

LAPORAN PRAKTIKUM 2
Pemrograman Berbasis Objek



Disusun Oleh :
Muhammad Wildan Gumilang (231511087)

Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Bandung

Soal 1 : Data Types

Kode program :

```
import java.util.Scanner;

public class Soal1 {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Masukkan angka : ");

        long n = scanner.nextLong();

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

        try {

            if (n >= Byte.MIN_VALUE && n <= Byte.MAX_VALUE) {

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

            }

            if (n >= Short.MIN_VALUE && n <= Short.MAX_VALUE) {

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

            }

            if (n >= Integer.MIN_VALUE && n <= Integer.MAX_VALUE) {

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

            }

            if (n >= Long.MIN_VALUE && n <= Long.MAX_VALUE) {

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

            }

        } catch (Exception e) {

            System.out.println(scanner.next() + " can't be fitted anywhere.");

        }

        scanner.close();

    }

}
```

Output :

```
n2_aa7c008b\bin' 'Soal11'  
Masukkan angka : 5  
5 can be fitted in :  
* byte  
* short  
* int  
* long  
PS D:\wg\Kuliah\semester
```

```
n2_aa7c008b\bin' 'Soal11'  
Masukkan angka : -150  
-150 can be fitted in :  
* short  
* int  
* long  
PS D:\wg\Kuliah\semester
```

```
n2_aa7c008b\bin' 'Soal11'  
Masukkan angka : 150000  
150000 can be fitted in :  
* int  
* long  
PS D:\wg\Kuliah\semester
```

```
n2_aa7c008b\bin' 'Soal11'  
Masukkan angka : 1500000000  
1500000000 can be fitted in :  
* int  
* long  
PS D:\wg\Kuliah\semester 3\PBO\
```

```
n2_aa7c008b\bin' 'Soal11'  
Masukkan angka : -1000000000000000  
-1000000000000000 can be fitted in :  
* long  
PS D:\wg\Kuliah\semester 3\PBO\praktek
```

Kesimpulan :

Untuk setiap bilangan yang diberikan, kita perlu menentukan tipe data primitif mana yang bisa menyimpan angka tersebut dengan benar. Tipe data yang digunakan adalah byte, short, int, dan long, mulai dari yang paling kecil hingga yang terbesar. Jika bilangan tersebut bisa disimpan dalam satu atau lebih tipe data, kita harus mencetak tipe-tipe tersebut dari yang terkecil hingga yang terbesar. Jika bilangan tersebut terlalu besar untuk disimpan dalam tipe data long, maka kita harus mencetak bahwa bilangan tersebut tidak bisa disimpan di tipe data mana pun. Ini memastikan bahwa kita bisa menyimpan bilangan dengan aman tanpa risiko kesalahan karena nilai yang terlalu besar atau kecil.

Soal 2 : Variable

Kode program :

```
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 Constant2 {  
    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);  
    }  
}
```

Output :

Constants.jawa

```
n2_aa7c008b\bin' 'Constants'  
Paper size in centimeters: 21.59 by 27.94  
PS D:\wg\Kuliah\semester 3\PBO\praktek\pertemuan2>
```

Constants2.java

```
n2_aa7c008b\bin' 'Constant2'  
Paper size in centimeters: 21.59 by 27.94  
PS D:\wg\Kuliah\semester 3\PBO\praktek\pertemuan2>
```

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

1. Bagaimana output dari masing masing class Constants dan Constants2?

Jawab :

Constants.java :

Paper size in centimeters: 21.59 by 27.94

Constants2.java :

Paper size in centimeters: 21.59 by 27.94

2. Apa perbedaan penggunaan final double dengan public static final double?

Jawab :

Perbedaan antara final double dan public static final double yang dipakai dalam kode tersebut ada pada skop dan aksesibilitas konstannya. Dalam class Constants, final hanya membuat konstan CM_PER_INCH tidak bisa diubah dan hanya dapat diakses di dalam metode main tempat ia dideklarasikan. Sebaliknya, dalam kelas Constants2, public static final menjadikan CM_PER_INCH dapat diakses secara global dari seluruh kelas dan bahkan dari kelas lain, Oleh karena itu, public static final memberikan jangkauan akses yang lebih luas dibandingkan final yang hanya bersifat lokal.

Kesimpulan :

Jadi, perbedaan utama antara penggunaan final double dan public static final double terletak pada jangkauan dan aksesibilitas konstan dalam kode. final double mendeklarasikan konstan yang hanya dapat diakses di dalam metode atau blok tempat konstan tersebut dideklarasikan, menjadikannya terbatas pada skop lokal. Sebaliknya, public static final double menyediakan konstan yang dapat diakses di seluruh kelas dan bahkan oleh kelas lain, berkat modifier public yang membuatnya dapat diakses luas, static yang memungkinkan akses langsung melalui nama kelas, dan final yang memastikan nilainya tetap.

Soal 3 : Operators

Kode program :

```
public class Floatingpoint {  
    public static void main(String[] args) {  
        double x = 92.98;  
        int nx = (int) Math.round(x);  
        System.out.print("variable x setelah dibualtkan : " + nx);  
    }  
}
```

```
}  
}
```

Output :

```
PS D:\wg\Kuliah\semester 3\PBO\praktek\pertemuan2> d;; cd 'd:\wg\Kuliah\semester  
ogram Files\Java\jdk-22\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInEx  
ldan Gumilang\AppData\Roaming\Code\User\workspaceStorage\ce2dd4b869e39e84054cffdbf  
n2_aa7c008b\bin' 'Floatingpoint'  
variable x setelah dibualtkn menggunakan math round : 93  
PS D:\wg\Kuliah\semester 3\PBO\praktek\pertemuan2> █
```

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);

Jawab :

Pada kode tersebut, variabel x bertipe double dengan nilai 92.98. Ketika Math.round(x) dipanggil, fungsi ini membulatkan nilai x ke bilangan bulat terdekat. Dalam kasus ini, 92.98 dibulatkan menjadi 93 karena lebih dekat ke 93 daripada 92. Hasil dari Math.round(x) adalah 93, yang kemudian di-cast menjadi int dan disimpan dalam variabel nx. Oleh karena itu, nilai nx setelah pembulatan adalah 93.

2. Kenapa dibutuhkan cast (int) dalam penggunaan Math.round(x) ?

Jawab :

Fungsi Math.round(x) mengembalikan nilai bertipe long jika x bertipe double, karena long memiliki jangkauan yang lebih luas untuk menampung hasil pembulatan angka desimal. Dalam kode tersebut, variabel nx dideklarasikan sebagai int, sehingga tipe long yang dihasilkan oleh Math.round(x) perlu diubah menjadi int agar sesuai dengan deklarasi nx. Casting (int) digunakan untuk melakukan konversi eksplisit dari long ke int, karena tanpa casting ini, akan terjadi kesalahan kompilasi (incompatible types) karena long tidak dapat langsung disimpan dalam int tanpa konversi. Casting memastikan hasil pembulatan sesuai dengan tipe data variabel yang dituju.

Kesimpulan :

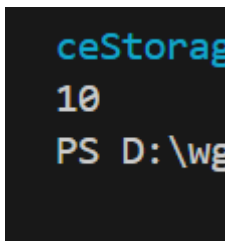
penggunaan Math.round(x) pada kode tersebut membulatkan nilai x ke bilangan bulat terdekat, menghasilkan tipe long jika inputnya bertipe double. Karena variabel nx bertipe int, diperlukan casting (int) untuk mengonversi hasil pembulatan dari long ke int, agar sesuai dengan tipe variabel yang digunakan. Tanpa casting ini, program akan mengalami kesalahan kompilasi karena ketidakcocokan tipe data. Oleh karena itu, casting (int) menjadi langkah penting untuk memastikan hasil pembulatan dapat disimpan dengan benar dalam variabel nx.

Soal 4 : Operators (1)

Kode program :

```
public 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);  
    }  
}
```

Output :



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.

Jawab :

Dari tipe data double **10.25**, di konversi menjadi float, menambahkan karakter f dibelakangnya sehingga menjadi **10.25f**, lalu di konversi menjadi long, yang menghilangkan bagian desimal dan hasilnya adalah **10**. Long tersebut dikonversi menjadi int, dan lalu dikonversi lagi menjadi short, sehingga hasilnya adalah tetap **10**. Dan dari tipe data int tersebut, dikonversi menjadi tipe data byte, sehingga output yang tertampil adalah **10**. Karena dalam kasus ini **10** masih dalam range tipe data byte, jadi nilainya tidak berubah.

2. Jelaskan apa yang berubah dari variable d menjadi variable b setelah dilakukan cast ?

Jawab :

Setelah dilakukan konversi tipe data, nilai variabel `d` yang awalnya berupa `double` dengan nilai `10.25` mengalami beberapa perubahan. Pertama, `d` dikonversi menjadi `float` dengan nilai `10.25f`, mengurangi presisi namun tetap mendekati nilai asli. Kemudian, `float` tersebut di-cast menjadi `long`, yang membulatkan nilai `10.25f` ke bilangan bulat terdekat, yaitu `10`. Selanjutnya, nilai `long` `10` diubah menjadi `int` dan kemudian `short`, yang tidak mengubah nilai karena `10` berada dalam jangkauan tipe-tipe tersebut. Terakhir, nilai `short` `10` di-cast menjadi `byte`, yang juga tetap `10` karena nilai tersebut berada dalam jangkauan `byte`.

Kesimpulan :

Perubahan dari `d` ke `b` melakukan beberapa konversi tipe data yang menghilangkan bagian desimal dari nilai awal. Meskipun nilai desimal hilang, hasil akhirnya tetap `10` setelah semua konversi. Ini menunjukkan bahwa meski presisi berkurang, nilai akhir dalam tipe data `byte` tetap stabil selama nilai awal masih dalam jangkauan tipe data tersebut.

Soal 5 : Strings**Kode program :**

```
import java.util.Scanner;

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

        System.out.print("Masukkan Kata pertama : ");
        String kataPertama = scanner.nextLine();

        System.out.print("Masukkan Kata kedua : ");
        String kataKedua = scanner.nextLine();

        int panjangkata = kataPertama.length() + kataKedua.length();

        System.out.println(panjangkata);

        if (kataPertama.compareTo(kataKedua) > 0) {
```



```

        System.out.println("Yes");
    } else {
        System.out.println("No");
    }

    String hasilKataPertama = kataPertama.substring(0, 1).toUpperCase()
+ kataPertama.substring(1);

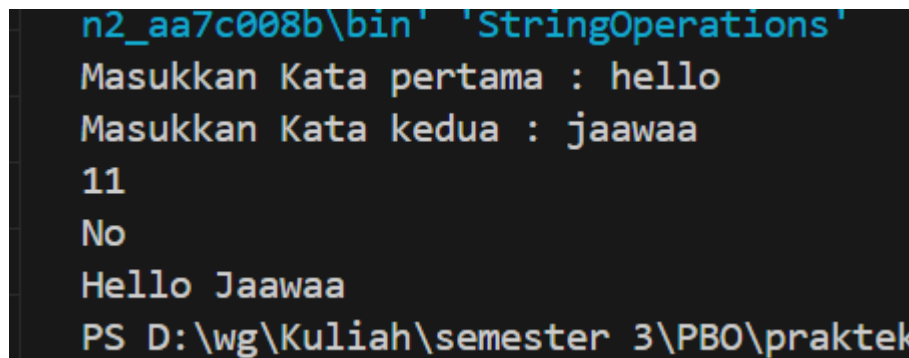
    String hasilkataKedua = kataKedua.substring(0, 1).toUpperCase() +
kataKedua.substring(1);

    System.out.println(hasilKataPertama + " " + hasilkataKedua);

    scanner.close();
}
}

```

Output :



```

n2_aa7c008b\bin\' 'StringOperations\'
Masukkan Kata pertama : hello
Masukkan Kata kedua : jaawaa
11
No
Hello Jaawaa
PS D:\wg\Kuliah\semester 3\PBO\praktek

```

Kesimpulan :

Pada kode tersebut, pengguna memasukkan kata pertama dan juga kata kedua, lalu program akan menghitung panjang dari kedua kata tersebut menggunakan fungsi `.length()`. Fungsi `compareTo()` dalam program ini digunakan untuk membandingkan kata pertama dengan kata kedua menurut alfabet. Misalnya, h dan j. h dalam alfabet adalah urutan ke-8, dan j adalah urutan ke-10, oleh karena itu h tidak lebih besar dari j. Maka output yang keluar dalam kasus ini adalah "No". Untuk mengubah huruf pertama dari kata pertama dan kata kedua menjadi huruf kapital, digunakan fungsi `.substring(0, 1).toUpperCase()`, fungsi tersebut akan mengubah huruf pertama dari sebuah kata menjadi huruf kapital. Jadi, dalam program ini digunakan

beberapa fungsi yang berguna untuk memanipulasi tipe data string dalam bahasa pemrograman java.

Link GitHub :

<https://github.com/WildanGumilang/PBO-praktek>