Chapter 9
Pewarisan

A. Konsep Pewarisan

- Pewarisan adalah proses penciptaan kelas baru dengan mewarisi karakteristik kelas yang telah ada, ditambah karakteristik unik kelas baru tersebut.
- Kelas yang menjadi turunannya mewarisi fungsionalitas yang telah ada
- Pewarisan dapat digunakan untuk mengorganisasikan program maupun penciptaan klasisfikasi berhirarki.
- Dalam Java kelas yang diwarisi disebut superkelas sedangkan kelas yang mewarisi disebut subkelas
- Subkelas mewarisi semua metode dan variabel superkelas, hal tersebut sama halnya dengan mengopi kode dari kelas lain

B. Pembuatan kelas turunan

- Kata kunci pembuatan kelas turunan adalah *extend*
- Contoh syntak program :

```
Class Identifier extends SuperClass {
        ClassBody
}
```

Penjelasan : identifier mengacu ke nama kelas turunan baru yang akan dibuat. Superclass mengacu nama kelas yang diwarisi. Sedangkan classBody merupakan badan kelas baru.

- Apabila kita ingin menurunkan superkelas eksternal, maka harus melakukan import dengan pernyataan import

Contoh kelas turunan

- Kelas StudentToy yang merupakan subkelas PersonToy. Pada kelas StudentToy ditambah data nrp, strata, department serta metode pengaksesannya.

Syntak contoh PersonToy.java

```
Package toy;

Public class PersonToy  {
                String name;
                String addressLine1;
                String addressLine2;
                String city;
                Int age;

Public PersonToy() {
                                Name  = “ ”;
AddressLine1 = “ “;
AddressLine2 = “ “;
City = “ “;
Age = 0;
}
```

Next syntax.....

```
Public PersonToy(String newName, String newAddressLine1, String newAddressLine2, String
newCity, int newAge) {
Name = newName;
Address1 = newAddressLine1;
.
.
}

Public void setName (String newName){
                Name = newName;
}
.
.
Public  String toString(){
String str =
                "Nama : "+ name + "\n"+
                Alamat : "+ addressLine1 + "\n"+....."";
Return str;
}
```

Next syntax…..

```
Static void test(){
                PersonToy t = new PersonToy(“Slamet”, “Jln. Warung Boto 94”, “Jln. Sudirman 78
Jakarta”,”Jakarta”,21);

System.out.println(“Slamet sebagai Person”);
System.out.println(t.getName());
System.out.println(t.getAddressLine1());
System.out.println(t.getAddressLine2());
System.out.println(t.getCity());
System.out.println(t.getAge());
System.out.println(t);
}
Public static void main (String[] args) {
                        Test();
}
}
```

Syntak contoh StudentToy.java

- Karena studentToy merupakan kelas turunan dari personToy, maka pada kelas ini dapat menambah variabel baru dengan tetap mengikutkan variabel yang dimiliki oleh super kelasnya yaitu personToy.

```
Package toy;

Public class StudentToy extend PersonToy  {
                String nrp;
                String strata;
                String department;

Public StudentToy() {
                                Super();
nrp= “ “;
strata= “ “;
department = “ “;
}
```

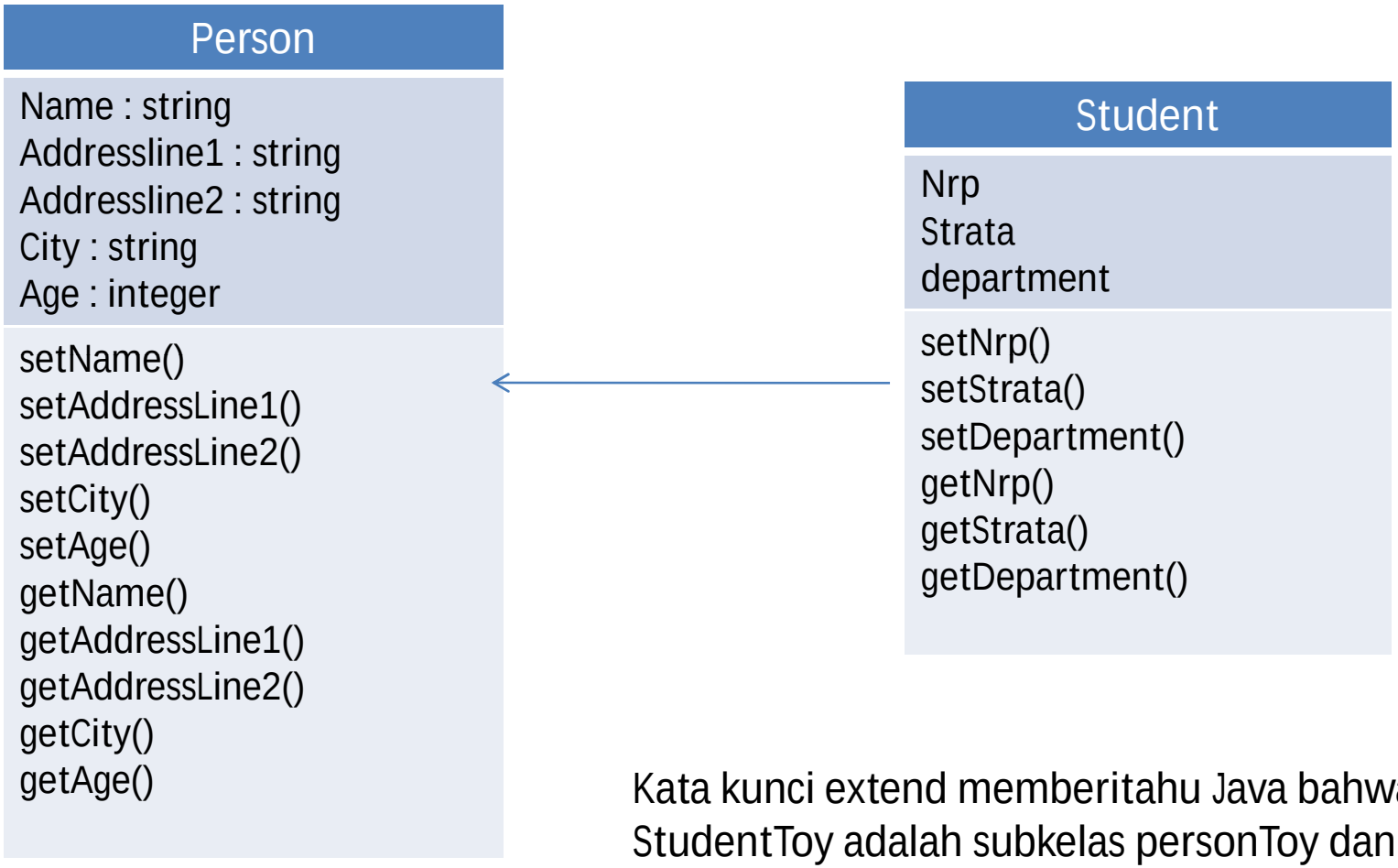
Next syntax.....

```
Public StudentToy(String newName, String newAddressLine1, String
newAddressLine2, String newCity, int newAge, String newNrp, String newStrata,
String newDepartment) {
Super(String newName, String newAddressLine1, String newAddressLine2, String
newCity, int newAge);
nrp = newNrp;
strata = newStrata;
department=newDepartment;
}

Public void setNrp (String newNrp){
                nrp = newNrp;
}
.
.
Public  String toString(){
String str =
                "Nama : "+ name + "\n"+
                Alamat : "+ addressLine1 + "\n"+.....
Nrp :"+ nrp +"\n"+.....";
Return str;
}
```

Next syntax.....

```
Static void test(){
        StudentToy t = new StudentToy("Slamet", "Jln. Warung Boto 94", "Jln. Sudirman 78
Jakarta","Jakarta",21);

System.out.println("Slamet sebagai Student");
System.out.println(t.getName());
System.out.println(t.getAddressLine1());
System.out.println(t.getAddressLine2());
System.out.println(t.getCity());
System.out.println(t.getAge());
System.out.println(t.getNrp())
System.out.println(t.getStrata())
System.out.println(t.getDepartment())
System.out.println(t);
}
```

Diagram UML dari PersonToy & StudentToy



Kata kunci extend memberitahu Java bahwa StudentToy adalah subkelas personToy dan mewarisi field dan metode di kelas personToy

C. Petunjuk ringkas penggunaan pewarisan

- Petunjuk penggunaan pewarisan, yaitu :
  - Tempatkan operasi-operasi dan field-field yang sama di superkelas
  - Jangan gunakan protected fields
  - Gunakan pewarisan untuk memodelkan hubungan “is-a”
  - Jangan gunakan pewarisan kecuali semua metode yang diturunkan berarti
  - Gunakan polymorphism, bukan informasi tentang tipe

Petunjuk ringkas penggunaan pewarisan....

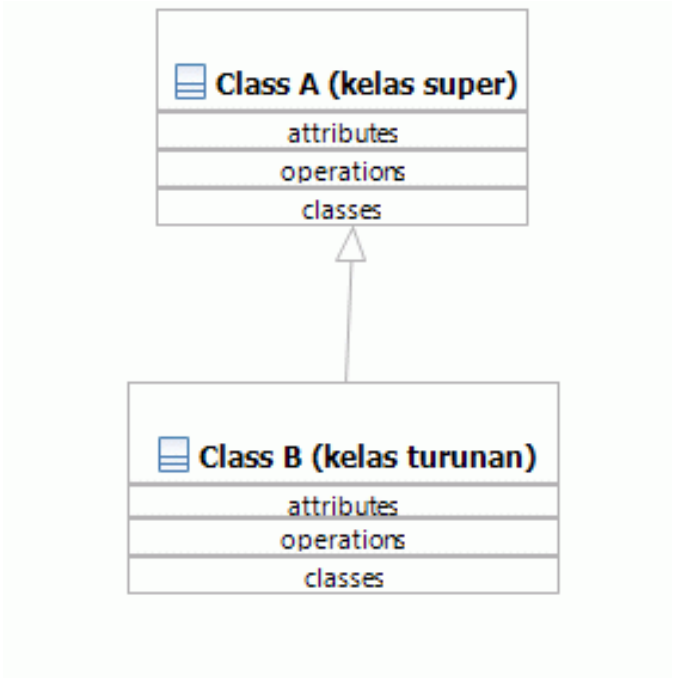
1. Tempatkan operasi-operasi pada field-field yang common di superkelas; apabila pada contoh personToy dan studentToy, maka field name, birthday di kelas PersonToy, tidak mereplikasinya di kelas StudentToy
2. Tidak menggunakan protected fields; Metode protected dapat berguna untuk mengindikasikan metode yang belum siap untuk penggunaan umum dan seharusnya diredefinisi di subkelas-subkelas.
3. Gunakan pewarisan untuk memodelkan hubungan “is-a” ; penggunaan pewarisan perlu memperhatikan bahwa subkelas memang merupakan kelas dari superkelasnya

Petunjuk ringkas penggunaan pewarisan....

4. Tidak menggunakan pewarisan kecuali semua metode yang diturunkan berarti; artinya untuk melakukan pewarisan perlu diperhatikan bahwa sifat-sifat dari superkelas memang dibutuhkan oleh subkelas

D. Contoh-contoh

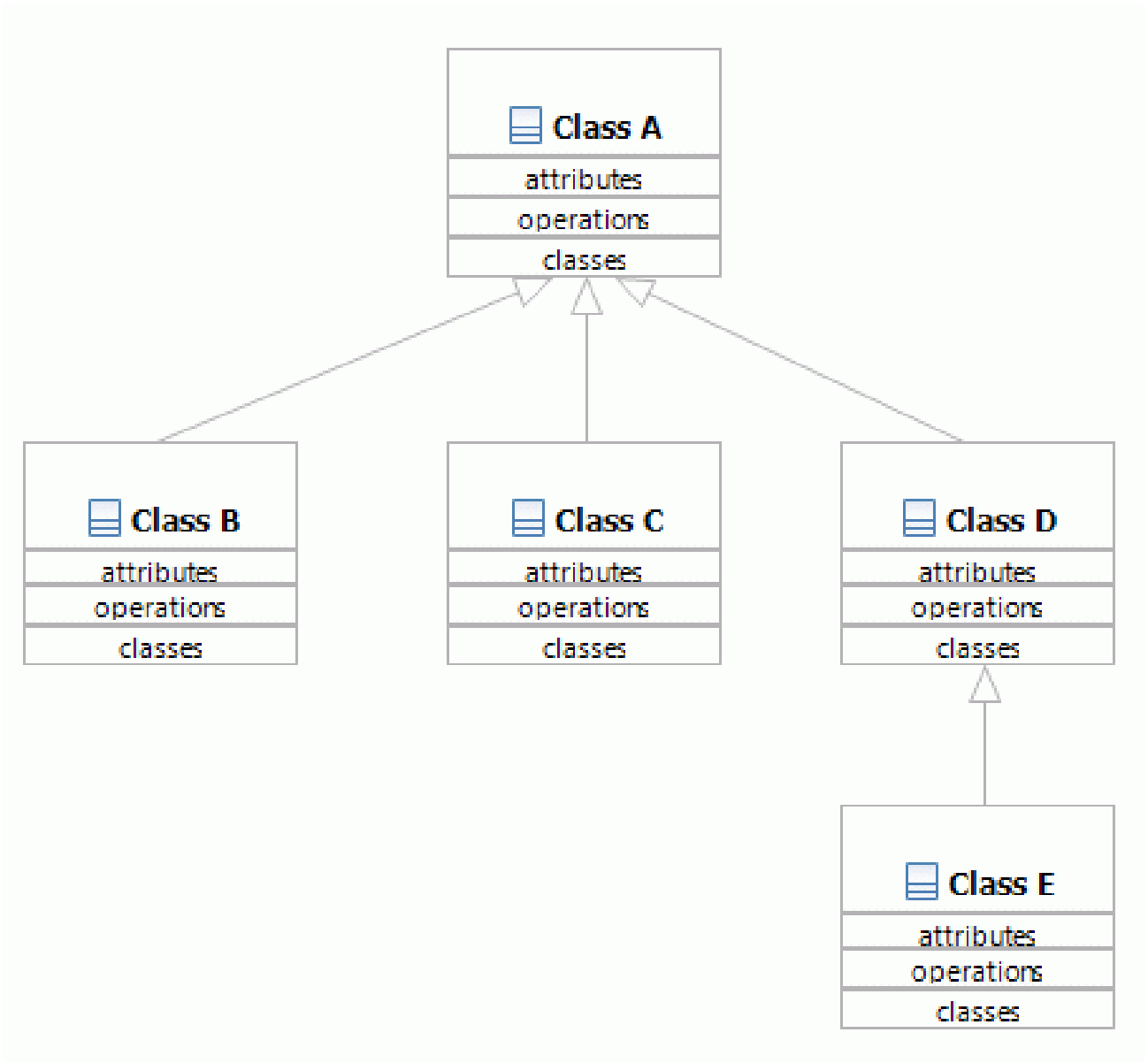
- Jika kelas B adalah kelas turunan dari kelas A, maka kita bisa juga menyebut kelas A adalah kelas super dari kelas B. Kelas turunan bisa memiliki struktur atau perilaku tambahan dari kelas supernya. Atau bahkan kelas turunan bisa mengubah atau mengganti perilaku kelas supernya. Hubungan antara kelas turunan dan kelas super sering dilukiskan dalam bentuk diagram di mana kelas turunan digambarkan di bawah kelas supernya, dan dihubungkan dengan garis penghubung dengan tanda segitiga yang diletakkan di dekat kelas supernya.



- Deklarasi dalam Java

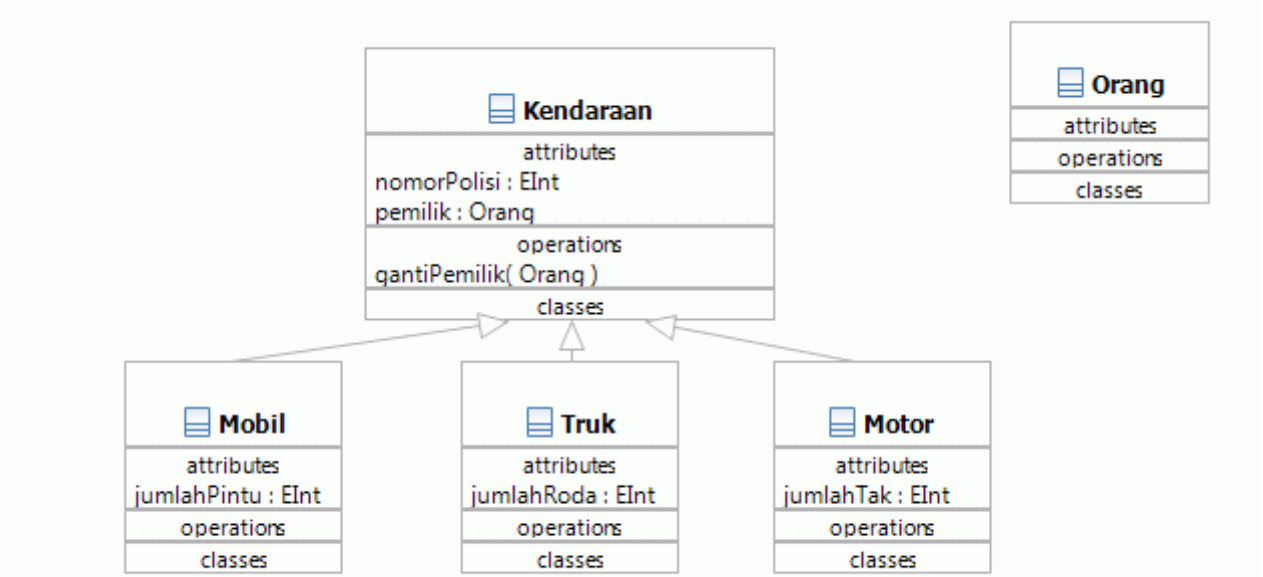
```
class B extends A {
                // tambahan atau perubahan
                // struktur dan perilaku dari kelas A
}
```

- Beberapa kelas dapat menurunkan kelas yang sama. Kelas-kelas turunan ini masing-masing disebut kelas saudara, yaitu diwariskan dari satu kelas super yang sama. Struktur dan perilaku kelas super ini akan dimiliki oleh masing-masing turunannya. Pada diagram berikut, kelas B, C, dan D adalah kelas saudara. Pewarisan juga bisa dilakukan beberapa kali, atau suatu kelas bisa memiliki cucu, buyut, dan seterusnya. Pada diagram, kelas E merupakan kelas turunan kelas D, sehingga kelas E adalah "cucu" dari kelas A. Kelas E masih bisa disebut turunan dari kelas A, walaupun bukan merupakan turunan langsungnya.


Class A
attributes
operations
classes
Class B
attributes
operations
classes
Class C
attributes
operations
classes
Class D
attributes
operations
classes
Class E
attributes
operations
classes

- Ada 3 kelas turunannya yaitu Mobil, Truk dan Motor yang akan menyimpan variabel dan metode khusus untuk setiap jenis kendaraan. Kelas Mobil misalnya memiliki variabel jumlahPintu, kelas Truk memiliki variabel jumlahRoda, dan kelas Motor memiliki variabel jumlahTak. Kelas-kelas ini bisa dideklarasikan dalam Java dalam bentuk

```
class Kendaraan {
int nomorPolisi;
Orang pemilik; // (anggap kelas Orang telah dibuat sebelumnya)

void gantiPemilik(Orang pemilikBaru) {
. . .
 }
. . .
}
class Mobil extends Kendaraan {
int jumlahPintu;
. . .
}
class Truk extends Kendaraan {
int jumlahRoda;
. . .
}
class Motor extends Kendaraan {
int jumlahTak;  // 2-tak atau 4-tak
. . .
}
```



- Apabila *mobilku* adalah variabel dengan tipe Mobil akan dideklarasikan dan diinisialisasi dengan pernyataan berikut

```
Mobil mobilku = new Mobil();
```

- Dengan deklarasi seperti ini, maka program akan bisa mengakses mobilku.jumlahPintu, karena jumlahPintu adalah variabel instansi dari kelas Mobil. Akan tetapi karena kelas Mobil merupakan turunan dari kelas Kendaraan, maka mobil ini juga memiliki stuktur dan perilaku dari kendaraan. Artinya program juga bisa mengakses mobilku.nomorPolisi, mobilku.pemilik, dan menjalankan metode mobilku.gantiPemilik()

- Dari diagram di atas, maka dapat dilakukan pencetakan informasi sebagai berikut :

```
System.out.println("Data Kendaraan:");
System.out.println("Nomor polisi: " + kendaraanku.nomorPolisi);
if (kendaraanku instanceof Mobil) {
        System.out.println("Jenis kendaraan: Mobil");
        Mobil m = (Mobil)kendaraanku;
        System.out.println("Jumlah pintu: " + m.jumlahPintu);
        }
else if (kendaraanku instanceof Truk) {
        System.out.println("Jenis kendaraan: Truk");
        Truk t = (Truk)kendaraanku ;
        System.out.println("Jumlah roda: " + t.jumlahRoda);
        }
else if (kendaraanku instanceof Motor) {
        System.out.println("Jenis kendaraan: Motor");
        Motor sm = (Motor)kendaraanku ;
        System.out.println("Jumlah tak: " + sm.jumlahTak);
        }
```

- Perhatikan bahwa untuk setiap jenis objek, komputer akan menguji satu per satu tipe objek yang disimpan dalam kendaraanku. Jika kendaraanku[code] merujuk pada objek bertipe Truk maka casting [code](Mobil)kendaraanku akan menampilkan pesan kesalahan.