

**PRAKTIKUM PARADIGMA PEMROGRAMAN**  
**MODUL 5**



**Disusun oleh :**

**Nama : Fidelia Ping**

**NIM : 245410012**

**Kelas : Informatika 1**

**PROGRAM STUDI INFORMATIKA**  
**PROGRAM SARJANA**  
**FAKULTAS TEKNOLOGI INFORMASI**  
**UNIVERSITAS TEKNOLOGI DIGITAL INDONESIA**  
**YOGYAKARTA**  
**2025**

## MODUL 5

### Package, Acces Modifier, Static

#### A. TUJUAN PRAKTIKUM

Dapat menggunakan paket java

Dapat membuat dan menggunakan package yang dibuat

Dapat mengatur hak akses member

#### B. PEMBAHASAN LISTING

##### Praktik 2

```
package mahasiswa;  
public class paket  
{  
    public static void isiPaket(){  
        System.out.println("ini hasil import");  
    }  
}
```

##### Pembahasan program :

- package mahasiswa;

Baris ini menunjukkan bahwa kelas paket berada di dalam package bernama mahasiswa. Package digunakan untuk mengelompokkan kelas-kelas Java agar lebih terstruktur dan mudah dikelola.

- Deklarasi kelas paket

Kelas ini bersifat public, artinya bisa diakses dari kelas lain (bahkan di luar package) selama di-import dengan benar.

- Metode isiPaket()

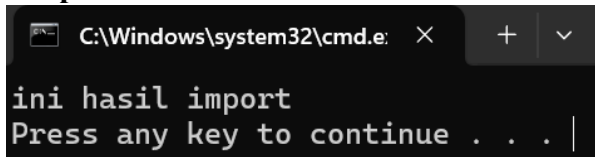
Merupakan metode static, sehingga dapat dipanggil langsung tanpa membuat objek.

##### Praktik 3

```
import mahasiswa.*;  
public class cobaPaket  
{  
    public static void main(String args[])  
    {  
        paket coba=new paket();  
        coba.isiPaket();  
    }  
}
```

**Pembahasan program :** Hubungan antara program cobaPaket dan program paket sebelumnya adalah bahwa **program cobaPaket menggunakan atau memanggil kelas paket yang berada di dalam package mahasiswa**. Program ini **mendefinisikan sebuah kelas bernama paket di dalam package mahasiswa**. Kelas tersebut memiliki metode isiPaket() yang menampilkan teks "ini adalah hasil import". Program paket berfungsi sebagai **sumber kode (kelas) dalam package mahasiswa**, sedangkan cobaPaket adalah **program utama yang menggunakan kelas tersebut** melalui perintah import. Hubungan ini menunjukkan konsep **modularitas dan reuse code** di Java, di mana satu kelas bisa digunakan kembali di program lain tanpa harus menulis ulang kodenya.

## Output



```
C:\Windows\system32\cmd.e  X  +  v
ini hasil import
Press any key to continue . . . |
```

Pembahasan output :

## Praktik 4

```
package mahasiswa;
public class paket
{
    String nama;
    int umur;
    public static void isiPaket(){
        System.out.println("ini hasil import");
    }
    public void setData(int umur, String nama){
        this.umur = umur;
        this.nama = nama;
    }
    public void info(){
        System.out.println("Nama : "+nama);
        System.out.println("Umur : "+umur);
    }
}
```

**Pembahasan program :** Program paket di atas merupakan sebuah kelas yang berada di dalam **package bernama** mahasiswa, yang berfungsi untuk menyimpan dan menampilkan data sederhana tentang seseorang. Kelas ini memiliki dua atribut, yaitu nama (tipe String) dan umur (tipe int) untuk menyimpan identitas. Di dalamnya terdapat tiga metode: `isiPaket()` yang bersifat **static** untuk menampilkan pesan "ini hasil import" sebagai penanda bahwa kelas berhasil digunakan dari luar package, `setData(int umur, String nama)` yang digunakan untuk mengisi nilai atribut umur dan nama, serta `info()` yang menampilkan informasi nama dan umur ke layar. Program ini menggambarkan konsep **enkapsulasi data dan pemisahan kode menggunakan package** di Java, sehingga kelas dapat dengan mudah diakses oleh program lain yang mengimpor package mahasiswa.

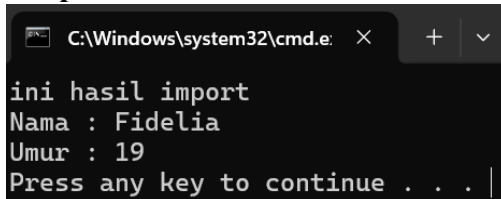
## Praktik 5

```
import mahasiswa.*;
public class cobaPaket
{
    public static void main(String args[])
    {
        paket coba=new paket();
        coba.isiPaket();
        coba.setData(19, "Fidelia");
        coba.info();
    }
}
```

**Pembahasan output :** Program cobaPaket di atas berfungsi untuk **mengakses dan menggunakan kelas paket** yang telah didefinisikan sebelumnya di dalam **package**

**mahasiswa.** Melalui perintah `import mahasiswa.*;`, program ini mengimpor seluruh kelas yang ada dalam package tersebut sehingga kelas paket dapat digunakan tanpa perlu menuliskan nama paketnya secara lengkap. Di dalam `main()`, objek `coba` dibuat dari kelas `paket`, kemudian program memanggil metode `isiPaket()` untuk menampilkan pesan *"ini hasil import"* sebagai tanda bahwa kelas berhasil diimpor. Selanjutnya, metode `setData(19, "Fidelia")` digunakan untuk mengisi atribut umur dan nama, dan metode `info()` menampilkan hasil data tersebut ke layar dalam format.

### Output



```
C:\Windows\system32\cmd.e: X + v
ini hasil import
Nama : Fidelia
Umur : 19
Press any key to continue . . . |
```

### Pembahasan output :

- Baris pertama *"ini hasil import"* muncul karena program memanggil metode `isiPaket()` yang ada di kelas `paket`. Metode ini bersifat `static`, sehingga bisa dipanggil langsung dari objek tanpa membutuhkan nilai atribut apa pun. Fungsinya hanya untuk menampilkan pesan bahwa kelas paket berhasil diimpor dari package `mahasiswa`.
- Baris kedua dan ketiga (*"Nama : Fidelia"* dan *"Umur : 19"*) ditampilkan setelah pemanggilan metode `setData(19, "Fidelia")`, yang menyimpan nilai nama = *"Fidelia"* dan umur = 19 ke dalam atribut objek `coba`. Setelah itu, metode `info()` digunakan untuk menampilkan isi atribut tersebut ke layar.

### Latihan

```
public class Film
{
    public String judul;
    public String jenis;
    public int durasi;

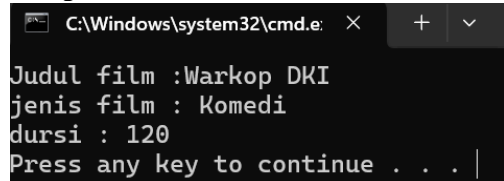
    public void setFilm(String judul,String jenis,int durasi)
    {
        this.judul=judul;
        this.jenis=jenis;
        this.durasi=durasi;
    }
}

public class TesFilm
{
    public static void main(String[] args)
    {
        Film film1=new Film();
        film1.setFilm("Warkop DKI","Komedi",120);
    }
}
```

**Pembahasan program :** Kedua program tersebut memiliki hubungan **kelas induk dan kelas penguji (tester)**, di mana kelas `Film` berperan sebagai **kelas utama (model)** yang mendefinisikan struktur data dan perilaku sebuah objek film, sedangkan kelas `TesFilm` berfungsi untuk **menguji dan menjalankan** kelas tersebut. Kelas `Film` memiliki tiga atribut, yaitu `judul`, `jenis`, dan `durasi`, serta metode `setFilm()` yang

digunakan untuk mengisi nilai pada ketiga atribut tersebut menggunakan keyword `this` agar membedakan antara variabel lokal dan variabel milik objek. Di sisi lain, kelas `TesFilm` membuat objek `film1` dari kelas `Film` dan memanggil metode `setFilm("Warkop DKI", "Komedi", 120)` untuk memberikan nilai pada atributnya. Dengan demikian, program `TesFilm` menunjukkan bagaimana kelas `Film` digunakan dalam praktik untuk membuat dan mengatur data sebuah objek film.

### Output



```

C:\Windows\system32\cmd.e: X + v
Judul film :Warkop DKI
jenis film : Komedi
dursi : 120
Press any key to continue . . . |
  
```

### Pembahasan output :

- Nilai "Warkop DKI", "Komedi", dan 120 dikirim sebagai argumen ke metode `setFilm()`, lalu disimpan ke dalam variabel `judul`, `jenis`, dan `durasi` milik objek `film1`.
- Saat dicetak, program menampilkan isi atribut tersebut sesuai dengan data yang telah diatur.  
Dengan kata lain, output program menunjukkan bahwa proses penyimpanan dan pengambilan data dalam objek `Film` berjalan dengan benar.

## C. PEMBAHASAN TUGAS

### Tugas

<b>PROGRAM BIODATA</b> <pre> package mahasiswa; public class Biodata {     private String nama;     private String nim;     private String jurusan;     public Biodata(String nama, String nim, String jurusan) {         this.nama = nama;         this.nim = nim;         this.jurusan = jurusan;     }     public void tampilBiodata() {         System.out.println("=== BIODATA MAHASISWA ===");         System.out.println("Nama      : " + nama);         System.out.println("NIM       : " + nim);         System.out.println("Jurusan   : " + jurusan);     } }   </pre>
<b>PROGRAM NILAI</b> <pre> package mahasiswa; public class NilaiMahasiswa {     private double tgs;     private double uts;     private double uas;     public NilaiMahasiswa(double tgs, double uts, double uas) {         this.tgs = tgs;         this.uts = uts;         this.uas = uas;     } }   </pre>

```

    }
    public double hitungNilaiAkhir() {
        return (tgs * 0.3) + (uts * 0.3) + (uas * 0.4);
    }
    public String getGrade() {
        double nilaiAkhir = hitungNilaiAkhir();
        if (nilaiAkhir >= 85) return "A";
        else if (nilaiAkhir >= 70) return "B";
        else if (nilaiAkhir >= 50) return "C";
        else return "D";
    }
    public void tampilNilai() {
        double akhir = hitungNilaiAkhir();
        System.out.println("\n=== NILAI MAHASISWA ===");
        System.out.println("Nilai Tugas : " + tgs);
        System.out.println("Nilai UTS : " + uts);
        System.out.println("Nilai UAS : " + uas);
        System.out.println("Nilai Akhir : " + akhir);
        System.out.println("Grade : " + getGrade());
    }
}

```

#### PROGRAM MAIN

```

import mahasiswa.Biodata;
import mahasiswa.NilaiMahasiswa;
import java.util.Scanner;
public class MainMahasiswa {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Masukkan Nama : ");
        String nama = input.nextLine();
        System.out.print("Masukkan NIM : ");
        String nim = input.nextLine();
        System.out.print("Masukkan Jurusan : ");
        String jurusan = input.nextLine();
        System.out.print("Nilai Tugas (0-100): ");
        double tgs = input.nextDouble();
        System.out.print("Nilai UTS (0-100) : ");
        double uts = input.nextDouble();
        System.out.print("Nilai UAS (0-100) : ");
        double uas = input.nextDouble();
        Biodata bio = new Biodata(nama, nim, jurusan);
        NilaiMahasiswa nilai = new NilaiMahasiswa(tgs, uts, uas);
        System.out.println("\n-----");
        bio.tampilBiodata();
        nilai.tampilNilai();
    }
}

```

**Pembahasan program :** Program di atas merupakan satu kesatuan aplikasi Java yang terdiri dari **tiga kelas berbeda dalam dua file package, yaitu Biodata, NilaiMahasiswa, dan MainMahasiswa**, yang bersama-sama digunakan untuk menampilkan data dan nilai akhir mahasiswa.

Kelas Biodata (dalam package mahasiswa) berfungsi menyimpan informasi dasar mahasiswa seperti **nama, NIM, dan jurusan**, yang diinisialisasi melalui konstruktor dan ditampilkan menggunakan metode `tampilBiodata()`.

Kelas NilaiMahasiswa juga berada di package yang sama dan bertanggung jawab untuk **mengelola nilai tugas, UTS, dan UAS**. Nilai akhir dihitung dengan rumus  $(tgs * 0.3) + (uts * 0.3) + (uas * 0.4)$ , lalu dikategorikan ke dalam **grade A, B, C, atau D** berdasarkan skor akhir.

Kemudian, kelas MainMahasiswa sebagai program utama (driver) mengimpor kedua kelas tersebut menggunakan `import mahasiswa.*;`. Program ini meminta pengguna untuk memasukkan data diri dan nilai melalui input keyboard, lalu membuat objek Biodata dan NilaiMahasiswa. Setelah itu, kedua objek tersebut digunakan untuk **menampilkan biodata dan hasil perhitungan nilai akhir mahasiswa** ke layar.

#### Output

```
C:\Windows\system32\cmd.e: X + v
Masukkan Nama      : Fidelia
Masukkan NIM       : 245410012
Masukkan Jurusan   : Informatika
Nilai Tugas (0-100): 90
Nilai UTS (0-100)  : 80
Nilai UAS (0-100)  : 85

=====
=== BIODATA MAHASISWA ===
Nama      : Fidelia
NIM       : 245410012
Jurusan   : Informatika

=== NILAI MAHASISWA ===
Nilai Tugas : 90.0
Nilai UTS   : 80.0
Nilai UAS   : 85.0
Nilai Akhir : 85.0
Grade      : A
Press any key to continue . . .
```

**Pembahasan output** : output program menampilkan data diri mahasiswa dan hasil akhir perhitungan nilai akademiknya secara lengkap mencakup nilai tugas, UTS, UAS, nilai akhir, serta grade, yang semuanya dihasilkan dari interaksi antara dua kelas (Biodata dan NilaiMahasiswa) melalui program utama MainMahasiswa.

#### D. KESIMPULAN

Dari hasil praktikum **Package, Access Modifier, dan Static**, saya dapat menyimpulkan bahwa penggunaan konsep-konsep ini sangat penting untuk mengatur struktur dan keamanan program dalam Java. Dengan **package**, saya bisa mengelompokkan kelas-kelas yang saling berhubungan agar program lebih rapi, mudah dikelola, dan tidak terjadi konflik nama antar kelas. Melalui **access modifier** seperti `public`, `private`, dan `protected`, saya belajar bagaimana mengatur tingkat akses terhadap variabel dan metode agar data di dalam kelas tetap aman serta hanya bisa diakses sesuai kebutuhan. Sedangkan penggunaan **static** memungkinkan saya membuat anggota kelas (variabel atau metode) yang dapat digunakan tanpa harus membuat objek terlebih dahulu, sehingga lebih efisien dalam situasi tertentu. Secara keseluruhan, praktikum ini membantu saya memahami

bagaimana mengimplementasikan prinsip **enkapsulasi, modularitas, dan efisiensi** dalam pemrograman berorientasi objek menggunakan Java.