

NAMA : Hilda Sava Alzena

NIM : 240202865

KELAS: 3 IKRA

LAPORAN PRAKTIKUM MINGGU KE 4

“Penerapan Polymorphism dalam Pemrograman Berorientasi Objek pada Sistem Agri-POS”

TUJUAN

1. Mahasiswa mampu menjelaskan konsep **polymorphism** dalam pemrograman berorientasi objek (OOP).
2. Mahasiswa mampu membedakan antara **method overloading** dan **method overriding**.
3. Mahasiswa mampu mengimplementasikan **polymorphism** melalui overloading, overriding, dan dynamic binding dalam program Java.
4. Mahasiswa mampu menerapkan konsep polymorphism pada kasus nyata, yaitu sistem informasi **Agri-POS** untuk pengelolaan data produk.

DASAR TEORI

1. Pengertian Polymorphism

Polymorphism adalah kemampuan suatu method atau objek untuk memiliki banyak bentuk. Konsep ini memungkinkan objek yang berbeda merespons method yang sama dengan cara yang berbeda sesuai jenis objeknya.

2. Jenis-jenis Polymorphism

- **Overloading** → method memiliki nama sama tetapi parameter berbeda.
- **Overriding** → subclass mengganti implementasi method dari superclass.
- **Dynamic Binding** → pemanggilan method ditentukan saat runtime sesuai tipe objek sebenarnya.

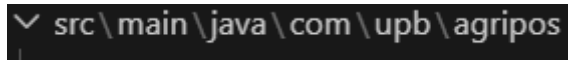
3. Penerapan dalam Agri-POS

Dalam sistem Agri-POS, class Produk berperan sebagai superclass, sedangkan Benih, Pupuk, dan AlatPertanian menjadi subclass yang mengoverride method getInfo() untuk menampilkan informasi berbeda. Method tambahStok() juga dioverload untuk menambah stok dengan tipe data berbeda.

LANGKAH PRAKTIKUM

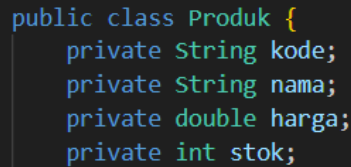
1. Membuat Class Superclass Produk

- Buat file Produk.java di folder :



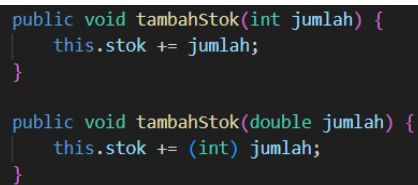
```
src\main\java\com\upb\agripus
```

- Tambahkan atribut: kode, nama, harga, dan stok



```
public class Produk {  
    private String kode;  
    private String nama;  
    private double harga;  
    private int stok;  
}
```

- Buat konstruktor untuk menginisialisasi atribut tersebut.
- Tambahkan dua method **overloading** dengan nama sama tetapi parameter berbeda:

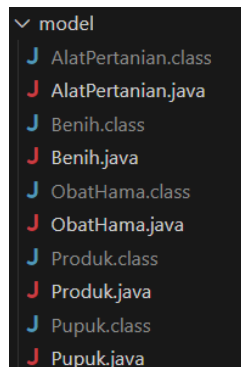


```
public void tambahStok(int jumlah) {  
    this.stok += jumlah;  
}  
  
public void tambahStok(double jumlah) {  
    this.stok += (int) jumlah;  
}
```

- Tambahkan method getInfo() untuk menampilkan informasi umum produk.

2. Membuat Class Subclass Benih, Pupuk, AlatPertanian, dan ObatHama.

- Buat empat file di folder src.main.java.com.upb.agripos.model:



```
model  
├── AlatPertanian.class  
├── AlatPertanian.java  
├── Benih.class  
├── Benih.java  
├── ObatHama.class  
├── ObatHama.java  
├── Produk.class  
├── Produk.java  
├── Pupuk.class  
└── Pupuk.java
```

- Masing-masing subclass **mewarisi class Produk** dengan keyword extends.
- Tambahkan atribut khusus untuk setiap subclass, misalnya:
- Benih → atribut varietas
- Pupuk → atribut jenisPupuk
- AlatPertanian → atribut bahan
- ObatHama → atribut kandunganAktif
- Lakukan **overriding** terhadap method `getInfo()` agar setiap subclass menampilkan informasi spesifik produknya.
Ini menunjukkan **runtime polymorphism (overriding & dynamic binding)**.

3. Membuat Class MainPolymorphism

- Buat file MainPolymorphism.java di folder com.upb.agripos.
- Import semua class dari package model dan util
- Di dalam main(), buat array Produk[] daftarProduk yang berisi objek dari keempat subclass:

```
Produk[] daftarProduk = {  
    new Benih(kode:"BNH-001", nama:"Benih Coklat 50", harga:90000, stok:100, varietas:"50"),  
    new Pupuk(kode:"PPK-101", nama:"Pupuk Kandang 25kg", harga:70000, stok:50, jenis:"Organik"),  
    new AlatPertanian(kode:"ALT-501", nama:"Penyiram Tanaman", harga:40000, stok:75, bahan:"Plastik"),  
    new ObatHama(kode:"HLD-012", nama:"Obat Hama Ulat Daun", harga:50000, stok:50, bahanAktif:"Hypsipyla robusta")  
};
```

- Gunakan perulangan for untuk memanggil getInfo() dari setiap objek. Perhatikan bahwa hasil yang muncul menyesuaikan jenis objeknya (contoh penerapan **dynamic binding**).

4. Menambahkan Class CreditBy

- Buat file CreditBy.java di folder com.upb.agripos.util
- tambahkan method static berikut:

```
public class CreditBy {  
    public static void print() {  
        System.out.println(x:"\nCredit By: 240202865 - hilda");  
    }  
}
```

- Panggil method ini di akhir program untuk menampilkan identitas mahasiswa.

5. Menjalankan Program

- Jalankan program dan pastikan hasil getInfo() dari semua subclass (termasuk ObatHama) tampil di console.

6. Melakukan Commit dan Push ke GitHub

- Pastikan semua file sudah tersimpan di folder praktikum/week4-polymorphism.
- Lakukan perintah berikut di terminal:

```
PS C:\Users\Hilda\OneDrive\Documents\Praktikum\oop-202501-240202865\praktikum\week4-polymorphism\src\main\java> git add .  
PS C:\Users\Hilda\OneDrive\Documents\Praktikum\oop-202501-240202865\praktikum\week4-polymorphism\src\main\java> git commit -m "week4-polymorphism"
```

KODE PROGRAM

1. Alat Pertanian.java

```
praktikum > week4-polymorphism > src > main > java > com > upb > agripos > model > J AlatPertanian.java > ...
1  package com.upb.agripos.model;
2
3  public class AlatPertanian extends Produk {
4      private String bahan;
5
6      public AlatPertanian(String kode, String nama, double harga, int stok, String bahan) {
7          super(kode, nama, harga, stok);
8          this.bahan = bahan;
9      }
10
11     @Override
12     public String getInfo() {
13         return "Alat Pertanian: " + super.getInfo() + ", Bahan: " + bahan;
14     }
15 }
16 |
```

2. Benih.java

```
praktikum > week4-polymorphism > src > main > java > com > upb > agripos > model > J Benih.java > ...
1  package com.upb.agripos.model;
2
3  public class Benih extends Produk {
4      private String varietas;
5
6      public Benih(String kode, String nama, double harga, int stok, String varietas) {
7          super(kode, nama, harga, stok);
8          this.varietas = varietas;
9      }
10
11     @Override
12     public String getInfo() {
13         return "Benih: " + super.getInfo() + ", Varietas: " + varietas;
14     }
15 }
16 |
```

3. ObatHama.java

```
praktikum > week4-polymorphism > src > main > java > com > upb > agripos > model > J ObatHama.java > ...
1  package com.upb.agripos.model;
2
3  public class ObatHama extends Produk {
4      private String bahanAktif;
5
6      public ObatHama(String kode, String nama, double harga, int stok, String bahanAktif) {
7          super(kode, nama, harga, stok);
8          this.bahanAktif = bahanAktif;
9      }
10
11     @Override
12     public String getInfo() {
13         return "Obat Hama: Produk: " + super.getInfo() + ", Bahan Aktif: " + bahanAktif;
14     }
15 }
16 }
17 |
```


5.Produk.java

```
praktikum > week4-polymorphism > src > main > java > com > upb > agripos > model > J Produk.jav
1  package com.upb.agripos.model;
2
3  public class Produk {
4      private String kode;
5      private String nama;
6      private double harga;
7      private int stok;
8
9      public Produk(String kode, String nama, double harga, int stok) {
10         this.kode = kode;
11         this.nama = nama;
12         this.harga = harga;
13         this.stok = stok;
14     }
15
16     public void tambahStok(int jumlah) {
17         this.stok += jumlah;
18     }
19
20     public void tambahStok(double jumlah) {
21         this.stok += (int) jumlah;
22     }
23
24     public String getKode() {
25         return kode;
26     }
27
28     public String getNama() {
29         return nama;
30     }
31
32     public double getHarga() {
33         return harga;
34     }
35
36     public int getStok() {
37         return stok;
38     }
39
40     public String getInfo() {
41         return "Produk: " + nama + " (Kode: " + kode + ")";
42     }
43 }
44
```


6.Pupuk.java

```
praktikum > week4-polymorphism > src > main > java > com > upb > agripos > model > J Pupuk.java > ...
1  package com.upb.agripos.model;
2
3  public class Pupuk extends Produk {
4      private String jenis;
5
6      public Pupuk(String kode, String nama, double harga, int stok, String jenis) {
7          super(kode, nama, harga, stok);
8          this.jenis = jenis;
9      }
10
11     @Override
12     public String getInfo() {
13         return "Pupuk: " + super.getInfo() + ", Jenis: " + jenis;
14     }
15 }
16
```

7.CreditBy.java


```
praktikum > week4-polymorphism > src > main > java > com > upb > agripas > util >  CreditBy.java
1  package com.upb.agripas.util;
2
3  public class CreditBy {
4      public static void print() {
5          System.out.println(x:"\nCredit By: 240202865 - hilda");
6      }
7  }
8
```

8.MainPolymorphism.java

```
praktikum > week4-polymorphism > src > main > java > com > upb > agripas >  MainPolymorphism.java > ...
1  package com.upb.agripas;
2
3  import com.upb.agripas.model.*;
4  import com.upb.agripas.util.CreditBy;
5
6  public class MainPolymorphism {
7      public static void main(String[] args) {
8
9          Produk[] daftarProduk = {
10             new Benih(kode:"BNH-001", nama:"Benih Coklat 50", harga:90000, stok:100, varietas:"50"),
11             new Pupuk(kode:"PPK-101", nama:"Pupuk Kandang 25kg", harga:70000, stok:50, jenis:"Organik"),
12             new AlatPertanian(kode:"ALT-501", nama:"Penyiram Tanaman", harga:40000, stok:75, material:"Plastik")
13             new ObatHama("HLD-012", "Obat Hama Ulat Daun", 50000, 50, "Hypsipyla robusta")
14         };
15
16         for (Produk p : daftarProduk) {
17             System.out.println(p.getInfo());
18         }
19
20         CreditBy.print();
21     }
22 }
23
```

HASIL EKSEKUSI

```
PS C:\Users\Hilda\OneDrive\Documents\Praktikum\oop-202501-240202865\praktikum\week4-polymorphism\src\main\java> java com.upb.agripas.MainPolymorphism
Benih: Produk: Benih Coklat 50 (Kode: BNH-001), Varietas: 50
Pupuk: Produk: Pupuk Kandang 25kg (Kode: PPK-101), Jenis: Organik
Alat Pertanian: Produk: Penyiram Tanaman (Kode: ALT-501), Bahan: Plastik
Obat Hama: Produk: Produk: Obat Hama Ulat Daun (Kode: HLD-012), Bahan Aktif: Hypsipyla robusta

Credit By: 240202865 - hilda
PS C:\Users\Hilda\OneDrive\Documents\Praktikum\oop-202501-240202865\praktikum\week4-polymorphism\src\main\java> 
```

ANALISIS

1. Jalannya Kode Program

Program membuat class Produk sebagai superclass dan subclass Benih, Pupuk, AlatPertanian, serta ObatHama yang mengoverride method getInfo(). Saat dijalankan, setiap objek menampilkan info berbeda meskipun dipanggil lewat referensi Produk (dynamic binding).

2. Perbedaan Minggu Ini

Minggu ini menambahkan konsep polymorphism (overloading, overriding, dan dynamic binding), sedangkan minggu sebelumnya hanya fokus pada inheritance.

3. Kendala dan Solusi

Beberapa error muncul karena nama package dan konstruktor belum sesuai. Setelah diperbaiki dan menambahkan @Override, program dapat berjalan dengan benar.

KESIMULAN

Dengan menerapkan polymorphism, program menjadi lebih fleksibel karena method yang sama dapat memiliki perilaku berbeda sesuai jenis objek. Konsep overloading, overriding, dan dynamic binding membuat kode lebih efisien, mudah dikembangkan, dan mendukung pengelolaan data produk pada sistem Agri-POS.

QUIZ

1. Apa perbedaan overloading dan overriding?

Jawaban:

Overloading terjadi saat method memiliki nama sama tetapi parameter berbeda, sedangkan overriding terjadi ketika subclass mengganti isi method superclass dengan implementasi yang berbeda.

2. Bagaimana Java menentukan method mana yang dipanggil dalam dynamic binding?

Jawaban:

Java menentukan method yang dipanggil saat runtime berdasarkan tipe objek sebenarnya, bukan tipe referensinya.

3. Berikan contoh kasus polymorphism dalam sistem POS selain produk pertanian.

Jawaban:

Contohnya pada sistem POS restoran, class Menu bisa memiliki subclass Makanan, Minuman, dan Dessert yang masing-masing mengoverride method getInfo() untuk menampilkan detail menu berbeda.

CHECKLIST KEBERHASILAN

- Overloading tambahStok berhasil.
- Overriding getInfo pada subclass berjalan.
- Dynamic binding berjalan melalui array produk.
- Output menampilkan identitas mahasiswa.
- Screenshot & laporan disertakan.