UTC OOP

1. Access Modifier
   1. Private : private merupakan access modifier yang scope pemanggilan suatu method atau attribute hanya pada scope class method atau attribute tersebut.
   2. Protected: protected merupakan access modifier yang scope pemanggilan/pengaksesan method atau attribitu yang diberi access modifier tersebut hanya sampai subclass dan package yang sama.
   3. No-Access Modifier/package: saat kita tidak mencantumkan access modifier, maka secara default system akan mengeset access modifier kita menjadi package artinya hanya scope pengakesannya hanya bisa diakses oleh class yang berada pada package yang sama.
   4. Public: pada access modifier public, method atau attribute yang diberi access modifier tersebut akan bias diakses oleh semua anggota package yang sama maupun anggota package yang berbeda.
2. Overloading and Overriding
   1. Overloading adalah salah satu jenis polymorphism dimana salah cara penerapannya adalah dengan cara memberi nama suatu method sama persis dengan method yang lainnya namun diadakan pembedaan dalam parameter nya. Cara memberi pembedaan parameternya adalah jumlah parameter dan tipe parameter. Overloading biasa tipe binding nya adalah static binding karena meskipun Namanya sama, IDE sudah mengetahui method yang mana yang akan digunakan sebelum Compile Time.

Contoh:

Class foo{

void bar(int a){

System.***out***.println(a);

}

Void bar(String a){

System.***out***.println(a);

}

**public** **static** **void** main(String args[]){

foo(1); //maka outputnya adalah 1

foo("agung"); //maka outputnya adalah agung

}

}

* 1. Overriding adalah salah satu jenis polymorphism yang salah satu cara penerapannya adalah memberikan suatu method dari sebuah class adalah hasil inheritance nama dan parameter yang sama persis dengan method parent classnya. Overriding biasanya tipe bindingnya adalah Dynamic Binding karena IDE baru bisa menentukan Method mana yang akan digunakan setelah Compile time.  
       
       
       
       
       
       
       
       
       
       
       
       
       
     Contoh:

Class Foo{

Void bar(){

System.out.println(“parent class”);

}

}

Class Bar extends Foo{

Void bar(){

System.out.println(“Child Class”);

}

**public** **static** **void** main(String args[]){

**super**.bar(); //outputnya : parent class

bar(); //outputnya : child class

}

}

1. Arraylist
   1. arraylist\_name.add(element E) : menambahkan element E kedalam suatu arraylist yaitu arraylist\_name.
   2. arraylist\_name.isEmpty(): method untuk mengecek apakah arraylist arraylist\_name kosong. Akan mereturn sebuat Boolean true/false.
   3. arraylist\_name.size(): method untuk mengetahui ukuran dari arraylist\_name. akan mereturn sebuah integer yaitu size dari arraylist\_name;