BAB 7
POLIMORFISME

Tujuan

1. Memberikan pemahaman kepada mahasiswa tentang polimorfisme
2. Dapat membedakan perbedaan antara polymorfisme dan inheritance

Ringkasan Materi

A. Polymorfisme

Polimorfisme (memiliki banyak bentuk) menyatakan kemampuan untuk memperlakukan objek-objek dengan cara yang seragam meskipun objek-objek tersebut berbeda perilaku. Polimorfisme adalah suatu sifat yang merupakan efek langsung dari sifat inheritance pada OOP. Jika pada inheritance semua perilaku yang diturunkan kepada subclass memiliki bentuk yang sama, namun pada polimorfisme ini subclass dapat mendefinisikan perilaku yang sama dengan cara yang berbeda, dimana perilaku ini merupakan turunan dari super class.

Untuk bisa mengimplementasikan sifat polimorfisme, kita harus memakai method yang memungkinkan dibuat tanpa didefinisikan. Pada sub classnya nanti kita bisa mendefinisikan isi method tersebut sesuai tujuan masing-masing sub class. Dalam java kita dapat mendefinisikan method semacam ini dengan kata kunci **abstrak**, **dengan syarat kelas yang menampung method tersebut juga harus berupa Kelas abstrak**. **Kelas abstrak harus mengandung satu atau lebih method abstrak tapi boleh ada method selain abstrak**. Semua method abstrak yang ada di superclass harus dioverride di subclassnya

Cara pendeklarasian class abstract :

```
public abstract class <nama_kelas> {
   …
}
```

Cara pendeklarasian method abstract :

```
<modifier> <return_value> abstract <nama_method>;
```

Sebagai contoh, ada kelas MakhlukHidup sebagai kelas induk, yang mempunyai method tertentu misalnya *bernafas()*, *makan()*, *tidur()*, dan *berjalan()*. Kelas MakhlukHidup memiliki kelas turunan Manusia, Sapi, dan Kanguru. Dengan konsep inheritance, semua class turunan MakhlukHidup harus mengimpementasikan semua perilaku yang dimiliki kelas super tersebut. Meskipun subclass (Manusia, Sapi, dan Kanguru) bisa mengimplementasikan semua perilaku yang didefinisikan, namun sub class tersebut mengimplementasikannya dengan cara yang berbeda. Misalnya perilaku *berjalan()*, Manusia, Sapi dan Kanguru berjalan dengan cara yang berbeda. Jika Manusia berjalan dengan dua kaki, Sapi berjalan dengan empat kaki dan Kanguru berjalan dengan cara melompat dengan dua kaki. Implementasi contoh diatas adalah sebagai berikut.

```
public abstract class MakhlukHidup {
    public void bernafas(){
        System.out.println("adalah    Makhluk    hidup    yang    dapat
        bernafas");
    }
    public abstract void berjalan();
}
```

Untuk subclass yang mengimplementasikan kelas MakhlukHidup :

- Manusia

```
public class Manusia extends MakhlukHidup {
     public void berjalan(){
         System.out.println("Manusi berjalan dengan dua kaki");
```

```
    }
}
```

- Sapi

```
public class Sapi extends MakhlukHidup{
    public void berjalan(){
        System.out.println("Sapi berjalan dengan 4 kaki");
    }
}
```

- Kanguru

```
public class Kanguru {
    public void berjalan(){
        System.out.println("Kanguru berjalan dengan melompat pada 2
        kakinya");
    }
}
```

Pelaksanaan Percobaan

Polimorfisme

Ketikkan program di bawah ini. Buatlah class-class berikut dalam package yang sama.

Employee.java

```
1  public abstract class Employee {
2      private String name; private
3      String noKTP;
4      public Employee(String name, String noKTP){
5          this.name = name;
6          this.noKTP = noKTP;
7      }
8      public String getName(){
9          return name;
10     }
11     public String getNoKTP(){
12         return noKTP;
13     }
14     public String toString(){
15         return     String.format("     "+getName()+"\nNo.
16 :"+getNoKTP());                                                KTP
17     }
18     public abstract double earnings();//pendapatan
19 }
```

SalariedEmployee.java

```
1  public class SalariedEmployee extends Employee {
2      private double weeklySalary; //gaji/minggu
3      public SalariedEmployee(String name, String noKTP, double
   salary) {
4          super(name, noKTP);
5          setWeeklySalary(salary);
6      }
7      public void setWeeklySalary(double salary) {
8          weeklySalary = salary;
9      }
10     public double getWeeklySalary() {
11         return weeklySalary;
```

```
12      }
13      public double earnings() {
14          return getWeeklySalary();
15      }
16      public String toString() {
17          return    String.format("Salaried    employee:    "    +
18  super.toString() +
19                  "\nweekly salary:" + getWeeklySalary());
20      }
21  }
```

HourlyEmployee.java

```
1   public class HourlyEmployee extends Employee {
2       private double wage; //upah per jam
3       private double hours; //jumlah jam tiap minggu
4       public HourlyEmployee(String name, String noKTP,
                double hourlyWage, double hoursWorked) {
5           super(name, noKTP);
6           setWage(hourlyWage);
7           setHours(hoursWorked);
8       }
9       public void setWage(double hourlyWage){
10          wage = hourlyWage;
11      }
12      public double getWage(){
13          return wage;
14      }
15      public void setHours(double hoursWorked){
16          hours = hoursWorked;
17      }
18      public double getHours(){
19          return hours;
20      }
21      public double earnings(){
22          if(getHours()<=40)
23              return getWage() * getHours();
24          else
25              return 40 * getWage() + ( getHours()-40) * getWage()
    * 1.5;
26      }
27      public String toString(){
28          return          String.format("Hourly          employee:
    "+super.toString()
                    +"\nhourly    wage"+getWage()+"\nhours    worked:
    "+getHours());
29      }
30  }
```

Commission.java

```
1   public class CommissionEmployee extends Employee {
2       private double grossSales;//penjualan per minggu
3       private double commissionRate;//komisi
4       public CommissionEmployee(String name, String noKTP, double
```

```
5  sales, double rate){
6          super(name, noKTP);
7          setGrossSales(sales);
8          setCommissionRate(rate);
9      }
10     public void setGrossSales(double sales){
11         grossSales = sales;
12     }
13     public double getGrossSales(){
14         return grossSales;
15     }
16     public void setCommissionRate(double rate){
17         commissionRate = rate;
18     }
19     public double getCommissionRate(){
20         return commissionRate;
21     }
22     public double earnings(){
23         return getCommissionRate()*getGrossSales();
24     }
25     public String toString(){
26         return         String.format("Commision         employee:
27 "+super.toString()+"\ngross                                   sales:
28 "+getGrossSales()+"\ncommission rate"+getCommissionRate());
29     }
30 }
31
```

BasePlusCommissionEmployee.java

```
1  public       class       BasePlusCommissionEmployee       extends
2  CommissionEmployee {
3
4       private double baseSalary;//gaji pokok tiap minggu
5
6      public   BasePlusCommissionEmployee(String    name,    String
7  noKTP, double sales, double rate, double salary) {
8          super(name, noKTP, sales, rate);
9          setBaseSalary(salary);
10     }
11
12     public void setBaseSalary(double salary) {
13         baseSalary = salary;
14     }
15
16     public double getBaseSalary() {
17         return baseSalary;
18     }
19
20     public double earnings() {
21          return getBaseSalary() + super.earnings();
22     }
23
24     public String toString() {
```

```
25          return       String.format("Base-Salaried        "       +
26  super.toString() + "\nbase salary " + getBaseSalary());
27      }
28  }
```

Data dan Analisis hasil percobaan

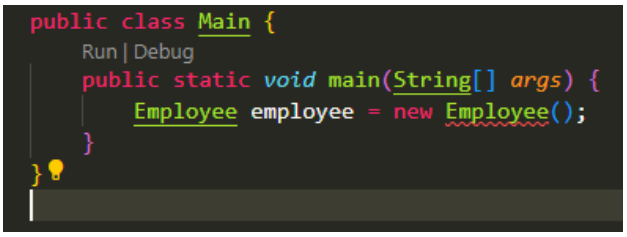
Pertanyaan

1. Ketikkan kode ini.

Main.java

```
1   public class Main {
2      public static void main(String[] args) {
3           Employee employee = new Employee();
4       }
5   }
```

Jalankan Main.java untuk polymorfisme Employee, analisis dan jelaskan keluaran program tersebut!



Hasilnya adalah error, hal ini dikarenakan kelas Employee merupakan kelas abstract, yang dimana ia digunakan sebagai kerangka kerja untuk kelas-kelas turunannya. Kelas abstract memiliki metode-metode abstrak (metode tanpa implementasi) yang harus diimplementasikan oleh kelas turunan. Sehingga kelas abstract tidak bisa diinstansiasi secara langsung.

2. Jalankan program dengan main sebagai berikut.

Main.java

```
1   public class Main {
2      public static void main(String[] args) {
3           SalariedEmployee       salariedEmployee       =       new
4   SalariedEmployee("Daniel", "135", 800.00);
5           HourlyEmployee         hourlyEmployee         =         new
6   HourlyEmployee("Karina", "234", 16.75, 40);
7           CommissionEmployee     commissionEmployee     =     new
8   CommissionEmployee("Keanu", "145", 10000, .06);
9             BasePlusCommissionEmployee basePlusCommissionEmployee =
10        new BasePlusCommissionEmployee("Bondan", "234", 5000, .04,
11  300);
12          System.out.println("Employees       diproses       secara
13  terpisah:\n");
14          System.out.printf("%s\n%s: $%,.2f\n\n",
15                  salariedEmployee,         "pendapatan:         ",
16  salariedEmployee.earnings());
17          System.out.printf("%s\n%s: $%,.2f\n\n",
18                  hourlyEmployee,          "pendapatan:          ",
19  hourlyEmployee.earnings());
20          System.out.printf("%s\n%s: $%,.2f\n\n",
21                  commissionEmployee,        "pendapatan:        ",
22  commissionEmployee.earnings());
23          System.out.printf("%s\n%s: $%,.2f\n\n",
24                  basePlusCommissionEmployee,
25                  "earned",
```

```
26  basePlusCommissionEmployee.earnings());
27
28          Employee[] employees = new Employee[4];
29          employees[0] = salariedEmployee;
30          employees[1] = hourlyEmployee;
31          employees[2] = commissionEmployee;
```

```
32          employees[3] = basePlusCommissionEmployee;
33          System.out.println("Employees       diproses       secara
34  polimorfisme:\n");
35          for (Employee currentEmployee : employees) {
36              System.out.println(currentEmployee);
37              if            (currentEmployee            instanceof
38  BasePlusCommissionEmployee) {
39                  BasePlusCommissionEmployee       employee       =
40  (BasePlusCommissionEmployee) currentEmployee;
41                  employee.setBaseSalary(1.10                      *
42  employee.getBaseSalary());
43                  System.out.printf(
44                          "Gaji pokok setelah dinaikkan 10%% :
45  $%,.2f\n",
46                          employee.getBaseSalary());
47              }
48              System.out.printf("pendapatan:          $%,.2f\n\n",
49  currentEmployee.earnings());
50          }
51          for (int j = 0; j < employees.length; j++) {
52              System.out.printf("Employee    %d    =    %s\n",    j,
53  employees[j].getClass().getName());
54          }
55      }
    }
```

Analisis dan jelaskan output program (berdasarkan konsep polimorfisme)!

```
Employees       diproses secara terpisah:

Salaried        employee:               Daniel
No. KTP:135
weekly salary:800.0
pendapatan:     : $800.00

Hourly  employee:       Karina
No. KTP:234
hourly  wage16.75
hours   worked:40.0
pendapatan:     : $670.00

Commision employee:     Keanu
No. KTP:145
gross   sales: 10000.0
commission rate0.06
pendapatan:     : $600.00

Base-Salaried   Commision employee:     Bondan
No. KTP:234
gross   sales: 5000.0
commission rate0.04
base salary 300.0
earned: $500.00

Employees       diproses secara polimorfisme:

Salaried        employee:               Daniel
No. KTP:135
weekly salary:800.0
pendapatan: $800.00

Hourly  employee:       Karina
No. KTP:234
hourly  wage16.75
hours   worked:40.0
```

```
Hourly  employee:       Karina
No. KTP:234
hourly  wage16.75
hours   worked:40.0
pendapatan: $670.00

Commision employee:     Keanu
No. KTP:145
gross   sales: 10000.0
commission rate0.06
pendapatan: $600.00

Base-Salaried   Commision employee:     Bondan
No. KTP:234
gross   sales: 5000.0
commission rate0.04
base salary 300.0
Gaji pokok setelah dinaikkan 10% : $330.00
pendapatan: $530.00

Employee 0 = SalariedEmployee
Employee 1 = HourlyEmployee
Employee 2 = CommissionEmployee
Employee 3 = BasePlusCommissionEmployee
PS C:\Users\Joe\Desktop\PBOSem2\Tugasqu\Tugas 7>
```

Output dari program di atas adalah data nama, nomor KTP, gaji pokok, jumlah jam kerja, dan pendapatan karyawan. Data dari setiap objek juga diproses secara polimorfisme. Data yang dicetak sama seperti pada pemrosesan secara terpisah, tetapi juga termasuk data gaji pokok setelah dinaikkan 10% bagi objek BasePlusCommissionEmployee. Terakhir, terdapat juga output yang menunjukkan jenis dari setiap objek yang disimpan dalam array employees. Jenis dari setiap objek ini dicetak dengan menggunakan getClass().getName().

3. Buat objek dari method Employee? Jelaskan hasil dari output program tersebut!

```
Employee employee = new Employee() {
    @Override
    public double earnings(){
        return 0;
    }
}
```

Yang terjadi adalah error, hal ini dikarenakan tidak dapat membuat objek method yang merupakan bagian dari sebuah kelas yang harus memiliki objek terlebih dahulu sebelum dilakukan pemanggilan.

4. Tambahkan atribut tanggal lahir di Kelas Employee, serta tambahkan method pendukungnya (accesor dan mutator). Modifikasi program agar sesuai. Asumsikan gaji yang diterima adalah per bulan, buat kelas uji untuk menguji program yang sudah anda modifikasi, kemudian buat objek dari semua class (salariedEmployee, hourlyEmployee, commissionEmployee, basePlusCommissionEmployee dan hitung gajinya secara polimorfisme, serta tambahkan gajinya sebesar 100.000 jika bulan ini adalah bulan ulang tahunnya.

```java
public abstract class Employee {
    private String name;
    private String noKTP;
    private String tanggalLahir;

    public Employee(String name, String noKTP, String tanggalLahir){
        this.name = name;
        this.noKTP = noKTP;
        this.tanggalLahir = tanggalLahir;
    }

    public String getName(){
        return name;
    }

    public String getNoKTP(){
        return noKTP;
    }

    public String toString(){
        return  String.format(" " + getName() + "\nNo. KTP:" + getNoKTP());
    }

    public String getTanggalLahir() {
        return tanggalLahir;
    }

    public void setTanggalLahir(String tanggalLahir) {
        this.tanggalLahir = tanggalLahir;
    }
```

```java
ublic class SalariedEmployee extends Employee {
    private double weeklySalary; // gaji/minggu

    public SalariedEmployee(String name, String noKTP, String tanggalLahir, double salary) {
        super(name, noKTP, tanggalLahir);
        setWeeklySalary(salary);
    }

    public void setWeeklySalary(double salary) {
        weeklySalary = salary < 0.0 ? 0.0 : salary;
    }

    public double getWeeklySalary() {
        return weeklySalary;

    }

    public double earnings() {
        return getWeeklySalary() * 4;
    }

    public String toString() {
        return String.format("Salaried  employee:   " + super.toString() + "\nTanggal lahir: " + getTanggalLahir() + "\nweekly salary:"
    }
```

```java
public class HourlyEmployee extends Employee {
    private double wage; // upah per jam
    private double hours; // jumlah jam tiap minggu

    public HourlyEmployee(String name, String noKTP, String tanggalLahir, double hourlyWage, double hoursWorked){
        super(name, noKTP, tanggalLahir);
        setWage(hourlyWage);
        setHours(hoursWorked);
        setTanggalLahir(tanggalLahir);
    }

    public void setWage(double hourlyWage) {
        wage = hourlyWage;
    }

    public double getWage() {
        return wage;
    }

    public void setHours(double hoursWorked) {
        hours = hoursWorked;
    }

    public double getHours() {
        return hours;
    }

    public double earnings() {
        if (getHours() <= 40)
            return getWage() * getHours();
```

```java
    public double getHours() {
        return hours;
    }

    public double earnings() {
        if (getHours() <= 40)
            return getWage() * getHours();
        else
            return 40 * getWage() + (getHours() - 40) * getWage() * 1.5;
    }

    public String toString() {
        return String.format("Hourly employee: " + super.toString() + "\nTanggal lahir: " + getTanggalLahir() + "\nhourly   wage" + get
    }
}
```

```java
public class CommissionEmployee extends Employee {
    private double grossSales;// penjualan per minggu
    private double commissionRate; //komisi

    public CommissionEmployee(String name, String noKTP, String tanggalLahir, double sales, double rate) {
        super(name, noKTP, tanggalLahir);
        setGrossSales(sales);
        setCommissionRate(rate);
    }

    public void setGrossSales(double sales) {
        grossSales = sales;
    }

    public double getGrossSales() {
        return grossSales;
    }

    public void setCommissionRate(double rate) {
        commissionRate = rate;
    }

    public double getCommissionRate() {
        return commissionRate;
    }

    public double earnings() {
        return getCommissionRate() * getGrossSales();
    }

    public String toString() {
```

```java
public String toString() {
    return String.format("Commision employee: " + super.toString() + "\nTanggal lahir: " + getTanggalLahir() +"\ngross  s
}
```

```java
public class BasePlusCommissionEmployee extends CommissionEmployee {

    private double baseSalary;// gaji pokok tiap minggu

    public BasePlusCommissionEmployee(String name, String noKTP, String tanggalLahir, double sales, double rate, double salary) {
        super(name, noKTP, tanggalLahir, sales, rate);
        setBaseSalary(salary);
        setTanggalLahir(tanggalLahir);
    }

    public void setBaseSalary(double salary) {
        baseSalary = salary;
    }

    public double getBaseSalary() {
        return baseSalary;
    }

    public double earnings() {
        return getBaseSalary() + super.earnings();
    }

    public String toString() {
        return String.format("Base-Salaried " + super.toString() +  "\nbase salary " + getBaseSalary());
    }
}
```

```
public class Main {
    Run | Debug
    public static void main(String[] args) {

        SalariedEmployee salariedEmployee = new SalariedEmployee(name:"Daniel", noKTP:"135", tanggalLahir:"25 April 2005", salary:800.0
        HourlyEmployee hourlyEmployee = new HourlyEmployee(name:"Karina", noKTP:"234", tanggalLahir:"5 April 2005", hourlyWage:16.75, h
        CommissionEmployee commissionEmployee = new CommissionEmployee(name:"Keanu", noKTP:"145", tanggalLahir:"10 Juli 2002", sales:10
        BasePlusCommissionEmployee basePlusCommissionEmployee = new BasePlusCommissionEmployee(name:"Bondan", noKTP:"234",tanggalLahir:

        System.out.println(x:"Employees diproses secara terpisah:\n");
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", salariedEmployee, "pendapatan:   ", salariedEmployee.earnings());
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", hourlyEmployee, "pendapatan: ", hourlyEmployee.earnings());
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", commissionEmployee, "pendapatan: ", commissionEmployee.earnings());
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", basePlusCommissionEmployee, "earned", basePlusCommissionEmployee.earnings());

        Employee[] employees = new Employee[4];
        employees[0] = salariedEmployee;
        employees[1] = hourlyEmployee;
        employees[2] = commissionEmployee;
        employees[3] = basePlusCommissionEmployee;

        System.out.println(x:"Employees diproses secara polimorfisme:\n");
        for (Employee currentEmployee : employees) {
            System.out.println(currentEmployee);
            if (currentEmployee instanceof BasePlusCommissionEmployee) {
                BasePlusCommissionEmployee employee = (BasePlusCommissionEmployee) currentEmployee;
                employee.setBaseSalary(1.10 * employee.getBaseSalary());
                System.out.printf(format:"Gaji pokok setelah dinaikkan 10%% : $%,.2f\n", employee.getBaseSalary());
            }
            System.out.printf(format:"pendapatan: $%,.2f\n\n", currentEmployee.earnings());
        }
                }
                System.out.printf(format:"pendapatan: $%,.2f\n\n", currentEmployee.earnings());
            }
            for (int j = 0; j < employees.length; j++) {
                System.out.printf(format:"Employee %d = %s\n", j, employees[j].getClass().getName());
            }
        }
    }
```

Output:

```
Employees diproses secara terpisah:

Salaried        employee:               Daniel
No. KTP:135
Tanggal lahir: 25 April 2005
weekly salary:800.0
pendapatan:     : $3,200.00

Hourly employee:        Karina
No. KTP:234
Tanggal lahir: 5 April 2005
hourly  wage16.75
hours   worked:40.0
pendapatan:     : $670.00

Commision employee:     Keanu
No. KTP:145
Tanggal lahir: 10 Juli 2002
gross   sales: 10000.0
commission rate0.06
pendapatan:     : $600.00

Base-Salaried   Commision employee:     Bondan
No. KTP:234
Tanggal lahir: 3 Maret 1999
gross   sales: 5000.0
commission rate0.04
base salary 300.0
```

```
Base-Salaried   Commision employee:     Bondan
No. KTP:234
Tanggal lahir: 3 Maret 1999
gross   sales: 5000.0
commission rate0.04
base salary 300.0
earned: $500.00

Employees       diproses secara polimorfisme:

Salaried        employee:               Daniel
No. KTP:135
Tanggal lahir: 25 April 2005
weekly salary:800.0
pendapatan: $3,200.00

Hourly employee:        Karina
No. KTP:234
Tanggal lahir: 5 April 2005
hourly  wage16.75
hours   worked:40.0
pendapatan: $670.00

Commision employee:     Keanu
No. KTP:145
Tanggal lahir: 10 Juli 2002
gross   sales: 10000.0
commission rate0.06
```

```
Commision employee:     Keanu
No. KTP:145
Tanggal lahir: 10 Juli 2002
gross   sales: 10000.0
commission rate0.06
pendapatan: $600.00

Base-Salaried   Commision employee:     Bondan
No. KTP:234
Tanggal lahir: 3 Maret 1999
gross   sales: 5000.0
commission rate0.04
base salary 300.0
Gaji pokok setelah dinaikkan 10% : $330.00
pendapatan: $530.00

Employee 0 = SalariedEmployee
Employee 1 = HourlyEmployee
Employee 2 = CommissionEmployee
Employee 3 = BasePlusCommissionEmployee
PS C:\Users\Joe\Desktop\PBOSem2\Tugasqu\Tugas 7>
```

5. Perusahaan yang mengaplikasikan program polimorfisme diatas ingin menambahkan kriteria baru untuk penggajian karyawannya, yaitu penggajian berdasarkan banyaknya barang yang diproduksi. Dengan ketentuan gaji karyawan tersebut adalah hasil dari banyaknya barang yang dipoduksi per minggu dikalikan upah per barangnya.
   a. Analisis dan jelaskan proses modifikasi program diatas (dimulai dari pemilihan jenis class, perancangan class, dan penempatan class)
   b. Implementasi hasil analisis tersebut ke dalam program dan buat kelas uji dengan minimal 4 objek yang dibentuk.

```java
public class PieceWorker extends Employee{
    private double wagePerPiece; // upah per barang
    private int quantityProduced; // jumlah barang yang diproduksi

    public PieceWorker(String name, String noKTP, String tanggalLahir, double wagePerPiece, int quantityProduced
        super(name, noKTP, tanggalLahir);
        setWagePerPiece(wagePerPiece);
        setQuantityProduced(quantityProduced);
    }

    public void setWagePerPiece(double wagePerPiece) {
        this.wagePerPiece = wagePerPiece;
    }

    public double getWagePerPiece() {
        return wagePerPiece;
    }

    public void setQuantityProduced(int quantityProduced) {
        this.quantityProduced = quantityProduced;
    }

    public int getQuantityProduced() {
        return quantityProduced;
    }

    public double earnings() {
```

```java
public void setQuantityProduced(int quantityProduced) {
    this.quantityProduced = quantityProduced;
}

public int getQuantityProduced() {
    return quantityProduced;
}

public double earnings() {
    return getWagePerPiece() * getQuantityProduced();
}

public String toString() {
    return String.format("PieceWorker: " + super.toString() + "\nTanggal lahir: " + getTanggalLahir() + "\nWage p
}
```

```java
+ "\nWage per piece: " + getWagePerPiece() + "\nQuantity produced: " + getQuantityProduced())
```

```java
public class Main {
    Run | Debug
    public static void main(String[] args) {

        SalariedEmployee salariedEmployee = new SalariedEmployee(name:"Daniel", noKTP:"135", tanggalLahir:"25 April 2005", salary:800.0
        HourlyEmployee hourlyEmployee = new HourlyEmployee(name:"Karina", noKTP:"234", tanggalLahir:"5 April 2005", hourlyWage:16.75, h
        CommissionEmployee commissionEmployee = new CommissionEmployee(name:"Keanu", noKTP:"145", tanggalLahir:"10 Juli 2002", sales:10
        BasePlusCommissionEmployee basePlusCommissionEmployee = new BasePlusCommissionEmployee(name:"Bondan", noKTP:"234",tanggalLahir:
        PieceWorker pieceWorker = new PieceWorker(name:"Sarah", noKTP:"456", tanggalLahir:"15 Juni 2000", wagePerPiece:2.5, quanti…1000
        PieceWorker pieceWorker2 = new PieceWorker(name:"Gala", noKTP:"247", tanggalLahir:"18 Mei 2001", wagePerPiece:2.5, quanti…1000)
        PieceWorker pieceWorker3 = new PieceWorker(name:"Fuji", noKTP:"902", tanggalLahir:"20 April 202", wagePerPiece:2.5, quanti…1000
        PieceWorker pieceWorker4 = new PieceWorker(name:"Faiz", noKTP:"572", tanggalLahir:"10 Januari 2005", wagePerPiece:2.5, quanti…1

        System.out.println(x:"Employees diproses secara terpisah:\n");
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", salariedEmployee, "pendapatan:   ", salariedEmployee.earnings());
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", hourlyEmployee, "pendapatan: ", hourlyEmployee.earnings());
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", commissionEmployee, "pendapatan: ", commissionEmployee.earnings());
        System.out.printf(format:"%s\n%s: $%,.2f\n\n", basePlusCommissionEmployee, "earned", basePlusCommissionEmployee.earnings());

        Employee[] employees = new Employee[8];
        employees[0] = salariedEmployee;
        employees[1] = hourlyEmployee;
        employees[2] = commissionEmployee;
        employees[3] = basePlusCommissionEmployee;
        employees[4] = pieceWorker;
        employees[5] = pieceWorker2;
        employees[6] = pieceWorker3;
        employees[7] = pieceWorker4;
```

```
        System.out.println(x:"Employees diproses secara polimorfisme:\n");
        for (Employee currentEmployee : employees) {
            System.out.println(currentEmployee);
            if (currentEmployee instanceof BasePlusCommissionEmployee) {
                BasePlusCommissionEmployee employee = (BasePlusCommissionEmployee) currentEmployee;
                employee.setBaseSalary(1.10 * employee.getBaseSalary());
                System.out.printf(format:"Gaji pokok setelah dinaikkan 10%% : $%,.2f\n", employee.getBaseSalary());
            }
            System.out.printf(format:"pendapatan: $%,.2f\n\n", currentEmployee.earnings());
        }
        for (int j = 0; j < employees.length; j++) {
            System.out.printf(format:"Employee %d = %s\n", j, employees[j].getClass().getName());
        }

    }
}
```

Output:

```
PieceWorker:    Sarah
No. KTP:456
Tanggal lahir: 15 Juni 2000
Wage per piece: 2.5
Quantity produced: 1000
pendapatan: $2,500.00

PieceWorker:    Gala
No. KTP:247
Tanggal lahir: 18 Mei 2001
Wage per piece: 2.5
Quantity produced: 1000
pendapatan: $2,500.00

PieceWorker:    Fuji
No. KTP:902
Tanggal lahir: 20 April 202
Wage per piece: 2.5
Quantity produced: 1000
pendapatan: $2,500.00

PieceWorker:    Faiz
No. KTP:572
Tanggal lahir: 10 Januari 2005
Wage per piece: 2.5
Quantity produced: 1000
pendapatan: $2,500.00
```

Karena perusahaan ingin menambahkan karyawan yang dibayar berdasarkan jumlah barang yang diproduksi, maka diperlukan kelas yang merepresentasikan jenis karyawan ini. Karena karyawan ini memiliki karakteristik khusus yang berbeda dari karyawan lainnya (misalnya, tidak memiliki gaji tetap atau upah per jam), maka pendekatan yang tepat adalah membuat kelas yang terpisah untuk jenis karyawan ini.

Atribut utama dari karyawan jenis ini adalah upah per barang dan jumlah barang yang diproduksi. Metode utama yang dibutuhkan adalah untuk menghitung pendapatan karyawan berdasarkan produksi barang. Karena karyawan ini masih merupakan bagian dari karyawan secara umum, maka kelas PieceWorker harus mewarisi sifat-sifat dasar dari kelas Employee.

Kelas PieceWorker harus ditempatkan di dalam paket yang sama dengan kelas-kelas karyawan lainnya agar mudah diakses dan digunakan bersama. Kelas PieceWorker ditempatkan di bawah kelas Employee, menunjukkan bahwa PieceWorker adalah turunan dari karyawan secara umum.

Tugas Praktikum

I. Buatlah sebuah klas **abstract** Kue yang memiliki attribut dan method sebagai berikut
- \- nama : String
- \- harga : double
- \+ *hitungHarga()***** : double
- \+ toString : String *(menampilkan nama kue dan harga)*

**** abstract**

II. Buatlah 2 subklas dari klas Kue yaitu
   a. KuePesanan
      - \- berat : double
      - \+ hitungHarga() : double
        Hitung harga berdasarkan harga x berat

   b. KueJadi
      - \- jumlah : double
      - \+ hitungHarga() : double
        Hitung harga berdasarkan harga x jumlah x 2

III. Berdasarkan 2 kelas tersebut, buatlah :
   1. Array yang terdiri dari 20 kue
   2. Isikan 20 objek kue dengan berbagai jenis kue (KuePesanan atau KueJadi)
   3. Dari array tersebut :
      a. Tampilkan semua kue dan harus ditampilkan jenis kuenya
      b. Hitung total harga yang didapat dari semua jenis kue
      c. Hitung total harga dan total berat dari KuePesanan
      d. Hitung total harga dan total jumlah dari KueJadi
      e. Tampilkan informasi kue dengan harga (harga akhir) terbesar