William Antony

PPTI 6 / 2201827673

UTC OOP

1. 4 access modifier di Java Programming Language

* Public

Access modifier yang digunakan agar suatu metode atau atribut bisa diakses di luar package, class turunannya, package yang sama, dan juga class itu sendiri. Umumnya public digunakan pada method / function.

* Protected

Access modifier yang digunakan pada atribut di class yang akan diturunkan, sehingga pada class turunannya bisa mengakses atribut tersebut. Tidak hanya class turunan, protected juga bisa diakses dari package yang sama dan juga class itu sendiri.

* Package

Access modifier default, jika tidak diset access modifier nya, maka secara otomatis menggunakan package access modifier nya. Dimana dengan adanya access modifier ini, atribut atau fungsi bisa diakses atau dipanggil pada package yang sama dan juga class itu sendiri.

* Private

Access modifier pada atribut atau fungsi, yang hanya bisa diakses pada class itu sendiri, biasanya private digunakan untuk atribut, untuk melindungi data / enkapsulasi data. Sehingga dibutuhkan setter / getter untuk memanipulasi atribut tersebut.

1. Perbedaan overloading dan overriding

* Overloading

Terdapat beberapa fungsi yang memiliki nama sama dengan parameter yang berbeda. Berbeda yang dimaksud adalah jumlah parameter, tipe data parameter, dan juga urutan parameter. Sehingga fungsi dipanggil sesuai dengan parameter yang ada. Cara kerja overloading adalah static binding, sehingga dihandle saat compile yang juga merupakan sifat trivial polymorphism.

public class Tes {

//dua fungsi sama, namun dengan parameter berbeda

public int jumlah (int a, int b){

return a + b;

}

public double jumlah (double a, double b){

return a + b;

}

public Tes(){

System.out.println(jumlah(1, 2));

System.out.println(jumlah(1.2, 3.4));

}

public static void main(String args[]){

new Tes();

}

}

* Overriding

Terdapat beberapa fungsi yang sama dengan parameter yang sama pula, namun cara kerja dan hasil yang direturn berbeda sesuai dengan class masing-masing. Biasanya terjadi saat inheritance, dimana fungsi yang terdapat di parent, juga ada di childnya dengan hasil yang sesuai dengan class masing-masing. Cara kerjanya dynamic binding yang dihandle saat runtime sehingga terjadi pure polymorphism.

public class Kendaraan {

public void suara (){

}

}

public class Motor extends Kendaraan {

public void suara (){

System.out.println(“NGENGG”);

}

}

public class Mobil extends Kendaraan {

public void suara (){

System.out.println(“BRUMM”);

}

}

Di Main.java

public static void main(String args[]){

Mobil Toyota = new Mobil();

Motor Kawasaki = new Motor();

Toyota.suara();//BRUMM

Kawasaki.suara();//NGENGG

}

1. Add()

Menambahkan data pada arraylist

Size()

Menampilkan jumlah data pada arraylist

Get(index)

Menampilkan data pada index tertentu