Lab: Methods

Problems for exercises and homework for the "Technology Fundamentals" course @ SoftUni.

You can check your solutions in Judge.

I. Declaring and Invoking Methods

1. Sign of Integer

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

Example

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

2. 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

Hint

Read the grade from the console and pass it to a method:

```
import java.util.Scanner;

public class Grades {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        double grade = Double.parseDouble(sc.nextLine());
        printInWords(grade);
    }
```

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){
        System.out.println("Fail");
    }

    //TODO: make the rest
}
```

3. 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, create a method **for printing a single line** from a **given start** to a **given end**. Choose a **meaningful name** for it, describing its purpose:

```
private static void printLine(int start, int end){
    for (int i = start; i <= end; i++) {
        System.out.print(i + " ");
    }
    System.out.println();
}
```

You will need two loops. In the first loop you can print the first half of the triangle without the middle line:

```
for (int i = 0; i < n; i++) {
    printLine(1, i);
}
```

Next, print the middle line:

```
printLine(1, i);
```

Lastly, print the rest of the triangle:

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

4. Calculations

Write a program that receives a string on the first line **("add", "multiply", "subtract", "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:

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    String command = sc.nextLine();
    int a = Integer.parseInt(sc.nextLine());
    int b = Integer.parseInt(sc.nextLine());

    switch (command){
        case "add":
            add(a,b);
            break;
        case "divide":
            divide(a,b);
            break;
        // TODO: check for the rest command
    }
}
```

Then create the four methods and print the result:

```
private static void add(int a, int b){
    System.out.println(a + b);
}

private  static void divide(int a, int b){
    System.out.println(a / b);
}

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

5. Orders

Write a method that calculates the total price of an order and prints it on the console. The method should receive **one of the following products**: coffee, water, coke, snacks; and a **quantity** of the product. The prices for a single piece 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

Hint

- Read the product and the quantity.
- Create a method the pass the two variables in.
- 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 by given width and length.

Examples

Input	Output
3 4	12

Hints

After reading 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**:

```
private static double getRectangleArea(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.parseDouble(sc.nextLine());
double height = Double.parseDouble(sc.nextLine());
double area = getRectangleArea(width, height);
System.out.println(area);
```

7. 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:

```
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;
}
```

8. Math Power

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

Examples

Input	Output
2	256

8	
5.5 3	166.375

Hints

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

```
private static double mathPower(double number, int power) {
    double result = 1;

    //TODO: Calculate result (use a loop or Math.pow())
    return result;
}
```

Invoke the method and use **`DecimalFormat`** to print the result. For the curious - you can read more here.

```
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);

    double number = Double.parseDouble(sc.nextLine());
    int power = Integer.parseInt(sc.nextLine());

    System.out.println(new DecimalFormat( pattern: "0.####").format(mathPower(number, power)));
}
```

9. 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:

```
static int getMax(int firstNum, int secondNum) {
    if (firstNum > secondNum) {
        return firstNum;
    }

    return secondNum;
}
```

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

```
static char getMax(char first, char second) {
    //TODO: Implement the method
}
```

4. 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 "**<**":

```
static String getMax(String first, String second) {
    if (first.compareTo(second) >= 0) {
        //TODO: Return value
    }
    //TODO: Return value
}
```

   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).

5. The last step is to read the input. Use appropriate variables and call the **`getMax()`** from your **`main()`**.

10. Multiply Evens by Odds

Create a program that reads an **integer** number and **multiplies the sum of all its even digits** by **the sum of all its odd digits**:

Examples

Input	Output	Comments
12345	54	12345 has **2 even digits** - 2 and 4. Even digits have **sum of 6.** Also it has **3 odd digits** - 1, 3 and 5. Odd digits have **sum of 9.** **Multiply 6 by 9** and you get **54.**
-12345	54	

Hints

1. Create a method with a **name describing its purpose** (like **`getMultipleOfEvensAndOdds`**). The method should have a **single integer parameter** and an **integer return value**. Also, the method will call two other methods:

```
private static int getMultipleOfEvensAndOdds(int n) {
    int evensSum = getSumOfEvenDigits(n);
    int odsSum = getSumOfOddDigits(n);

    return evensSum * odsSum;
}
```

2. Create two other methods each of which will sum either even or odd digits.
3. Implement the logic for summing even digits:

```
private static int getEvenSum(int n){
    int evensSum = 0;

    // TODO check the digits of the number and if they are even, sum them

    return evensSum;
}
```

4. Do the same for the method that will sum odd digits.
5. As you test your solution you may notice that it doesn't work for negative numbers. Following the program execution line by line, find and fix the bug (**hint: you can use `Math.abs()`**).

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: **/ * + -**

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

Example

Input	Output
5 * 5	25
4 + 8	12

Hint

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 = 0.0;

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

    return result;
}
```