Nama : Yunan Faila Sofi

NIM : H1D024033

ShiftLama : G

ShiftBaru : C

Class KendaraanGalaksi

```
package pert6;

public abstract class KendaraanGalaksi {

    // ============================================================
    // ATRIBUT/STATE (semua private, diakses via getter/protected)
    // ============================================================

    private String namaKendaraan;
    protected int levelEnergi; // Menggunakan protected agar subclass mudah mengakses
    private int kapasitasPenumpang;


    // ============================================================
    // CONSTRUCTOR
    // ============================================================

    public KendaraanGalaksi(String namaKendaraan, int kapasitasPenumpang) {
        this.namaKendaraan = namaKendaraan;
        this.kapasitasPenumpang = kapasitasPenumpang;
        this.levelEnergi = 100; // levelEnergi default = 100
    }


    // ============================================================
    // GETTER
    // ============================================================

    public String getNamaKendaraan() {
        return namaKendaraan;
    }

    public int getLevelEnergi() {
```

```
        return levelEnergi;
    }

    public int getKapasitasPenumpang() {
        return kapasitasPenumpang;
    }


    // ============================================================
    // METHOD BIASA
    // ============================================================

    /**
     * Method final tampilStatus()
     * Format: "[namaKendaraan] | Energi: [levelEnergi]% | Kapasitas: [kapasitasPenumpang] orang"
     */
    public final void tampilStatus() {
        System.out.println(
                this.namaKendaraan +
                        " | Energi: " + this.levelEnergi + "%" +
                        " | Kapasitas: " + this.kapasitasPenumpang + " orang"
        );
    }


    // ============================================================
    // ABSTRACT METHOD (WAJIB DIIMPLEMENTASI SUBCLASS)
    // ============================================================

    public abstract void aktifkanMesin();

    public abstract void jelajah(int jarak);

    public abstract void isiEnergi(int jumlah);
}
```

Class KapalEksplorasi

```
package pert6;
```

```
public class KapalEksplorasi extends KendaraanGalaksi {

    // ============================================================
    // ATRIBUT KHUSUS
    // ============================================================

    private int modulScan; // kemampuan scan planet (level 1–5)


    // ============================================================
    // CONSTRUCTOR
    // ============================================================

    public KapalEksplorasi(String namaKendaraan, int kapasitasPenumpang, int modulScan) {
        super(namaKendaraan, kapasitasPenumpang);
        this.modulScan = modulScan;
    }


    // ============================================================
    // IMPLEMENTASI ABSTRACT METHOD
    // ============================================================

    @Override
    public void aktifkanMesin() {
        // Tidak dapat memulai misi jika energi di bawah 15%
        if (this.levelEnergi < 15) {
            System.out.println("Energi tidak mencukupi untuk memulai ekspedisi!");
        } else {
            System.out.println("Kapal eksplorasi siap berangkat!");
        }
    }

    @Override
    public void jelajah(int jarak) {
        // Konsumsi energi: 2% per 1 km.
        double konsumsi = jarak * 2.0;

        if (this.levelEnergi < konsumsi) {
            System.out.println("Energi tidak mencukupi untuk menjelajah sejauh " + jarak + " km.");
```

```
        } else {
            this.levelEnergi -= (int) konsumsi;
            System.out.println("Kapal eksplorasi menjelajah sejauh " + jarak + " km.");
        }
    }

    @Override
    public void isiEnergi(int jumlah) {
        int energiAwal = this.levelEnergi;
        this.levelEnergi = Math.min(100, this.levelEnergi + jumlah);
        System.out.println("Mengisi energi Kapal Eksplorasi. Energi naik dari " + energiAwal + "%
menjadi " + this.levelEnergi + "%.");
    }


    // ============================================================
    // METHOD KHUSUS
    // ============================================================

    public void scanPlanet(String namaPlanet) {
        // Dapat melakukan scan planet untuk mengumpulkan data ilmiah
        System.out.println(
                "Melakukan scan pada planet " + namaPlanet +
                        " dengan modul level " + this.modulScan + "."
        );
    }
}
```

Class PeswatTempur

```
package pert6;

public class PesawatTempur extends KendaraanGalaksi {

    // ============================================================
    // ATRIBUT KHUSUS
    // ============================================================

    private int jumlahRudal;
```

```
// ============================================================
// CONSTRUCTOR
// ============================================================

public PesawatTempur(String namaKendaraan, int kapasitasPenumpang, int jumlahRudal) {
    // Panggil constructor parent
    super(namaKendaraan, kapasitasPenumpang);
    this.jumlahRudal = jumlahRudal;
}


// ============================================================
// IMPLEMENTASI ABSTRACT METHOD
// ============================================================

@Override
public void aktifkanMesin() {
    // Tidak boleh dinyalakan jika energi di bawah 20%
    if (this.levelEnergi < 20) {
        System.out.println("Energi terlalu rendah! Mesin tidak dapat diaktifkan.");
    } else {
        System.out.println("Mesin pesawat tempur diaktifkan.");
    }
}

@Override
public void jelajah(int jarak) {
    // Konsumsi energi: 3% per 1 km.
    double konsumsi = jarak * 3.0;

    if (this.levelEnergi < konsumsi) {
        System.out.println("Energi tidak mencukupi untuk menjelajah sejauh " + jarak + " km.");
    } else {
        this.levelEnergi -= (int) konsumsi;
        System.out.println("Pesawat tempur menjelajah sejauh " + jarak + " km.");
    }
}

@Override
public void isiEnergi(int jumlah) {
```

```
        int energiAwal = this.levelEnergi;
        this.levelEnergi = Math.min(100, this.levelEnergi + jumlah);
        System.out.println("Mengisi energi Pesawat Tempur. Energi naik dari " + energiAwal + "% menjadi " + this.levelEnergi + "%.");
    }


    // ============================================================
    // METHOD KHUSUS
    // ============================================================

    public void tembakRudal(int jumlah) {
        // Hanya dapat menembakkan rudal jika jumlahnya mencukupi
        if (this.jumlahRudal >= jumlah) {
            this.jumlahRudal -= jumlah;
            System.out.println("Menembakkan " + jumlah + " rudal!");
        } else {
            System.out.println("Gagal menembak. Jumlah rudal (" + this.jumlahRudal + ") tidak mencukupi.");
        }
    }
}
```

1. Alur Kerja Program (Workflow)

Program berjalan secara sekuensial dari kelas UjiGalaksi. Berikut adalah langkah-langkah logisnya:

1) Inisialisasi Program: Dimulai dari method main.
2) Skenario 1: Pesawat Tempur (Astra-Fury)
    a. Instansiasi: Membuat objek astraFury. Constructor PesawatTempur dipanggil, yang kemudian memanggil constructor KendaraanGalaksi (Parent) untuk menyetel nama dan energi awal 100%.
    b. Aktivasi: aktifkanMesin() dijalankan. Karena energi 100% (>20%), mesin menyala.
    c. Penjelajahan Berhasil: jelajah(10) dipanggil. Konsumsi energi: $10 \text{ km} \times 3\% = 30\%$. Sisa energi: $100 - 30 = 70\%$.

d. Penjelajahan Gagal: jelajah(30) dipanggil. Konsumsi yang dibutuhkan: $30 \text{ km} \times 3\% = 90\%$. Karena sisa energi hanya 70% (kurang dari 90%), misi dibatalkan.

e. Aksi Khusus: tembakRudal(3) dijalankan. Stok rudal berkurang dari 8 menjadi 5.

f. Cetak Status: tampilStatus() (method final dari parent) dipanggil untuk menampilkan data akhir.

3) Skenario 2: Kapal Eksplorasi (Voyager X)

a. Instansiasi: Membuat objek voyagerX.

b. Aktivasi: aktifkanMesin() dijalankan. Syarat minimal hanya 15%, mesin menyala.

c. Penjelajahan: jelajah(15) dipanggil. Konsumsi energi: $15 \text{ km} \times 2\% = 30\%$. Sisa energi: $100 - 30 = 70\%$.

d. Aksi Khusus: scanPlanet(...) dijalankan untuk menampilkan simulasi scanning.

e. Cetak Status: Menampilkan data akhir kapal.

2. Fungsi dan Komponen yang Digunakan

Kode ini menerapkan konsep Abstraksi yang ketat untuk memastikan standar desain pada semua kendaraan.

A. Kelas KendaraanGalaksi (Abstract Class)
Ini adalah "kontrak" atau cetak biru dasar yang tidak bisa dibuat objeknya secara langsung (*cannot be instantiated*).
- abstract Method (aktifkanMesin, jelajah, isiEnergi): Method ini tidak memiliki body (isi) di kelas parent. Fungsinya memaksa setiap kelas anak (KapalEksplorasi, PesawatTempur) untuk membuat logika versi mereka sendiri. Jika tidak dibuat, program akan *error*.
- final Method (tampilStatus): Method ini sudah didefinisikan secara utuh dan dilarang di-override oleh subclass. Ini menjaga agar format tampilan status selalu seragam.
- protected Attribute: Variabel levelEnergi dibuat protected agar kelas anak bisa memodifikasi nilainya secara langsung (misal: pengurangan energi saat menjelajah).

B. Kelas PesawatTempur (Concrete Class)
Mengimplementasikan semua method abstrak dengan logika "boros energi" tapi memiliki senjata.
- Logika jelajah: Konsumsi energi lebih besar (3% per km).
- Method Unik: Memiliki fitur tembakRudal yang tidak dimiliki kendaraan lain.

C. Kelas KapalEksplorasi (Concrete Class)
Mengimplementasikan semua method abstrak dengan logika "hemat energi" untuk perjalanan jauh.

- Logika jelajah: Konsumsi energi lebih irit (2% per km).
- Method Unik: Memiliki fitur scanPlanet.

3. Hasil Output Program

```
=== UJI SISTEM KENDARAAN GALAKSI ===

--- PESAWAT TEMPUR ---
Mesin pesawat tempur diaktifkan.
Pesawat tempur menjelajah sejauh 10 km.
Energi tidak mencukupi untuk menjelajah sejauh 30 km.
Menembakkan 3 rudal!
Astra-Fury | Energi: 70% | Kapasitas: 2 orang

--- KAPAL EKSPLORASI ---
Kapal eksplorasi siap berangkat!
Kapal eksplorasi menjelajah sejauh 15 km.
Melakukan scan pada planet Kepler-442b dengan modul level 4.
Voyager X | Energi: 70% | Kapasitas: 10 orang

Process finished with exit code 0
```

Analisis Output:

1. Kegagalan Astra-Fury: Perhatikan baris ke-3 output Pesawat Tempur. Ia gagal menjelajah 30 km karena butuh 90% energi, padahal sisa energinya tinggal 70% (akibat perjalanan sebelumnya).
2. Sisa Energi Sama: Kebetulan kedua kendaraan berakhir dengan sisa energi 70%, namun penyebabnya berbeda (Pesawat: $10 \times 3\% = 30\%$, Kapal: $15 \times 2\% = 30\%$).