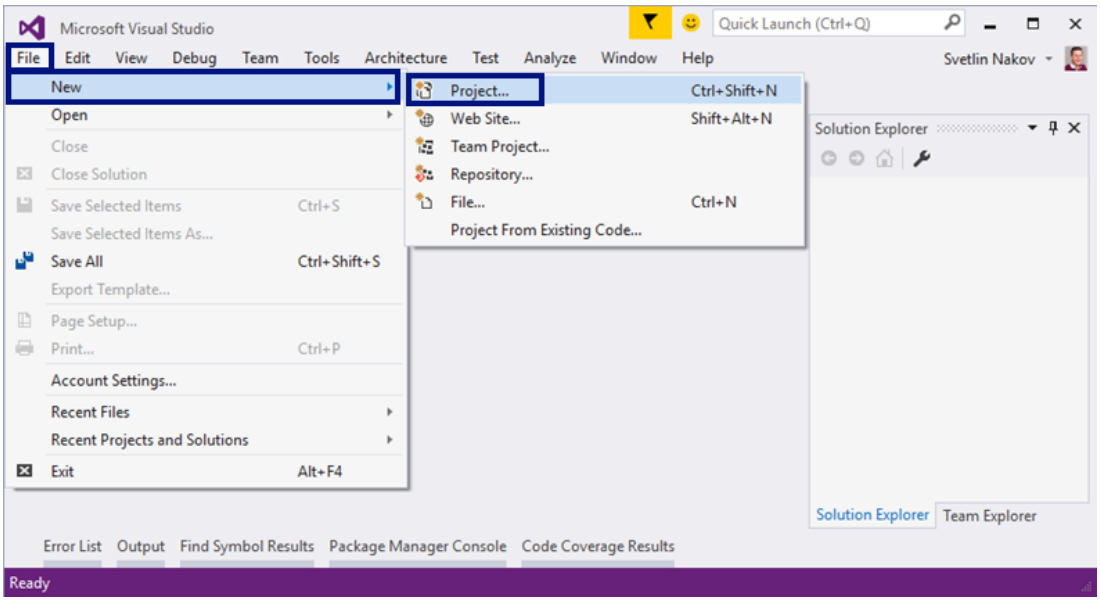
Упражнения: По-сложни проверки

Задачи за упражнение в клас и за домашно към курса „Основи на програмирането“ @ СофтУни.

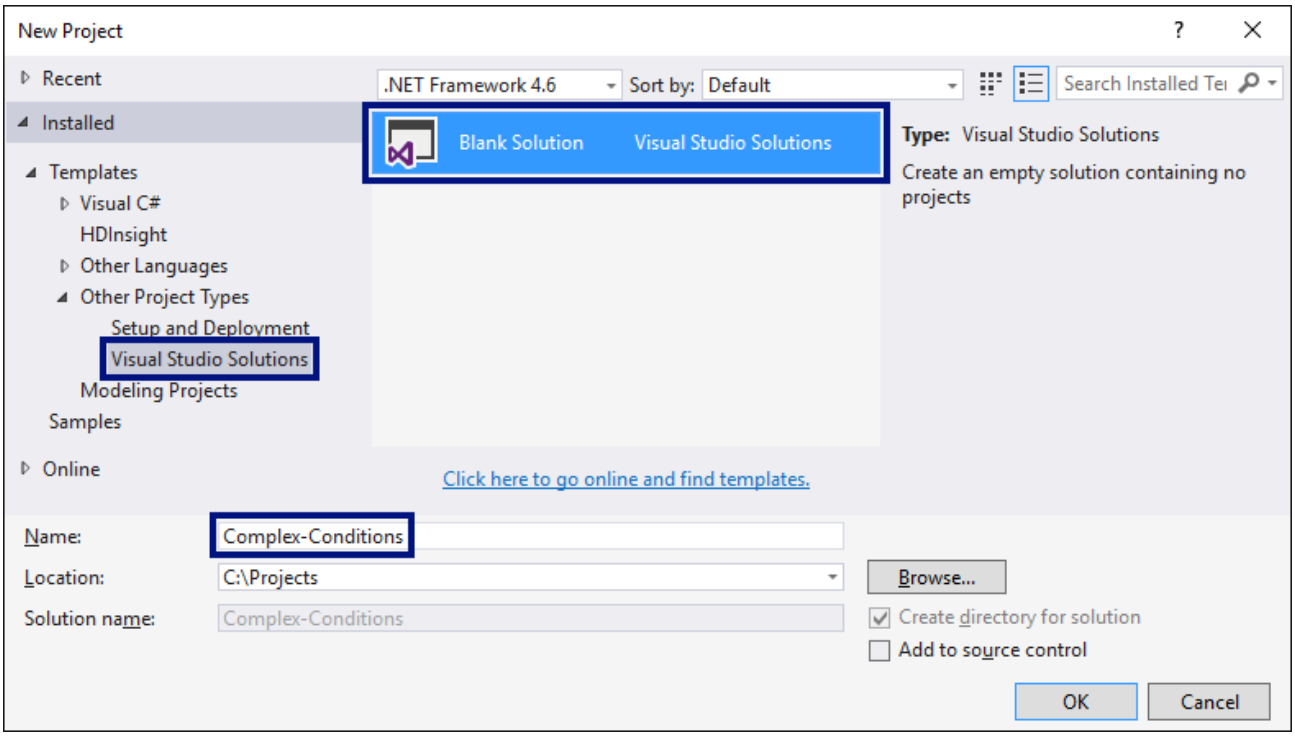
0. Празно Visual Studio решение (Blank Solution)

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.

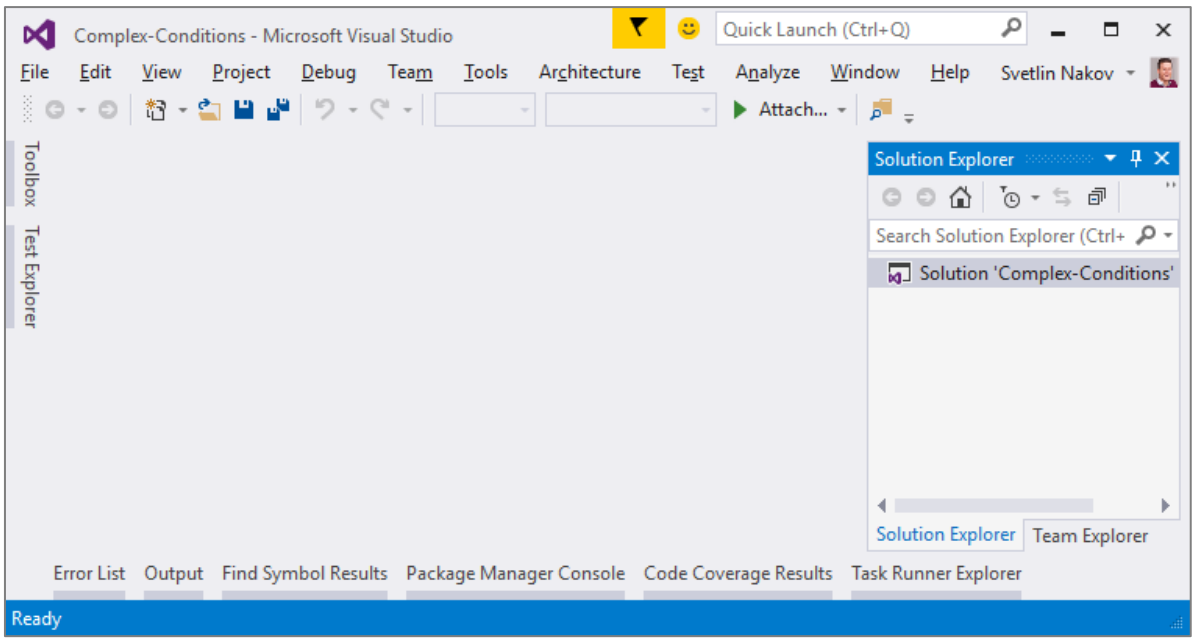
1. Стартирайте **Visual Studio**.
2. Създайте нов **Blank Solution**: [File]→ [New] → [Project].



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



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



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

1. Обръщение според възраст и пол

Първата задача от тази тема е да се напише **конзолна програма**, която **въвежда възраст** (десетично число) и **пол** (“`m`” или “`f`”) и отпечатва **обръщение** измежду следните:

- “**`Mr.`**” – мъж (пол “`m`”) на 16 или повече години
- “**`Master`**” – момче (пол “`m`”) под 16 години
- “**`Ms.`**” – жена (пол “`f`”) на 16 или повече години
- “**`Miss`**” – момиче (пол “`f`”) под 16 години

Примери:

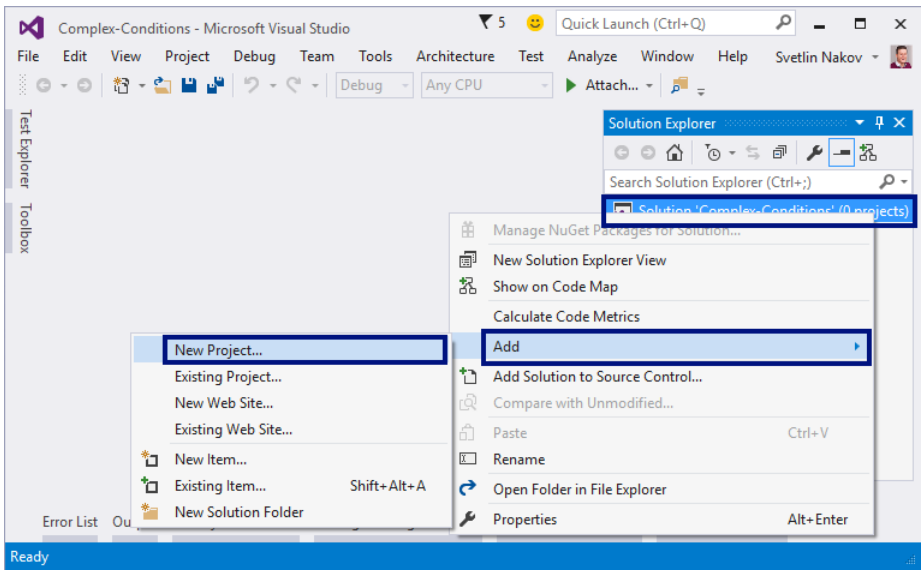
вход	изход
12 f	Miss

вход	изход
17 m	Mr.

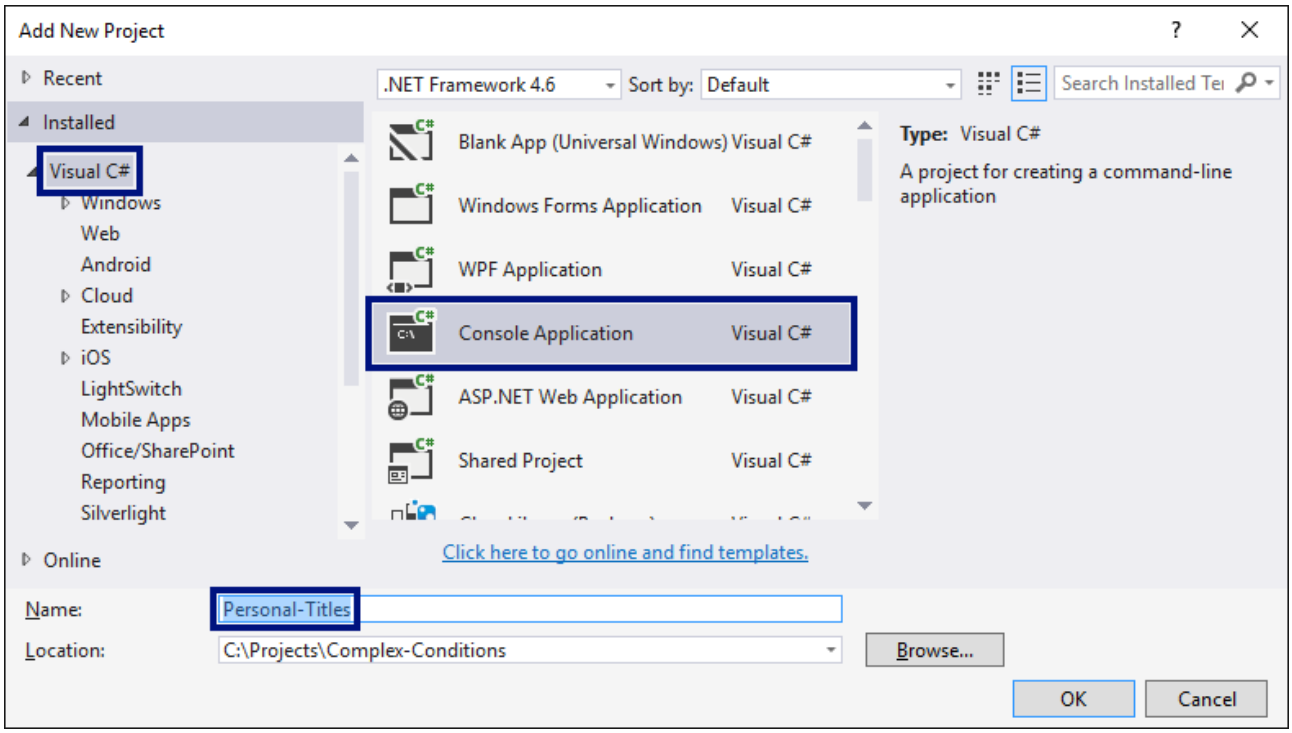
вход	изход
25 f	Ms.

вход	изход
13.5 m	Master

1. Създайте **нов проект** в съществуващото Visual Studio решение. В Solution Explorer кликнете с десен бутон на мишката върху **Solution** реда и изберете [Add] → [New Project…]:



2. Ще се отвори диалогов прозорец за избор на тип проект за създаване. Изберете **C# конзолно приложение** и задайте подходящо име, например "**`Personal-Titles`**":

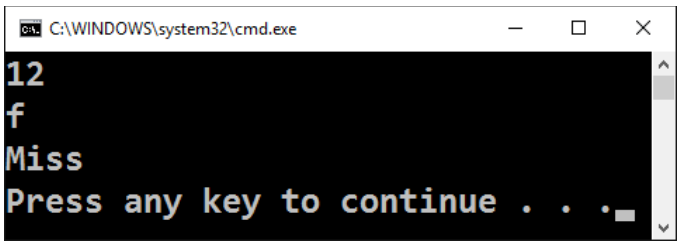


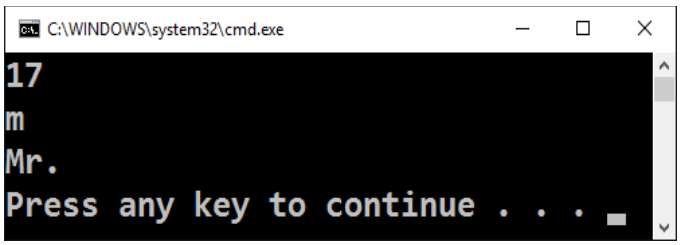
Вече имате solution с едно конзолно приложение в него. Остава да напишете кода за решаване на задачата.

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

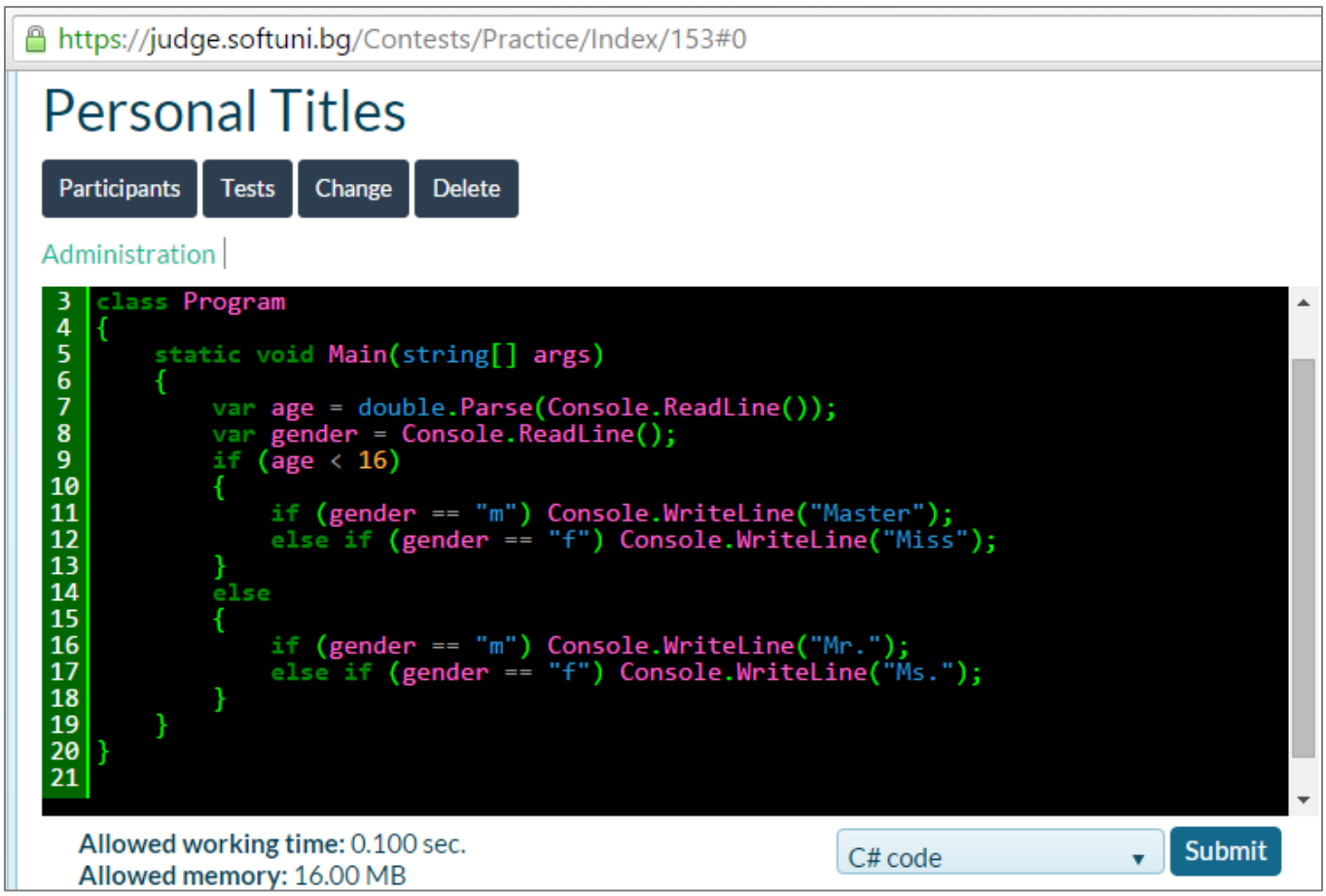
```
var age = double.Parse(Console.ReadLine());
var gender = Console.ReadLine();
if (age < 16)
{
    if (gender == "m") Console.WriteLine("Master");
    else if (gender == "f") Console.WriteLine("Miss");
}
else
{
    if (gender == "m") Console.WriteLine("Mr.");
    else if (gender == "f") Console.WriteLine("Ms.");
}
```

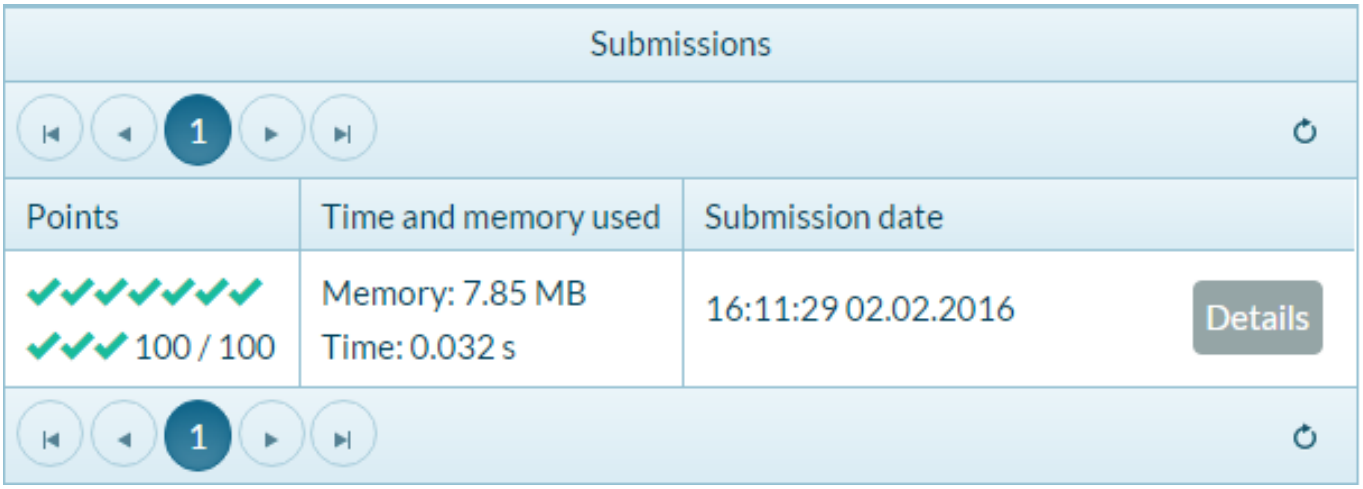
4. **Стартирайте** програмата с [Ctrl+F5] и я **тествайте** с различни входни стойности:





5. **Тествайте** решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#0. Трябва да получите **100 точки** (напълно коректно решение):





2. Квартално магазинче

Следващата задача има за цел да тренира работата с **вложени проверки** (nested `if`). Ето го и условието: предприемчив българин отваря **квартални магазинчета** в **няколко града** и продава на **различни цени**:

град / продукт	coffee	water	beer	sweets	peanuts
Sofia	0.50	0.80	1.20	1.45	1.60
Plovdiv	0.40	0.70	1.15	1.30	1.50
Varna	0.45	0.70	1.10	1.35	1.55

Напишете програма, която чете от конзолата **град** (стринг), **продукт** (стринг) и **количество** (десетично число) и пресмята и отпечатва **колко струва** съответното количество от избрания продукт в посочения град. Примери:

вход	изход
coffee Varna 2	0.9

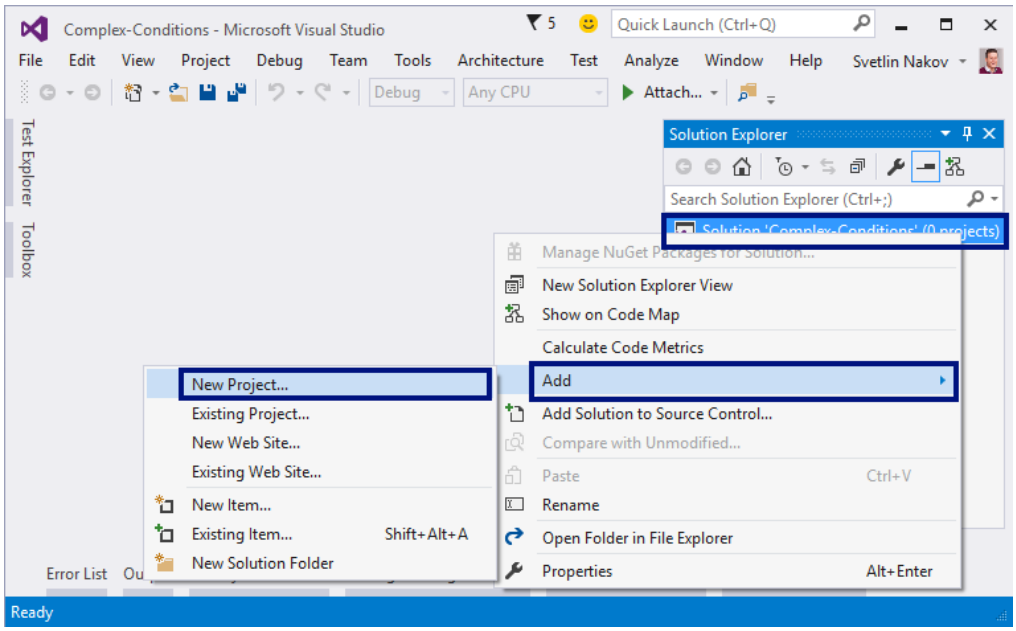
вход	изход
peanuts Plovdiv 1	1.5

вход	изход
beer Sofia 6	7.2

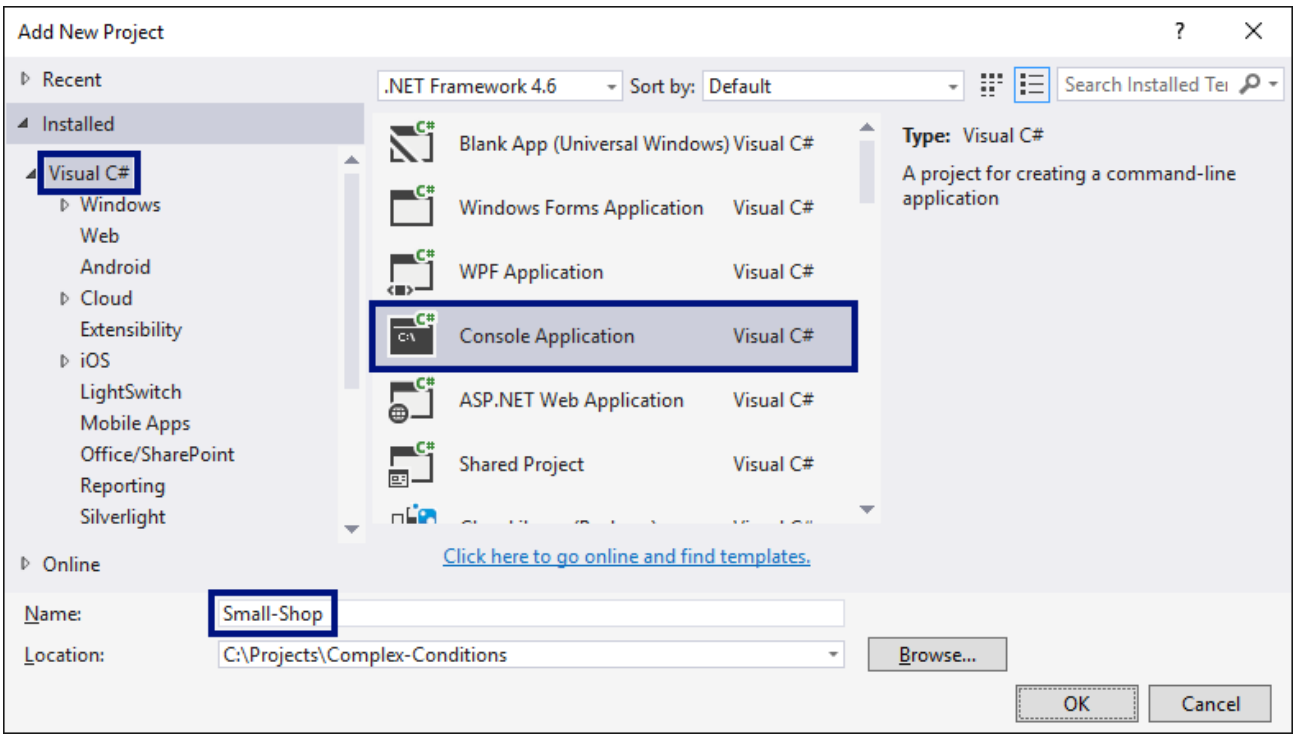
вход	изход
water Plovdiv 3	2.1

вход	изход
sweets Sofia 2.23	3.2335

1. Създайте **нов проект** в съществуващото Visual Studio решение. В Solution Explorer кликнете с десен бутон на мишката върху **Solution** реда и изберете [Add] → [New Project…]:



2. Ще се отвори диалогов прозорец за избор на тип проект за създаване. Изберете **C# конзолно приложение** и задайте подходящо име, например "**`Small-Shop`**":

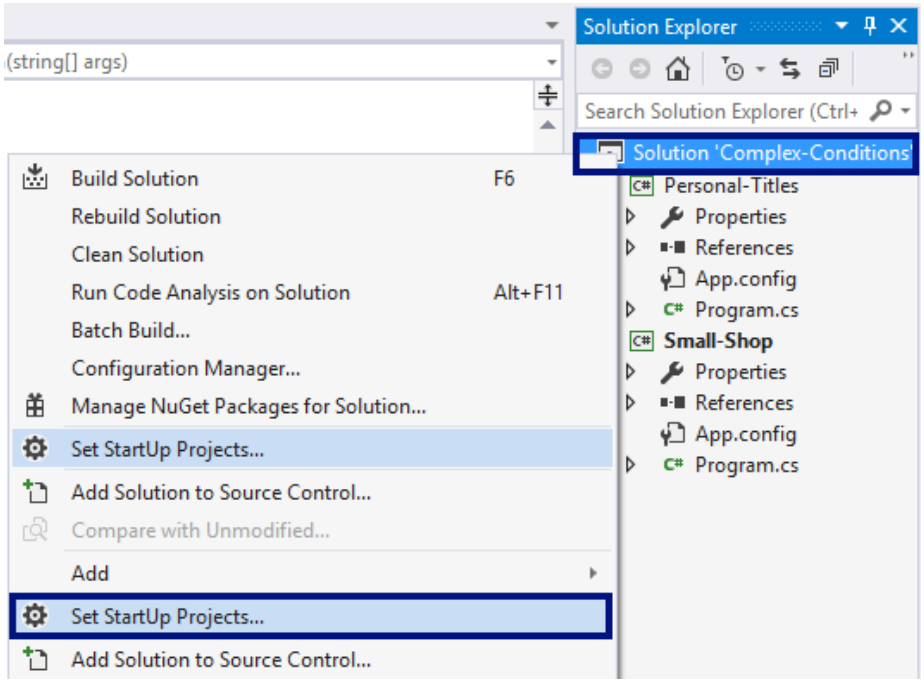


Вече имате ново конзолно приложение и остава да напишете кода за решаване на задачата.

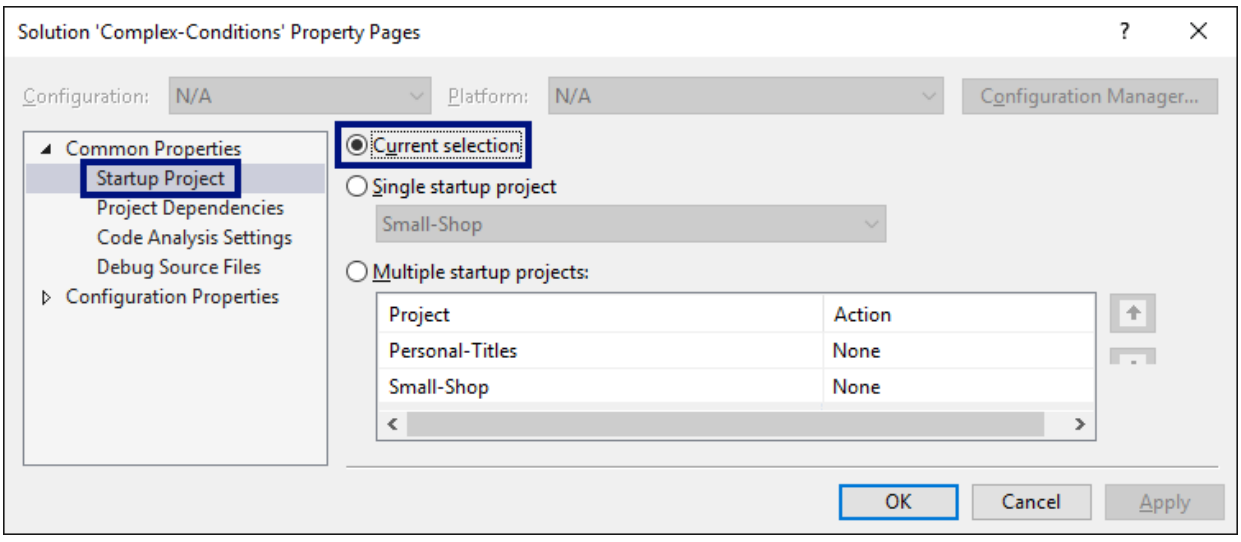
3. Отидете в тялото на метода **`Main(string[] args)`** и напишете решението на задачата. Можете да си помогнете с кода от картинката по-долу. Можете да прехвърлите всички букви в долен регистър с **`.ToLower()`** за да сравнявате продукти и градове без значение на малки / главни букви:

```
var product = Console.ReadLine().ToLower();
var town = Console.ReadLine().ToLower();
var quantity = double.Parse(Console.ReadLine());
if (town == "sofia")
{
    if (product == "coffee") Console.WriteLine(0.50 * quantity);
    // TODO: check the other products ...
}
if (town == "plovdiv")
{
    // TODO: check for each product here ...
}
if (town == "varna")
{
    // TODO: check for each product here ...
}
```

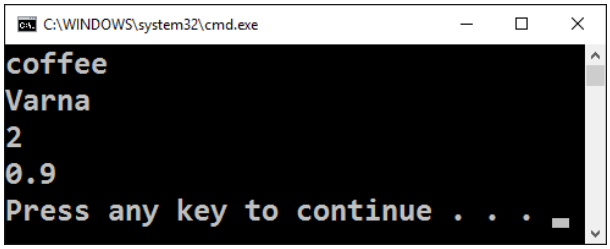
4. За **да активирате текущия проект** да стартира при [Ctrl+F5], изберете "**Set StartUp Projects…**":

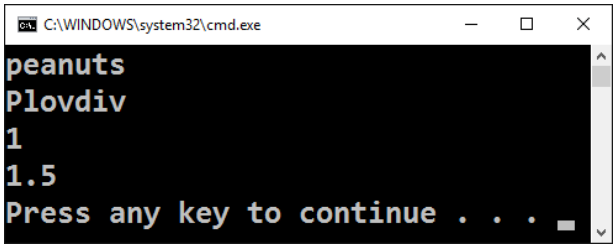


Изберете първата опция:



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

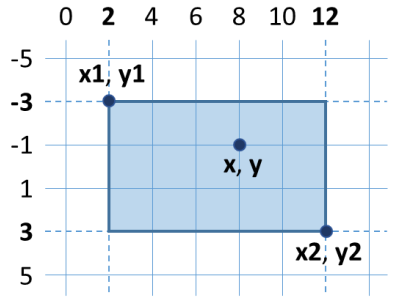


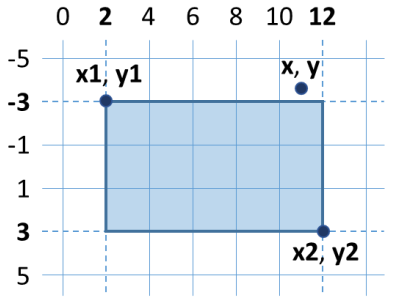


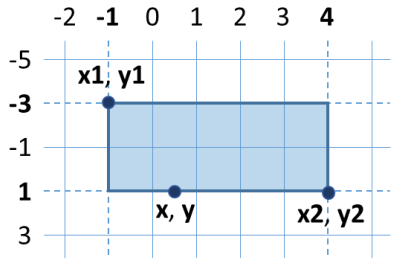
6. **Тествайте** решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#1.

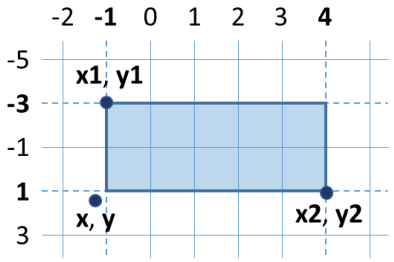
3. Точка в правоъгълник

Напишете програма, която проверява дали **точка {x, y}** се намира **вътре в правоъгълник {x1, y1} – {x2, y2}**. Входните данни се четат от конзолата и се състоят от 6 реда: десетичните числа **x1**, **y1**, **x2**, **y2**, **x** и **y** (като се гарантира, че **x1 < x2** и **y1 < y2**). Една точка е вътрешна за даден правоъгълник, ако се намира някъде във вътрешността му или върху някоя от страните му. Отпечатайте "**Inside**" или "**Outside**". Примери:

вход	изход	визуализация
2 -3 12 3 8 -1	Inside	

вход	изход	визуализация
2 -3 12 3 11 -3.5	Outside	

вход	изход	визуализация
-1 -3 4 1 0.5 1	Inside	

вход	изход	визуализация
-1 -3 4 1 -1.2 1.4	Outside	

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#2.

* **Подсказка**: една точка е вътрешна за даден многоъгълник, ако едновременно са изпълнени следните четири условия (можете да ги проверите с **`if`** проверка с логическо „**и**" – оператор **&&**):

- Точката е надясно от лявата стена на правоъгълника (**x >= x1**)
- Точката е наляво от дясната стена на правоъгълника (**x <= x2**)
- Точката е надолу от горната стена на правоъгълника (**y >= y1**)
- Точката е нагоре от долната стена на правоъгълника (**y <= y2**)

4. Плод или зеленчук?

Да се напише програма, която **въвежда име на продукт** и проверява дали е **плод** или **зеленчук**.

- Плодовете "**fruit**" са **banana**, **apple**, **kiwi**, **cherry**, **lemon** и **grapes**

- Зеленчуците "**vegetable**" са **tomato**, **cucumber**, **pepper** и **carrot**
- Всички останали са "**unknown**"

Да се изведе "**fruit**", "**vegetable**" или "**unknown**" според въведения продукт. Примери:

вход	изход
banana	fruit

вход	изход
apple	fruit

вход	изход
tomato	vegetable

вход	изход
water	unknown

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#3.

* **Подсказка**: използвайте условна **`if`** проверка с логическо „**или**" – operator **`||`**.

5. Невалидно число

Дадено **число е валидно**, ако е в диапазона [**100**…**200**] или е **0**. Да се напише програма, която **въвежда цяло число** и печата "**invalid**" ако въведеното число **не е валидно**. Примери:

вход	изход
75	invalid

вход	изход
150	*(няма изход)*

вход	изход
220	invalid

вход	изход
199	*(няма изход)*

вход	изход
-1	invalid

вход	изход
100	*(няма изход)*

вход	изход
200	*(няма изход)*

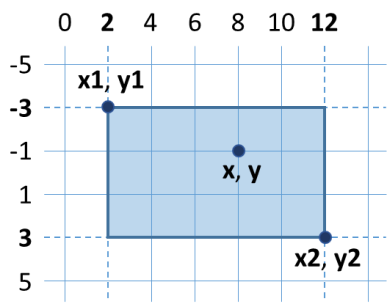
вход	изход
0	*(няма изход)*

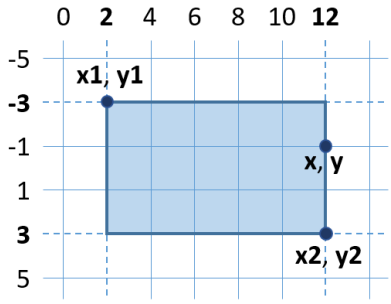
Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#4.

* **Подсказка**: използвайте условна **`if`** проверка с **отрицание** и логически операции.

6. Точка върху страната на правоъгълник

Напишете програма, която проверява дали **точка {x, y}** се намира **върху някоя от страните на правоъгълник {x1, y1} – {x2, y2}**. Входните данни се четат от конзолата и се състоят от 6 реда: десетичните числа **x1**, **y1**, **x2**, **y2**, **x** и **y** (като се гарантира, че **x1 < x2** и **y1 < y2**). Да се отпечата "**Border**" (точката лежи на някоя от страните) или "**Inside / Outside**" (в противен случай). Примери:

вход	изход	визуализация
2 -3 12 3 8 -1	Inside / Outside	

вход	изход	визуализация
2 -3 12 3 12 -1	Border	

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#5.

* **Подсказка**: използвайте една или няколко условни **`if`** проверки с логически операции. Точка **{x, y}** лежи върху някоя от страните на правоъгълник **{x1, y1} – {x2, y2}**, ако е изпълнено едно от следните условия:

- **x** съвпада с **x1** или **x2** и същевременно **y** е между **y1** и **y2**
- **y** съвпада с **y1** или **y2** и същевременно **x** е между **x1** и **x2**

Можете да проверите горните условия с една по-сложна **`if-else`** конструкция или с няколко по-прости проверки или с **вложени `if-else`** проверки.

7. Магазин за плодове

Магазин за плодове през **работните дни** работи на следните **цени**:

плод	banana	apple	orange	grapefruit	kiwi	pineapple	grapes
цена	2.50	1.20	0.85	1.45	2.70	5.50	3.85

Събота и **неделя** магазинът работи на **по-високи цени**:

плод	banana	apple	orange	grapefruit	kiwi	pineapple	grapes
цена	2.70	1.25	0.90	1.60	3.00	5.60	4.20

Напишете програма, която чете от конзолата **плод** (banana / apple / orange / grapefruit / kiwi / pineapple / grapes), **количество** (десетично число) и **ден от седмицата** (Monday / Tuesday / Wednesday / Thursday / Friday / Saturday / Sunday) и пресмята **цената** според цените от таблиците по-горе. Резултатът да се отпечата **закръглен с 2 цифри** след десетичната точка. При невалиден ден от седмицата или невалидно име на плод да се отпечата "**error**". Примери:

вход	**изход**
apple Tuesday 2	2.40

вход	**изход**
orange Sunday 3	2.70

вход	**изход**
kiwi Monday 2.5	6.75

вход	**изход**
grapes Saturday 0.5	2.10

вход	**изход**
tomato Monday 0.5	error

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#6.

* **Подсказки**:

- Прочетете входа и обърнете името на плода и деня от седмицата в **малки букви**:

```
var fruit = Console.ReadLine().ToLower();
var day = Console.ReadLine().ToLower();
var quantity = double.Parse(Console.ReadLine());
```

- Първоначално задайте цена **-1**:

```
var price = -1.0;
```

- Използвайте вложени **if** проверки, за да изчислите цената за дадения плод и ден от седмицата:

```
if (day == "monday" || day == "tuesday" || day == "wednesday" ||
    day == "thursday" || day == "friday")
{
    if (fruit == "banana") price = 2.50;
    else if (fruit == "apple") price = 1.20;
    // TODO: more fruits come here …
}
else if (day == "saturday" || day == "sunday")
{
    if (fruit == "banana") price = 2.70;
    // TODO: more fruits come here …
}
```

- Накрая проверете цената. Ако все още е **-1**, значи даденият плод или денят от седмицата е **невалиден**. За да отпечатате точно **2 цифри след десетичната точка** (със закръгляне), използвайте форматиращ низ "**`{0:f2}`**". Кодът може да е подобен на следния:

```
if (price >= 0)
    Console.WriteLine("{0:f2}", price * quantity);
else
    Console.WriteLine("error");
```

8. Търговски комисионни

Фирма дава следните **комисионни** на търговците си според **града**, в който работят и обема на **продажбите s**:

Град	0 ≤ s ≤ 500	500 < s ≤ 1 000	1 000 < s ≤ 10 000	s > 10 000
Sofia	5%	7%	8%	12%
Varna	4.5%	7.5%	10%	13%
Plovdiv	5.5%	8%	12%	14.5%

Напишете **конзолна програма**, която чете име на **град** (стринг) и обем на **продажби** (десетично число) и изчислява и извежда размера на търговската **комисионна** според горната таблица. Резултатът да се изведе закръглен с **2 цифри след десетичната точка**. При **невалиден** град или обем на продажбите (отрицателно число) да се отпечата "**error**". Примери:

вход	изход
Sofia 1500	120.00

вход	изход
Plovdiv 499.99	27.50

вход	изход
Varna 3874.50	387.45

вход	изход
Kaspichan -50	error

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#7.

* **Подсказки**:

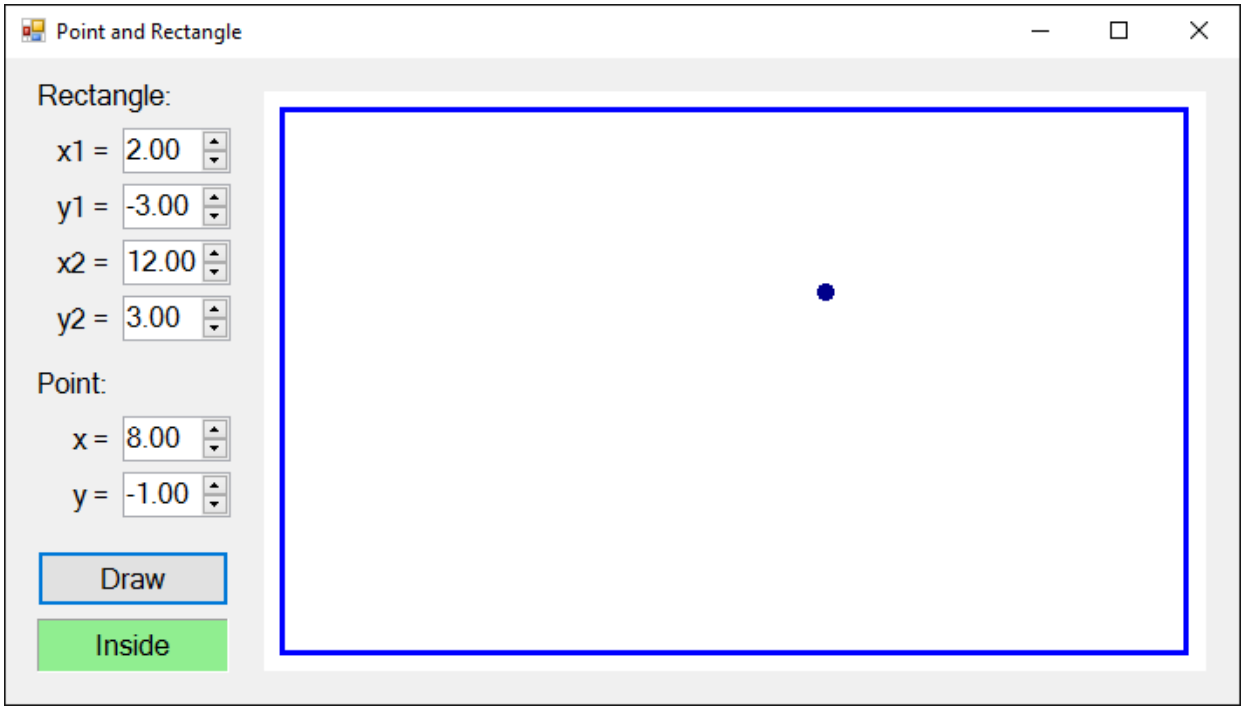
- Прочетете входа и **обърнете града в малки букви** (като в предходната задача).
- Първоначално задайте **комисионна -1**. Тя ще бъде променена, ако градът и ценовият диапазон бъдат намерени в таблицата с комисионните.
- Използвайте вложени **`if`** проверки, за **да изчислите комисионната** според града и според обема на продажбите. Може да си помогнете с кода по-долу:

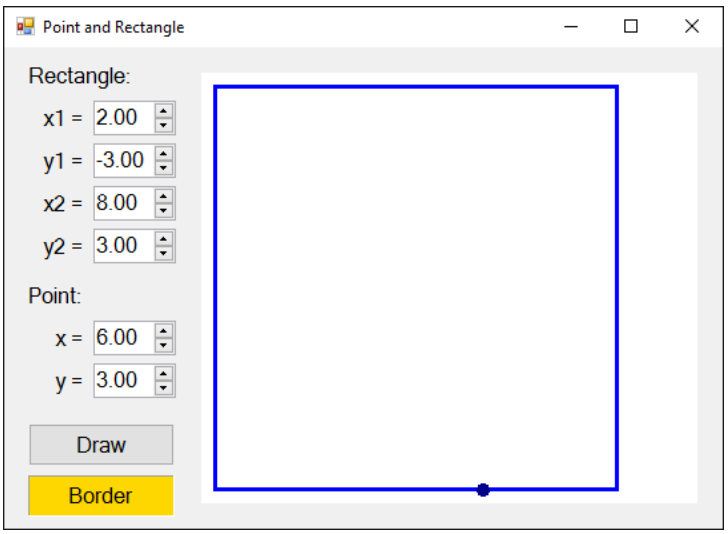
```
if (town == "sofia")
{
    if (0 <= sales && sales <= 500) comission = 0.05;
    else if (500 < sales && sales <= 1000) comission = 0.07;
    // TODO: add more price ranges here ...
}
else if (town == "varna")
{
    // TODO: check the price ranges here ...
}
else if (town == "plovdiv")
{
    // TODO: check the price ranges here ...
}
```

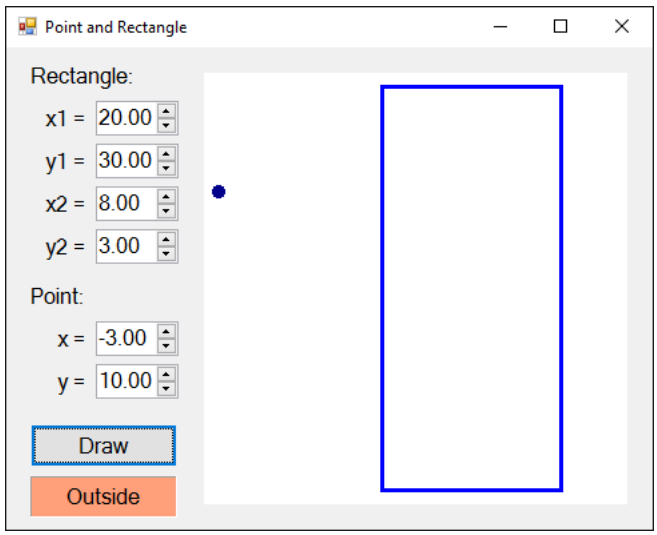
- Накрая проверете комисионната. Ако все още е **-1**, значи въведеният град или обем продажби не се срещат в таблицата с комисионните и трябва да се отпечата "**error**". В противен случай трябва да се изчисли комисионната (процент комисионна по обем на продажбите) и да се отпечата със закръгляне с точно **2 цифри след десетичната точка**. Може да използвате **`Console.WriteLine("{0:f2}", …)`**.

9. * Точка и правоъгълник – графично (GUI) приложение

Да се разработи графично (**GUI**) приложение за **визуализация на точка и правоъгълник**. Приложението трябва да изглежда приблизително по следния начин:







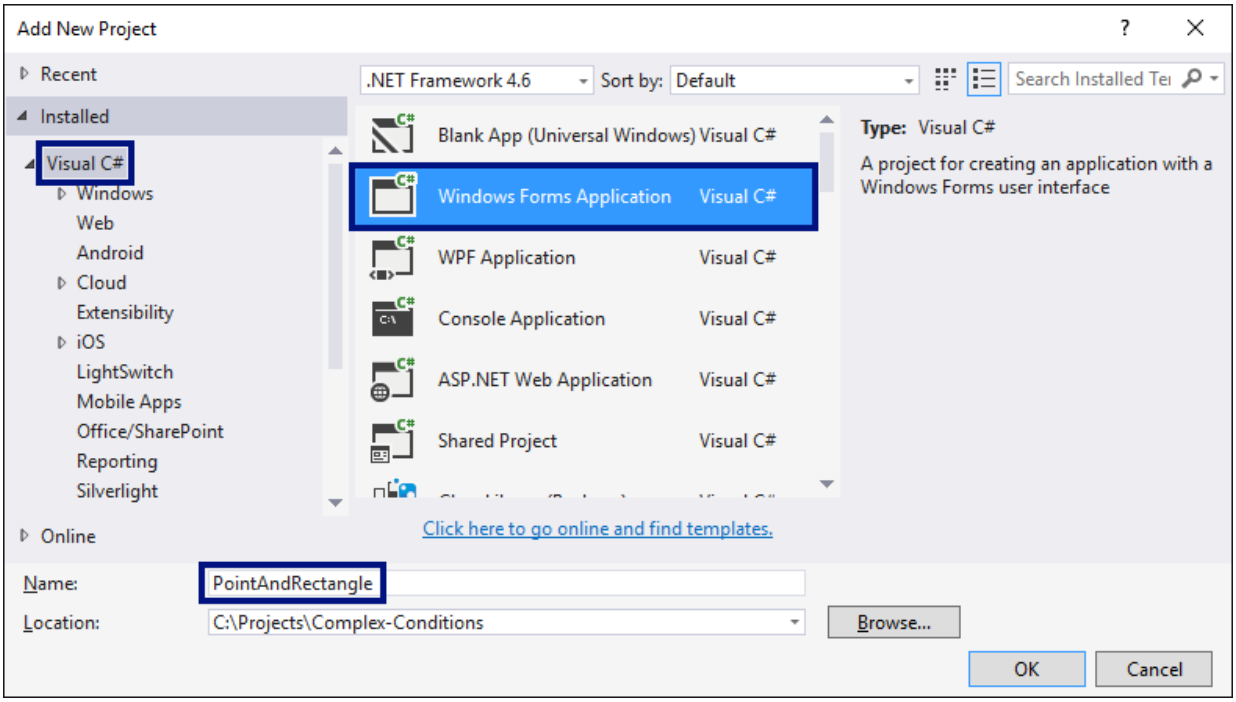
От контролите вляво се задават координатите на **два от ъглите на правоъгълник** (десетични числа) и координатите на **точка**. Приложението **визуализира графично** правоъгълника и точката и изписва дали точката е **вътре** в правоъгълника (**Inside**), **вън** от него (**Outside**) или на някоя от стените му (**Border**).

Приложението **премества** и **мащабира** координатите на правоъгълника и точката, за да бъдат максимално големи, но да се събират в полето за визуализация в дясната страна на приложението.

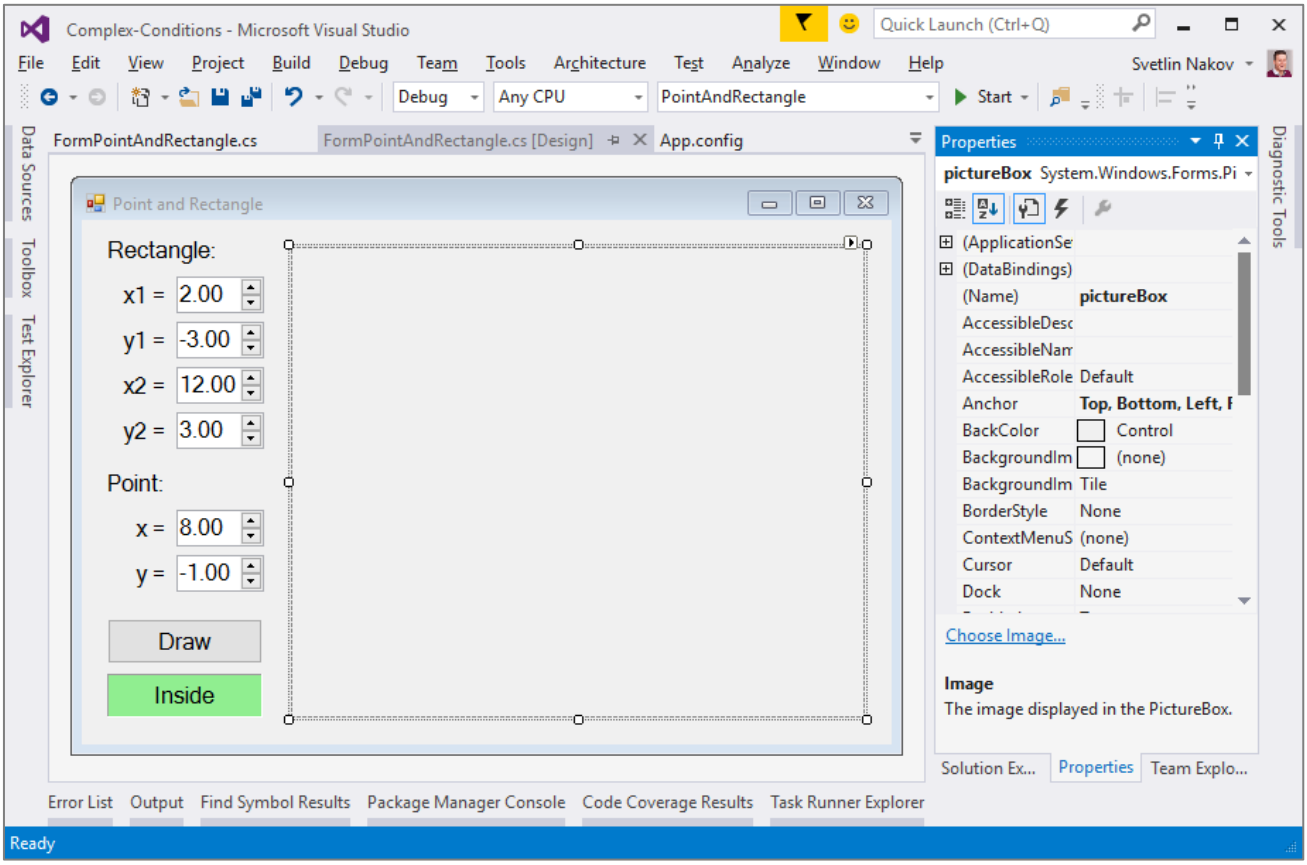
Внимание: това приложение е значително **по-сложно** от предходните графични приложения, които разработвахте до сега, защото изисква ползване на функции за чертане и нетривиални изчисления за

преоразмеряване и преместване на правоъгълника и точката. Следват инструкции за изграждане на приложението стъпка по стъпка.

1. Създайте нов **Windows Forms Application** с подходящо име, например “**`Point-and-Rectangle`**”:



2. **Наредете контролите** във формата както е показано на фигурата по-долу: 6 кутийки за въвеждане на число (**`NumericUpDown`**) с надписи (**`Label`**) пред всяка от тях, бутон (**`Button`**) за изчертаване на правоъгълника и точката и текстов блок за резултата (**`Label`**). Нагласете **размерите** и свойствата на контролите, за да изглеждат долу-горе като на картинката:



3. Задайте следните препоръчителни **настройки на контролите**:

За **главната форма** (**Form**), която съдържа всички контроли:

- (name) = **FormPointAndRectangle**
- **Text** = "**Point and Rectangle**"
- **Font.Size** = **12**
- **Size** = **700, 410**
- **MinimumSize** = **500, 400**
- **FormBorderStyle** = **FixedSingle**

За **полетата за въвеждане на число** (**NumericUpDown**):

- (name) = **numericUpDownX1**; **numericUpDownY1**; **numericUpDownX2**; **numericUpDownY2**; **numericUpDownX**; **numericUpDownY**
- **Value** = **2**; **-3**; **12**; **3**; **8**; **-1**
- **Minimum** = **-100000**
- **Maximum** = **100000**
- **DecimalPlaces** = **2**

За **бутона** (**Button**) **за визуализация** на правоъгълника и точката:

- (name) = **buttonDraw**
- **Text** = "**Draw**"

За **текстовия блок за резултата** (**Label**):

- (name) = **labelLocation**
- **AutoSize** = **False**
- **BackColor** = **PaleGreen**
- **TextAlign** = **MiddleCenter**

За **полето с чертежа** (**PictureBox**):

- (name) = **pictureBox**
- **Anchor** = **Top**, **Bottom**, **Left**, **Right**

4. Хванете следните **събития**, за да напишете C# кода, който ще се изпълни при настъпването им:
   - Събитието **Click** на бутона **buttonDraw** (извиква се при натискане на бутона).
   - Събитието **ValueChanged** на контролите за въвеждане на числа **numericUpDownX1**, **numericUpDownY1**, **numericUpDownX2**, **numericUpDownY2**, **numericUpDownX** и **numericUpDownY** (извиква се при промяна на стойността в контролата за въвеждане на число).
   - Събитието **Load** на формата **FormPointAndRectangle** (извиква се при стартиране на приложението, преди да се появи главната форма на екрана).
   - Събитието **Resize** на формата **FormPointAndRectangle** (извиква се при промяна на размера на главната формата).
5. Всички изброени по-горе събития ще изпълняват едно и също действие – **Draw()**, което ще визуализира правоъгълника и точката и ще показва дали тя е вътре, вън или на някоя от страните. Кодът трябва да прилича на този:

```
private void buttonDraw_Click(object sender, EventArgs e)
{
    Draw();
}
```

```
private void FormPointAndRectangle_Load(object sender, EventArgs e)
{
    Draw();
}

private void FormPointAndRectangle_Resize(object sender, EventArgs e)
{
    Draw();
}

private void numericUpDownX1_ValueChanged(object sender, EventArgs e)
{
    Draw();
}

// TODO: implement the same way event handlers numericUpDownY1_ValueChanged,
numericUpDownX2_ValueChanged, numericUpDownY2_ValueChanged,
numericUpDownX_ValueChanged and numericUpDownY_ValueChanged

private void Draw()
{
    // TODO: implement this a bit later …
}
```

6. Започнете от по-лесната част: **печат на информация къде е точката спрямо правоъгълника** (Inside, Outside или Border). Можете да ползвате следния код:

```
private void Draw()
{
    // Get the rectangle and point coordinates from the form
    var x1 = this.numericUpDownX1.Value;
    var y1 = this.numericUpDownY1.Value;
    var x2 = this.numericUpDownX2.Value;
    var y2 = this.numericUpDownY2.Value;
    var x = this.numericUpDownX.Value;
    var y = this.numericUpDownY.Value;

    // Display the location of the point: Inside / Border / Outside
    DisplayPointLocation(x1, y1, x2, y2, x, y);
}

private void DisplayPointLocation(
    decimal x1, decimal y1, decimal x2, decimal y2, decimal x, decimal y)
{
    var left = Math.Min(x1, x2);
    var right = Math.Max(x1, x2);
    var top = Math.Min(y1, y2);
    var bottom = Math.Max(y1, y2);
    if (x > left && x < right && …)
    {
        this.labelLocation.Text = "Inside";
        this.labelLocation.BackColor = Color.LightGreen;
    }
    else if (… || y < top || y > bottom)
    {
        this.labelLocation.Text = "Outside";
```

```
            this.labelLocation.BackColor = Color.LightSalmon;
        }
        else
        {
            this.labelLocation.Text = "Border";
            this.labelLocation.BackColor = Color.Gold;
        }
}
```

Помислете как **да допишете** недовършените (нарочно) условия в if-проверките! Кодът по-горе нарочно не се компилира, защото целта му е да помислите как и защо работи и да допишете липсващите части.

Горният код взима координатите на правоъгълника и точките и проверява дали точката е вътре, вън или на страната на правоъгълника. При визуализацията на резултата се сменя и цвета на фона на текстовия блок, който го съдържа.

7. Остава да се имплементира най-сложната част: визуализация на правоъгълника и точката в контролата **`pictureBox`** с преоразмеряване. Можете да ползвате **кода по-долу**, който прави малко изчисления и рисува син правоъгълник и тъмносиньо кръгче (точката) според зададените във формата координати. За съжаление сложността на кода надхвърля изучавания до момента материал и е сложно да се обясни в детайли как точно работи. Можете да разгледате коментарите за ориентация. Това е пълната версия на действието **`Draw()`**:

```
private void Draw()
{
  // Get the rectangle and point coordinates from the form
  var x1 = this.numericUpDownX1.Value;
  var y1 = this.numericUpDownY1.Value;
  var x2 = this.numericUpDownX2.Value;
  var y2 = this.numericUpDownY2.Value;
  var x = this.numericUpDownX.Value;
  var y = this.numericUpDownY.Value;

  // Display the location of the point: Inside / Border / Outside
  DisplayPointLocation(x1, y1, x2, y2, x, y);

  // Calculate the scale factor (ratio) for the diagram holding the
  // rectangle and point in order to fit them well in the picture box
  var minX = Min(x1, x2, x);
  var maxX = Max(x1, x2, x);
  var minY = Min(y1, y2, y);
  var maxY = Max(y1, y2, y);
  var diagramWidth = maxX - minX;
  var diagramHeight = maxY - minY;
  var ratio = 1.0m;
  var offset = 10;
  if (diagramWidth != 0 && diagramHeight != 0)
  {
    var ratioX = (pictureBox.Width - 2 * offset - 1) / diagramWidth;
    var ratioY = (pictureBox.Height - 2 * offset - 1) / diagramHeight;
    ratio = Math.Min(ratioX, ratioY);
  }

  // Calculate the scaled rectangle coordinates
  var rectLeft = offset + (int)Math.Round((Math.Min(x1, x2) - minX) * ratio);
```

```
    var rectTop = offset + (int)Math.Round((Math.Min(y1, y2) - minY) * ratio);
    var rectWidth = (int)Math.Round(Math.Abs(x2 - x1) * ratio);
    var rectHeight = (int)Math.Round(Math.Abs(y2 - y1) * ratio);
    var rect = new Rectangle(rectLeft, rectTop, rectWidth, rectHeight);

    // Calculate the scalled point coordinates
    var pointX = (int)Math.Round(offset + (x - minX) * ratio);
    var pointY = (int)Math.Round(offset + (y - minY) * ratio);
    var pointRect = new Rectangle(pointX - 2, pointY - 2, 5, 5);

    // Draw the rectangle and point
    pictureBox.Image = new Bitmap(pictureBox.Width, pictureBox.Height);
    using (var g = Graphics.FromImage(pictureBox.Image))
    {
      // Draw diagram background (white area)
      g.Clear(Color.White);

      // Draw the rectangle (scalled to the picture box size)
      var pen = new Pen(Color.Blue, 3);
      g.DrawRectangle(pen, rect);

      // Draw the point (scalled to the picture box size)
      pen = new Pen(Color.DarkBlue, 5);
      g.DrawEllipse(pen, pointRect);
    }
}

private decimal Min(decimal val1, decimal val2, decimal val3)
{
  return Math.Min(val1, Math.Min(val2, val3));
}

private decimal Max(decimal val1, decimal val2, decimal val3)
{
  return Math.Max(val1, Math.Max(val2, val3));
}
```

В горния код се срещат доста **преобразувания на типове**, защото се работи с различни типове числа (десетични числа, реални числа и цели числа) и понякога се изисква да се преминава между тях.

8. **Компилирайте кода**. Ако има някакви грешки, ги отстранете. Най-вероятната причина за грешка е несъответстващо име на някоя от контролите или ако сте написали кода на неправилно място.
9. **Стартирайте приложението** и го **тествайте** (с разцъкване). Пробвайте да въвеждате различни правоъгълници и позиционирайте точката на различни позиции, преоразмерявайте приложението и вижте дали се държи коректно.

10. Кино

В една кинозала столовете са наредени в правоъгълна форма в **r** реда и **c** колони. Има три вида прожекции с билети на различни цени:

- **Premiere** – премиерна прожекция, на цена **12.00** лева.
- **Normal** – стандартна прожекция, на цена **7.50** лева.
- **Discount** – прожекция за деца, ученици и студенти на намалена цена от **5.00** лева.

Напишете програма, която въвежда **тип прожекция** (стринг), брой **редове** и брой **колони** в залата (цели числа) и изчислява общите приходи от билети при пълна зала. Резултатът да се отпечата във формат като в примерите по-долу, с 2 знака след десетичната точка. Примери:

вход	изход
Premiere 10 12	1440.00 leva

вход	изход
Normal 21 13	2047.50 leva

вход	изход
Discount 12 30	1800.00 leva

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#8.

* **Подсказка**: използвайте прости проверки и елементарни изчисления. За да изведете резултата с точно 2 цифри след десетичната точка, използвайте **`Console.WriteLine("{0:f2}", result)`**.

11. Волейбол

Влади е студент, живее в София и си ходи от време на време до родния град. Той е много запален по волейбола, но е зает през работните дни и играе **волейбол** само през **уикендите** и в **празничните дни**. Влади играе **в София** всяка **събота**, когато **не е на работа** и **не си пътува до родния град**, както и в **2/3 от празничните дни**. Той пътува до **родния си град h пъти** в годината, където играе волейбол със старите си приятели в **неделя**. Влади **не е на работа 3/4 от уикендите**, в които е в София. Отделно, през **високосните години** Влади играе с **15% повече** волейбол от нормалното. Приемаме, че годината има точно **48 уикенда**, подходящи за волейбол.

Напишете програма, която изчислява **колко пъти Влади е играл волейбол** през годината. **Закръглете резултата** надолу до най-близкото цяло число (например 2.15 → 2; 9.95 → 9).

Входните данни се четат от конзолата:

- Първият ред съдържа думата "**`leap`**" (високосна година) или "**`normal`**" (невисокосна).
- Вторият ред съдържа цялото число **p** – брой празници в годината (които не са събота и неделя).
- Третият ред съдържа цялото число **h** – брой уикенди, в които Влади си пътува до родния град.

Примери:

вход	изход	Коментари
leap 5 2	45	48 уикенда в годината, разделени по следния начин: • 46 уикенда в София → 46 * 3 / 4 → **34.5** съботни игри в София • 2 уикенда в родния си град → 2 недели → **2** игри в неделя в родния град 5 празника: • 5 * 2/3 → **3.333** игри в София в празничен ден Общо игри през уикенди и празници в София и в родния град: 34.5 + 2 + 3.333 → **39.833** Годината е високосна: • Влади играе допълнителни 15% * 39.833 → **5.975** игри волейбол Общо игри през цялата година: • 39.833 + 5.975 = **45.808** игри • Резултатът е **45** (закръгля се надолу)

вход	изход
normal 3 2	38

вход	изход
leap 2 3	43

вход	изход
normal 11 6	44

вход	изход
leap 0 1	41

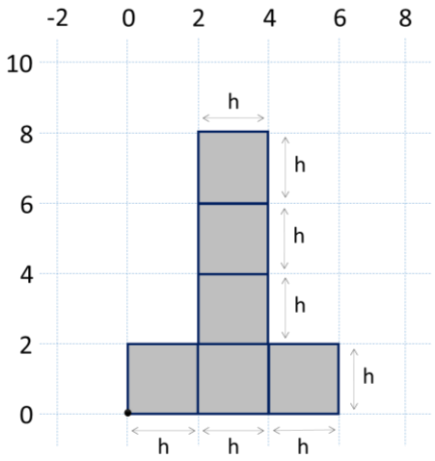
вход	изход
normal 6 13	43

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#9.

* **Подсказки**:

- Пресметнете **уикендите в София** (48 минус уикендите в родния град). Пресметнете **броя игри в уикендите в София**: умножете уикендите в София с (3.0 / 4). Обърнете внимание, че трябва да се използва **дробно деление** (3.0 / 4), а не целочислено (3 / 4).
- Пресметнете **броя игри в родния град**. Те са точно колкото са пътуванията до родния град.
- Пресметнете **броя игри в празничен ден**. Те са броя празници умножени по (2.0 / 3).
- **Сумирайте** броя на всички игри. Той е дробно число. Не бързайте да закръгляте още.
- Ако годината е **високосна**, добавете **15%** към общия брой игри.
- Накрая **закръглете** надолу до най-близкото цяло число с `Math.Truncate(result)`.

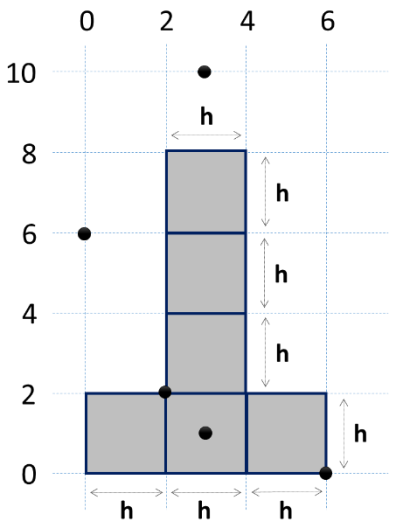
12. * Точка във фигурата

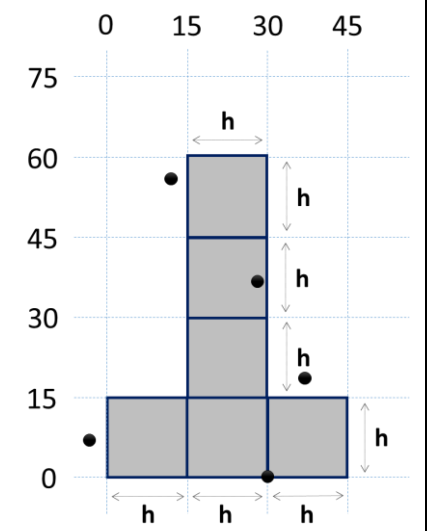


Фигура се състои от **6 блокчета с размер h * h**, разположени като на фигурата вдясно. Долният ляв ъгъл на сградата е на позиция {0, 0}. Горният десен ъгъл на фигурата е на позиция {**2*h**, **4*h**}. На фигурата координатите са дадени при **h = 2**.

Напишете програма, която въвежда цяло число **h** и координатите на дадена **точка** {**x**, **y**} (цели числа) и отпечатва дали точката е вътре във фигурата (`inside`), вън от фигурата (`outside`) или на някоя от стените на фигурата (`border`).

Примери:

вход	изход	визуализация
2 3 10	outside	
2 3 1	inside	
2 2 2	border	
2 6 0	border	
2 0 6	outside	

вход	изход	визуализация
15 13 55	outside	
15 29 37	inside	
15 37 18	outside	
15 -4 7	outside	
15 30 0	border	

Тествайте решението си в **judge системата**: https://judge.softuni.bg/Contests/Practice/Index/153#10.

* **Подсказки**:

- Може да разделите фигурата на **два правоъгълника** с обща стена.
- Една точка е **външна** (`outside`) за фигурата, когато е едновременно **извън** двата правоъгълника.
- Една точка е **вътрешна** (`inside`) за фигурата, ако е вътре в някой от правоъгълниците (изключвайки стените им) или лежи върху общата им стена.
- В **противен случай** точката лежи на стената на правоъгълника (`border`).