Lab: Methods

Problems for in-class lab for the "Programming Fundamentals: Arrays and Lists" course from the official "Applied Programmer" curriculum.
You can check your solutions in https://judge.softuni.bg/Contests/2909.

I. Declaring and Invoking Methods

1. Grades

Write a method that **receives a grade** between **2.00** and **6.00** and **prints the corresponding grade in words**

- 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

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

```
double grade = double.Parse(Console.ReadLine());

PrintInWords(grade);
```

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

```
static void PrintInWords(double grade)
{
    string gradeInWords = string.Empty;

    if (grade >= 2 && grade <= 2.99) gradeInWords = "Fail";
    //TODO: make the rest

    Console.WriteLine(gradeInWords);
}
```

2. Sign of Integer Numbers

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.

3. Calculations

Write a program that receives a **string** on the first line ("**add**", "**multiply**", "**subtract**" or "**divide**") and on the next **two lines** receives **two numbers**. Create **four methods** (for each calculation) and **invoke the right one** 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

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

```
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: Chek for the rest of the commands
}
```

Then create the **four methods** and **print** the result:

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

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
4	1 1 2 1 2 3 1 2 3 4 1 2 3 1 2 1

Hints

After you read the input 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();
}
```

Create another method for printing the whole triangle. Again choose a **meaningful name** for it, describing its purpose. Think how you can use the **PrintLine()** method to solve the problem. After you spent some time thinking, you should have come to the conclusion that you will need **two loops**.

In the first loop you can print the **first half** of the triangle:

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

In the second loop you can **print the second half** of the triangle:

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

II. Returning Values and Overloading

5. Calculate Rectangle Area

Create a method that calculates and **returns** the area of a rectangle by given width and height:

Examples

Input	Output

3 4	12
6 2	12

Hints

Read the input. 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**:

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

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 = RectangleArea(width, height);

Console.WriteLine(area);
```

6. Repeat String

Write a method that **receives a string** and a **repeat count n** (integer). The method should **return a new string** (the old one repeated **n** times).

Example

Input	Output
abc 3	abcabcabc
String 2	StringString

Hints

Firstly read the **string** and the repeat count **n**. Then create the **method** and **pass it the variables.**

```
static string NewText(string text, double n)
{
    string result = "";

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

    return result;
}
```

7. 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

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

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

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

    return result;
}
```

Print the result.

8. Greater of Two Values

Create a method **GetMax()** that **returns the greater** of two values (the values can be of type **int**, **char** or **string**)

Examples

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

9. 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

10. Math operations

Write a method that receives **two number** 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: '**/**', '*****', '**+**', '**-**'.

Print the result **rounded up to the second decimal point**.

Example

Input	Output
5 * 5	25
4 + 8	12

Hint

Read the input and create a method that returns a **double** (the result of the operation)

```
static double Calculate(double a, string command, double b)
{
    double result = 0;

    switch (command)
    {
        //TODO: Check for all the possible operands and calculate the result
    }

    return result;
}
```