JOBSHEET II
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

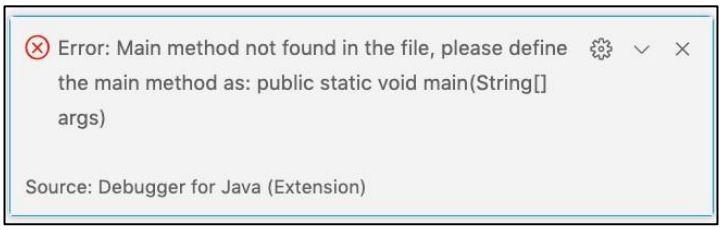
1. Buatlah folder baru dengan nama **Praktikum02** kemudian buatlah file baru dengan nama Sepeda.java
2. Lengkapi class **Sepeda** dengan atribut dan method yang telah digambarkan di dalam class diagram di atas

```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? iya



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?

- Ada 2 yaitu kecepatan dan gear baris 4-5

3. Ada berapa method yang dimiliki oleh class tersebut? Sebutkan!
- Ada 3 yaitu tambahkecepatan, kurangikecepatan, cetakinfo

4. Sebutkan parameter dari method tambahKecepatan()

- Float increment

5. Mengapa method **tambahKecepatan()** memerlukan parameter **increment**?

- Karena digunakan untuk menentukan tambahan nilai kecepatan sepeda

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

- Karena pada implementasi method tambah kecepatan,penambahannya dilakukan dengan menggunakan nilai tetap

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

- Karena digunakan untuk mencetak informasi tentang objek dan tidak mengembalikan nilai

apapun

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

```
public void kurangiKecepatan(float decrement){
    kecepatan -= decrement;
    if(kecepatan < 0 ){
        kecepatan = 0;
    }

}
```

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

```
public void tambahKecepatan(float increment){
    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.

```
1   package Praktikum02;
2
3   public class SepedaMain {
        Run | Debug
4       public static void main(String[] args) {
5           Sepeda sepeda1 = new Sepeda();
6           sepeda1.kecepatan = 5;
7           sepeda1.gear = 1;
8           sepeda1.tambahKecepatan(3);
9           sepeda1.cetakInfo();
10
11          Sepeda sepeda2 = new Sepeda();
12          sepeda2.cetakInfo();
13      }
14  }
```

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

2.3.1 Pertanyaan

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

   - 5 dan 11,sepeda1 dan sepeda2

2. Sebutkan perbedaan class dan object

   - Class berupa rancangan atau template
   - Object merupakan bentuk nyata ang dibentuk dari class

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

   - Menggunakan format objek.atribut | menggunakan format objek.method();

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

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

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

- Kesimpulannya yaitu Java menyediakan konstruktor default untuk kelas tersebut secara otomatis sehingga tetap bisa diinstansiasi

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.

2.4.2 Pertanyaan

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

- Iya,constructor dan method memiliki perbedaan dalam pemanggilan yakni constructor dipanggil secara otomatis saat objek dibuat sedangkan method harus dipanggil secara eksplisit menggunakan nama methodnya.

2. Apa yang sebenarnya dilakukan ketika constructor dipanggil?

- Constructor bertanggung jawab untuk menciptakan objek baru dan constructor digunakan untuk menginisialisasi nilai awal untuk variable objek

3. Apakah SepedaMain dapat di-run? Mengapa?

- Tidak bisa karena tidak ada constructor yang didefinisikan

4. Modifikasi 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();
    }
}
```

5. Perhatikan bahwa **SepedaMain** sudah dapat di run
6. 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;
}
```

7. Modifikasi class 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();

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

8. Run SepedaMain dan amati hasilnya. Object sepeda1 dibuat dengan constructor yang mana? Bagaimana dengan object sepeda2? Buatlah kesimpulan terkait constructor.

- Sepeda1 dibuat dengan constructor yang memiliki parameter sedangkan pada sepeda2 terjadi error karena nilai dari kecepatan sepeda dan gear tidak terdefinisi pada parameter sedangkan pada contructor di sepeda_23.java memiliki parameter yang harus di isi Kesimpulannya, Constructor digunakan untuk menginisialisasi nilai awal dari variabel-variabel instance.

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.

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

3. Implementasikan soal nomor 2 pada tugas ASD Teori ke dalam kode program