Nama : Kelvianto Pratama Harum

NIM : 200210500016

MK : Pemrograman Lanjut

Dosen : Muhammad Fajar B, S.Pd., M.Cs.

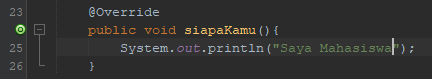
**PERTEMUAN V**

**POLYMORPHISM**

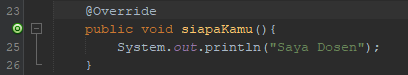
1. **Contoh 5.1**

Source code:

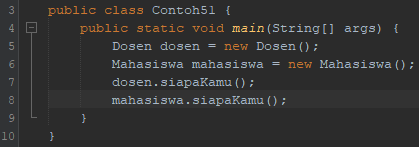
**Mahasiswa.java**



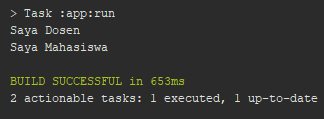
**Dosen.java**



**Contoh51.java**



Output:



Penjelasan:

Ini adalah modifikasi project pada Praktikum 4.1; Pertemuan IV – Enkapsulasi.

Pada class **Mahasiswa** dan **Dosen** ditambahkan sebuah method **siapaKamu()** yang menimpa method **siapaKamu()** pada class **Manusia**.

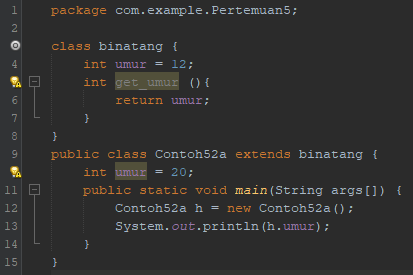
Keyword **@Override** menandakan method setelah keyword ini adalah **overriding** dari method yang ada pada **super** class.

Method ini di-**override** untuk memberikan output yang berbeda, pada **super** class **Manusia** output yang diberikan adalah console output **“Saya manusia”** sedangkan pada **subclass Mahasiswa** output yang diberikan adalah **“Saya Mahasiswa”** dan pada **subclass Dosen** output yang diberikan adalah **“Saya Dosen”**.

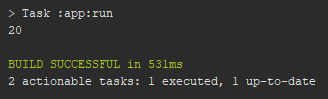
Hal ini dibuktikan pada saat pemanggilan method **siapaKamu()** pada object **dosen** dan **mahasiswa** pada Main class (**Contoh51**), terlihat pada output console yang diberikan sesuai dengan **overriding** yang telah dilakukan.

1. **Contoh 5.2a**

Source code:



Output:



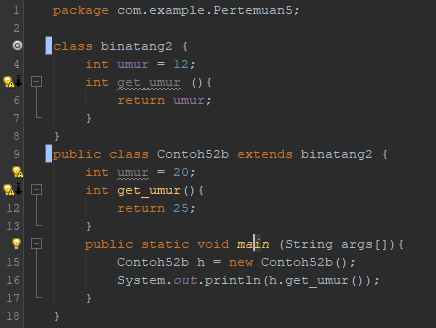
Penjelasan:

Pada contoh ini dibuat sebuah class bernama **binatang** yang memiliki **field** dengan nama **umur** tipe data **integer** dengan nilai **12**, dan sebuah method dengan nama **get\_umur** yang pada intinya mengembalikan nilai yang ada pada **field umur**. Seluruh **member** ini menggunakan modifier **default**.

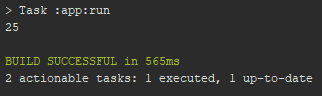
Kemudian dibuat sebuah main class **(Contoh52a)** yang diturunkan dari class **binatang**, pada class ini **field umur** dioverride dengan deklarasi ulang dengan nilai **20**. Lalu pada main method dibuat sebuah object dari class **Contoh52a** dengan nama **h**. Kemudian nilai **field umur** pada object **h** dioutput pada console, yang dapat dilihat bernilai **20**, sesuai dengan yang telah dideklarasikan. Pada dasarnya program akan mencari field/method dengan nama tersebut didalam class itu sendiri, lalu jika tidak ditemukan akan dipanggil dari **super** class object tersebut.

1. **Contoh 5.2b**

Source code:



Output:



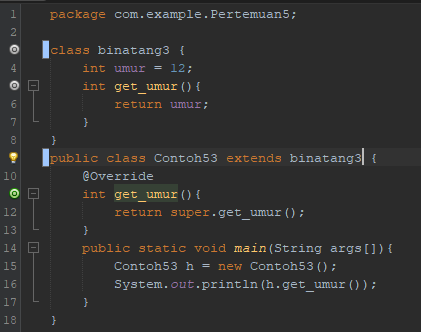
Penjelasan:

Pada contoh ini dibuat sebuah class bernama **binatang2** yang memiliki **field** dengan nama **umur** tipe data **integer** dengan nilai **12**, dan sebuah method dengan nama **get\_umur** yang pada intinya mengembalikan nilai yang ada pada **field umur**. Seluruh **member** ini menggunakan modifier **default**.

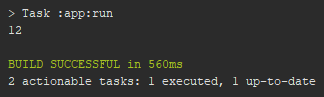
Kemudian dibuat sebuah main class **(Contoh52b)** yang diturunkan dari class **binatang2**, pada class ini **field umur** dioverride dengan deklarasi ulang dengan nilai **20**. Method **get\_umur** juga di-override dengan mengubah perilaku method menjadi mengembalikan nilai literal **25**. Lalu pada main method dibuat sebuah object dari class **Contoh52b** dengan nama **h**. Kemudian nilai balik method **get\_umur** pada object **h** dioutput pada console, yang dapat dilihat bernilai **25**, sesuai dengan yang telah dideklarasikan. Pada dasarnya program akan mencari field/method dengan nama tersebut didalam class itu sendiri, lalu jika tidak ditemukan akan dipanggil dari **super** class object tersebut.

1. **Contoh 5.3**

Source code:



Output:



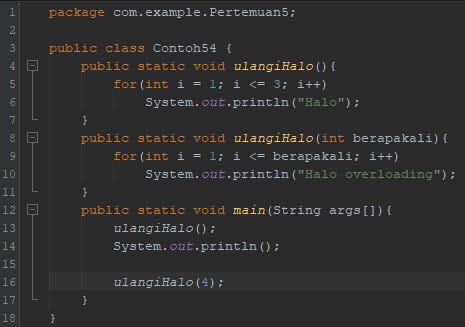
Penjelasan:

Pada contoh ini dibuat sebuah class bernama **binatang3** yang memiliki **field** dengan nama **umur** tipe data **integer** dengan nilai **12**, dan sebuah method dengan nama **get\_umur** yang pada intinya mengembalikan nilai yang ada pada **field umur**. Seluruh **member** ini menggunakan modifier **default**.

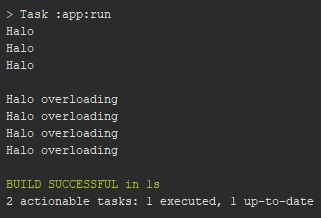
Kemudian dibuat sebuah main class **(Contoh53)** yang diturunkan dari class **binatang3**, pada class ini method **get\_umur** di-override dengan mengubah perilaku method menjadi mengembalikan nilai balik pemanggilan method **get\_umur** yang ada pada **super** class. Lalu pada main method dibuat sebuah object dari class **Contoh53** dengan nama **h**. Kemudian nilai balik method **get\_umur** pada object **h** dioutput pada console, yang dapat dilihat bernilai **12**, sesuai dengan yang telah dideklarasikan. Pada dasarnya program akan mencari field/method dengan nama tersebut didalam class itu sendiri, lalu jika tidak ditemukan akan dipanggil dari **super** class object tersebut.

1. **Contoh 5.4**

Source code:



Output:



Penjelasan:

Pada contoh ini dibuat sebuah class dengan nama **Contoh54** yang berisikan:

* Method **ulangiHalo** tanpa menggunakan parameter

Method ini akan memberikan output **“Halo”** pada console sebanyak 3 kali, dengan for-loop.

* Method **ulangiHalo** dengan parameter variabel **berapakali** tipe data **integer**

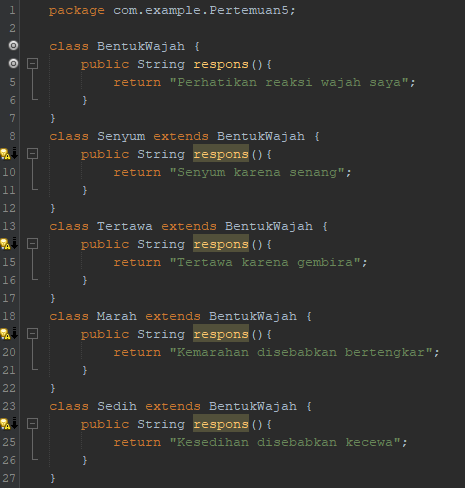
Method ini adalah **overloading** dari method dengan nama yang sama, dikatakan overload karena memiliki nama yang sama namun cara pemanggilan yang berbeda, pada method sebelumnya dipanggil tanpa parameter, sedangkan method ini dipanggil dengan sebuah argumen variable **integer**.

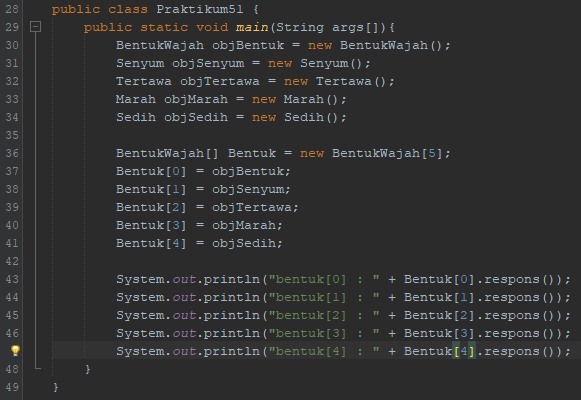
Method ini akan memberikan output **“Halo overloading”** sebanyak nilai **berapakali** saat pemanggilan method.

Kemudian pada method **main** dipanggil method **ulangiHalo()** tanpa parameter dan terlihat pada output ditampilkan string **“Halo”** sebanyak 3 kali. Lalu, dilakukan pemanggilan method **ulangiHalo** dengan argumen nilai **4**, terlihat pada output ditampilkan string **“Halo overloading”** sebanyak **4** kali sesuai nilai argumen. Intinya, program akan memilih method yang sesuai untuk dilakukan eksekusi sesuai dengan argumen yang diberikan.

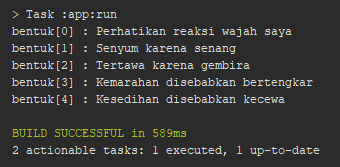
1. **Praktikum 5.1**

Source code:





Output:



Penjelasan:

Pada contoh ini dibuat sebuah base class yaitu **BentukWajah** dengan sebuah method **respons** yang menampilkan string **“Perhatikan reaksi wajah saya”** pada console.

Lalu dibuat empat (4) **subclass** turunan dari class **BentukWajah**, yaitu:

* Class **Senyum** yang meng-**override** method **respons** menjadi menampilkan string **“Senyum karena senang”** pada console.
* Class **Tertawa** yang meng-**override** method **respons** menjadi menampilkan string **“Tertawa karena gembira”** pada console.
* Class **Marah** yang meng-**override** method **respons** menjadi menampilkan string **“Kemarahan disebabkan bertengkar”** pada console.
* Class **Sedih** yang meng-**override** method **respons** menjadi menampilkan string **“Kesedihan disebabkan kecewa”** pada console.

Kemudian, pada method **main** masing-masing class dibuat menjadi object, yaitu:

* Class **BentukWajah** dengan nama object **objBentuk**.
* Class **Senyum** dengan nama object **objSenyum**.
* Class **Tertawa** dengan nama object **objTertawa**.
* Class **Marah** dengan nama object **objMarah**.
* Class **Sedih** dengan nama object **objSedih**.

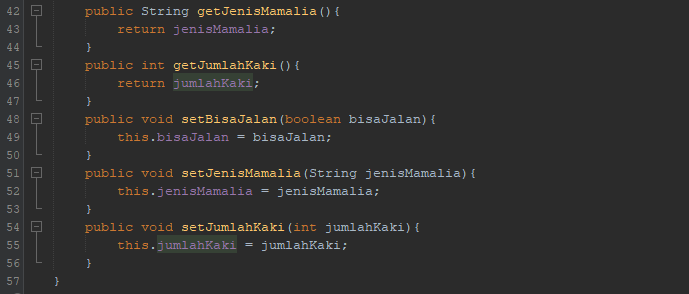
Lalu dibuat sebuah array dengan dari class **BentukWajah** dengan nama **Bentuk** dengan jumlah member **5 member**. Member tersebut adalah object yang telah dibuat sebelumnya, **objBentuk, objSenyum, objTertawa, objMarah, dan objSedih**.

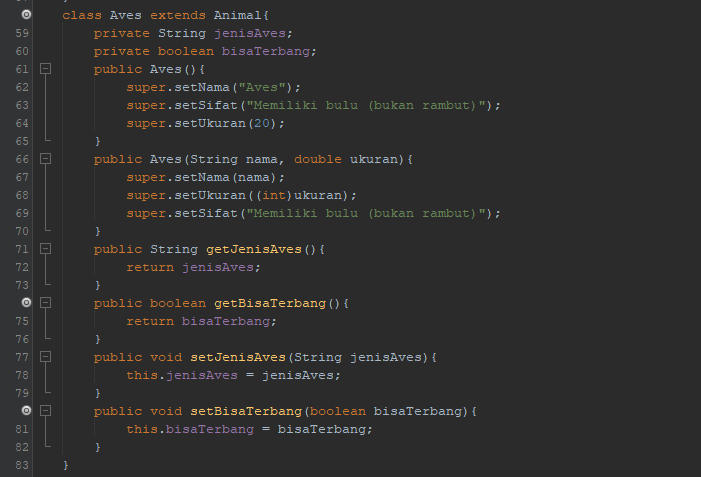
Yang kemudian dipanggil method **respons** pada masing-masing member array tersebut, dan dapat dilihat pada console **method respons** yang digunakan adalah method yang sama dideklarasikan pada classnya.

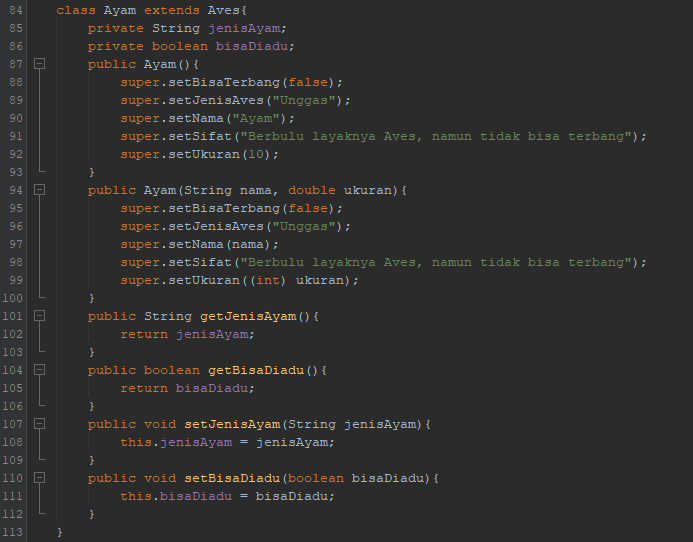
1. **Tantangan**

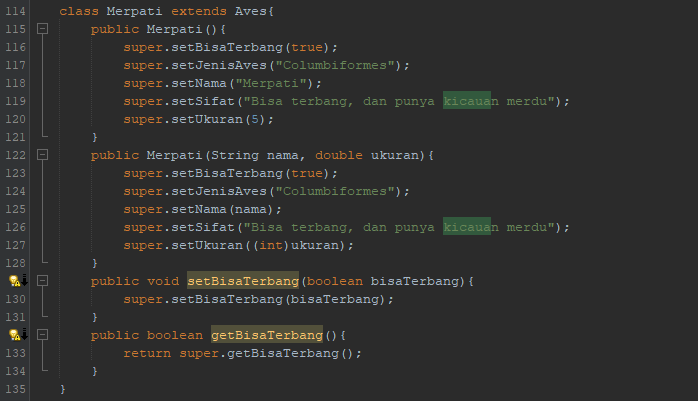
Source code:

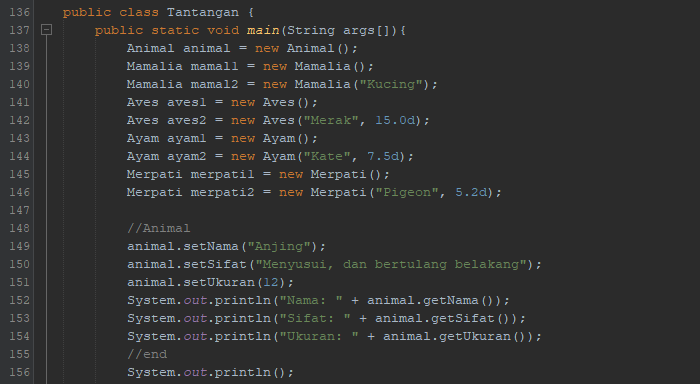




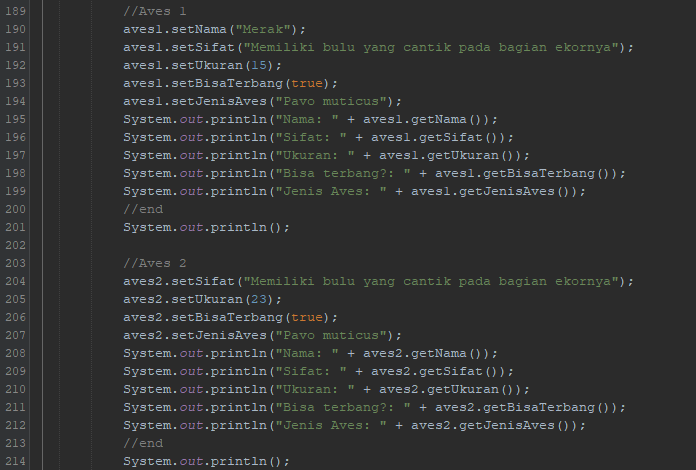








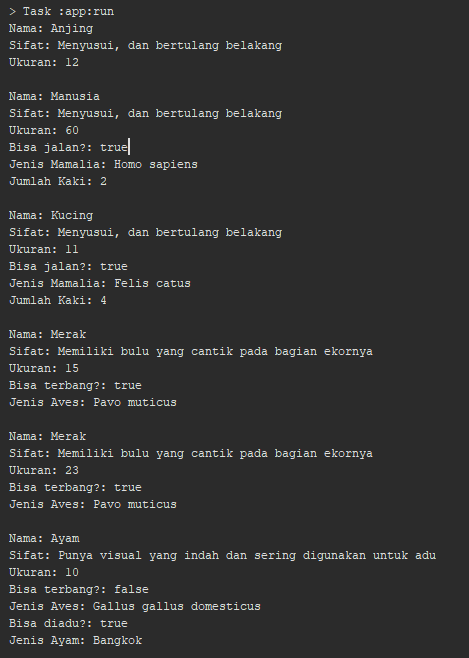


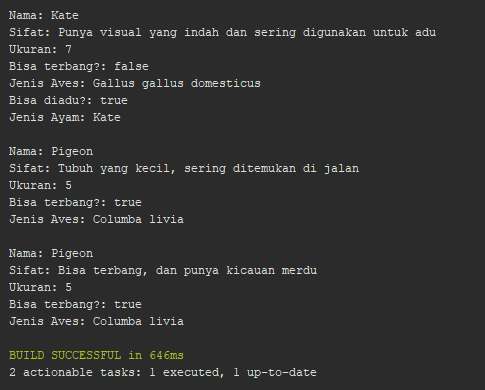






Output:





Penjelasan:

1. `package com.example.Pertemuan5;`: Mendefinisikan paket tempat kelas-kelas tersebut berada.

2. `class Animal {`: Mendeklarasikan kelas `Animal`.

3. `private String nama, sifat;`: Mendefinisikan dua atribut `String` privat: `nama` dan `sifat`.

4. `private int ukuran;`: Mendefinisikan satu atribut `int` privat: `ukuran`.

5-13. Metode getter dan setter untuk atribut `nama`, `sifat`, dan `ukuran`.

14-49. Kelas `Mamalia` yang merupakan subkelas dari `Animal`:

- `private String jenisMamalia;`: Atribut tambahan `jenisMamalia`.

- `private boolean bisaJalan;`: Atribut tambahan `bisaJalan`.

- `private int jumlahKaki;`: Atribut tambahan `jumlahKaki`.

- Konstruktor default yang mengatur nilai awal atribut `nama`, `sifat`, dan `ukuran` menggunakan metode setter dari kelas induk.

- Konstruktor yang menerima parameter `String nama`.

- Metode getter dan setter untuk atribut tambahan `jenisMamalia`, `bisaJalan`, dan `jumlahKaki`.

50-80. Kelas `Aves` yang merupakan subkelas dari `Animal`:

- `private String jenisAves;`: Atribut tambahan `jenisAves`.

- `private boolean bisaTerbang;`: Atribut tambahan `bisaTerbang`.

- Konstruktor default yang mengatur nilai awal atribut `nama`, `sifat`, dan `ukuran` menggunakan metode setter dari kelas induk.

- Konstruktor yang menerima parameter `String nama` dan `double ukuran`.

- Metode getter dan setter untuk atribut tambahan `jenisAves` dan `bisaTerbang`.

81-109. Kelas `Ayam` yang merupakan subkelas dari `Aves`:

- `private String jenisAyam;`: Atribut tambahan `jenisAyam`.

- `private boolean bisaDiadu;`: Atribut tambahan `bisaDiadu`.

- Konstruktor default yang mengatur nilai awal atribut `nama`, `sifat`, dan `ukuran` menggunakan metode setter dari kelas induk.

- Konstruktor yang menerima parameter `String nama` dan `double ukuran`.

- Metode getter dan setter untuk atribut tambahan `jenisAyam` dan `bisaDiadu`.

110-125. Kelas `Merpati` yang merupakan subkelas dari `Aves`:

- Konstruktor default yang mengatur nilai awal atribut `nama`, `sifat`, dan `ukuran` menggunakan metode setter dari kelas induk.

- Konstruktor yang menerima parameter `String nama` dan `double ukuran`.

- Metode getter dan setter untuk atribut tambahan `bisaTerbang`.

126-238. Kelas `Tantangan` dengan metode `main`:

- Membuat beberapa objek dari kelas `Animal`, `Mamalia`, `Aves`, `Ayam`, dan `Merpati`.

- Mengatur nilai atribut objek-objek tersebut menggunakan metode setter.

- Menampilkan nilai atribut objek-objek tersebut menggunakan metode getter

Metode `main` mengilustrasikan penggunaan kelas dan hierarki pewarisan dalam kode dengan membuat objek-objek dan memanipulasi atribut-atributnya, kemudian menampilkan hasilnya ke konsol.