

Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
Syifa Ariqah Pajriyanti (G1F024009)	Kelas Java	14 september 2024

[Nomor 1] Identifikasi Masalah:

1) Uraikan permasalahan dan variable

```
classKelasku{  
    //deklarasi field,konstruktor dan method  
}
```

contoh 1:

```
public class Manusia { // deklarasi kelas  
    //deklarasi atribut Manusia dalam variabel  
    String nama, rambut;  
  
    //deklarasi constructor  
    public Manusia1 (String nama) {  
        System.out.println(" Nama saya : "+ nama +  
            "\n Warna Rambut : " + rambut);  
    }  
  
    //deklarasi method utama  
    public static void main( String[] args) {  
        Manusia1 satu = new Manusia1("Putri", "hitam");  
    }  
}
```

Luaran 1:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:  
The constructor Manusia1(String, String) is undefined  
at Manusia1.main(Manusia1.java:13)
```

1.1 Perbaiki pesan kesalahan Contoh 1!

```
public class Manusia { // deklarasi kelas  
  
    String nama, rambut;  
  
    //deklarasi constructor  
  
    public Manusia (String nama, String rambut) {  
  
        System.out.println(" Nama saya : "+ nama +  
  
            "\n Warna Rambut : " + rambut);  
  
    }  
  
    public static void main( String[] args) {  
  
        Manusia satu = new Manusia("Putri", "hitam");  
  
    }  
  
}
```

2.1 Analisa ciri-ciri lain Kelas Manusia yang dapat menjadi:

a. atribut variabel,:

1. String nama: atribut ini digunakan untuk menyimpan nama orang.
2. String rambut: atribut ini digunakan untuk menyimpan warna rambut orang.

b. perilaku/ behavior!

Perilaku atau behavior dalam sebuah kelas adalah cara kelas berfungsi atau beroperasi melalui metode-metodenya.

1. Public manusia(String nama, String rambut) : ini adalah konstruktor dari kelas manusia.konstruktor ini menerima dua parameter (nama dan rambut) dan mencetak informasi tentang nama dan warna rambut ke konsol. Ini memungkinkan objek manusia untuk diinisialisasi dengan nilai-nilai tertentu saat dibuat.
2. Public static void main(String[] args) : ini adalah metode main, yang merupakan titik masuk dari program java. Metode main membuat sebuah objek manusia dengan nama "putri" dan warna rambut "hitam".

2) Rincikan sumber informasi yang relevan (buku / webpage)

- **Video Materi 1 tentang Kelas, Objek, Method –**
<https://www.youtube.com/watch?v=60ldOc8m8Es>
- **Video Materi 2 tentang –** <https://www.youtube.com/watch?v=6qULMlcv-eg>

[Nomor 1] Analisis dan Argumentasi

- 1) Uraikan rancangan solusi yang diusulkan.
Saya mengusulkan permasalahan ini dapat diatasi dengan cara meningkatkan dan memperluas kelas *Manusia* yang telah diberikan, berikut adalah rancangan solusi yang diusulkan. Rancangan ini mencakup peningkatan fitur, penambahan atribut dan metode, serta implementasi prinsip-prinsip pemrograman berorientasi objek (OOP) untuk membuat kelas *Manusia* lebih fungsional dan terstruktur.
- 2) Analisis solusi, kaitkan dengan permasalahan.
 1. Penambahan Atribut Instance : Agar kelas Manusia lebih representatif, kita dapat menambahkan beberapa atribut instance tambahan yang dapat menggambarkan lebih banyak informasi tentang seorang manusia:
 - int umur: Untuk menyimpan umur manusia.
 - String jenisKelamin: Untuk menyimpan jenis kelamin manusia.
 - String alamat: Untuk menyimpan alamat tempat tinggal manusia.
 2. Penyimpanan dan Penggunaan Atribut : Ubah konstruktor untuk menginisialisasi semua atribut instance dan buat metode untuk mengakses (getter) dan mengubah (setter) atribut tersebut. Ini membantu dalam menjaga enkapsulasi data dan memberikan kontrol lebih besar terhadap data yang dimiliki objek.
 3. Penambahan Metode : Tambahkan metode untuk mencetak informasi lengkap tentang manusia dan metode lain yang mungkin relevan.
 4. Menggunakan Prinsip OOP : Pastikan untuk menggunakan prinsip OOP seperti enkapsulasi dengan menyembunyikan data melalui akses kontrol dan memberikan akses melalui metode getter dan setter.

[Nomor 1] Penyusunan Algoritma dan Kode Program

- 1) Rancang desain solusi atau algoritma.

Algoritmanya :

- Deklarasi kelas
- Deklarasi atribut instance
- Deklarasi konstruktor
- Deklarasi metode tambahan
- Implementasi metode *main*

2) Tuliskan kode program dan luaran

a) Beri komentar pada kode

1. Deklarasi kelas : Ini mendeklarasikan kelas `Manusia`. Di Java, semua kode harus berada di dalam kelas.
2. Deklarasi atribut : Atribut `nama` dan `rambut` digunakan untuk menyimpan data tentang objek `Manusia`. Ini adalah variabel yang menjadi bagian dari objek.
3. Deklarasi konstruktor: Constructor ini digunakan untuk menginisialisasi objek `Manusia`. Ketika Anda membuat objek baru, constructor ini dipanggil dengan parameter yang diberikan (dalam hal ini `nama` dan `rambut`). Di sini, constructor hanya mencetak informasi ke konsol.
4. Method utama : Ini adalah method `main` yang merupakan titik masuk program Java. Ketika program dijalankan, metode ini akan dieksekusi. Dalam metode ini, sebuah objek `Manusia` baru dengan nama "Putri" dan rambut "hitam" dibuat, yang secara otomatis memanggil constructor dan mencetak informasi ke konsol.

b) Uraikan luaran yang dihasilkan

Nama saya : Putri

Warna Rambut : hitam

c) Screenshot/ Capture potongan kode dan hasil luaran

The screenshot shows an IDE with a Java file named `Manusia.java`. The code defines a class `Manusia` with two attributes, `nama` and `rambut`, a constructor that prints the name and hair color, and a `main` method that creates a `Manusia` object with the name "Putri" and hair color "hitam". The output window on the right shows the result of running the program: "Nama saya : Putri" and "Warna Rambut : hitam".

```
1 public class Manusia { // deklarasi kelas
2     //deklarasi atribut Manusia dalam variabel
3     String nama, rambut;
4
5     //deklarasi constructor
6     public Manusia (String nama, String rambut) {
7         System.out.println(" Nama saya : "+ nama +
8             "\n Warna Rambut : " + rambut);
9     }
10
11     //deklarasi method utama
12     public static void main( String[] args) {
13         Manusia satu = new Manusia("Putri", "hitam");
14     }
15 }
16
17
```

Input/Output

Language Version

Input Arguments

Stdin Inputs

Output Generated Files

Nama saya : Putri
Warna Rambut : hitam

[Nomor 1] Kesimpulan

1) Evaluasi

a) Apa konsekuensi dari skenario pemrograman ini?

- Enkapsulasi dan kontrol akses: Kode ini tidak menerapkan enkapsulasi dengan baik karena atribut dideklarasikan secara default public. Untuk praktik yang lebih baik, gunakan private untuk atribut dan tambahkan getter dan setter.
- Output dalam constructor: Mencetak informasi dalam constructor bukanlah praktik umum; biasanya constructor hanya digunakan untuk inisialisasi objek.
- Fleksibilitas dan kontrol: Kode ini tidak memberikan fleksibilitas untuk validasi atau kontrol yang lebih baik terhadap data.

Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
Syifa Ariqah Pajriyanti (G1F024009)	Kelas Java	14 september 2024

[Nomor 2] Identifikasi Masalah:

1. Uraikan permasalahan dan variable

```
public class Ortu {  
    //deklarasi constructor (variabel constructor)  
    public Ortu {  
        //nama dan rambut adalah variabel constructor  
        System.out.println(" Nama saya : "+ nama +  
            "\n Warna Rambut : " + rambut);  
    }  
    public static void main (String[] args) {  
        Ortu satu = new Ortu("Putri", "hitam");  
    }  
}
```

Luaran 2:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:  
    The constructor Ortu(String, String) is undefined  
    at Ortu.main(Ortu.java:9)
```

- 2.1 Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

```
public class Ortu {  
  
    String nama, rambut;  
  
    public Ortu (String nama, String rambut) {  
  
        //nama dan rambut adalah variabel constructor  
  
        System.out.println(" Nama saya : "+ nama +  
            "\n Warna Rambut : " + rambut);  
    }  
  
    public static void main (String[] args) {  
  
        Ortu satu = new Ortu("Putri", "hitam");  
  
    }  
}
```

- 2.3 Apabila Ortu memiliki data variabel umur = 25 dan jenis kelamin = P (untuk Perempuan), rekomendasikan constructor dengan parameter yang baru untuk ditambahkan dalam program!

Constructor ini sekarang menerima parameter untuk semua atribut baru (umur dan jenis kelamin), serta yang lama (nama dan rambut). Menginisialisasi atribut dengan nilai yang diberikan dan mencetak informasi lengkap ke konsol.

- 3) Rincikan sumber informasi yang relevan (buku / webpage)
 - **Video Materi 1 tentang Kelas, Objek, Method –**
<https://www.youtube.com/watch?v=60ldOc8m8Es>
 - **Video Materi 2 tentang –** <https://www.youtube.com/watch?v=6qULMlcV-eg>

[Nomor 2] Analisis dan Argumentasi

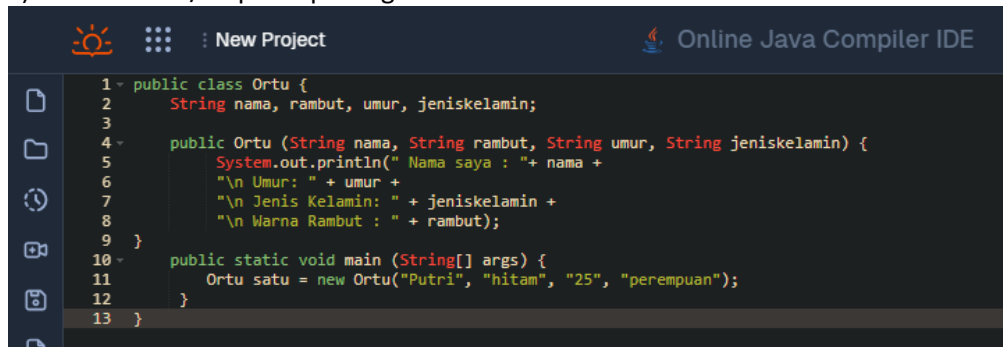
1. Uraikan rancangan solusi yang diusulkan.
 - Keterbacaan dan Pemeliharaan: Dengan memisahkan logika pencetakan dari konstruktor, kode lebih bersih dan lebih mudah dipelihara. Anda bisa dengan mudah mengubah cara informasi ditampilkan tanpa mempengaruhi cara data diinisialisasi.
 - Modularitas: Metode `printDetails()` membuat kelas `Ortu` lebih modular dan memungkinkan perubahan pada tampilan informasi tanpa mempengaruhi bagian lain dari kode.
 - Fleksibilitas: Dengan pendekatan ini, Anda bisa dengan mudah menambahkan metode lain untuk menangani data atau mengubah format pencetakan tanpa mengubah konstruktor.
2. Analisis solusi, kaitkan dengan permasalahan.
 - Pengelolaan Data: Solusi yang diusulkan menyelesaikan masalah pengelolaan data dengan memanfaatkan variabel instansi dan konstruktor yang sesuai. Ini memastikan bahwa setiap objek `Ortu` memiliki data yang lengkap.
 - Modularitas: Memisahkan logika pencetakan ke dalam metode terpisah meningkatkan modularitas dan mempermudah pemeliharaan. Ini mematuhi prinsip desain perangkat lunak yang baik, yaitu *single responsibility principle* (SRP).
 - Keterbacaan: Dengan memisahkan tanggung jawab dan memastikan kode yang bersih, solusi ini meningkatkan keterbacaan dan mempermudah pengembangan dan pemeliharaan kode di masa depan.

[Nomor 2] Penyusunan Algoritma dan Kode Program

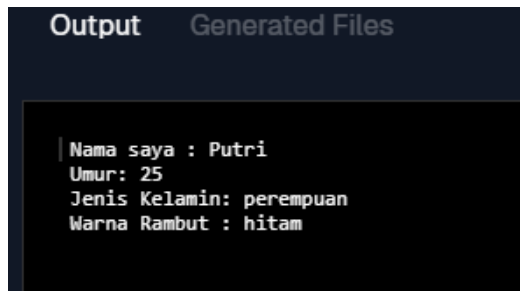
1. Rancang desain solusi atau algoritma.
 Algoritamanya :
 - Deklarasi kelas
 - Deklarasi atribut instance
 - Deklarasi konstruktor
 - Deklarasi metode `PrintDetails()`
 - Implementasi metode *main*
 - Selesai
2. Tuliskan kode program dan luaran
 - a) Beri komentar pada kode
 - Variabel Instansi: Menyimpan informasi tentang individu (nama, rambut, umur, jenis kelamin).
 - Konstruktor: Menginisialisasi variabel instansi dengan data yang diberikan saat objek dibuat.
 - Metode `printDetails`: Mencetak informasi individu ke konsol dalam format yang jelas.
 - Metode `main`: Titik masuk program, membuat objek `Ortu` dan memanggil `printDetails` untuk menampilkan informasi.
 - b) Uraikan luaran yang dihasilkan

Nama saya : Putri
 Umur : 25
 Jenis Kelamin : perempuan
 Warna Rambut : hitam

c) Screenshot/ Capture potongan kode dan hasil luaran



```
1 public class Ortu {  
2     String nama, rambut, umur, jeniskelamin;  
3  
4     public Ortu (String nama, String rambut, String umur, String jeniskelamin) {  
5         System.out.println(" Nama saya : "+ nama +  
6             "\n Umur: " + umur +  
7             "\n Jenis Kelamin: " + jeniskelamin +  
8             "\n Warna Rambut : " + rambut);  
9     }  
10  
11     public static void main (String[] args) {  
12         Ortu satu = new Ortu("Putri", "hitam", "25", "perempuan");  
13     }  
14 }
```



```
Output  
Generated Files  
  
Nama saya : Putri  
Umur: 25  
Jenis Kelamin: perempuan  
Warna Rambut : hitam
```

[Nomor 2] Kesimpulan

2) Evaluasi

2. Apa konsekuensi dari skenario pemrograman ini?

Konsekuensi dari skenario pemrograman ini umumnya positif dalam hal modularitas, keterbacaan, dan pemeliharaan, namun juga ada beberapa potensi masalah yang perlu diperhatikan, seperti fleksibilitas untuk pencetakan, perubahan data di masa depan, dan kinerja. Dengan pendekatan yang tepat, masalah ini bisa diatasi atau diminimalkan, dan keuntungan dari desain yang baik dapat dimaksimalkan.

Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
Syifa Ariqah Pajriyanti (G1F024009)	Kelas Java	15 september 2024

[Nomor 3] Identifikasi Masalah:

1. Uraikan permasalahan dan variable

```
public class Manusia {  
    //deklarasi atribut Manusia dalam variabel  
    String nama, rambut;  
  
    //deklarasi constructor  
    public Manusia1(String nama, String rambut) {  
        System.out.println(" Nama saya : "+ nama +  
            "\n Warna Rambut : " + rambut);  
    }  
  
    //deklarasi method  
    void sukaNonton {  
        System.out.println(" Hobi Menonton : " + film);  
    }  
  
    int sukaNonton {  
        episode*durasi;  
    }  
  
    //deklarasi method utama  
    public static void main( String[] args) {  
        Manusia satu = new Manusia("Putri", "hitam");  
        satu.sukaNonton("Drakor");  
        int jumlahJam = satu.sukaNonton(2, 2);  
        System.out.println("Jam nonton = " +jumlahJam + " jam");  
    }  
}
```

- 3.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

```
public class Manusia {  
    //deklarasi atribut Manusia dalam variabel  
    String nama, rambut;  
  
    //deklarasi constructor  
    public Manusia(String nama, String rambut) {  
        System.out.println(" Nama saya : "+ nama +  
            "\n Warna Rambut : " + rambut);  
    }  
  
    //deklarasi method  
    void sukaNonton (String film) {  
        System.out.println(" Hobi Menonton : " + film);  
    }  
  
    int sukaNonton (int episode, int durasi) {  
        return episode * durasi;  
    }  
  
    //deklarasi method utama  
    public static void main( String[] args) {  
        Manusia satu = new Manusia("Putri", "hitam");  
        satu.sukaNonton("Drakor");  
    }  
}
```

```

        int jumlahJam = satu.sukaNonton(2, 2);
        System.out.println("Jam nonton = " + jumlahJam + " jam");
    }
}

```

3.2. Ubahlah method dan constructor Contoh 3 sesuai dengan perilaku/ behavior anda

```

    public class Manusia {
        //deklarasi atribut Manusia dalam variabel
        String nama, rambut;
        char jenisKelamin;

        //deklarasi constructor
        public Manusia(String nama, String rambut, char jenisKelamin) {
            System.out.println(" Nama saya : "+ nama +
                "\n Warna Rambut : " + rambut + "\n Jenis Kelamin : " + jenisKelamin);
        }

        //deklarasi method
        void sukamakan (String makan) {
            System.out.println(" Suka Makan : " + makan);
        }

        int sukaMakan (int porsi, int hari) {
            return porsi * hari;
        }

        //deklarasi method utama
        public static void main (String[] args) {
            Manusia satu = new Manusia("Syifa", "hitam", 'p');
            satu.sukamakan("Mie Ayam");
            int jumlahPorsi = satu.sukaMakan(3, 7);
            System.out.println("Porsi yang dimakan perminggu = " + jumlahPorsi + " porsi");
        }
    }
}

```

3.3. Berdasarkan Contoh 3 dan Latihan 3.2. simpulkan perbedaan:

a) constructor overloading dan overriding

Constructor overloading terjadi ketika sebuah kelas memiliki lebih dari satu konstruktor dengan nama yang sama tetapi dengan parameter yang berbeda. Sedangkan Constructor overriding tidak berlaku dalam Java karena konstruktor tidak dapat diwarisi. Konstruktor hanya dapat di-overload di dalam kelas yang sama.

b) method overloading, dan method overriding

Method overloading terjadi ketika sebuah kelas memiliki lebih dari satu metode dengan nama yang sama tetapi dengan parameter yang berbeda (jumlah, tipe, atau urutan parameter). Sedangkan Method overriding terjadi ketika sebuah kelas turunan menyediakan implementasi spesifik untuk metode yang sudah didefinisikan di kelas induk dengan nama yang sama dan parameter yang sama.

c) method yang mengembalikan nilai dan method tidak mengembalikan nilai

Metode yang mengembalikan nilai memiliki tipe data yang dikembalikan, dan biasanya diakhiri dengan return untuk mengembalikan hasil dari perhitungan atau operasi. Sedangkan Metode yang tidak mengembalikan nilai memiliki tipe void dan tidak menggunakan return untuk mengembalikan hasil.

2. Rincikan sumber informasi yang relevan (buku / webpage)

- **Video Materi 1 tentang Kelas, Objek, Method –**
<https://www.youtube.com/watch?v=60ldOc8m8Es>
- **Video Materi 2 tentang –** <https://www.youtube.com/watch?v=6qULMlcv-eg>

[Nomor 3] Analisis dan Argumentasi

1. Uraikan rancangan solusi yang diusulkan.
 - Kelas Manusia:
 - Atribut: nama (String), rambut (String), dan jenisKelamin (char).
 - Konstruktor: Mencetak informasi nama, warna rambut, dan jenis kelamin saat objek dibuat, tapi tidak menyimpan nilai ke atribut.
 - Metode sukaMakan:
 - Void sukaMakan(String makan): Mencetak makanan yang disukai.
 - Int sukaMakan(int porsi, int hari): Mengembalikan total porsi makanan dalam seminggu (porsi × hari).
 - Metode main:
 - Membuat objek Manusia dan mencetak informasi.
 - Menggunakan kedua metode sukaMakan untuk mencetak makanan yang disukai dan menghitung total porsi makanan dalam seminggu.
2. Analisis solusi, kaitkan dengan permasalahan.

Solusi saat ini memiliki dasar yang baik tetapi memerlukan perbaikan pada konstruksi objek untuk menyimpan dan mengelola data secara efektif. Dengan memperbaiki konstruktor dan menambahkan fitur penyimpanan, kelas Manusia dapat mengatasi masalah dengan lebih baik.

[Nomor 3] Penyusunan Algoritma dan Kode Program

1. Rancang desain solusi atau algoritma.

Algoritamanya :

 - 1)algoritma class manusia
 - mulai
 - deklarasi atribut
 - konstruktor
 - definisikan metode sukamakan (String)
 - Definisikan metode sukamakan (int, int)
 - 2)Algoritma metode main
 - Mulai
 - Inisialisasi objek manusia
 - Panggil metode sukamakan (string)
 - Panggil metode sukamakan (int, int)
 - Cetak jumlah porsi makanan
 - Akhiri
2. Tuliskan kode program dan luaran
 - a. Beri komentar pada kode
 - informasi Pribadi: Program mencetak nama, warna rambut, dan jenis kelamin berdasarkan input yang diberikan saat pembuatan objek Manusia.
 - Makanan yang Disukai: Program menampilkan makanan yang disukai, yang dicetak menggunakan metode sukaMakan(String makan).

- Perhitungan Porsi: Program menghitung total porsi makanan yang dimakan dalam seminggu menggunakan metode `sukaMakan(int porsi, int hari)` dan menampilkan hasilnya.
- b. Uraikan luaran yang dihasilkan
 - Nama saya : Syifa
 - Warna Rambut : hitam
 - Jenis Kelamin : p
 - Suka Makan : Mie Ayam
 - Porsi yang dimakan perminggu = 21 porsi
- c. Screenshot/ Capture potongan kode dan hasil luaran

```

1- public class Manusia {
2-     //deklarasi atribut Manusia dalam variabel
3-     String nama, rambut;
4-     char jenisKelamin;
5-
6-     //deklarasi constructor
7-     public Manusia(String nama, String rambut, char jenisKelamin) {
8-         System.out.println(" Nama saya : " + nama +
9-             "\n Warna Rambut : " + rambut + "\n Jenis Kelamin : " + jenisKelamin);
10-    }
11-
12-    //deklarasi method
13-    void sukamakan (String makan) {
14-        System.out.println(" Suka Makan : " + makan);
15-    }
16-
17-    int sukaMakan (int porsi, int hari) {
18-        return porsi * hari;
19-    }
20-
21-    //deklarasi method utama
22-    public static void main (String[] args) {
23-        Manusia satu = new Manusia("Syifa", "hitam", 'p');
24-        satu.sukamakan("Mie Ayam");
25-        int jumlahPorsi = satu.sukaMakan(3, 7);
26-        System.out.println("Porsi yang dimakan perminggu = " + jumlahPorsi + " porsi");
27-    }
28- }
29-

```

Nama saya : Syifa
 Warna Rambut : hitam
 Jenis Kelamin : p
 Suka Makan : Mie Ayam
 Porsi yang dimakan perminggu = 21 porsi

i Compiled and executed in 1.452 sec(s)

[Nomor 3] Kesimpulan

3) Evaluasi

3. Apa konsekuensi dari skenario pemrograman ini?

Konsekuensi dari skenario pemrograman ini mencakup kebutuhan untuk perbaikan dalam penyimpanan data, pengelolaan atribut dan metode, serta bagaimana data diproses dan digunakan. Untuk meningkatkan desain, Anda mungkin perlu memperkenalkan penyimpanan data dalam atribut, mengoptimalkan penggunaan metode, dan memastikan informasi yang dihasilkan dapat digunakan secara efisien dalam aplikasi yang lebih kompleks.

Template Lembar Kerja Individu

Nama & NPM	Topik:	Tanggal:
Syifa Ariqah Pajriyanti (G1F024009)	Kelas Java	15 september 2024

[Nomor 4] Identifikasi Masalah:

4. Uraikan permasalahan dan variable

```
public class Ortu {    // membuat kelas induk
    void sukaMenonton(String a) {    // method induk spesifik
        System.out.println("Nonton " + a);
    }
    void sukaMembaca(String a) {    // method induk umum bisa diubah anak
        System.out.println("Suka Baca " + a);
    }
}

public static void main(String [] args) {
    System.out.println("Sifat Orang Tua :");
    Ortu objek0 = new Ortu();    // memanggil objek induk
    objek0.sukaMenonton("Berita");    // memanggil sifat spesifik induk
    objek0.sukaMembaca("Koran");    // memanggil method dengan variabel dapat
    diubah

    System.out.println("\n Sifat Anak :");
    Anak objekA = new Anak();    //memanggil objek anak
    objekA.sukaMenonton(9, "Film Drakor");    //memanggil sifat spesifik anak
    yang diturunkan induk
    objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis
    diturunkan tanpa deklarasi ulang di anak
}    }
```

```
class Anak extends Ortu {
    void sukaMenonton(int a, String b) {
        System.out.println("Nonton Jam " + a + " Malam " + b);
    }
    void sukaMenonton(String a) {    // method induk spesifik
        System.out.println("Nonton " + a);
    }
    void sukaMembaca(String a) {    // method induk umum bisa diubah anak
        System.out.println("Suka Baca " + a);
    }
}

public static void main(String [] args) {
    System.out.println("Sifat Orang Tua :");
    Ortu objek0 = new Ortu();    // memanggil objek induk
    objek0.sukaMenonton("Berita");    // memanggil sifat spesifik induk
    objek0.sukaMembaca("Koran");    // memanggil method dengan variabel dapat
    diubah

    System.out.println("\n Sifat Anak :");
    Anak objekA = new Anak();    //memanggil objek anak
    objekA.sukaMenonton(9, "Film Drakor");    //memanggil sifat spesifik anak
    yang diturunkan induk
    objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis
    diturunkan tanpa deklarasi ulang di anak
}    }
```

4.1. Evaluasi method yang dimiliki `class Anak extends Ortu` dengan method di `class Ortu`!

Apakah penulisan method ini sudah efisien?

Penulisan metode di kelas Anak sudah efisien jika tujuannya adalah untuk mengimplementasikan atau mengubah perilaku metode yang diwarisi dari kelas Ortu dan menyediakan versi metode yang berbeda melalui overloading.

4.2 Setelah dirunning di JDoodle, catat waktu eksekusinya.

Rekomendasikan perbaikan penulisan kode method untuk dapat mengefisienkan waktu eksekusi!

- Menghapus Method main yang Redundan di Kelas Anak: Method main sebaiknya hanya ada di satu kelas. Saya memindahkannya ke kelas terpisah yang bernama Main untuk struktur yang lebih baik.
- Penggunaan Anotasi Override: Saya menggunakan Override untuk menunjukkan bahwa method di kelas Anak adalah pengganti method dari kelas Ortu. Ini membantu meningkatkan keterbacaan dan mendeteksi kesalahan saat kompilasi.
- Mengurangi Redundansi: Method spesifik sukaMenonton(String a) hanya didefinisikan sekali di kelas Anak, dan versi overloaded digunakan ketika perlu menambah fungsionalitas (dengan parameter int).
- Pemisahan yang Jelas: Kelas dan method diatur secara logis, sehingga kode menjadi lebih mudah dibaca dan dipelihara.

3. Rincikan sumber informasi yang relevan (buku / webpage)

- **Video Materi 1 tentang Kelas, Objek, Method –**
<https://www.youtube.com/watch?v=60ldOc8m8Es>
- **Video Materi 2 tentang –** <https://www.youtube.com/watch?v=6qULMlcv-eg>

[Nomor 4] Analisis dan Argumentasi

1. Uraikan rancangan solusi yang diusulkan.
Saya mengkonversi rancangan solusi ini mengilustrasikan penggunaan konsep pewarisan dan polimorfisme dalam pemrograman berorientasi objek melalui dua kelas: Ortu sebagai kelas induk dan Anak sebagai kelas turunan. Kelas Ortu mendefinisikan metode dasar untuk aktivitas menonton dan membaca, sementara kelas Anak menambahkan fungsionalitas spesifik dan mengoverride metode induk.
2. Analisis solusi, kaitkan dengan permasalahan.
Analisis solusi ini menunjukkan bahwa pendekatan menggunakan pewarisan dan polimorfisme tidak hanya memberikan cara untuk mengorganisir kode, tetapi juga menangani masalah umum yang dihadapi dalam pengembangan perangkat lunak. Dengan solusi ini, pengembang dapat menciptakan aplikasi yang lebih terstruktur, mudah dipahami, dan lebih mudah untuk dikembangkan di masa depan.

[Nomor 4] Penyusunan Algoritma dan Kode Program

1. Rancang desain solusi atau algoritma.
Algoritmanya :
 - Definisikan kelas ortu
 - Defnisikan kelas anak yang mewarisi dari ortu
 - Kelas ortu
 - Kelas anak
2. Tuliskan kode program dan luaran
 - a. Beri komentar pada kode
 - Kelas Ortu: Di sini dijelaskan bahwa ini adalah kelas induk, dan metode di dalamnya memiliki fungsi spesifik dan umum.
 - Method sukaMenonton dan sukaMembaca: Komentar menjelaskan fungsi dari masing-masing metode.

- Metode main di kelas Ortu: Menjelaskan proses pembuatan objek dan pemanggilan metode.
- Kelas Anak: Mengindikasikan bahwa kelas ini adalah turunan dari kelas Ortu, dengan komentar untuk setiap metode yang menjelaskan tujuan dan fungsinya.
- Override: Menjelaskan bahwa metode di kelas anak mengubah perilaku dari metode induk.

b. Uraikan luaran yang dihasilkan

Sifat Orang Tua :

Nonton Berita

Suka Baca Koran

Sifat Anak :

Nonton Jam 9 Malam Film Drakor

Suka Baca Komik One Piece

c. Screenshot/ Capture potongan kode dan hasil luaran

```

6      System.out.println("Suka Baca " + a);
7  }
8
9  public static void main(String [] args) {
10     System.out.println("Sifat Orang Tua :");
11     Ortu objekO = new Ortu(); // memanggil objek induk
12     objekO.sukaMenonton("Berita"); // memanggil sifat spesifik induk
13     objekO.sukaMembaca("Koran"); // memanggil method dengan variabel dapat diubah
14
15     System.out.println("\n Sifat Anak :");
16     Anak objekA = new Anak(); //memanggil objek anak
17     objekA.sukaMenonton(9, "Film Drakor"); //memanggil sifat spesifik anak yang diturun.
18     objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis diturunkan.
19 }
20
21 class Anak extends Ortu {
22     void sukaMenonton(int a, String b) {
23         System.out.println("Nonton Jam " + a + " Malam " + b);
24     }
25     void sukaMenonton(String a) { // method induk spesifik
26         System.out.println("Nonton " + a);
27     }
28     void sukaMembaca(String a) { // method induk umum bisa diubah anak
29         System.out.println("Suka Baca " + a);
30     }
31 }
32 public static void main(String [] args) {
33     System.out.println("Sifat Orang Tua :");
34     Ortu objekO = new Ortu(); // memanggil objek induk
35     objekO.sukaMenonton("Berita"); // memanggil sifat spesifik induk
36     objekO.sukaMembaca("Koran"); // memanggil method dengan variabel dapat diubah
37
38     System.out.println("\n Sifat Anak :");
39     Anak objekA = new Anak(); //memanggil objek anak
40     objekA.sukaMenonton(9, "Film Drakor"); //memanggil sifat spesifik anak yang diturun.
41     objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis diturunkan.
42 }

```

Sifat Orang Tua :
Nonton Berita
Suka Baca Koran

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

Compiled and executed in 2.494 sec(s)

[Nomor 4] Kesimpulan

3. Evaluasi

1. Apa konsekuensi dari skenario pemrograman ini?

Skenario pemrograman yang menggunakan pewarisan dan polimorfisme membawa banyak manfaat dalam hal efisiensi dan organisasi, tetapi juga datang dengan tantangan dan risiko tertentu. Pengembang perlu mempertimbangkan konsekuensi ini dan merancang sistem mereka dengan hati-hati untuk memaksimalkan keuntungannya sambil meminimalkan potensi masalah. Kejelasan dalam desain dan dokumentasi yang baik menjadi kunci untuk mengatasi tantangan yang mungkin muncul.

--

