:



1. **Стартирайте програмата**, както обикновено с [Ctrl+F5]:



Изненада! Како става? Програмата не работи правилно… Всъщност това не е ли предходната програма?

Във Visual Studio **текущият активен проект** в един solution е маркиран в получерно и може да се сменя:



1. За да включите режим на **автоматично преминаване към текущия проект**, 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 отново **стартирайте програмата**, както обикновено с [Ctrl+F5]. Този път ще се стартира текущата отворена програма, която преобразува инчове в сантиметри. Изглежда работи коректно:



1. Сега **превключете към преходната програма** (лице на квадрат). Това става с двоен клик на мишката върху файла Program.cs от предходния проект “Square-Area” в панела [Solution Explorer] на Visual Studio:



1. Натиснете пак **[Ctrl+F5]**. Този път трябва да се стартира другият проект:



1. Превключете обратно към проекта “Inches-to-Centimeters” и го стартирайте с **[Ctrl+F5]**:



**Превключването между проектите** е много лесно, нали? Просто избираме файла със сорс кода на програмата, кликваме го два пъти с мишката и при стартиране тръгва програмата от този файл.

1. Тествайте с **дробни числа**, например с 2.5:



**Внимание:** в зависимост от регионалните настройки на операционната система, е възможно вместо **десетична точка** (US настройки) да се използва **десетична запетая** (BG настройки). Ако програмата очаква десетична точка и бъде въведено число с десетична запетая или на обратно (бъде въведена десетична точка когато се очаква десетична запетая), ще се получи следната грешка:



Препоръчително е **да промените настройките на компютъра си**, така че да се използва **десетична точка**:





## Поздрав по име

Да се напише програма, която **чете от конзолата име на човек**, въведено от потребителя, и отпечатва “Hello, <name>!”, където <name> е въведеното преди това име.

1. Първо създайте **нов C# конзолен проект** с име “Greeting” в решението “Simple-Calculations”:



1. **Напишете кода** на програмата. Ако се затруднявате, може да ползвате примерния код по-долу:



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



## Съединяване на текст и числа

Напишете C# програма, която прочита от конзолата име, фамилия, възраст и град, въведени от потребителя, и печата съобщение от следния вид: “You are <firstName> <lastName>, a <age>-years old person from <town>.”.

1. Добавете към текущото Visual Studio решение още един **конзолен C# проект** с име “Concatenate-Data”.
2. **Напишете кода**, който чете входните данни от конзолата:



1. **Допишете код**, който отпечатва описаното в условието на задачата съобщение.



На горната картинка кодът е нарочно даден размазан, за да помислите как да си го напишете сами.

1. **Тествайте** решението с [Ctrl+F5] и въвеждане на примерни данни.

## \* Конвертор от °C към °F

Напишете програма, която чете **градуси по скалата на Целзий** (°C), въведени от потребителя, и ги преобразува до **градуси по скалата на Фаренхайт** (°F). Потърсете в Интернет подходяща [формула](http://bfy.tw/3rGh), с която да извършите изчисленията. Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 25 | 77 | 0 | 32 | -5.5 | 22.1 | 32.3 | 90.14 |

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

 double farenhait = gradus \* 1.8 + 32;

 Console.WriteLine(farenhait);

## Конвертор от радиани в градуси

Напишете програма, която чете **ъгъл в** [**радиани**](https://bg.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B4%D0%B8%D0%B0%D0%BD) (rad), въведен от потребителя, и го преобразува в [**градуси**](https://bg.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D0%B4%D1%83%D1%81_%28%D1%8A%D0%B3%D1%8A%D0%BB%29) (deg). Потърсете в Интернет подходяща формула. Числото **π** в C# програми е достъпно чрез Math.PI. Закръглете резултата до най-близкото цяло число използвайки Math.Round(). Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 3.1416 | 180 | 6.2832 | 360 | 0.7854 | 45 | 0.5236 | 30 |

## \* Конвертор от USD към BGN

Напишете програма за **конвертиране на щатски долари** (USD) **в български лева** (BGN). **Закръглете** резултата до **2 цифри** след десетичната точка. Използвайте фиксиран **курс** между долар и лев: **1 USD** = **1.79549 BGN**.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 20 | 35.91 BGN | 100 | 179.55 BGN | 12.5 | 22.44 BGN |