Ujian Tengah Semester
PEMROGRAMAN BERORIENTASI OBJEK (PBO)

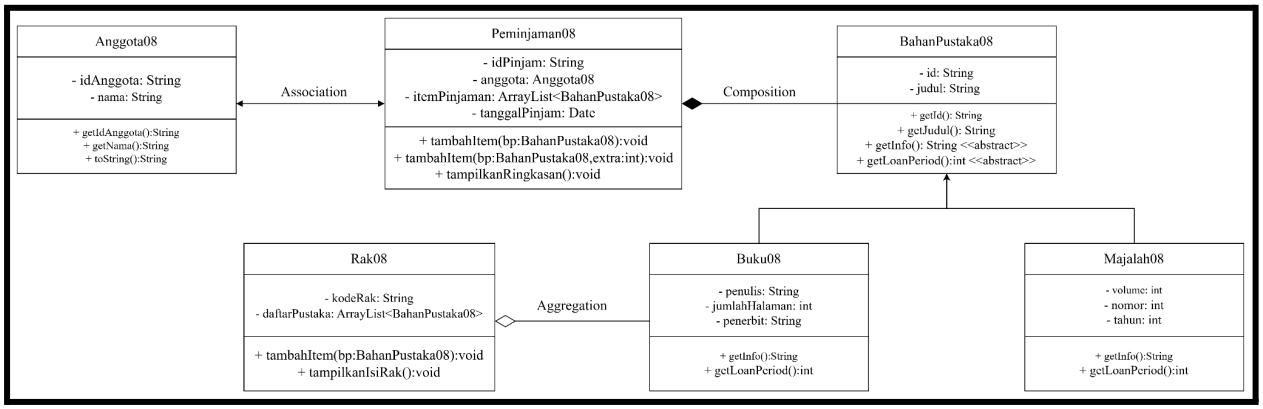
Nama : Mochammad Tanggaq Dirat Saputra
NIM : 244107060126
Kelas : SIB 2D

INTERAKSI UMUM

- Terapkan enkapsulasi penuh untuk atribut (private + getter/setter seperlunya), minimal 1 association dan 1 composition/aggregation, 1 pewarisan, 1 method overloading, dan 1 method overriding.
- Buat main() untuk demonstrasi singkat.

Bagian A – Perancangan (30%)

1. Gambar class diagram lengkap untuk skenario varian Anda. Tunjukkan multiplicity dan jenis relasi (association/aggregation/composition). (15%)



2. Tuliskan metode yang akan di override. (5%)

+ getInfo():String
+ getLoanPeriod():int

3. Tulis 3 keputusan desain terkait access modifier, enkapsulasi, dan pewarisan. (10%)

- Access Modifier - Private untuk Atribut
  Keputusan: Semua atribut (seperti idAnggota, nama, id, judul, kodeRak, dll.) dibuat private.
- Enkapsulasi - Penggunaan Getter untuk Akses Aman
  Keputusan: Kelas seperti Anggota08 dan BahanPustaka08 menyediakan getter (getIdAnggota(), getJudul(), dll.) untuk membaca data atribut private.
- Pewarisan (Inheritance) — Kelas Abstrak dan Override
  Keputusan: Kelas BahanPustaka08 dijadikan **abstract class**, dan subclass-nya (Buku08 dan Majalah08) mewarisi serta **meng-override** method getInfo() dan getLoanPeriod().

Bagian B – Implementasi (50%)

Implementasikan kelas inti berikut: Anggota, BahanPustaka (abstrak), Buku, Majalah, Peminjaman, Rak.

4. Syarat minimal implementasi:
    - Semua atribut private, sediakan getter/setter yang diperlukan saja. (10%)
    - 1 konstruktor berparameter di tiap kelas domain utama. (5%)
    - 1 inheritance + 1 override di tiap subclass. (15%)
    - 1 method overloading fungsional (beda signature). (10%)
    - Method main singkat yang mendemokan relasi & polimorfisme. (10%)

Anggota08.java

```java
package UTS;

public class Anggota08 {
    private String idAnggota;
    private String nama;

    public Anggota08(String idAnggota, String nama) {
        this.idAnggota = idAnggota;
        this.nama = nama;
    }

    public String getIdAnggota() {
        return idAnggota;
    }

    public String getNama() {
        return nama;
    }

    @Override
    public String toString() {
        return nama + " (" + idAnggota + ")";
    }
}
```

BahanPustaka08.java > BahanPustaka08 > getInfo()

```java
package UTS;

public abstract class BahanPustaka08 {
    private String id;
    private String judul;

    public BahanPustaka08(String id, String judul) {
        this.id = id;
        this.judul = judul;
    }

    public String getId() {
        return id;
    }

    public String getJudul() {
        return judul;
    }

    public abstract String getInfo();

    // Method yang akan dioverride
    public abstract int getLoanPeriod();
}
```

Buku08.java > Buku08 > getLoanPeriod()

```java
package UTS;

public class Buku08 extends BahanPustaka08 {
    private String penulis;
    private int jumlahHalaman;
    private String penerbit;

    public Buku08(String id, String judul, String penulis, int jumlahHalaman, String penerbit) {
        super(id, judul);
        this.penulis = penulis;
        this.jumlahHalaman = jumlahHalaman;
        this.penerbit = penerbit;
    }

    @Override
    public String getInfo() {
        return "Buku [" + getId() + "] " + getJudul() + " oleh " + penulis +
               " (" + jumlahHalaman + " halaman) - " + penerbit;
    }

    @Override
    public int getLoanPeriod() {
        return 14; // 14 hari
    }
}
```

Majalah08.java > ...

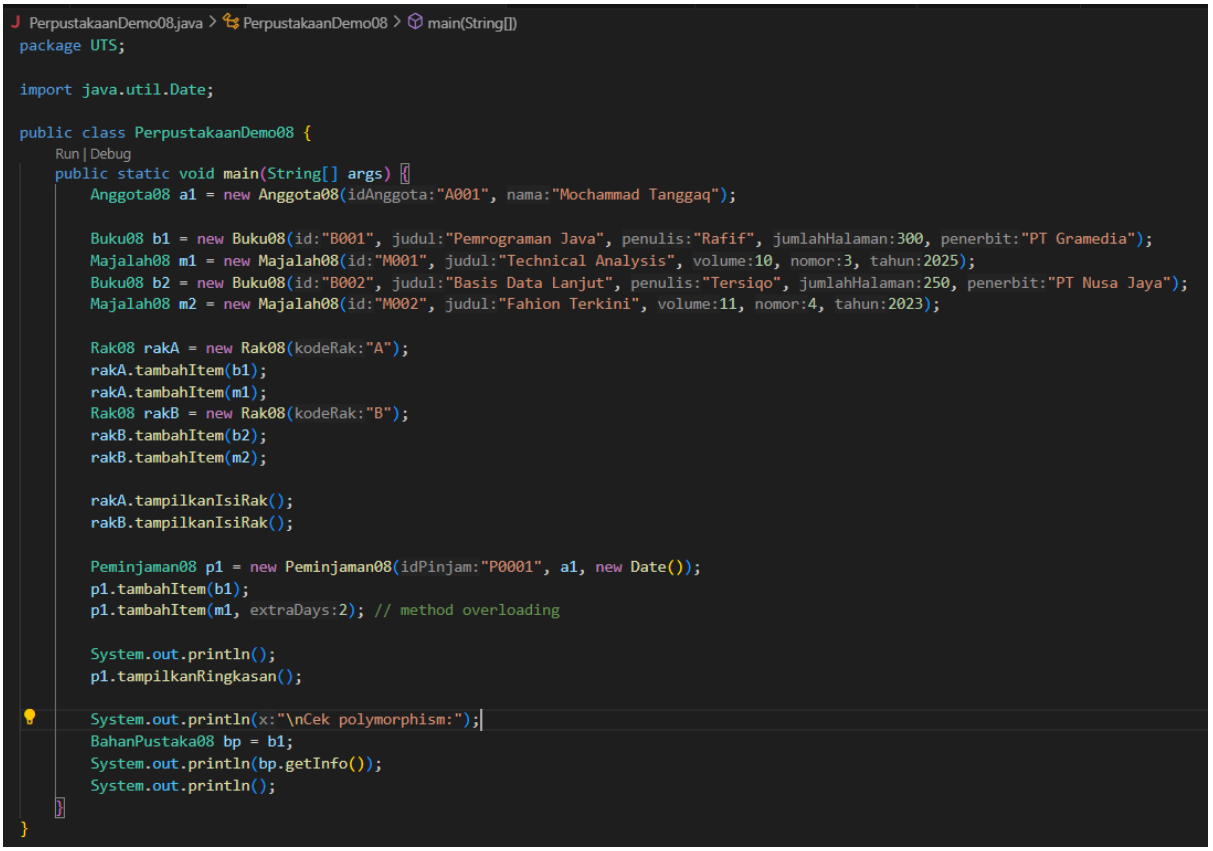
```java
package UTS;

public class Majalah08 extends BahanPustaka08 {
    private int volume;
    private int nomor;
    private int tahun;

    public Majalah08(String id, String judul, int volume, int nomor, int tahun) {
        super(id, judul);
        this.volume = volume;
        this.nomor = nomor;
        this.tahun = tahun;
    }

    @Override
    public String getInfo() {
        return "Majalah [" + getId() + "] " + getJudul() +
               " Vol." + volume + " No." + nomor + " (" + tahun + ")";
    }

    @Override
    public int getLoanPeriod() {
        return 3; // 3 hari
    }
}
```

Peminjaman08.java > Peminjaman08 > tambahItem(BahanPustaka08, int)

```java
package UTS;

import java.util.ArrayList;
import java.util.Date;

public class Peminjaman08 {
    private String idPinjam;
    private Anggota08 anggota;
    private ArrayList<BahanPustaka08> itemPinjaman = new ArrayList<>();
    private Date tanggalPinjam;

    public Peminjaman08(String idPinjam, Anggota08 anggota, Date tanggalPinjam) {
        this.idPinjam = idPinjam;
        this.anggota = anggota;
        this.tanggalPinjam = tanggalPinjam;
    }

    // Overloading
    public void tambahItem(BahanPustaka08 item) {
        itemPinjaman.add(item);
    }

    public void tambahItem(BahanPustaka08 item, int extraDays) {
        itemPinjaman.add(item);
        System.out.println("Item " + item.getJudul() + " dipinjam dengan tambahan hari = " + extraDays);
    }

    public void tampilkanRingkasan() {
        System.out.println("Peminjaman " + idPinjam + " oleh " + anggota + " pada " + tanggalPinjam);
        System.out.println(x:"Item yang dipinjam:");
        for (BahanPustaka08 bp : itemPinjaman) {
            System.out.println(" - " + bp.getInfo() + " (" + bp.getLoanPeriod() + " hari)");
        }
    }
}
```

Rak08.java > ...

```java
package UTS;

import java.util.ArrayList;

public class Rak08 {
    private String kodeRak;
    private ArrayList<BahanPustaka08> daftarPustaka = new ArrayList<>();

    public Rak08(String kodeRak) {
        this.kodeRak = kodeRak;
    }

    public void tambahItem(BahanPustaka08 item) {
        daftarPustaka.add(item);
    }

    public void tampilkanIsiRak() {
        System.out.println("Isi Rak " + kodeRak + ":");
        for (BahanPustaka08 bp : daftarPustaka) {
            System.out.println(" - " + bp.getInfo());
        }
        System.out.println();
    }
}
```



```java
PerpustakaanDemo08.java > PerpustakaanDemo08 > main(String[])
package UTS;

import java.util.Date;

public class PerpustakaanDemo08 {
    Run | Debug
    public static void main(String[] args) {
        Anggota08 a1 = new Anggota08(idAnggota:"A001", nama:"Mochammad Tanggaq");

        Buku08 b1 = new Buku08(id:"B001", judul:"Pemrograman Java", penulis:"Rafif", jumlahHalaman:300, penerbit:"PT Gramedia");
        Majalah08 m1 = new Majalah08(id:"M001", judul:"Technical Analysis", volume:10, nomor:3, tahun:2025);
        Buku08 b2 = new Buku08(id:"B002", judul:"Basis Data Lanjut", penulis:"Tersiqo", jumlahHalaman:250, penerbit:"PT Nusa Jaya");
        Majalah08 m2 = new Majalah08(id:"M002", judul:"Fahion Terkini", volume:11, nomor:4, tahun:2023);

        Rak08 rakA = new Rak08(kodeRak:"A");
        rakA.tambahItem(b1);
        rakA.tambahItem(m1);
        Rak08 rakB = new Rak08(kodeRak:"B");
        rakB.tambahItem(b2);
        rakB.tambahItem(m2);

        rakA.tampilkanIsiRak();
        rakB.tampilkanIsiRak();

        Peminjaman08 p1 = new Peminjaman08(idPinjam:"P0001", a1, new Date());
        p1.tambahItem(b1);
        p1.tambahItem(m1, extraDays:2); // method overloading

        System.out.println();
        p1.tampilkanRingkasan();

        System.out.println(x:"\nCek polymorphism:");
        BahanPustaka08 bp = b1;
        System.out.println(bp.getInfo());
        System.out.println();
    }
}
```

Bagian C – Analisis (20%)

5. Jelaskan perbedaan overloading vs overriding dan berikan contoh dari kode Anda. (6%)
Jawab :
Method overloading berarti kondisi dimana ada method dengan nama yang sama, tetapi memiliki method signature yang berbeda di class yang sama. Method signature: jumlah, tipe data dan susunan parameter

6. Mengapa Anda memilih composition untuk X dan aggregation untuk Y? (6%)
Jawab :
Composition → Kelas Peminjaman08 dengan BahanPustaka08
Alasan:

- Objek BahanPustaka08 (buku/majalah) yang dimasukkan ke dalam daftar itemPinjaman hanya ada selama peminjaman berlangsung.
- Jika objek Peminjaman08 dihapus, maka daftar item pinjaman tersebut juga akan ikut hilang.

Aggregation → Kelas Rak08 dengan BahanPustaka08
Alasan:

- Rak08 menyimpan referensi ke objek Buku08, tetapi tidak memiliki siklus hidupnya.
- Buku tetap ada walaupun rak dihapus — bisa saja dipindahkan ke rak lain.

7. Sebutkan aturan method signature dan contoh yang tidak valid. (4%)
Jawab :
Aturan Metode Signature :

- Method signature ditentukan oleh nama method + urutan dan tipe parameter.
- Tipe data kembalian (return type) *tidak termasuk* dalam signature.
- Jadi, dua method dengan nama sama dan parameter identik tidak boleh ada dalam satu kelas, meskipun return type-nya berbeda.

Contoh Valid :
public void tambahItem(BahanPustaka08 item) { ... }
public void tambahItem(BahanPustaka08 item, int extraDays) { ... }

Contoh tidak Valid :
public int getJudul() { ... }
public String getJudul() { ... } // Tidak valid, karena nama & parameter sama

8. Alasan pemilihan access modifier pada minimal 2 atribut & 1 method. (4%)
Jawab :
Attribute 1 — private String idAnggota (kelas Anggota08)

- Alasan: Supaya ID anggota tidak dapat diubah langsung dari luar kelas.
  → Menghindari kesalahan atau manipulasi data.

Attribute 2 — private ArrayList<BahanPustaka08> itemPinjaman (kelas Peminjaman08)

- Alasan: Daftar item hanya boleh dimodifikasi melalui method tambahItem().
  → Menjaga integritas daftar pinjaman agar tetap konsisten.

Method — public String getInfo() (kelas Buku08 dan Majalah08)

- Alasan: Dibuat public agar informasi pustaka bisa ditampilkan di luar kelas, misalnya saat menampilkan isi rak atau ringkasan peminjaman.
  → Mendukung prinsip **abstraksi** dan **polymorphism** (dapat dipanggil dari referensi BahanPustaka08).