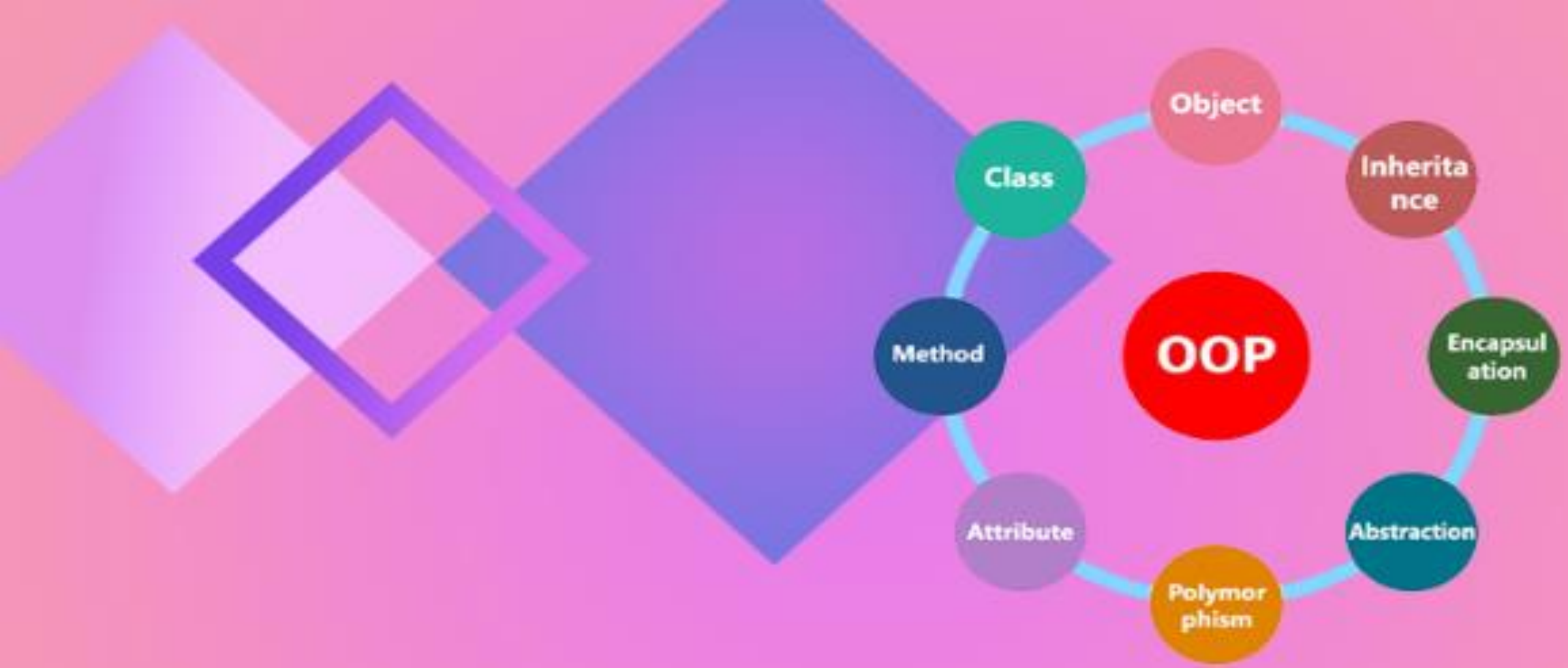


Menerapkan Inheritance, Polymorphisme, Encapsulation

INHERITANCE

- Inheritance merupakan proses pewarisan data dan method dari suatu class yang telah ada kepada suatu class baru. Class yang mewariskan disebut dengan superclass / parent class / base class, sedangkan class yang mewarisi (class yang baru) disebut dengan subclass / child class / derived class. Subclass tidak dapat mewarisi anggota private dari superclass-nya. Dengan inheritance, class yang baru (subclass) akan mirip dengan class yang lama (superclass) namun memiliki karakteristik yang baru. Dalam Java, subclass hanya bisa memiliki satu superclass (single inheritance) sedangkan superclass bisa memiliki satu subclass atau lebih. Untuk menerapkan inheritance, gunakan statement "extends". Keyword "super" digunakan oleh subclass untuk memanggil constructor, atribut dan method yang ada pada superclass-nya.

INHERITANCE

Ada 2 Method dalam Inheritance yaitu:

- Overriding adalah suatu cara untuk mendefinisikan ulang method yang ada pada class induk apabila class anak menginginkan adanya informasi yang lain. Overriding dilakukan dengan cara menulis ulang method yang ada pada class induk dengan syarat bahwa nama dan parameter fungsi tersebut harus sama (tidak boleh diubah). Meskipun fungsi telah ditulis ulang oleh class anak, fungsi yang asli pada class induk masih dapat dipanggil di class anak dengan menggunakan class super.
-
- Overloading fungsi adalah penulisan beberapa fungsi (dua atau lebih) yang memiliki nama yang sama. Pada bahasan overloading dikenal istilah signature. Signature sebuah fungsi adalah parameter lengkap dengan tipe datanya yang terdapat dalam fungsi tersebut.

CONTOH INHERITANCE DAN OVERRIDING

Pendefinisian class dosen

```
Public class Dosen {

        protected String nama

        protected String nik;

        protected String jurusan;

Dosen (String namaX, String nikX, String
jurX)

{

nama = namaX;

nik = nikX;

jurusan = jurX;}

public void view() {

System.out.println("Nama : " +nama);

System.out.println("nik : " +nik);

System.out.println("jurusan : " +jurusan)
```

Constructor untuk kelas Dosen

Method yang akan dilakukan overriding

Pendefinisian class rektor

```
public class rector extends Dosen{

        private int th mulai;

        private int jabatan_ke;

Rektor (String namaX, String nikX,
String jurX, int thX, int keX){

super(namaX, nikX, jurX);

{th mulai = thX;

jabatan_ke = keX;

}

public void viewrektor(){

System.out.println("Th mulai jabatan:"
+th mulai);

System.out.println("jabatan rector
ke:" +jabatan ke);
```

Memanggil variable yang mengacu pada dosen (super class)

Overriding metode view pada superclass dosen

CONTOH INHERITANCE DAN OVERLOADING

Contoh overloading (1)

```
Public class mahasiswa{

Public void infoMahasiswa(int laki2,
int pr, String kelas){

int jumlah = laki2 + pr;

System.out.prinyln("jumlah mhasiswa="
+jumlah);

}

Public void infoMahasiswa(int mhsLama,
int mhsBaru, int mhsCuti, int
angkatan){

Int jumlah = mhsLama+mhsBaru+mhsCuti);

System.ou.println("sampai tahun"
+angkatan+ "jumlah mahasiswa"
+jumlah);

}

}
```

Contoh overloading (1)

```
public class Main {

public static void main(String[] args){

mahasiswa M= new mahasiswa();

M.infoMahasiswa(60,18,"kelas A angkatan
2013");

M.infoMahasiswa(1000,400,25,2008);

}}
```

overloading

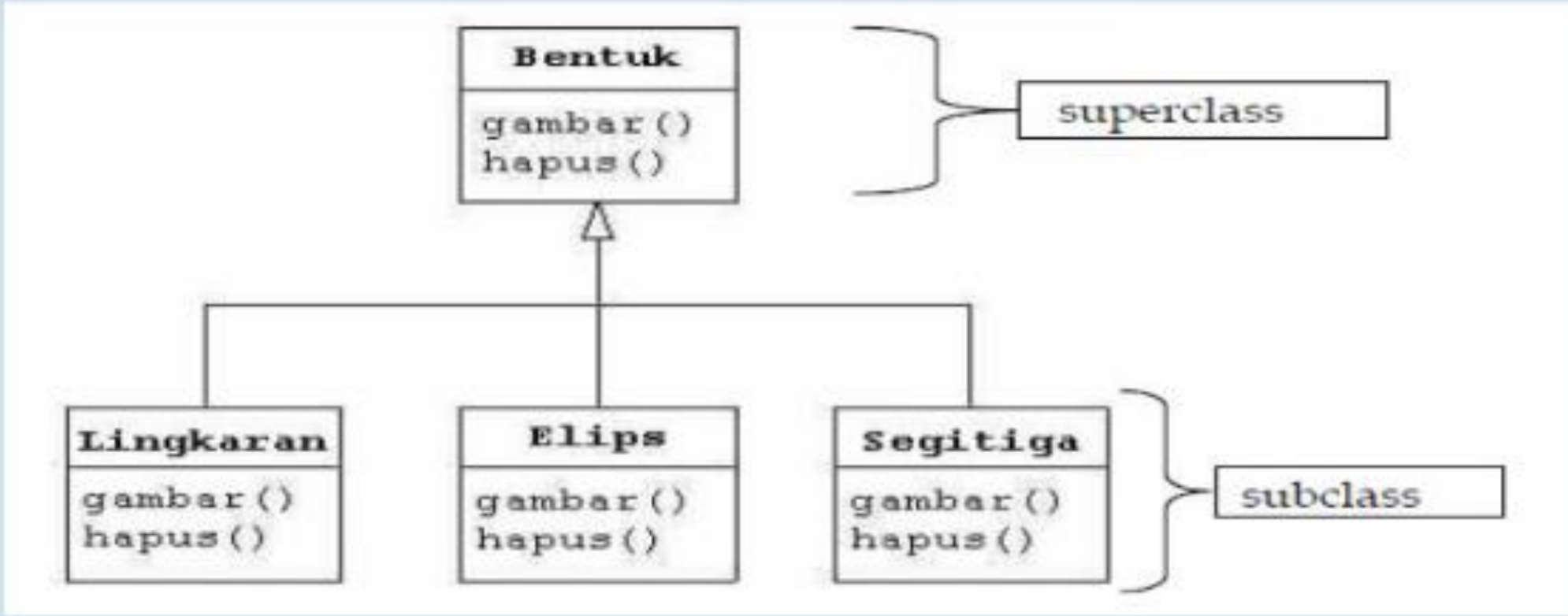
Pada class tersebut terdapat dua subclass dengan nama yang sama yaitu infoMahasiswa namun memiliki signature yang berbeda. Subclass pertama digunakan untuk menghitung jumlah mahasiswa kelas A angkatan 2008. Sedangkan pada subclass kedua digunakan untuk menghitung jumlah mahasiswa aktif sampai tahun 2008.

Overloading fungsi ditentukan oleh signature nya, tidak ditentukan oleh nilai balikan fungsi. Dalam satu class, apabila ada dua atau lebih fungsi yang memiliki nilai balikan yang berbeda namun memiliki signatue yang sama, maka fungsi tersebut tidak dapat dikatakan sebagai overloading.

POLIMORPHISME

- **Polimorfisme** adalah kemampuan mengungkap suatu hal yang berbeda melalui satu cara yang sama. Apabila mempunyai objek yang bertipe superkelas, variable objek ini bisa diisi dengan objek superkelas ataupun objek subkelas tanpa perlu melakukan perubahan tipe.Pada polimorfisme kondisi yang harus terpenuhi supaya fungsi dapat dijalankan yaitu:
- Semua kelas diturunkan dari suatu kelas yang sama.
- Method yang dipanggil harus melalui variabel dari super class.
- Method yang dipanggil juga harus merupakan method yang ada pada sup er class.
- Signature method harus sama baik yang ada pada super class maupun di subclass.

POLIMORPHISME



CONTOH POLIMORHISME

Program polimorfisme(1)

```
class Binatang{
public void info() {
System.out.println(" Info tentang
Hewan : ");
}
} class Herbivora extends Binatang {
public void info() {
System.out.println ("Info pada
herbivora: Memakan makanan
berupa tumbuh - tumbuhan");
}
}
class Kelinci extends Herbivora{
public void info(){
Syatem.out.println("Info pada Kelinci:
Memakan makanan berupa
wortel");
}
}
public class Polimorfisme {
public static void main(String[] args)
{
Herbivora herbivora;
Kelinci kelinciku;
Binatang hewan;
herbivora=new Herbivora();
kelinciku=new Kelinci();
hewan=herbivora;
hewan.info();
hewan=kelinciku;
hewan.info();
}
}
```

Program polimorfisme (2)

Pendefinisian class kendaraan

```
public class kendaraan {
private String model;
public kendaraan (String model){
this.model = model;
}
Public void informasi(){}
}
```

merupakan program untuk membangun class kendaraan. Pada class kendaraan mewarisi ke tiga class, yaitu class pesawat, mobil, dan kapal.

Pendefinisian class pesawat

```
Public class pesawat extends kendaraan{
Private String nama;
Private String jenis;
}
Public pesawat(String nama){
Super("pesawat");
This.nama = nama;
Jenis = "belum teridentifikasi";
}
Public pesawat (String nama, String
jenis){
Super("pesawat");
This.nama=nama;
This.jenis=jenis;
}
Public void informasi(){
Syatem.out.println("nama pesawat
adalah"+nama);
Syatem.out.println("jenis pesawat
adalah"+jenis);}}
```

ENKAPSULASI

- Enkapsulasi (*encapsulation*) merupakan cara untuk melindungi property(atribut)/method tertentu dari sebuah class agar tidak sembarangan diakses dan dimodifikasi oleh suatu bagian program. Cara untuk melindungi data yaitu dengan menggunakan *access modifiers* (hak akses). Ada 4 hak akses yang tersedia,yaitu default,public,protected,private.

ENKAPSULASI

No	Modifier	Pada class dan interface	Pada method dan variabel
1	Default (tidak ada modifier)	Dapat diakses oleh yang sepaket	Diwarisi oleh subkelas dipaket yang sama, dapat diakses oleh method-method yang sepaket
2	Public	Dapat diakses dimanapum	Diwarisi oleh subkelasnya, dapat diakses dimanapun
3	Protected	Tidak bisa diterapkan	Diwarisi oleh subkelasnya, dapat diakses oleh method-method yang sepaket
4	private	Tidak bisa diterapkan	Tidak dapat dikases dimanapun kecuali oleh method-method yang ada dalam kelas itu sendiri

Aksesabilitas	public	private	protected	default
Dari kelas yang sama	Ya	Ya	Ya	Ya
Dari sembarang kelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari sembarang kelas di luar paket	Ya	Tidak	Tidak	Tidak
Dari subkelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari subkelas di luar paket	Ya	Tidak	Ya	Tidak

IMPLEMENTASI ENKAPSULASI (AKSES MODIFIER)

Suatu cara untuk meyembunyikan informasi detail dari suatu class, berupa information hiding dan interface untuk akses data. Hak akses data dan perilaku objek memiliki tingkatan:

- Metode serta variable bersifat private: atribut Class hanya dapat di akses oleh metode dalam Class dimana Class tersebut didefinisikan
- Metode serta variable bersifat public : variabel dan metode dapat diakses dari dalam maupun luar Class
- Metode serta variable bersifat protected: atribut Class hanya dapat di akses oleh Class dan subClass tersebut
- Metode serta variable bersifat default : hanya Class dalam paket dapat mengakses variable dan metode Class