

**LAPORAN TUGAS**  
**PEMROGRAMAN BERBASIS OBJEK**  
**PERTEMUAN KETIGA 5 SEPTEMBER 2023**



**UIN SUNAN AMPEL**  
**S U R A B A Y A**

**DOSEN PEMBIMBING**

Bayu Adhi Nugroho, Ph.D.

(197905182014031001)

**DISUSUN OLEH**

Mochamad Roiyan Rintiarno

(09020622033)

**UIN SUNAN AMPEL SURABAYA**

**TAHUN 2023**

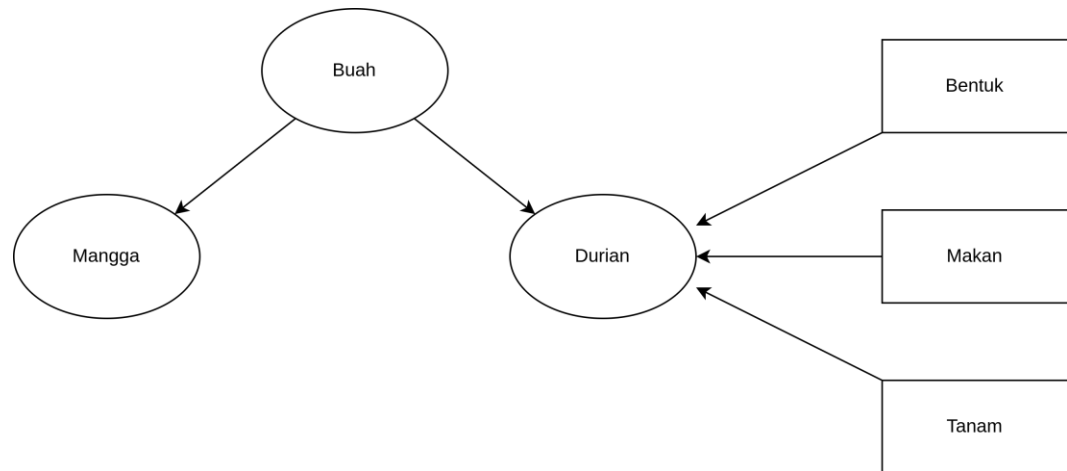
## 1. Tugas

- Membuat satu kelas parent dan turunannya (minimal 2)
- Salah satu atau dua-duanya mengimplementasikan 3 interface
- Membuat diagram
- Menggunakan keyword super

## 2. Pembahasan dan Isi

### a. Persiapan dan Langkah-langkah

#### 1. Membuat Diagram



Pada diagram diatas, Buah adalah sebagai parent, dan Mangga dan Durian adalah sebagai child dari kelas Buah tersebut, dimana Durian mengimplementasikan interface dari kelas Bentuk, Makan, dan Tanam.

## 2. Menyiapkan file Source Code

- a. Kelas Buah sebagai parent dari beberapa kelas child, yakni kelas Durian dan kelas Mangga

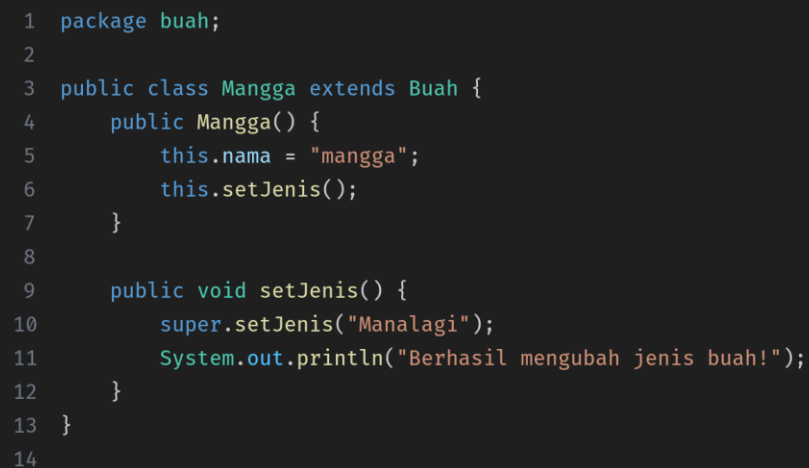
```
1  package buah;
2
3  public class Buah {
4      protected String nama = "jeruk";
5      protected String jenis;
6
7      public void setName(String nama) {
8          this.nama = nama;
9      }
10
11     public String getName() {
12         return nama;
13     }
14
15     public void setJenis(String jenis) {
16         this.jenis = jenis;
17     }
18
19     public String getJenis() {
20         return jenis;
21     }
22
23     public void beli() {
24         System.out.println("Saya membeli buah");
25     }
26
27     public void beli(String subjek) {
28         System.out.println(subjek + " membeli buah");
29     }
30
31     public void beli(int banyaknya) {
32         System.out.println("saya membeli buah sebanyak " + banyaknya + "kg");
33     }
34
35     public void sout(String[] data) {
36         for (int i = 0; i < data.length; i++) {
37             System.out.println((i + 1) + ". " + data[i]);
38         }
39     }
40 }
41
```

Berikut adalah kelas Buah yang bersifat sebagai parent, yang berisikan beberapa atribut dan method, yakni terdapat nama dan jenis sebagai atribut, kemudian juga terdapat method getter dan setter untuk mengset dan

mendapatkan value dari atribut tersebut. Kemudian ada method overload, yakni beberapa method yang mempunyai nama yang sama tetapi memiliki parameter yang berbeda. Dalam hal ini method overload tersebut adalah method beli. Kemudian terdapat method sout yang digunakan untuk menampilkan data.

b. Kelas Child

1. Kelas Mangga

A screenshot of a code editor with a dark background and light-colored text. The code is written in Java and defines a class named 'Mangga' that extends a class named 'Buah'. The code includes a package declaration 'package buah;', a constructor 'public Mangga()' that initializes 'this.nama' to 'mangga' and calls 'this.setJenis()', and a method 'public void setJenis()' that calls 'super.setJenis("Manalagi")' and prints 'Berhasil mengubah jenis buah!'. The code is numbered from 1 to 14 on the left side of the editor.

```
1 package buah;
2
3 public class Mangga extends Buah {
4     public Mangga() {
5         this.nama = "mangga";
6         this.setJenis();
7     }
8
9     public void setJenis() {
10         super.setJenis("Manalagi");
11         System.out.println("Berhasil mengubah jenis buah!");
12     }
13 }
14
```

Child pertama adalah kelas Mangga yang meng-*extend* dari kelas Buah sebagai parentnya, dimana didalamnya terdapat satu buah konstruktor tanpa parameter dan satu buah method, yakni setJenis.

## 2. Kelas Durian


```
1 package buah;
2
3 import appInt.Bentuk;
4 import appInt.Makan;
5 import appInt.Tanam;
6
7 public class Durian extends Buah implements Bentuk, Makan, Tanam {
8     public Durian() {
9         this.nama = "durian";
10        this.setJenis();
11    }
12
13    public void setJenis() {
14        super.setJenis("Musang King");
15        System.out.println("Berhasil mengubah jenis buah!");
16    }
17
18    @Override
19    public void deskripsiBentuk() {
20        System.out.println("Deskripsi bentuk:");
21        String[] bentuk = {
22            "Berbentuk tidak teratur.",
23            "Baunya menyengat.",
24            "Memiliki duri.",
25            "Ada yang besar dan ada yang kecil."
26        };
27        this.sout(bentuk);
28    }
29
30    @Override
31    public void caraMemakan() {
32        System.out.println("Cara memakan:");
33        String[] caraMakan = {
34            "Membeli durian.",
35            "Membuka kulitnya dengan pisau.",
36            "Lalu memakannya."
37        };
38        this.sout(caraMakan);
39    }
40
41    @Override
42    public void caraMenanam() {
43        System.out.println("Cara menanam:");
44        String[] caraTanam = {
45            "Menyiapkan bibit durian",
46            "Menyiapkan lahan penanaman",
47            "Bibit ditanam",
48            "Bibit disiram setiap hari",
49            "Diberi pupuk",
50            "Membasmi hama",
51            "Dipanen"
52        };
53        this.sout(caraTanam);
54    }
55 }
56
```

Pada child kedua, yakni pada kelas Durian, sama seperti kelas mangga sebelumnya, dimana didalamnya terdapat satu buah konstruktor tanpa parameter dan satu buah method, yakni setJenis. Lalu kemudian kelas durian tersebut mengimplementasikan 3 buah interface, yakni Bentuk dengan method deskripsiBentuk didalamnya, lalu Makan dengan caraMakan, dan yang terakhir terdapat Tanam dengan caraMenanamnya.

#### c. Kelas Interface

Interface adalah cara untuk mendefinisikan apa yang harus dilakukan oleh suatu kelas, tanpa memberikan implementasi khusus untuk metode-metode tersebut. Dalam kasus yang kita buat, maka kita membuat 3 interface yang nantinya akan diimplementasikan pada kelas Durian.

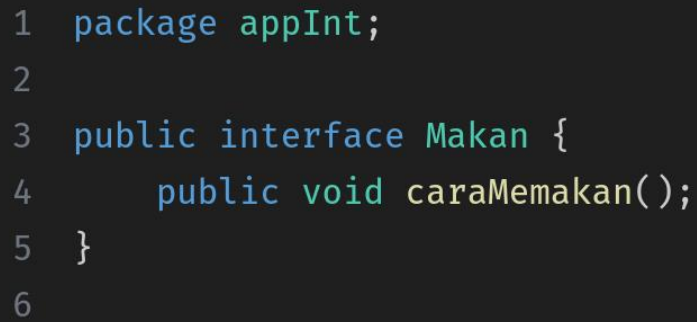
##### 1. Interface Bentuk

A screenshot of a code editor with a dark background and light-colored text. The code defines a package and an interface. The package is named 'appInt'. The interface is named 'Bentuk' and contains a single method 'deskripsiBentuk()' with a return type of 'void'. The code is numbered from 1 to 6.

```
1 package appInt;
2
3 public interface Bentuk {
4     public void deskripsiBentuk();
5 }
6
```

Yang pertama ada interface Bentuk yang berisi method deskripsiBentuk.

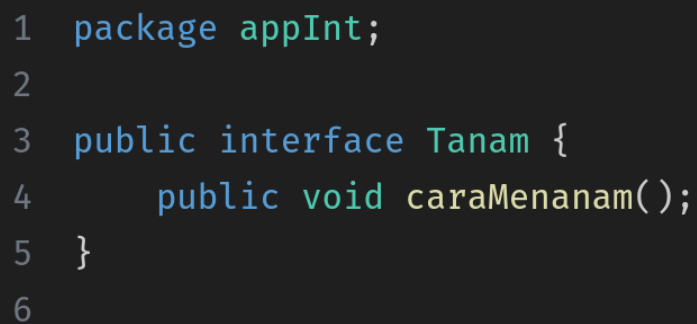
## 2. Interface Makan



```
1 package appInt;  
2  
3 public interface Makan {  
4     public void caraMemakan();  
5 }  
6
```

Lalu yang kedua terdapat interface Makan yang berisi caraMemakan.

## 3. Interface Tanam



```
1 package appInt;  
2  
3 public interface Tanam {  
4     public void caraMenanam();  
5 }  
6
```

Kemudian yang ketiga terdapat interface Tanam yang berisi method caraMenanam.

d. Kelas Output, yakni kelas App

```
1 import buah.Buah;
2 import buah.Durian;
3 import buah.Mangga;
4
5 public class App {
6     public static void main(String[] args) {
7         Mangga mangga = new Mangga();
8         System.out.println("Ini adalah buah " + mangga.getNama());
9         System.out.println("Buah ini berjenis " + mangga.getJenis());
10        mangga.beli();
11        System.out.println("~~~~~");
12        System.out.println("~~~~~");
13        Durian durian = new Durian();
14        System.out.println("Ini adalah buah " + durian.getNama());
15        System.out.println("Buah ini berjenis " + durian.getJenis());
16        mangga.beli("Ibu Titin");
17        durian.deskripsiBentuk();
18        durian.caraMemakan();
19        durian.caraMenanam();
20        System.out.println("~~~~~");
21        System.out.println("~~~~~");
22        Buah b = new Buah();
23        System.out.println("b adalah buah " + b.getNama());
24        b.setJenis("Bali");
25        System.out.println("b adalah jenis buah " + b.getJenis());
26        b.beli(12);
27    }
28 }
29
```

Ini adalah kelas output, dimana kelas untuk menampilkan kelas-kelas maupun interface-interface yang telah dibuat sebelumnya, terdapat 3 buah instance, yakni mangga yang merepresentasikan kelas Mangga, kemudian Durian yang merepresentasikan kelas Durian, lalu yang terakhir b yang merepresentasikan kelas Buah.



### 3. Output

Berikut adalah hasil output dari kelas App.java

[illegible]