Modul 02 Class
dan Object

1. Kompetensi

- Mahasiswa dapat memahami deskripsi dari class dan object
- Mahasiswa memahami implementasi dari class
- Mahasiswa dapat memahami implementasi dari attribute
- Mahasiswa dapat memahami implementasi dari method
- Mahasiswa dapat memahami implementasi dari proses instansiasi
- Mahasiswa dapat memahami implementasi dari try-catch
- Mahasiswa dapat memahami proses pemodelan class diagram menggunakan *UML*

2. Pendahuluan

2.1 Class dan Object

Pada pertemuan sebelumnya anda sudah diberikan banyak penjelasan secara *semantic* (makna) mengenai deskripsi dari class dan object. Secara singkat class adalah abstraksi dari sebuah object (nyata ataupun tdk nyata) (roger s pressman). Apabila kita ingin membuat class mahasiswa, maka kita perlu melakukan abstraksi (mengindikasi bagian – bagian penting yang merepresentasikan benda itu sendiri) dari object mahasiswa itu sendiri. Contoh salah satu attribute yang mengidentifikasi jika seseorang itu mahasiswa adalah **Nim** (Nomor Induk Mahasiswa), dan **Nim** tidak akan anda temui pada attribute dosen. Selain attribute abstraksi juga digunakan untuk behavior (perilaku) , contoh salah satu perilaku yang bisa dilakukan oleh mahasiswa adalah mengikuti **UAS**, dan anda juga tidak akan pernah menemui perilaku tersebut pada object dosen. Oleh karena itu sangat mudah untuk seorang perancang system dalam memodelkan sebuah class dari sebuah object tertentu.

Setelah kita memahami secara semantic pengertian dari class dan object, maka langkah selanjutnya adalah bagaimana cara melakukan implementasi class pada pendekatan Object Oriented Programming, terutama pada bahasa pemrograman java. Berikut adalah sintaks dari deklarasi class pada pemrograman java :

```
<modifier> class <nama_class> {
      //deklarasi atribut dan method
}
```

Aturan penulisan class adalah sebagai berikut:

1. Berupa kata benda,
2. Diawali dengan **HURUF BESAR,**
3. Jika terdiri dari lebih dari 1 kata, maka antar kata satu dengan kata yang lain digandeng, dan tiap huruf awal dari tiap kata menggunakan **HURUF BESAR**.

Untuk Access Modifier tidak dibahas pada modul ini, melainkan akan dibahas pada modul berikutnya.

Contoh deklarasi class:

```
public class Mahasiswa{

}
```

2.2 Attribute

Untuk melakukan pendeklarasian *attribute* dapat dilakukan dengan sintaks sebagai berikut:

```
<modifier> <tipe> <nama_atribut> ;
```

Aturan penulisan atribut adalah sebagai berikut:

1. Berupa kata benda,
2. Diawali dengan HURUF KECIL,
3. Jika terdiri dari 2 atau lebih kata, kata pertama diawali HURUF KECIL sedangkan kata selanjutnya diawali HURUF BESAR. Dan antar kata disambung tidak (dipisah).

Contoh deklarasi ***attribute***:

```
public class Mahasiswa{

  public int nim;
  public String nama;
  public String alamat;
  public float luas;

}
```

Nb : *attribute* yang dituliskan dengan **huruf tebal**.

2.3 Method

Method adalah suatu blok dari program yang berisi kode program nama dan properti yang dapat digunakan kembali. Method dapat mempunyai nilai balik atau tidak. Method yang tidak mempunyai nilai balik dipanggil dalam pernyataan yang akan dikerjakan, sedangkan method yang mempunyai nilai balik dipanggil dari suatu ekpresi. Kata kunci untuk mengembalikan/mengeluarkan suatu nilai adalah return

Method dengan tipe data void, berarti tidak memiliki nilai balik, berarti tidak memerlukan kata kunci return di dalamnya. Method dengan tipe data bukan void, berarti memerlukan suatu nilai balik, yaitu harus memerlukan return di dalamnya

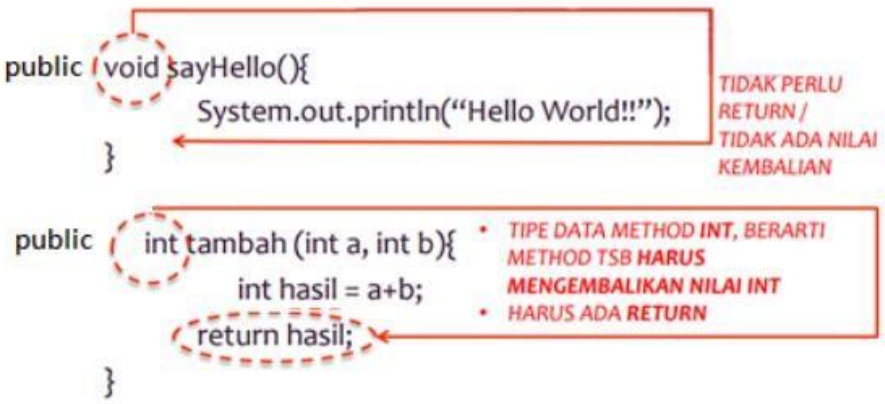
Deklarasi method dapat dilakukan dengan sintaks sebagai berikut:

```
public class Mahasiswa {


  <modifier> <tipe_data> <nama_metode> ([daftar_argumen]) {
      //statement
  }
}
```

]) {

Contoh method dengan tipe void dan method yang mengembalikan nilai (*return*)



Aturan penulisan method adalah sebagai berikut:

1. Berupa kata kerja,
2. Diawali dengan **HURUF KECIL**,
3. Jika terdiri dari 2 atau lebih kata, kata pertama diawali HURUF KECIL sedangkan kata selanjutnya diawali HURUF BESAR. Dan antar kata disambung tidak (dipisah).

Contoh deklarasi method:

```
public void tampil(){

   System.out.println ("Hallo PBO!!");
   }

   public int tambah(int a, int b){
   return a+b;

}
```

2.4 Object

Setelah Class dibuat, langkah selanjutnya adalah membuat Object. Proses pembuatan Object dari suatu Class disebut instansiasi. Format dasar instansiasi adalah sebagai berikut:

```
NamaClass namaObject = new NamaClass();
```

Proses membuat objek dari suatu class adalah INSTANSIASI, dan ditandai kata kunci new. Aturan penulisan objek adalah sama seperti penulisan atribut.

Contoh :

```
Random r = new Random();
Pegawai p2 = new Pegawai();
Mahasiswa mhs1= new Mahasiswa();
```

2.5 Try – catch

Untuk menangani ***error*** di Java, digunakan sebuah statement yang bernama *try - catch*. Statement tersebut digunakan untuk mengurung eksekusi yang menampilkan *error* dan dapat membuat program tetap berjalan tanpa dihentikan secara langsung. *Error* yang ditangani oleh *try – catch* biasa disebut dengan ***exception***.

Ada beberapa hal yang perlu diingat ketika akan menggunakan try - catch di Java:

1. Kita dapat membuat multiple try-catch,
2. Kita dapat menambahkan statement finally untuk menangani berbagai hal ketika error terjadi atau tidak,
3. Kita dapat membuat exception sendiri disamping menggunakan bawaan Java.

Untuk melihat hasil dari implementasi dari try-catch, maka kita perlu melakukan komparasi sintaks tanpa try-catch dengan sintaks yang menggunakan try-catch. Berikut percobaannya :

Tanpa menggunakan try-catch :

```
public class SourceErrorExp {
	public static void main(String[]args){
		System.out.println("awal program");

		int x = 10;
		x = x / 0;

		System.out.println(x);
		System.out.println("akhir program");
	}
}
```

Hasil :

```
$ javac SourceErrorExp.java
$ java SourceErrorExp
awal program
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at SourceErrorExp.main(SourceErrorExp.java:7)
```

Berbeda bila kita kurung operasi pembagian nol diatas dengan try - catch, maka hasil eksekusi program akan sedikit berbeda:

```
public class TanpaTC {
	public static void main(String[]args){
		System.out.println("awal program");

		int x = 10;
		try{
			x = x / 0;
		}catch(Exception e){
			e.printStackTrace();
			System.out.println("error karena pembagian nol");
		}
```

```
            System.out.println(x);
            System.out.println("akhir program");
        }
}
```

Hasil :

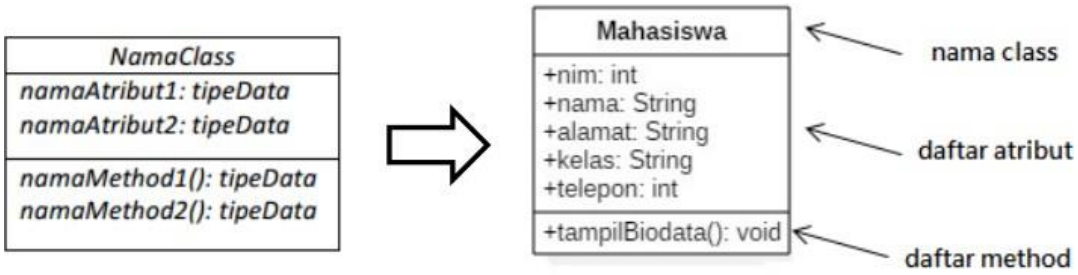
```
$ javac SourceErrorExp2
$ java SourceErrorExp2
awal program
java.lang.ArithmeticException: / by zero
    at SourceErrorExp2.main(SourceErrorExp2.java:10)
error karena pembagian nol
10
akhir program
```

3. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah tujuan umum, perkembangan, bahasa pemodelan di bidang rekayasa perangkat lunak , yang dimaksudkan untuk menyediakan cara standar untuk memvisualisasikan desain sistem. UML menyediakan sembilan jenis diagram yaitu Diagram kelas (Class Diagram), Diagram paket (Package Diagram), Diagram use-case (Usecase Diagram), Diagram interaksi dan sequence (Sequence Diagram), Diagram komunikasi (Communication Diagram), Diagram statechart (Statechart Diagram), Diagram aktivitas (Activity Diagram), Diagram komponen (Component Diagram), dan Diagram deployment (deployment diagram). Pada materi ini yang akan dipelajari adalah diagram class (class diagram).

Class diagram adalah sebuah class yang menggambarkan struktur dan penjelasan class, paket, dan objek serta hubungan satu sama lain seperti pewarisan, asosiasi, dan lain-lain. Class diagram juga menjelaskan hubungan antar class dalam sebuah sistem yang sedang dibuat dan bagaimana caranya agar mereka saling berkolaborasi untuk mencapai sebuah tujuan. Class juga memiliki 3 area pokok (utama) yaitu : nama,atribut,dan operasi. Nama berfungsi untuk member identitas pada sebuah kelas, atribut fungsinya adalah untuk member karakteristik pada data yang dimiliki suatu objek di dalam kelas, sedangkan

operasi fungsinya adalah memberikan sebuah fungsi ke sebuah objek. Berikut ini merupakan contoh dari class diagram:



4. Percobaan

4.1 Percobaan 1: Membuat Class Diagram

Studi Kasus 1:

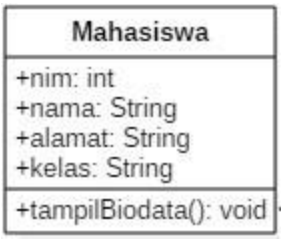
Dalam suatu perusahaan salah satu data yang diolah adalah data karyawan. Setiap karyawan memiliki id, nama, jenis kelamin, jabatan, jabatan, dan gaji. Setiap mahasiswa juga bisa menampilkan data diri pribadi dan melihat gajinya.

1. Gambarkan desain class diagram dari studi kasus 1!,
2. Sebutkan Class apa saja yang bisa dibuat dari studi kasus 1!,
3. Sebutkan atribut beserta tipe datanya yang dapat diidentifikasi dari masing-masing class dari studi kasus 1!
4. Sebutkan method-method yang sudah anda buat dari masing-masing class pada studi kasus 1!

4.2 Percobaan 2: Membuat dan mengakses anggota suatu class

Studi Kasus 2:

Perhatikan class diagram dibawah ini. Buatlah program berdasarkan class diagram tersebut!



Langkah kerja:

1. Bukalah text editor atau IDE, misalnya Notepad ++ / netbeans.
2. Ketikkan kode program berikut ini:

```
1  public class Mahasiswa {
2      public int nim;
3      public String nama;
4      public String alamat;
5      public String kelas;
6
7      public void tampilBiodata(){
8      System.out.println ("Nim    : "+nim);
9      System.out.println ("Nama   : "+nama);
10     System.out.println ("Alamat : "+alamat);
11     System.out.println ("Kelas  : "+kelas);
12     }
13 }
```

3. Simpan dengan nama file Mahasiswa.java.
4. Untuk dapat mengakses anggota-anggota dari suatu obyek, maka harus dibuat instance dari class tersebut terlebih dahulu. Berikut ini adalah cara pengaksesan anggota-anggota dari class Mahasiswa dengan membuka file baru kemudian ketikkan kode program berikut:

```
1  public class TestMahasiswa {
2      public static void main (String  args[]){
3      Mahasiswa mhs1=new Mahasiswa();
4      mhs1.nim=101;
5      mhs1.nama="Lestari";
6      mhs1.alamat="Jl. Vinolia No 1A";
7      mhs1.kelas="1A";
8      mhs1.tampilBiodata();
9      }
10 }
```

5. Simpan file dengan TestMahasiswa.java
6. Jalankan class TestMahasiswa
7. Jelaskan pada bagian mana proses pendeklarasian atribut pada program diatas!
8. Jelaskan pada bagian mana proses pendeklarasian method pada program diatas!
9. Berapa banyak objek yang di instansiasi pada program diatas!
10. Apakah yang sebenarnya dilakukan pada sintaks program “mhs1.nim=101” ?
11. Apakah yang sebenarnya dilakukan pada sintaks program “mhs1.tampilBiodata()” ?
12. Instansiasi 2 objek lagi pada program diatas!

4.3 Percobaan 3: Menulis method yang memiliki argument/parameter dan memiliki return

Langkah kerja:

1. Bukalah text editor atau IDE, misalnya Notepad ++ / netbeans.
2. Ketikkan kode program berikut ini:

```java
public class Barang {
    public String namaBrg;
    public String jenisBrg;
    public int stok;

    public void tampilBarang(){
    System.out.println ("Nama Barang     : "+namaBrg);
    System.out.println ("Jenis Barang    : "+jenisBrg);
    System.out.println ("Stok        : "+stok);
    }

    //method dengan argumen dan nilai balik (return)
    public int tambahStok(int brgMasuk){
    int stokBaru=brgMasuk+stok;
    return stokBaru;
    }
}
```

3. Simpan dengan nama file Barang.java
4. Untuk dapat mengakses anggota-anggota dari suatu obyek, maka harus dibuat instance dari class tersebut terlebih dahulu. Berikut ini adalah cara pengaksesan anggota-anggota dari class Barang dengan membuka file baru kemudian ketikkan kode program berikut:

```java
public class TestBarang{
    public static void main (String args[]){
        Barang brg1=new Barang();
        brg1.namaBrg="Pensil";
        brg1.jenisBrg="ATK";
        brg1.stok=10;
        brg1.tampilBarang();
        //  menampilkan dan mengisi argumen untuk menambahkan stok barang
        System.out.println ("Stok Baru adalah " +brg1.tambahStok(20));
        }
}
```

5. Simpan dengan nama file TestBarang.java
6. Jalankan program tersebut!
7. Apakah fungsi argumen dalam suatu method?
8. Ambil kesimpulan tentang kegunaan dari kata kunci return , dan kapan suatu method harus memiliki return!

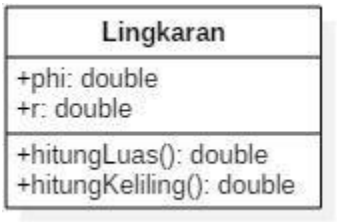
4.4 Tugas

1. Suatu toko persewaan video game salah satu yang diolah adalah peminjaman, dimana data yang dicatat ketika ada orang yang melakukan peminjaman adalah id, nama member, nama game, dan harga yang harus dibayar. Setiap peminjaman bisa menampilkan data hasil peminjaman dan harga yang harus dibayar. Buatlah class diagram pada studi kasus diatas!

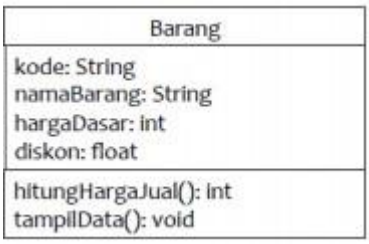
   Penjelasan:

   - Harga yang harus dibayar diperoleh dari lama sewa x harga.
   - Diasumsikan 1x transaksi peminjaman game yang dipinjam hanya 1 game saja.

2. Buatlah program dari class diagram yang sudah anda buat di no 1!
3. Buatlah program sesuai dengan class diagram berikut ini:



4. Buatlah program sesuai dengan class diagram berikut ini:



Deskripsi / Penjelasan :

- Nilai atribut hargaDasar dalam Rupiah dan atribut diskon dalam %
- Method hitungHargaJual() digunakan untuk menghitung harga jual dengan perhitungan berikut ini:

  harga jual = harga dasar – (diskon x harga dasar)

- Method tampilData() digunakan untuk menampilkan nilai dari kode, namaBarang, hargaDasar, diskon dan harga jual.