BAB 6
Inheritance

Tujuan

1. Praktikan dapat memahami sub konsep sub class dan super class dalam konsep Pewarisan
2. Praktikan mampu menerapkan konsep encapsulation dalam Pewarisan

Ringkasan Materi

A. Konsep Dasar Inheritance

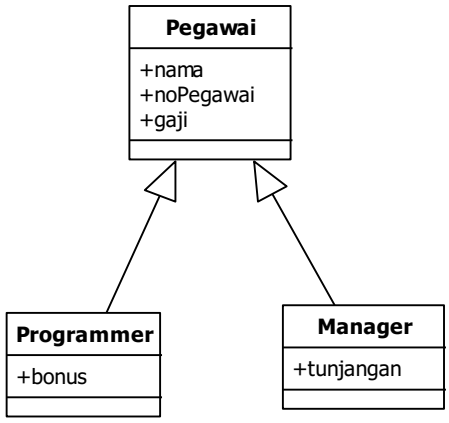
Inheritance (Pewarisan) merupakan salah satu dari tiga konsep dasar OOP. Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan. Dengan konsep inheritance, sebuah class dapat mempunyai class turunan.

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

Karena suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class-nya, maka member dari suatu subclass adalah terdiri dari apa-apa yang ia punyai dan juga apa-apa yang ia warisi dari class parent-nya.
Kesimpulannya, boleh dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parent class-nya.

Diagram UML

Adalah sebuah Bahasa yang berdasarkan grafik/gambar untuk memvisualisasi, menspesifikasikan, membangun dan mendokumentasikan dari sebuah system pengembangan software OOP (Object Oriented Programming).
Contoh Inheritance pada diagram UML :



B. 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 akan melakukan perluasan class, contoh deklarasi :

```
public class Programmer extends Pegawai { … }
public class Manager extends Pegawai { … }
```

Pelaksanaan Percobaan

Employee.java

```
1  import java.util.*;
2  public class Employee {
3      private String name;
4      private double salary;
5      private Date hireday;
6      public Employee(String name, double salary, int year,int
7  month, int day){
8          this.name = name;
9          this.salary = salary;
10         GregorianCalendar calendar = new
11 GregorianCalendar(year,month-1,day);
12         this.hireday = calendar.getTime();
13     }
14     public Date getHireDay(){
15         return hireday;
16     }
18     public String getName(){
19         return name;
20     }
21     public double getSalary(){
22         return salary;
23     }
24     public void raiseSalary(double byPercent){
25         double raise = salary * byPercent/100;
26         salary+=raise;
27     }
28 }
```

Ketikkan SubClass di bawah ini

Manager.java

```
1  public class Manager extends Employee {
2      private double bonus;
3      public Manager(String name, double salary, int year, int
4  month, int day){
5          super(name, salary, year, month, day);
6          bonus = 0;
7      }
8      public void setBonus(double bonus){
9          this.bonus = bonus;
10     }
11     public double getSalary(){
12         double baseSalary = super.getSalary();
13         return baseSalary+bonus;
14     }
15
16 }
```

Main Class

Employee.java

```
1  public class MainEmployee {
2      public static void main(String[] args) {
3          Manager boss = new Manager("Steven", 80000, 1987, 12,
4  15);
5          boss.setBonus(5000);
6          Employee staff = new Employee("Donni", 50000, 1989, 10,
7  1);
8          System.out.println("nama boss : "+boss.getName()+",
9  salary = "+boss.getSalary());
10         System.out.println("nama staff : "+staff.getName()+",
11 salary = "+staff.getSalary());
12     }
13 }
```

Data dan Analisis hasil percobaan

Pertanyaan

1. Jalankan code program diatas dan benahi jika menemukan kesalahan!
   ……
   ……

2. Bagaimana cara konstruktor pada subclass memanggil konstruktor di superclass nya? Apakah hal itu perlu dilakukan? Sertakan alasan anda !
   ……
   ……

3. Tambahkan constructor pada class Employee dengan parameter *String name*! amati perubahan apa yang terjadi, jelaskan jawaban anda!
   ……
   ……

4. Pada Class Manager baris ke 5, setelah variable day tambahkan variable bonus! Amati apa yang terjadi dan mengapa demikian?
   ……
   ……

5. Untuk apa digunakan keyword this pada class manager dan employee? Hapus keyword this dan amati apa yang terjadi?
   ……
   ……

6. Tambahkan constructor pada class Employee dengan parameter Bertipe data string bernama name yang nantinya bila constructor ini akan dipanggil akan menginisialisasi variable name! Amati perubahannya pada class anak dan jelaskan! Benahi bila terjadi kesalahan!
   ……
   ……

7. Pada bab sebelumnya anda telah belajar mengenai konsep encapsulation, jelaskan mengapa pada super class menggunakan modifier protected? Apa yang terjadi jika modifier anda ubah menjadi private atau public? Jelaskan !

……..
……..

8. Ubahlah acces modifier method pada kelas employee menjadi :
   a. Private
   b. Protected

   Amati perubahan apa yang terjadi? Jelaskan jawaban anda dengan detail!
   ……..
   ……..

Tugas Praktikum

Susunlah program sesuai studi kasus dibawah.

1. Pegawai.java

Buatlah class induk Pegawai dengan definisi sebagai berikut
- nama: String
- noIndukPegawai: String
- tahunMasuk: Date
- gajiPokok: double
- keluarga: boolean
- jumlahAnak: int

+ double getGajiPokok()
+ double getBonus()
+ double getTunjangan()
+ double getTotalGaji()

a. Bonus didapatkan dari lama kerja (tahun sekarang dikurangi tahun masuk)
  - Jika masa kerja dibawah 1 tahun tidak mendapat bonus
  - Jika masa kerja 1-5 tahun mendapat bonus 5% x gajiPokok x lama kerja
  - Jika masa kerja diatas 5 tahun mendapat bonus 10% x gajiPokok x lama kerja

b. Tunjangan didapatkan bila pegawai sudah berkeluarga
  - Jika sudah berkeluarga, mendapat tunjangan 5% dari gaji pokok
  - Jika memiliki anak mendapat tunjangan 5% dari gaji pokok per Anak yang dimiliki (maksimal 2 anak)

c. Total gaji diperoleh dari gaji pokok + bonus + tunjangan

2. Manager.java

Buatlah class Manager yang inherit class Pegawai dengan ketentuan Manager mendapat tunjangan jabatan sebesar 10% dari total gaji

3. Sales.java

Buatlah class Sales yang inherit class Pegawai dengan definisi:
- hargaBarang: double
- stockBarang: int
- barangTerjual: int

+ double getBonusTambahan()

Sales memiliki bonusTambahan dengan ketentuan jika barangTerjual lebih dari 70% stockBarang, maka mendapat bonusTambahan 10% dari hargaBarang x barangTerjual
Jika tidak, maka bonusTambahan yang diperoleh hanya 3% dari hargaBarang x barangTerjual

4. Programmer.java

Buatlah class Programmer yang inherit class Pegawai dengan definisi
- jamKerja: int

+ double getBonusLembur()
Freelancer akan mendapat bonusLembur jika jamKerja lebih dari 10 jam, sejumlah 5% x gajiPokok x jamKerja - 10 jam.

Test Case:
1. Pegawai tahunMasuk 2015, berkeluarga, 2 anak, gajiPokok 3jt
2. Pegawai tahunMasuk 2010, tidak berkeluarga, gajiPokok 3jt
3. Pegawai tahunMasuk 2005, berkeluarga, 5 anak, gajiPokok 3jt
4. Manager tahunMasuk 2009, tidak berkeluarga, gajiPokok 4jt
5. Manager tahunMasuk 2014, berkeluarga, 2 anak, gajiPokok 4jt
6. Sales tahunMasuk 2012, berkeluarga, 4 anak, hargaBarang 25rb, stockBarang 100, barangTerjual 50, gajiPokok 2jt
7. Sales tahunMasuk 2014, berkeluarga, 1 anak, hargaBarang 10rb, stockBarang 250, barangTerjual 220, gajiPokok 2jt
8. Freelancer tahunMasuk 2011, tidak berkeluarga, jamKerja 10 jam, gajiPokok 2,5jt
9. Freelancer tahunMasuk 2013, tidak berkeluarga, jamKerja 12 jam, gajiPokok 2,5jt

Tampilkan totalGaji berserta perhitungannya (gajipokok, bonus, tunjangan) dari setiap Test Case