1. 4 access modifiers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Same Class | Same Package | SubClass | Other Package |
| Private | Y | N | N | N |
| No access modifier  (private – package) | Y | Y | N | N |
| Protected | Y | Y | Y | N |
| Public | Y | Y | Y | Y |

1. Private

Private paling aman , karena dia hanya bisa diakses di kelas tersebut saja. Jika mau diakses kelas/package lain dia dapat digunakan setter getter. Sebaiknya untuk program yang baik, menggunakan access ini.

1. No Access Modifier

Meskipun access ini bisa di akses kelas lain yang package nya masih sama, namun access ini di private, sehingga apa yang dibuat tidak dapat diubah.

1. Protected

Protected itu dapat di access oleh subclass nya yang berada di package yang sama. Ini merupakan contohnya ialah inheritance, maka parent perlu memakai protected, agar subclass mampu access.

1. Public

Public yang paling tidak aman, mengapa? Karena ini dapat diakses di kelas lain, package lain. Maka data yang ada pun dapat di modif dimana pun. Untuk method menggunakan public, sehingga dapat di access di class lain.

1. Overloading and Overriding

Ini merupakan dari Polymorphism

1. Overloading

-berada di satu kelas yang sama

-dijalankan saat sebelum di compile

-static binding

Contoh: kalau ada dua constructore

Public Teacher(String nama, String nIP){

This.nama = nama;

This.nIP = nIP;

}

Public Teacher(){

}

Nah di panggilnya itu yang Public Teacher().

Maka program pun sudah tau yang akan dijalankan ialah Public Teacher()

1. Overriding

-berada di kelas yang berbeda

-dijalankan saat di compile

-dynamic binding

Contoh: jadi di parent (Layanan) itu ada method

Public void countSaldo(int nominal){

Saldo = 0;

}

Kemudian di anaknya (Tabungan) ada

@Overriding

Public void countSaldo(int nominal){

Saldo += nominal;

}

Kemudian method yang dipanggil ialah t.countSaldo(nominal).

Saat di compile, program mencari method tsb di anaknya, sehingga menghasilkan dari saldo += nominal.

1. 3 methods in ArrayList Class
2. Add()

Method ini berfungsi untuk menambahkan data kedalam ArrayList yang kita buat. Ini menjadi kan kita mudah dalam menambahkan data baru yang diinginkan.

ArrayList<Product> vProduct = **new** ArrayList<>();

Product a = new Product();

vProduct.add(a);

Data a pun masuk dalam array list vProduct.

1. Remove()

Method ini berfungsi untuk menghapus data yang terdapat pada ArrayList yang kita sudah buat.

ArrayList<Product> vProduct = **new** ArrayList<>();

Product a = new Product();

vProduct.remove(a);

Data a pun terhapus dari array list vProduct.

1. Get(index)

Method ini berfungsi untuk mengambil index keberapa sih yang mau kita ambil.

ArrayList<Product> vProduct = **new** ArrayList<>();

vProduct.get(i).showData;

maka data dari Array List vProduct index ke i pun dapat ditampilkan.