

# **Pemrograman Berorientasi Objek**



**UNIVERSITAS PGRI WIRANEGARA PASURUAN**

# **INHERITANCE**

# TOPIK

Pengertian inheritance

Deklarasi inheritance

Single inheritance

Multilevel inheritance

Access Control

Konstruktor tidak diwariskan

super keyword

# PENGERTIAN DASAR INHERITANCE

- ❑ **Inheritance** (Pewarisan) merupakan salah satu dari konsep dasar OOP.
- ❑ Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan.
- ❑ Dengan konsep inheritance, sebuah class dapat mempunyai class turunan.

# PENGERTIAN DASAR INHERITANCE

- ❑ Suatu class yang mempunyai class turunan dinamakan **parent class** atau **base class**.
- ❑ Sedangkan class turunan itu sendiri seringkali disebut **subclass** atau **child class**.
- ❑ Suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class.

# DEKLARASI INHERITANCE

- ❑ Dengan menambahkan kata kunci **extends** setelah deklarasi nama class, kemudian diikuti dengan nama parent class-nya.
- ❑ Kata kunci extends tersebut memberitahu kompiler Java bahwa kita ingin melakukan perluasan class.

# DEKLARASI INHERITANCE

```
public class B extends A
{
    ...
}
```

# KAPAN KITA MENERAPKAN INHERITANCE?

Kita baru perlu menerapkan inheritance pada saat kita jumpai ada suatu class yang dapat diperluas dari class lain.

## CLASS PEGAWAI

```
public class Pegawai {  
    public String nama;  
    public double gaji;  
}
```

## CLASS MANAGER

```
public class Manajer {  
    public String nama;  
    public double gaji;  
    public String departemen;  
}
```



Dari 2 buah class diatas, kita lihat class Manajer mempunyai data member yang identik sama dengan class Pegawai, hanya saja ada tambahan data member departemen.

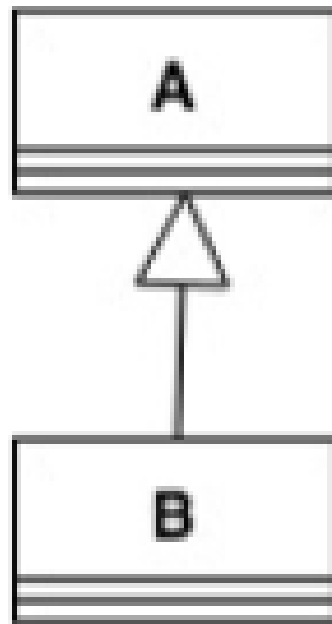
Sebenarnya yang terjadi disana adalah class Manajer merupakan perluasan dari class Pegawai dengan tambahan data member departemen.

Disini perlu memakai konsep inheritance, sehingga class Manajer dapat kita tuliskan seperti berikut

```
public class Manajer extends Pegawai {  
    public String departemen;  
}
```

# SINGLE INHERITANCE

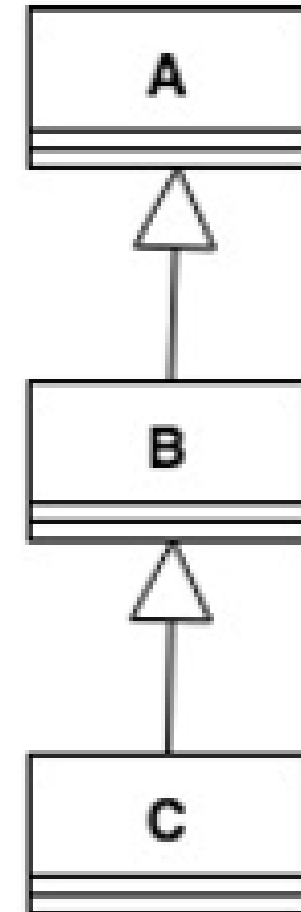
- Konsep inheritance yang ada di Java adalah Java hanya memperkenalkan adanya **single inheritance**.
- Konsep single inheritance hanya memperbolehkan suatu subclass mempunyai satu parent class.



# MULTILEVEL INHERITANCE

Konsep inheritance yang ada di Java memperkenalkan adanya **multilevel inheritance**.

Konsep multilevel inheritance memperbolehkan suatu subclass mempunyai subclass lagi.



# KONTROL PENGAKSESAN

Dalam dunia riil, suatu entitas induk bisa saja tidak mewariskan sebagian dari apa-apa yang ia punyai kepada entitas turunan karena sesuatu hal.

Demikian juga dengan konsep inheritance dalam OOP.

Suatu parent class dapat tidak mewariskan sebagian member-nya kepada subclass-nya.

Sebagai contoh, kita coba untuk memodifikasi class Pegawai.

```
public class Pegawai {  
    private String nama;  
    public double gaji;  
}
```

Coba untuk mengkompilasi class *Manajer* pada contoh sebelumnya.

Apa yang terjadi?

Pesan kesalahan akan muncul seperti ini :

```
Manajer.java:5: nama has private access in Pegawai
        nama=n;
```

Ini membuktikan bahwa class *Manajer* tidak mewarisi data member *nama* dari parent class-nya (*Pegawai*).

# KATA KUNCI SUPER

Kata kunci super dipakai untuk merujuk pada member dari parent class.

Sebagaimana kata kunci this yang dipakai untuk merujuk pada member dari class itu sendiri.

Format penulisannya adalah sebagai berikut :

- **super.data\_member**  
→ merujuk pada data member pada parent class
- **super.function\_member()**  
→ merujuk pada function member pada parent class
- **super()**  
→ merujuk pada konstruktor pada parent class

# CONTOH

```
class Parent {  
    public int x = 5;  
}  
  
class Child extends Parent {  
    public int x = 10;  
  
    public void Info(int x) {  
        System.out.println("Nilai x sebagai parameter = " + x);  
        System.out.println("Data member x di class Child = " + this.x);  
        System.out.println("Data member x di class Parent = " + super.x);  
    }  
}  
  
public class NilaiX {  
    public static void main(String args[]) {  
        Child tes = new Child();  
        tes.Info(20);  
    }  
}
```



# HASIL

Nilai x sebagai parameter = 20

Data member x di class Child = 10

Data member x di class Parent = 5

# KESIMPULAN

**x**

→ merujuk pada x terdekat, yaitu parameter Info()

**this.x**

→ merujuk pada data member dari class-nya sendiri, yaitu data member pada class Child

**super.x**

→ merujuk pada data member dari parent class-nya, yaitu data member pada class Parent

# KONSTRUKTOR TIDAK DIWARISKAN

Konstruktor dari parent class tidak dapat diwariskan ke subclass-nya.

Konsekuensinya, setiap kali kita membuat suatu subclass, maka kita harus memanggil konstruktor parent class di konstruktor subclass.

Pemanggilan konstruktor parent harus dilakukan pada baris pertama dari konstruktor subclass.

# KONSTRUKTOR TIDAK DIWARISKAN

Sebelum subclass menjalankan konstruktornya sendiri, subclass akan menjalankan constructor superclass terlebih dahulu.

Hal ini terjadi karena secara implisit pada constructor subclass ditambahkan pemanggilan `super()` yang bertujuan memanggil constructor superclass oleh kompiler.

MISALNYA SAJA KITA MEMPUNYAI DUA BUAH CLASS SEBAGAI BERIKUT :

```
public class Parent  
{  
  
}
```

```
public class Child extends Parent {  
  
}
```




Pada saat program tersebut dikompilasi, maka kompiler Java akan menambahkan :

- konstruktor class Parent
- konstruktor class Child
- pemanggilan konstruktor class Parent di konstruktor class Child

SEHINGGA PROGRAM TERSEBUT SAMA SAJA DENGAN YANG BERIKUT INI :

```
public class Parent {  
    public Parent() {  
  
    }  
}
```

```
public class Child extends Parent {  
    public Child() {  
        super();  
    }  
}
```



pemanggilan kostruktor class Parent

```
public class Child extends Parent {  
    int x;  
    public Child() {  
        x = 5;  
        super();  
    }  
}
```

X

```
public class Child extends Parent {  
    int x;  
    public Child() {  
        super();  
        x = 5;  
    }  
}
```

✓



# SOAL

1. Tuliskan program yang berfungsi untuk menghitung luas dan keliling bangun datar.

