



**MODUL PERKULIAHAN**

# **PEMROGRAMAN IV (Object I)**

Manipulasi Objek dan Kelas Menggunakan Java

**Fakultas**  
Teknik

**Program Studi**  
Informatika

**Tatap Muka**

**11**

**Kode MK**  
06610005

**Disusun Oleh**

Ardiles Sinaga, S.T., M.T.  
Kurnia Jaya Eliazar, S.T., M.T.

## **Abstract**

Modul Pertemuan 11 Berisi Mengenai Manipulasi Objek dan Kelas menggunakan Java.

## **Kompetensi**

Mahasiswa memiliki kemampuan untuk menerapkan manipulasi objek dan kelas menggunakan Java dan menyelesaikan Studi Kasus.

### Definisi OOP

OOP (Object Oriented Programming) merupakan teknik membuat suatu program berdasarkan objek dan apa yang bisa dilakukan objek tersebut. OOP terdiri dari objek-objek yang berinteraksi satu sama lain untuk menyelesaikan sebuah tugas. Kode-kode di-breakdown agar lebih mudah di-manage. Breakdown berdasarkan objek-objek yang ada pada program tersebut. Dianjurkan diimplementasikan untuk program dengan berbagai ukuran karena lebih mudah untuk men-debug.

### Modifier

Modifier merupakan bentuk pengimplementasian konsep enkapsulasi. Dengan adanya modifier maka class, interface, method, dan variabel akan terkena suatu dampak tertentu.

Kelompok Modifier	Berlaku Untuk			Meliputi
	Class	Method	Variabel	
Access modifier	√	√	√	public, protected, private, dan friendly (default/ tak ada modifier).
Final modifier	√	√	√	final
Static modifier		√	√	static
Abstract modifier	√	√		abstract

Kelompok Modifier



Modifier	Class dan Interface	Method dan Variabel
<b>Default (tak ada modifier ) Friendly</b>	Dikenali di paketnya	Diwarisi subclass di paket yang sama dengan superclassnya. Dapat diakses oleh method-method di class-class yang sepaket.
<b>Public</b>	Dikenali di manapun	Diwarisi oleh semua subclassnya. Dapat diakses dimanapun.
<b>Protected</b>	Tidak dapat diterapkan	Diwarisi oleh semua subclassnya. Dapat diakses oleh method-method di class-class yang sepaket.
<b>Private</b>	Tidak dapat diterapkan	Tidak diwarisi oleh subclassnya Tidak dapat diakses oleh class lain.

Karakteristik Access Modifier

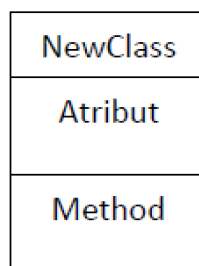
Modifier	Class	Interface	Method	Variabel
<b>Abstract</b>	Class dapat berisi method abstract. Class tidak dapat diinstantiasi Tidak mempunyai constructor	Optional untuk dituliskan di interface karena interface secara inheren adalah abstract.	Tidak ada method body yang didefinisikan. Method memerlukan class kongkrit yang merupakan subclass yang akan mengimplementasikan method abstract.	Tidak dapat diterapkan.
<b>Final</b>	Class tidak dapat diturunkan.	Tidak dapat diterapkan.	Method tidak dapat ditimpa oleh method di subclass-subclassnya	Berperilaku sebagai konstanta
<b>Static</b>	Tidak dapat diterapkan.	Tidak dapat diterapkan.	Mendefinisikan method (milik) class. Tidak memerlukan instant object untuk menjalankannya. Method ini tidak dapat menjalankan method yang bukan static serta tidak dapat mengacu variable yang bukan static.	Mendefinisikan variable milik class. Tidak memerlukan instant object untuk mengacunya. Variabel ini dapat digunakan bersama oleh semua instant objek.

### Class

Class adalah cetak biru (rancangan) atau prototipe atau template dari objek. Kita bisa membuat banyak objek dari satu macam class. Class mendefinisikan sebuah tipe dari objek. Di dalam class kita dapat mendeklarasikan variabel dan menciptakan objek (instansiasi). Sebuah class mempunyai anggota yang terdiri dari atribut dan method. Atribut adalah semua field identitas yang kita berikan pada suatu class.

### Class Diagram

Salah satu tools yang biasa digunakan untuk memodelkan/ menggambarkan/ merancang aplikasi berparadigma Object Oriented adalah UML atau Unified Modelling Language. Pada UML terdapat beberapa diagram yang mempunyai peran tersendiri. Salah satu diagram yang terdapat didalam UML dikenal dengan CLASS DIAGRAM. Diagram ini secara khusus menggambarkan sebuah class dan hubungan dengan class yang lain dalam sebuah sistem. CLASS DIAGRAM dan diagram-diagram yang lainnya diperkenalkan pada mata kuliah Analisis dan Perancangan Sistem Berorientasi Objek, sehingga dalam modul ini, tidak secara detail menggambarkan dan bagaimana merancang sebuah class diagram, hanya sebagai diagram konseptual untuk membantu membuat class yang sesungguhnya didalam program. Bentuk umum dari class diagram digambarkan sebagai berikut:



Class diagram mempunyai tiga bagian. Bagian atas atau pertama digunakan sebagai nama class/kelas. Aturan membuat nama class sesuai disesuaikan dengan aturan bahasa pemrograman yang digunakan. Biasanya nama dari sebuah class sesuai dengan kumpulan object yang ditampung oleh class tersebut.

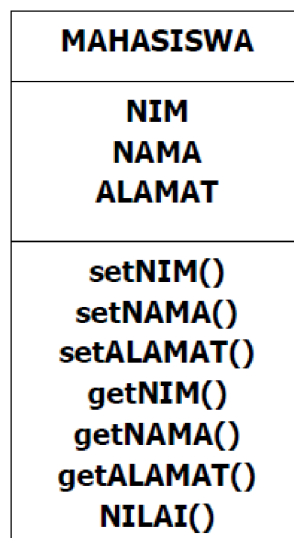
Bagian tengah atau kedua digunakan untuk menampung atribut dari class. Pada basis data atribut disebut juga sebagai FIELD. Pada bahasa java atribut disebut juga dengan INSTANCE VARIABLE.

Pada bagian yang bawah atau ketiga digunakan untuk menampung method dari class. Biasanya method berfungsi untuk memanipulasi data dari data member/atribut, tetapi tidak menutup kemungkinan sebuah method tidak berfungsi untuk memanipulasi data dari data member, tergantung dengan sifat method itu sendiri. Untuk method yang berfungsi untuk memanipulasi atribut, diberi nama dengan awalan set dan get disertai dengan INSTANCE VARIABLE nya.

Contoh :

Desain sebuah class yang berfungsi sebagai kumpulan data dari mahasiswa. Class tersebut memiliki atribut diantaranya NIM, NAMA dan ALAMAT. Class tersebut juga terdapat method untuk menentukan dan mengambil data NIM, NAMA dan ALAMAT. Selain itu, juga terdapat method untuk melihat nilai yang telah diperoleh mahasiswa.

Dari uraian tersebut, class diagram untuk mahasiswa dapat didesain sebagai berikut:



### Clas Diagram Menjadi Class Pada Java

Dari contoh pada point 2 diatas, apabila desain class mahasiswa tersebut diimplementasikan kedalam bahasa java, maka class mahasiswa tersebut menjadi sebagai berikut:

#### a. Class

Pada bagian atas atau pertama akan menjadi nama class. Aturan pemberian nama class pada java sama seperti memberi nama variable pada java. Class mahasiswa dapat digambarkan sebagai berikut:

```
class mahasiswa
{
    //statements
}
```

Tanda kurung kurawal penanda awal dan akhir dari sebuah class. Atribut dan method dari sebuah class akan ditulis diantara tanda kurung kurawal tersebut.

#### b. Atribut

Bagian kedua dari class diagram adalah atribut. Nama lain dari atribut adalah INSTANCE VARIABLE. Artinya, atribut pada konsep class diagram akan menjadi INSTANCE VARIABLE dalam pemrograman. Pada sebagian besar bahasa pemrograman, saat pendeklarasian variabel harus menyertakan apa tipe data dari variabel tersebut, artinya variabel tersebut digunakan untuk menampung data jenis apa.

Dari contoh program modul sebelumnya, LOCAL VARIABLE harus menyertakan tipe data yang digunakan. Sama seperti LOCAL VARIABLE, pembuatan INSTANCE VARIABLE juga harus menyertakan tipe datanya.

Pada contoh class mahasiswa diatas, mempunyai tiga atribut yaitu NIM, NAMA dan ALAMAT. Tipe data yang sesuai untuk tiga atribut tersebut adalah Tipe Data String karena menampung jenis data karakter.

Penulisan atribut dalam class mahasiswa digambarkan sebagai berikut:

```
class mahasiswa
{
    String NIM;
    String NAMA;
    String ALAMAT;
}
```

#### c. Method

Bagian ketiga dari class diagram adalah method. Untuk penulisan method dalam class ada yang diawali dengan void dan ada pula yang diawali dengan tipe data dari method

tersebut. Method yang diawali dengan void, menandakan bahwa method ini tidak mengembalikan nilai (return value), sedangkan method yang diawali dengan tipe data, menandakan ada nilai yang akan dikembalikan (return value). Method yang mempunyai return value, akan terdapat perintah return() didalam method tersebut. Biasanya method yang diawali dengan void adalah method yang bernama set.. dan method yang diawali dengan tipe data adalah metod yang bernama get.. Kesimpulannya, apakah dengan awalan void atau tipe data, tergantung dengan fungsi method itu sendiri.

Penulisan method dalam class mahasiswa sebagai berikut:

```
class mahasiswa
{
    void setNIM()
    { //statements
    }
    void setNAMA()
    { //statements
    }
    void setALAMAT()
    { //statements
    }
    String getNIM()
    { //statements
        return(..);
    }
    String getNAMA()
    { //statements
        return(..);
    }
    String getALAMAT()
    { //statements
        return(..);
    }
    int NILAI()
    { //statements
        return(..);
    }
}
```

d. Class lengkap dengan atribut dan method.

Bentuk lengkap class mahasiswa yang mempunyai atribut dan method digambarkan sebagai berikut:



```

class mahasiswa
{
    String NIM;
    String NAMA;
    String ALAMAT;
    void setNIM()
    { //statements
    }
    void setNAMA()
    { //statements
    }
    void setALAMAT()
    { //statements
    }
    String getNIM()
    { //statements
        return(..);
    }
    String getNAMA()
    { //statements
        return(..);
    }
    String getALAMAT()
    { //statements
        return(..);
    }
    int NILAI()
    { //statements
        return(..);
    }
}

```

Catatan:

Bentuk class diatas baru menggambarkan apabila sebuah class diagram yang diimplementasikan menjadi class pada java, tetapi bentuk class tersebut belum dapat di compile dan di eksekusi. Class tersebut merupakan implementasi dari class diagram.

Contoh:

- Class Manusia → Obyek :
  - Data : nama (String), umur (integer).

Script:





```
public class Manusia {  
    //definisi atribut  
    private String nama;  
    private int umur;  
    //definisi method  
    public void setName(String a){  
        nama=a;  
    }  
    public String getName(){  
        return nama;  
    }  
    public void setUmur(int a){  
        umur=a;  
    }  
    public int getUmur(){  
        return umur;  
    }  
}
```



## Instansiasi Objek

Instansiasi objek adalah proses membuat/menciptakan sebuah objek dari sebuah class. Pembuatan objek adalah dengan cara membuat sebuah variabel yang akan menunjuk ke objek tersebut. Variabel seperti ini disebut dengan variabel objek. Bentuk umum dari instansiasi object pada java adalah sebagai berikut :

```
namaclass namavariabel = new namaclass();
```

Sebuah objek diciptakan di luar class objek itu sendiri. Pada java, biasanya objek dari sebuah class diciptakan pada class yang mempunyai main() method (disebut saja sebagai class main() ). Melalui class main() ini objek akan menggunakan seluruh atribut dan method dari classnya.

Contoh : berikut adalah contoh pembuatan sebuah objek pada class main() .

```
class testMahasiswa
{
    public static void main (String args[])
    {
        mahasiswa mhs1 = new mahasiswa();
    }
}
```

Proses instansiasi tersebut dapat diartikan membuat sebuah objek bernama mhs1 dari class mahasiswa, objek dibuat dari class main() bernama testMahasiswa.

## Penggunaan Method Melalui Sebuah Objek

Method yang digunakan sebuah objek dipanggil dengan cara menyebutkan objeknya diikuti dengan method yang akan digunakan.

Pada contoh berikut akan diperkenalkan method tanpa parameter dan method dengan parameter. Pada contoh berikut, satu file java terdapat 2 buah class, yaitu class yang akan dibuatkan objeknya dan class main(). Apabila terdapat kondisi seperti ini, nama file java dinamai sama dengan nama class main()nya.

- a. Method tanpa parameter

Pada contoh berikut, pada class mahasiswa akan dibuat sebuah method bernama hello(). Method ini tidak mempunyai return value, sehingga awalnya menggunakan void. Method ini akan dipanggil oleh sebuah objek dari class main(). Objek yang dibuat bernama mhs1.

Nama file: testMethodNon.Java

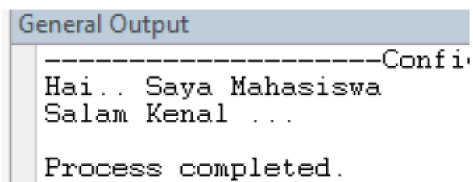
Sintaks program:

```
class mahasiswa
{
    void hello()
    {
        System.out.println("Hai.. Saya Mahasiswa");
        System.out.println("Salam Kenal ...");
    }
}

class testMethodNon
{
    public static void main (String[] args)
    {
        mahasiswa mhs1=new mahasiswa();

        mhs1.hello();
    }
}
```

Output program :



General Output

```
-----Confir
Hai.. Saya Mahasiswa
Salam Kenal ...
Process completed.
```

Penjelasan :

- mahasiswa mhs1 = new mahasiswa(), adalah proses instansiasi objek dengan nama mhs1.
- mhs1.hello(), adalah perintah yang menunjukkan objek mhs1 memanggil/ menggunakan method hello().

#### b. Method menggunakan parameter

Parameter pada method berfungsi untuk menampung data yang akan dikirimkan ke dalam method ataupun mengirimkan data dari method ke bagian pemanggilnya. Parameter sebenarnya adalah variabel, sehingga perlu didefinisikan juga tipe datanya.

Tidak ada batasan sebuah method dapat memiliki beberapa parameter. Berikut adalah contoh sebuah class dengan method yang menggunakan satu buah parameter.

Nama file: testMethodParameter.java

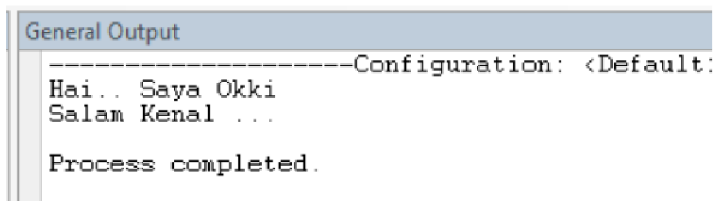
Sintaks program:

```
class mahasiswa
{
    void hello(String Nm)
    {
        System.out.println("Hai.. Saya "+Nm);
        System.out.println("Salam Kenal ...");
    }
}

class testMethodParameter
{
    public static void main (String[] args)
    {
        mahasiswa mhs1=new mahasiswa();

        mhs1.hello("Okki");
    }
}
```

Output program :



```
General Output
-----Configuration: <Default:
Hai.. Saya Okki
Salam Kenal ...
Process completed.
```

Penjelasan :

- void hello(String Nm), adalah sebuah method yang memiliki sebuah parameter bernama Nm dengan tipe data String.
- mhs1.hello("Okki") adalah objek mhs1 menggunakan method hello dengan mengirimkan kata "Okki" ke parameter Nm.

## Static Method

Static Method adalah method yang dimiliki oleh class, bukan milik objeknya. Berbeda dengan contoh penggunaan method diatas, penggunaan static method dalam pemrograman,

dapat dipanggil langsung tanpa menggunakan objek. Tanpa harus membuat objek untuk memanggil method tersebut. Selain method, atribut juga dapat diberlakukan menjadi milik class, menjadi static variabel. Berikut adalah contoh penerapan static method dalam pemrograman.

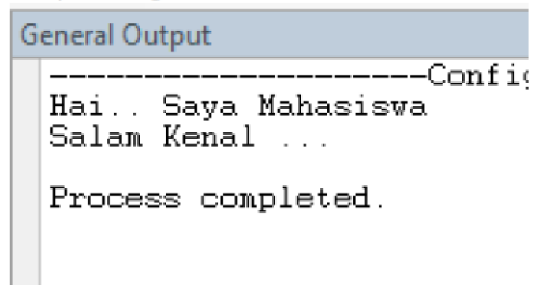
Nama file: testStaticMethod.java

Sintaks program:

```
class mahasiswa
{
    static void tampil()
    {
        System.out.println("Hai.. Saya Mahasiswa");
        System.out.println("Salam Kenal ...");
    }
}

class testStaticMethod
{
    public static void main (String[] args)
    {
        mahasiswa.tampil();
    }
}
```

Output Program ;



Penjelasan Program:

- Pada class mahasiswa terdapat sebuah method berjenis static dengan nama tampil(). Ciri method tersebut bertipe static adalah terdapat keyword static pada saat pendeklarasian methodnya.
- Pada class testStaticMethod, tidak dibuat sebuah objek dari class mahasiswa. method tampil() langsung dipanggil dengan menyebutkan nama classnya.

### Penggunaan Atribut Melalui Sebuah Objek

Atribut adalah ciri sebuah objek. Pada pemrograman atribut berfungsi untuk menampung data dari objek. Pada pemrograman, atribut adalah INSTANCE VARIABLE (silahkan baca modul sebelumnya)

Sebuah atribut dapat digunakan sebuah objek dengan cara diakses melalui method. Ada beberapa method yang berfungsi untuk memanipulasi data dari atributnya. Ciri method tersebut biasanya diawali dengan set atau get kemudian diikuti dengan nama atributnya. Method set dan get biasanya berpasangan untuk memanipulasi sebuah atribut. Perbedaan Method set dan get adalah:

- Method yang menggunakan set digunakan untuk menerima data yang dikirimkan oleh objek ke atribut.
- Method yang menggunakan get digunakan untuk mengirimkan data dari atribut ke objek.
- Method set menggunakan parameter dan tidak mempunyai return value, sehingga awalan method menggunakan void.
- Method get tidak menggunakan parameter dan mempunyai return value, sehingga awalan method menggunakan tipe data methodnya.

Berikut contoh program dengan sebuah class yang mempunyai sebuah atribut dan method untuk digunakan objeknya.

a. Atribut dengan method set

Nama file: testMethodSet.Java

Sintak program:



```

class mahasiswa
{
    String Nama;
    void setName(String Nm)
    {
        Nama = Nm;
        System.out.println("Hai.. Saya "+Nama);
        System.out.println("Salam Kenal ...");
    }
}

class testMethodSet
{
    public static void main (String[] args)
    {
        mahasiswa mhs1=new mahasiswa();

        mhs1.setName("Okki");
    }
}

```

Penjelasan program :

- Parameter yang dikirim oleh objek mhs1 berupa kata “Okki”. Kata ini ditampung ke parameter Nm pada method setName, dari parameter Nm kata tersebut dimasukkan ke atribut Nama. Melalui atribut Nama kata yang telah dikirim akan ditampilkan.
- Method setName menggunakan awalan void, seluruh statements langsung dikerjakan pada method tersebut tanpa dikembalikan ke class main().

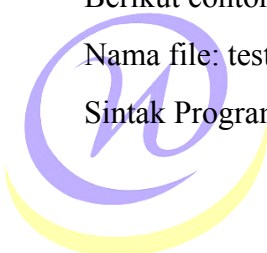
#### b. Mengenal keyword this

Salah satu kelebihan dari java adalah dapat membuat variabel parameter sebuah method dengan nama yang sama dengan atributnya (instance variable). Cara ini bisa dilakukan dikarenakan java menyediakan sebuah keyword yang dapat membedakan antara variabel parameter dengan atribut. Keyword tersebut dinamai dengan “this”. Variabel yang menggunakan keyword this, menunjukkan bahwa variabel tersebut adalah atribut/instance variabel dari class nya.

Berikut contoh program menggunakan keyword this.

Nama file: testMethodThis.java

Sintak Program:



```

class mahasiswa
{
    String Nama;
    void setName (String Nama)
    {
        this.Nama = Nama;
        System.out.println("Hai.. Saya "+Nama);
        System.out.println("Salam Kenal ...");
    }
}

class testMethodThis
{
    public static void main (String[] args)
    {
        mahasiswa mhs1=new mahasiswa();

        mhs1.setName("Okki");
    }
}

```

Penjelasan program :

- Nama variabel parameter dibuat sama dengan nama atribut classnya yaitu Nama. Keyword this menunjukkan perbedaan keduanya.
- this.nama menunjukkan variabel tersebut adalah atribut / instance variabel dari class mahasiswa.

#### c. Atribut dengan Method get

Berikut adalah contoh method get yang akan mengembalikan nilai (return value) pada objek.

Nama File: testMethodGet.java

Sintaks program:





```

class mahasiswa
{
    String Nama = "Okki";
    String getName()
    {
        return (Nama);
    }
}

class testMethodGet
{
    public static void main (String[] args)
    {
        mahasiswa mhs1=new mahasiswa();

        System.out.println("Hai.. Saya "+mhs1.getName());
        System.out.println("Salam Kenal ...");

    }
}

```

Penjelasan program :

- Pada contoh program ini, kata “Okki” tidak dikirim melalui parameter, tetapi langsung didefinisikan pada atributnya.
- Tipe data yang digunakan pada method getName adalah String, dikarenakan nilai/data yang dikembalikan (return) bertipe karakter, mengikuti tipe data dari atributnya.
- Return menunjukkan data yang dikembalikan pada objek yang meminta, pada contoh ini, data diambil melalui method getName().

d. Atribut dengan method set dan get.

Berikut adalah contoh program dengan class yang mempunyai dua buah method yaitu method set dan method get. Program pada umumnya menggunakan method set dan get secara berpasangan apabila class tersebut membutuhkan method untuk memanipulasi atribut.

Nama File: testMethodSetGet.Java

Sintaks Program:



```

class mahasiswa
{
    String Nama;

    void setNama(String Nama)
    {
        this.Nama = Nama;
    }
    String getNama()
    {
        return (Nama);
    }
}

class testMethodSetGet
{
    public static void main (String[] args)
    {
        mahasiswa mhs1=new mahasiswa();

        mhs1.setNama("Okki");

        System.out.println("Hai.. Saya "+mhs1.getNama());
        System.out.println("Salam Kenal ...");

    }
}

```

Penjelasan Program :

- Kata “Okki” dimasukkan ke atribut Nama pada class mahasiswa melalui method setNama().
- Setelah dimasukkan ke atribut Nama, kata “Okki” dikirimkan kembali ke objek menggunakan method getNama().



## Daftar Pustaka

- [1] Deitel P.J., Deitel H.M., “Java: How to Program”, Prentice Hall, 2004
- [2] Holmes B. J., Joyce D.T., “Object-Oriented Programming With Java, Second Edition”, JONES AND BARTLETT PUBLISHERS, 2001
- [3] Keogh J., “JAVA DEMYSTIFIED”, McGraw-Hill/Osborne, 2004
- [4] Wu C.T., “AN INTRODUCTION TO OBJECT-ORIENTED PROGRAMMING WITH JAVA™, FIFTH EDITION”, McGraw-Hill, 2010
- [5] Wu C.T., “A COMPREHENSIVE INTRODUCTION TO OBJECT-ORIENTED PROGRAMMING WITH JAVA”, McGraw-Hill, 2008
- [6] Poo D., Kiong D., Ashok S., “Object-Oriented Programming and Java Second edition”, Springer, 2008
- [7] Sintes T., “Sams Teach Yourself Object Oriented Programming in 21 Days”, Sam Publishing, 2002

