Nama & NPM	Topik:	Tanggal:
Zaira ayu wandira G1F024055	**Kelas,Objek,Method**	**12 septemper 2024**

[No.1] Identifikasi Masalah:

1.1. Perbaiki pesan kesalahan Contoh 1!

1.2. Cermati contoh 1. susun kode menggunakan constructor dengan parameter data pribadi anda!

[No.1l] Analisis dan Argumentasi

1.1 Masalah:

1.Nama Constructor Salah: Di dalam kelas Manusia, nama constructor seharusnya adalah Manusia dan bukan Manusia1.

2.Parameter Constructor Tidak Sesuai: Constructor Manusia hanya memiliki satu parameter (nama), padahal di dalam metode main, mencoba mengirimkan dua parameter ("Putri", "hitam").

3.penggunaan Constructor yang Tidak Sesuai: Di dalam metode main, Anda menggunakan constructor yang tidak sesuai dengan yang telah dideklarasikan.

Perbaikan Kode:

1.Ubah Nama Constructor: Ubah nama constructor Manusia1 menjadi Manusia.: 2.Tambahkan parameter rambut ke dalam constructor untuk menangani dua atribut

3.Set Nilai Atribut: Dalam constructor, atur nilai atribut nama dan rambut

[No.1] Penyusunan Algoritma dan Kode Program

1.1 Algorima:

1).mulai

2).deklarasi kelas

3). Deklarasi atribut kelas

4).deklarasi konstruktor

5). Deklarasi method main

6).akhir

```java
public class Manusia {
    // Deklarasi atribut kelas
    String nama;
    String rambut;

    // Konstruktor dengan dua parameter
    public Manusia(String nama, String rambut) {
        // Menggunakan 'this' untuk merujuk pada atribut kelas
        this.nama = nama;
        this.rambut = rambut;
        System.out.println("Nama saya : " + nama +
            "\nWarna Rambut : " + rambut);
    }

    // Method utama
    public static void main(String[] args) {
        // Membuat objek dari kelas Manusia dengan parameter yang
            diberikan
        Manusia satu = new Manusia("Putri", "hitam");
    }
}
```

```
java -cp /tmp/EMXp7mVZlc/Manusia
Nama saya : Putri
Warna Rambut : hitam

=== Code Execution Successful ===
```

Dengan perbaikan tersebut, program sekarang berjalan dengan benar tanpa kesalahan kompilasi.

1.2 Algoritma

1).mulai

2).deklarasi kelas

3). Deklarasi atribut kelas

4).deklarasi konstruktor

5). Deklarasi method main

6).akhir

```java
public class Manusia {
    // Deklarasi atribut kelas
    String nama;
    String rambut;
    String hobi;
    int umur;

    // Konstruktor dengan dua parameter
    public Manusia(String nama, String rambut, String hobi, int umur) {
        // Menggunakan 'this' untuk merujuk pada atribut kelas
        this.nama = nama;
        this.rambut = rambut;
        this.hobi = hobi;
        this.umur = umur;
        System.out.println("Nama saya : " + nama +
            "\nWarna Rambut : " + rambut +  "\nhobi saya : " + hobi+ "\numur saya :
                " + umur);
    }

    // Method utama
    public static void main(String[] args) {
        // Membuat objek dari kelas Manusia dengan parameter yang diberikan
        Manusia satu = new Manusia("zaira ayu wandira", "hitam","memasak",18);
    }
}
```

```
java -cp /tmp/qJdD6oU4oN/Manusia
Nama saya : zaira ayu wandira
Warna Rambut : hitam
hobi saya : memasak
umur saya : 18

=== Code Execution Successful ===
```

Luaran sudah sesuai dengan program yang disusun

[No.1] Kesimpulan

Kode yang diberikan mendefinisikan sebuah kelas Manusia yang memiliki dua atribut: nama dan rambut. Tujuannya adalah untuk mendemonstrasikan penggunaan konstruktor dengan parameter dalam Java serta bagaimana menginisialisasi dan mencetak informasi dari objek kelas tersebut.

[No. 2] Identifikasi Masalah:

2.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!
2.2. Apabila nanti Anda akan memiliki keturunan, analisa sifat (atribut) dan constructor sebagai Ortu apa yang akan diturunkan (gunakan data karakter pribadi anda) ?
2.3. Rancanglah kode program untuk sifat (atribut) dan constructor overloaded dari Latihan 2.2!

[No.2] Analisis dan Argumentasi

2.1.Kode yang berikan memiliki beberapa masalah:

1. Nama Konstruktor Tidak Sesuai: Nama konstruktor harus sesuai dengan nama kelas. Dalam kode tersebut konstruktor diberi nama ortu, sedangkan nama kelasnya adalah Ortu.
2. Variabel Belum Dideklarasikan: Variabel nama dan rambut tidak dideklarasikan di dalam konstruktor.

Penjelasan Perbaikan:

1. Nama Konstruktor: Ubah nama konstruktor dari ortu menjadi Ortu untuk mencocokkan nama kelas.
2. Deklarasi Atribut: Tambahkan deklarasi atribut nama dan rambut di dalam kelas Ortu dan inisialisasi atribut tersebut di dalam konstruktor.

2.2. Analisis Sifat dan Konstruktor Keturunan

Jika memiliki keturunan dari kelas Ortu, bisa membuat kelas baru yang mewarisi Ortu. Misalkan memiliki data karakter pribadi seperti nama, rambut, umur, dan hobi. Kelas keturunan bisa disebut Anak. Berikut adalah bagaimana bisa menganalisis dan mendesain kelas keturunan:

- **Atribut dari Kelas Ortu:**
  - nama (tipe String)
  - rambut (tipe String)
- **Atribut Tambahan di Kelas Anak:**
  - umur (tipe int)
  - hobi (tipe String)
- **Konstruktor di Kelas Anak:**

- Konstruktor ini akan memanggil konstruktor kelas Ortu untuk menginisialisasi nama dan rambut.
- Konstruktor ini juga akan menginisialisasi atribut tambahan umur dan hobi.

[No.2] Penyusunan Algoritma dan Kode Program

2.1.Algoritma:

1) Mulai
2) Definisikan Kelas
3) Deklarasi Atribut:
4) Definisikan Konstruktor Ortu:
5) Definisikan Metode main:
6) Selesai:

2.1.code program

```
1  public class Ortu {
2      // Deklarasi atribut
3      String nama;
4      String rambut;
5
6      // Konstruktor dengan parameter
7      public Ortu(String nama, String rambut) {
8          this.nama = nama;        // Inisialisasi atribut nama
9          this.rambut = rambut;    // Inisialisasi atribut rambut
10         System.out.println("Nama saya: " + nama +
11                            "\nWarna Rambut: " + rambut);
12     }
13
14     public static void main(String[] args) {
15         // Membuat objek Ortu dengan parameter
16         Ortu satu = new Ortu("Putri", "hitam");
17     }
18 }
19
```

```
java -cp /tmp/dgceFhiKB8/Ortu
Nama saya: Putri
Warna Rambut: hitam

=== Code Execution Successful ===
```

Luaran sudah sesuai dengan program yang disusun

2.3.Algoritma:

- Deklarasi Kelas Ortu
- Deklarasi Atribut
- Constructor Pertama (Lengkap)
- Constructor Kedua (Tanpa Hobi)
- Constructor Ketiga (Partial)
- Metode tampilkanInformasi
- Metode main (Entry Point)

2.3.code program

```java
public class Ortu {
  // Deklarasi atribut
  private String nama;
  private String warnaRambut;
  private int umur;
  private String hobi;
  // Constructor pertama (lengkap)
  public Ortu(String nama, String warnaRambut, int umur, String hobi) {
  this.nama = nama;
  this.warnaRambut = warnaRambut;
  this.umur = umur;
  this.hobi = hobi;
  System.out.println("Constructor 1 (lengkap)");
  tampilkanInformasi();
  }
  // Constructor kedua (tanpa hobi, default nilai)
  public Ortu(String nama, String warnaRambut,int umur) {
  this(nama, warnaRambut, umur, "Tidak Diketahui");
  System.out.println("Constructor 2 (Tanpa hobi)");
  }
  // Constructor ketiga (nama dan warna rambut saja, default nilai)
  public Ortu(String nama, String warnaRambut) {
  this(nama, warnaRambut, 0, "Tidak Diketahui");
  System.out.println("Constructor 3 (Partial)");
  }
  // Method untuk menampilkan informasi
  private void tampilkanInformasi() {
  System.out.println("Nama: " + nama);
  System.out.println("Warna Rambut: " + warnaRambut);
  System.out.println("Tumur: " + umur + "tahun");
  System.out.println("hobi: " + hobi);}
  // Main method sebagai entry point
  public static void main(String[] args) {
  // Membuat objek dengan constructor lengkap
  Ortu ortu1 = new Ortu("zaira", "Hitam", 18, "menonton");
  // Membuat objek dengan constructor tanpa hobi
  Ortu ortu2 = new Ortu("ayu", "Hitam", 19);
  // Membuat objek dengan constructor hanya nama dan warna rambut
  Ortu ortu3 = new Ortu("wandira", "hitam"); }}
```

```
java -cp /tmp/LwdNt7C8hc/Ortu
Constructor 1 (lengkap)
Nama: zaira
Warna Rambut: Hitam
Tumur: 18tahun
hobi: menonton
Constructor 1 (lengkap)
Nama: ayu
Warna Rambut: Hitam
Tumur: 19tahun
hobi: Tidak Diketahui
Constructor 2 (Tanpa hobi)
Constructor 1 (lengkap)
Nama: wandira
Warna Rambut: hitam
Tumur: 0tahun
hobi: Tidak Diketahui
Constructor 3 (Partial)

=== Code Execution Successful ===
```

[No.2] Kesimpulan

Sebagai kesimpulan, dalam Java, kita dapat menggunakan constructor overloading untuk mendefinisikan beberapa konstruktor dengan parameter yang berbeda dalam satu kelas. Hal ini memungkinkan kita untuk menginisialisasi objek dengan berbagai metode sesuai kebutuhan. Kode di atas menggambarkan penerapan berbagai konstruktor untuk mengatur atribut objek dan menampilkan informasi yang relevan.

[No. 3] Identifikasi Masalah:

3.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!
3.2. Berdasarkan Latihan 2.2. Anda sudah punya kode program untuk atribut dan constructor sebagai Ortu.

Kembangkanlah kode program untuk method dari Ortu dengan data perilaku pribadi Anda yang menggunakan:

a) method overloading,
b) method dengan return value
c) method tanpa return value

[No.3] Analisis dan Argumentasi

3.1 **Masalah dalam Kode:**

1. **Nama Konstruktor yang Salah**:
   - Konstruktor dideklarasikan sebagai Manusia1 padahal harusnya Manusia.
2. **Sintaks Method yang Salah**:
   - Definisi method sukaNonton tidak memiliki tanda kurung () dan tidak dapat dideklarasikan tanpa tipe data kembali (return type).
   - Method sukaNonton seharusnya memiliki tipe data kembali dan tanda kurung, serta tidak dapat memiliki dua method dengan nama yang sama jika tidak memiliki parameter berbeda.

Perbaikan:

1. Perbaiki nama konstruktor.
2. Tambahkan tanda kurung () untuk deklarasi method.

3. Sesuaikan tipe data kembalian untuk method.

3.2 Untuk kelas Ortu, mari kita tambahkan beberapa method dengan berbagai tipe data kembalian dan juga method overloading. Ini akan mencakup:

- **Method Overloading**: Menyediakan beberapa versi dari method dengan nama yang sama namun parameter yang berbeda.
- **Method dengan Return Value**: Method yang mengembalikan hasil dari perhitungan atau operasi.
- **Method Tanpa Return Value**: Method yang tidak mengembalikan hasil tetapi hanya melakukan operasi.

[No.3] Penyusunan Algoritma dan Kode Program

3.1 Algoritma:

1) mulai
2) Deklarasi Kelas Manusia
3) Konstruktor
4) Metode sukaNonton(String film)
5) Metode main
6) Selesai

3**.1 code program:**

```
public class Manusia {
    // Deklarasi atribut Manusia
    String nama, rambut;

    // Konstruktor dengan parameter
    public Manusia(String nama, String rambut) {
        this.nama = nama;
        this.rambut = rambut;
        System.out.println("Nama saya: " + nama +
                           "\nWarna Rambut: " + rambut);
    }

    // Method tanpa return value, dengan parameter
    void sukaNonton(String film) {
        System.out.println("Hobi Menonton: " + film);
    }

    // Method dengan return value
    int sukaNonton(int episode, int durasi) {
        return episode * durasi;
    }

    // Method utama
    public static void main(String[] args) {
        Manusia satu = new Manusia("Putri", "hitam");
        satu.sukaNonton("Drakor"); // Memanggil method tanpa return value
        int jumlahJam = satu.sukaNonton(2, 2); // Memanggil method dengan return value
        System.out.println("Jam nonton = " + jumlahJam + " jam");
    }
}
```

```
java -cp /tmp/vr0Mevv8Se/Manusia
Nama saya: Putri
Warna Rambut: hitam
Hobi Menonton: Drakor
Jam nonton = 4 jam

=== Code Execution Successful ===
```

Luaran sudah sesuai dengan program yang disusun.

3.2 algoritma

1. **Mulai**
2. **Definisikan Kelas**
3. **Definisikan Constructor**
4. **Definisikan Metode**
5. **Metode main (Entry Point)**
6. **Selesai**

3.2 code program

```java
public class Ortu {
    // Deklarasi atribut
    private String nama;
    private String warnaRambut;
    private double tinggiBadan;
    private String hobi;
    // Constructor pertama (lengkap)
    public Ortu(String nama, String warnaRambut, double tinggiBadan, String hobi) {
        this.nama = nama;
        this.warnaRambut = warnaRambut;
        this.tinggiBadan = tinggiBadan;
        this.hobi = hobi;
        System.out.println("Constructor 1 (lengkap)");
        tampilkanInformasi(); }
    // Constructor kedua (tanpa hobi, default nilai)
    public Ortu(String nama, String warnaRambut, double tinggiBadan) {
        this(nama, warnaRambut, tinggiBadan, "Tidak Diketahui");
        System.out.println("Constructor 2 (Tanpa hobi)"); }
    // Constructor ketiga (nama dan warna rambut saja, default nilai)
    public Ortu(String nama, String warnaRambut) {
        this(nama, warnaRambut, 0.0, "Tidak Diketahui");
        System.out.println("Constructor 3 (Partial)"); }
    // Method untuk menampilkan informasi
    private void tampilkanInformasi() {
        System.out.println("Nama: " + nama);
        System.out.println("Warna Rambut: " + warnaRambut);
        System.out.println("Tinggi Badan: " + tinggiBadan + " cm");
        System.out.println("Hobi: " + hobi); }
    // Method overloading: berbicara dengan satu parameter
    public void berbicara(String pesan) {
        System.out.println("Pesan: " + pesan); }
    // Method overloading: berbicara dengan dua parameter
    public void berbicara(String pesan, int volume) {
        System.out.println("Pesan: " + pesan + " dengan volume " + volume);  }
    // Method dengan return value: menghitung tinggi badan ideal
    public double hitungTinggiBadanIdeal(double usia) {
        double tinggiIdeal = 0.0;
        if (usia < 18) {
            tinggiIdeal = 150 + (usia * 2.5);
        } else {
            tinggiIdeal = 170 + (usia - 18) * 1.2; }
        return tinggiIdeal; }
    // Method tanpa return value: memberikan motivasi
    public void berikanMotivasi() {
        System.out.println("Mahasiswa Sistem Informasi, teruslah berusaha!"); }
    // Main method sebagai entry point
    public static void main(String[] args) {
        // Membuat objek dengan constructor lengkap
        Ortu ortu1 = new Ortu("zaira", "Hitam", 154.0, "Tinggi");
        // Membuat objek dengan constructor tanpa hobi
        Ortu ortu2 = new Ortu("ayu", "Hitam", 157.0);
        // Membuat objek dengan constructor hanya nama dan warna rambut
        Ortu ortu3 = new Ortu("wandira", "Hitam");
        // Menggunakan method overloading
        ortu1.berbicara("Selamat pagi!");
        ortu1.berbicara("Selamat pagi!", 5);
        // Menggunakan method dengan return value
        double tinggiIdeal = ortu1.hitungTinggiBadanIdeal(18);
        System.out.println("Tinggi badan ideal untuk usia 18 tahun: " + tinggiIdeal + " cm");
```

```
java -cp /tmp/nrKwjZvFin/Ortu
Constructor 1 (lengkap)
Nama: zaira
Warna Rambut: Hitam
Tinggi Badan: 154.0 cm
Hobi: Tinggi
Constructor 1 (lengkap)
Nama: ayu
Warna Rambut: Hitam
Tinggi Badan: 157.0 cm
Hobi: Tidak Diketahui
Constructor 2 (Tanpa hobi)
Constructor 1 (lengkap)
Nama: wandira
Warna Rambut: Hitam
Tinggi Badan: 0.0 cm
Hobi: Tidak Diketahui
Constructor 3 (Partial)
Pesan: Selamat pagi!
Pesan: Selamat pagi! dengan volume 5
Tinggi badan ideal untuk usia 18 tahun: 170.0 cm
Mahasiswa Sistem Informasi, teruslah berusaha!

--- Code Execution Successful ---
```

Luaran sudah sesuai dengan program yang disusun.

[No.3] Kesimpulan

Kesimpulannya, kode kelas Ortu mengilustrasikan penggunaan berbagai teknik pemrograman Java, seperti method overloading untuk meningkatkan fleksibilitas fungsi, method yang mengembalikan nilai untuk perhitungan, dan method yang tidak mengembalikan nilai untuk melakukan aksi sederhana. Hal ini menghasilkan struktur yang jelas dan fungsionalitas yang teratur.

[No. 4] Identifikasi Masalah:

4.1. Evaluasi method yang dimiliki Contoh 4 pada class Anak extends Ortu dengan method di class ortu. Simpulkan hasil evaluasi Anda agar method ini menjadi efisien!
4.2.Setelah dirunning di JDoodle, catat waktu eksekusinya.
Susun kembali kode program yang dapat mengefisienkan waktu eksekusi!

[No.4] Analisis dan Argumentasi

4.1 Evaluasi Method pada Kelas Anak

Di dalam kode Anda, kelas Anak memiliki beberapa method yang di-overload dan di-override dari kelas Ortu. Mari kita tinjau satu per satu:

1. **Method yang di-override dari Ortu:**
    - void sukaMenonton(String a) di kelas Anak adalah method yang di-override dari kelas Ortu. Ini memungkinkan kelas Anak untuk mengganti implementasi method yang ada di kelas Ortu.
    - void sukaMembaca(String a) di kelas Anak juga di-override dari kelas Ortu. Ini memberikan kemampuan bagi kelas Anak untuk mengganti implementasi method yang ada di kelas Ortu.
2. **Method yang di-overload di kelas Anak:**
    - void sukaMenonton(int a, String b) di kelas Anak adalah overload dari method sukaMenonton di kelas Ortu. Ini merupakan method baru yang tidak ada di kelas Ortu.

Kesimpulan dan Rekomendasi untuk Efisiensi

1. **Penggunaan Overriding dan Overloading:**
    - Pastikan hanya method yang memang perlu diubah implementasinya yang di-override. Jika method di Ortu tidak perlu diubah, jangan override.
    - Gunakan overload jika Anda ingin menambah variasi penggunaan method, seperti yang dilakukan dengan sukaMenonton(int a, String b).
2. **Penerapan Polimorfisme:**
    - Jika Anda menggunakan method yang di-override, pastikan Anda benar-benar memerlukan perubahan pada implementasinya. Implementasi baru harus sesuai dengan kebutuhan spesifik dari kelas anak.

[No.4] Penyusunan Algoritma dan Kode Program

Algorima:

1. **mulai**
2. **Inisialisasi Program**
3. **Deklarasi Kelas Ortu**
4. **Instansiasi Objek Ortu**
5. **Cetak ke layar:**
6. **Deklarasi Kelas Anak**
7. **Instansiasi Objek Anak**
8. **Selesai**

Code program:

```java
public class Ortu {
    String nama, rambut;

    void sukaMenonton(String a) {
        System.out.println("Nonton " + a);
    }

    void sukaMembaca(String a) {
        System.out.println("Suka Baca " + a);
    }

    public static void main(String[] args) {
        System.out.println("Sifat Orang Tua :");
        Ortu objekO = new Ortu();
        objekO.sukaMenonton("Berita");
        objekO.sukaMembaca("Koran");

        System.out.println("\nSifat Anak :");
        Anak objekA = new Anak();
        objekA.sukaMenonton(9, "Film Drakor"); // Ini akan memanggil metode overload
        objekA.sukaMembaca("Komik One Piece");
    }
}

class Anak extends Ortu {
    // Method ini adalah overload, bukan override
    void sukaMenonton(int a, String b) {
        System.out.println("Nonton Jam " + a + " Malam " + b);
    }
```

```
java -cp /tmp/vj6fmBzOMX/Ortu
Sifat Orang Tua :
Nonton Berita
Suka Baca Koran

Sifat Anak :
Nonton Jam 9 Malam Film Drakor
Suka Baca Komik One Piece

=== Code Execution Successful ===
```

Luaran sudah sesuai dengan program yang disusun

[No.4] Kesimpulan

Kesimpulannya, metode dalam kelas `Anak` menggunakan overloading untuk menambahkan fungsionalitas baru (sukaMenonton(int a, String b)) dan overriding untuk menyesuaikan fungsionalitas yang diwarisi (sukaMenonton(String a)). Metode sukaMembaca(String a) di kelas `Anak` mengulang metode dari kelas `Ortu` tanpa adanya perubahan, yang mungkin tidak diperlukan jika tidak ada modifikasi yang dilakukan..

Refleksi:pada pembelajaran ini saya belajar bahwa konstruktor adalah metode khusus yang digunakan untuk menginisialisasi objek. Proses memahami perbedaan antara method overloading dan overriding memberi saya wawasan tentang fleksibilitas dalam mendesain kelas, Proses evaluasi dan perbaikan kode memberi saya pengalaman berharga dalam pemecahan masalah, Mengintegrasikan data karakter pribadi saya ke dalam kode program memberikan pengalaman yang menyenangkan dan kreatif.