Nama : M. Hasan basri
Kelas : SIB-2C
NIM : 2241760139

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

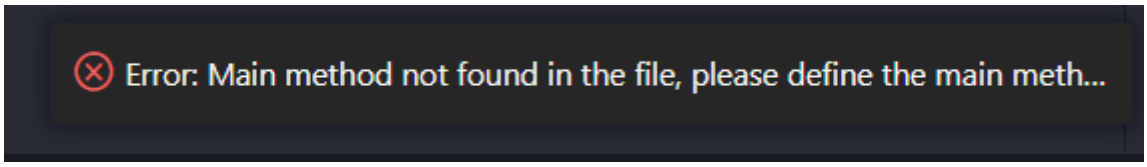
Berdasarkan class diagram di atas, akan dibuat program class dalam Java.

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(x:"=========");
    }
}
```

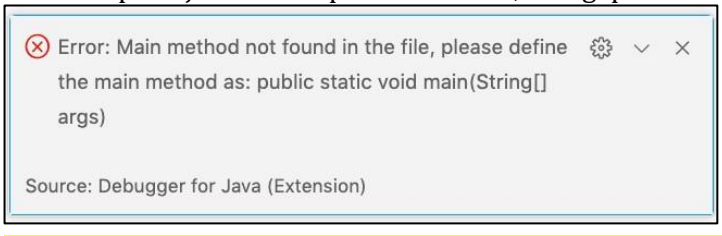
3. Compile dan run Sepeda.java.



Kode error tidak bisa berjalan karena tidak ada class main yang berfungsi memanggil method.

2.2.2 Pertanyaan

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



Jawab : Error muncul karena tidak ada class main yang digunakan untuk memanggil method. Dalam Java, setiap program membutuhkan method main sebagai entry point untuk eksekusi.

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: kecepatan dan gear. Deklarasi atribut dilakukan pada baris 4 dan 5.

3. Ada berapa method yang dimiliki oleh class tersebut? Sebutkan!
   Jawab : Class Sepeda memiliki tiga method: tambahKecepatan, kurangiKecepatan, dan cetakInfo.
4. Sebutkan parameter dari method tambahKecepatan()
   Jawab : Parameter dari method tambahKecepatan() adalah increment, yang bertipe data float.
5. Mengapa method **tambahKecepatan()** memerlukan parameter **increment**?
   Jawab : Method tambahKecepatan() memerlukan parameter increment karena kita ingin menambahkan jumlah tertentu pada kecepatan sepeda.
6. Mengapa method **tambahKecepatan()** tidak memerlukan parameter **kecepatanAwal**?
   Jawab : Method tambahKecepatan() tidak memerlukan parameter kecepatanAwal karena kita sudah memiliki atribut kecepatan di dalam kelas Sepeda yang dapat diakses langsung di dalam method tersebut.
7. Mengapa method **cetakInfo()** memiliki return type void?
   Jawab : Method cetakInfo() memiliki return type void karena hanya bertugas untuk mencetak informasi ke layar dan tidak mengembalikan nilai apapun.
8. Modifikasi method **kurangiKecepatan()** sehingga kecepatan minimum adalah 0
   jawab:

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

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

```
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.

```java
// SepedaMain.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(increment: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:\KULIAH\Semester 4\OOP Praktikum\Praktikum2>
```

Setelah ditambahkan class Main, program bisa dijalankan dengan normal tanpa error karena ada class main untuk melakukan eksekusi.

2.3.1 Pertanyaan

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

   Jawab : Pada class SepedaMain, instansiasi dilakukan pada baris 5 dan baris 11. Objek yang dihasilkan adalah sepeda1 dan sepeda2.

2. Sebutkan perbedaan class dan object

   Jawab : Class adalah blueprint atau cetakan untuk menciptakan objek. Itu mendefinisikan perilaku dan atribut objek.

   Objek adalah instance dari class. Ini adalah entitas yang nyata yang memiliki keadaan (atribut) dan perilaku (metode) sesuai dengan yang ditentukan dalam class.

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

   Jawab : Cara mengakses atribut dan memanggil method dari suatu objek adalah dengan menggunakan operator . (titik). Misalnya, jika Anda memiliki objek bernama obj, Anda dapat mengakses atribut dengan obj.namaAtribut dan memanggil method dengan obj.namaMethod().

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

   Jawab : Hasil dari cetakInfo() untuk objek sepeda2 akan mencetak kecepatan dan gear dengan nilai default, yaitu 0 untuk kecepatan dan 1 untuk gear. Kesimpulannya, sepeda2 tidak mengalami perubahan kecepatan atau gear setelah objek dibuat.

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

   Jawab : Objek Sepeda tetap dapat diinstansiasi karena Java secara otomatis memberikan constructor default jika tidak ada constructor yang didefinisikan secara eksplisit dalam class. Constructor default akan menginisialisasi nilai-nilai default untuk atribut-atribut class. Dalam hal ini, nilai default untuk kecepatan dan gear adalah 0 dan 1, sehingga objek Sepeda dapat diinstansiasi tanpa adanya constructor eksplisit.

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

```
Praktikum02 > Sepeda.java > Sepeda
1  package Praktikum02;
2
3  public class Sepeda {
4      float kecepatan;
5      int gear;
6
7      public Sepeda(float newKecepatan, int newGear){
8              kecepatan = newKecepatan;
9              gear = newGear;
10     }
11
12     public void tambahKecepatan(float increment) {
13         kecepatan += increment;
14     }
15
16     public void kurangiKecepatan(float decrement) {
17         kecepatan -= decrement;
18     }
19
20     public void cetakInfo() {
21         System.out.println("Kecepatan: " + kecepatan);
22         System.out.println("Gear: " + gear);
23         System.out.println(x:"=========");
24     }
25 }
26
```

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

```
The constructor Sepeda() is undefined
The constructor Sepeda() is undefined

at Praktikum02.SepedaMain.main(SepedaMain.java:5
```

2.4.2 Pertanyaan

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

   Jawab : Constructor bukanlah method seperti method lainnya dalam Java. Meskipun secara konsep, constructor dapat dianggap sebagai sebuah jenis method, namun ada perbedaan utama antara constructor dan method lainnya:

   Constructor tidak memiliki tipe kembalian (void atau tipe data lainnya), sementara method memiliki tipe kembalian yang dapat berupa void atau tipe data lainnya.

   Constructor memiliki nama yang sama dengan nama kelasnya, sementara method memiliki nama yang berbeda dari nama kelasnya.

   Constructor dipanggil saat objek dari kelasnya dibuat (diinstansiasi), sedangkan method harus dipanggil secara eksplisit untuk dieksekusi.

2. Apa yang sebenarnya dilakukan ketika constructor dipanggil?

   Jawab : Ketika constructor dipanggil, apa yang sebenarnya terjadi adalah alokasi memori untuk objek dan inisialisasi nilai-nilai awal untuk atribut-atribut objek. Constructor dijalankan sebagai bagian dari proses pembuatan objek baru (instansiasi).

3. Apakah SepedaMain dapat di-run? Mengapa?

```
The constructor Sepeda() is undefined
The constructor Sepeda() is undefined

at Praktikum02.SepedaMain.main(SepedaMain.java:5
```

   Jawab : Tidak, class SepedaMain tidak dapat di-run karena setelah membuat constructor ,kita harus melakukan instansiasi terlebih dahulu

   Di class sepedaMain sudah ada instansiasi tetapi masih belum tepat (masih default), karena harus ditambahkan nilai untuk parameter seperti dibawah :

```
Sepeda sepeda1 = new Sepeda(newKecepatan:5,newGear:1);
```

4. Modifikasi SepedaMain sebagai berikut

```
Praktikum02 > SepedaMain.java > SepedaMain > main(String[])
1  package Praktikum02;
2
3  public class SepedaMain {
       Run | Debug
4      public static void main(String[] args) {
5          Sepeda sepeda1 = new Sepeda(newKecepatan:5,newGear:1);
6          sepeda1.tambahKecepatan(increment:5);
7          sepeda1.cetakInfo();
8      }
9  }
```

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

```
Kecepatan: 10.0
Gear: 1
==================
PS C:\KULIAH\Semester 4\OOP Praktikum\Praktikum2>
```

Jawab : SepedaMain Sudah bisa dijalankan karena sudah dilakukan instansiasi terhadap constructor yang dibuat sebelumnya.

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

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

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

```
Kecepatan: 10.0
Gear: 1
==================
Kecepatan: 7.0
Gear: 1
==================
PS C:\KULIAH\Semester 4\OOP Praktikum\Praktikum2>
```

Jawab :

Objek sepeda1 dibuat menggunakan constructor Sepeda(float newKecepatan, int newGear) dan diatur kecepatannya menjadi 5 serta gearnya menjadi 1. Kemudian, kecepatannya ditambah 5 dengan method tambahKecepatan(5).

Objek sepeda2 dibuat menggunakan constructor default Sepeda() dan langsung diatur kecepatan menjadi 7 dan gear menjadi 1 secara langsung setelah pembuatan objek.

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.

KODE :

```
Praktikum02 > Snake.java > Snake > moveRight()
1  package Praktikum02;
2  public class Snake {
3      int x;
4      int y;
5
6      public void moveLeft() {
7          x--;
8      }
9
10     public void moveRight() {
11         x++;
12     }
13
14     public void moveUp() {
15         y++;
16     }
17
18     public void moveDown() {
19         y--;
20     }
21
22     public void printPosition() {
23         System.out.println("Posisi snake: (" + x + ", " + y + ")");
24     }
25 }
26
```

```
package Praktikum02;
public class SnakeMain {
    Run | Debug
    public static void main(String[] args) {
        Snake snake1 = new Snake();
        Snake snake2 = new Snake();


        snake1.moveRight();
        snake1.moveUp();

        snake2.moveLeft();
        snake2.moveDown();

        System.out.println(x:"Posisi Snake 1:");
        snake1.printPosition();

        System.out.println(x:"Posisi Snake 2:");
        snake2.printPosition();
    }
}
```

Output

```
Posisi Snake 1:
Posisi snake: (1, 1)
Posisi Snake 2:
Posisi snake: (-1, -1)
PS C:\KULIAH\Semester 4\OOP Praktikum\Praktikum2>
```

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

Jawab : Jika method move() dipanggil persis setelah objek diinstansiasi tanpa melakukan perubahan pada nilai awal x dan y, maka objek akan tetap berada pada posisi awalnya yang ditetapkan secara default, yaitu x = 0 dan y = 0. Hal ini terjadi karena tidak ada perubahan posisi yang diberikan sebelum memanggil method move().

-

KODE :

```java
package Praktikum02;

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

    public void changeDirection(int newDirection) {
        if (newDirection >= 1 && newDirection <= 4) {
            direction = newDirection;
        } else {
            System.out.println("Arah yang dimasukkan tidak valid.");
        }
    }

    public void move(int steps) {
        switch (direction) {
            case 1: // Atas

                y += steps;
                break;
            case 2: // Kanan
                x += steps;
                break;
            case 3: // Bawah
                y -= steps;
                break;
            case 4: // Kiri
                x -= steps;
                break;
        }
    }

    public void printStatus() {
        String arahString = "";
        switch (direction) {
            case 1:
                arahString = "atas";
                break;
            case 2:
                arahString = "kanan";
                break;
            case 3:
                arahString = "bawah";
                break;
            case 4:
                arahString = "kiri";
                break;
        }
        System.out.println("Posisi Dragon: (" + x + ", " + y + ")");
        System.out.println("Arah Dragon: " + direction + " (" + arahString + ")");
        System.out.println("==================");
    }
}
```

```java
package Praktikum02;
public class DragonMain {
    Run | Debug
    public static void main(String[] args) {
        Dragon dragon1 = new Dragon();
        Dragon dragon2 = new Dragon();

        // Mengubah arah dan posisi dragon
        dragon1.changeDirection(newDirection:2);
        dragon1.move(steps:3);

        dragon2.changeDirection(newDirection:3);
        dragon2.move(steps:2);

        // Mencetak status dragon
        System.out.println(x:"Status Dragon 1:");
        dragon1.printStatus();

        System.out.println(x:"Status Dragon 2:");
        dragon2.printStatus();
    }
}
```

Output:

```
Status Dragon 1:
Posisi Dragon: (3, 0)
Arah Dragon: 2 (kanan)
==================
Status Dragon 2:
Posisi Dragon: (0, -2)
Arah Dragon: 3 (bawah)
==================
PS C:\KULIAH\Semester 4\OOP Praktikum\Praktikum2>
```