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){this.name =
5          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. KTP
16 :"+getNoKTP());
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

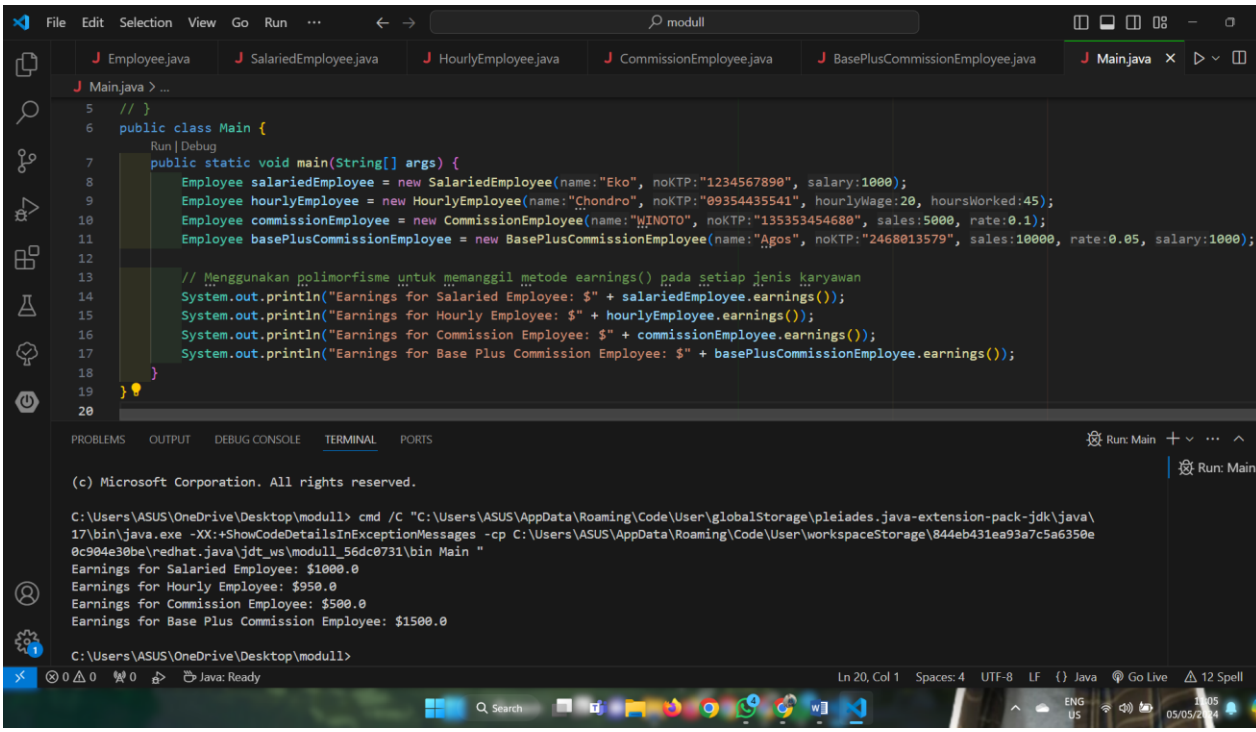
```
1  public class Main {
2     public static void main(String[] args) {
3          Employee employee = new Employee();
4      }
5  }
```

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

Kode diatas salah pada bagian Employee, karena employee merupakan kelas dan tidak dapat diinstansiasi.

Jadi perubahan yang saya lakukan adalah membuat instansi dari kelas turunannya, seperti SalariedEmployee, HourlyEmployee, CommissionEmployee, atau BasePlusCommissionEmployee :

nnya



Hasil keluarannnya akan mencetak pendapatan yang dihasilkan oleh setiap jenis karyawan yang telah dibuat.

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)!

Output dari pemrosesan setiap jenis karyawan secara terpisah:

```
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
```

Untuk yang secara terpisah, Setiap jenis karyawan diproses dengan informasi yang sesuai: nama, nomor KTP, dan detail gaji mereka masing-masing. Kemudian, pendapatan mereka dihitung sesuai dengan logika bisnis yang telah diimplementasikan di setiap kelas turunan.

Output ketika karyawan diproses secara polimorfisme:

```
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
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
```

Nah pada output bagian ini, bisa kita lihat output untuk setiap karyawan seperti sebelumnya. Namun, pada saat diproses secara polimorfisme, kita menggunakan array dari kelas induk Employee untuk menyimpan objek dari kelas turunannya.

Ketika looping melalui array tersebut, metode toString() dari masing-masing objek dipanggil secara dinamis sesuai dengan jenis objeknya. Ini menunjukkan konsep polimorfisme di mana metode yang sesuai dipanggil tergantung pada tipe aktual objek yang ditunjuk oleh referensi Employee.

Ada juga peningkatan gaji pokok sebesar 10% hanya untuk BasePlusCommissionEmployee. Hal ini dilakukan dengan memeriksa apakah karyawan saat ini adalah instance dari BasePlusCommissionEmployee menggunakan operator instanceof. Jika ya, maka objek tersebut dicasting menjadi BasePlusCommissionEmployee dan gaji pokoknya ditingkatkan sebelum pendapatan keseluruhan dihitung.

Pada bagian akhir outputan menunjukkan jenis setiap objek dalam array.

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

```
// Metode statis untuk membuat objek Employee
public static Employee createEmployee(String type, String name, String noKTP, double param1, double param2, double param3) {
    switch(type) {
        case "Salaried":
            return new SalariedEmployee(name, noKTP, param1);
        case "Hourly":
            return new HourlyEmployee(name, noKTP, param1, param2);
        case "Commission":
            return new CommissionEmployee(name, noKTP, param1, param2);
        case "BasePlusCommission":
            return new BasePlusCommissionEmployee(name, noKTP, param1, param2, param3);
        default:
            throw new IllegalArgumentException(s:"Invalid employee type");
    }
}
}
```

Pada kelas Main

```
// Membuat objek Employee dengan metode statis createEmployee
Employee salariedEmployee = Employee.createEmployee(type:"Salaried", name:"Daniel", noKTP:"135", param1:800.00, param2:0, param3:0);
Employee hourlyEmployee = Employee.createEmployee(type:"Hourly", name:"Karina", noKTP:"234", param1:16.75, param2:40, param3:0);
Employee commissionEmployee = Employee.createEmployee(type:"Commission", name:"Keanu", noKTP:"145", param1:10000, param2:0.06, param3:0);
Employee basePlusCommissionEmployee = Employee.createEmployee(type:"BasePlusCommission", name:"Bondan", noKTP:"234", param1:5000, param2:0.04, param
```

Output

```
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
```

Hasilnya sama seperti sebelumnya, namun penambahan metode createEmployee hanya membuat proses pembuatan objek menjadi lebih terorganisir dan fleksibel.

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.

   Modifikasi kelas Employee dengan menambahkan atribut tanggal lahir beserta accessor dan mutator :

```java
import java.time.LocalDate;
public abstract class Employee {
    private String name;
    private String noKTP;
    private LocalDate birthDate; // Tambahkan atribut tanggal lahir

    public Employee(String name, String noKTP, LocalDate birDate){
    this.name = name;
    this.noKTP = noKTP;
    this.birthDate = birthDate;
    }
    // Getter dan setter untuk tanggal lahir
    public LocalDate getBirthDate() {
        return birthDate;
    }

    public void setBirthDate(LocalDate birthDate) {
        this.birthDate = birthDate;
    }

    // Metode untuk menghitung gaji per bulan
    public abstract double monthlyEarnings(); // Gaji per bulan

```

Membuat kelas TestEmployee untuk menguji program yang sudah dimodifikasi:

```java
import java.time.LocalDate;

public class TestEmployee {
    Run | Debug
    public static void main(String[] args) {
        LocalDate today = LocalDate.now(); // Mendapatkan tanggal hari ini

        // Membuat objek dari semua kelas
        SalariedEmployee salariedEmployee = new SalariedEmployee(name:"Daniel", noKTP:"135", LocalDate.of(year:1990, month:5, dayOfMonth:15), salary:80
        HourlyEmployee hourlyEmployee = new HourlyEmployee(name:"Karina", noKTP:"234", LocalDate.of(year:1995, month:8, dayOfMonth:20), hourlyWage…16.7
        CommissionEmployee commissionEmployee = new CommissionEmployee(name:"Keanu", noKTP:"145", LocalDate.of(year:1988, month:10, dayOfMonth:10), sal
        BasePlusCommissionEmployee basePlusCommissionEmployee = new BasePlusCommissionEmployee(name:"Bondan", noKTP:"234", sales:10000, rate:0.04, sala


        // Hitung gaji secara polimorfisme
        Employee[] employees = {salariedEmployee, hourlyEmployee, commissionEmployee, basePlusCommissionEmployee};

        for (Employee employee : employees) {
            double earnings = employee.monthlyEarnings(); // Hitung gaji per bulan

            // Jika bulan ini adalah bulan ulang tahunnya, tambahkan gaji sebesar 100.000
            if (employee.getBirthDate().getMonth() == today.getMonth() && employee.getBirthDate().getDayOfMonth() == today.getDayOfMonth()) {
                earnings += 100000;
            }

            System.out.printf(format:"%s\n%s: $%,.2f\n\n",
                employee, "Earnings: ", earnings);
        }
    }
}
```

Dalam kelas uji ini,saya membuat objek dari semua kelas turunan Employee dan menghitung gaji per bulan mereka. Jika bulan ini adalah bulan ulang tahun karyawan, kita menambahkan gaji sebesar 100.000. Hal ini dilakukan dengan memeriksa apakah bulan dan hari ini sama dengan bulan dan hari ulang tahun karyawan.

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)

      Jawab!

      1. Pemilihan jenis class
      - **Kriteria baru untuk penggajian karyawan melibatkan perhitungan berdasarkan banyaknya barang yang diproduksi per minggu.**
      - **Diperlukan kelas baru yang mewakili jenis karyawan yang memproduksi barang.**
      - **Membuat kelas baru bernama PieceWorker, sebagai subkelas dari Employee.**

      2. Perancangan class
      - **Kelas PieceWorker memiliki atribut wagePerPiece untuk menyimpan upah per barang dan quantityProduced untuk menyimpan banyaknya barang yang diproduksi per minggu.**
      - **Implementasikan metode earnings() untuk menghitung pendapatan berdasarkan kriteria baru.**

      3. Penempatan calss
      - **Kelas PieceWorker memiliki atribut wagePerPiece untuk menyimpan upah per barang dan quantityProduced untuk menyimpan banyaknya barang yang diproduksi per minggu.**
      - **Implementasikan metode earnings() untuk menghitung pendapatan berdasarkan kriteria baru.**

   b. Implementasi hasil analisis tersebut ke dalam program dan buat kelas uji dengan minimal 4 objek yang dibentuk.

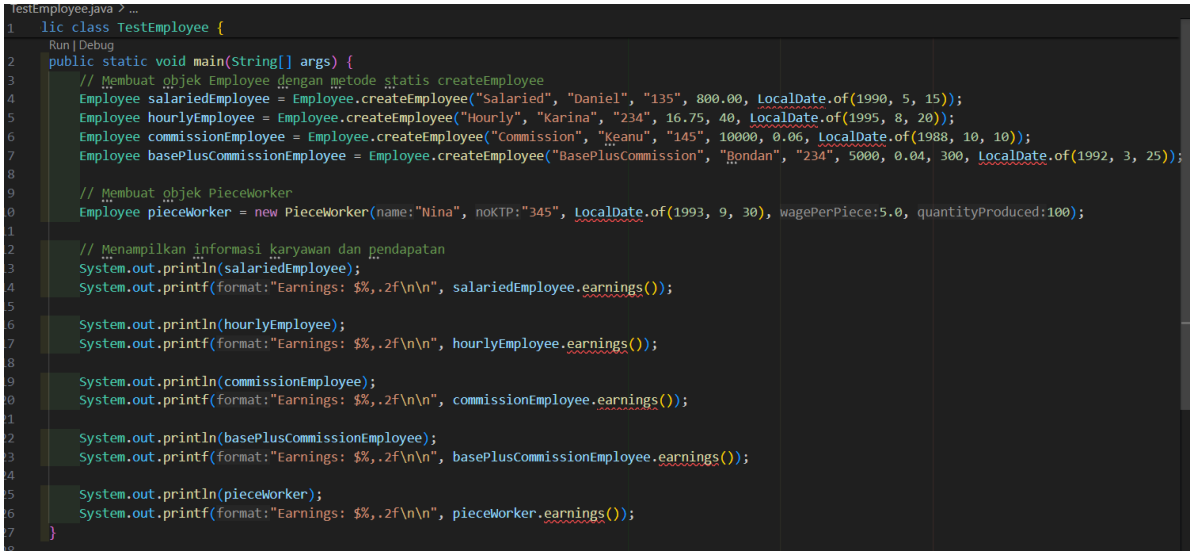
      Jawab!

      Implementasi program dari analis diatas

      Buat kelas baru yaitu PieceWorker:

```java
public class PieceWorker extends Employee {
    private double wagePerPiece; // Upah per barang
    private double quantityProduced; // Banyaknya barang yang diproduksi per minggu

    public PieceWorker(String name, String noKTP, LocalDate birthDate, double wagePerPiece, double quantityProduced) {
        super(name, noKTP, birthDate);
        this.wagePerPiece = wagePerPiece;
        this.quantityProduced = quantityProduced;
    }

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

    public double getWagePerPiece() {
        return wagePerPiece;
    }

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

    public double getQuantityProduced() {
        return quantityProduced;
    }

    @Override
    public double earnings() {
        return wagePerPiece * quantityProduced;
    }

    @Override
    public String toString() {
        return String.format("PieceWorker: " + super.toString() + "\nwage per piece: %.2f\nquantity produced: %.2f", wagePerPiece, quantityProduced);
    }
```

Membuat kelas uji dengan minimal 4 objek yang dibentuk:



```
TestEmployee.java > ...
lic class TestEmployee {
    Run | Debug
    public static void main(String[] args) {
        // Membuat objek Employee dengan metode statis createEmployee
        Employee salariedEmployee = Employee.createEmployee("Salaried", "Daniel", "135", 800.00, LocalDate.of(1990, 5, 15));
        Employee hourlyEmployee = Employee.createEmployee("Hourly", "Karina", "234", 16.75, 40, LocalDate.of(1995, 8, 20));
        Employee commissionEmployee = Employee.createEmployee("Commission", "Keanu", "145", 10000, 0.06, LocalDate.of(1988, 10, 10));
        Employee basePlusCommissionEmployee = Employee.createEmployee("BasePlusCommission", "Bondan", "234", 5000, 0.04, 300, LocalDate.of(1992, 3, 25));

        // Membuat objek PieceWorker
        Employee pieceWorker = new PieceWorker(name:"Nina", noKTP:"345", LocalDate.of(1993, 9, 30), wagePerPiece:5.0, quantityProduced:100);

        // Menampilkan informasi karyawan dan pendapatan
        System.out.println(salariedEmployee);
        System.out.printf(format:"Earnings: $%,.2f\n\n", salariedEmployee.earnings());

        System.out.println(hourlyEmployee);
        System.out.printf(format:"Earnings: $%,.2f\n\n", hourlyEmployee.earnings());

        System.out.println(commissionEmployee);
        System.out.printf(format:"Earnings: $%,.2f\n\n", commissionEmployee.earnings());

        System.out.println(basePlusCommissionEmployee);
        System.out.printf(format:"Earnings: $%,.2f\n\n", basePlusCommissionEmployee.earnings());

        System.out.println(pieceWorker);
        System.out.printf(format:"Earnings: $%,.2f\n\n", pieceWorker.earnings());
    }
```

Nah dalam kelas uji di atas, saya membuat objek PieceWorker dan menampilkannya bersama dengan objek-objek karyawan lainnya. Objek PieceWorker ini menggunakan kriteria baru untuk menghitung pendapatannya berdasarkan banyaknya barang yang diproduksi per minggu.

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