CLASS & OBJECT

Class adalah blueprint/cetakan dari object yang dibentuk. Sedangkan object adalah hasil instansiasi/bentukan dari sebuah class. Dengan kata lain satu class dapat membentuk object sampai tak terhingga.

Syntax class :

```
class nama_class
{

        //data-field
        .....
        //method
        .....

}
```

contoh :

```
class Sepeda{

        //data-field :
        int speed;
        //method :
        void ubahGear()
        {

        }
        public static void main(String[] args)
        {
          System.out.println("ini adalah main method");
        }
}
```

Syntax pembuatan object :

```
nama_class nama_object = new nama_class();
```

contoh :

Sepeda.bike = new Sepeda(); //perintah untuk membuat object yang mengacu ke class Sepeda.

Variabel

Syntax pengaksesan field :

```
nama_object.nama_field;
```

contoh :

bike.speed; //perintah untuk mengakses variabel speed melalui object speed.

Method

Syntax deklarasi :

```
<modifier> <return_type> <nama_metode> ([parameter])
{
        [<statement>]
}
```

Return type berhubungan dengan nilai yang akan dikembalikan oleh method (return value) ketika proses invoking/pemanggilan.
Return type bisa berupa tipe data primitif maupun tipe reference.
Method yang tidak memiliki return value memiliki return type **void**.
Method yang memiliki return value memiliki statement **return** didalam blok-nya.
Parameter adalah variabel yang akan menangkap argument yang dikirim ketika proses invoking method.
Aturan penulisan parameter sama seperti deklarasi variabel.
Ditulis diantara tanda (dan) pada method signature.
Parameter bersifat optional. Method bisa memiliki parameter kosong.
Untuk method yang memiliki lebih dari satu parameter penulisannya dipisahkan dengan tanda koma (,)

Syntax pengaksesan method :

```
nama_object.nama_method();
```

contoh :
bike.ubahGear();

PASS by Value
Mengirimkan value/nilai ke parameter method

```
String nama = "Shanti";

Mahasiswa.getNama(nama);
```

PASS by REFERENCE
Mengirimkan alamat memori/referensi ke parameter method

```
Student maba = new Mahasiswa();

Mahasiswa.getNama(maba);
```

Method Overload
Sebuah class dapat memiliki beberapa method dengan nama yang sama.
Pembeda antara method-method tersebut adalah parameter.

```
public void print (String temp){
    System.out.println("Name:"+name);
    System.out.println("Address:"+address);
    System.out.println("Age"+age);
}

public void print(double eGrade, double mGrade, double sGrade){
    System.out.println("Name"+name);
    System.out.println("Math Grade"+mGrade);
    System.out.println("English Grade"+eGrade);
    System.out.println("Science Grade"+sGrade);
}
```

Method Accessor
digunakan untuk **membaca** *value (variabel)* dari *class.*
ditulis dengan menggunakan sintaks berikut:

get<NameOfInstanceVariable>
dapat me-*return value*.

```
public class StudentRecord {
    private String name;

    public String getName(){
        return name;

    }
}
```

Method Mutator
digunakan untuk **menulis atau mengubah** *value* (variabel) dari *class.*
Ditulis dengan menggunakan sintaks berikut:
set<NameOfInstanceVariable>

```
public class StudentRecord {
    private String name;

    public void setName( String temp ){
        name = temp;
    }
}
```

MODIFIER
Digunakan untuk mengatur hak akses class dan member class.
Terdiri dari access modifier : public, private, protected, dan no modifier.
dan modifier jenis lain: static, final, abstract, dll.

CONSTRUCTOR
Digunakan untuk memberikan inisialisasi pada object.

Syntax constructor sama seperti method namun tidak memiliki return value.
Nama constructor **harus sama** dengan nama class.
Contoh :

```
class Buku()
{
        Buku()
        {
                judul = "apa aja";
                pengarang = "siapa saja";
        }
}
```

Pengaksesan constructor :

namaClass namaVariabel = new namaClass(nilai constructor);

Contoh pengaksesan constructor :
Pengaksesan constructor tanpa parameter :
Mahasiswa maba = new Mahasiswa();
Pengaksesan constructor dengan 1 parameter :
Mahasiswa maba = new Mahasiswa("Santi");
Pengaksesan constructor dengan 2 parameter :
Mahasiswa maba = new Mahasiswa("1234","Santi");

OverLoad Constructor
Sebuah class dapat memiliki lebih dari satu constructor.
Pembeda constructor overloading tersebut adalah parameter.

```
public StudentRecord(){
   //beberapa kode inisialisasi di sini
}
public StudentRecord(String temp){
   this.name = temp;
}
public StudentRecord(String name, String address){
   this.name = name;
   this.address = address;
}
public StudentRecord(double mGrade,double eGrade,double
sGrade){
   mathGrade = mGrade;
   englishGrade = eGrade;
   scienceGrade = sGrade;
}
```

I. PROSEDUR PELAKSANAAN

Prosedur pelaksanaan praktikum adalah sebagai berikut :

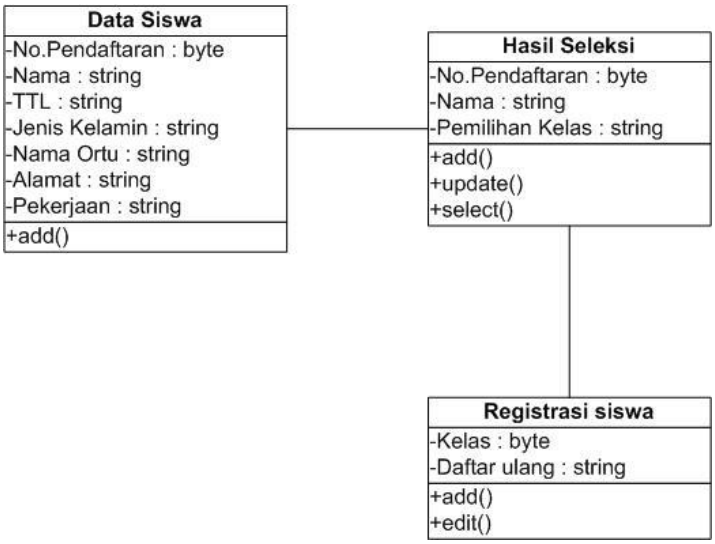
1. Mahasiswa mencoba latihan yang ada pada modul praktikum
2. Mahasiswa menganalisa hasil dari program pada latihan yang telah dijalankan
3. Mahasiswa mengerjakan tugas yang diberikan

4. Mahasiswa membuat laporan dari tugas yang telah dikerjakan (dilengkapi dengan Hasil praktikum dan kesimpulan).
5. Upload laporan melalui e-labit.umm.ac.id
6. Mahasiswa mendemonstrasikan program yang telah dikerjakan
7. Asisten/dosen menilai pekerjaan mahasiswa

II. LATIHAN

III. TUGAS PRAKTIKUM

Gambar 1. Class diagram



Petunjuk : perhatikan gambar 1 diatas untuk mengerjakan tugas berikut secara berkelompok!

1. Buatlah Sebuah class yang merepresentasikan diagram "Data Siswa" pada gambar 1 diatas. Lengkapi member class dengan memahami penjelasan berikut :
   - Deklarasikan seluruh atribut/variable yang ada pada diagram tersebut. Sesuaikan juga tipe data untuk masing-masing variable sesuai dengan keterangan yang ada pada diagram.
   - Method add() : tambahkan 1 parameter pada method add() dengan nama nomor bertipe byte. Parameter ini digunakan untuk inisialisasi nilai ke variable "no pendaftaran". Selanjutnya tambahkan statement pada body method untuk menampilkan pesan ke layar sebagai berikut : "No pendaftaran = [XXXX], nama = [XXXX] telah terdaftar". Dimana XXXX adalah nilai hasil pengaksesan variable.
   - Tuliskan konstruktor pada class, dimana konstruktor akan melakukan inisialisasi seluruh variable kecuali "no pendaftaran".

2. Buatlah Sebuah class yang merepresentasikan diagram "Hasil Seleksi" pada gambar 1 diatas. Lengkapi member class dengan memahami penjelasan berikut :
   - Deklarasikan ketiga variable sesuai dengan tipe data yang diberikan.
   - Method add() : [sesuaikan dengan tugas nomor 1] namun tambahkan 1 parameter lagi bernama pilihKelas bertipe String untuk inisialisasi nilai ke variable "pemilihan kelas". Jadi method ini memiliki 2 parameter. Selanjutnya tambahkan statement pada body method untuk menampilkan pesan ke layar sebagai berikut : "No pendaftaran = [XXXX], nama = [XXXX], pilihan kelas = [XXXX]". Dimana XXXX adalah nilai

hasil pengaksesan variable.

- Method update() : jadikan return type-nya adalah string. Selanjutnya tambahkan 1 parameter bernama tanggal bertipe string. Pada body method tuliskan statement untuk menampilkan pesan ke layar sebagai berikut : “Data telah ter-update”. Jadikan variable tanggal sebagai return value.
- Method select() : tambahkan 1 parameter bernama status bertipe Boolean. Pada body method tuliskan percabangan untuk mengecek nilai variable status. Jika status=true maka tampilkan pesan “Nama = [XXXX] lulus seleksi”. Jika status=0 tampilkan pesan “Nama = [XXXX] tidak lulus seleksi”. Dimana XXXX adalah nilai hasil pengaksesan variable nama.

3. Buatlah Sebuah class yang merepresentasikan diagram “Registrasi siswa” pada gambar 1 diatas. Lengkapi member class dengan memahami penjelasan berikut :
   - Deklarasikan kedua variable sesuai dengan tipe data yang diberikan.
   - Tambahkan konstruktor pada class, dimana konstruktor digunakan untuk memberikan nilai default pada variable Kelas dengan nilai 1 (dimana nilai variable kelas yang dapat diberikan nantinya adalah dari nilai 1 sampai 6) sedangkan nilai default dari variable “daftar ulang” adalah 0000.
   - Method add() : Tuliskan statement pada body method untuk menampilkan pesan sebagai berikut “No pendaftaran = [XXXX] telah melakukan daftar ulang untuk kelas [XXXX]”. Dimana XXXX adalah nilai hasil pengaksesan variable.
   - Method edit() : tambahkan 2 parameter bernama kelas dan “no pendaftaran”, dimana masing-masing parameter memiliki tipe byte dan string. Parameter kelas digunakan untuk inisialisasi variable kelas, sedangkan variable “no pendaftaran” digunakan untuk inisialisasi variable “daftar ulang”. Berikan return value berupa true. Sesuaikan sendiri return-type yang ditambahkan pada deklarasi method.