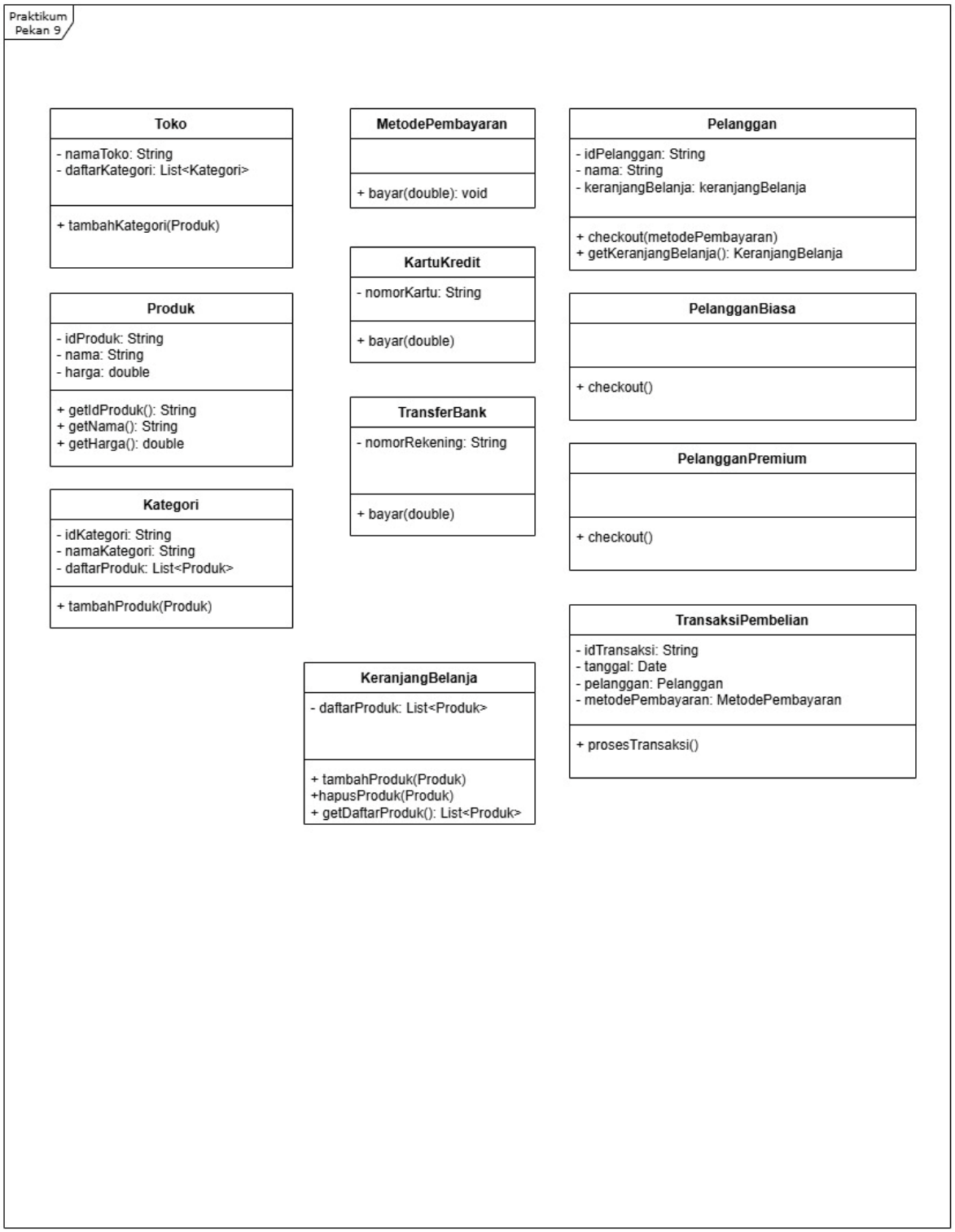
PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK MINGGU 9 (RELASI ANTAR KELAS)

Telkom University

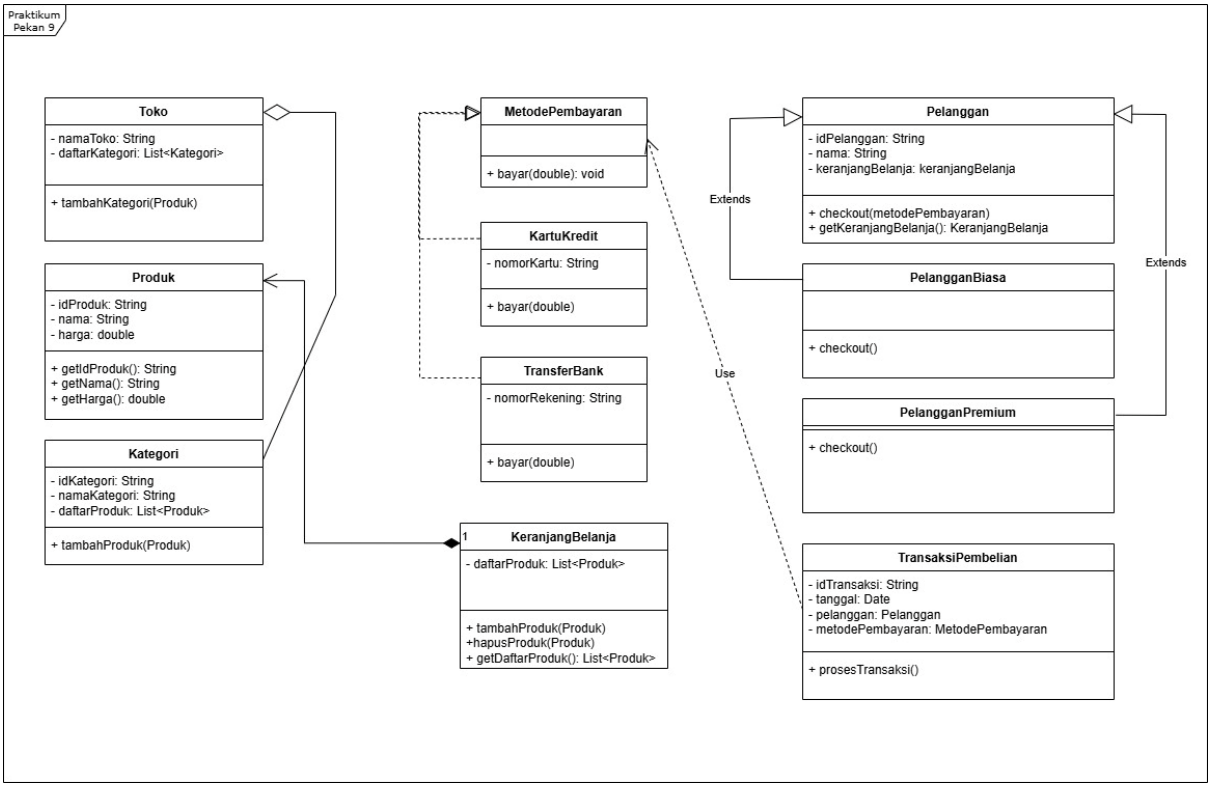
Nama	:	Muhammad Taqi Izdihar
NIM	:	607012430006
Kelas	:	D3SI-48-01
Tanggal	:	23 April 2025

Jurnal Implementasi

1. **Diagram praktikum_kelas_diagram pada situs web drawio/appdiagrams.net menggunakan UML 1:**



2. **Menghubungkan kelas-kelas menggunakan panah sesuai dengan jenis relasinya:**



3. **Implementasi program ImpRelasikelasDiagram**

```
import java.util.ArrayList;
import java.util.Date;
import java.util.List;

// Interface MetodePembayaran
interface MetodePembayaran {
        void bayar(double jumlah);
    }

    // Implementasi KartuKredit
    class KartuKredit implements MetodePembayaran {
        private String nomorKartu;

        public KartuKredit(String nomorKartu) {
            this.nomorKartu = nomorKartu;
        }

        @Override
        public void bayar(double jumlah) {
            System.out.println(" Pembayaran sebesar " + jumlah + " berhasil menggunakan kartu kredit : " + nomorKartu);
        }
    }

```

```
27.      // Implementasi TransferBank
28.      class TransferBank implements MetodePembayaran {
29.          private String nomorRekening;
30.
31.          public TransferBank(String nomorRekening) {
32.              this.nomorRekening = nomorRekening;
33.          }
34.
35.          @Override
36.          public void bayar(double jumlah) {
37.              System.out.println(
38.                      " Pembayaran sebesar " + jumlah + " berhasil
   melalui transfer bank ke rekening : " + nomorRekening);
39.          }
40.      }
41.
42.      // Kelas Produk
43.      class Produk {
44.          private String idProduk;
45.          private String nama;
46.          private double harga;
47.
48.          public Produk(String idProduk, String nama, double harga) {
49.              this.idProduk = idProduk;
50.              this.nama = nama;
51.              this.harga = harga;
52.          }
53.
54.          public String getIdProduk() {
55.              return idProduk;
56.          }
57.
58.          public String getNama() {
59.              return nama;
60.          }
61.
62.          public double getHarga() {
63.              return harga;
64.          }
65.      }
66.
67.      // Kelas Kategori
68.      class Kategori {
69.          private String idKategori;
70.          private String namaKategori;
71.          private List<Produk> daftarProduk;
72.
73.          public Kategori(String idKategori, String namaKategori) {
```

```
74.             this.idKategori = idKategori;
75.             this.namaKategori = namaKategori;
76.             this.daftarProduk = new ArrayList<>();
77.         }
78.
79.         public void tambahProduk(Produk produk) {
80.             daftarProduk.add(produk);
81.             System.out.println(" Produk '" + produk.getNama() + "'
    ditambahkan ke kategori " + namaKategori);
82.         }
83.     }
84.
85.     // Kelas Toko
86.     class Toko {
87.         private String namaToko;
88.         private List<Kategori> daftarKategori;
89.
90.         public Toko(String namaToko) {
91.             this.namaToko = namaToko;
92.             this.daftarKategori = new ArrayList<>();
93.         }
94.
95.         public void tambahKategori(Kategori kategori) {
96.             daftarKategori.add(kategori);
97.             System.out.println(" Kategori '" + kategori +
    "'ditambahkan ke toko " + namaToko);
98.         }
99.     }
100.
101.    // Kelas KeranjangBelanja
102.    class KeranjangBelanja {
103.        private List<Produk> daftarProduk;
104.
105.        public KeranjangBelanja() {
106.            this.daftarProduk = new ArrayList<>();
107.        }
108.
109.        public void tambahProduk(Produk produk) {
110.            daftarProduk.add(produk);
111.            System.out.println(" Produk '" + produk.getNama() + "'
    ditambahkan ke keranjang belanja .");
112.        }
113.
114.        public void hapusProduk(Produk produk) {
115.            if (daftarProduk.remove(produk)) {
116.                System.out.println(" Produk '" + produk.getNama() +
    "' dihapus dari keranjang belanja.");
117.            } else {
```

```
                System.out.println(" Produk tidak ditemukan dalam keranjang belanja.");
            }
        }

        public List<Produk> getDaftarProduk() {
            return daftarProduk;
        }
    }

    // Kelas Pelanggan
    abstract class Pelanggan {
        protected String idPelanggan;
        protected String nama;
        protected KeranjangBelanja keranjangBelanja;

        public Pelanggan(String idPelanggan, String nama) {
            this.idPelanggan = idPelanggan;
            this.nama = nama;
            this.keranjangBelanja = new KeranjangBelanja();
        }

        public abstract void checkout(MetodePembayaran metodePembayaran);

        public KeranjangBelanja getKeranjangBelanja() {
            return keranjangBelanja;
        }
    }

    // Subclass PelangganBiasa
    class PelangganBiasa extends Pelanggan {
        public PelangganBiasa(String idPelanggan, String nama) {
            super(idPelanggan, nama);
        }

        @Override
        public void checkout(MetodePembayaran metodePembayaran) {
            double total = 0;
            for (Produk produk : keranjangBelanja.getDaftarProduk()) {
                total += produk.getHarga();
            }
            System.out.println(" Total pembelian : " + total);
            metodePembayaran.bayar(total);
        }
    }

```

```
163.     // Subclass PelangganPremium
164.     class PelangganPremium extends Pelanggan {
165.         public PelangganPremium(String idPelanggan, String nama) {
166.             super(idPelanggan, nama);
167.         }
168.
169.         @Override
170.         public void checkout(MetodePembayaran metodePembayaran) {
171.             double total = 0;
172.             for (Produk produk : keranjangBelanja.getDaftarProduk())
    {
173.                 total += produk.getHarga();
174.             }
175.             double diskon = total * 0.1; // Diskon 10% untuk
    pelanggan premium
176.             total -= diskon;
177.             System.out.println(" Total pembelian setelah diskon: " +
    total);
178.             metodePembayaran.bayar(total);
179.         }
180.     }
181.
182.     // Kelas TransaksiPembelian
183.     class TransaksiPembelian {
184.         private String idTransaksi;
185.         private Date tanggal;
186.         private Pelanggan pelanggan;
187.         private MetodePembayaran metodePembayaran;
188.
189.         public TransaksiPembelian(String idTransaksi,
190.                 Pelanggan pelanggan, MetodePembayaran
    metodePembayaran) {
191.             this.idTransaksi = idTransaksi;
192.             this.tanggal = new Date();
193.             this.pelanggan = pelanggan;
194.             this.metodePembayaran = metodePembayaran;
195.         }
196.
197.         public void prosesTransaksi() {
198.             System.out.println(" Memproses transaksi ... ");
199.             pelanggan.checkout(metodePembayaran);
200.         }
201.     }
202.
203.     // Main Class
204.     public class ImpRelasiKelasDiagram {
205.         public static void main(String[] args) {
206.             // Inisialisasi produk
```

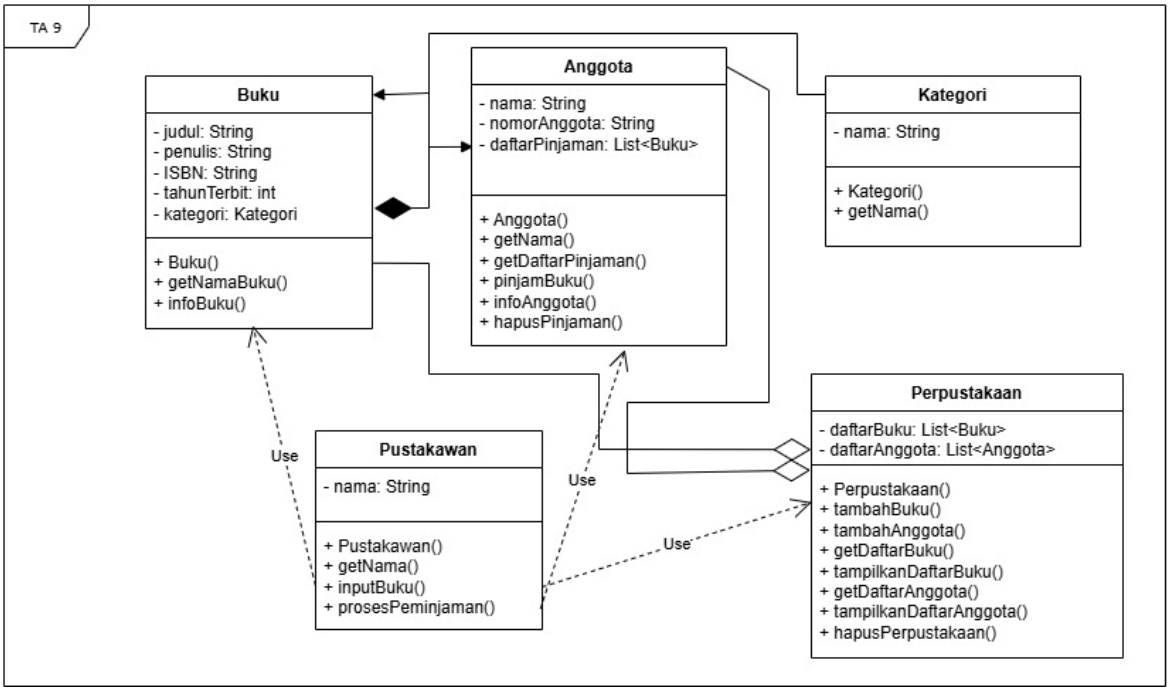
```
207.             Produk produk1 = new Produk(" P001 ", " Laptop ",
   10000000);
208.             Produk produk2 = new Produk(" P002 ", " Smartphone ",
   5000000);
209.
210.             // Inisialisasi kategori
211.             Kategori elektronik = new Kategori(" K001 ", "Elektronik
   ");
212.             elektronik.tambahProduk(produk1);
213.             elektronik.tambahProduk(produk2);
214.
215.             // Inisialisasi toko
216.             Toko toko = new Toko(" Toko Elektronik ");
217.             toko.tambahKategori(elektronik);
218.
219.             // Inisialisasi pelanggan
220.             Pelanggan pelangganBiasa = new PelangganBiasa("C001 ", "
   Alice ");
221.             Pelanggan pelangganPremium = new PelangganPremium(" C002
   ", "Bob ");
222.
223.             // Menambahkan produk ke keranjang belanja
224.             pelangganBiasa.getKeranjangBelanja().tambahProduk(produk1
   );
225.             pelangganPremium.getKeranjangBelanja().tambahProduk(produ
   k2);
226.
227.             // Proses pembayaran
228.             MetodePembayaran kartuKredit = new KartuKredit("1234 -
   5678 -9012 -3456 ");
229.             MetodePembayaran transferBank = new
   TransferBank("987654321 ");
230.
231.             TransaksiPembelian transaksi1 = new TransaksiPembelian("
   T001 ", pelangganBiasa, kartuKredit);
232.             transaksi1.prosesTransaksi();
233.
234.             TransaksiPembelian transaksi2 = new TransaksiPembelian("
   T002 ", pelangganPremium, transferBank);
235.             transaksi2.prosesTransaksi();
236.         }
237.     }
238.
```

```
Produk ? Laptop ? ditambahkan ke kategori Elektronik
Produk ? Smartphone ? ditambahkan ke kategori Elektronik
Kategori ?Kategori@1218025c'ditambahkan ke toko  Toko Elektronik
Produk ? Laptop ? ditambahkan ke keranjang belanja .
Produk ? Smartphone ? ditambahkan ke keranjang belanja .
Memproses transaksi ...
Total pembelian : 1.0E7
Pembayaran sebesar 1.0E7 berhasil menggunakan kartu kredit : 1234 -5678 -9012 -3456
Memproses transaksi ...
Total pembelian setelah diskon: 4500000.0
Pembayaran sebesar 4500000.0 berhasil melalui transfer bank ke rekening : 987654321
```

Tugas Akhir

Anda diminta untuk merancang sebuah sistem perpustakaan sederhana menggunakan Java. Sistem ini melibatkan beberapa kelas dan relasi di antara mereka. Dimana dalam sistem ini terdapat buku yang informasinya terdiri dari judul, pengarang, dan tahun terbit. Kemudian peminjam buku merupakan anggota perpustakaan yang dibuktikan dengan adanya nama, nomor Anggota, daftar Pinjaman (daftar Pinjaman merupakan list buku yang dipinjam). Selain itu, terdapat pustakawan yang memiliki tugas melakukan input buku dan melakukan proses peminjaman. Pustakawan akan selalu bergantung pada sistem untuk memproses peminjaman. Selanjutnya setiap buku akan dikelompokan bedasarkan kategorinya. Tidak ada buku yang tidak memiliki kategori. Buku dan anggota harus (wajib) memiliki fungsi yang dapat mencetak informasinya. Yang terakhir, terdapat kelas Perpustakaan dimana jika perpustakaan dihapus maka buku dan anggota juga akan terhapus. Tugas anda adalah membuat kelas diagram beserta hubungan antar kelasnya, dan juga membuat kode program sesuai dengan kelas diagram yang anda kerjakan.

Diagram:



Penjelasan tipe relasi antar kelas:

1) Perpustakaan – Buku (Agregasi)
   Kelas Perpustakaan menggunakan List<Buku> untuk daftar buku. Meskipun begitu, keduanya masih bisa berdiri sendiri atau independen sebagaimana pada awal Kelas Main di-*run*, belum terdapat buku yang memiliki nilai atau sama dengan null.

2) Perpustakaan – Anggota (Agregasi)
   Seperti Buku, Kelas Perpustakaan juga menyimpan List<Anggota> di dalamnya untuk daftar anggota di dalam perpustakaan.
3) Anggota – Buku (Agregasi)
   Kelas Anggota juga menyimpan List<Buku> untuk menyimpan daftar buku yang dimiliki oleh anggota yang bisa bertambah.
4) Pustakawan – Buku & Anggota & Perpustakaan (Dependensi)
   Kelas Pustakawan bergantung pada kelas Perpustakaan untuk menambahkan buku serta memproses peminjaman buku oleh anggota.
5) Buku – Kategori (Komposisi)
   Setiap buku harus memiliki satu kategori, hal ini menunjukkan bahwa siklus hidup kelas yang dimiliki ditentukan oleh kelas yang memilikinya.

Kode Blok:

1. **Kelas Buku**

```
2. public class Buku {
3.     private String judul; //field untuk judul buku
4.     private String penulis; //field untuk penulis buku
5.     private String ISBN; //field untuk ISBN buku
6.     private int tahunTerbit; //field untuk tahun terbit buku
7.     private Kategori kategori; //field untuk kategori buku yang diambil
   dari kelas Kategori
8.
9.     public Buku(String judul, String penulis, String ISBN, int
   tahunTerbit, Kategori kategori) { ////Konstruktor Buku
10.        this.judul = judul;
11.        this.penulis = penulis;
12.        this.ISBN = ISBN;
13.        this.tahunTerbit = tahunTerbit;
14.        this.kategori = kategori;
15.    }
16.
17.    public String getNamaBuku() { //Getter untuk memanggil judul buku
18.        return judul;
19.    }
20.
21.    public void infoBuku() { //Method untuk menampilkan informasi buku
   karena kelas Buku harus menampilkan informasi dari setiap buku yang ada
22.        System.out.println("Judul Buku: " + judul);
23.        System.out.println("Penulis Buku: " + penulis);
24.        System.out.println("Tahun Terbit: " + tahunTerbit);
25.        System.out.println("ISBN: " + ISBN);
26.        System.out.println("Kategori Buku: " + kategori.getNama());
27.    }
28.}
```

2. Kelas Kategori

```
public class Kategori {
    private String nama; //field untuk nama kategori

    public Kategori(String nama) { //konstruktor kategori
        this.nama = nama;
    }

    public String getNama() { //Getter untuk mendapatkan nilai nama Kategori
        return nama;
    }
}
```

3. Kelas Anggota

```
import java.util.ArrayList;
import java.util.List;

public class Anggota {
    private String nama; //field untuk nama anggota
    private String nomorAnggota; //field untuk nomor anggota
    private List<Buku> daftarPinjaman; //field untuk daftar buku yang dipinjam oleh anggota berupa List<> dari kelas Buku

    public Anggota(String nama, String nomorAnggota) { //Konstruktor Anggota
        this.nama = nama;
        this.nomorAnggota = nomorAnggota;
        this.daftarPinjaman = new ArrayList<>();
    }

    public String getNama() { //Getter untuk memanggil nama anggota
        return nama;
    }

    public List<Buku> getDaftarPinjaman() { //Getter untuk memanggil daftar pinjaman buku yang dimiliki oleh anggota
        return daftarPinjaman;
    }

    public void pinjamBuku(Buku buku) { //Method untuk menambahkan buku ke dalam daftar pinjaman buku
        daftarPinjaman.add(buku);
    }

```

```
30.     public void infoAnggota() { //Method untuk menampilkan informasi
    anggota karena kelas Anggota harus menampilkan informasi dari setiap
    pengguna
31.         System.out.println("Nama Anggota: " + nama);
32.         System.out.println("Nomor Anggota: " + nomorAnggota);
33.         System.out.println("Daftar Buku yang Dipinjam Anggota:");
34.         System.out.println("==================================");
35.         for (Buku dafbuk : daftarPinjaman) {
36.             dafbuk.infoBuku();
37.             System.out.println(" ");
38.         }
39.     }
40.
41.     public void hapusPinjaman() { //Method untuk menghapus seluruh buku
    yang dipinjam oleh anggota
42.         daftarPinjaman.clear();
43.     }
44. }
```

4. Kelas Pustakawan

```
3. public class Pustakawan {
4.     private String nama; //field untuk nama pustakawan
5.
6.     public Pustakawan(String nama) { //Konstruktor Pustakawan
7.         this.nama = nama;
8.     }
9.
10.    public String getNama() { //Getter untuk memanggil nama pustakawan
11.        return nama;
12.    }
13.
14.    public void inputBuku(Perpustakaan perpustakaan, Buku buku) {
    //Method untuk menambahkan buku ke dalam perpustakaan sebagai seorang
    pustakawan
15.        perpustakaan.tambahBuku(buku);
16.    }
17.
18.    public void prosesPeminjaman(Perpustakaan perpustakaan, Anggota
    anggota, Buku buku) { //Method untuk menampilkan proses peminjaman buku
    oleh anggota
19.        if (anggota.getDaftarPinjaman().contains(buku)) {
20.            anggota.infoAnggota();
21.        } else {
22.            System.out.println("Tidak ada proses peminjaman.");
23.        }
24.    }
25. }
```

5. Kelas Perpustakaan

```
import java.util.ArrayList;
import java.util.List;

public class Perpustakaan {
    private List<Buku> daftarBuku; //field untuk daftar buku dalam perpustakaan berupa List<> dari kelas Buku
    private List<Anggota> daftarAnggota; //field untuk daftar anggota dalam perpustakaan berupa List<> dari kelas Anggota

    public Perpustakaan() { //Konstruktor Perpustakaan
        daftarBuku = new ArrayList<>();
        daftarAnggota = new ArrayList<>();
    }

    public void tambahBuku(Buku buku) { //Method untuk menambahkan buku ke dalam perpustakaan yang digunakan oleh pustakawan
        daftarBuku.add(buku);
    }

    public void tambahAnggota(Anggota anggota) { //Method untuk menambahkan anggota ke dalam perpustakaan
        daftarAnggota.add(anggota);
    }

    public List<Buku> getDaftarBuku() { //Getter untuk memanggil daftar buku di perpustakaan
        return daftarBuku;
    }

    public void tampilkanDaftarBuku() { //Method untuk menampilkan daftar buku yang ada di perpustakaan
        for (Buku buku : daftarBuku) {
            buku.infoBuku();
            System.out.println();
        }
    }

    public List<Anggota> getDaftarAnggota() { //Getter untuk memanggil daftar anggota di perpustakaan
        return daftarAnggota;
    }

    public void tampilkanDaftarAnggota() { //Method untuk menampilkan daftar anggota yang ada di perpustakaan
        for (Anggota anggota : daftarAnggota) {
            anggota.infoAnggota();
            System.out.println();
```

```
        }
    }

    public void hapusPerpustakaan() { //Method untuk menghapus perpustakaan
        for (Anggota anggota : daftarAnggota) {
            anggota.hapusPinjaman();
        }

        daftarBuku.clear();
        daftarAnggota.clear();
        System.out.println("Perpustakaan telah dihapus beserta semua buku dan anggota.");
        System.out.println("Tekan Enter untuk kembali");
    }
}
```

5. Kelas Main

```
import java.util.Scanner;

public class Main { //Kelas Main
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in); //Kelas Scanner sehingga pengguna bisa melakukan input
        Clear clear = new Clear(); //Kelas Clear sehingga bisa memasuki "antarmuka" baru
        Pustakawan pustakawan = new Pustakawan("Alnilam Lambda"); //Pustakawan yang ada untuk mengelola perpustakaan
        Anggota anggota = new Anggota("Mintaka Maveen", "A512"); //Anggota awal yang ada di perpustakaan
        Buku buku = new Buku(null, null, null, 0, null); //Buku yang ada di perpustakaan tetapi masih bernilai kosong dan bisa ditambahkan oleh pustakawan sesuai dengan tugasnya
        Perpustakaan perpustakaan = new Perpustakaan(); //Kelas Perpustakaan
        perpustakaan.tambahAnggota(anggota); //Menambahkan anggota ke dalam perpustakaan


        while (true) { //Menggunakan perulangan while untuk melakukan pilihan opsi menu kepada pengguna sampai mereka ingin berhenti
            clear.clearScreen(); //Memasuki "antarmuka" baru
            System.out.println("Pilih peran Anda:"); //Opsi untuk memilih peran
            System.out.println("1. Pustakawan");
            System.out.println("2. Anggota");
            System.out.println("3. Keluar");
            System.out.print("Pilihan: ");
```

```
            int pilihan = scanner.nextInt();
            scanner.nextLine();

            if (pilihan == 1) { //Percabangan if-else, jika yang dipilih adalah peran sebagai seorang pustakawan
                clear.clearScreen();
                System.out.println("<=== Selamat datang Master " + pustakawan.getNama() + " ===>"); //Nama pustakawan
                System.out.println();
                System.out.println("Menu Pustakawan:"); //Daftar pilihan untuk menu pustakawan
                System.out.println("1. Daftar Peminjaman Buku");
                System.out.println("2. Tambah Buku");
                System.out.println("3. Tambah Anggota");
                System.out.println("4. Tampilkan Daftar & Informasi Anggota");
                System.out.println("5. Tampilkan Daftar Buku");
                System.out.println("6. Hapus Perpustakaan");
                System.out.println("7. Kembali");
                System.out.println();
                System.out.print("Pilihan: ");
                int pilihPustakawan = scanner.nextInt();
                scanner.nextLine();

                switch (pilihPustakawan) { //Percabangan switch untuk setiap aksi pustakawan yang dipilih pada menu pustakawan
                    case 1: //Aksi untuk menampilkan proses peminjaman buku oleh pengguna yang akan selalu kosong jika pengguna tidak meminjam (semua) buku di perpustakaan
                        clear.clearScreen();
                        pustakawan.prosesPeminjaman(perpustakaan, anggota, buku); //Menggunakan kelas pustakawan pada method prosesPeminjaman
                        System.out.print("Tekan Enter untuk kembali");
                        scanner.nextLine();
                    break;
                    case 2: //Aksi untuk menambahkan buku secara manual sebagai seorang pustakawan
                        clear.clearScreen();
                        System.out.print("Masukkan Judul: ");
                        String judulBuku = scanner.nextLine();
                        System.out.print("Masukkan Nama Penulis: ");
                        String namaPenulis = scanner.nextLine();
                        System.out.print("Masukkan ISBN: ");
                        String noISBN = scanner.nextLine();
                        System.out.print("Masukkan Tahun Terbit: ");
                        int tahunTerbitBuku = scanner.nextInt();
                        System.out.print("Masukkan Kategori: ");
```

```
63.                         scanner.nextLine();
64.                         String kategoriBuku = scanner.nextLine();
65.                         Kategori kategoriBaru = new
   Kategori(kategoriBuku);
66.                         System.out.print("1 Buku berhasil ditambahkan!
   Tekan enter untuk melanjutkan");
67.                         scanner.nextLine();
68.                         Buku bukuBaru = new Buku(judulBuku,
   namaPenulis, noISBN, tahunTerbitBuku, kategoriBaru); //Membuat objek
   dari kelas Buku untuk menerima input nilai dari variabel lokal dalam
   parameternya
69.                         buku = bukuBaru; //menyamakan objek buku dengan
   bukuBaru
70.                         pustakawan.inputBuku(perpustakaan, bukuBaru);
   //Menambahkan buku ke dalam perpustakaan sebagai seorang pustakawan
   menggunakan kelas Pustakawan dengan method inputBuku
71.                         System.out.println();
72.                     break;
73.                     case 3: //Aksi untuk menambahkan anggota lain ke
   dalam perpustakaan sebagai seorang pustakawan
74.                         clear.clearScreen();
75.                         System.out.print("Masukkan nama anggota: ");
76.                         String namaAnggota = scanner.nextLine();
77.                         System.out.print("Masukkan nomor anggota: ");
78.                         String nomorAnggota = scanner.nextLine();
79.                         Anggota anggotaBaru = new Anggota(namaAnggota,
   nomorAnggota); //Membuat objek dari kelas Anggota untuk menambahkan
   anggota lain (selain Mintaka Maveen) ke dalam perpustakaan
80.                         perpustakaan.tambahAnggota(anggotaBaru);
   //Menambahkan anggota lain ke dalam perpustakaan sebagai seorang
   pustakawan menggunakan kelas Perpustakaan dengan method tambahAnggota
81.                         System.out.println("Anggota berhasil
   ditambahkan! Tekan enter untuk melanjutkan");
82.                         scanner.nextLine();
83.                     break;
84.                     case 4: //Aksi untuk menampilkan informasi seluruh
   anggota yang ada di dalam perpustakaan
85.                         clear.clearScreen();
86.                         System.out.println("<=== Informasi Anggota
   dalam Perpustakaan ===>");
87.                         System.out.println();
88.                         perpustakaan.tampilkanDaftarAnggota();
   //Menampilkan daftar anggota dalam perpustakaan menggunakan method
   tampilkanDaftarAnggota
89.                         System.out.print("Tekan Enter untuk kembali");
90.                         scanner.nextLine();
91.                     break;
```

```
92.                     case 5: //Aksi untuk menampilkan informasi seluruh
   buku yang ada di dalam perpustakaan
93.                         clear.clearScreen();
94.                         System.out.println("<=== Informasi Buku dalam
   Perpustakaan ===>");
95.                         System.out.println();
96.                         perpustakaan.tampilkanDaftarBuku();
   //Menampilkan daftar buku dalam perpustakaan menggunakan method
   tampilkanDaftarBuku
97.                         System.out.print("Tekan Enter untuk kembali");
98.                         scanner.nextLine();
99.                     break;
100.                        case 6: //Aksi untuk menghapus perpustakaan
101.                            clear.clearScreen();
102.                            System.out.println("PERHATIAN");
103.                            System.out.println("Apakah Anda yakin
   ingin menghapus perpustakaan?");
104.                            System.out.print("Anda akan ikut terhapus
   juga! [Y/N]: ");
105.                            String pilihanHapus = scanner.nextLine();
106.                            if (pilihanHapus == "Y") {
107.                                perpustakaan.hapusPerpustakaan();
   //Menghapus perpustakaan menggunakan method hapusPerpustakaan dalam
   kelas Perpustakaan
108.                                anggota = null;
109.                                buku = null;
110.                                scanner.nextLine();
111.                            }
112.                            break;
113.                        case 7: //Aksi untuk kembali ke menu utama
114.                            break;
115.                        default:
116.                            System.out.println("Pilihan tidak
   valid.");
117.                    }
118.                } else if (pilihan == 2) { //Pilihan untuk mengambil
   peran sebagai seorang anggota
119.                    clear.clearScreen();
120.                    System.out.println("<=== Selamat Datang " +
   anggota.getNama() + " ===>"); //Menu untuk seorang anggota
121.                    System.out.println();
122.                    System.out.println("Menu Anggota:");
123.                    System.out.println("1. Informasi Diri");
124.                    System.out.println("2. Pinjam Buku");
125.                    System.out.println("3. Kembali");
126.                    System.out.println();
127.                    System.out.print("Pilihan: ");
128.                    int pilihAnggota = scanner.nextInt();
```

```
129.                     scanner.nextLine();
130.
131.                     switch (pilihAnggota) {
132.                         case 1: //Aksi untuk melihat informasi
    anggota sebagai anggota itu sendiri
133.                             clear.clearScreen();
134.                             System.out.println("Informasi Anda");
135.                             anggota.infoAnggota(); //Menampilkan
    informasi anggota atau diri sendiri (sebagai Mintaka Maveen) melalui
    method infoAnggota
136.                             System.out.println();
137.                             System.out.print("Tekan Enter untuk
    kembali");
138.                             scanner.nextLine();
139.                             break;
140.                         case 2: //Aksi untuk meminjam buku dengan
    melihat daftar buku yang tersedia di perpustakaan yang sudah
    ditambahkan oleh pustakawan
141.                             clear.clearScreen();
142.                             System.out.println("<=== Buku tersedia
    ===>");
143.                             System.out.println();
144.                             perpustakaan.tampilkanDaftarBuku();
    //Menampilkan daftar buku yang tersedia untuk dipinjam dari
    perpustakaan yang sebelumnya sudah ditambahkan oleh pustakawan
145.                             System.out.println();
146.                             System.out.println("==================
    ==");
147.                             System.out.println();
148.                             System.out.println("Buku mana yang ingin
    Anda pinjam?");
149.                             System.out.print("Tuliskan nama buku: ");
150.                             String pilihanNamaBuku =
    scanner.nextLine();
151.                             boolean cariBuku = false; //Variabel
    untuk mengecek jika buku tersedia atau tidak
152.                             for (Buku cekBuku :
    perpustakaan.getDaftarBuku()) { //Perulangan foreach pada daftarBuku
    dalam kelas Perpustakaan untuk mengecek apakah buku yang dimaksud
    tersedia atau tidak
153.                                 if
    (cekBuku.getNamaBuku().equalsIgnoreCase(pilihanNamaBuku)) {
    //Percabangan if jika buku yang dimaksud ada di dalam perpustakaan,
    menggunakan getter getNamaBuku untuk melihat nama buku dan
    equalsIgnoreCase untuk menghindari kesalahan besar-kecil kata
154.                                     anggota.pinjamBuku(cekBuku);
    //Menambahkan buku ke dalam daftar pinjaman buku oleh anggota
```

```
                                    cariBuku = true; //Mengubah nilai variabel pengecek menjadi true atau buku tersedia
                                    System.out.println("Buku berhasil dipinjam!");
                                    System.out.print("Tekan Enter untuk kembali");
                                    scanner.nextLine();
                                    break;
                                }
                            }

                            if (!cariBuku) { //Percabangan if jika buku yang dimaksud tidak tersedia atau nilai variabel pengecek tidak true.
                                System.out.println("Buku tidak ada di perpustakaan");
                                System.out.println("Tekan Enter untuk kembali");
                                scanner.nextLine();
                            }
                        case 3: //Aksi untuk kembali ke menu utama
                            break;
                        default:
                            System.out.println("Pilihan tidak valid");
                    }
                } else if (pilihan == 3) { //Pilihan jika ingin keluar dari sistem/program perpustakaan
                    System.out.println("Terima kasih telah menggunakan sistem perpustakaan!");
                    break;
                } else { //Ketika input pilihan menu utama salah
                    System.out.println("Pilihan tidak valid. Silakan coba lagi.");
                    scanner.close();
                }

            }
        }
    }
```

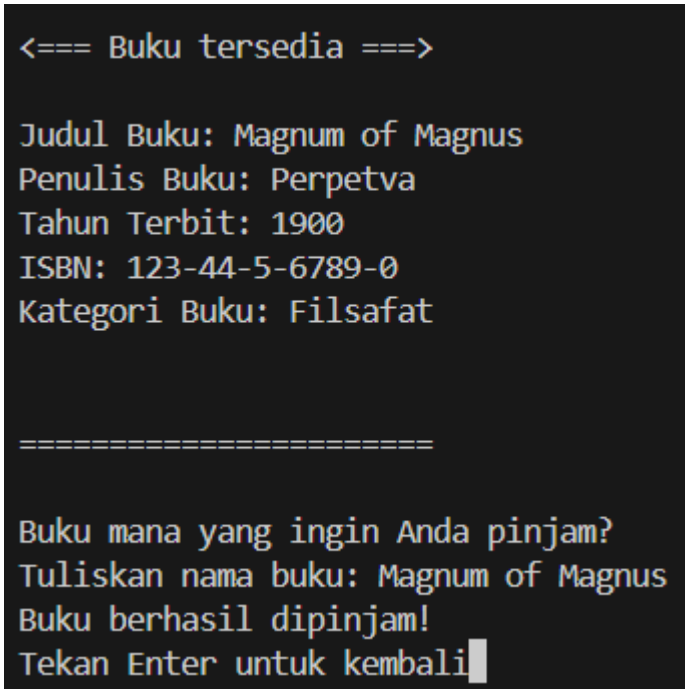
Output:

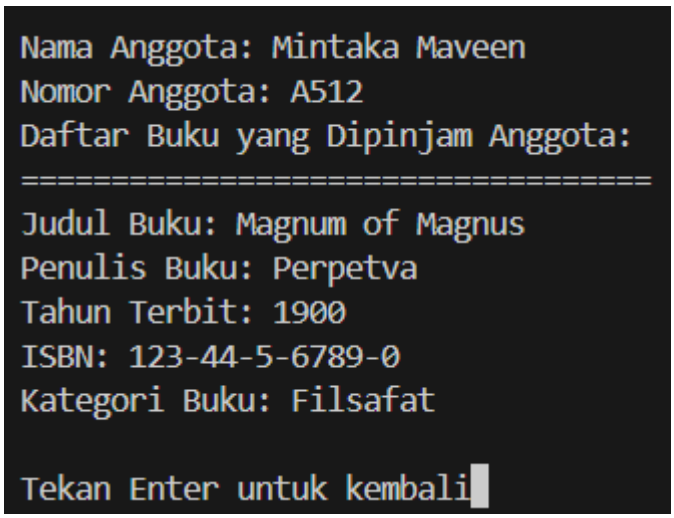
```
Pilih peran Anda:
1. Pustakawan
2. Anggota
3. Keluar
Pilihan:
```

```
<=== Selamat datang Master Alnilam Lambda ===>

Menu Pustakawan:
1. Daftar Peminjaman Buku
2. Tambah Buku
3. Tambah Anggota
4. Tampilkan Daftar & Informasi Anggota
5. Tampilkan Daftar Buku
6. Hapus Perpustakaan
7. Kembali

Pilihan:
```

```
Masukkan Judul: Magnum of Magnus
Masukkan Nama Penulis: Perpetva
Masukkan ISBN: 123-44-5-6789-0
Masukkan Tahun Terbit: 1900
Masukkan Kategori: Filsafat
1 Buku berhasil ditambahkan! Tekan enter untuk melanjutkan
```

```
<=== Selamat Datang Mintaka Maveen ===>

Menu Anggota:
1. Informasi Diri
2. Pinjam Buku
3. Kembali

Pilihan:
```





Penjelasan:

Pertama-tama, buat kelas Buku terlebih dahulu dengan field untuk judul, penulis, ISBN bertipe String dan tahun terbit bertipe integer serta kategori yang diambil dari kelas Kategori. Setelah itu buat konstruktor dari kelas Buku dan juga getter untuk judul. Selanjutnya, tambahkan method infoBuku() untuk menampilkan informasi dari buku yang terdiri dari judul, penulis, tahun terbit, ISBN, dan kategori yang diambil dari nilai field.

Ketika kelas Buku sudah dibuat, selanjutnya adalah membuat kelas Kategori agar bisa dipakai pada kelas Buku. Kelas Kategori sendiri terdiri dari field untuk nama beserta konstruktor dan getter untuk nama tersebut.

Setelah kelas Kategori sudah selesai dibuat, setelah itu adalah membuat kelas Anggota dengan konstruktor berupa String nama anggota dan nomor anggota serta kelas Buku yang dijadikan List dengan nama daftarPinjaman. Selanjutnya adalah membuat getter untuk memanggil nama anggota dan getter untuk memanggil daftarPinjaman. Tambahkan pula method pinjamBuku() untuk meminjam buku sebagai anggota dan method infoAnggota() untuk menampilkan informasi lengkap mengenai anggota serta method hapusPinjaman() untuk menghapus seluruh buku yang dipinjam oleh anggota.

Kelas Anggota yang sudah selesai dibuat dilanjutkan dengan pembuatan kelas Pustakawan yang terdiri dari field untuk nama pustakawan bertipe String. Setelah itu, tambahkan konstruktor, getter untuk memanggil nama pustakawan, dan method inputBuku() untuk menambahkan buku ke perpustakaan sebagai seorang pustakawan serta method prosesPeminjaman() untuk melihat proses peminjaman yang dilakukan oleh anggota perpustakaan.

Kelas Perpustakaan adalah langkah selanjutnya setelah Kelas Pustakawan sudah selesai dibuat. Kelas ini terdiri dari konstruktor yang memiliki List dari kelas Buku bernama daftarBuku dan Anggota bernama daftarAnggota untuk menyimpan informasi dari setiap buku dan anggota yang ada. Setelah itu, tambahkan method tambahBuku() untuk menambahkan buku ke dalam perpustakaan, method tambahAnggota() untuk menambahkan anggota ke dalam perpustakaan, lalu getter getDaftarBuku() untuk memanggil daftarBuku dan getter getDaftarAnggota() untuk memanggil daftarAnggota. Jangan lupa membuat method tampilkanDaftarBuku() yang berisi perulangan foreach pada daftarBuku untuk menampilkan buku-buku yang ada. Sama halnya dengan method tampilkanDaftarAnggota() yang menampilkan daftar anggota yang ada di perpustakaan. Setelah itu, tambahkan pula method hapusPerpustakaan() untuk menghapus data perpustakaan.

Setelah kelima kelas tersebut selesai, buatlah kelas Main yang akan menjadi kelas utama yang bisa di-*run* dan menghasilkan outputnya. Pertama-tama, cantumkan dulu kelas Scanner karena akan melibatkan input dari pengguna serta kelas Clear. Setelah itu, tambahkan kelas Pustakawan, Anggota, Buku, dan Perpustakaan untuk melakukan inisialisasi di awal kelas. Setelah itu, buatlah perulangan while karena kelas Main ini berupa sebuah sistem perpustakaan yang memadukan relasi antar kelima kelas tersebut. Pengguna bisa memilih peran sebagai pustakawan atau anggota. Apabila pengguna memilih peran sebagai pustakawan, maka ia bisa melihat proses peminjaman buku oleh pengguna, menambahkan buku dan anggota secara manual serta menampilkan informasi buku maupun anggota beserta pilihan untuk menghapus perpustakaan jika mau. Apabila pengguna memilih peran sebagai anggota, maka ia bisa melihat informasi dirinya sebagai anggota dan juga meminjam buku dengan melihat daftar buku yang tersedia di perpustakaan. Seluruh opsi atau pilihan aksi dalam kedua peran tersebut melibatkan penggunaan kelas yang saling memiliki relasi tersebut. Sebagai contoh, jika memilih peran sebagai pustakawan, maka pasti kebanyakan kelas yang dipakai adalah Pustakawan untuk menambahkan buku atau anggota. Setiap kelas

memiliki relasi dengan yang lainnya dan dengan tipe yang berbeda-beda sesuai dengan kebutuhan yang ada.