

Laporan Jobsheet 04 - Relasi Kelas

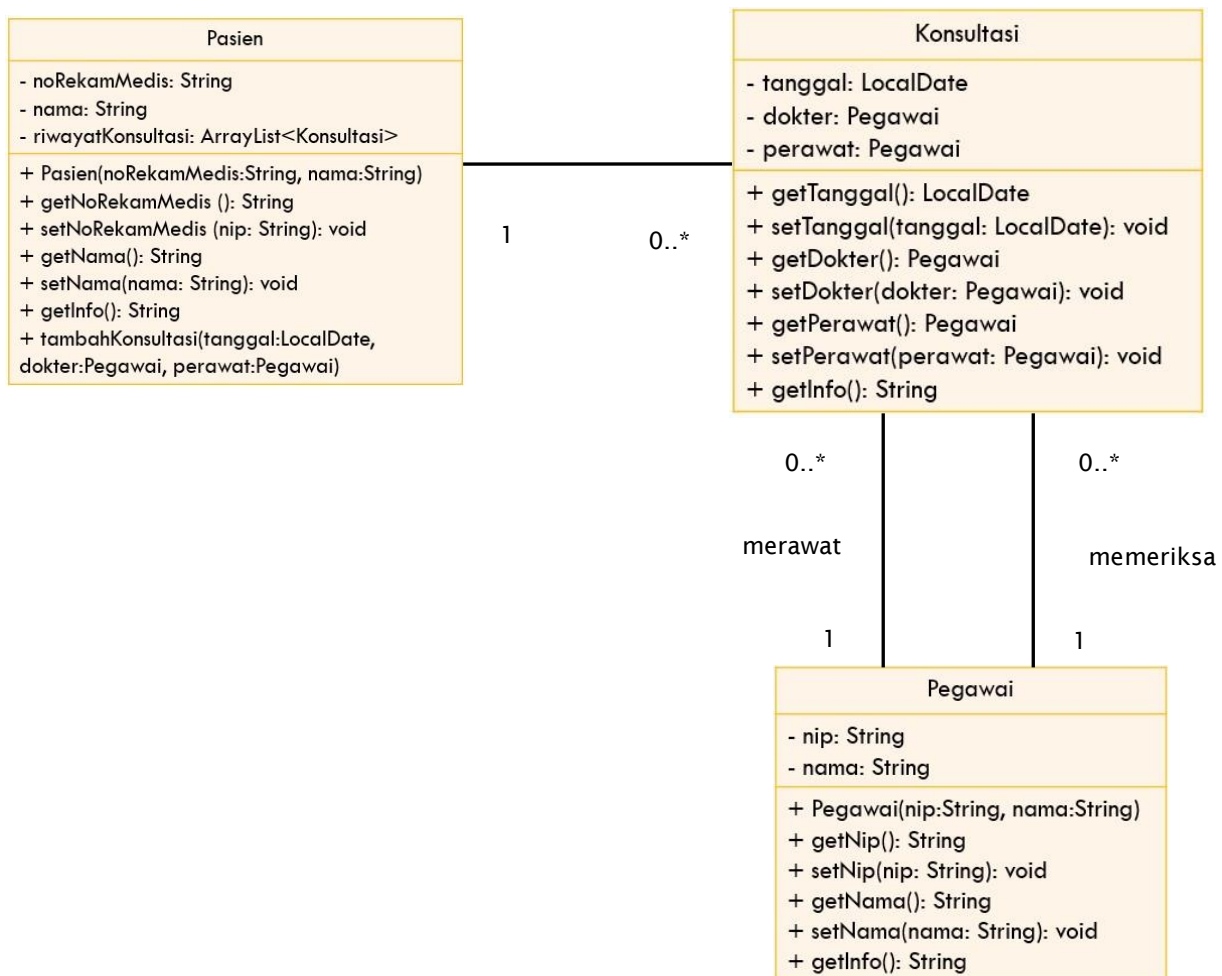
Nama : Rizqi Rohmatul Huda
Kelas / No.Abs : 2G -TI / 26
NIM : 2141720264

I. Pendahuluan

Pada kasus yang lebih kompleks, dalam suatu sistem akan ditemukan lebih dari satu *class* yang saling memiliki keterkaitan antara *class* satu dengan yang lain. Pada percobaan-percobaan sebelumnya, mayoritas kasus yang sudah dikerjakan hanya fokus pada satu *class* saja. Pada jobsheet ini akan dilakukan percobaan yang melibatkan beberapa *class* yang saling berkaitan.

II. Praktikum

- Pada praktikum ini akan dikembangkan suatu sistem informasi rumah sakit yang menyimpan data riwayat konsultasi pasien.
- Perhatikan diagram *class* berikut:



- Buatlah *project* baru di *Netbeans* dengan nama *RumahSakit*

- d. Buatlah class Pegawai dalam *package* rumahsakit. Tambahkan atribut nip dan nama pada class Pegawai dengan akses modifier private

```
public class Pegawai {  
    private String nip;  
    private String nama;  
}
```

- e. Buatlah *constructor* untuk class Pegawai dengan parameter nip dan nama.

```
public Pegawai (String nip, String nama){  
    this.nip = nip;  
    this.nama = nama;  
}
```

- f. Implementasikan **setter** dan **getter** untuk class Pegawai.

```
public String getNip() {  
    return this.nip;  
}  
  
public void setNip(String nip) {  
    this.nip = nip;  
}  
  
public String getNama() {  
    return this.nama;  
}  
  
public void setNama(String nama) {  
    this.nama = nama;  
}
```

- g. Implementasikan *method* getInfo() sebagai berikut:

```
public String getInfo() {  
    return nama + " (" + nip + ")";  
}
```

- h. Selanjutnya buatlah class Pasien dengan cara yang sama dengan class Pegawai. Tambahkan atribut noRekamMedis dan nama pada class Pasien dengan access level modifier private. Sediakan pula setter dan getter untuk kedua atribut tersebut.

```

public class Pasien {
    private String noRekamMedis;
    private String nama;
    private ArrayList <Konsultasi> riwayatKonsultasi;
    public String getNoRekamMedis () {
        return noRekamMedis;
    }

    public void setNoRekamMedis(String noRekamMedis) {
        this.noRekamMedis = noRekamMedis;
    }

    public String getNama () {
        return nama;
    }

    public void setNama(String nama){
        this.nama = nama;
    }

    public Pasien(String nama, String noRekamMedis) {
        this.noRekamMedis = noRekamMedis;
        this.nama = nama;
        this.riwayatKonsultasi = new ArrayList<>();
    }
}

```

- i. Buatlah constructor untuk class Pegawai dengan parameter noRekamMedis dan nama

```

public Pasien(String nama, String noRekamMedis) {
    this.noRekamMedis = noRekamMedis;
    this.nama = nama;
}

```

- j. Implementasikan *method* getInfo () sebagai berikut:

```

public String getInfo () {
    String info = "";
    info += "No Rekam Medis          : " + this.noRekamMedis + "\n";
    info += "Nama                      : " + this.nama + "\n";
    info += "\n";

    return info;
}

```

- k. Sistem ini akan menyimpan data setiap konsultasi yang dilakukan pasien. Pasien bisa melakukan konsultasi lebih dari sekali. Oleh karena itu, data konsultasi akan disimpan dalam bentuk ArrayList dari objek-objek yang bertipe konsultasi.

1. Buatlah class dengan nama `Konsultasi` dengan atribut tanggal bertipe `LocalDate`, dokter bertipe `Pegawai`, dan perawat bertipe `Pegawai` dengan access level modifier `private`. Lakukan `import java.time.LocalDate` agar dapat mendeklarasikan atribut tanggal bertipe `LocalDate`.

```
import java.time.LocalDate;

public class Konsultasi {
    private LocalDate tanggal;
    private Pegawai dokter;
    private Pegawai perawat;
```

- m. Sediakan setter dan getter untuk masing-masing atribut pada class `Konsultasi`

```
public LocalDate getTanggal() {
    return this.tanggal;
}

public void setTanggal(LocalDate tanggal) {
    this.tanggal = tanggal;
}

public Pegawai getDokter() {
    return this.dokter;
}

public void setDokter(Pegawai dokter) {
    this.dokter = dokter;
}

public Pegawai getPerawat() {
    return this.perawat;
}

public void setPerawat(Pegawai perawat) {
    this.perawat = perawat;
}
```

- n. Implementasikan method `getInfo()` sebagai berikut:

```
public String getInfo () {
    String info = "";
    info += "No Rekam Medis          : " + this.noRekamMedis + "\n";
    info += "Nama                      : " + this.nama + "\n";
    info += "\n";

    return info;
}
```

- o. `riwayatKonsultasi` pada class `Pasien` dengan tipe `arrayList<Konsultasi>`. Atribut ini akan menyimpan serangkaian objek bertipe `Konsultasi`. Import `java.util.ArrayList` agar dapat mendeklarasikan atribut bertipe `ArrayList` of object.

```
private String noRekamMedis;
private String nama;
private ArrayList <Konsultasi> riwayatKonsultasi;
```

- p. Lakukan inisialisasi `arrayList` `riwayatKonsultasi` pada constructor class

```

public Pasien(String nama, String noRekamMedis) {
    this.noRekamMedis = noRekamMedis;
    this.nama = nama;
    this.riwayatKonsultasi = new ArrayList<>();
}

```

- q. Lakukan import `java.time.LocalDate` agar dapat mendeklarasikan atribut tanggal bertipe `LocalDate` pada class `Pasien`. Selanjutnya, implementasikan method

```

public void tambahKonsultasi(LocalDate tanggal, Pegawai dokter, Pegawai
perawat){
    Konsultasi konsultasi = new Konsultasi();
    konsultasi.setTanggal(tanggal);
    konsultasi.setDokter(dokter);
    konsultasi.setPerawat(perawat);
    riwayatKonsultasi.add(konsultasi);
}

```

- r. Modifikasi method `getInfo()` untuk mengembalikan daftar konsultasi yang pernah dilakukan oleh pasien

```

public String getInfo () {
    String info = "";
    info += "No Rekam Medis          : " + this.noRekamMedis + "\n";
    info += "Nama                    : " + this.nama + "\n";

    if (!riwayatKonsultasi.isEmpty() ) {
        info += "Riwayat Konsultasi : \n";

        for (Konsultasi konsultasi : riwayatKonsultasi){
            info += konsultasi.getInfo();
        }
    }

    else {
        info += "Belum ada riwayat konsultasi";
    }

    info += "\n";

    return info;
}

```

- s. Lakukan import `java.time.LocalDate` agar dapat mendeklarasikan atribut tanggal bertipe `LocalDate` pada class `RumahSakitDemo`. Test program yang sudah dibuat dengan

membuat objek-objek pada class RumahSakitDemo. Instansiasi objek baru bertipe Pegawai dengan nama ani menggunakan constructor Pegawai(String nip, String nama) dengan nilai argumen nip "1234" dan nama "dr. Ani". Lanjutkan instansiasi objek sebagai berikut:

```
package pemrogramanberbasisobjek.pertemuan4.rumahsakit;

import java.time.LocalDate;

public class RumahSakitDemo {
    public static void main(String[] args) {
        Pegawai ani = new Pegawai("1234", "dr. Ani");
        Pegawai bagus = new Pegawai("4567", "dr. Bagus");

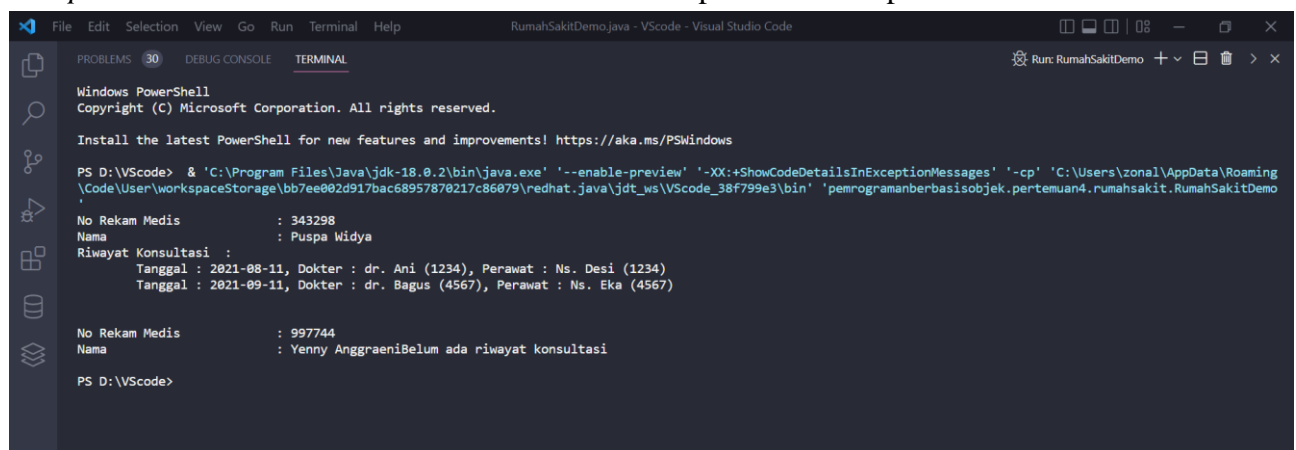
        Pegawai desi = new Pegawai("1234", "Ns. Desi");
        Pegawai eka = new Pegawai("4567", "Ns. Eka");

        Pasien pasien1 = new Pasien("Puspa Widya", "343298");
        pasien1.tambahKonsultasi(LocalDate.of(2021, 8, 11), ani, desi);
        pasien1.tambahKonsultasi(LocalDate.of(2021, 9, 11), bagus, eka);

        System.out.println(pasien1.getInfo());

        Pasien pasien2 = new Pasien("Yenny Anggraeni", "997744");
        System.out.println(pasien2.getInfo());
    }
}
```

- t. Compile kemudian run RumahSakitDemo dan didapatkan hasil seperti berikut:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\VSCode> & 'C:\Program Files\Java\jdk-18.0.2\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\zonal\AppData\Roaming\Code\User\workspaceStorage\bb7ee002d917bac68957870217c86079\redhat.java\jdt_ws\VScode_38f799e3\bin' 'pemrogramanberbasisobjek.pertemuan4.rumahsakit.RumahSakitDemo'

No Rekam Medis      : 343298
Nama                : Puspa Widya
Riwayat Konsultasi :
    Tanggal : 2021-08-11, Dokter : dr. Ani (1234), Perawat : Ns. Desi (1234)
    Tanggal : 2021-09-11, Dokter : dr. Bagus (4567), Perawat : Ns. Eka (4567)

No Rekam Medis      : 997744
Nama                : Yenny Anggraeni
Belum ada riwayat konsultasi

PS D:\VSCode>
```

Pertanyaan

Berdasarkan percobaan 1, jawablah pertanyaan-pertanyaan yang terkait:

1. Di dalam *class* Pegawai, Pasien, dan Konsultasi, terdapat method *setter* dan *getter* untuk masing-masing atributnya. Apakah gunanya *method setter* dan *getter* tersebut ?

Jawab :

- Method setter berguna untuk memberi nilai dari suatu atribut pada method.
- Method getter berguna untuk mengambil informasi data.

2. Di dalam *class* Konsultasi tidak secara eksplisit terdapat constructor dengan parameter. Apakah ini berarti *class* Pegawai tidak memiliki constructor?

Jawab : tidak, semua kelas memiliki konstruktor secara default, jika tidak membuat konstruktor kelas sendiri, java akan membuatnya. Namun, tidak dapat mengatur nilai awal untuk atribut objek

3. Di dalam *class* Pegawai dan *class* Pasien, terdapat konstruktor dengan parameter. Apakah tujuan dari constructor tersebut?

Jawab : agar dapat mengset/memberi nilai awal untuk setiap atribut

4. Perhatikan *class* Konsultasi, atribut mana saja yang bertipe *object*?

Jawab : atribut private Pegawai dokter dan private Pegawai perawat;

5. Perhatikan *class* Konsultasi, pada baris manakah yang menunjukkan bahwa *class*

Konsultasi memiliki relasi dengan *class* Pegawai?

Jawab : pada baris ke 7 dan 8 *class* Konsultasi, yaitu ketika mendeklarasikan variable dokter dan perawat yang bertipe data pegawai.

```
3  import java.time.LocalDate;
4
5  public class Konsultasi {
6      private LocalDate tanggal;
7      private Pegawai dokter;
8      private Pegawai perawat;
```


6. Perhatikan pada *class* Pasien, apa yang dilakukan pada baris `konsultasi.getInfo()` ?

Jawab : `konsultasi.getInfo()` yang dilakukan adalah mengakses method `getInfo()` dari class *Konsultasi* untuk mendapatkan info data dari method tersebut.

7. Pada method `getInfo()` class *Pasien*, terdapat baris kode: `if (!riwayatKonsultasi.isEmpty())`

Apakah yang dilakukan oleh baris tersebut?

Jawab : `if (!riwayatKonsultasi.isEmpty())` yang dilakukan adalah mengecek kondisi, jika Riwayat konsultasi tidak kosong maka method `getInfo()` class *Pasien* akan menampilkan info dari data pasien. Namun jika kosong akan mengeksekusi pada bagian *else* nya, yaitu "Belum ada riwayat konsultasi" .

8. Pada constructor class *Pasien*, terdapat baris kode:

```
this.riwayatKonsultasi = new ArrayList<>();
```

Apakah yang dilakukan oleh baris tersebut? Apakah yang terjadi jika baris tersebut dihilangkan?

Jawab :

```
this.riwayatKonsultasi = new ArrayList<>();
```

Pada baris kode program di atas yang dilakukan adalah mendeklarasikan atribut `riwayatKonsultasi` kemudian menginstansiasi agar dapat mengakses atributnya dan melakukan penyimpanan data karena riwayatKonsultasi dibuat kelas array list. Jadi, ketika baris program tersebut dihilangkan maka method tidak dapat menyimpan data riwayat konsultasi dari pasien.