

Nama : Yunan Faila Sofi

NIM : H1D024033

ShiftLama : G

ShiftBaru : C

Class defaultMesin

```
package pert5;
```

```
public class defaultMesin {  
    // Variabel umum mesin (protected agar bisa diakses subclass)  
    protected String namaMesin;  
    protected int tenagaHP;  
  
    // Constructor  
    public defaultMesin(String nama, int hp) {  
        this.namaMesin = nama;  
        this.tenagaHP = hp;  
    }  
  
    public void tampilInfo() {  
        System.out.printf("Mesin %s | Tenaga: %d HP\n", this.namaMesin, this.tenagaHP);  
    }  
  
    public double nilaiPerforma() {  
        // Perhitungan standar (default)  
        return this.tenagaHP * 1.0;  
    }  
  
    public String kategoriMesin() {  
        // Kategori default  
        return "Mesin Umum";  
    }  
}
```

Class mesinMotor

```
package pert5;
```

```
public class mesinMotor extends defaultMesin {  
    // Variabel khusus mesin motor
```

```

protected String tipeMotor;

// Constructor
public mesinMotor(String nama, int hp, String tipe) {
    super(nama, hp);
    this.tipeMotor = tipe;
}

@Override
public void tampilInfo() {
    System.out.printf("Mesin Motor %s | Tipe: %s | Tenaga: %d HP\n",
        this.namaMesin, this.tipeMotor, this.tenagaHP);
}

@Override
public double nilaiPerforma() {
    // performa = tenagaHP x 1.2
    return this.tenagaHP * 1.2;
}

@Override
public String kategoriMesin() {
    return "Mesin Motor";
}

// Method unik
public String suaraMesin() {
    return "Brummm! Mesin motor menyala!";
}
}

```

Class mesinTraktor

```

package pert5;

public class mesinTraktor extends defaultMesin {
    // Variabel khusus mesin traktor
    protected double kapasitasTarik;

    // Constructor
    public mesinTraktor(String nama, int hp, double tarik) {

```

```

        super(nama, hp);
        this.kapasitasTarik = tarik;
    }

    @Override
    public void tampilInfo() {
        System.out.printf("Mesin Traktor %s | Tarik: %.1f ton | Tenaga: %d HP\n",
            this.namaMesin, this.kapasitasTarik, this.tenagaHP);
    }

    @Override
    public double nilaiPerforma() {
        // Rumus performa traktor: (tenagaHP * 1.05) + (kapasitasTarik * 5)
        return (this.tenagaHP * 1.05) + (this.kapasitasTarik * 5.0);
    }

    @Override
    public String kategoriMesin() {
        return "Mesin Traktor";
    }

    // Method unik
    public String suaraMesin() {
        return "GGGRRRRR! Hidup mesinnn!";
    }
}

```

Class mesinTraktor

```

package pert5;

public class mesinTraktorListrik extends mesinTraktor {
    // Variabel khusus mesin traktor listrik
    protected double kapasitasBaterai;

    // Constructor
    public mesinTraktorListrik(String nama, int hp, double tarik, double baterai) {
        super(nama, hp, tarik); // Panggil constructor parent (mesinTraktor)
        this.kapasitasBaterai = baterai;
    }
}

```

```

@Override
public void tampilInfo() {
    System.out.printf("Mesin Traktor Listrik %s | Tarik: %.1f ton | Baterai: %.0f kWh | Tenaga:
%d HP\n",
        this.namaMesin, this.kapasitasTarik, this.kapasitasBaterai, this.tenagaHP);
}

@Override
public double nilaiPerforma() {
    // Rumus performa traktor listrik
    // performa = tenagaHP x 0.9 + kapasitasTarik x 10 + kapasitasBaterai x 5
    return (this.tenagaHP * 0.9) + (this.kapasitasTarik * 10.0) + (this.kapasitasBaterai * 5.0);
}

@Override
public String kategoriMesin() {
    return "Mesin Traktor Listrik";
}

@Override // Override suaraMesin dari mesinTraktor
public String suaraMesin() {
    return "Bzzzzz! Mesin traktor listrik aktif!";
}
}

```

1. Alur Kerja Program (Workflow)

Program ini memiliki logika yang lebih kompleks dibandingkan pertemuan sebelumnya karena melibatkan pengolahan data dalam *array* dan algoritma *sorting*. Berikut langkah-langkahnya:

- 1) Inisialisasi Array Polimorfik:
 - a. Program membuat array `inventarisMesin` dengan tipe induk (`defaultMesin`).
 - b. Array ini diisi oleh berbagai jenis objek anak: `mesinMotor`, `mesinTraktor`, dan `mesinTraktorListrik`. Inilah implementasi Polimorfisme, di mana satu tipe variabel (parent) dapat menyimpan berbagai bentuk objek (children).
- 2) Iterasi 1: Menampilkan Informasi & Performa (Polimorfisme):
 - a. Program melakukan *looping* (perulangan) pada array.
 - b. Saat memanggil `mesin.tampilInfo()` dan `mesin.nilaiPerforma()`, program secara otomatis memilih method milik objek aslinya (Dynamic Binding). Contoh: Jika

objeknya mesinTraktorListrik, maka rumus performa yang dijalankan adalah rumus kompleks (baterai + tarik + hp), bukan rumus standar.

3) Iterasi 2: Menampilkan Suara (Downcasting):

- a. Method suaraMesin() tidak ada di kelas induk (defaultMesin).
- b. Oleh karena itu, program menggunakan pengecekan instanceof untuk mengetahui tipe asli objek.
- c. Setelah tipe diketahui, dilakukan Downcasting (contoh: (mesinMotor) mesin) agar program bisa mengakses method unik suaraMesin() yang hanya dimiliki oleh subclass.

4) Pencarian Nilai Tertinggi (Max Value Algorithm):

- a. Program melakukan loop untuk membandingkan nilaiPerforma() setiap mesin guna mencari angka terbesar.

5) Pengurutan Data (Bubble Sort):

- a. Program mengurutkan mesin berdasarkan nilai performa dari terbesar ke terkecil (*descending*).
- b. Karena tidak menggunakan library bawaan (Arrays.sort), program menggunakan logika Bubble Sort (membandingkan elemen j dengan j+1, lalu menukar posisi jika urutannya salah).
- c. Hasil akhirnya, 3 mesin dengan performa terbaik ditampilkan.

2. Fungsi dan Komponen yang Digunakan

Berikut adalah elemen teknis utama dalam kode pert5:

- a. Konsep Inheritance (Pewarisan)
 - extends: Digunakan untuk mewarisi sifat. Perhatikan pada mesinTraktorListrik extends mesinTraktor. Ini disebut Multilevel Inheritance (Mesin Umum -> Traktor -> Traktor Listrik).
 - super: Digunakan di constructor untuk meneruskan data ke parent class di atasnya.
- b. Konsep Polimorfisme
 - Overriding Method: Setiap subclass (Motor, Traktor) memiliki cara menghitung nilaiPerforma yang berbeda-beda, menimpa rumus defaultMesin.
 - Array of Parent Type: defaultMesin[] memungkinkan kita menyimpan objek Motor dan Traktor dalam satu wadah yang sama.
- c. Type Casting & Instanceof

- instanceof: Operator logika untuk mengecek "Apakah objek ini adalah instan dari kelas X?".
 - Downcasting: (mesinMotor) mesin. Mengubah referensi tipe parent kembali ke tipe child spesifik untuk mengakses fitur unik (seperti suaraMesin).
- d. Algoritma
- Bubble Sort: Algoritma pengurutan sederhana yang digunakan di akhir program untuk menukar posisi objek di dalam array berdasarkan nilai performanya.

3. Hasil Output Program

```
=== DATA MESIN MEGATECH ===
Mesin Motor Honda Supra X | Tipe: Bebek | Tenaga: 125 HP
Kategori: Mesin Motor
Performa: 150.0

Mesin Traktor Kubota MX5200 | Tarik: 5.0 ton | Tenaga: 520 HP
Kategori: Mesin Traktor
Performa: 571.0

Mesin Traktor Listrik EcoTrac Z900 | Tarik: 4.2 ton | Baterai: 70 kWh | Tenaga: 300 HP
Kategori: Mesin Traktor Listrik
Performa: 662.0

Mesin Motor Yamaha R25 | Tipe: Sport | Tenaga: 250 HP
Kategori: Mesin Motor
Performa: 300.0

Mesin Traktor Listrik Volta FarmX | Tarik: 3.5 ton | Baterai: 80 kWh | Tenaga: 300 HP
Kategori: Mesin Traktor Listrik
Performa: 705.0
```

```
=== SUARA MESIN ===
Honda Supra X -> Brummm! Mesin motor menyala!
Kubota MX5200 -> GGGRRRR! Hidup mesin!
EcoTrac Z900 -> Bzzzzz! Mesin traktor listrik aktif!
Yamaha R25 -> Brummm! Mesin motor menyala!
Volta FarmX -> Bzzzzz! Mesin traktor listrik aktif!

=== MESIN PERFORMA TERTINGGI ===
Volta FarmX -> 705.0

=== TOP 3 MESIN TERBAIK ===
1. Volta FarmX -> 705.0
2. EcoTrac Z900 -> 662.0
3. Kubota MX5200 -> 571.0

Process finished with exit code 0
```

Perhitungan Manual Performa:

1. Honda Supra X: $\$125 \times 1.2 = 150.0\$$
2. Kubota MX5200: $\$(520 \times 1.05) + (5.0 \times 5) = 546 + 25 = 571.0\$$
3. EcoTrac Z900: $\$(300 \times 0.9) + (4.2 \times 10) + (70 \times 5) = 270 + 42 + 350 = 662.0\$$
4. Yamaha R25: $\$250 \times 1.2 = 300.0\$$
5. Volta FarmX: $\$(300 \times 0.9) + (3.5 \times 10) + (80 \times 5) = 270 + 35 + 400 = 705.0\$$