
JOBSHEET II

OBJEK DAN ARRAY OF OBJECT

2.1 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
6. Memahami dan menjelaskan fungsi array yang berisikan variabel objek.
7. Mahasiswa mampu menangkap logika tentang permasalahan array of object dalam Java
8. Mahasiswa mampu menerapkan pembuatan array of object dalam Java

2.2 Deklarasi Class, Atribut dan Method

Waktu : 45 Menit

Perhatikan Diagram Class berikut ini:

Film
judul: String genre: String rate: String jumlahTiket: int hargaTiket: int
tampilFilm(): void tambahTiket(n: int): void kurangiTiket(n: int): void totalRevenue(jumlah: int): int

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

2.2.1 Langkah-langkah Percobaan

1. Buat Project baru, dengan nama **StrukturData**. Buat paket dengan nama minggu2 (opsional).
buatlah class baru dengan nama **Film**.

2. Lengkapi class `Film` dengan atribut dan method yang telah digambarkan di dalam diagram class di atas, sebagai berikut:

```
public class Film {
    String judul, genre, rate;
    int jumlahTiket, hargaTiket;

    void tampilFilm() {
        System.out.println("Judul: "+judul);
        System.out.println("Genre:"+genre);
        System.out.println("Rate: "+rate);
        System.out.println("Jumlah Tiket: "+jumlahTiket);
        System.out.println("Harga Tiket: "+hargaTiket);
    }

    void tambahTiket(int n) {
        jumlahTiket += n;
    }

    void kurangiTiket(int n) {
        jumlahTiket -= n;
    }

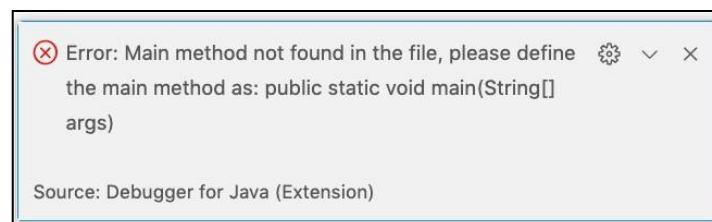
    int totalRevenue(int jumlah) {
        return jumlah*hargaTiket;
    }
}
```

3. Coba jalankan (Run) class `Barang` tersebut. Apakah bisa?

Karena konstruktor yang menerima parameter untuk menginisialisasi atribut-atribut objek tersebut.

2.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.



2.2.3 Pertanyaan

1. Sebutkan 2 karakteristik class/objek!

atribut dan method.

2. Kata kunci apakah yang digunakan untuk mendeklarasikan class?

Kata kunci yang digunakan untuk mendeklarasikan class adalah class

3. Perhatikan class `Film` yang ada di Praktikum di atas, ada berapa atribut yang dimiliki oleh class tersebut? Sebutkan! Dan pada baris berapa saja deklarasi atribut dilakukan?

Class Film memiliki lima atribut, yaitu:

- *judul*, tipe data *String*, pada baris 2
- *genre*, tipe data *String*, pada baris 3
- *rate*, tipe data *String*, pada baris 4
- *jumlahTiket*, tipe data *int*, pada baris 5
- *hargaTiket*, tipe data *int*, pada baris 6

Jadi, deklarasi atribut dilakukan pada baris 2 hingga baris 6.

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

- Ada 4 yaitu *Film()*, *tampilFilm()*, *tambahTiket()*, *kurangiTiket()*.

5. Perhatikan method **kurangiTiket()** yang ada di class *Film*, modifikasi isi method tersebut sehingga proses pengurangan hanya dilakukan jika stok masih ada (masih lebih besar dari 0)

```
Film film = new Film("Judul Film", "Genre Film", "Rate Film", 100, 50000);
film.kurangiTiket(10);
// Jumlah tiket berkurang menjadi 90

film.kurangiTiket(100);
// Jumlah tiket tidak mencukupi
```

-
6. Menurut Anda, mengapa method `tambahTiket()` dibuat dengan memiliki 1 parameter berupa bilangan int?

karena parameter tersebut digunakan untuk menentukan jumlah tiket yang akan ditambahkan. Parameter ini dideklarasikan sebagai tipe data int karena jumlah tiket harus berupa bilangan bulat.

7. Menurut Anda, mengapa method `totalRevenue()` memiliki tipe data int?

karena hasil dari perhitungannya adalah berupa bilangan bulat. Perhitungan total revenue dilakukan dengan mengalikan jumlah tiket dengan harga tiket. Jumlah tiket dan harga tiket keduanya merupakan bilangan bulat, sehingga hasil dari perkalian tersebut juga merupakan bilangan bulat.

8. Menurut Anda, mengapa method `tambahTiket()` memiliki tipe data void?

Method `tambahTiket()` memiliki tipe data void karena method ini tidak mengembalikan nilai apa pun. Method ini hanya melakukan operasi penambahan jumlah tiket.

Jika method `tambahTiket()` memiliki tipe data int, maka method tersebut harus mengembalikan nilai jumlah tiket yang baru. Namun, dalam hal ini, jumlah tiket yang baru sudah dapat diketahui langsung dari operasi penambahan yang dilakukan oleh method tersebut. Oleh karena itu, tidak perlu mengembalikan nilai tersebut secara eksplisit

2.3 Instansiasi Objek dan Mengakses Atribut & Method

Waktu : 45 Menit

Sampai tahap ini, kita telah membuat class `Film` dengan sukses. Selanjutnya, apabila diinginkan untuk mulai menggunakan class `Film` tersebut, mengakses atribut-atribut dan method-method yang ada di dalamnya, maka selanjutnya perlu dibuat objek/instance dari class `Film` terlebih dahulu.

2.3.1 Langkah-langkah Percobaan

1. Buatlah class baru dengan nama `FilmMain`. Dan di dalam class `FilmMain` tersebut, buatlah method `main()`.
2. Di dalam method `main()`, lakukan instansiasi, dan kemudian lanjutkan dengan mengakses atribut dan method dari objek yang telah terbentuk.

```

public class FilmMain {
    public static void main(String[] args) {
        Film film1 = new Film();

        film1.judul = "Quantumania Mancing";
        film1.genre = "Action Comedy";
        film1.rate = "Remaja";
        film1.jumlahTiket = 3000;
        film1.hargaTiket = 40000;

        film1.tambahTiket(1);
        film1.kurangiTiket(3);
        film1.tampilFilm();

        int income = film1.totalRevenue(4);

        System.out.println("Total jual 4 tiket = "+income);
    }
}

```

3. Jalankan (Run) class **FilmMain** tersebut dan amati hasilnya.

```

Jumlah Tiket:2000
Harga Tiket:40000
PS C:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2> c:: cd '
c:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2' & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\YHEZHANDYA\AppData\Roaming\Code\User\workspacestorage\2eb9f83b7ab05cd5afad4a05dce862e04\redhat.java\jdk_ws\praktikum_2_ad\code\2\bin' 'FilmMain'
Judul:Quantum Mancing
Genre:Action Comedy
Rate:Remaja
Jumlah Tiket:2998
Harga Tiket:40000
Total jual 4 tiket= 160000
=====
Judul:Merajuantum
Genre:koror
Rate:Dewasa
Jumlah Tiket:2000
Harga Tiket:40000
PS C:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2>

```

2.3.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```

Judul: Quantumania Mancing
Genre: Action Comedy
Rate: Remaja
Jumlah Tiket: 2998
Harga Tiket: 40000
Total jual 4 tiket = 160000

```

2.3.3 Pertanyaan

1. Pada class `FilmMain`, pada kode apa yang digunakan untuk proses instansiasi? Apa nama objek yang dihasilkan? **Dalam bahasa pemrograman Java, proses pembuatan instance objek dilakukan dengan menggunakan kata kunci new diikuti dengan nama kelas (konstruktor) yang akan dibuat instance-nya. Jika Anda memiliki kelas bernama film**

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

Dalam bahasa pemrograman Java, Anda dapat menggunakan operator "." untuk mengakses properti (variabel) dan metode (fungsi) suatu objek. (Melihat). Berikut adalah contoh cara mengakses properti dan metode suatu objek:

Misalkan kita memiliki kelas film, yang memiliki properti title dan metode displayInformation, lalu kita membuat instance objek film dan mengakses properti dan metode objek tersebut.

Kesimpulannya

Kami menggunakan objek film untuk mengakses properti judul dan metode film `object.showInformation()` untuk mengakses metode `displayInformation`. Perhatikan bahwa kita menggunakan objek film yang dipakai sebelumnya untuk mengakses properti dan metode yang terdapat dalam kelas film.

2.4 Membuat Konstruktor

Waktu : 45 Menit

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

2.4.1 Langkah-langkah Percobaan

1. Perhatikan kembali class `Film`. Tambahkan di dalam class `Film` tersebut 2 buah konstruktor. 1 konstruktor default dan 1 konstruktor berparameter.

```
public class Film {
    String judul, genre, rate;
    int jumlahTiket, hargaTiket;

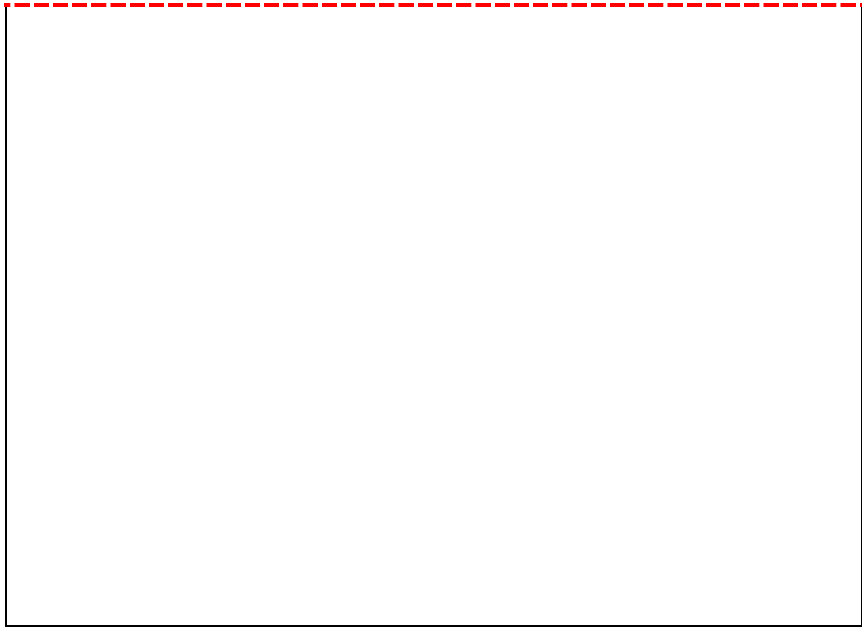
    Film() {
    }

    Film(String jd, String gr, String rt, int jt, int ht) {
        judul = jd;
        genre = gr;
        rate = rt;
        jumlahTiket = jt;
        hargaTiket = ht;
    }

    void tampilFilm() {
        System.out.println("Judul: "+judul);
        System.out.println("Genre: "+genre);
        System.out.println("Rate: "+rate);
        System.out.println("Jumlah Tiket: "+jumlahTiket);
        System.out.println("Harga Tiket: "+hargaTiket);
    }

    void tambahTiket(int n) {
        jumlahTiket += n;
    }

    void kurangiTiket(int n) {
        jumlahTiket -= n;
    }
}
```



2. Buka kembali class **FilmMain**. Dan buat sebuah objek lagi, kali ini dengan menggunakan konstruktor berparameter.

```
public class FilmMain {
    public static void main(String[] args) {
        Film film1 = new Film();

        film1.judul = "Quantumania Mancing";
        film1.genre = "Action Comedy";
        film1.rate = "Remaja";
        film1.jumlahTiket = 3000;
        film1.hargaTiket = 40000;

        film1.tambahTiket(1);
        film1.kurangiTiket(3);
        film1.tampilFilm();

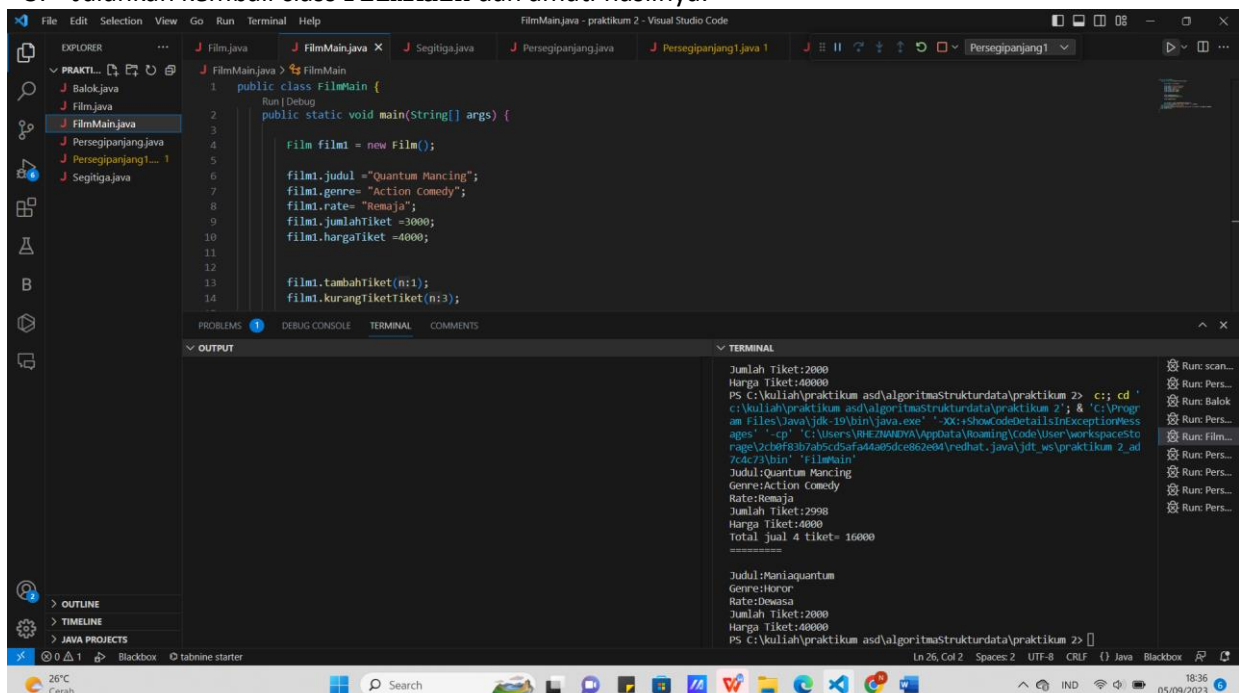
        int income = film1.totalRevenue(4);

        System.out.println("Total jual 4 tiket = "+income);

        System.out.println("=====\n");

        Film film2 = new Film("Maniaquantum", "Horor", "Dewasa", 2000, 40000);
        film2.tampilFilm();
    }
}
```

3. Jalankan kembali class **FilmMain** dan amati hasilnya.



The screenshot shows the Visual Studio Code interface with the **FilmMain.java** file open. The code is as follows:

```
public class FilmMain {
    public static void main(String[] args) {
        Film film1 = new Film();

        film1.judul = "Quantumania Mancing";
        film1.genre = "Action Comedy";
        film1.rate = "Remaja";
        film1.jumlahTiket = 3000;
        film1.hargaTiket = 40000;

        film1.tambahTiket(1);
        film1.kurangiTiket(3);
        film1.tampilFilm();

        int income = film1.totalRevenue(4);

        System.out.println("Total jual 4 tiket = "+income);

        System.out.println("=====\n");

        Film film2 = new Film("Maniaquantum", "Horor", "Dewasa", 2000, 40000);
        film2.tampilFilm();
    }
}
```

The terminal output shows the results of the program execution:

```
Jumlah Tiket:2000
Harga Tiket:40000
PS C:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2> cd '
c:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2' & 'c:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\WHE\I\ANOWA\AppData\Roaming\Code\User\workspacestorage\2c8b0f83b7ab5cd5af44a05dce862e04\redhat.java\jdt_ws\praktikum_2_ad7cd293\bin' 'FilmMain'
Judul:Quantumania Mancing
Genre:Action Comedy
Rate:Remaja
Jumlah Tiket:2998
Harga Tiket:4000
Total jual 4 tiket= 16000
=====
Judul:Maniaquantum
Genre:Horor
Rate:Dewasa
Jumlah Tiket:2000
Harga Tiket:40000
PS C:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2>
```

2.4.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

Judul: Quantumania Mancing
Genre: Action Comedy
Rate: Remaja
Jumlah Tiket: 2998
Harga Tiket: 40000
Total jual 4 tiket = 160000

=====

Judul: Maniaquantum
Genre: Horror
Rate: Dewasa
Jumlah Tiket: 2000
Harga Tiket: 40000

2.4.3 Pertanyaan

1. Perhatikan class `Film` yang ada di Praktikum 2.4.1, pada baris berapakah deklarasi konstruktor berparameter dilakukan? **6 yaitu beberapa parameter seperti judul, genre, rate, jumlah tiket, dan harga tiket.**
2. Perhatikan class `FilmMain` di Praktikum 2.4.1, apa sebenarnya yang dilakukan pada baris program dibawah ini?

untuk membuat objek sekaligus mengatur nilainya saat objek tersebut dibuat.

```
Film film2 = new Film("Maniaquantum", "Horor", "Dewasa", 2000, 40000);
```

3. Coba buat objek dengan nama `film3` dengan menggunakan konstruktor berparameter dari class `Barang`.

```
Film film3 = new Film("Film3", "Genre3", "Rate3", 100, 50000);
```

2.5 Membuat Array dari Object, Mengisi dan Menampilkan

Didalam praktikum ini, kita akan mempraktekkan bagaimana membuat array dari object, kemudian mengisi dan menampilkan array tersebut.

2.5.1 Langkah-langkah Percobaan

1. Buat Project baru, dengan nama `"ArrayObjects"`.
2. Buat class `PersegiPanjang`:

```
public class PersegiPanjang {  
    public int panjang;  
    public int lebar;  
  
}
```

3. Pada fungsi main yaitu pada class `ArrayObjects`, buatlah array `PersegiPanjang` yang berisi 3 elemen:

```
public static void main(String[] args) {  
    PersegiPanjang[] ppArray = new PersegiPanjang[3];  
}
```

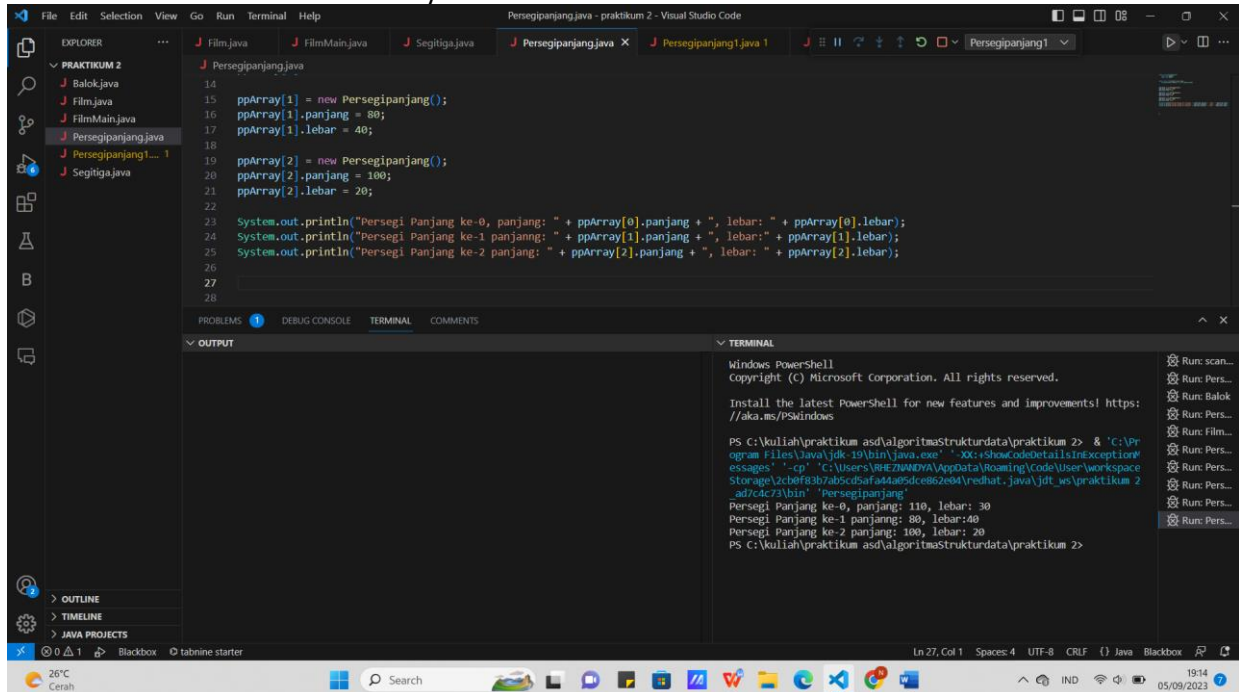
4. Kemudian isikan masing-masing atributnya:

```
ppArray[0] = new PersegiPanjang();  
ppArray[0].panjang = 110;  
ppArray[0].lebar = 30;  
  
ppArray[1] = new PersegiPanjang();  
ppArray[1].panjang = 80;  
ppArray[1].lebar = 40;  
  
ppArray[2] = new PersegiPanjang();  
ppArray[2].panjang = 100;  
ppArray[2].lebar = 20;
```

5. Cetak ke layar semua atribut dari objek `ppArray`:

```
System.out.println("Persegi Panjang ke-0, panjang: " + ppArray[0].panjang + ", lebar: " + ppArray[0].lebar);
System.out.println("Persegi Panjang ke-1, panjang: " + ppArray[1].panjang + ", lebar: " + ppArray[1].lebar);
System.out.println("Persegi Panjang ke-2, panjang: " + ppArray[2].panjang + ", lebar: " + ppArray[2].lebar);
```

6. Jalankan dan amati hasilnya.



2.5.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```
run:
Persegi Panjang ke-0, panjang: 110, lebar: 30
Persegi Panjang ke-1, panjang: 80, lebar: 40
Persegi Panjang ke-2, panjang: 100, lebar: 20
BUILD SUCCESSFUL (total time: 0 seconds)
```

2.5.3 Pertanyaan

1. Berdasarkan uji coba 3.2, apakah class yang akan dibuat array of object harus selalu memiliki atribut dan sekaligus method?Jelaskan! **Tidak, class yang akan dibuat sebagai array of object tidak harus memiliki atribut dan metode secara bersamaan. Anda dapat memiliki class yang hanya memiliki atribut tanpa metode, atau sebaliknya, hanya memiliki metode tanpa atribut. Ketergantungan antara atribut dan metode dalam class seharusnya bergantung pada kebutuhan program Anda.**
2. Apakah class `PersegiPanjang` memiliki konstruktor?Jika tidak, kenapa dilakukan pemanggilan

konstruktur pada baris program berikut :

```
ppArray[1] = new PersegiPanjang();
```

tidak memiliki deklarasi konstruktor khusus yang ditulis oleh Anda. Ketika Anda tidak mendeklarasikan konstruktor dalam class, Java akan memberikan konstruktor default tanpa parameter secara otomatis.

3. Apa yang dimaksud dengan kode berikut ini:

```
PersegiPanjang[] ppArray = new PersegiPanjang[3];
```

"Setiap elemen dalam array ini akan menjadi objek PersegiPanjang, dan karena Anda tidak memberikan nilai awal untuk setiap elemen, maka objek-objek tersebut akan diinisialisasi dengan nilai default. Anda perlu membuat dan menginisialisasi setiap objek PersegiPanjang secara terpisah sebelum Anda dapat mengakses atau menggunakan mereka dalam array. Namun, perlu diperhatikan bahwa penggunaan `new PersegiPanjang` hanya akan mengalokasikan memori untuk array tersebut. `new PersegiPanjang` adalah bagian yang mengalokasikan memori untuk array tersebut dengan panjang 3."

4. Apa yang dimaksud dengan kode berikut ini:

```
ppArray[1] = new PersegiPanjang();  
ppArray[1].panjang = 80;  
ppArray[1].lebar = 40;
```

"Setiap elemen dalam array tersebut perlu diinisialisasi secara terpisah, karena array hanya mengalokasikan memori untuk objek tanpa menginisialisasi nilainya secara otomatis. Dengan `ppArray.panjang = 110;` dan `ppArray.lebar = 30;`, Anda mengatur nilai-nilai atribut panjang dan lebar dari objek dalam elemen pertama dari array."

5. Mengapa class main dan juga class PersegiPanjang dipisahkan pada uji coba 3.2?

karena praktik yang umum dalam pemrograman berorientasi objek dan Java

2.6 Menerima Input Isian Array Menggunakan Looping

Pada praktikum ini kita akan mengubah hasil program dari praktikum 3.2 sehingga program dapat menerima input dan menggunakan looping untuk mengisi atribut dari semua persegi panjang yang ada di `ppArray`.

2.6.1 Langkah-langkah Percobaan

1. Import scanner pada class **ArrayObjects**.

```
import java.util.Scanner;
```

Note: Letakkan kode import dibawah kode package.

2. Pada praktikum 3.2 poin nomor 4, ganti kodenya dengan kode berikut ini, yaitu membuat objek **Scanner** untuk menerima input, kemudian melakukan looping untuk menerima input:

```
Scanner sc = new Scanner(System.in);

for(int i = 0; i < 3; i++)
{
    ppArray[i] = new PersegiPanjang();
    System.out.println("Persegi panjang ke-" + i);
    System.out.print("Masukkan panjang: ");
    ppArray[i].panjang = sc.nextInt();
    System.out.print("Masukkan lebar: ");
    ppArray[i].lebar = sc.nextInt();
}
```

3. Pada praktikum 3.2 poin nomor 5, ganti kodenya dengan berikut ini, yaitu melakukan looping untuk mengakses isi array **ppArray** dan menampilkannya ke layar:

```
for(int i = 0; i < 3; i++)
{
    System.out.println("Persegi Panjang ke-" + i);
    System.out.println("Panjang: " + ppArray[i].panjang + ", lebar: " + ppArray[i].lebar);
}
```

4. Jalankan dan amati hasilnya.

2.6.2 Verifikasi Hasil Percobaan

Contoh verifikasi hasil percobaan ini.

```
Persegi panjang ke-0  
Masukkan panjang: 5  
Masukkan lebar: 6  
Persegi panjang ke-1  
Masukkan panjang: 5  
Masukkan lebar: 3  
Persegi panjang ke-2  
Masukkan panjang: 4  
Masukkan lebar: 8  
Persegi Panjang ke-0  
Panjang: 5, lebar: 6  
Persegi Panjang ke-1  
Panjang: 5, lebar: 3  
Persegi Panjang ke-2  
Panjang: 4, lebar: 8
```

2.6.3 Pertanyaan

1. Apakah array of object dapat diimplementasikan pada array 2 Dimensi?

Ya, array of object dapat diimplementasikan pada array 2 dimensi.

2. Jika jawaban soal no satu iya, berikan contohnya! Jika tidak, jelaskan!

```

import java.util.Scanner;

public class ArrayOfObject {
    public static void main(String[] args) {
        Student[] students = new Student[2][3];
        Scanner sc = new Scanner(System.in);

        for (int i = 0; i < 2; i++) {
            for (int j = 0; j < 3; j++) {
                students[i][j] = new Student();

                System.out.println("Mahasiswa ke-" + (i * 3 + j));
                System.out.print("Masukkan nama: ");
                students[i][j].nama = sc.nextLine();

                System.out.print("Masukkan NIM: ");
                students[i][j].nim = sc.nextInt();

                System.out.print("Masukkan IPK: ");
                students[i][j].ipk = sc.nextDouble();
            }
        }

        for (int i = 0; i < 2; i++) {
            for (int j = 0; j < 3; j++) {
                System.out.println("Mahasiswa ke-" + (i * 3 + j));
                System.out.println("Nama: " + students[i][j].nama);
                System.out.println("NIM: " + students[i][j].nim);
                System.out.println("IPK: " + students[i][j].ipk);
            }
        }
    }

    class Student {
        public String nama;
        public int nim;
        public double ipk;
    }
}


```


-
3. Jika diketahui terdapat class **Persegi** yang memiliki atribut sisi bertipe integer, maka kode dibawah ini akan memunculkan error saat dijalankan. Mengapa?

```
Persegi[] pgArray = new Persegi[100];  
pgArray[5].sisi = 20;
```

Kode tersebut akan memunculkan error saat dijalankan karena indeks array `pgArray` tidak boleh melebihi kapasitas array tersebut. Pada kode tersebut, kapasitas array `pgArray` adalah 100, sedangkan indeks 5 lebih besar dari 100.

4. Modifikasi kode program pada praktikum 3.3 agar length array menjadi inputan dengan Scanner!



```

import java.util.Scanner;

public class ArrayOfObject {
    public static void main(String[]
args) {
        Scanner sc = new
Scanner(System.in);

        System.out.print("Masukkan
panjang array: ");
        int length = sc.nextInt();

        Persegi[] persegiArray = new
Persegi[length];

        for (int i = 0; i < length;
i++) {
            persegiArray[i] = new
Persegi();

            System.out.println("Persegi
ke-" + (i + 1));
            System.out.print("Masukkan
sisi: ");
            persegiArray[i].sisi =
sc.nextInt();
        }

        for (int i = 0; i < length;
i++) {
            System.out.println("Persegi
ke-" + (i + 1));
            System.out.println("Sisi: "
+ persegiArray[i].sisi);
        }
    }
}

class Persegi {
    public int sisi;
}

```

5. Apakah boleh jika terjadi duplikasi instansiasi array of objek, misalkan saja instansiasi dilakukan pada `ppArray[i]` sekaligus `ppArray[0]`? Jelaskan!

tidak boleh jika terjadi duplikasi instansiasi array of objek. Duplikasi instansiasi array of objek akan menyebabkan adanya dua referensi ke objek yang sama. Hal ini dapat menyebabkan masalah jika salah satu referensi tersebut diubah, maka referensi lainnya juga akan ikut berubah.

2.7 Operasi Matematika Atribut Object Array

Pada praktikum ini kita akan melakukan pengoperasian matematika beberapa atribut pada masing-masing anggota array.

2.7.1 Langkah-langkah Percobaan

1. Buat package baru "ArrayBalok".
2. Buat class **Balok**:

```
public class Balok {
    public int panjang;
    public int lebar;
    public int tinggi;

    public Balok(int p, int l, int t)
    {
        panjang = p;
        lebar = l;
        tinggi = t;
    }

    public int hitungVolume()
    {
        return panjang * lebar * tinggi;
    }
}
```

3. Pada fungsi **main** yaitu pada class **ArrayBalok**, buat array **Balok** yang berisi 3 elemen:

```
public static void main(String[] args) {
    Balok[] blArray = new Balok[3];
}
```

4. Kemudian tambahkan kode berikut ini untuk mengisi array **blArray** menggunakan konstruktor dari class **Balok**:

```
blArray[0] = new Balok(100, 30, 12);
blArray[1] = new Balok(120, 40, 15);
blArray[2] = new Balok(210, 50, 25);
```

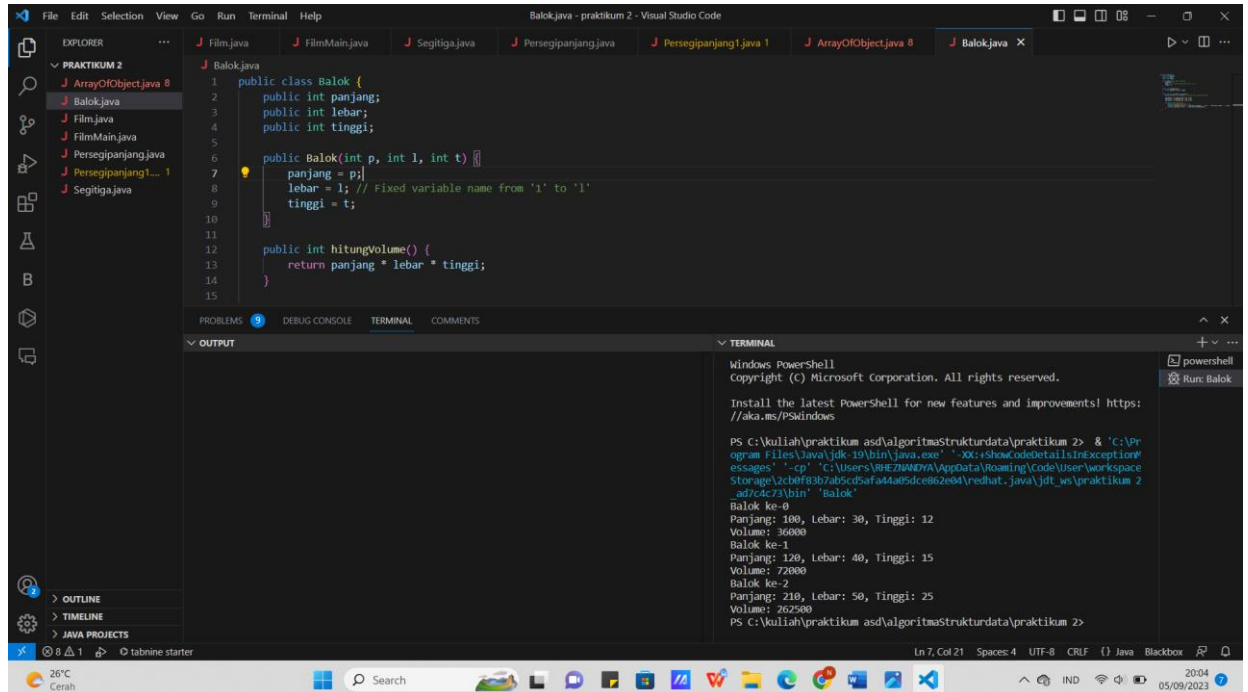
5. Tampilkan semua volume balok tersebut dengan cara memanggil method **hitungVolume()** di dalam looping seperti berikut ini:

```

for(int i = 0; i < 3; i++)
{
    System.out.println("Volume balok ke " + i + ": " + blArray[i].hitungVolume());
}

```

6. Jalankan dan amati hasilnya.



2.7.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program anda dengan gambar berikut ini.

```

run:
Volume balok ke 0: 36000
Volume balok ke 1: 72000
Volume balok ke 2: 262500

```

2.7.3 Pertanyaan

1. Dapatkah konstruktor berjumlah lebih dalam satu kelas? Jelaskan dengan contoh!

Ya, konstruktor dapat berjumlah lebih dalam satu kelas. Hal ini memungkinkan kita untuk membuat konstruktor dengan berbagai parameter yang berbeda untuk memenuhi kebutuhan yang berbeda.

Contoh:



```

public class Mobil {
    public String merk;
    public String tipe;
    public int tahun;

    // Konstruktor tanpa parameter
    public Mobil() {
        merk = "Tidak Diketahui";
        tipe = "Tidak Diketahui";
        tahun = 0;
    }

    // Konstruktor dengan parameter
    public Mobil(String merk, String
tipe, int tahun) {
        this.merk = merk;
        this.tipe = tipe;
        this.tahun = tahun;
    }
}

```

2. Jika diketahui terdapat class **Segitiga** seperti berikut ini:

```

public class Segitiga {
    public int alas;
    public int tinggi;
}

```

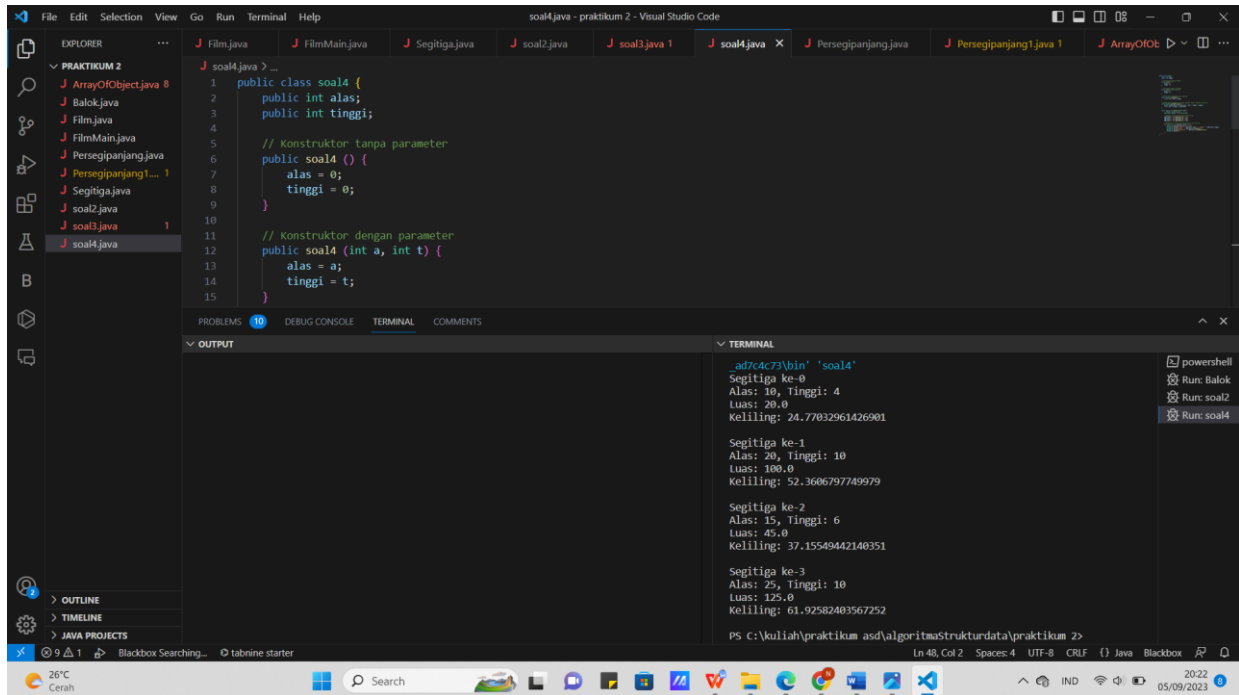
Tambahkan konstruktor pada class **Segitiga** tersebut yang berisi parameter **int a**, **int t** yang masing-masing digunakan untuk mengisi atribut **alas** dan **tinggi**.

3. Tambahkan method `hitungLuas()` dan `hitungKeliling()` pada class `Segitiga` tersebut. Asumsi segitiga adalah segitiga siku-siku. (Hint: Anda dapat menggunakan bantuan library `Math` pada Java untuk mengkalkulasi sisi miring)

4. Pada fungsi `main`, buat array `Segitiga sgArray` yang berisi 4 elemen, isikan masing-masing atributnya sebagai berikut:

sgArray ke-0	alas: 10, tinggi: 4
sgArray ke-1	alas: 20, tinggi: 10
sgArray ke-2	alas: 15, tinggi: 6
sgArray ke-3	alas: 25, tinggi: 10

Kemudian menggunakan looping, cetak luas dan keliling dengan cara memanggil method `hitungLuas()` dan `hitungKeliling()`.



```
1 public class soal4 {
2     public int alas;
3     public int tinggi;
4
5     // Konstruktor tanpa parameter
6     public soal4 () {
7         alas = 0;
8         tinggi = 0;
9     }
10
11    // Konstruktor dengan parameter
12    public soal4 (int a, int t) {
13        alas = a;
14        tinggi = t;
15    }
16
17    // Method hitungLuas
18    public void hitungLuas() {
19        System.out.println("Luas: " + alas * tinggi);
20    }
21
22    // Method hitungKeliling
23    public void hitungKeliling() {
24        System.out.println("Keliling: " + 2 * alas + 2 * tinggi);
25    }
26
27    // Main Method
28    public static void main (String[] args) {
29        // Objek soal4
30        soal4 s0 = new soal4 ();
31        soal4 s1 = new soal4 (20, 10);
32        soal4 s2 = new soal4 (15, 6);
33        soal4 s3 = new soal4 (25, 10);
34
35        // Looping
36        for (int i = 0; i < 4; i++) {
37            s0.hitungLuas();
38            s0.hitungKeliling();
39        }
40    }
41 }
```

Terminal Output:

```
PS c:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2>
Segitiga ke-0
Alas: 10, Tinggi: 4
Luas: 20.0
Keliling: 24.77832961426901

Segitiga ke-1
Alas: 20, Tinggi: 10
Luas: 100.0
Keliling: 52.3606797749979

Segitiga ke-2
Alas: 15, Tinggi: 6
Luas: 45.0
Keliling: 37.15549442148351

Segitiga ke-3
Alas: 25, Tinggi: 10
Luas: 125.0
Keliling: 61.92582403567252

PS c:\kuliah\praktikum asd\algoritmaStrukturdata\praktikum 2>
```

2.8 Tugas Praktikum

Waktu : 60 Menit

1. Buat program berdasarkan diagram class berikut ini!

Nasabah
Id: int nama: String alamat: String noHP: int noRek: int saldo: int jmlHutang: float statusAktif: boolean
lihatSaldo(): int menabung(deposit: int): int tarikTunai(jmlTarik: int): int bukaRekening(): void tutupRekening(): void berhutang(jmlHutang: int): void

- Method lihatSaldo() digunakan untuk melihat kondisi saldo tabungan
- Method menabung() digunakan untuk menambah saldo sesuai nominal parameter deposit yang dimasukkan
- Method tarikTunai() digunakan untuk mengurangi saldo sejumlah parameter jmlTarik
- Method bukaRekening() digunakan nasabah untuk pertama kali. Jika nasabah belum memiliki rekening, maka statusAktif false dan tidak boleh melakukan aktifitas lihatSaldo(), menabung(), tarikTunai(), tutupRekening(), dan berhutang().
- Method tutupRekening() digunakan untuk menonaktifkan rekening sehinggalan nasabah tidak lagi dapat melakukan aktifitas lihatSaldo(), menabung(), tarikTunai(), tutupRekening(), dan berhutang().
- Method berhutang() digunakan untuk mengajukan pinjaman dengan aturan berikut:
 - Jika jmlHutang >= saldo, akan muncul peringatanajuan peminjaman ditolak
 - Jika jmlHutang <= saldo, pinjaman disetujui dan dimunculkan simulasi skema cicilan per bulan
 - Seluruh cicilan harus dihitung dalam waktu 6 bulan
 - Simulasi skema cicilan per bulan dihitung dari jumlah hutang di rekening/6. Jangan lupa menggunakan konversi tipe data int ke float.

2. Berdasarkan soal nomor 1, terapkan kasus lebih dari 1 nasabah menggunakan object of array!



3. Berdasarkan soal nomor 1 dan 2, terapkan kasus 1 nasabah dapat memiliki lebih dari 1 rekening menggunakan array biasa!

