**Jobsheet 04 ‑ Relasi Kelas**

M. Hasan Basri

2241760139

SIB2C

# Kompetensi

Setelah menempuh pokok bahasan ini, mahasiswa mampu:

1. Memahami konsep relasi kelas;
2. Mengimplementasikan relasi asosiasi kedalam program.

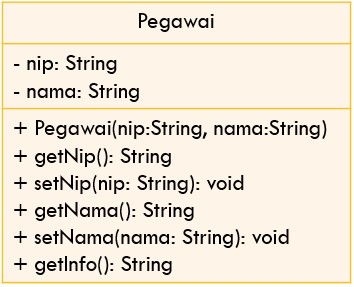
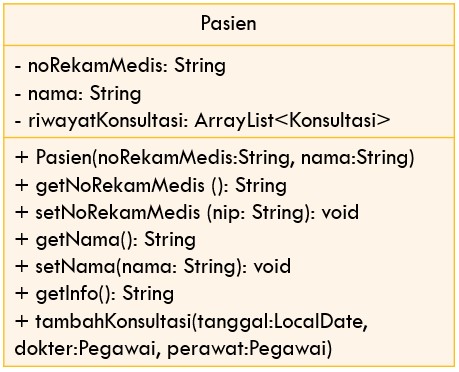
# Pendahuluan

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:



merawat

memeriksa

0..\*

0..\*

1

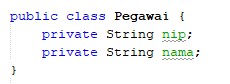
1

0..\*

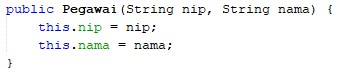
1

1. Buatlah folder baru dengan nama RumahSakit

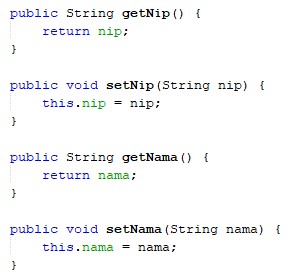
1. Buatlah class Pegawai. Tambahkan atribut nip dan nama pada class Pegawai dengan akses modifier private



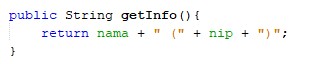
1. Buatlah *constructor* untuk class Pegawai dengan parameter nip dan nama.



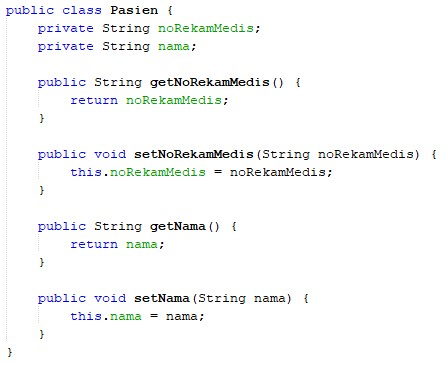
1. Implementasikan **setter** dan **getter** untuk class Pegawai.



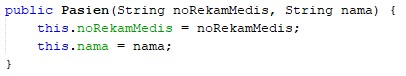
1. Implementasikan *method* getInfo()sebagai berikut:



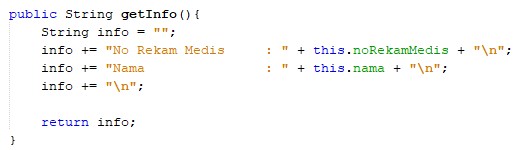
1. Selanjutnya buatlah class Pasien kemudian tambahkan atribut noRekamMedis dan nama pada class Pasien dengan access level modifier private. Sediakan pula setter dan getter untuk kedua atribut tersebut.



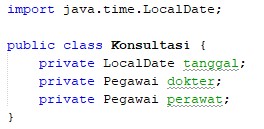
1. Buatlah constructor untuk class Pegawai dengan parameter noRekamMedis dan nama



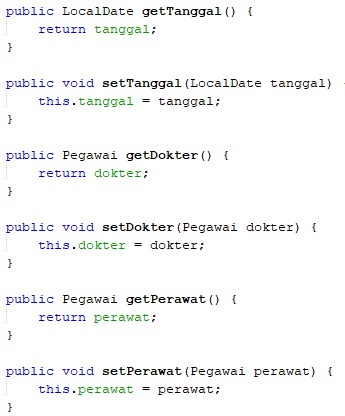
1. Implementasikan *method* getInfo()sebagai berikut:



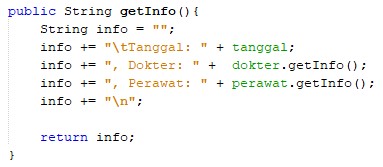
1. Sistem ini akan menyimpan data setiap konsultasi yang dilakukan pasien. Pasien bisa melakukan konsultasi lebih dari sekali. Oleh karena itu, data konsultasi akan disimpan dalam bentuk ArrayList dari objek-objek yang bertipe Konsultasi.
2. Buatlah class dengan nama Konsultasi dengan atribut tanggal 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.



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



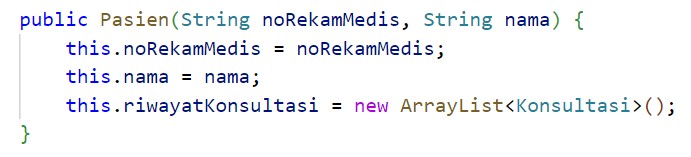
1. Implementasikan method getInfo() sebagai berikut:



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.



1. Buatlah constructor berparameter untuk class Pasien. Inisiasi nilai atribut noRekamMedis dan nama berdasarkan atribut nama. Instansiasi/buat ArrayList baru untuk atribut riwayatKonsultasi;

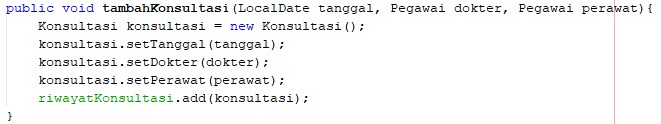


1. Lakukan import java.time.LocalDate agar dapat mendeklarasikan atribut tanggal bertipe LocalDate pada class Pasien. Selanjutnya, implementasikan method

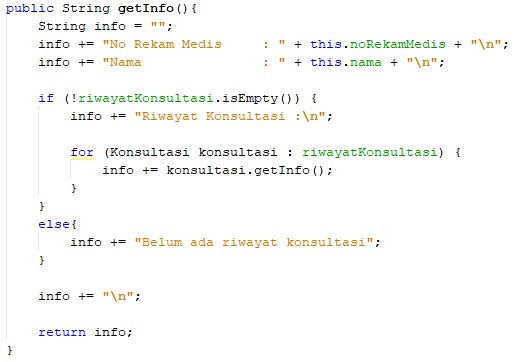
tambahKonsultasi

()

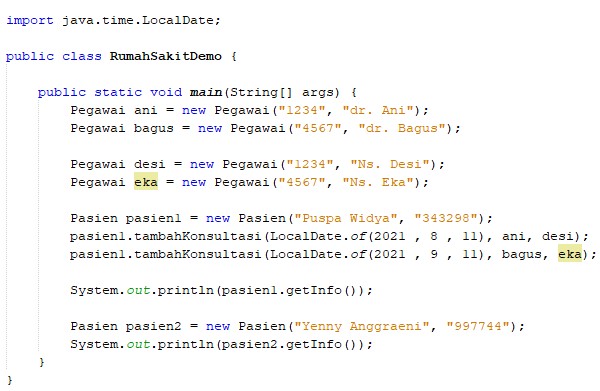
sebagai berikut:



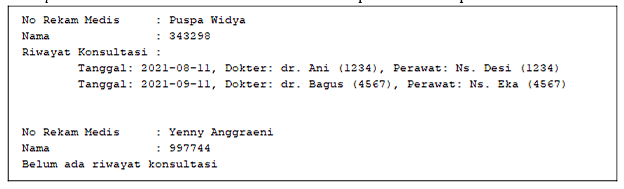
1. Modifikasi method getInfo() untuk mengembalikan info pasien dan daftar konsultasi yang pernah dilakukan



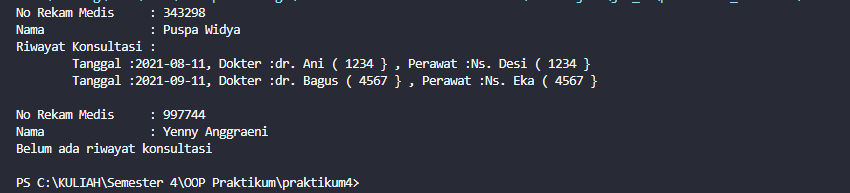
1. Lakukan import java.time.LocalDate agar dapat mendeklarasikan atribut tanggal bertipe LocalDate pada class RumahSakitDemo. Test program yang sudah dibuat dengan membuat objek-objek pada class RumahSakitDemo. Instansiasi objek baru bertipe Pegawai dengan nama ani menggunakan constructor Pegawai(String 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:



Output dari kode diatas tebalik



# 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 dan getter digunakan untuk mengatur (set) dan mengambil (get) nilai dari atribut dalam sebuah objek. Setter digunakan untuk mengubah nilai dari atribut, sedangkan getter digunakan untuk mengambil nilai dari atribut tersebut. Dengan adanya method setter dan getter, akses terhadap atribut dapat dikontrol dan dilakukan dengan cara yang aman, karena nilai atribut tidak dapat diakses atau diubah secara langsung dari luar kelas.

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

Jawab : Tidak secara eksplisit terdapat constructor dengan parameter dalam class Konsultasi. Namun, secara default Java akan menyediakan constructor tanpa parameter jika tidak ada constructor yang didefinisikan di dalam sebuah kelas. Jadi, walaupun tidak ada constructor yang didefinisikan, class Konsultasi tetap memiliki constructor.

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

Jawab : Atribut bertipe object dalam class Konsultasi adalah dokter dan perawat, yang keduanya merupakan objek dari class Pegawai.

1. Perhatikan *class* Konsultasi, pada baris manakah yang menunjukan bahwa *class* Konsultasi memiliki relasi dengan *class* Pegawai?

Jawab : Pada baris info += ", Dokter :" + dokter.getInfo(); dan info += ", Perawat :" + perawat.getInfo(); dalam method getInfo() class Konsultasi menunjukkan bahwa class Konsultasi memiliki relasi dengan class Pegawai.

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

Jawab : Pada kode konsultasi.getInfo(), ini mengambil informasi tentang konsultasi yang terjadi. Method getInfo() dalam class Konsultasi mengembalikan informasi tentang tanggal konsultasi, dokter yang melakukan konsultasi, dan perawat yang terlibat dalam konsultasi tersebut.

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

if (!riwayatKonsultasi.isEmpty())

Apakah yang dilakukan oleh baris tersebut?

Jawab : Pada baris tersebut, kita memeriksa apakah daftar riwayatKonsultasi tidak kosong. Jika tidak kosong, artinya pasien memiliki riwayat konsultasi dan informasi tentang konsultasi-konsultasi tersebut akan ditambahkan ke informasi pasien. Jika kosong, maka akan ditampilkan pesan "Belum ada riwayat konsultasi".

1. Pada constructor class Pasien, terdapat baris kode:

this.riwayatKonsultasi = new ArrayList<>();

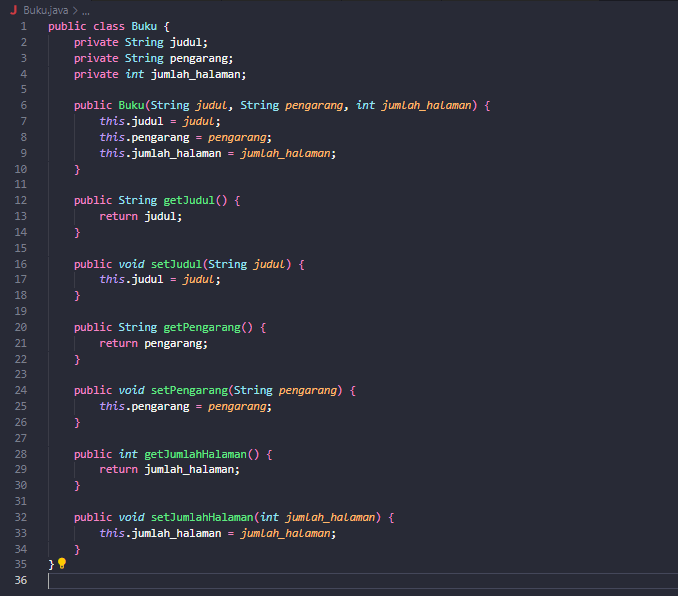
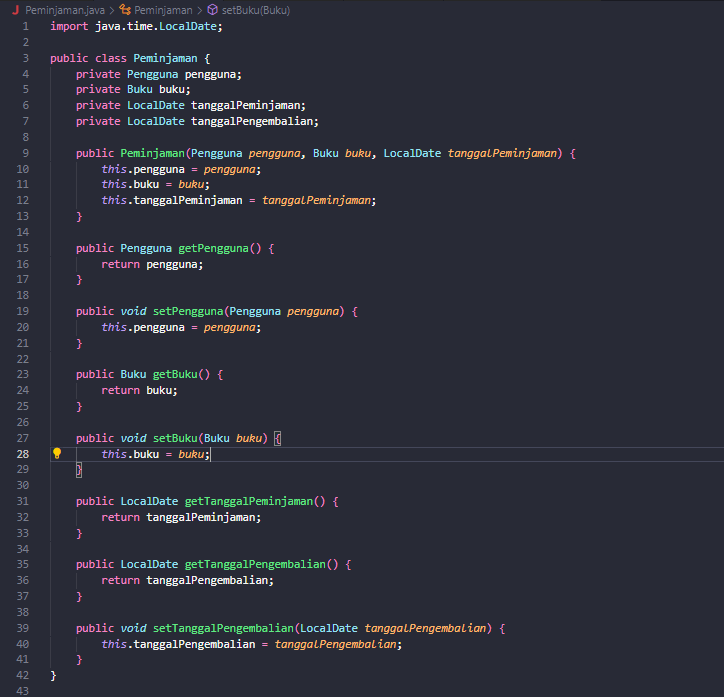
Apakah yang dilakukan oleh baris tersebut? Apakah yang terjadi jika baris tersebut dihilangkan?

Jawab : Baris this.riwayatKonsultasi = new ArrayList<>(); dalam constructor class Pasien digunakan untuk membuat objek ArrayList baru yang akan menampung riwayat konsultasi pasien tersebut. Jika baris tersebut dihilangkan, maka riwayatKonsultasi akan tetap null (belum diinisialisasi) dan akan menyebabkan NullPointerException ketika mencoba menambahkan konsultasi ke dalamnya.

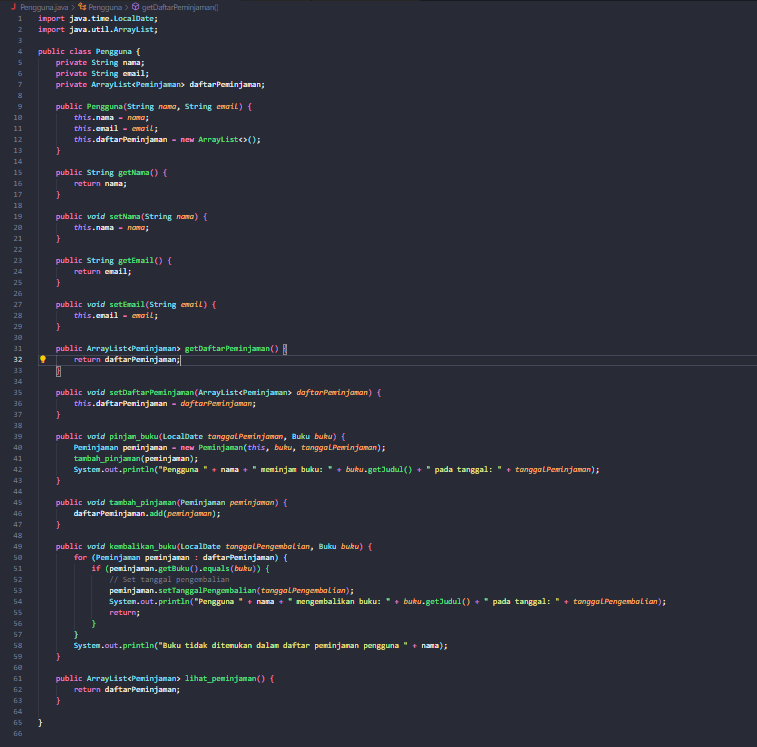
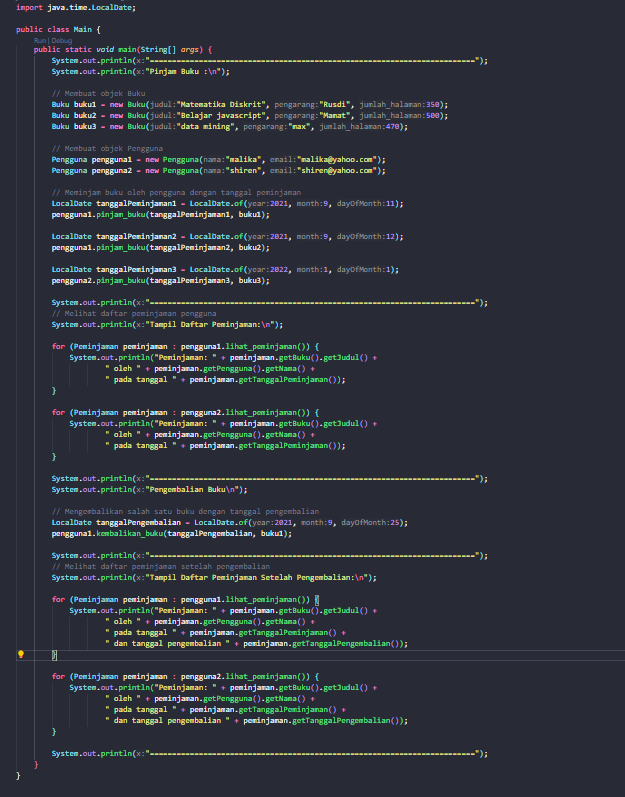
# Tugas

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

Class Buku Class Peminjaman

Class Pengguna Class Main

Output :

