NAMA: FAHRI RIZQON ARSIANSYAH

NRP: 233040062

KELAS: D

MATA KULIAH: PEMROGRAMAN BERORIENTASI OBJEK (PBO)

Referensi Objek >1 variabel

```java
package Tugas2;

public class RefObjek {
    Run | Debug
    public static void main(String[] args) {
        Lingkaran l1 = new Lingkaran(jari2:5);
        Lingkaran l2 = l1;  // l2 mereferensi objek yang sama dengan l1

        System.out.println(l1.getJari2()); // Output: 5
        System.out.println(l2.getJari2()); // Output: 5

        l2.setJari2(jari2:10);

        System.out.println(l1.getJari2()); // Output: 10
        System.out.println(l2.getJari2()); // Output: 10
    }
}
```

Output:

```
5.0
5.0
10.0
10.0
PS C:\PBO>
```

Penjelasan:

dalam program ini, l1 dan l2 sebenarnya mengacu pada objek yang sama di memori, sehingga perubahan yang dilakukan melalui salah satu referensi akan mempengaruhi keduanya. Ketika objek Lingkaran pertama kali dibuat dengan jari-jari 5 dan direferensikan oleh l1, lalu l2 dibuat dengan menyalin referensi l1, keduanya tetap menunjuk ke objek yang sama. Oleh karena itu, saat metode ubahJari2() dipanggil dengan l2 dan mengubah nilai jari-jari menjadi 4, perubahan ini juga tercermin ketika nilai jari-jari diakses melalui l1. Ini menunjukkan konsep bahwa dalam Java, objek yang dilewatkan ke metode sebenarnya adalah referensi ke objek asli, bukan salinan baru, sehingga setiap perubahan akan langsung berdampak pada objek yang sama di memori.

Referensi Objek

```java
package Tugas2;

public class RefObjek {
    Run | Debug
    public static void main(String[] args) {
        Lingkaran l1 = new Lingkaran(jari2:5);
        Lingkaran l2 = l1;  // l2 mereferensi objek yang sama dengan l1

        System.out.println(l1.getJari2()); // Output: 5
        System.out.println(l2.getJari2()); // Output: 5

        l2.setJari2(jari2:10);

        System.out.println(l1.getJari2()); // Output: 10
        System.out.println(l2.getJari2()); // Output: 10
    }
}
```

Output:

```
5.0
5.0
10.0
10.0
PS C:\PBO>
```

Penjelasan:

Kode ini menunjukkan bagaimana referensi objek bekerja dalam Java. Ketika objek Lingkaran pertama kali dibuat dengan jari-jari 5 dan direferensikan oleh l1, kemudian l2 dibuat dengan menyalin referensi dari l1, keduanya sebenarnya menunjuk ke objek yang sama di memori. Oleh karena itu, ketika l2.setJari2(10) dipanggil, perubahan ini mempengaruhi objek yang sama, sehingga nilai jari-jari yang ditampilkan melalui l1.getJari2() dan l2.getJari2() sama-sama menjadi 10. Ini membuktikan bahwa dalam Java, objek yang dibuat di heap dapat memiliki banyak referensi, dan perubahan melalui salah satu referensi akan mempengaruhi semua referensi lain yang menunjuk ke objek tersebut.

Referensi Objek 3 variabel

```java
package Tugas2;

public class RefObjek3 {
        Run | Debug
        public static void main(String[] args) {
            Lingkaran l1 = new Lingkaran(jari2:5);
            Lingkaran l2 = l1;
            Lingkaran l3 = new Lingkaran(jari2:7);

            System.out.println(l1.getJari2()); // Output: 5
            System.out.println(l2.getJari2()); // Output: 5
            System.out.println(l3.getJari2()); // Output: 7

            l2 = l3; // l2 sekarang mereferensi objek baru

            System.out.println(l1.getJari2()); // Output: 5
            System.out.println(l2.getJari2()); // Output: 7
            System.out.println(l3.getJari2()); // Output: 7
        }
    }
```

Output:

```
5.0
5.0
7.0
5.0
7.0
7.0
PS C:\PBO>
```

Penjelasan:

Kode ini menunjukkan bahwa dalam Java, referensi objek bisa dialihkan tanpa mengubah objek aslinya. Awalnya, l1 dan l2 menunjuk ke objek yang sama dengan jari-jari 5, sedangkan l3 menunjuk ke objek lain dengan jari-jari 7. Setelah l2 = l3;, l2 tidak lagi menunjuk ke objek yang sama dengan l1, tetapi ke objek l3. Akibatnya, perubahan referensi ini membuat l2 dan l3 memiliki nilai jari-jari yang sama (7), sementara l1 tetap memiliki jari-jari 5.

Referensi Objek = NULL

```java
package Tugas2;

public class RefObjekNull {
    Run | Debug
    public static void main(String[] args) {
        Lingkaran l1 = new Lingkaran(jari2:5);
        Lingkaran l2 = l1;
        Lingkaran l3 = new Lingkaran(jari2:7);

        System.out.println(l1.getJari2());
        System.out.println(l2.getJari2());
        System.out.println(l3.getJari2());

        l2 = null; // l2 tidak lagi mereferensi objek apapun

        System.out.println(l1.getJari2()); // Masih bisa dijalankan
        // System.out.println(l2.getJari2()); // Akan menyebabkan error NullPointerException
    }
}
```

Output:

```
5.0
5.0
7.0
5.0
PS C:\PBO>
```

Penjelasan:

Kode ini menunjukkan bagaimana referensi objek bekerja di Java, terutama ketika sebuah referensi diubah menjadi null. Awalnya, l1 dibuat sebagai objek Lingkaran dengan jari-jari 5, dan l2 diberikan referensi yang sama dengan l1, sehingga keduanya menunjuk ke objek yang sama. Kemudian, dibuat objek Lingkaran lain dengan jari-jari 7 yang direferensikan oleh l3. Setelah mencetak nilai jari-jari masing-masing objek, pernyataan l2 = null; membuat l2 tidak lagi menunjuk ke objek mana pun. Namun, l1 masih memiliki referensi ke objek awalnya, sehingga System.out.println(l1.getJari2()); tetap bisa dijalankan. Jika System.out.println(l2.getJari2()); dieksekusi setelah l2 diubah menjadi null, akan terjadi NullPointerException karena l2 tidak lagi memiliki objek yang bisa diakses.

Latihan

```java
package Tugas2;

public class StudentMain {
    Run | Debug
    public static void main(String[] args) {
        Student x = new Student();
        Student y = x;

        x.setNrp(nrp:"01");
        y.setNrp(nrp:"02");

        System.out.println("x.getNrp(): " + x.getNrp()); // Output: 02

        Student z = new Student();
        z.setNrp(nrp:"03");

        x = z; // Sekarang x menunjuk ke z, bukan y

        System.out.println("x.getNrp(): " + x.getNrp()); // Output: 03
        System.out.println("y.getNrp(): " + y.getNrp()); // Output: 02
    }
}
```

Output:

```
x.getNrp(): 02
x.getNrp(): 03
y.getNrp(): 02
PS C:\PBO>
```

Penjelasan:

Kode ini menunjukkan bagaimana referensi objek bekerja dalam Java, khususnya saat mereferensikan objek yang sama dan mengubah referensinya.

Pada awalnya, objek Student dibuat dan direferensikan oleh x. Kemudian, y diberikan referensi yang sama dengan x, sehingga keduanya menunjuk ke objek yang sama di memori. Saat x.setNrp("01") dipanggil, nilai nrp pada objek tersebut menjadi "01", tetapi ketika y.setNrp("02") dipanggil, nilai nrp berubah menjadi "02" karena x dan y masih menunjuk ke objek yang sama. Akibatnya, saat mencetak x.getNrp(), hasilnya adalah "02".

Setelah itu, objek baru z dibuat dengan nrp "03". Ketika x = z; dieksekusi, x sekarang menunjuk ke objek baru z, sementara y masih menunjuk ke objek lama. Oleh karena itu, x.getNrp() mencetak "03", sedangkan y.getNrp() tetap mencetak "02" karena masih mereferensikan objek sebelumnya. Ini menunjukkan bahwa mengubah referensi suatu variabel tidak mengubah objek sebelumnya, melainkan hanya mengalihkan referensinya ke objek lain.