MODUL

PEMROGRAMAN BERORIENTASI OBJEK

Disusun oleh :

TIM PENYUSUN

SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER

AKAKOM

YOGYAKARTA

2019

KATA PENGANTAR

Puji Syukur kepada Allah SWT karena modul Pemrograman Berorientasi Obyek ini dapat terselesaikan. Modul ini dibuat karena tuntutan kurikulum baru atau disebut sebagai kurikulum 2019 STMIK AKAKOM yang mulai diberlakukan mulai tahun akademik 2019/2020.

Modul ini membahas tentang Kelas, Obyek, atribut, method, konstruktor, enkapsulasi dan komposisi, pewarisan, package, polimorfisme, kelas abstrak dan interface, exception handling dan tread.

Tentu saja dengan segala keterbatasan yang ada, modul ini jauh dari sempurna. Untuk itu masukan dan saran untuk perkembangan modul ini sangat diharapkan.

Akhirnya semoga modul ini berguna dalam membantu mahasiswa untuk mempelajari tentang Pemrograman Berorientasi Obyek.

Yogyakarta, Agustus 2019.

Penulis

MODUL 1
KELAS dan OBJEK

CAPAIAN PEMBELAJARAN

Dapat membaca class diagram sederhana, Dapat menyebutkan bagian-bagian kelas, Dapat menjelaskan sintak kelas, Dapat membuat kelas sederhana, Dapat membedakan antara kelas dan objek, Dapat menciptakan Objek

KEBUTUHAN ALAT/BAHAN/SOFTWARE

1. Textpad sebagai teks editor / netbeans
2. JDK sebagai kompiler java

DASAR TEORI

Sebuah sistem yang dibangun berdasarkan metoda berorientasi objek adalah sebuah sistem yang komponennya di enkapsulasi menjadi kelompok data dan fungsi, yang dapat mewarisi atribut dan sifat dari komponen lainnya, dan komponen-komponen tersebut saling berinteraksi satu sama lain.

Bahasa Pemrograman yang berorientasi OBJEK memiliki kemampuan dalam pengelolaan program yang lebih diarahkan pada pembentukan objek. Dengan menerapkan konsep ini program akan lebih mudah untuk dikembangkan karena sifatnya yang lebih modular.

Dalam konsep object oriented akan kita temukan kata object dan class, class merupakan pola / template yang menggambarkan kumpulan object yang mempunyai sifat yang sama, perilaku, atau disebut dengan himpunan object sejenis. Sementara object adalah implementasi dari class. Adapun proses pembuatan obyek dari kelas disebut instantiasi atau implementasi dari kelas.

Tabel 1 adalah contoh ilustrasi kelas dan objek.

Tabel 1 Contoh class Mobil dan object-object nya

Kelas Mobil		Objek Mobil A	Objek Mobil B
Instan Variabel	nomor Plat	AB 3313 SY	AB 1234 AT
	Warna	Biru	Merah
	Manufaktur	Mitsubishi	Toyota
	Kecepatan	50 km/h	100 km/h
Instan Metode	method akselerasi		
	method belok		
	method rem		

Membuat Class dan Obyek dalam Java

- Elemen-elemen dasar dalam mendefinisikan kelas :
  1. Field (variabel) : menyimpan data untuk setiap objek (implementasi dari atribut)
  2. Constructor : setup objek di awal
  3. Method : implementasi perilaku objek

```
class NamaKelas
{
        Fields
        Constructor
        Methods
}
```

Untuk membuat obyek digunakan kata kunci new :

NamaKelas NamaObyek = new NamaKelas();

Untuk memanggil method dapat digunakan sintaks sebagai berikut:

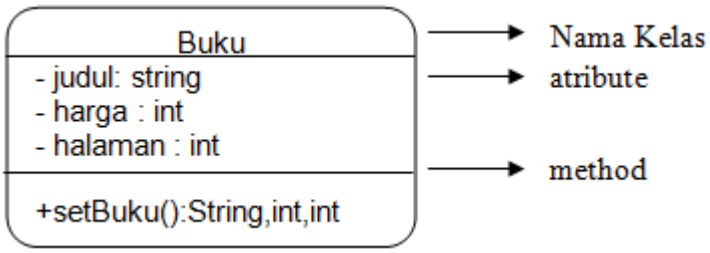
namaObyek.nama_method([parameter]);

Terdapat 3 konsep penting dalam OOP, yaitu : enkapsulasi, pewarisan, dan polymorfisme. Enkapsulasi bertujuan untuk menyembunyikan kerumitan implementasi dan menyediakan interface bagi pengguna. Selain itu enkapsulasi juga berguna untuk proteksi data (read/write). Pewarisan adalah salah satu cara untuk menggunakan ulang kode-kode yang telah dibuat sebelumnya. Sedangkan polymorfisme adalah kemampuan obyek untuk bereaksi secara berbeda terhadap “pesan yang sama”, tergantung obyek yang menerima pesan tersebut. Kemampuan ini berkat adanya konsep method overriding dan method overloading.

PRAKTIK

Praktik 1 . Membuat kelas

Berikut ini adalah contoh class diagram untuk kelas Buku



Kode program:

```
class  Buku
{
      String judul;
      int harga;
      int halaman;

      public void setBuku(String judul,int harga,int
halaman)
      {
            this.judul=judul;
            this.harga=harga;
            this.halaman=halaman;
      }
      public void tampil()
      {
            System.out.println("Judulbuku :"+judul);
            System.out.println("Harga buku : "+harga);
            System.out.println("Jumlah halaman : "+halaman);
      }
}
```

Praktik 2. Membuat dan Menggunakan Obyek

Kelas Buku di atas hanya merupakan deklarasi dari obyek buku. Agar dapat digunakan maka Kelas Buku tersebut harus diinstantiasi. Dari satu buah kelas dapat dibuat banyak obyek. Obyek-obyek tersebut dapat dibedakan dengan suatu pengenal yang berupa nama.

```
public class ObyekBuku
{
   public static void main(String[] args)
   {
          Buku buku1=new Buku();
          buku1.setBuku("PBO",75000,200);
          buku1.tampil();
   }
}
```

LATIHAN

1. Tambahkan 2 buah objek dari kelas Buku !
2. Panggil method setBuku() dan tampil()
3. Buatlah kelas Pegawai !
4. Buatlah 2 buah objek dari kelas Pegawai !

TUGAS

1. Implementasikan ke dalam program dari contoh Tabel 1 pada dasar teori !
2. Buatlah kelas handpone!
3. Buatlah 2 buah objek dari kelas handphone !

REFERENSI

Wagito, Sumiyatun, Fx. Henry Nugroho. 2015. Modul Praktikum Pemrograman Berorientasi Objek. STMIK AKAKOM. Yogyakarta

MODUL 2
ATRIBUT

CAPAIAN PEMBELAJARAN

1. Dapat mendefinisikan atribut kelas,
2. Dapat membuat dan memanggil atribut,
3. Dapat membedakan identifier dan konstanta,
4. Dapat mengidentikasi atribut suatu kelas dan mengimplementasi menjadi kelas

KEBUTUHAN ALAT/BAHAN/SOFTWARE

1. Textpad / netbeans
2. JDK

DASAR TEORI

Variabel

Variabel adalah wadah yang menampung nilai-nilai yang digunakan dalam program Java. Setiap variabel harus dideklarasikan menggunakan tipe data. Misalnya, variabel dapat dinyatakan menggunakan salah satu dari delapan tipe data primitif: byte, short, int, long, float, double, char atau boolean. Dan, setiap variabel harus diberi nilai awal sebelum bisa digunakan.

contoh:
int myAge = 21;
Variabel "myAge" dinyatakan sebagai tipe int data dan diinisialisasi ke nilai 21.

Dalam definisi kelas, ada tiga jenis variabel.

- instance variables: Setiap metode dalam definisi kelas dapat mengakses variabel tersebut
- Parameter variabel: Hanya metode dimana parameter muncul dapat mengakses variabel tersebut. Ini adalah bagaimana informasi akan diteruskan ke objek.
- local variables: Hanya metode dimana parameter muncul dapat mengakses variabel ini lokal. Variabel ini digunakan untuk menyimpan hasil antara.

Konstanta

Sebuah konstanta adalah suatu tempat untuk menampung data yang nilainya selalu tetap dan tidak pernah berubah. Di java, kata kunci final dapat digunakan dengan tipe data primitif dan objek tetap (misalnya, String) untuk membuat konstanta.

Misal

final int DAYS_IN_JANUARY = 31;

PRAKTIK

Praktik 1: membuat kelas Lingkaran

```
public class Lingkaran
{
  final double phi = 3.14;
  double jari;

  public void setJari(double jari0)
  {
    this.jari=jari0;
  }

  public double jari()
  {
    return(jari);
  }

  public  double luas()
  {
    double luas0;
    luas0=phi*jari*jari;
    return(luas0);
  }

  public void tampil()
  {
```

```
    System.out.println("jari jari: "+jari);
    System.out.println("luas: "+luas());
  }

}
```

Dari praktik 1 manakah yang merupakan atribut dan mana yang merupakan konstanta !

Praktik 2. Menggunakan kelas Lingkaran

```
public class Main1
{
  public static void main(String[] args)
  {
    Lingkaran a;
    a=new Lingkaran();
    a.setJari(10.00);
    a.tampil();
  }
}
```

Praktik 3. Menggunakan kelas Lingkaran

```
public class Main2
{
  public static void main(String[] args)
  {
    int i;
    Lingkaran[] a;
    a=new Lingkaran[5];

    for(i=0;i<5;i++)
    {
      a[i]=new Lingkaran();
    }
    a[0].setJari(00.00);
    a[1].setJari(10.00);
    a[2].setJari(20.00);
    a[3].setJari(30.00);
    a[4].setJari(40.00);
    for(i=0;i<5;i++)
    {
      System.out.println("Lingkaran ke: "+i);
      a[i].tampil();
    }
  }
}
```

LATIHAN

1. Modifkasi Kelas Buku pada modul 1 dengan menambahkan 2 atribut lagi !
2. Buat kelas mahasiswa dengan 5 atribut !

TUGAS

1. Jelaskan perbedaan antara vriabel dan konstanta !
2. Buatlah kelas persegi panjang lengkap dengan atributnya untuk mentukan luas dan keliling persegi panjang !

REFERENSI

Wagito, Sumiyatun, Fx. Henry Nugroho. 2015. Modul Praktikum Pemrograman Berorientasi Objek. STMIK AKAKOM. Yogyakarta

MODUL 3
METHOD

CAPAIAN PEMBELAJARAN

Dapat membuat dan menggunakan berbagai method

KEBUTUHAN ALAT/BAHAN/SOFTWARE

1. Textpad
2. JDK

DASAR TEORI

Method

Sebuah method menjelaskan behaviour dari sebuah object. Method juga dikenal sebagai fungsi atau prosedur. Pendeklarasian method biasa dituliskan seperti berikut ini :

```
<modifier> <returnType> <name>(<parameter>*) {
        <statement>*
    }
```

dimana,

<modifier> dapat menggunakan beberapa *modifier* yang berbeda

<returnType> dapat berupa seluruh tipe data, termasuk *void*

<name> *identifier* atas *class*

<parameter> ::= <tipe_parameter> <nama_parameter>[,]

Method Tanpa Return Value

Jenis method ini ditandai dengan *return type* yang berupa *void* dan pada bagian *statement* tidak terdapat keyword **return**. Pada pemrograman berorientasi obyek method jenis ini digunakan untuk membuat method *mutator*. Nama untuk method *mutator* sebaiknya diawali dengan kata set, misalnya: setNama(), setAlamat().

Method Dengan Return Value

Jenis method ini ditandai dengan *return type* selain *void* dan pada bagian *statement* terdapat keyword **return**. Pada pemrograman berorientasi obyek method jenis ini digunakan untuk membuat method *acessor*, dimana method *acessor* fungsinya adalah untuk membaca/mendapatkan nilai suatu atribut. Nama untuk method *acessor* sebaiknya diawali dengan kata get, misalnya: getNama(), getAlamat().

Method Overloading

Bahasa java mendukung method *overloading* , java dapat membedakan beberapa *method* dengan nama yang sama di dalam sebuah kelas namun parameternya berbeda. Hal ini sangat menguntungkan karena memudahkan kita dalam mengingat nama method, bayangkan bila program pada class Gambar harus diberi nama drawInterger(int i), drawString(String s), drawDouble(double d). Method *overloading* dibedakan oleh jumlah dan jenis tipe data parameternya.

Contoh method overloading :

```
public class Gambar{
 public void draw(int i){
 .....
 }
 public void draw(String s){
 .....
 }
 public void draw(double d){
```

```
  .....
  }
  public void draw(int i, double d){
  .....
  }
}
```

PRAKTIK

1. Praktik 1 (membuat method mutator dan method acessor)

```
public class Mahasiswa{
 private String nim,nama;

 //method mutator
 public void setNim(String nim) {
    this.nim=nim;
    }
 public void setNama(String nama) {
    this.nama=nama;
    }

 //method acessor
 public String getNim()  {
    return nim;
    }
 public String getNama()  {
    return nama;
    }

 public static void main(String args[]){
  Mahasiswa obj=new Mahasiswa();
  obj.setNim("175410001");
  obj.setNama("Azkiya");

  System.out.println("==Data==");
  System.out.println("Nim  : "+ obj.getNim());
  System.out.println("Nama : "+ obj.getNama());
  }
```

```
}
```

2. Praktik 2 (Memuat method overloading)

```
class Hitung{
 private int a,b,c,d;

void tambah(int a,int b){
     System.out.println(a+b);
 }
 void tambah(int a,int b,int c){
  System.out.println(a+b+c);
 }
 void tambah(int a,int b,int c, int d){
   System.out.println(a+b+c+d);
 }

  public static void main(String args[]){
  Hitung obj=new Hitung();
  obj.tambah(10,10,10);
  obj.tambah(20,20);
  obj.tambah(20,5,10,20);
 }
}
```

3. Praktik membuat method overloading untuk dengan perbedaan tipe parameter

```
class Calculation2{
 private int a,b;
 private double c,d;

  void sum(int a,int b) {
    System.out.println(a+b);
  }
  void sum(double c,double d) {
    System.out.println(c+d);
 }

  public static void main(String args[]){
  Calculation2 obj=new Calculation2();
  obj.sum(10.5,10.5);
  obj.sum(20,20);
 }
}
```

LATIHAN

1. Modifikasi praktik 3 dengan menambahkan method sum dengan tipe parameter yang berbeda selanjutnya panggil melali main program !
2. Modifikasi praktik 1 dengan menambahkan method TampilkanData yang digunakan untuk menampilkan data nim dan nama !
3. Buatlah kelas lengkap dengan atribut dan oprasi untuk menentukan luas seitiga !

TUGAS

1. Buatlah program lengkap dengan atribut dan oprasi yang digunakan untuk meghitung nilai pangkat dari suatu bilangan !
2. Soal dar dosen pengampu

REFERENSI

Wagito, Sumiyatun, Fx. Henry Nugroho. 2015. Modul Praktikum Pemrograman Berorientasi Objek. STMIK AKAKOM. Yogyakarta

MODUL 4
Konstruktor

CAPAIAN PEMBELAJARAN

Dapat membuat dan menggunakan konstruktor

KEBUTUHAN ALAT/BAHAN/SOFTWARE

1. Textpad
2. JDK

DASAR TEORI

Konstruktor adalah sebuah tipe khusus dari method yang digunakan untuk membuat dan menginisialisasi sebuah object baru.

Berikut ini adalah *property* dari konstruktor :

1. konstruktor memiliki nama yang sama dengan class
2. konstruktor tidak memiliki return value
3. konstruktor tidak dapat dipanggil secara langsung, namun harus dipanggil dengan menggunakan operator ***new*** pada pembentukan sebuah *class*.

Untuk mendeklarasikan konstruktor, kita tulis

```
<modifier> <className> (<parameter>) {

<statement>
```

}

Perbedaan method biasa dengan konstruktor adalah bahwa konstruktor harus memiliki nama yang sama dengan nama classnya dan tidak memiliki nilai kembalian (tipe-data).
Konstruktor dijalankan pada saat sebuah object diinisialisasi (menggunakan kata new).

Konstruktor default

Apabila anda tidak mendeklarasikan satu pun konstruktor, maka Java secara otomatis menambahkan konstruktor default ke dalam class yang kita buat walaupun tidak kelihatan pada kode program. Apabila kita mendeklarasikan satu atau lebih konstruktor maka java tidak akan menambahkan kostruktor default.

Konstruktor overloading

Pada konstruktor juga berlaku overloading, artinya boleh mendeklarasikan lebih dari satu konstruktor, asalkan memiliki parameter yang berbeda – beda.

Pada konstruktor :

1. Memiliki konsep yang sama dengan overloading method.
2. Dibedakan berdasarkan paramater (jumlah atau tipe data).

Pemanggilan konstruktor tergantung pada instansiasi objek.

PRAKTIK

1. Praktik 1 Membuat konstruktor

```
public class Titik{
      int x;
      int y;
      public Titik(){
      System.out.println("Konstruktor titik dijalankan!");
      }
}
```

2. Praktik 2 Membuat obyek dan pemanggilan konstruktor

```
public class TesTitik {
   public static void main(String[] args) {
        Titik a=new Titik();
    }
}
```

3. Praktik 3 Membuat konstruktor overloading dan pemanggilan konstruktor

   Tambahkan konstruktor baru pada class Titik

```
public Titik(int x, int y) {
        this.x = x;
        this.y = y;
        System.out.println("Konstruktor titik 2
dijalankan!");
    }
```

   Tambahkan skrip berikut pada class TesTitik

```
Titik b=new Titik(10,10);
```

4. Praktik 4 Overloading konstruktor

```
public class Pegawai
{
      String NamaPegawai;
      int IdPegawai;
      String PosisiPegawai;

Pegawai (String nama,int Id, String posisi) // konstruktor
      {
            NamaPegawai = nama;
            IdPegawai = Id;
            PosisiPegawai = posisi;
      }
      Pegawai () //konstruktor
            {
                  NamaPegawai = "Azkiya";
                  IdPegawai = 2514;
                  PosisiPegawai = "Staf pengajar";
      }
      void Show()
      {
            System.out.println("Informasi Pegawai");
            System.out.println("Nama     : "+NamaPegawai);
            System.out.println("Id       : "+IdPegawai);
            System.out.println("Posisi   : "+PosisiPegawai);
      }
```

```
      public static void main(String args[])
      {

            Pegawai pegawai1 = new Pegawai();
            Pegawai pegawai2 = new Pegawai("Zahwa",3313,"Staf
Akademik");
            pegawai1.Show();
            pegawai2.Show();
      }
}
```

LATIHAN

1. Modifikasi class buku pada modul 1 dengan menggunakan kontruktor !

TUGAS

1. Buatlah class Komputer lengkap dengan atribut, method dan konstruktor !

REFERENSI

Wagito, Sumiyatun, Fx. Henry Nugroho. 2015. Modul Praktikum Pemrograman Berorientasi Objek. STMIK AKAKOM. Yogyakarta