PEMROGRAMAN BERBASIS OBJECT

Minggu 6: **Inheritance**

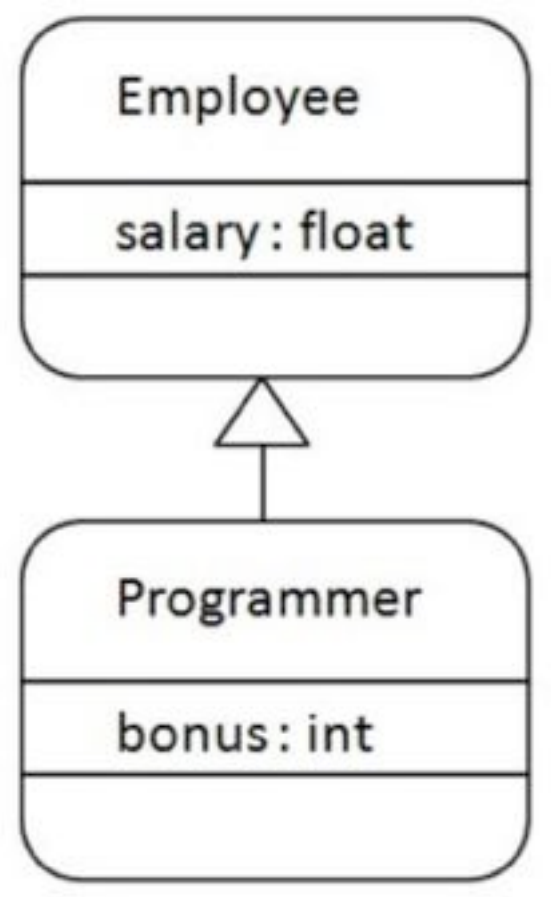
Outline

- Jenis Inheritance
- Rules pada Inheritance di Java

Review

- **Inheritance (Pewarisan)** merupakan mekanisme dimana suatu objek dapat memperoleh property dan behaviour dari objek induk.
- Konsep inheritance: dapat membuat kelas baru yang dibangun di atas kelas yang sudah ada.
- Atribut dan method pada kelas induk dapat digunakan kembali di kelas turunan, serta dapat menambahkan atribut dan method baru di kelas turunan.
- Pewarisan adalah hubungan **IS-A**

Review




```
class Employee{
 float salary=40000;
}
class Programmer extends Employee{
 int bonus=10000;
 public static void main(String args[]){
   Programmer p=new Programmer();
   System.out.println("Programmer salary is:"+p.salary);
   System.out.println("Bonus of Programmer is:"+p.bonus);  }
}
```

Jenis Inheritance

- ❑ Single
- ❑ Multilevel
- ❑ Hierarchical
- ❑ Hybrid
- ❑ Multiple → tidak bisa digunakan pada Java

Single Inheritance

- Konsep single inheritance hanya memperbolehkan suatu sublass mempunyai satu parent class.

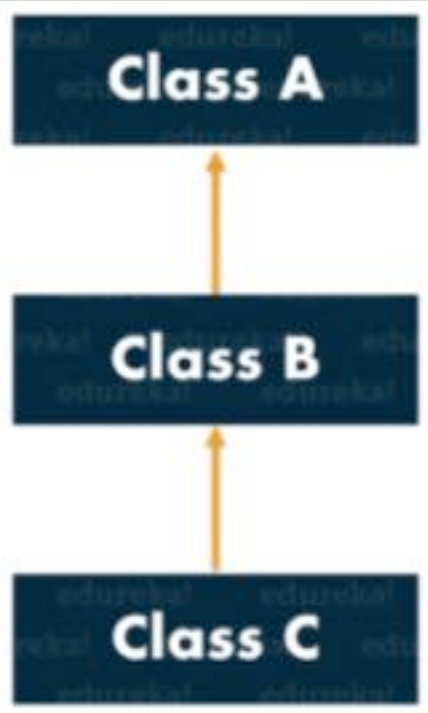

```
class Animal{
void eat(){System.out.println("eating...");}
}

class Dog extends Animal{
void bark(){System.out.println("barking...");}
}

class TestInheritance{
    public static void main(String args[]){
            Dog d=new Dog();
            d.bark();
            d.eat();
     }
}
```

Multilevel Inheritance

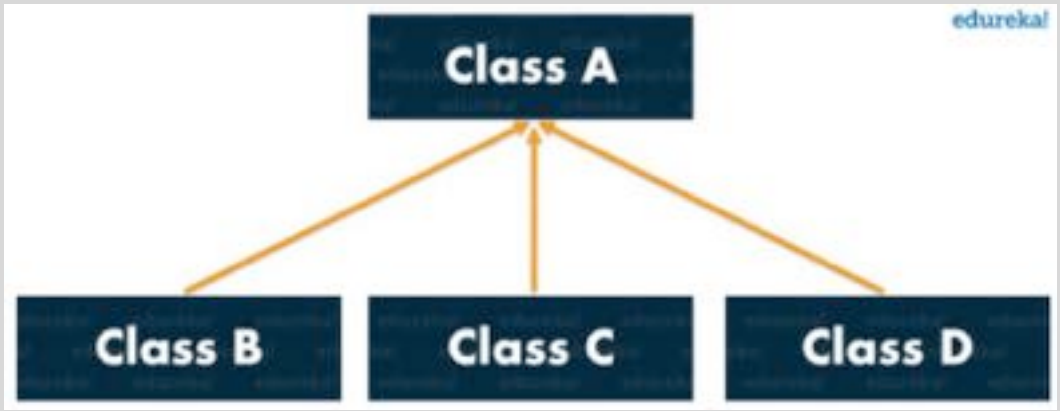
- Ketika kelas diturunkan dari kelas yang juga diturunkan dari kelas lain, yaitu kelas yang memiliki lebih dari satu kelas induk tetapi pada tingkat yang berbeda




```
class Animal{
    void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
     void bark(){System.out.println("barking...");}
}
class BabyDog extends Dog{
      void weep(){System.out.println("weeping...");}
}
class TestInheritance2{
     public static void main(String args[]){
           BabyDog d=new BabyDog();
           d.weep();
           d.bark();
           d.eat();
     }
}
```

Hierarchical Inheritance

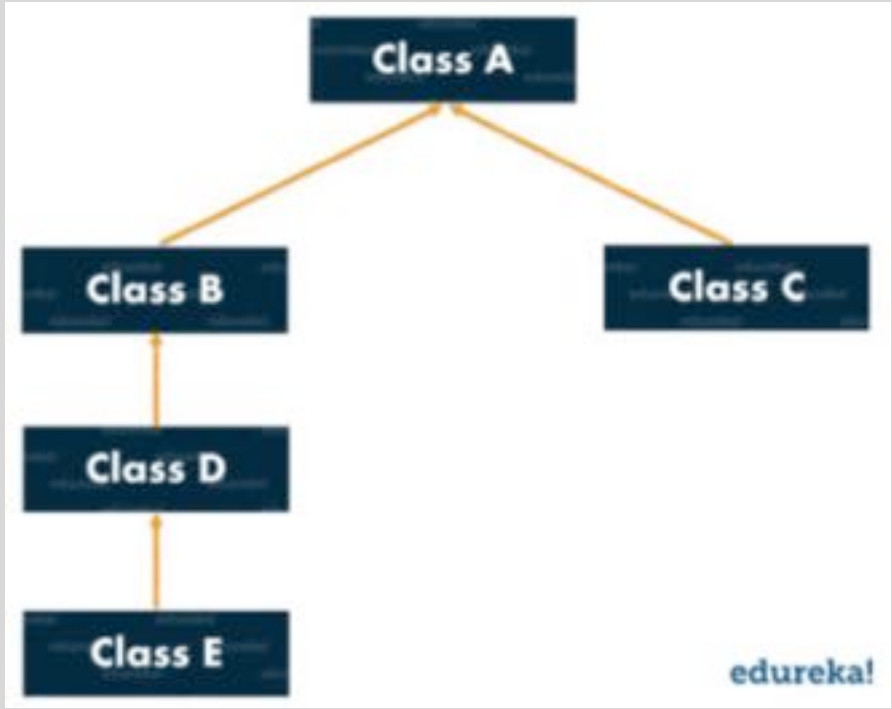
- Ketika sebuah kelas memiliki lebih dari satu kelas turunan (*subclass*) atau dengan kata lain, lebih dari satu kelas turunan memiliki kelas induk yang sama.




```
class Animal{
   void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
   void bark(){System.out.println("barking...");}
}
class Cat extends Animal{
   void meow(){System.out.println("meowing...");}
}
class TestInheritance3{
    public static void main(String args[]){
            Cat c=new Cat();
            c.meow();
            c.eat();
            //c.bark();//C.T.Error
    }
}
```

Hybrid Inheritance

- Kombinasi dua atau lebih jenis inheritance.



Hybrid Inheritance

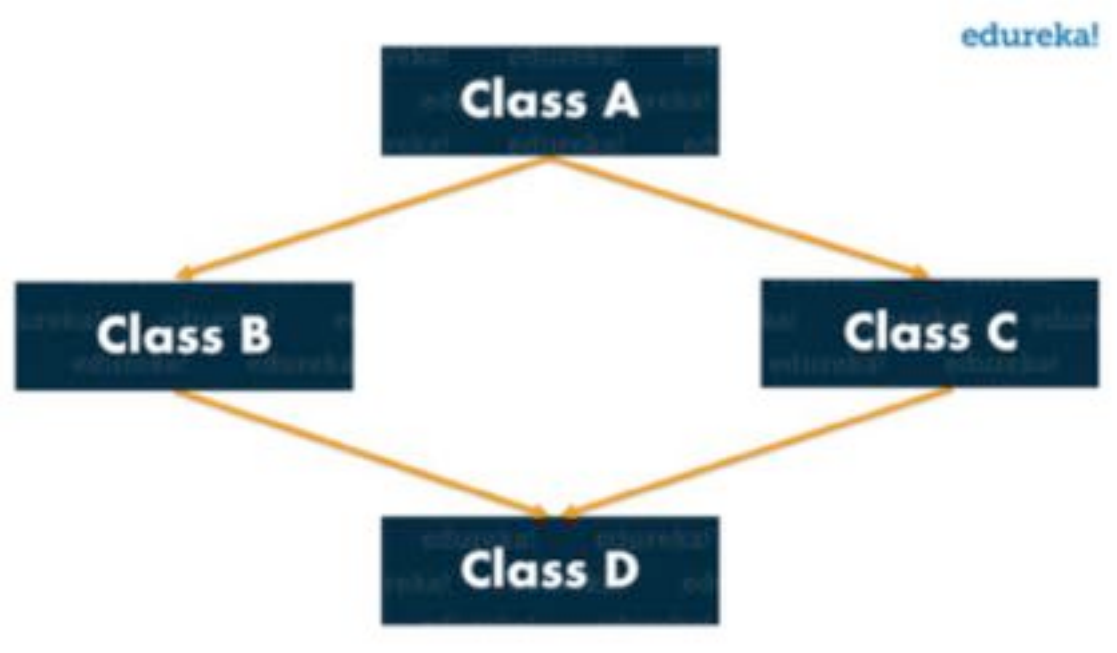
```
class A
{
   public void display()
   {
          System.out.println(“A");
   }
}

class B extends A
{
   public void display()
   {
          System.out.println(“B");
   }
}
```

```
class C extends A
{
   public void display()
   {
          System.out.println(“C");
   }
}

class D extends B
{
   public void display()
   {
          System.out.println("D");
   }
   public static void main(String args[]){
          D obj = new D();
          obj.display();
   }
}
```

Mengapa Multiple Inheritance tidak bisa digunakan di Java?



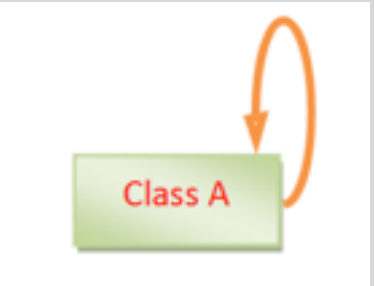
- Multiple inheritamce mengacu pada proses dimana satu kelas turunan mencoba untuk **memperluas (extend) lebih dari satu kelas induk**.
- Misal terdapat method *show()* pada kelas B dan C dengan fungsi yang berbeda. Kemudian kelas D meng-extend kelas B dan C. Ketika objek dari kelas D mencoba memanggil method *show()*, kompilator akan bingung method di kelas mana yang akan dieksekusi (dari kelas B atau C)
- Mengarah pada ambiguitas.

Mengapa Multiple Inheritance tidak bisa digunakan di Java?

```
class A{
void message(){System.out.println("Hello");}
}
class B extends A{
void show(){System.out.println("Welcome");}
}
class C extends A{
void show(){System.out.println("Good Morning");}
}
```

```
class D extends B,C{
    void print(){System.out.println("This is multiple inheritance");}
}
class TestInheritance4{
     public static void main(String args[]){
            D obj=new D();
            obj.print();
            obj.show();//method show() mana yang akan dipanggil
      }
}
```

Rules Inhertiance di Java



- Cyclic inheritance tidak dapat digunakan di Java.

Cyclic inheritance merupakan jenis inheritance dimana sebuah kelas memperluas (extend) dirinya sendiri. Jenis ini tidak diizinkan oleh Java karena tidak ada peluang untuk memperluas kelas Object.

- Atribut dan method dengan access modifier Private tidak diwariskan.
- Constructor tidak dapat diwariskan. Jika ingin menggunakan constructor kelas induk, harus menggunakan perintah super() pada constructor kelas anak.

Latihan

- Carilah sebuah studi kasus dari hierarchical dan hybrid inheritance, kemudian gambarkan UML class diagramnya.

TERIMA KASIH!