

Assignment Paper

**OBJECT ORIENTED PROGRAMMING (OOP)
PRACTICUM
MOCHAMAD WILDANI AZIZI**

Percobaan 1

- a. Buatlah sebuah class parent/superclass dengan nama ClassA.java

```
1 package JS6;
2
3 public class ClassA {
4     public int x;
5     public int y;
6
7     public void getNilai(){
8         System.out.println("Nilai x: " +x);
9         System.out.println("nilai y: " +y);
10    }
11 }
```

- b. Buatlah sebuah class anak/subclass dengan nama ClassB.java

```
1 package JS6;
2
3 public class ClassB {
4     public int z;
5
6     public void getNilaiZ(){
7         System.out.println("Nilai z: " +z);
8     }
9
10    public void getJumlah(){
11        System.out.println("Jumlah: " +(x+y+z));
12    }
13 }
14
```

- c. Buatlah class Percobaan1.java untuk menjalankan program diatas!

```
1 package JS6;
2
3 public class Percobaan1 {
4     public static void main(String[] args) {
5         ClassB hitung = new ClassB();
6         hitung.x=20;
7         hitung.y=30;
8         hitung.z=5;
9         hitung.getNilai();
10        hitung.getNilaiZ();
11        hitung.getJumlah();
12    }
13 }
14
```

- d. Jalankan program diatas, kemudian amati apa yang terjadi!

```
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
  x cannot be resolved or is not a field
  y cannot be resolved or is not a field
  The method getNilai() is undefined for the type ClassB

  at JS6.Percobaan1.main(Percobaan1.java:6)
```

Pertanyaan

1. Pada percobaan 1 diatas program yang dijalankan terjadi error, kemudian perbaiki sehingga program tersebut bisa dijalankan dan tidak error!

Answer:

Berikut adalah perbaikan dengan menggunakan inheritance:



```
1  package JS6;
2
3  public class ClassB extends ClassA {
4      public int z;
5
6      public void getNilaiZ() {
7          System.out.println("Nilai z: " + z);
8      }
9
10     public void getJumlah() {
11         System.out.println("Jumlah: " + (x + y + z));
12     }
13 }
14
```

```
Nilai x: 20
nilai y: 30
Nilai z: 5
Jumlah: 55
```

2. Jelaskan apa penyebab program pada percobaan 1 ketika dijalankan terdapat error!

Answer:

Pada percobaan 1 di atas, program mengalami error karena variabel x dan y yang digunakan di dalam class ClassB tidak dideklarasikan di class tersebut. Variabel x dan y hanya dideklarasikan di class ClassA. Oleh karena itu, class ClassB tidak tahu apa itu x dan y, sehingga terjadi error pada saat pemanggilan metode getJumlah().

Percobaan 2

1. Buatlah sebuah class parent/superclass dengan nama ClassA.java

```
1  package JS6;
2
3
4  public class ClassA {
5      private int x;
6      private int y;
7
8      public void setX(int x){
9          this.x = x;
10     }
11
12     public void setY(int y){
13         this.y = y;
14     }
15
16     public void getNilai(){
17         System.out.println("Nilai x: " +x);
18         System.out.println("nilai y: " +y);
19     }
20 }
```

2. Buatlah sebuah class anak/subclass dengan nama ClassB.java

```
1  package JS6;
2
3  public class ClassB extends ClassA {
4      private int z;
5
6      public void setZ(int z){
7          this.z = z;
8      }
9
10     public void getNilaiZ() {
11         System.out.println("Nilai z: " + z);
12     }
13
14     public void getJumlah() {
15         System.out.println("Jumlah: " + (x + y + z));
16     }
17 }
18
```

3. Buatlah class Percobaan2.java untuk menjalankan program diatas!

```
1 package JS6;
2
3 public class Percobaan2 {
4     public static void main(String[] args) {
5         ClassB hitung = new ClassB();
6         hitung.setX(20);
7         hitung.setY(30);
8         hitung.setZ(5);
9         hitung.getNilai();
10        hitung.getNilaiZ();
11        hitung.getJumlah();
12    }
13 }
14
```

4. Jalankan program diatas, kemudian amati apa yang terjadi!

```
Nilai x: 20
nilai y: 30
Nilai z: 5
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
    The field ClassA.x is not visible
    The field ClassA.y is not visible

    at JS6.ClassB.getJumlah(ClassB.java:15)
    at JS6.Percobaan2.main(Percobaan2.java:11)
```

Pertanyaan

1. Pada percobaan 2 diatas program yang dijalankan terjadi error, kemudian perbaiki sehingga program tersebut bisa dijalankan dan tidak error!

Answer:

Berikut adalah perbaikan pada class ClassB:



```
1 package JS6;
2
3
4 public class ClassA {
5     private int x;
6     private int y;
7
8     public void setX(int x){
9         this.x = x;
10    }
11
12    public void setY(int y){
13        this.y = y;
14    }
15
16    public int getX() {
17        return x;
18    }
19
20    public int getY() {
21        return y;
22    }
23
24    public void getNilai(){
25        System.out.println("Nilai x: " +x);
26        System.out.println("nilai y: " +y);
27    }
28 }
```



```
1 package JS6;
2
3 public class ClassB extends ClassA {
4     private int z;
5
6     public void setZ(int z) {
7         this.z = z;
8     }
9
10    public void getNilaiZ() {
11        System.out.println("Nilai z: " + z);
12    }
13
14    public void getJumlah() {
15        System.out.println("Jumlah: " + (getX() + getY() + z));
16    }
17 }
18
```



```
1 package JS6;
2
3 public class Percobaan2 {
4     public static void main(String[] args) {
5         ClassB hitung = new ClassB();
6         hitung.setX(20);
7         hitung.setY(30);
8         hitung.setZ(5);
9         hitung.getNilai();
10        hitung.getNilaiZ();
11        hitung.getJumlah();
12    }
13 }
14
```

Nilai x: 20
nilai y: 30
Nilai z: 5
Jumlah: 55

2. Jelaskan apa penyebab program pada percobaan 1 ketika dijalankan terdapat error!

Answer:

Pada percobaan 2, program tersebut tidak mengalami error karena telah diatasi masalah pada percobaan 1. Beberapa perubahan yang telah dilakukan untuk mengatasi error antara lain:

1. Inheritance: ClassB sekarang meng-extend ClassA, sehingga ClassB dapat mengakses atribut x dan y yang bersifat private di ClassA.
2. Setter Methods: Setter methods (setX, setY, dan setZ) digunakan untuk menginisialisasi nilai atribut x, y, dan z dari luar class. Dengan demikian, nilai atribut tersebut dapat diatur dengan cara yang terkontrol.
3. Pemanggilan Metode getJumlah(): Metode getJumlah() pada ClassB sekarang dapat dijalankan tanpa error karena dapat mengakses atribut x dan y melalui inheritance dari ClassA.

Oleh karena itu, program pada percobaan 2 dapat dijalankan tanpa masalah dan menghasilkan output yang sesuai dengan nilai yang diatur.

Sebaliknya, pada percobaan 1 terdapat error karena tidak ada hubungan inheritance antara ClassA dan ClassB, sehingga ClassB tidak dapat mengakses atribut x dan y yang bersifat public di ClassA. Selain itu, tidak ada inisialisasi nilai untuk atribut x dan y sehingga metode getJumlah() pada ClassB mengalami kegagalan saat mencoba mengakses atribut yang tidak dideklarasikan di class tersebut.

Percobaan 3

1. Buatlah sebuah class parent/superclass dengan nama Bangun.java



```
1 package JS6;
2
3 public class Bangun {
4     protected double phi;
5     protected int r;
6 }
7
```

2. Buatlah sebuah class anak/subclass dengan nama Tabung.java



```
1 package JS6;
2
3 public class Tabung extends Bangun {
4     protected int t;
5
6     public void setSuperPhi (double phi){
7         super.phi = phi;
8     }
9
10    public void setSuperR (int r){
11        super.r = r;
12    }
13
14    public void setT(int t){
15        this.t = t;
16    }
17
18    public void volume(){
19        System.out.println("Volume Tabung adalah: " +(super.phi*super.r*super.r*this.t));
20    }
21 }
22
```

3. Buatlah class Percobaan3.java untuk menjalankan program diatas!


```
1 package JS6;
2
3 public class Percobaan3 {
4     public static void main(String[] args) {
5         Tabung tabung = new Tabung();
6         tabung.setSuperPhi(3.14);
7         tabung.setSuperR(10);
8         tabung.setT(3);
9         tabung.volume();
10    }
11 }
12
```

4. Jalankan program diatas

```
Volume Tabung adalah: 942.0
```

Pertanyaan

1. Jelaskan fungsi “super” pada potongan program berikut di class Tabung!

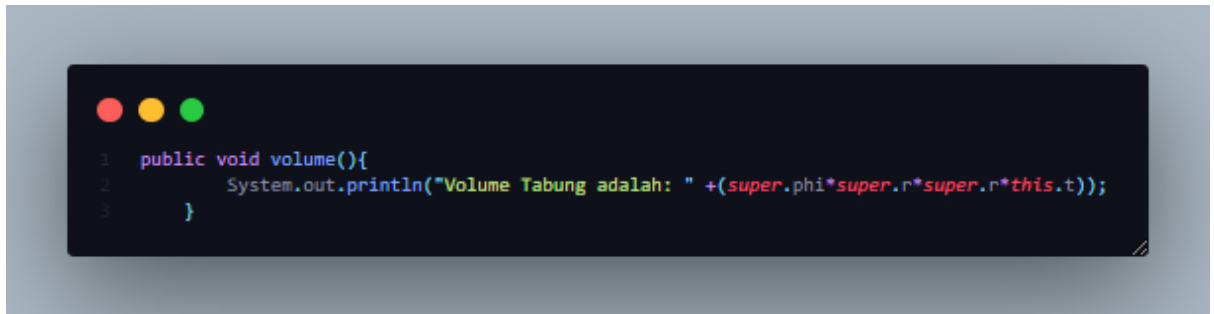
```
1 public void setSuperPhi (double phi){
2     super.phi = phi;
3 }
4
5 public void setSuperR (int r){
6     super.r = r;
7 }
```

Answer: Kata kunci super digunakan dalam pemrograman berorientasi objek (OOP) untuk merujuk ke anggota (atribut atau metode) dari superclass atau class induk. Dalam konteks program yang diberikan, super digunakan untuk merujuk ke variabel phi dan r yang dideklarasikan di class induk (Bangun).

Fungsi super pada potongan program tersebut adalah untuk mengakses variabel phi dan r yang dideklarasikan di class Bangun (superclass). Karena variabel phi dan r dideklarasikan dengan modifier akses protected di class Bangun, mereka dapat diakses dari class turunan (Tabung) menggunakan kata kunci super.

Dengan menggunakan `super.phi` dan `super.r`, kita merujuk ke variabel `phi` dan `r` di class `Bangun`, dan kita menginisialisasi atau mengubah nilainya melalui metode `setSuperPhi` dan `setSuperR` di class `Tabung`. Ini memungkinkan class `Tabung` untuk menggunakan nilai dari superclassnya dan menerapkannya dalam perhitungan volume pada metode `volume()`.

2. Jelaskan fungsi “super” dan “this” pada potongan program berikut di class `Tabung`!



```
1 public void volume(){
2     System.out.println("Volume Tabung adalah: " + (super.phi*super.r*super.r*this.t));
3 }
```

Answer: Dengan menggunakan `super` dan `this`, program tersebut mengambil nilai dari variabel `phi` dan `r` dari superclass (`Bangun`) dan variabel `t` dari class saat ini (`Tabung`). Selanjutnya, nilai-nilai tersebut digunakan untuk menghitung volume tabung sesuai dengan rumus matematika yang sesuai.

Jadi, `super` dan `this` digunakan untuk membedakan antara variabel-variabel yang mungkin memiliki nama yang sama di class induk dan class saat ini. Dalam kasus ini, `super` membantu kita mengakses variabel dari superclass, sedangkan `this` digunakan untuk mengakses variabel lokal di class saat ini.

3. Jelaskan mengapa pada class `Tabung` tidak dideklarasikan atribut “phi” dan “r” tetapi class tersebut dapat mengakses atribut tersebut!

Answer:

Pada class `Tabung`, atribut `phi` dan `r` tidak perlu dideklarasikan secara eksplisit karena class `Tabung` adalah subclass dari class `Bangun`, dan `phi` dan `r` sudah dideklarasikan di class `Bangun`. Oleh karena itu, class `Tabung` dapat mengakses atribut `phi` dan `r` tanpa perlu mendeklarasikannya kembali.

Percobaan 4

1. Buatlah tiga file dengan nama ClassA.java , ClassB.java , dan ClassC.java, seperti pada kode program dibawah ini!

```
1 package JS6;  
2  
3 public class KelasA {  
4     KelasA(){  
5         System.out.println("Konstruktor A dijalankan");  
6     }  
7 }  
8
```

```
1 package JS6;  
2  
3 public class KelasB extends KelasA {  
4     KelasB(){  
5         System.out.println("Konstruktor B dijalankan");  
6     }  
7 }  
8
```

```
1 package JS6;  
2  
3 public class KelasC extends KelasB {  
4     KelasC(){  
5         System.out.println("Konstruktor C dijalankan");  
6     }  
7 }  
8
```

2. Buatlah class Percobaan4.java untuk menjalankan program diatas!



```
1 package JS6;
2
3 public class Percobaan4 {
4     public static void main(String[] args) {
5         KelasC test = new KelasC();
6     }
7 }
8
```

3. Jalankan program kemudian amati apa yang terjadi!

```
Konstruktor A dijalankan
Konstruktor B dijalankan
Konstruktor C dijalankan
```

Pertanyaan

1. Pada percobaan 4 sebutkan mana class yang termasuk superclass dan subclass, kemudian jelaskan alasannya!

Answer:

Dalam percobaan 4, terdapat tiga class: KelasA, KelasB, dan KelasC. Berikut adalah penjelasan mengenai superclass dan subclass dalam hierarki class tersebut:

- KelasA:
 - a. Superclass: Tidak memiliki superclass lagi di atasnya.
 - b. Subclass: KelasB.
 - c. Alasan: KelasA adalah class paling dasar dalam hierarki ini, dan tidak ada class lain yang diwarisi dari KelasA. Oleh karena itu, KelasA tidak memiliki superclass dan menjadi awal dari hierarki.
- KelasB:
 - a. Superclass: KelasA.
 - b. Subclass: KelasC.
 - c. Alasan: KelasB mewarisi dari KelasA, sehingga KelasA menjadi superclass dari KelasB. Pada saat yang sama, KelasB menjadi superclass bagi KelasC.
- KelasC:
 - a. Superclass: KelasB.
 - b. Subclass: Tidak memiliki subclass lagi di bawahnya.
 - c. Alasan: KelasC mewarisi dari KelasB, sehingga KelasB menjadi superclass dari KelasC. Karena tidak ada class lain yang diwarisi dari KelasC, maka KelasC menjadi akhir atau leaf dalam hierarki.

- Ubahlah isi konstruktor default ClassC seperti berikut:



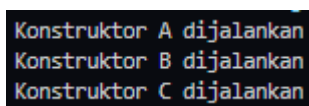
```
1 public class KelasC extends KelasB {
2     KelasC(){
3         super();
4         System.out.println("Konstruktor C dijalankan");
5     }
6 }
```

Tambahkan kata `super()` di baris Pertama dalam konstruktor defaultnya. Coba jalankan kembali class Percobaan4 dan terlihat tidak ada perbedaan dari hasil outputnya!

Answer:



```
1 public class KelasC extends KelasB {
2     KelasC(){
3         super();
4         System.out.println("Konstruktor C dijalankan");
5     }
6 }
```



```
Konstruktor A dijalankan
Konstruktor B dijalankan
Konstruktor C dijalankan
```

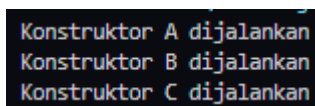
Ketika menjalankan class Percobaan4, outputnya akan tetap sama, Hal ini karena konstruktor default dari class KelasC secara otomatis memanggil konstruktor default dari class KelasB, yang kemudian memanggil konstruktor default dari class KelasA. Dengan penambahan `super()` di baris pertama konstruktor default class KelasC, hasilnya tidak berubah karena konstruktor default class KelasC secara otomatis memanggil konstruktor default superclassnya (KelasB).

- Ubah isi konstruktor default ClassC seperti berikut:



```
1 package JS6;
2
3 public class KelasC extends KelasB {
4     KelasC(){
5         System.out.println("Konstruktor C dijalankan");
6         super();
7     }
8 }
9
```

Ketika mengubah posisi `super()` di baris kedua dalam konstruktor defaultnya dan terlihat ada error. Kemudian kembalikan `super()` ke baris pertama seperti sebelumnya, maka errornya akan hilang. Perhatikan hasil keluaran ketika class Percobaan4 dijalankan. Kenapa bisa tampil output seperti berikut pada saat instansiasi objek test dari class ClassC



```
Konstruktor A dijalankan
Konstruktor B dijalankan
Konstruktor C dijalankan
```

Jelaskan bagaimana urutan proses jalannya konstruktor saat objek test dibuat!

Answer: Ketika memindahkan pemanggilan `super()` ke baris kedua dalam konstruktor default class `KelasC`, hal ini menyebabkan terjadinya error. Pemanggilan `super()` atau `this()` (jika ada) harus selalu berada di baris pertama konstruktor, karena itulah urutan eksekusi yang diperlukan. Jika Anda mencoba memindahkan pemanggilan `super()` ke baris kedua, maka compiler akan menghasilkan error.

4. Apakah fungsi `super()` pada potongan program dibawah ini di `ClassC`!



```
1 public class KelasC extends KelasB {
2     KelasC(){
3         super();
4         System.out.println("Konstruktor C dijalankan");
5     }
6 }
```

Answer: Fungsi dari `super()` adalah untuk memanggil konstruktor dari superclass, yaitu class `KelasB`. Dalam konteks ini, pemanggilan `super()` digunakan untuk memastikan bahwa konstruktor dari class `KelasB` dijalankan sebelum konstruktor dari class `KelasC`.

Tugas

Buatlah sebuah program dengan konsep pewarisan seperti pada class diagram berikut ini. Kemudian buatlah instansiasi objek untuk menampilkan data nama pegawai dan gaji yang didapatkannya.

```
1  package Tugas;
2
3  public class Pegawai {
4      private String nip;
5      private String nama;
6      private String alamat;
7
8      public Pegawai(String nip, String nama, String alamat) {
9          this.nip = nip;
10         this.nama = nama;
11         this.alamat = alamat;
12     }
13
14     public String getNama() {
15         return nama;
16     }
17
18     public int getGaji() {
19         return 0; // Gaji dasar untuk Pegawai (belum diimplementasikan)
20     }
21 }
22
```

```

1  package Tugas;
2  import java.util.ArrayList;
3
4  public class Dosen extends Pegawai{
5      private int jumlahSKS;
6      private static final int TARIF_SKS = 50000;
7
8      public Dosen(String nip, String nama, String alamat) {
9          super(nip, nama, alamat);
10     }
11
12     public void setSKS(int jumlahSKS) {
13         this.jumlahSKS = jumlahSKS;
14     }
15
16     @Override
17     public int getGaji() {
18         return TARIF_SKS * jumlahSKS;
19     }
20 }
21

```

```

1  package Tugas;
2  import java.util.ArrayList;
3
4  public class DaftarGaji {
5      private ArrayList<Pegawai> listPegawai;
6
7      public DaftarGaji(int kapasitas) {
8          listPegawai = new ArrayList<>(kapasitas);
9      }
10
11     public void addPegawai(Pegawai pegawai) {
12         listPegawai.add(pegawai);
13     }
14
15     public void printSemuaGaji() {
16         for (Pegawai pegawai : listPegawai) {
17             System.out.println("Nama Pegawai: " + pegawai.getNama());
18             System.out.println("Gaji: " + pegawai.getGaji());
19             System.out.println("-----");
20         }
21     }
22 }
23

```



```

1  package Tugas;
2
3  public class Main {
4      public static void main(String[] args) {
5          DaftarGaji daftarGaji = new DaftarGaji(3);
6
7          Pegawai pegawai1 = new Pegawai("123", "John Doe", "Jl. ABC");
8          Dosen dosen1 = new Dosen("456", "Jane Smith", "Jl. XYZ");
9
10         dosen1.setSKS(10); // Set jumlah SKS untuk Dosen
11
12         daftarGaji.addPegawai(pegawai1);
13         daftarGaji.addPegawai(dosen1);
14
15         daftarGaji.printSemuaGaji();
16     }
17 }
18

```

```

Nama Pegawai: John Doe
Gaji: 0
-----
Nama Pegawai: Jane Smith
Gaji: 500000
-----

```

Explanation:

1. Class Pegawai memiliki atribut NIP, nama, dan alamat. Metode getGaji() belum diimplementasikan dan dikembalikan nilai 0.
2. Class Dosen merupakan subclass dari Pegawai dan menambahkan atribut jumlahSKS dan konstanta TARIF_SKS. Metode getGaji() di-override untuk menghitung gaji dosen berdasarkan jumlah SKS dan tarif per SKS.
3. Class DaftarGaji memiliki atribut listPegawai untuk menyimpan daftar pegawai. Metode addPegawai digunakan untuk menambahkan pegawai ke dalam list, dan printSemuaGaji digunakan untuk mencetak nama dan gaji semua pegawai dalam list.
4. Dalam method main, kita membuat objek dari class DaftarGaji, Pegawai, dan Dosen. Lalu, menambahkan pegawai ke dalam daftar dan mencetak gaji semua pegawai.