# Упражнения: Основни команди

1. **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.

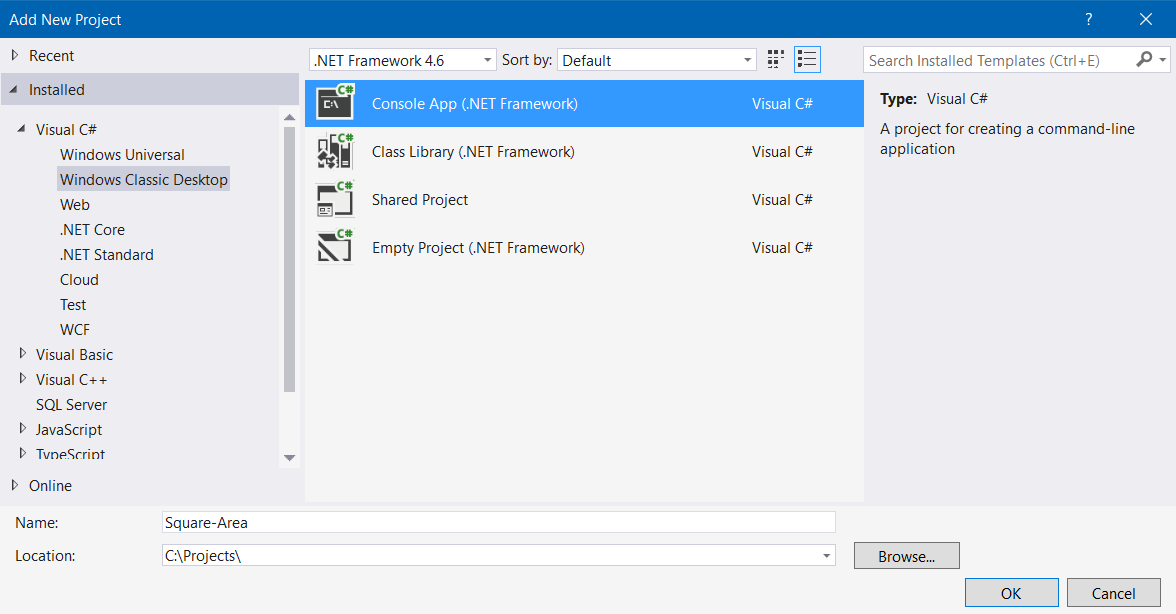
1. **Пресмятане на лице на квадрат**

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:



1. **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**:





1. **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:



1. **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.
2. **Лице на трапец**

Напишете програма, която чете от конзолата три числа **b1**, **b2** и **h**, въведени от потребителя, и **пресмята лицето на трапец** с основи **b1** и **b2** и височина **h**. **Формулата за лице на трапец** е  
**(b1** **+** **b2)** **\*** **h** **/** **2**.

На фигурата е показан трапец със страни 8 и 13 и височина 7. Той има лице (8 + 13) \* 7 / 2 = 73.5.

1. Добавете към текущото Visual Studio решение още един **конзолен C# проект** с име “**Trapezoid-Area**”.
2. **Напишете кода**, който чете входните данни от конзолата, пресмята лицето на трапеца и го отпечатва:



Кодът на картинката е нарочно размазан, за да си го доизмислите и допишете сами.

1. **Тествайте** решението локално с [Ctrl+F5] и въвеждане на примерни данни.
2. **Периметър и лице на кръг**

Напишете програма, която чете от конзолата **число** **r**,въведено от потребителя, и пресмята и отпечатва **лицето** и **периметъра на кръг** / окръжност с радиус **r**. Закръглете резултата до **2 знака след десетичната точка,** използвайки **Math.Round()**.

|  |  |
| --- | --- |
| **вход** | **изход** |
| 3 | Area = 28.27  Perimeter = 18.85 |
| 4.5 | Area = 63.62  Perimeter = 28.27 |

За изчисленията можете да използвате следните формули:

* **area = Math.PI \* r \* r**
* **perimeter = 2 \* Math.PI \* r**

1. **Лице на правоъгълник в равнината**

**Правоъгълник** е зададен с **координатите** на два от своите срещуположни ъгъла (**x1**, **y1**) – (**x2**, **y2**). Да се пресметнат **площта** и **периметъра** му. **Входът** се въвежда от потребителя. Числата **x1**, **y1**, **x2** и **y2** са дадени по едно наред. **Изходът** се извежда на конзолата и трябва да съдържа два реда с по една число на всеки от тях – лицето и периметъра.

|  |  |
| --- | --- |
| **вход** | **изход** |
| 60  20  10  50 | 1500  160 |
| 30  40  70  -10 | 2000  180 |
| 600.25  500.75  100.50  -200.5 | 350449.6875  2402 |

1. **Лице на триъгълник**

Напишете програма, която чете от конзолата **страна** и **височина** на **триъгълник,** въведени от потребителя,и пресмята неговото лице. Използвайте **формулата** за лице на триъгълник: **area** **=** **a** **\*** **h** **/** **2**. Закръглете резултата до **2 знака след десетичната точка**.

|  |  |
| --- | --- |
| **вход** | **изход** |
| 20  30 | Triangle area = 300 |
| 15  35 | Triangle area = 262.5 |
| 7.75  8.45 | Triangle area = 32.74 |
| 1.23456  4.56789 | Triangle area = 2.82 |

1. **\* 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 |

1. **\* 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_(%D1%8A%D0%B3%D1%8A%D0%BB)) (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 |

1. **\* 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 |

1. **\* Междувалутен конвертор**

Напишете програма за **конвертиране на парична сума от една валута в друга**. Трябва да се поддържат следните валути: **BGN**, **USD**, **EUR**, **GBP**. Използвайте следните фиксирани валутни курсове:

|  |  |  |  |
| --- | --- | --- | --- |
| **Курс** | **USD** | **EUR** | **GBP** |
| **1 BGN** | 1.79549 | 1.95583 | 2.53405 |

**Входът** e **сума за конвертиране**, **входна валута**, **изходна валута**, въведени от потребителя. **Изходът** е едно число – преобразуваната сума по посочените по-горе курсове, закръглен до **2 цифри** след десетичната точка. Примери:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |  | **вход** | **изход** |
| 20  USD  BGN | 35.91 BGN | 100  BGN  EUR | 51.13 EUR | 12.35  EUR  GBP | 9.53 GBP | 150.35  USD  EUR | 138.02 EUR |

1. **\*\* 1000 дни на Земята**

Напишете програма, която чете **рождена дата** във формат “**dd-MM-yyyy**”, въведена от потребителя, и пресмята датата, на която се навършват **1000 дни** от тази рождена дата и я отпечатва в същия формат.

|  |  |
| --- | --- |
| **вход** | **изход** |
| 25-02-1995 | 20-11-1997 |
| 07-11-2003 | 02-08-2006 |
| 30-12-2002 | 24-09-2005 |
| 01-01-2012 | 26-09-2014 |
| 14-06-1980 | 10-03-1983 |

**\* Подсказки**: потърсете информация за типа **DateTime** в C# и по-конкретно разгледайте методите **ParseExact(str,** **format)**, **AddDays(count)** и **ToString(format)**. С тяхна помощ може да решите задачата, без да е необходимо да изчислявате дни, месеци и високосни години.

**Изпитни задачи от "ИТ Кариера"**

1. **\* Учебна зала**

*Първа задача от изпита на 6 март 2016.*

**Учебна зала** има правоъгълен размер **w** на **h** метра, без колони във вътрешността си. Залата е разделена на две части – лява и дясна, с коридор приблизително по средата. В лявата и в дясната част има **редици с бюра**. В задната част на залата има голяма **входна врата**. В предната част на залата има **катедра** с подиум за преподавателя. Едно **работно място** заема **70 на 120 cm** (маса с размер 70 на 40 cm + място за стол и преминаване с размер 70 на 80 cm). **Коридорът** е широк поне **100 cm**. Изчислено е, че заради **входната врата** (която е с отвор 160 cm) се губи точно **1 работно място**, а заради **катедрата** (която е с размер 160 на 120 cm) се губят точно **2 работни места**. Напишете програма, която прочита размерите на учебната зала и изчислява **броя работни места в нея** при описаното разположение (вж. фигурата).

**Вход**

От конзолата се четат 2 **числа**, по едно на ред: **h** (дължина в метри) и **w** (широчина в метри) , въведени от потребителя.

Ограничения: **3** ≤ **h** ≤ **w** ≤ **100**.

**Изход**

Да се отпечата на конзолата едно цяло число: **броят места** в учебната зала.

**Примерен вход и изход**

|  |  |  |  |
| --- | --- | --- | --- |
| **Вход** | **Изход** | **Чертеж** | **Обяснения** |
| 15  8.9 | 129 |  | Залата е дълга 1500 cm. В тях могат да бъдат разположени **12 реда** (12 \* 120 cm = 1440 + 60 cm остатък).  Залата е широка 890 cm. От тях 100 cm отиват за коридора в средата. В останалите 790 cm могат да се разположат по **11 бюра** **на ред** (11 \* 70 cm = 770 cm + 20 cm остатък).  **Брой места** = **12 \* 11 - 3** = 132 - 3 = **129** (имаме 12 реда по 11 места = 132 минус 3 места за катедра и входна врата). |
| 8.4  5.2 | 39 |  | Залата е дълга 840 cm. В тях могат да бъдат разположени **7 реда** (7 \* 120 cm = 840, без остатък).  Залата е широка 520 cm. От тях 100 cm отиват за коридора в средата. В останалите 420 cm могат да се разположат по **6 бюра** **на ред** (6 \* 70 cm = 420 cm, без остатък).  **Брой места** = **7 \* 6 - 3** = 42 - 3 = **39** (имаме 7 реда по 6 места = 42 минус 3 места за катедра и входна врата). |

1. **\* Зеленчукова борса**

*Първа задача от изпита на 26 март 2016.*

Градинар продавал реколтата от градината си на зеленчуковата борса. Продава **зеленчуци за** **N лева на килограм** и **плодове за M лева за килограм**. Напишете програма, която да **пресмята приходите от реколтата в евро** ( ако приемем, че **едно евро** е равно на **1.94лв**).

**Вход**

От конзолата се четат **4 числа**, по едно на ред, въведени от потребителя:

* Първи ред – **Цена за килограм зеленчуци** – число с плаваща запетая
* Втори ред – **Цена за килограм плодове** – число с плаваща запетая
* **Трети ред –** Общо килограми на зеленчуците **– цяло число**
* **Четвърти ред –** Общо килограми на плодовете **– цяло число**

**Ограничения**: **Всички числа ще са в интервала от 0.00 до 1000.00**

**Изход**

Да се отпечата на конзолата **едно число с плаваща запетая**: **приходите от всички плодове и зеленчуци в** **евро**. Резултатът **да се форматира до втория знак след запетаята.**

**Примерен вход и изход**

|  |  |  |
| --- | --- | --- |
| **Вход** | **Изход** | **Обяснения** |
| 0.194  19.4  10  10 | 101.00 | Зеленчуците струват – 0.194лв. \* 10кг. = 1.94лв.  Плодовете струват – 19.4лв. \* 10кг. = 194лв.  Общо – 195.94лв. = 101евро |
| 1.5  2.5  10  10 | 20.62 |  |