* **Enkapsulasi**

Merupakan proses untuk menutupi semua detail dari sebuah object yang tidak menyokong karakteristik khas dari object. Artinya, data property dan method-method dari object hasil abstraksi dibungkus oleh struktur class agar terlindungi.

Rincian implementasi internal suatu object dibuat tersembunyi sehingga tidak bisa diketahui dan diakses oleh object lain. Object lain hanya bisa berkomunikasi dengan object tersebut melalui antarmuka (interface) yang disediakan oleh object tersebut.

Misal pada class *Car*, data variabel *current\_gear* dibuat *private* sehingga object lain tidak bisa mengakses variabel tersebut. Object lain hanya bisa menjalankan method *changeGear(int gear)* tanpa mengetahui posisi gear saat sebelum diganti.

Contoh:

public class MainProgram {

public static void main(String[] args){

Car myCar = new Car();

/\* INI TIDAK BISA DILAKUKAN \*/

myCar.currentGear = 1;

/\* SEDANGKAN INI BISA DILAKUKAN \*/

myCar.changeGear(2);

}

}

* **Inheritance (pewarisan)**

Sebuah class bisa mewariskan atribut property-nya dan method-nya ke class yang lain. Class yang mewarisi disebut superclass (kelas induk). Class yang diberi warisan disebut subclass (kelas turunan).

Pewarisan memungkinkan kita untuk menciptakan kelas baru (kelas turunan) yang merupakan perluasan atau spesialisasi dari kelas lain yang telah ada (kelas induk).

Kelas turunan akan memiliki property dan method yang sama dengan kelas induk, dimana pada kelas turunan memungkinkan untuk menambahkan property atau method-method baru. Kelas turunan bisa melakukan definisi ulang terhadap method-method yang sudah ada di kelas induknya (*overriding*).

Contoh: anda bisa membuat class *ModernCar* yang merupakan turunan dari class *Car.*Artinya, class *ModernCar* memiliki property dan method yang sama dengan induknya tanpa perlu dideklarasikan ulang. ?Agar menjadi unik, class *ModernCar* menambahkan method *retreat()* pada definisi class-nya.

public class ModernCar extends Car {

public void retreat(){

   // Implementasi method untuk memundurkan mobil

}

}

* **Polymorphism**

Merupakan kemampuan suatu object untuk melakukan hal yang berbeda-beda dengan menggunakan interface yang sama. Hal ini tercermin dengan menggunakan teknik overriding dan overloading. Overriding adalah penulisan ulang method warisan dari kelas induk. Overloading adalah penulisan method yang sama dengan parameter input yang berbeda.

Contoh overriding:

public class Animal {

String nama;

public void walk(){

System.out.println("Walk with 2 legs");

}

}

public class Dog extends Animal {

// overriding method walk() yang sudah ada di class Animal

public void walk(){

System.out.println("Walk with 4 legs");

}

}

public class Fish extends Animal {

        // overriding method walk() yang sudah ada di class Animal

public void walk(){

System.out.println("Sorry can't walk");

}

}

Contoh overloading:

public class Dog extends Animal {

public void walk(){

System.out.println("Walk with 4 legs");

}

// overloading method walk() di atas dengan penambahan parameter integer

public void walk(int leg){

if ((leg<=4) && (leg>0)){

System.out.println("Walk with "+leg+" legs");

}

}

}

**Contoh kasus**

**Alata counter**

#include <stdio.h>

int value=0;

void next();

void reset();

void display();

int main(){

int menu;

scanf("%d",&menu);

if (menu==1){

next();

}else if (menu==2){

reset();

}

display();

}

void next(){

value++;

}

void reset(){

value=0;

}

void display(){

printf("%d",value);

}

Pada contoh kasus alat counter di atas, dikerjakan dengan pendekatan top-down, yaitu:

1. identifikasi program counter tersebut memiliki fungsi-fungsi apa saja. Fungsi yang dimiliki adalah:
   1. next (untuk menjalankan nilai counter)
   2. reset (untuk mengembalikan nilai counter menjadi nol)
   3. display (untuk menampilkan nilai counter di layar)
2. identifikasi program counter tersebut memiliki data apa saja. Data yang dimiliki adalah:
   1. value (untuk menampung nilai counter saat ini)
   2. menu (untuk menampung tombol mana yang dipilih oleh user)
3. deklarasi data dan fungsi-fungsi tersebut sebagai variabel dan prosedur (baris 3 sampai baris 7)
4. buat kerangka program seperti code di atas (baris 8 sampai baris 18) yang berfungsi sebagai alur berjalannya program
5. buat detail dari tiap-tiap prosedur (baris 19 sampai baris akhir)