JOBSHEET II
OBJECT

Nama: Rizqi Bagus Andrean
Kelas TI-1D
Absen 25

1. Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Mengenal objek dan class sebagai konsep mendasar pada pemrograman berorientasi objek
2. Mendeklarasikan class, atribut dan method
3. Membuat objek (instansiasi)
4. Mengakses atribut dan method dari suatu objek
5. Menerapkan konstruktor

2. Praktikum

2.1 Percobaan 1: Deklarasi Class, Atribut dan Method

Waktu Percobaan : 50 Menit

Pada Percobaan 1 ini dilakukan pembuatan class beserta atribut dan method yang dimilikinya.

Perhatikan Class Diagram berikut ini:

Buku
judul: String pengarang: String halaman: int stok: int harga: int
tampilInformasi(): void terjual(jml: int): void restock(n: int): void gantiHarga(hrg: int): int

Berdasarkan class diagram tersebut, akan dibuat program mengguanakan bahasa Java.

2.1.1 Langkah-langkah Percobaan

1. Buka text editor. Buat file baru, beri nama **Buku<NoAbsen>.java**
2. Lengkapi class **Buku** dengan atribut yang telah digambarkan di dalam class diagram tersebut

```
String judul, pengarang;
int halaman, stok, harga;
```

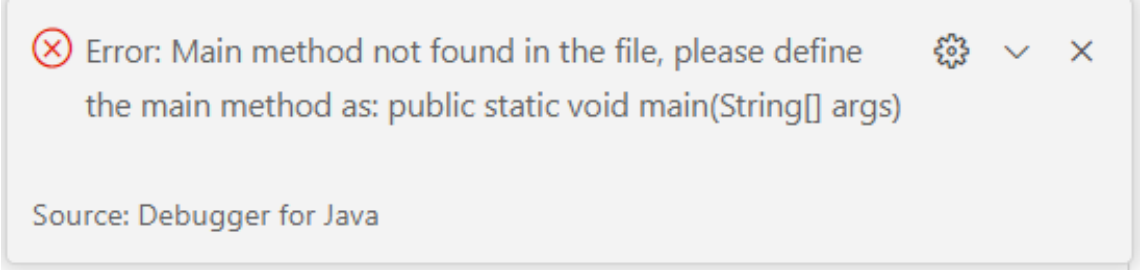
3. Lengkapi class **Buku** dengan method yang telah digambarkan di dalam class diagram tersebut

```
void tampilInformasi() {
    System.out.println("Judul: " + judul);
    System.out.println("Pengarang: " + pengarang);
    System.out.println("Jumlah halaman: " + halaman);
    System.out.println("Sisa stok: " + stok);
    System.out.println("Harga: Rp " + harga);
}

void terjual(int jml) {
    stok -= jml;
}

void restock(int jml) {
    stok += jml;
}

void gantiHarga(int hrg) {
    harga = hrg;
}
```

4. Compile dan run program.

2.1.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.



2.1.3 Pertanyaan

1. Sebutkan dua karakteristik class atau object!

   Class adalah sebuah cetakan atau kerangka untuk membuat **object**. **Object** adalah sebuah entitas yang memiliki **data** (atribut) dan **tindakan** (behavior)

2. Perhatikan class **Buku** pada Praktikum 1 tersebut, ada berapa atribut yang dimiliki oleh class Buku? Sebutkan apa saja atributnya! Ada 5, yaitu judul, pengarang, halaman, stock, harga

3. Ada berapa method yang dimiliki oleh class tersebut? Sebutkan apa saja methodnya!
   Ada 4 method, tampilInformasi, terjual, restock, gantiHarga

4. Perhatikan method **terjual()** yang terdapat di dalam class **Buku**. Modifikasi isi method tersebut sehingga proses pengurangan hanya dapat dilakukan jika stok masih ada (lebih besar dari 0)!

```
    }

    private static void terjual(int jml) {    The met
        if (stock >= 0) {
            stock -= jml;

        }else{
            System.out.println(x:"Stock Habis");
        }
```


5. Menurut Anda, mengapa method **restock()** mempunyai satu parameter berupa bilangan int? Untuk memasukkan jumlah restocknya
6. **Commit dan push kode program ke Github**

2.2 Percobaan 2: Instansiasi Object, serta Mengakses Atribut dan Method

Waktu Percobaan: 50 Menit

Sampai tahap ini, class **Buku** telah berhasil dibuat pada Percobaan 1. Selanjutnya, apabila class Buku tersebut ingin digunakan dan diakses atribut serta method-nya, maka perlu dibuat object/instance dari class **Buku** terlebih dahulu melalui proses instansiasi.

2.2.1 Langkah-langkah Percobaan

1. Buat file baru, beri nama **BukuMain<NoAbsen>.java**
2. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi **main()**
3. Di dalam fungsi **main(),** lakukan instansiasi, kemudian lanjutkan dengan mengakses atribut dan method dari objek yang telah terbentuk.

```
Buku bk1 = new Buku();
bk1.judul = "Today Ends Tomorrow Comes";
bk1.pengarang = "Denanda Pratiwi";
bk1.halaman = 198;
bk1.stok = 13;
bk1.harga = 71000;

bk1.tampilInformasi();
bk1.terjual(jml:5);
bk1.gantiHarga(hrg:60000);
bk1.tampilInformasi();
```

4. Compile dan run program.
5. **Commit dan push kode program ke Github**

2.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```
Judul: Today Ends Tomorrow Comes
Pengarang: Denanda Pratiwi
Jumlah halaman: 198
Sisa stok: 13
Harga: Rp 71000
Judul: Today Ends Tomorrow Comes
Pengarang: Denanda Pratiwi
Jumlah halaman: 198
Sisa stok: 8
Harga: Rp 60000
```

2.2.3 Pertanyaan

1. Pada class **BukuMain**, tunjukkan baris kode program yang digunakan untuk proses instansiasi! Apa nama object yang dihasilkan?

```
public static void main(String[] args) {
    Buku25 buku = new Buku25();
```

2. Bagaimana cara mengakses atribut dan method dari suatu objek?

```
buku.judul = "Today Ends Tomorrow Comes";    The s
buku.pengarang = "Denanda Pratiwi";    The static
buku.halaman = 198;    The static field Buku25.hal
buku.stock = 13;    The static field Buku25.stock
buku.harga = 71000;    The static field Buku25.har

buku.tampilInformasi();    The static method tampi
buku.terjual(jml:5);    The static method terjual(
buku.gantiHarga(hrg:60000);    The static method g
buku.tampilInformasi();    The static method tampi
```

3. Mengapa hasil output pemanggilan method **tampilInformasi()** pertama dan kedua berbeda?

Karena sebelum pemanggila kedua, meng eksekusi fungsi terjual dan ganti harga, sehingga membuat variable stock dan harga berubah.

2.3 Percobaan 3: Membuat Konstruktor

Waktu Percobaan: 60 Menit

Pada percobaan ini, dilakukan pembuatan kode program untuk mengimplementasikan berbagai macam konstruktor berdasarkan parameternya.

2.3.1 Langkah-langkah Percobaan

1. Buka kembali class **Buku**. Tambahkan dua buah konstruktor di dalam class **Buku** tersebut, yang terdiri dari satu konstruktor default dan satu konstruktor berparameter. Konstruktor merupakan method Istimewa, penempatan kode program untuk konstruktor dapat diperlakukan sama seperti method yang lain (setelah atribut).

```
public Buku() {

}

public Buku(String jud, String pg, int hal, int stok, int har) {
    judul = jud;
    pengarang = pg;
    halaman = hal;
    this.stok = stok;
    harga = har;
}
```

*Catatan: Apabila nama parameter sama dengan nama atribut, maka untuk merujuk pada variabel atribut ditambahkan sintaks **this** di depan nama **atribut***

2. Buka kembali class **BukuMain**. Buat sebuah object lagi bernama **bk2** dengan menggunakan konstruktor berparameter.

```
Buku bk1 = new Buku();
bk1.judul = "Today Ends Tomorrow Comes";
bk1.pengarang = "Denanda Pratiwi";
bk1.halaman = 198;
bk1.stok = 13;
bk1.harga = 71000;

bk1.tampilInformasi();
bk1.terjual(jml:5);
bk1.gantiHarga(hrg:60000);
bk1.tampilInformasi();
```

```
Buku bk2 = new Buku(jud:"Self Reward", pg:"Maheera Ayesha", hal:160, stok:29, har:59000);
bk2.terjual(jml:11);
bk2.tampilInformasi();
```

3. Compile dan run program.
4. **Commit dan push kode program ke Github**

2.3.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```
Judul: Today Ends Tomorrow Comes
Pengarang: Denanda Pratiwi
Jumlah halaman: 198
Sisa stok: 13
Harga: Rp 71000
Judul: Today Ends Tomorrow Comes
Pengarang: Denanda Pratiwi
Jumlah halaman: 198
Sisa stok: 8
Harga: Rp 60000
Judul: Self Reward
Pengarang: Maheera Ayesha
Jumlah halaman: 160
Sisa stok: 18
Harga: Rp 59000
```

2.3.3 Pertanyaan

1. Pada class **Buku** di Percobaan 3, tunjukkan baris kode program yang digunakan untuk mendeklarasikan konstruktor berparameter!

```
Buku25 buku2 = new Buku25(jud:"Self Reward", pg:"Maheer Ayesha", hal:160, stok:29, har:59000);
```

2. Perhatikan class **BukuMain**. Apa sebenarnya yang dilakukan pada baris program berikut?

```
Buku bk2 = new Buku(jud:"Self Reward", pg:"Maheera Ayesha", hal:160, stok:29, har:59000);
```

Menginisialisai class buku dengan mengisi konstruktornya agar instance buku memiliki nilai awal.

3. Hapus konstruktor default pada class **Buku**, kemudian compile dan run program. Bagaimana hasilnya? Jelaskan mengapa hasilnya demikian!
4. Setelah melakukan instansiasi object, apakah method di dalam class **Buku** harus diakses secara berurutan? Jelaskan alasannya!
5. Buat object baru dengan nama **buku<NamaMahasiswa>** menggunakan konstruktor berparameter dari class **Buku**!
6. **Commit dan push kode program ke Github**

2.4 Latihan Praktikum

Waktu : 150 Menit

1. Pada class Buku yang telah dibuat, tambahkan tiga method yaitu **hitungHargaTotal()**, **hitungDiskon()**, dan **hitungHargaBayar()** dengan penjelasan sebagai berikut:
   - Method **hitungHargaTotal()** digunakan untuk menghitung harga total yang merupakan

perkalian antara harga dengan jumlah buku yang terjual

- Method **hitungDiskon()** digunakan untuk menghitung diskon dengan aturan berikut:
  - Jika harga total lebih dari 150000, maka harga didiskon sebesar 12%
  - Jika harga total antara 75000 sampai 150000, maka harga didiskon sebesar 5%

- Jika harga total kurang dari 75000, maka harga tidak didiskon

- Method **hitungHargaBayar()** digunakan untuk menghitung harga total setelah dikurangi diskon

Class diagram **Buku** setelah penambahan ketiga method tersebut adalah sebagai berikut.

Buku
judul: String pengarang: String halaman: int stok: int harga: int
tampilInformasi(): void terjual(jml: int): void restock(n: int): void gantiHarga(hrg: int): int **hitungHargaTotal(): int** **hitungDiskon(): int** **hitungHargaBayar(): int**

2. Buat program berdasarkan class diagram berikut ini!

Dragon
x: int y: int width: int height: int
moveLeft(): void moveRight(): void moveUp(): void moveDown(): void printPosition(): void detectCollision(x: int, y: int): void

Penjelasan dari atribut dan method pada class Dragon tersebut adalah sebagai berikut:

- Atribut **x** digunakan untuk menyimpan posisi koordinat x (mendatar) dari dragon, sedangkan atribut **y** untuk posisi koordinat y (vertikal)
- Atribut **width** digunakan untuk menyimpan lebar dari area permainan, sedangkan **height** untuk menyimpan panjang area
- Method **moveLeft()** digunakan untuk mengubah posisi dragon ke kiri (koordinat x akan berkurang 1), sedangkan **moveRight()** untuk bergerak ke kanan (koordinat x akan bertambah 1). Perlu diperhatikan bahwa koordinat x tidak boleh lebih kecil dari 0 atau lebih besar dari nilai width. Jika koordinat x < 0 atau x > width maka panggil method **detectCollision()**
- Method moveUp() digunakan untuk mengubah posisi dragon ke atas (koordinat y akan berkurang 1), sedangkan moveDown() untuk bergerak ke bawah (koordinat y akan bertambah

1). Perlu diperhatikan bahwa koordinat y tidak boleh lebih kecil dari 0 atau lebih besar dari nilai height. Jika koordinat y < 0 atau y > height maka panggil method **detectCollision()**

- Method **detectCollision()** akan mencetak pesan “Game Over” apabila dragon menyentuh ujung area permainan.