JOBSHEET II
CLASS & OBJECT

2.1 Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Mengenal class dan object sebagai konsep dasar pada pemrograman berorientasi objek
2. Mendeklarasikan class beserta atribut dan methodnya
3. Mendeklarasikan constructor
4. Melakukan instansiasi (pembuatan objek baru)
5. Mengakses atribut dan memanggil method dari suatu objek

2.2 Deklarasi Class, Atribut dan Method

Waktu: 40 Menit

Perhatikan Diagram Class berikut ini:

Sepeda
kecepatan: float gear: int
tambahKecepatan(increment:int): void kurangiKecepatan(decrement:int): void cetakInfo(): void

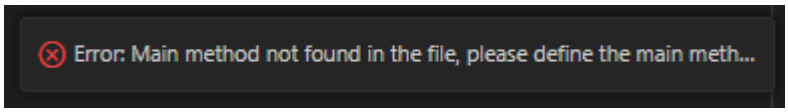
Berdasarkan class diagram di atas, akan dibuat program class dalam Java.

2.2.1 Langkah-langkah Percobaan

1. Buatlah folder baru dengan nama **Praktikum02** kemudian buatlah file baru dengan nama Sepeda.java
2. Lengkapi class **Sepeda** dengan atribut dan method yang telah digambarkan di dalam class diagram di atas

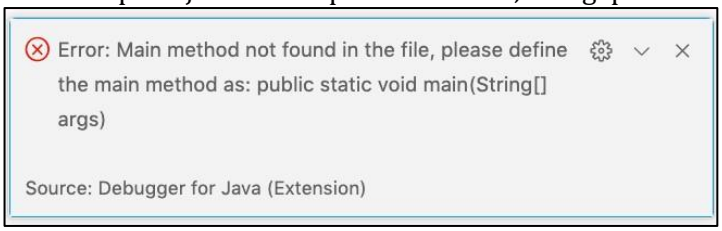
```java
package Praktikum02;

public class Sepeda {
    float kecepatan;
    int gear;

    public void tambahKecepatan(float increment) {
        kecepatan += increment;
    }

    public void kurangiKecepatan(float decrement) {
        kecepatan -= decrement;
    }

    public void cetakInfo() {
        System.out.println("Kecepatan: " + kecepatan);
        System.out.println("Gear: " + gear);
        System.out.println("====================");
    }
}
```

3. Compile dan run Sepeda.java.



2.2.2 Pertanyaan

1. Ketika Sepeda.java di-compile dan di-run, mengapa error berikut muncul?



   JAWAB: karena tidak ada method main dari kode program tersebut

2. Perhatikan class **Sepeda** yang ada di Praktikum di atas, ada berapa atribut yang dimiliki oleh class tersebut? Sebutkan! Dan pada baris berapa saja deklarasi atribut dilakukan?

   JAWAB: class sepeda memiliki dua atribut, yaitu kecepatan dan gear. Deklarasi atribut dilakukan pada baris 4 dan baris 5

3. Ada berapa method yang dimiliki oleh class tersebut? Sebutkan!

   Jawab: class tersebut memiliki 3 method, tambahKecepatan(), kurangiKecepatan(), dan cetakInfo()

4. Sebutkan parameter dari method tambahKecepatan()

   Jawab: parameter dari method tambahKecepatan() adalah increment

5. Mengapa method **tambahKec**? **epatan()** memerlukan parameter **increment**

   Jawab: karena parameter ini menentukan seberapa besar peningkatan kecepatan yang diinginkan oleh objek sepeda

6. Mengapa method **tambahKecepatan()** tidak memerlukan parameter **kecepatanAwal**?

   Jawab: karena method ini hanya bertujuan untuk menambah kecepatan sepeda sesuai dengan parameter increment yang diberikan

7. Mengapa method **cetakInfo()** memiliki return type void?

   Jawab: karena method ini tidak mengembalikan nilai apapun kepada pemanggilnya

8. Modifikasi method **kurangiKecepatan()** sehingga kecepatan minimum adalah 0

   Jawab:

```
kecepatan -= decrement;
if (kecepatan < 0) {
    kecepatan = 0;
}
```

9. Modifikasi method **tambahKecepatan()** sehingga kecepatan maksimum adalah 20

   Jawab:

```
kecepatan += increment;
if (kecepatan > 20) {
    kecepatan = 20;
}
```

2.3 Instansiasi Objek dan Mengakses Atribut & Method

Waktu: 40 Menit

Class Sepeda telah dibuat sebagai template/cetakan untuk membuat objek-objek bertipe sepeda. Untuk membuat objek baru, perlu dilakukan instansiasi objek.

1. Buatlah class baru dengan nama **SepedaMain** beserta method **main()**.
2. Di dalam method **main()**, lakukan instansiasi objek bernama **sepeda1** dan **sepeda2** kemudian cobalah untuk memodifikasi atribut dan memanggil method.

```java
package Praktikum02;

public class SepedaMain {
    Run | Debug
    public static void main(String[] args) {
        Sepeda sepeda1 = new Sepeda();
        sepeda1.kecepatan = 5;
        sepeda1.gear = 1;
        sepeda1.tambahKecepatan(3);
        sepeda1.cetakInfo();

        Sepeda sepeda2 = new Sepeda();
        sepeda2.cetakInfo();
    }
}
```

3. Run class **SepedaMain** tersebut dan amati hasilnya.

```
Kecepatan: 8.0
Gear: 1
==================
Kecepatan: 0.0
Gear: 0
==================
PS C:\Users\LENOVO\Do
```

2.3.1 Pertanyaan

1. Pada class **SepedaMain**, pada baris berapa dilakukan instansiasi? Apa nama objek yang dihasilkan?

   Jawab: instansiasi dilakukan pada baris 5 dan baris 11. Nama objek yang dihasilkan adalah sepeda1 dan sepeda2.

2. Sebutkan perbedaan class dan object

   Jawab: class adalah cetak biru atau rancangan yang mendefinisikan atribut dan method yang dimiliki oleh suatu objek, sedangkan object lebih tertuju pada suatu nama

3. Bagaimana cara mengakses atribut dan memanggil method dari suatu objek?

   Jawab: untuk mengakses atribut dan memanggil method dari suatu objek, Anda bisa menggunakan operator titik (.) setelah nama objek

4. Bagaimana hasil **cetakInfo()** untuk objek **sepeda2**? Apa kesimpulannya?

   Jawab:

   ```
   Kecepatan: 0.0
   Gear: 0
   ==================
   ```

   method cetakInfo() menampilkan informasi tentang kecepatan dan gear sepeda yang disimpan dalam atribut-atribut objek sepeda2.

5. Pada class **Sepeda** tidak terdapat constructor **Sepeda()** secara eksplisit, mengapa objek sepeda tetap dapat diinstansiasi?

Jawab: karena java secara otomatis menyediakan constructor default yang tidak menerima argumen apapun

2.4 Constructor

Waktu: 40 Menit

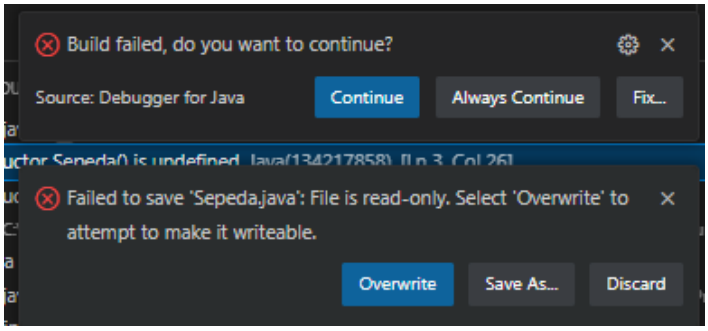
Di dalam percobaan ini, kita akan mempraktekkan bagaimana membuat berbagai macam konstruktor berdasarkan parameternya.

2.4.1 Langkah-langkah Percobaan

1. Pada class **Sepeda**, deklarasikanlah constructor berparameter sebagai berikut

```java
package Praktikum02;

public class Sepeda {
    float kecepatan;
    int gear;

    public Sepeda(float newKecepatan, int newGear) {
        kecepatan = newKecepatan;
        gear = newGear;
    }

    public void tambahKecepatan(float increment) {
        kecepatan += increment;
    }

    public void kurangiKecepatan(float decrement) {
        kecepatan -= decrement;
    }

    public void cetakInfo() {
        System.out.println("Kecepatan: " + kecepatan);
        System.out.println("Gear: " + gear);
        System.out.println("====================");
    }
}
```

2. Run kembali class **SepedaMain** dan amati hasilnya.



Terjadi nya eror

2.4.2 Pertanyaan

1. Apakah constructor juga merupakan method? Jika iya, apa perbedaan constructor dengan method lainnya?

   Jawab: ya, constructor juga merupakan method, tetapi ada beberapa perbedaan antara constructor dan method lainnya. constructor digunakan untuk menginisialisasi keadaan objek, sedangkan method digunakan untuk menunjukkan perilaku objek

2. Apakah SepedaMain dapat di-run? Mengapa?

   Jawab: tidak, karena di dalam class sepeda, ada tambahan konstruktor yang menerima dua parameter

3. Modifikasi SepedaMain sebagai berikut

```
public class SepedaMain {
    Run | Debug
    public static void main(String[] args) {
        Sepeda sepeda1 = new Sepeda(5, 1);
        sepeda1.tambahKecepatan(3);
        sepeda1.cetakInfo();
    }
}
```

4. Perhatikan bahwa **SepedaMain** sudah dapat di run

```
Kecepatan: 8.0
Gear: 1
==================
PS C:\Users\LENOVO
```

5. Suatu class dapat memiliki lebih dari 1 constructor, tambahkan constructor tanpa parameter pada class Sepeda

```
public Sepeda(){

}

public Sepeda(float newKecepatan, int newGear) {
    kecepatan = newKecepatan;
    gear = newGear;
}
```

6. Modifikasi class SepedaMain sebagai berikut

```java
public class SepedaMain {
    Run | Debug
    public static void main(String[] args) {
        Sepeda sepeda1 = new Sepeda(5, 1);
        sepeda1.tambahKecepatan(3);
        sepeda1.cetakInfo();

        Sepeda sepeda2 = new Sepeda();
        sepeda2.kecepatan = 7;
        sepeda2.gear = 1;
        sepeda2.cetakInfo();
    }
}
```

7. Run SepedaMain dan amati hasilnya. Object sepeda1 dibuat dengan constructor yang mana? Bagaimana dengan object sepeda2?

```
Kecepatan: 8.0
Gear: 1
==================
Kecepatan: 7.0
Gear: 1
==================
```

- objek sepeda1 dibuat dengan constructor parameterized yang menerima dua parameter, yaitu newKecepatan dan newGear.
- objek sepeda2 dibuat dengan constructor default yang tidak menerima parameter apapun

2.5 Tugas

Waktu: 180 menit

1. Program Game Snake sederhana

Snake
x: int y: int
moveLeft(): void moveRight(): void moveUp(): void moveDown(): void printPosition(): void

- Buatlah class Snake sesuai class diagram di atas
- Atribut **x** digunakan untuk menyimpan posisi koordinat x (mendatar) dari snake, sedangkan atribut **y** untuk posisi koordinat y (vertikal)
- Method **moveLeft()** digunakan untuk mengubah posisi snake ke kiri (koordinat x akan berkurang 1), sedangkan **moveRight()** untuk bergerak ke kanan (koordinat x akan bertambah 1).

- Method moveUp() digunakan untuk mengubah posisi snake ke atas (koordinat y akan bertambah 1), sedangkan moveDown() untuk bergerak ke bawah (koordinat y akan berkurang 1).
- Method **printPosition()** digunakan untuk mencetak koordinat x dan y untuk objek snake
- Buat class SnakeMain lalu lakukan instansiasi 2 objek dari class Snake. Cobalah melakukan perubahan posisi untuk kedua objek tersebut.

Tugas > Snake.java > Snake > moveLeft()

```java
package Tugas;

public class Snake {
    int x;
    int y;

    public Snake() {
        x = 0;
        y = 0;
    }

    public void moveLeft() {
        x--;
    }

    public void moveRight() {
        x++;
    }

    public void moveUp() {
        y++;
    }

    public void moveDown() {
        y--;
    }

    public void printPosition() {
        System.out.println("x: " + x + ", y: " + y);
    }
}
```

Tugas > SnakeMain.java > SnakeMain > main(String[])

```java
package Tugas;

public class SnakeMain {
    Run | Debug
    public static void main(String[] args) {
        Snake snake1 = new Snake();
        snake1.printPosition();
        snake1.moveRight();
        snake1.moveUp();
        snake1.printPosition();

        Snake snake2 = new Snake();
        snake2.printPosition();
        snake2.moveLeft();
        snake2.moveDown();
        snake2.printPosition();
    }
}
```

```
x: 0, y: 0
x: 1, y: 1
x: 0, y: 0
x: -1, y: -1
PS C:\Users\LEN
```

2. Program Game Dragon sederhana

Dragon
x: int y: int direction: int
changeDirection(newDirection: int): void move(steps: int): void printStatus(): void

- Buatlah class Dragon sesuai class diagram di atas
- Atribut **x** digunakan untuk menyimpan posisi koordinat x (mendatar) dari snake, sedangkan atribut **y** untuk posisi koordinat y (vertikal)
- Atribut direction digunakan untuk menyimpan arah dragon:
  1: atas
  2: kanan
  3: bawah
  4: kiri
- Method **changeDirection()** digunakan untuk mengubah arah dragon berdasarkan parameter newDirection. Implementasikan method changeDirection sedemikian rupa sehingga atribut direction hanya dapat bernilai 1, 2, 3, atau 4.
- Method **move()** digunakan untuk mengubah posisi dragon dengan jumlah langkah sesuai parameter steps. Posisi akan berubah bergantung dengan direction saat ini. Misalnya, jika direction bernilai 1 maka dragon akan berpindah ke arah atas, dan seterusnya.
- Method **printStatus()** digunakan untuk mencetak koordinat dan arah objek dragon
- Buatlah class DragonMain kemudian lakukan instansiasi 2 buah objek dari class dragon. Cobalah melakukan perubahan posisi untuk kedua objek tersebut.
- Apa yang terjadi jika method move() dipanggil persis setelah objek diinstansiasi? Ke mana objek berpindah? Perbaiki kode program untuk mengatasi masalah tersebut

```java
package Tugas;

public class Dragon {
    int x, y;
    int direction;

    public static final int UP = 1;
    public static final int RIGHT = 2;
    public static final int DOWN = 3;
    public static final int LEFT = 4;

    public Dragon() {
        x = 0;
        y = 0;
        direction = 0;
    }

    public void changeDirection(int newDirection) {
        if (newDirection >= UP && newDirection <= LEFT) {
            direction = newDirection;
        } else {
            System.out.println(x:"Arah tidak valid");
        }
    }
    public void move(int steps) {
        switch (direction) {
            case 1:
                y += steps;
                break;
            case 2:
                x += steps;
                break;
            case 3:
                y -= steps;
                break;
            case 4:
                x -= steps;
                break;
            default:
```

```java
package Tugas;

public class DragonMain {
    Run | Debug
    public static void main(String[] args) {
        Dragon dragon1 = new Dragon();
        dragon1.printStatus();
        dragon1.changeDirection(Dragon.RIGHT);
        dragon1.move(steps:5);
        dragon1.printStatus();

        Dragon dragon2 = new Dragon();
        dragon2.printStatus();
        dragon2.changeDirection(Dragon.UP);
        dragon2.move(steps:3);
        dragon2.printStatus();
    }
}
```

```
x: 0, y: 0
Arah: tidak valid
x: 5, y: 0
Arah: kanan
x: 0, y: 0
Arah: tidak valid
x: 0, y: 3
Arah: atas
```