MAKALAH, PERCOBAAN, LATIHAN, DAN TUGAS

MODUL PRAKTIKUM 6

Disusun sebagai salah satu tugas

mata kuliah PBO I



Patricia Joanne

140810160065

Dikumpulkan tanggal

3 Oktober 2017

PROGRAM STUDI S-1 TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

2017

**Tugas Pendahuluan**

1. Apakah yang dimaksud dengan overloading?

2. Apakah yang dimaksud dengan overriding?

3. Jelaskan aturan tentang overriden!

4. Apakah yang dimaksud dengan polimorfisme?

5. Jelaskan proses terjadinya Virtual Method Invocation!

6. Apakah yang dimaksud dengan polymorphic arguments?

7. Apakah kegunaan kata kunci instanceof?

8. Apa perbedaan antara Abstract dan Interface?

1. Overloading adalah suatu keadaan dimana beberapa method sekaligus dapat mempunyai nama yang sama, akan tetapi mempunyai fungsionalitas yang berbeda. Contoh penggunaan overloading dilihat di bawah ini:

Gambar(Titik t1) 🡪 1 parameter titik, untuk menggambar titik

Gambar(Titik t1, Titik t2) 🡪 2 parameter titik, untuk menggambar garis

Gambar(Titik t1,Titik t2,Titik t3) 🡪 3 parameter titik, untuk menggambar segitiga

Gambar(Titik t1,Titik t2,Titik t3,Titik t4) 🡪 4 parameter titik, untuk menggambar persegi empat

Overloading ini dapat terjadi pada class yang sama atau pada suatu parent class dan subclass-nya. Overloading mempunyai ciri-ciri sebagai berikut:

* Nama method harus sama
* Daftar parameter harus berbeda
* Return type boleh sama, juga boleh berbeda

2. Overriding adalah suatu keadaan dimana method pada subclass menolak method pada parent class-nya. Overriding mempunyai ciri-ciri sebagai berikut:

* Nama method harus sama
* Daftar parameter harus sama
* Return type harus sama

3. Method yang terkena override (overridden method) diharuskan tidak boleh mempunyai modifier yang lebih luas aksesnya dari method yang meng-override (overriding method).

4. Polimorfisme adalah kemampuan untuk mempunyai beberapa bentuk class yang berbeda. Polimorfisme ini terjadi pada saat suatu obyek bertipe parent class, akan tetapi pemanggilan constructornya melalui subclass.

5. Virtual Method Invocation (VMI) bisa terjadi jika terjadi polimorfisme dan overriding. Pada saat obyek yang sudah dibuat tersebut memanggil overridden method pada parent class, kompiler Java akan melakukan invocation (pemanggilan) terhadap overriding method pada subclass, dimana yang seharusnya dipanggil adalah overridden method.

6. Polymorphic arguments adalah tipe suatu parameter yang menerima suatu nilai yang bertipe subclass-nya.

7. Pernyataan instanceof sangat berguna untuk mengetahui tipe asal dari suatu polymorphic arguments. Seringkali pemakaian instanceof diikuti dengan casting object dari tipe parameter ke tipe asal.

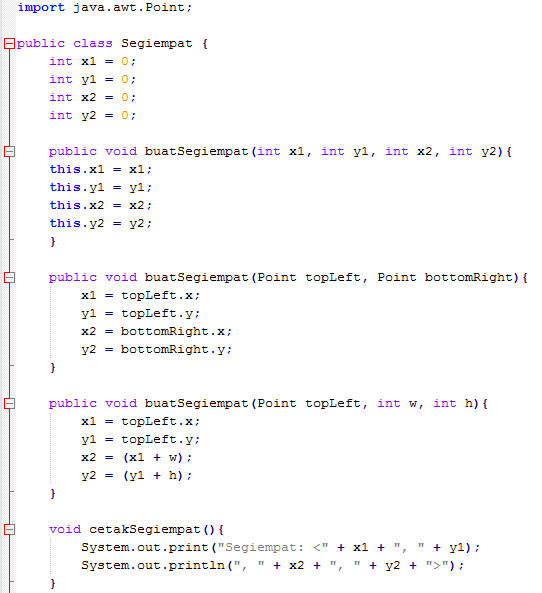
8. Perbedaan Abstract dan Interface:

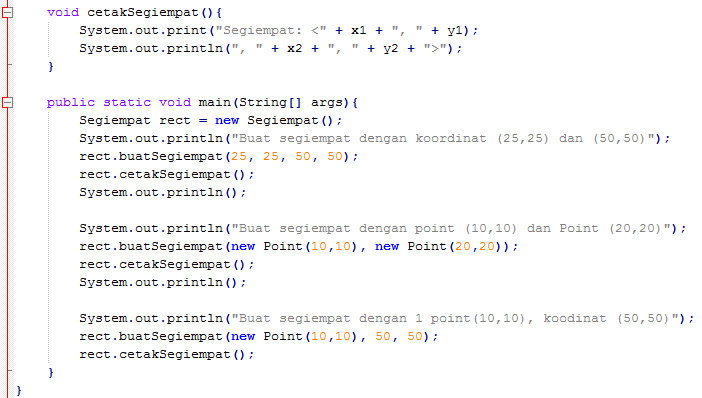
|  |  |  |
| --- | --- | --- |
| Perbedaan | Abstract | Interface |
| Attribute | Tipe data apa saja | Hanya berupa konstanta |
| Method | Boleh deklarasi, boleh method lengkap | Berupa deklarasi |
| Syntax | Sebagian abstract | Seluruhnya abstract |

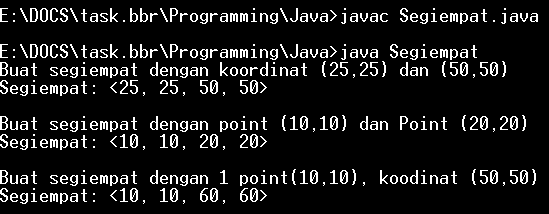
**Bab 6**

**Overloading, Overriding, dan Polimorfisme**

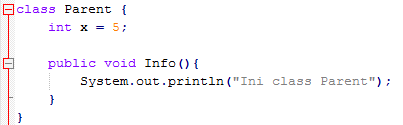
Percobaan 1: Melakukan overloading pada method

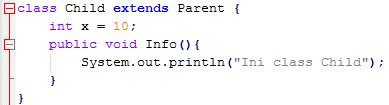


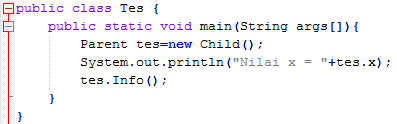


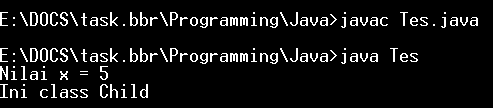


Percobaan 2: Memahami proses terjadinya Virtual Method Invocation



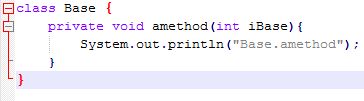


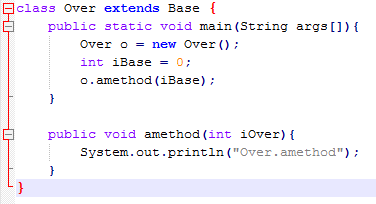


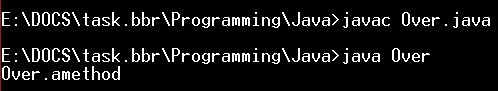


Latihan 1: Overriding

Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan!

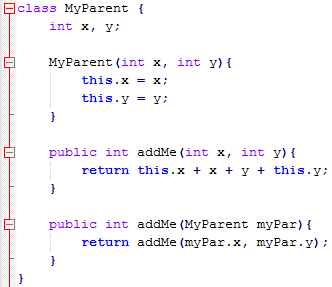


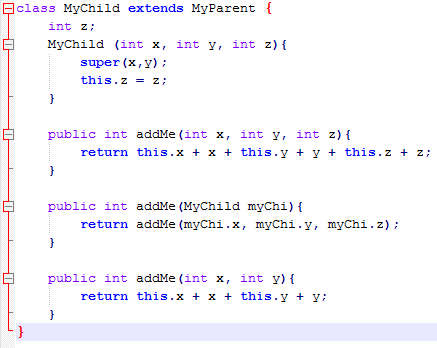


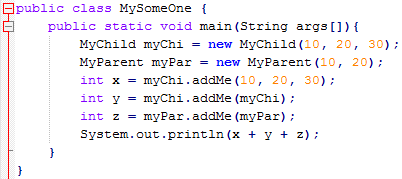


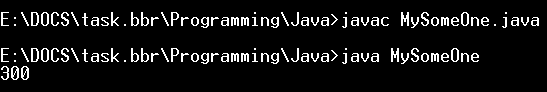
Latihan 2: Overloading

Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan!



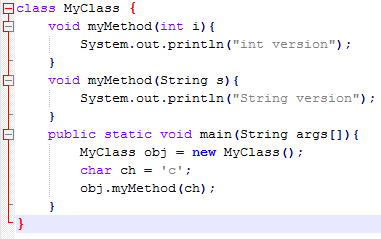


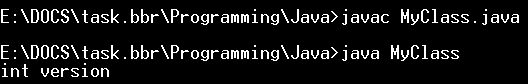




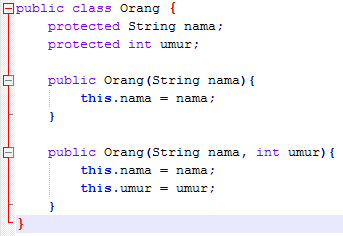
Latihan 3: Overloading

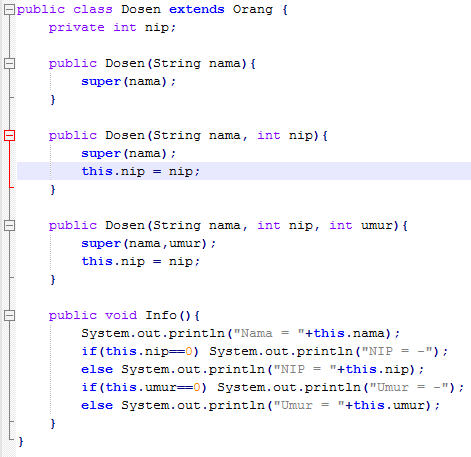
Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan!

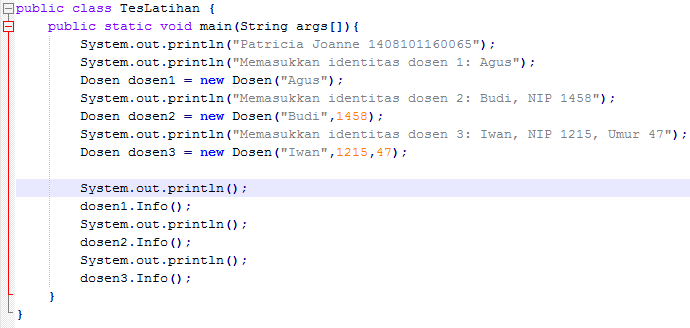


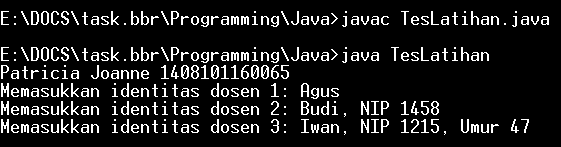


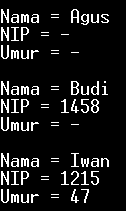
Latihan 4: Mengimplementasikan UML class diagram dalam program



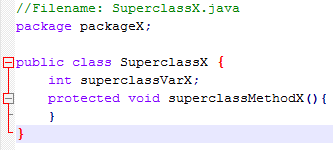


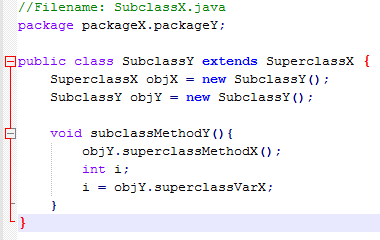


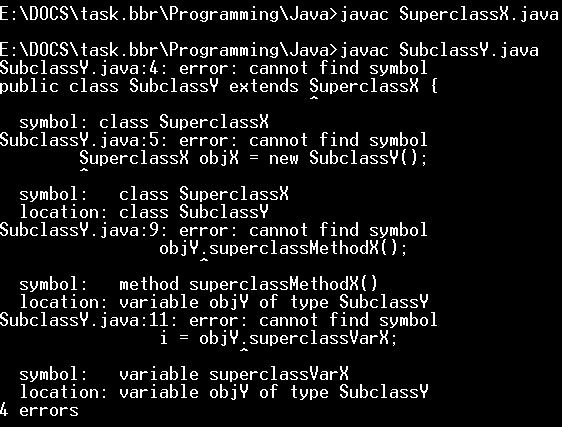




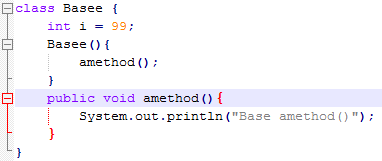
Latihan 5. Apa yang terjadi bila kode di bawah ini dikompile dan dijalankan jika sebelumnya Base.java belum dikompile? Jelaskan!

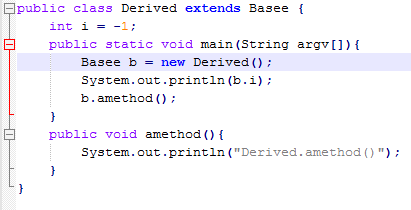


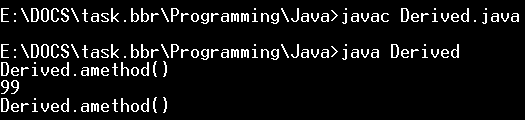




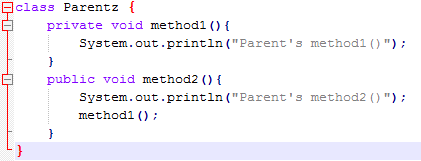
Latihan 6. Apa yang tampil di layar jika kode di bawah ini dijalankan? Jelaskan!

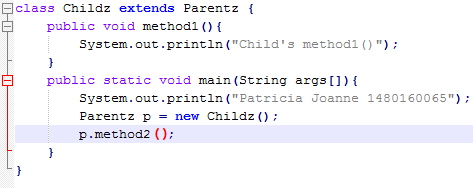






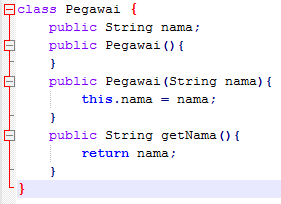
Latihan 7: Apa yang tampil di layar jika kode dibawah ini dijalankan? Jelaskan!

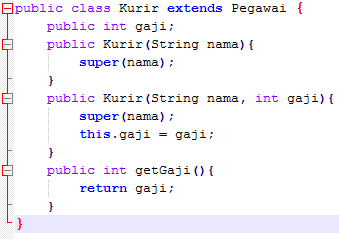


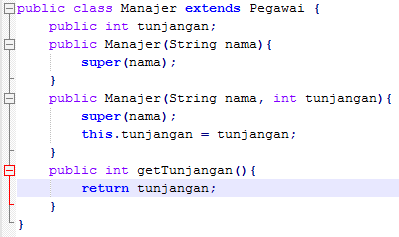


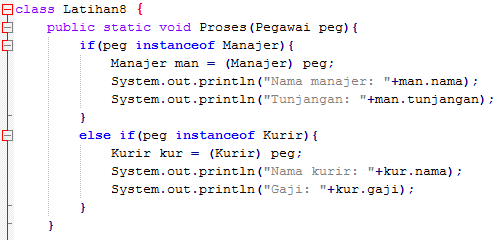


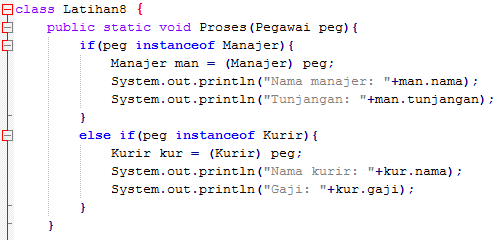
Latihan 8: Mengimplementasikan UML class diagram dalam program

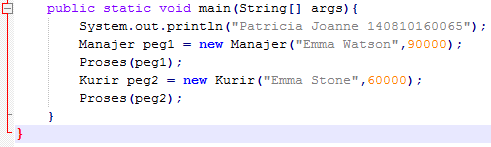


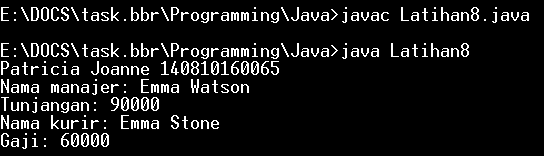




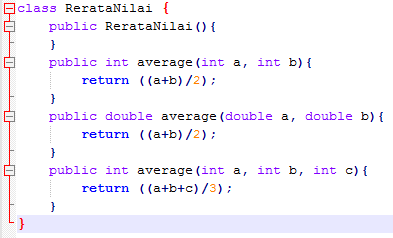


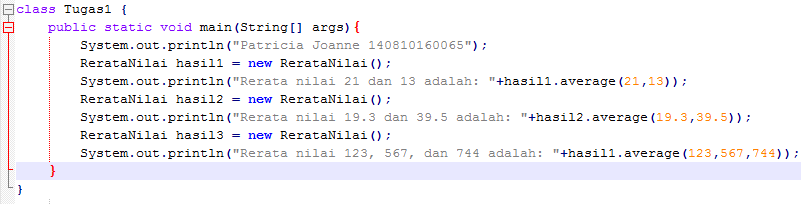


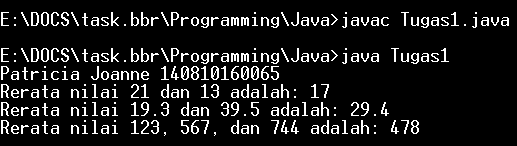




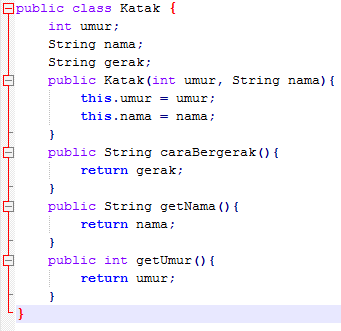
Tugas 1: Mengimplementasikan UML class diagram dalam program

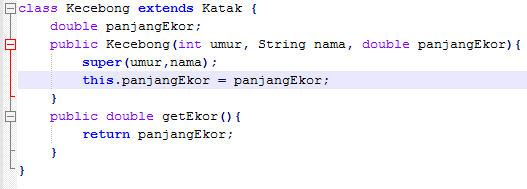


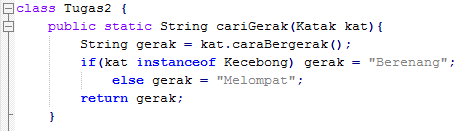


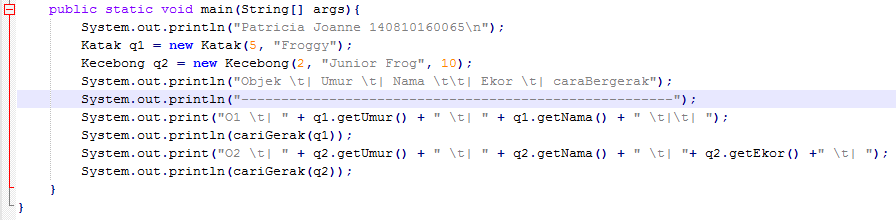


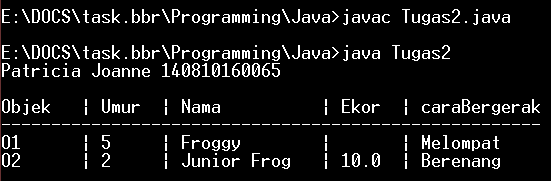
Tugas 2: Mengimplementasikan UML class diagram dalam program











Tugas 3: Mengimplementasikan UML class Diagram dalam program

