Jobsheet 04 ‑ Relasi Kelas

# Kompetensi

Setelah menempuh pokok bahasan ini, mahasiswa mampu:

* 1. Memahami konsep relasi kelas;
  2. Mengimplementasikan relasi asosiasi ke dalam program.

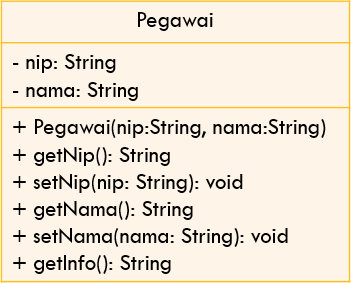
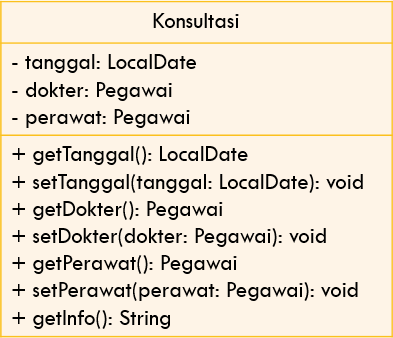
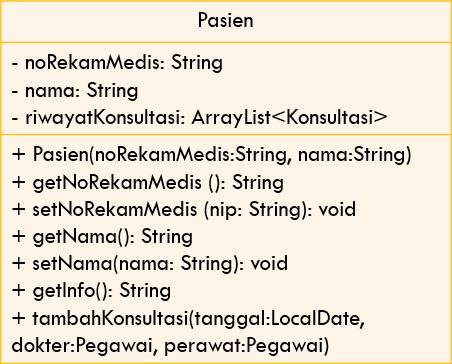
# Pendahuluan

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.

# Praktikum

Pada praktikum ini akan dikembangkan suatu sistem informasi rumah sakit yang menyimpan data riwayat konsultasi pasien.

Perhatikan diagram *class* berikut:



1

0..\*

0..\*

0..\*

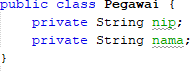
merawat

memeriksa

1

1

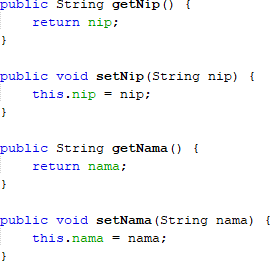
1. Buatlah folder baru dengan nama RumahSakit
2. Buatlah class Pegawai. Tambahkan atribut nip dan nama pada class Pegawai dengan akses modifier private



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

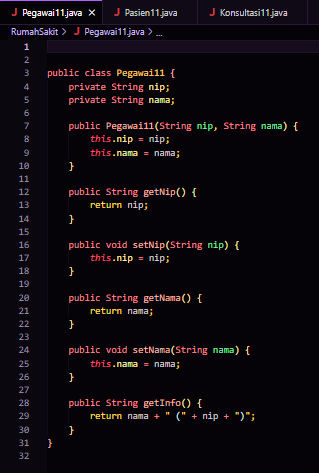


1. Implementasikan **setter** dan **getter** untuk class Pegawai.

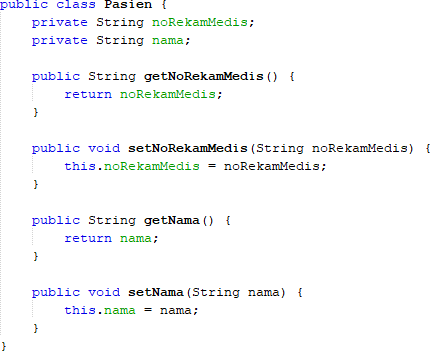


1. Implementasikan *method* getInfo()sebagai berikut:

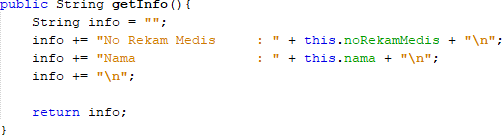


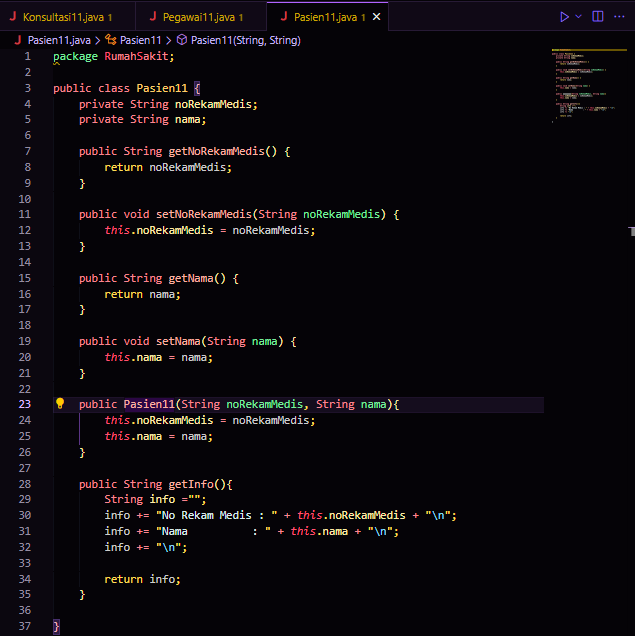


1. Selanjutnya buatlah class Pasien kemudian tambahkan atribut noRekamMedis dan nama pada class Pasien dengan access level modifier private. Sediakan pula setter dan getter untuk kedua atribut tersebut.
2. Buatlah constructor untuk class Pasien dengan parameter noRekamMedis dan nama

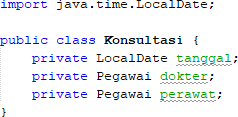


1. Implementasikan *method* getInfo()sebagai berikut:

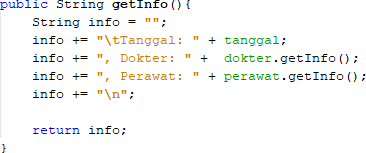
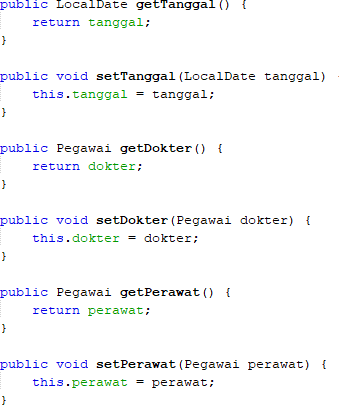




1. Sistem ini akan menyimpan data setiap konsultasi yang dilakukan pasien. Pasien bisa melakukan konsultasi lebih dari sekali. Oleh karena itu, data konsultasi akan disimpan dalam bentuk ArrayList dari objek-objek yang bertipe Konsultasi.
2. Buatlah class dengan nama Konsultasi dengan atribut tanggal bertipe LocalDate, dokter bertipe Pegawai, dan perawat bertipe Pegawai. Set access level modifier private untuk seluruh atribut. Lakukan import java.time.LocalDate agar dapat mendeklarasikan atribut tanggal bertipe LocalDate.
3. Sediakan setter dan getter untuk masing-masing atribut pada class Konsultasi

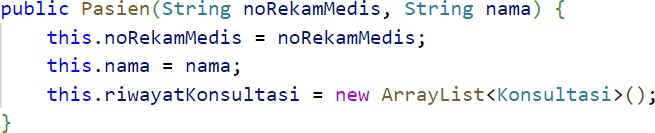


1. Implementasikan method getInfo() sebagai berikut:

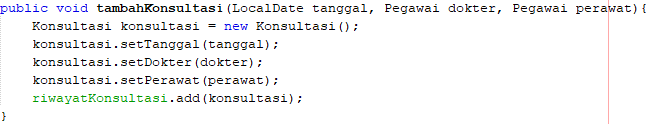




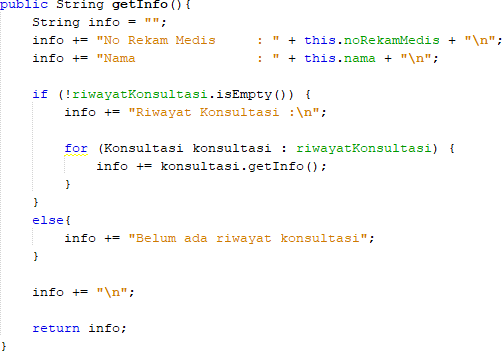
1. Untuk menyimpan data riwayat konsultasi pasien, maka tambahkan atribut riwayatKonsultasi pada class Pasien dengan tipe arrayList<Konsultasi>. Atribut ini akan menyimpan serangkaian objek bertipe Konsultasi. Import java.util.ArrayList agar dapat mendeklarasikan atribut bertipe ArrayList of object.
2. Buatlah constructor berparameter untuk class Pasien. Inisiasi nilai atribut noRekamMedis dan nama berdasarkan atribut nama. Instansiasi/buat ArrayList baru untuk atribut riwayatKonsultasi;

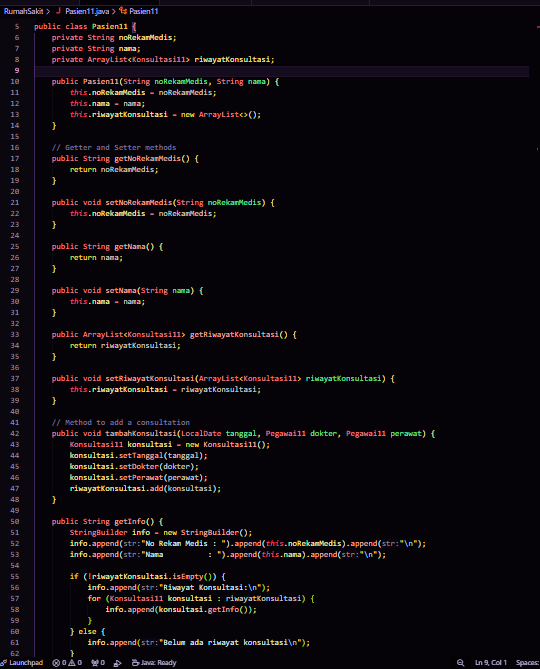


1. Lakukan import java.time.LocalDate agar dapat mendeklarasikan atribut tanggal bertipe LocalDate pada class Pasien. Selanjutnya, implementasikan method tambahKonsultasi() sebagai berikut:

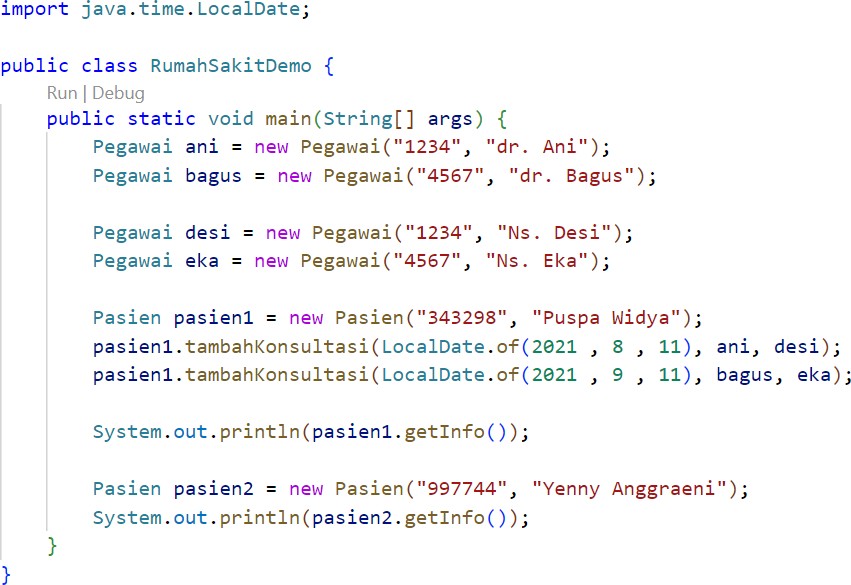


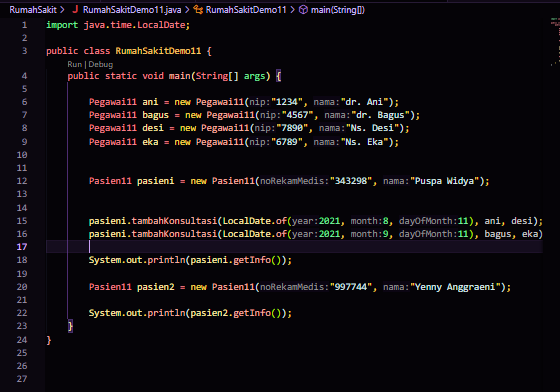
1. Modifikasi method getInfo() untuk mengembalikan info pasien dan daftar konsultasi yang pernah dilakukan



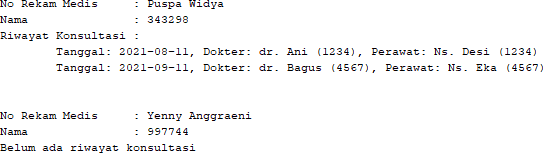


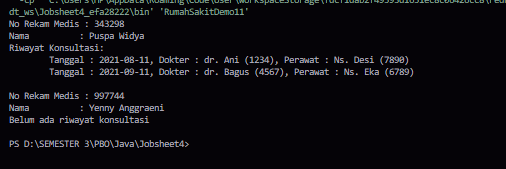
1. Lakukan import java.time.LocalDate agar dapat mendeklarasikan atribut tanggal bertipe LocalDate pada class RumahSakitDemo. Test program yang sudah dibuat dengan membuat objek-objek pada class RumahSakitDemo. Instansiasi objek baru bertipe Pegawai dengan nama ani menggunakan constructor Pegawai(String nip, String nama) dengan nilai argumen nip “1234” dan nama “dr. Ani”. Lanjutkan instansiasi objek sebagai berikut:





1. *Compile* kemudian *run* RumahSakitDemo dan didapatkan hasil seperti berikut:





# 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 ?

* **Kelas Pegawai:**

**Getter: Untuk mengambil nilai nip dan nama pegawai.**

**Setter: Untuk mengubah nilai nip dan nama pegawai, dan bisa menambahkan pemeriksaan validitas.**

* **Kelas Pasien:**

**Getter: Untuk mengambil nilai noRekamMedis dan nama pasien.**

**Setter: Untuk mengubah nilai noRekamMedis dan nama pasien, dan bisa memeriksa validitas data.**

* **Kelas Konsultasi:**

**Getter: Untuk mengambil nilai tanggal, dokter, dan perawat konsultasi.**

**Setter: Untuk mengubah nilai tanggal, dokter, dan perawat konsultasi, dan bisa menyesuaikan data jika perlu.**

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

**Ya, class Konsultasi tetap memiliki constructor, meskipun tidak didefinisikan secara eksplisit. Java secara otomatis menyediakan constructor default tanpa parameter.**

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

**Atribut yang bertipe objek :**

* **dokter**
* **perawat**

1. Perhatikan *class* Konsultasi, pada baris manakah yang menunjukan bahwa *class*

Konsultasi memiliki relasi dengan *class* Pegawai?

**Dalam class Konsultasi, relasi dengan class Pegawai terlihat pada baris :**

* + **private Pegawai dokter;**
  + **private Pegawai perawat;**
  + **Baris ini menunjukkan bahwa Konsultasi menggunakan class Pegawai untuk atribut dokter dan perawat, sehingga ada hubungan antara Konsultasi dan Pegawai.**

1. Perhatikan pada *class* Pasien, apa yang dilakukan oleh kode konsultasi.getInfo()?

**Dalam class Pasien, kode konsultasi.getInfo() digunakan untuk mengembalikan informasi detail tentang objek konsultasi, seperti tanggal, dokter, dan perawat, dalam bentuk string**

1. Pada method getInfo() dalam class Pasien, terdapat baris kode:

if (!riwayatKonsultasi.isEmpty())

Apakah yang dilakukan oleh baris tersebut?

**Baris kode if (!riwayatKonsultasi.isEmpty()) memeriksa apakah daftar riwayatKonsultasi berisi data.**

**Jika Ada Data : Kode di dalam blok if akan dijalankan, menampilkan informasi riwayat konsultasi.**

**Jika Kosong : Kode di dalam blok else akan dijalankan**, **Menyatakan bahwa belum ada riwayat konsultasi**.

1. Pada constructor class Pasien, terdapat baris kode: this.riwayatKonsultasi = new ArrayList<>();

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

**Baris kode this.riwayatKonsultasi = new ArrayList<>(); dalam constructor class Pasien melakukan :**

**Menginisialisasi Daftar: Membuat dan menetapkan daftar kosong untuk riwayatKonsultasi.**

**Jika dihilangkan:**

**riwayatKonsultasi Tidak Terinisialisasi : riwayatKonsultasi akan bernilai null, menyebabkan error saat mencoba mengaksesnya.**

# Tugas

Implementasikan studi kasus yang telah dibuat pada tugas PBO Teori ke dalam program