Lab: Methods

Problems for in-class lab for the "C# Fundamentals" course @ SoftUni

You can check your solutions in Judge

I. Declaring and Invoking Methods

1. Sign of Integer Numbers

A single integer is given, create a method that checks if the given number is **positive**, **negative,** or **zero.** As a result print:

- For positive number: **"The number {number} is positive."**
- For negative number: **"The number {number} is negative."**
- For zero number: **"The number {number} is zero."**

Examples

Input	Output
2	The number 2 is positive.
-9	The number -9 is negative.

2. Grades

Create a method **that receives** a grade between **2.00** and **6.00** and **prints the corresponding grade definition**:

- 2.00 – 2.99 - "**Fail**"
- 3.00 – 3.49 - "**Poor**"
- 3.50 – 4.49 - "**Good**"
- 4.50 – 5.49 - "**Very good**"
- 5.50 – 6.00 - "**Excellent**"

Examples

Input	Output
3.33	Poor
4.50	Very good
2.99	Fail

Hints

1. Read the grade from the console and pass it to a method

```
using System;

public class Test
{
    public static void Main()
    {
        double grade = double.Parse(Console.ReadLine());

        PrintInWords(grade);
    }
```

2. Then create the method and make the `if` statements for each case

```
private static void PrintInWords(double grade)
{
    if (grade >= 2.00 && grade <= 2.99)
    {
        Console.WriteLine("Fail");
    }

    //TODO: make the rest
}
```

3. Calculations

Create a program that receives three lines of input:

- On the first line – a **string** – "**add**", "**multiply**", "**subtract**", "**divide**".
- On the second line – a number.
- On the third line – another number.

You should create **four methods** (for each calculation) and invoke the corresponding method depending on the command. The method should also print the result (needs to be void).

Example

Input	Output
subtract 5 4	1
divide 8 4	2

Hints

1. Read the command on the first line, and the two numbers, and then make an `if/switch` statement for each type of calculation

```
static void Main(string[] args)
{
    string command = Console.ReadLine();
    int a = int.Parse(Console.ReadLine());
    int b = int.Parse(Console.ReadLine());

    switch (command)
    {
        case "add":
            Add(a, b);
            break;
        case "subtract":
            Subtract(a, b);
            break;

        //TODO: check for the rest of the commands
    }
}
```

2. Then create the four methods and print the result

```
private static void Multiply(int a, int b)
{
    Console.WriteLine(a * b);
}

private static void Divide(int a, int b)
{
    Console.WriteLine(a / b);
}

//TODO: create the rest of the methods
```

4. Printing Triangle

Create a method for printing triangles as shown below:

Examples

Input	Output

3	1
	1 2
	1 2 3
	1 2
	1

2	1
	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:

```
static void PrintLine(int start, int end)
{
    for (int i = start; i <= end; i++)
    {
        Console.Write(i + " ");
    }
    Console.WriteLine();
}
```

3. Create another method for printing the whole triangle. Again choose a **meaningful name** for it, describing its purpose.
4. Think how you can use the **`PrintLine()`** method to solve the problem.
5. After you spent some time thinking, you should have concluded that you need two loops.
6. In the first loop you can print the first half of the triangle:

```
for (int i = 1; i <= n; i++)
{
    PrintLine(1, i);
}
```

7. In the second loop you can print the second half of the triangle:

```
for (int i = n - 1; i >= 1; i--)
{
    PrintLine(1, i);
}
```

5. Orders

Create a program that calculates and prints the total price of an order. The method should receive two parameters:

- A **string, representing a product** - **`"coffee"`**, **`"water"`**, **`"coke"`**, **`"snacks"`**
- An integer, representing the **quantity** of the product

The prices for a single item of each product are:

- coffee – 1.50
- water – 1.00
- coke – 1.40
- snacks – 2.00

Print the result, rounded to the second decimal place.

Example

Input	Output
water 5	5.00
coffee 2	3.00

Hints

1. Read the first two lines
2. Create a method to pass the two variables in
3. Print the result in the method

II. Returning Values and Overloading

6. Calculate Rectangle Area

Create a method that calculates and **returns** the area of a rectangle.

Examples

Input	Output
3 4	12
6 2	12

Hints

1. Read 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 **return a value of type double**:

```
static double GetRectangleArea(double width, double height)
{
    return width * height;
}
```

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

```
double width = double.Parse(Console.ReadLine());
double height = double.Parse(Console.ReadLine());
double area = GetRectangleArea(width, height);
Console.WriteLine(area);
```

7. Repeat String

Create a method that receives two parameters:

- A **string**
- A number **n** (integer) represents how many times the string will be repeated

The method should return a new **string**, containing the initial one, repeated **n** times without space.

Example

Input	Output

abc 3	abcabcabc
String 2	StringString

Hints

1. First, read the **string** and the repeat count **n**
2. Then create the method and pass the variables to it

```
private static string RepeatString(string str, int count)
{
    string result = "";

    for (int i = 0; i < count; i++)
    {
        //TODO: append the string to the result
    }

    return result;
}
```

3. In the main method, print the result

8. Math Power

Create a method, which receives two numbers as parameters:

- The first number – the **base**
- The second number – the **power**

The method should return the **base** raised to the given **power**.

Examples

Input	Output
2 8	256
3 4	81

Hints

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

```
static double RaiseToPower(double number, int power)
{
    double result = 0d;

    // TODO: Calculate result (use a loop, or Math.Pow())

    return result;
}
```

3. Print the result.

9. Greater of Two Values

You are given an input of two values of the same type. The values can be of type **`int`**, **`char`** or **`string`**. Create methods called **GetMax(), which can compare `int`, `char` or `string`** and returns the **biggest of the two values**.

Examples

Input	Output
int 2 16	16
char a z	z
string aaa bbb	bbb

Hints

Use method overloading.

10. Multiply Evens by Odds

Create a program that **multiplies the sum** of **all even digits** of a number **by the sum of all odd digits** of the same number:

- Create a method called **GetMultipleOfEvenAndOdds()**
- Create a method **GetSumOfEvenDigits()**
- Create **GetSumOfOddDigits()**
- You may need to use **Math.Abs()** for negative numbers

Examples

Input	Output	Comment
-12345	54	Evens: 2 4 Odds: 1 3 5 Even sum: 6 Odd sum: 9 6 * 9 = 54

3453466	220	

11. Math Operations

Write a method that receives **two numbers** and an **operator**, calculates the result and returns it. You will be given **three lines of input**. The first will be the first **number**, the second one will be the **operator** and the last one will be the **second number**.

The possible operators are: **/**, *****, **+** and **-**.

Example

Input	Output
5 * 5	25
4 + 8	12

Hint

1. Read the inputs and create a method that returns a double (the result of the operation)

```
private static double Calculate(int a, string @operator, int b)
{
    double result = 0;

    switch (@operator)
    {
        //TODO: check for all the possible operands and calculate the result
    }

    return result;
}
```