LAPORAN HASIL PRAKTIKUM 06

PEMROMGRAMAN BERBASIS OBJEK

ATHAULLA HAFIZH

244107020030

TI 2A

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

3. Percobaan

Percobaan 1

Kode program class Hewan

```
package HewanOOP1;

class Hewan {

    String nama;


    public Hewan(String nama) {

        this.nama = nama;

    }


    void bersuara() {

        System.out.println("Suara hewan...");

    }

}
```

Kode program class Kucing

```
package HewanOOP1;


public class Main {

    public static void main(String[] args) {

        Kucing kucing = new Kucing("Milo");

        System.out.println("Nama kucing: " + kucing.nama); //
Mengakses atribut dari superclass

        kucing.bersuara(); // Memanggil metode dari superclass

        kucing.mengeong(); // Memanggil metode dari subclass

    }

}
```

Kode program class Kucing

```
package HewanOOP1;

class Kucing extends Hewan {
    public Kucing(String nama) {
        super(nama); // Memanggil konstruktor superclass Hewan
    }

    void mengeong() {
        System.out.println("Meong!");
    }
}
```

Tugas Praktikum

Kode program class Anjing

```
class Anjing extends Hewan {
    public Anjing(String nama) {
        super(nama); // Memanggil konstruktor superclass Hewan
    }

    void menggonggong() {
        System.out.println("Guk... Guk!");
    }
}
```

Percobaan 2

Kode program class Hewan

```
package HewanOOP1;
class Hewan {
    String nama;

    public Hewan(String nama) {
        this.nama = nama;
    }
```

```
    void bersuara() {
        System.out.println("Suara hewan...");
    }
}
```

Kode program class Main

```
package HewanOOP2;


public class Main {
    public static void main(String[] args) {
        Kucing kucing = new Kucing("Milo", 2);
        System.out.println("Nama: " + kucing.nama + ", Umur: " +
kucing.umur + " tahun");
    }
}
```

Tugas Praktikum

Kode program class Kucing

```
class Kucing extends Hewan {
    private String ras; // Atribut tambahan


    // Konstruktor dimodifikasi untuk menerima parameter ras
    public Kucing(String nama, int umur, String ras) {
        super(nama, umur); // Memanggil konstruktor Hewan dengan
benar
        this.ras = ras;
        System.out.println("Konstruktor Kucing dipanggil");
    }


    public void info() {
        System.out.println("Nama: " + nama);
        System.out.println("Umur: " + umur + " tahun");
        System.out.println("Ras: " + ras);
    }
}
```

Kode program class Main

```
package HewanOOP2;


public class Main {
    public static void main(String[] args) {
        // Membuat objek Kucing dengan konstruktor baru
        Kucing kucing = new Kucing("Milo", 2, "Persia");
        System.out.println("--------------------");
        kucing.info();
    }
}
```

Percobaan 3

Kode program class Hewan

```
class Hewan {
    String nama;
    int umur;

    public Hewan(String nama, int umur) {
        this.nama = nama;
        this.umur = umur;
        System.out.println("Konstruktor Hewan dipanggil");
    }
}
```

Kode program class Kucing

```
class Kucing extends Hewan {
    public Kucing(String nama, int umur) {
        super(nama, umur); // Memanggil konstruktor Hewan
        System.out.println("Konstruktor Kucing dipanggil");
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Kucing kucing = new Kucing("Milo", 2);
        System.out.println("Nama: " + kucing.nama + ", Umur: " +
kucing.umur + " tahun");
    }
}
```

Tugas Praktikum

Kode program class Kucing

```
class Kucing extends Hewan {
    private String ras; // Atribut tambahan

    // Konstruktor dimodifikasi untuk menerima parameter ras
    public Kucing(String nama, int umur, String ras) {
        super(nama, umur); // Memanggil konstruktor Hewan dengan
benar
        this.ras = ras;
        System.out.println("Konstruktor Kucing dipanggil");
    }

    public void info() {
        System.out.println("Nama: " + nama);
        System.out.println("Umur: " + umur + " tahun");
        System.out.println("Ras: " + ras);
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        // Membuat objek Kucing dengan konstruktor baru
        Kucing kucing = new Kucing("Milo", 2, "Persia");
```

```
        System.out.println("--------------------");
        kucing.info();
    }
}
```

Percobaan 4

Kode program class Hewan

```
class Hewan {
    String nama;
    int umur;

    public Hewan(String nama, int umur) {
        this.nama = nama;
        this.umur = umur;
    }

    public void berjalan() {
        System.out.println(this.nama + " sedang berjalan...");
    }
}
```

Kode program class Mamalia

```
class Mamalia extends Hewan {
    public Mamalia(String nama, int umur) {
        super(nama, umur);
    }

    void menyusui() {
        System.out.println(this.nama + " menyusui anaknya...");
    }
}
```

Kode program class Kucing

```
class Kucing extends Mamalia {
    public Kucing(String nama, int umur) {
        super(nama, umur);
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Kucing kucing = new Kucing("Milo", 2);
        kucing.berjalan(); // Mewarisi dari Hewan
        kucing.menyusui(); // Mewarisi dari Mamalia
    }
}
```

Tugas Praktikum

Kode program class Anjing

```
class Anjing extends Mamalia {
    private String jenisAnjing;

    public Anjing(String nama, int umur, String jenisAnjing) {
        super(nama, umur);
        this.jenisAnjing = jenisAnjing;
    }

    public void menggonggong() {
        System.out.println(this.nama + " menggonggong: Guk! Guk!");
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Anjing anjing = new Anjing("Doggy", 3, "Golden Retriever");
        anjing.berjalan();   // Metode dari kelas Hewan
        anjing.menyusui(); // Metode dari kelas Mamalia
        anjing.menggonggong(); // Metode dari kelas Anjing itu
sendiri
    }
}
```

Percobaan 5

Kode program class Hewan

```
class Hewan {
    private String nama;
    protected int umur;

    public Hewan(String nama, int umur) {
        this.nama = nama;
        this.umur = umur;
    }

    public void berjalan() {
        System.out.println(nama + " berjalan...");
    }

    public void info() {
        System.out.println("Nama: " + nama);
        System.out.println("Umur: " + umur);
    }
}
```

Kode program class Kucing

```
class Kucing extends Hewan {
    private String warnaBulu;

    public Kucing(String nama, int umur, String warnaBulu) {
        super(nama, umur);
        this.warnaBulu = warnaBulu;
    }

    @Override
    public void info() {
        super.info(); // Menggunakan 'super' untuk mengakses metode
dari parent class
        System.out.println("Warna bulu: " + warnaBulu);
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Kucing kucing = new Kucing("Milo", 2, "Oranye");
        kucing.info();
        kucing.berjalan();
        // System.out.println(kucing.nama); // Error, 'nama' adalah
private
        System.out.println("Umur bisa diakses: " + kucing.umur); //
Valid, 'umur' adalah protected
    }
}
```

Tugas Praktikum

Kode program class Anjing

```
class Anjing extends Hewan {
    private String jenisBulu; // Atribut bersifat private

    public Anjing(String nama, int umur, String jenisBulu) {
        super(nama, umur);
        this.jenisBulu = jenisBulu;
    }

    // Metode public untuk mengakses atribut private
    public String getJenisBulu() {
        return this.jenisBulu;
    }

    @Override
    public void info() {
        super.info(); // Memanggil metode info() dari superclass
        System.out.println("Jenis Bulu: " + this.jenisBulu);
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Anjing anjing = new Anjing("Doggy", 5, "Kasar");
        anjing.info(); // Memanggil metode info() yang telah di-
override

        // Mengakses atribut private melalui metode public (getter)
        System.out.println("Jenis bulu anjing ini adalah " +
anjing.getJenisBulu());
    }
}
```

Percobaan 6

Kode program class Kendaraan

```
abstract class Kendaraan {
    protected String merk;
    protected int tahunProduksi;

    public Kendaraan(String merk, int tahunProduksi) {
        this.merk = merk;
        this.tahunProduksi = tahunProduksi;
    }

    abstract void jalankan(); // Abstract method, tanpa implementasi

    void info() {
        System.out.println("Merk: " + merk);
        System.out.println("Tahun produksi: " + tahunProduksi);
    }
}
```

Kode program class Mobil

```
class Mobil extends Kendaraan {
    int jumlahPintu;

    public Mobil(String merk, int tahunProduksi, int jumlahPintu) {
        super(merk, tahunProduksi);
        this.jumlahPintu = jumlahPintu;
    }

    @Override
    void jalankan() {
        System.out.println("Mobil " + merk + " berjalan...");
    }

    @Override
    void info() {
```

```
        super.info();
        System.out.println("Jumlah pintu: " + jumlahPintu);
    }
}
```

Kode program class Motor

```
class Motor extends Kendaraan {
    int kapasitasCC;

    public Motor(String merk, int tahunProduksi, int kapasitasCC) {
        super(merk, tahunProduksi);
        this.kapasitasCC = kapasitasCC;
    }

    @Override
    void jalankan() {
        System.out.println("Motor " + merk + " berjalan...");
    }

    @Override
    void info() {
        super.info();
        System.out.println("Kapasitas CC: " + kapasitasCC);
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Mobil mobil = new Mobil("Toyota", 2022, 4);
        Motor motor = new Motor("Honda", 2023, 150);

        mobil.info();
        mobil.jalankan();

        System.out.println("-----");

        motor.info();
        motor.jalankan();
    }
}
```

Tugas Praktikum

Kode program class Truk

```
class Truk extends Kendaraan {
    private double kapasitasMuatan; // dalam ton

    public Truk(String merk, int tahunProduksi, double
kapasitasMuatan) {
        super(merk, tahunProduksi);
        this.kapasitasMuatan = kapasitasMuatan;
    }

    @Override
    void jalankan() {
        System.out.println("Truk " + merk + " berjalan dengan
muatan...");
    }

    @Override
    void info() {
        super.info();
```

```
        System.out.println("Kapasitas Muatan: " + kapasitasMuatan +
" ton");
    }
}
```

Kode program class Main

```
public class Main {
    public static void main(String[] args) {
        Mobil mobil = new Mobil("Toyota", 2022, 4);
        Motor motor = new Motor("Honda", 2023, 150);
        Truk truk = new Truk("Hino", 2020, 10.5); // Objek Truk baru

        mobil.info();
        mobil.jalankan();
        System.out.println("-----");

        motor.info();
        motor.jalankan();
        System.out.println("-----");

        // Menjalankan metode pada objek Truk
        truk.info();
        truk.jalankan();
    }
}
```

Pertanyaan

1. Kelas Kucing memanfaatkan inheritance dengan menggunakan keyword extends Hewan. Ini berarti Kucing adalah sebuah Hewan (hubungan "is-a") dan secara otomatis mewarisi anggota non-private dari kelas Hewan.
   Atribut yang diwarisi: String nama.
   Metode yang diwarisi: void bersuara().

2. Method overriding adalah kemampuan subclass untuk menyediakan implementasi spesifik dari sebuah metode yang sudah ada di superclass-nya. Cara mengimplementasikannya adalah dengan mendeklarasikan kembali metode tersebut di subclass dengan nama, parameter, dan tipe data return yang sama, serta menambahkan anotasi @Override di atasnya. Contoh: Di latihan kedua, kelas Kucing meng-override metode bersuara() dari kelas Hewan. Implementasi di Hewan: System.out.println("Suara hewan..."); Implementasi di Kucing: System.out.println("Meong!");

3. Keyword super() harus digunakan di dalam konstruktor subclass karena konstruktor tidak diwariskan. super() berfungsi untuk memanggil konstruktor dari superclass guna menginisialisasi atribut-atribut yang diwariskan. Panggilan ini harus menjadi pernyataan pertama di dalam konstruktor subclass.
   Contoh: Dalam konstruktor Kucing(String nama, int umur), terdapat panggilan super(nama, umur);. Ini mengeksekusi konstruktor dari kelas Hewan untuk mengisi nilai atribut nama dan umur.

4. - Single Inheritance: Sebuah subclass hanya mewarisi dari satu superclass.
   Contoh: class Mamalia extends Hewan adalah single inheritance.
   - Multilevel Inheritance: Sebuah subclass mewarisi dari subclass lain, membentuk rantai pewarisan.
   Contoh: class Kucing extends Mamalia, di mana Mamalia sendiri adalah turunan dari Hewan. Ini membentuk rantai: Hewan → Mamalia → Kucing.

5. Access control mengatur visibilitas anggota kelas dalam pewarisan:
   - public: Bisa diakses dari mana saja, termasuk oleh subclass.
   - protected: Bisa diakses oleh subclass (bahkan di package berbeda) dan kelas lain dalam package yang sama. Contohnya adalah atribut umur di kelas Hewan.
   - private: Hanya bisa diakses di dalam kelas itu sendiri dan tidak bisa diakses langsung oleh subclass. Contohnya adalah atribut nama di kelas Hewan.

   Keyword super digunakan untuk mengakses anggota dari superclass. Contohnya, super.info(); di dalam metode info() milik Kucing digunakan untuk memanggil metode info() asli milik Hewan.

6. Kombinasi abstract class dan method overriding menciptakan struktur yang fleksibel dengan cara:
    a. abstract class Kendaraan: Berperan sebagai kerangka dasar (blueprint) yang tidak bisa dibuat objeknya secara langsung. Kelas ini mendefinisikan atribut dan metode umum yang pasti dimiliki oleh semua jenis kendaraan.
    b. abstract void jalankan(): Ini adalah metode kontrak tanpa implementasi. Kelas Kendaraan mewajibkan setiap subclass konkret (seperti Mobil, Motor, dan Truk) untuk menyediakan implementasi sendiri dari metode jalankan() melalui method overriding.

    Struktur ini fleksibel karena kita bisa dengan mudah menambahkan jenis kendaraan baru (misal: Bus) hanya dengan extend Kendaraan dan mengimplementasikan metode jalankan(), tanpa mengubah kode yang sudah ada.