W2 - Instruksi Praktikum PBO
Fundamental Programming Structures in Java

Kerjakan 5 soal dibawah ini dengan mengikuti ketentuan sebagai berikut:

1. Isi sheet monitoring berdasarkan ketentuan yang ada di sheet tersebut.
2. Source code setiap pengerjaan soal, simpan di Github, lampirkan komentar dari hasil pengerjaan tersebut.
3. Buat laporan hasil pengerjaan berbentuk dokumen, upload laporan di folder Hasil Praktikum, laporan harus mencakup:
    1. Cover.
    2. Persoalan yang telah dikerjakan. Setiap persoalan, harus menjawab beberapa deskripsi berikut ini:
        1. Screenshoot hasil akhir program.
        2. Screenshoot setiap jawaban soal yang dipertanyakan.
        3. Permasalahan yang dihadapi.
        4. Solusi dari permasalahan yang dihadapi.
        5. Nama teman yang membantu memecahkan permasalahan di persoalan ini.

Selamat Mengerjakan

Soal 1
Data Types

Java has 8 primitive data types; char, boolean, byte, short, int, long, float, and double. For this exercise, we'll work with the primitives used to hold integer values (byte, short, int, and long):

- A byte is an 8-bit signed integer.
- A short is a 16-bit signed integer.
- An int is a 32-bit signed integer.
- A long is a 64-bit signed integer

Given an input integer, you must determine which primitive data types are capable of properly storing that input.

Input Format
The first line contains an integer, ***T***, denoting the number of test cases. Each test case, ***T***, is comprised of a single line with an integer, ***n***, which can be arbitrarily large or small.

Output Format
For each input variable ***n*** and appropriate primitive ***datatype***, you must determine if the given primitives are capable of storing it. If yes, then print:

```
N can be fitted in:
* datatype
```

If there is more than one appropriate data type, print each one on its own line and order them by size (i.e.: ***byte*** < ***short*** < ***int*** < ***long***).

If the number cannot be stored in one of the four aforementioned primitives, print the line:

```
N can’t be fitted anywhere
```

Sample Input:

```
5
-150
150000
1500000000
213333333333333333333333333333333333
-100000000000000
```

Sample Output:

```
-150 can be fitted in:
* short
* int
* long
150000 can be fitted in:
* int
* long
1500000000 can be fitted in:
* int
* long
21333333333333333333333333333333333333 can't be fitted anywhere.
-100000000000000 can be fitted in:
* long
```

Explanation:

150 can be stored in a short, an int, or a long.
21333333333333333333333333333333333333 is very large and is outside of the allowable range of values for the primitive data types discussed in this problem.

Soal 2
Variables

Perhatikan baris program dibawah ini:

```
public class Constants {
 public static void main(String[] args) {
   final double CM_PER_INCH = 2.54;
   double paperWidth = 8.5;
   double paperHeight = 11;
   System.out.println("Paper size in centimeters: " +
   paperWidth * CM_PER_INCH + " by " + paperHeight *
   CM_PER_INCH);
 }
}
```

```
public class Constants2 {
     public static final double CM_PER_INCH = 2.54;
     public static void main(String[] args) {
        double paperWidth = 8.5;
        double paperHeight = 11;
        System.out.println("Paper size in centimeters: " + paperWidth *
        CM_PER_INCH + " by " + paperHeight * CM_PER_INCH);
     }
}
```

Dari 2 contoh baris program diatas, jawablah pertanyaan dibawah ini:

1. Bagaimana output dari masing masing class Constants dan Constants2?
2. Apa perbedaan penggunaan final double dengan public static final double?

Soal 3
Operators

Perhatikan baris program dibawah ini:

```
Class FloatingPoint{
    public static void main(String[] args) {
         double x = 92.98;
         int nx = (int) Math.round(x);
    }
}
```

Math Class berisi bermacam-macam fungsi matematika seperti pada contoh diatas pada penggunaan round(x), terdapat beberapa pertanyaan yang perlu untuk dijelaskan:

1. Pada kasus berikut jelaskan nilai ***nx*** setelah digunakan **Math.round(x);**
2. Kenapa dibutuhkan cast (int) dalam penggunaan **Math.round(x) ?**

Soal 4
Operators (1)

Perhatikan baris program dibawah ini:

```
class ConvertDataType
{
    static short methodOne(long l)
    {
        int i = (int) l;
        return (short)i;
    }
    public static void main(String[] args)
    {
        double d = 10.25;
        float f = (float) d;
        byte b = (byte) methodOne((long) f);
        System.out.println(b);
    }
}
```

Program berikut melakukan convert tipe data yang berukuran besar ke kecil (long -> int -> short) dan (double -> float -> byte).

1. Jelaskan output nilai dari variable b.
2. Jelaskan apa yang berubah dari variable **d** menjadi variable **b** setelah dilakukan cast ?

Soal 5
Strings

This exercise is to test your understanding of Java Strings. A sample String declaration:

```
String myString = "Hello World!"
```

The elements of a String are called characters. The number of characters in a String is called the length, and it can be retrieved with the String.length() method.

Given two strings of lowercase English letters, ***A*** and ***B***, perform the following operations:

1. Sum the lengths of ***A*** and ***B.***
2. Determine if ***A*** is lexicographically larger than ***B*** (i.e: does ***B*** come before ***A*** in the dictionary?)
3. Capitalize the first letter in ***A*** and ***B*** and print them on a single line, separated by a space.

Input Format

The first line contains a string ***A***. The second line contains another string ***B***. The strings are comprised of only lowercase English letters.

Output Format

There are three lines of output:
For the first line, sum the lengths of ***A*** and ***B***.
For the second line, write Yes if ***A*** is lexicographically greater than ***B*** otherwise print No instead.
For the third line, capitalize the first letter in both ***A*** and and ***B*** print them on a single line, separated by a space.

Sample Input 0

```
hello
java
```

Sample Output 0

```
9
No
Hello Java
```

Explanation 0
String ***A*** is "hello" and ***B*** is "java".
A has a length of 5, and ***B*** has a length of 4; the sum of their lengths is 9.
When sorted alphabetically/lexicographically, "hello" precedes "java"; therefore ***A***, is not greater than ***B*** and the answer is No.

When you capitalize the first letter of both ***A*** and ***B*** and then print them separated by a space, you get "Hello Java".