

LAPORAN PRAKTIKUM

Fundamental Programming Structures in Java

PEMROGRAMAN BERBASIS OBJEK

Laporan ini disusun untuk memenuhi tugas mata kuliah praktikum Pemrograman Berbasis Objek



Disusun oleh:

Nurul Anisah 211511052

**PROGRAM STUDI D3 TEKNIK INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI BANDUNG**

2021

Tugas 1. Data type

```
package DataTypes;

import java.util.Scanner; // Import the Scanner class

/**
 *
 * @author Nurul
 */

public class DataTypes {

    public static void main(String[] args) {

        // TODO Auto-generated method stub

        char temp = 'n';
        while(temp != 'y') {

            Scanner myObj = new Scanner(System.in); // Create a Scanner object

            System.out.print("Enter number :");

            String user = myObj.nextLine(); // Read user input

            try {

                long inp = Long.parseLong(user);

                if(inp < Byte.MAX_VALUE && inp > Byte.MIN_VALUE) {

                    System.out.println(inp + " can be fitted in");

                    System.out.println(" * short\n * int \n * long ");

                }else if(inp < Short.MAX_VALUE && inp < Short.MIN_VALUE) {

                    System.out.println(inp + "can be fitted in");

                    System.out.println(" * int \n *long ");

                }else if(inp > Integer.MIN_VALUE && inp < Integer.MAX_VALUE) {

                    System.out.println(inp + "can be fitted in");

                    System.out.println(" * int \n * long ");

                }else if(inp < Long.MAX_VALUE && inp > Long.MIN_VALUE) {

                    System.out.println(inp + "can be fitted in");

                    System.out.println(" * long ");

                }

            }

        }

    }

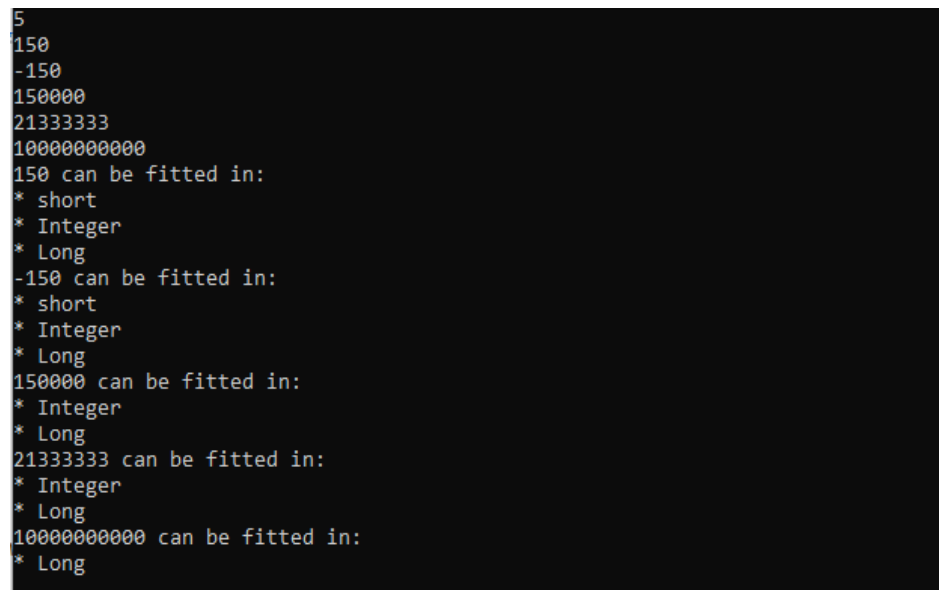
}
```

```

        }catch(NumberFormatException e) {
            System.out.println("can't be fitted anywhere");
        }
        System.out.println();
        System.out.print("Exit ? ");
        temp = myObj.nextLine().charAt(0);
    }
}
}

```

Hasilnya



```

5
150
-150
150000
21333333
10000000000
150 can be fitted in:
* short
* Integer
* Long
-150 can be fitted in:
* short
* Integer
* Long
150000 can be fitted in:
* Integer
* Long
21333333 can be fitted in:
* Integer
* Long
10000000000 can be fitted in:
* Long

```

Karena soal ini merupakan program untuk menentukan sebuah bilangan bertipe byte, short, integer, dan long maka solusi yang didapat adalah :

1. Masukkan input dari user yang berbentuk string lalu masukkan kedalam sebuah array string
2. Melakukan pemeriksaan dengan try and catch memastikan bahwa number tidak melebihi batas maximal type data long.

Tugas 2.

```
public class Constant {  
    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);  
    }  
}  
  
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);  
    }  
}
```

Hasilnya :

```
Paper size in centimeters: 21.59 by 27.94
```

```
Paper size in centimeters: 21.59 by 27.94
```

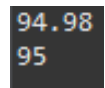
Dari 2 contoh baris program diatas, jawablah pertanyaan dibawah ini:

1. Bagaimana output dari masing masing class Constants dan Constants2?
Outputnya sama
2. Apa perbedaan penggunaan final double dengan public static final double?
Di final double ternyata constantnya tidak bisa diubah karena termasuk perintah lokal, sedangkan di public static final double termasuk variabel global bertipe constant.

Tugas 3.

```
class FloatingPoint{  
    public static void main(String[] args) {  
        double x = 94.98;  
        int nx = (int) Math.round(x);  
  
        System.out.println(x);  
        System.out.println(nx);  
    }  
}
```

Hasilnya

A screenshot of a Java program's output. It shows two lines of text: "94.98" on the first line and "95" on the second line. The text is white on a dark background.

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

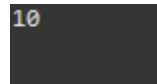
1. Pada kasus berikut jelaskan nilai nx setelah digunakan Math.round(x)
Nilai nx terjadinya pembulatan dari 94.98 menjadi 95
2. Kenapa dibutuhkan cast (int) dalam penggunaan Math.round(x) ?
math.round menghasilkan bilangan bulat maka harus cast (int)

Tugas 4

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);  
    }  
}
```

Hasilnya



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.
Variable b diisi oleh hasil return dari methodOne dengan log f dimana f merupakan hasil cast dari tipe data double variable d. Pada methodOne terdapat 2 cast yaitu long ke integer yang dimasukkan ke variable i. yang kemudian hasil returnnya bertipe short.
2. Jelaskan apa yang berubah dari variable d menjadi variable b setelah dilakukan cast ?
Variable d menjadi variable b dilakukan cast, yang berubah adalah nilai dan juga type datanya karena melalui beberapa cast, dari yang awalnya var d bertipe double dicast menjadi float lalu long yang dikirim ke method dan di cast lagi ke integer lalu di sort dan di return dan di cast dengan byte sebagai assign dari var b.

Tugas 5

```
import java.util.Scanner;

public class soal5 {

    public static void main(String[] args) {

        Scanner obj = new Scanner(System.in);

        String str1, str2;

        str1 = obj.nextLine();

        //str2 = obj.nextLine();

        System.out.println(str1.length() + str2.length());

//

        if(str1.charAt(0) < str2.charAt(0)) {

            System.out.println("No");

        }

        else {

            System.out.println("Yes");

        }

        String KataDepan_Kata1 = str1.substring(0,1);

        String KataBelakang_Kata1 = str1.substring(1);

        String KataDepanDiperbesar = KataDepan_Kata1.toUpperCase();

        //System.out.println(KataDepanDiperbesar + KataBelakang_Kata1);

        String KataDepan_Kata2 = str2.substring(0,1);

        String KataBelakang_Kata2 = str2.substring(1);

        String KataDepanDiperbesar2 = KataDepan_Kata2.toUpperCase();

        System.out.println(KataDepanDiperbesar + KataBelakang_Kata1 + " " +

        KataDepanDiperbesar2 + KataBelakang_Kata2);

    }

}
```

Hasilnya



```
himakom
polban
13
```

1. Menjumlahkan karakter dari 2 inputan
`System.out.println(str1.length() + str2.length());`

Script diatas merupakan proses menjumlahkan length 2 variable. Misal inputan himakom dan polban maka length dari himakom adalah 7 dan polban adalah 6 sehingga $7 + 6 = 13$.

2. Menentukan apakah lexicographi atau str1 lebih besar dari str2
Untuk menentukan variable mana yang lebih besar yaitu dengan menggunakan inputan `(str1.charAt(0) < str2.charAt(0))`. Setelah dianalisis ketika hasil string 1 lebih kecil dari string ke 2 maka menghasilkan **no** lexicography.
3. Menggabungkan 2 string
menggabungkan 2 string di sini adalah huruf pertama harus kapital sehingga yang dilakukan pertamakali adalah mengubahnya menjadi uppercase