19

LAPORAN PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

PERTEMUAN KE-11

****

**Disusun oleh :**

**NAMA : FERDI DIRGANTARA**

**NIM : 175410039**

**JURUSAN : TEKNIK INFORMATIKA**

**JENJANG : S1**

LABORATORIUM TERPADU

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTERAKAKOMYOGYAKARTA

2019

1. **TEORI SINGKAT**

Polimorfisme adalah kemampuan sebuah variabel reference untuk merubah behavior sesuai dengan apa yang dipunyai object.

Polimorfisme membuat objek-objek yang berasal dari subclass yang berbeda, diperlakukan sebagai objek-objek dari satu superclass. Hal ini terjadi ketika memilih method yang sesuai untuk diimplementasikan ke objek tertentu berdasarkan pada subclass yang memiliki method bersangkutan.

Polimorfisme dapat diterjemahkan pula sebagai “sebuah method yang sama namanya (homonim) tetapi mempunyai tingkah laku yang berbeda”. Komputer membedakan method berdasarkan signature method (saat compile) atau berdasarkan reference object (saat runtime).

Ada dua bentuk polimorfisme

1. Overloading ( compile-time (early binding) polimorphisme )

Overloading adalah salah satu cara penerapan dalam konsep polimorfisme. Overload merupakan pendefinisian ulang suatu metode dalam class yang sama. Syarat overload yaitu metode dan tipe parameter harus berbeda dalam kelass yang sama.

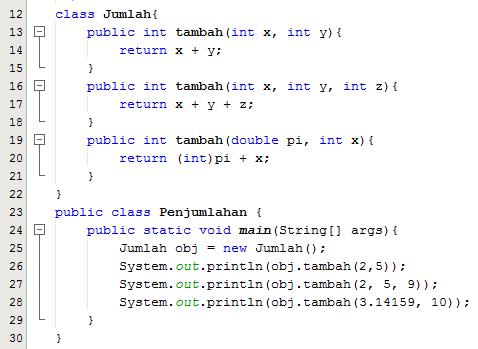
1. Overriding ( runtime (late binding) polimorphisme )

Override merupakan pendefinisian ulang suatu metode oleh subclass. Syarat Override yaitu metode, return type, dan parameter harus sama. Jika tidak sama maka bukan dianggap sebagai override tetapi metode yang baru pada subclass.

Dikatakan **late-binding (run time) polymorphism** karena saat compile  
komputer tidak tahu method mana yang harus dipanggil, dan baru akan  
diketahui saat run-time. Run-time polymorphism dapat diterapkan melalui  
**overridden method.** Run time polymorphism juga disebut dengan istilah  
**dynamic binding.**

1. **PEMBAHASAN LISTING**

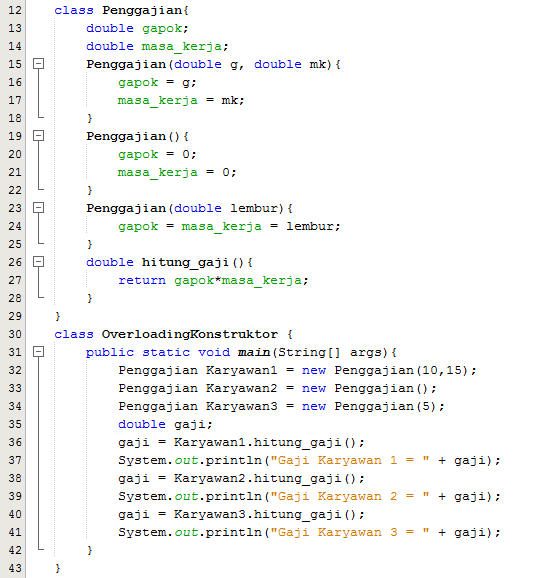
Praktik 1



Polymorphism ditunjukkan dengan menggunakan method tambah. Bentuk polymorphism diatas dinamakan Overloading (early-binding (atau compile-time) polymorphism) karena komputer mengetahui mana method tambah yang dipanggil setelah mengcompile byte code. Jadi setelah proses compile ketika code sudah menjadi byte code, komputer akan mengetahui method mana yang akan dieksekusi.

Setiap parameter yang dimiliki method tambah memiliki tipe atau jumlah yang berbeda sehingga hal tersebut memenuhi syarat untuk melakukan overloading dan dibuktikan dengan pemanggilan setiap method melalui objek **obj** dengan argumen yang berbeda-beda.

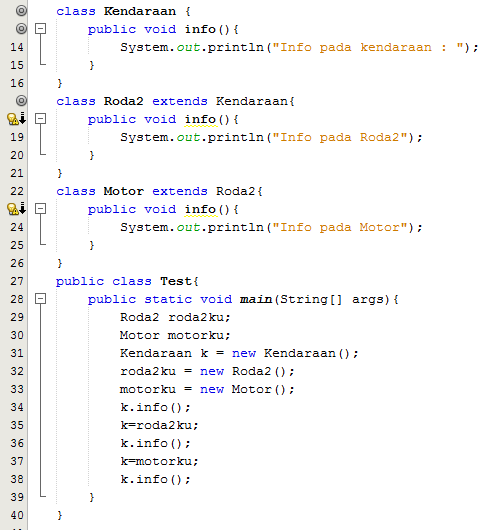
Praktik 2



Jika pada praktik sebelumnya dilakukan polymorphism (overloading) pada suatu method yaitu method **tambah**, maka pada praktik diatas dilakukan overloading pada konstruktor kelas yaitu konstruktor kelas **Penggajian**. Mirip seperti overloading method yang memiliki tipe atau jumlah parameter yang berbeda untuk masing-masing konstruktor hanya saja untuk overloading konstruktor tidak dapat menerima nilai balik (*return value*).

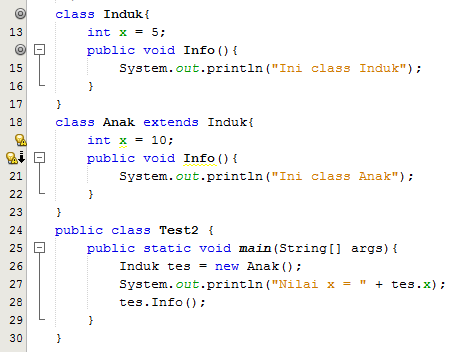
Kelas Penggajian kemudian dibuat menjadi 3 objek dengan argumen yang berbeda-beda pada saat instansiasi objek seperti yang dituliskan pada baris 32 sampai baris 34. Nilai argumen masing-masing konstrukter kemudian digunakan sebagai nilai untuk menghitung gaji karyawan dengan menggunakan method **hitung\_gaji**.

Praktik 3



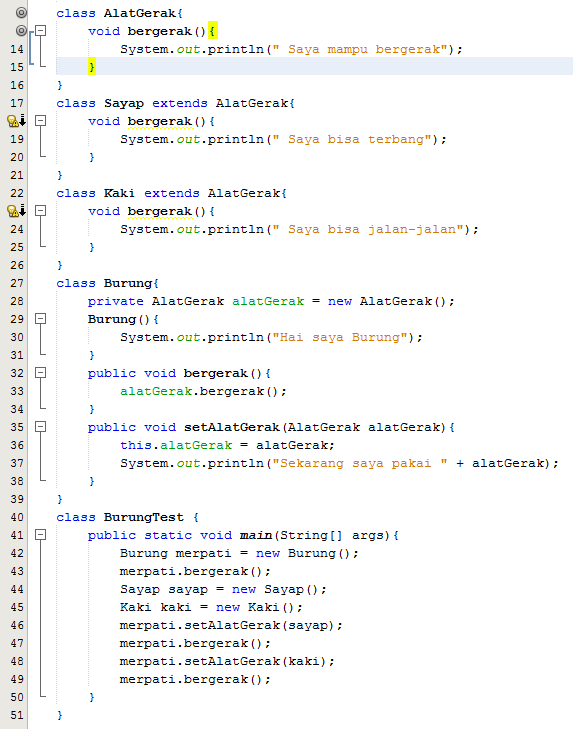
Ketika program diatas dijalankan akan menampilkan pesan yang terdapat pada method **info**. Pada saat method **info** dipanggil melalui objek kelas **Kendaraan** maka akan muncul pesan “**Info pada kendaraan :”**, pada saat method **info** dipanggil melalui objek kelas **Roda2** maka akan muncul pesan **“Info pada Roda2”**, dan begitu pula pada kelas **Motor** akan manampilkan pesan **“Info pada Motor”**. Hal tersebut disebut dengan **overriding polymorphism**.

Praktik 4



Ketika program di atas dijalankan akan terlihat bahwa ketika diakses atribut x dari objek tes, maka yang muncul adalah nilai x dari super kelas, bukan atribut x dari subkelas. Akan tetapi ketika diakses method Info() dari objek tes, yang muncul adalah method Info() dari sub kelas, bukan method Info() dari superkelas.

Praktik 5



Method setAlatGerak dapat menerima berbagai type yaitu Sayap dan Kaki. Hal ini legal karena pada kenyataannya Sayap dan Kaki adalah AlatGerak. Saat method bergerak() milik object trutle dipanggil, method yang dipanggil adalah sesuai dengan method yang dimiliki oleh type yang diberikan.

1. **PEMBAHASAN LATIHAN**
2. **TUGAS**
3. **KESIMPULAN**

Polimorfisme adalah kemampuan sebuah variabel reference untuk merubah behavior sesuai dengan apa yang dipunyai object. Hal tersebut memberikan kemampuan untuk melakukan modifikasi pada method, attribut, atau konstruktor pada superclass dapat dimodifikasi pada subclass sesuai dengan kebutuhan dari suatu kelas.

1. **LISTING**

*Terlampir*