# Lab: Methods - Defining and Calling Methods

Problems for exercises and homework for the [“Programming Fundamentals Extended” course @ SoftUni](https://softuni.bg/courses/programming-fundamentals).

You can check your solutions here: <https://judge.softuni.bg/Contests/417>.

# Declaring and Invoking Methods

## Blank Receipt

Create a method that prints a blank cash receipt. The method should invoke three other methods: one for printing the header, one for the body and one for the footer of the receipt.

|  |  |
| --- | --- |
| The header should contain the following text: | CASH RECEIPT  ------------------------------ |
| The body should contain the following text: | Charged to\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Received by\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| And the text for the footer: | ------------------------------  © SoftUni |

### Examples

|  |
| --- |
| **Output** |
| CASH RECEIPT  ------------------------------  Charged to\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Received by\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ------------------------------  © SoftUni |

### Hints

1. First create a method with no parameters for printing the header starting with **static void**. Give it a **meaningful name** like "**PrintReceiptHeader**" and write the code that it will execute:



1. Do the same for printing the receipt body and footer.
2. Create a **method that will call all three methods** in the necessary order. Again, give it a **meaningful and descriptive name** like "PrintReceipt" and write the code:



1. For printing **"©"** use Unicode **"\u00A9"**
2. **Call** (invoke) the PrintReceipt method from the Main method.



## Sign of Integer Number

Create a method that prints the sign of an integer number n.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2 | The number 2 is positive. |
| -5 | The number -5 is negative. |
| 0 | The number 0 is zero. |

### Hints

1. Create a method with a **descriptive name** like "PrintSign" which should receive **one parameter** of type **int**.



1. Implement the body of the method by handling different cases:
   1. If the number is greater than zero
   2. If the number is less than zero
   3. And if the number is equal to zero
2. Call (invoke) the newly created method from the main.



## Printing Triangle

Create a method for printing triangles as shown below:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3 | 1  1 2  1 2 3  1 2  1 |
| 4 | 1  1 2  1 2 3  1 2 3 4  1 2 3  1 2  1 |

### Hints

1. After you read the input
2. Start by creating a method **for printing a single line** from a **given start** to a **given end**. Choose a **meaningful name** for it, describing its purpose:



1. Think how you can use it to solve the problem
2. After you spent some time thinking, you should have come to the conclusion that you will need two loops
3. In the first loop you can print the first half of the triangle without the middle line:



1. Next, print the middle line:



1. Lastly, print the rest of the triangle:



## Draw a Filled Square

Draw at the console a filled square of size n like in the example:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4 | --------  -\/\/\/-  -\/\/\/-  -------- |

### Hints

1. Read the input
2. Create a method which will print the top and the bottom rows (they are the same). Don’t forget to give it a descriptive name and to give it as a parameter some length
   1. Instead of loop you can use the "new string" command which creates a new string consisting of a character repeated some given times:



1. Create the method which will print the middle rows. Well, of course, you should probably name it "PrintMiddleRow"



1. Use the methods that you've just created to draw a square



# Returning Values and Overloading

## Calculate Triangle Area

Create a method that calculates and **returns** the [area](http://www.mathopenref.com/trianglearea.html) of a triangle by given base and height:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3  4 | 6 |

### Hints

1. After reading the input
2. Create a method, but this time **instead** of typing **"static void"** before its name, type **"static double"** as this will make it to **return a value of type double**:



1. **Invoke** the method in the main and **save the return value in a new variable**:



## Math Power

Create a method that calculates and returns the value of a number raised to a given power:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2  8 | 256 |
| 3  4 | 81 |

### Hints

1. As usual, read the input
2. Create a method which will have two parameters - the number and the power, and will return a result of type double:



1. Print the result

## Greater of Two Values

You are given two values of the same type as input. The values can be of type int, char of string. Create a method GetMax() that returns the greater of the two values:

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| int  2  16 | 16 |
| char  a  z | z |
| string  Ivan  Todor | Todor |

### Hints

1. For this method you need to create three methods with the same name and different signatures
2. Create a method which will compare integers:



1. Create a second method with the same name which will compare characters. Follow the logic of the previous method:



1. Lastly you need to create a method to compare strings. This is a bit different as strings don't allow to be compared with the operators > and <



You need to use the method "CompareTo()", which returns an integer value (greater than zero if the compared object is greater, less than zero if the compared object is lesser and zero if the two objects are equal.

1. The last step is to read the input, use appropriate variables and call the GetMax() from your Main():

