NAMA : TIARA YEMELDA

NPM : G1F021008

TUGAS 3

[No. 1] Identifikasi Masalah

1) Uraikan permasalahan dan variabel

- soal:

Analisa ciri-ciri umum Kelas `Manusia` yang dapat menjadi atribut variabel dan perilaku/behavior untuk method.

- Diketahui dari soal:

Kelas `Manusia` memiliki dua variabel: `nama` dan `rambut`. Kelas ini juga memiliki constructor tanpa parameter yang menampilkan pesan ketika objek dibuat tanpa parameter.

[No. 1] Analisis dan Argumentasi

1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara: Menambah atribut yang relevan seperti `nama` dan `rambut`, serta menambah method yang merepresentasikan perilaku manusia seperti `berjalan()` dan `berbicara()`.
2) Alasan solusi ini karena: Atribut tersebut merepresentasikan ciri-ciri fisik manusia, sedangkan method merepresentasikan perilaku umum manusia.
3) Perbaikan kode program dengan cara: Menambahkan atribut tambahan dan method yang menggambarkan perilaku manusia.

[No. 1] Penyusunan Algoritma dan Kode Program

1) Algoritma:

- Definisikan kelas `Manusia` dengan atribut `nama` dan `rambut`.

- Tambahkan constructor tanpa parameter untuk menampilkan pesan.

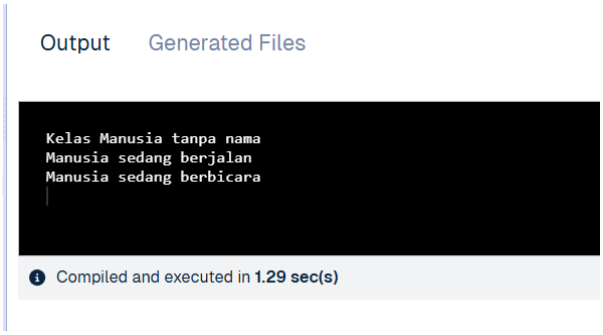
- Buat method `berjalan()` dan `berbicara()` untuk merepresentasikan perilaku manusia.

2) Kode program dan luaran:

```
public class Manusia {
    String nama;
    String rambut;

    public Manusia() {
        System.out.println("Kelas Manusia tanpa nama");
    }

    void berjalan() {
        System.out.println("Manusia sedang berjalan");
    }

    void berbicara() {
        System.out.println("Manusia sedang berbicara");
    }

    public static void main(String[] args) {
        Manusia manusia1 = new Manusia();
        manusia1.berjalan();
        manusia1.berbicara();
    }
}
```

a) Screenshot/Capture potongan kode dan hasil luaran:

- Hasil luaran:



b) Analisa luaran yang dihasilkan:

Luaran sudah sesuai dengan program yang disusun. Tipe data yang digunakan sudah sesuai untuk menggambarkan karakteristik dan perilaku manusia.

[No. 1] Kesimpulan

1) Analisa:

a) Saya menggunakan kelas `public` untuk kelas `Manusia` karena kelas ini digunakan untuk menggambarkan manusia secara umum.

b) Perbaikan program dilakukan dengan menambahkan method untuk merepresentasikan perilaku manusia, yaitu `berjalan()` dan `berbicara()`.

[No. 2] Identifikasi Masalah

1) Uraikan permasalahan dan variabel

- Tuliskan kembali soal:

Susun kembali kode di Contoh 2 dengan menambahkan data ciri-ciri Anda di dalam variabel constructor.

- Diketahui dari soal:

Terdapat class `Ortu` dengan constructor yang menerima parameter `nama` dan `rambut`.

[No. 2] Analisis dan Argumentasi

1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara:

Menambahkan ciri-ciri saya ke dalam variabel constructor ketika objek `Ortu` dibuat.

2) Alasan solusi ini karena:

Constructor menerima parameter untuk inisialisasi atribut `nama` dan `rambut`, sehingga saya bisa menambahkan data diri saya.

3) Perbaikan kode program dengan cara:

Mengisi constructor dengan nama saya dan ciri-ciri rambut saya.

[No. 2] Penyusunan Algoritma dan Kode Program

1) Algoritma:

- Definisikan kelas `Ortu` dengan constructor yang menerima parameter `nama` dan `rambut`.

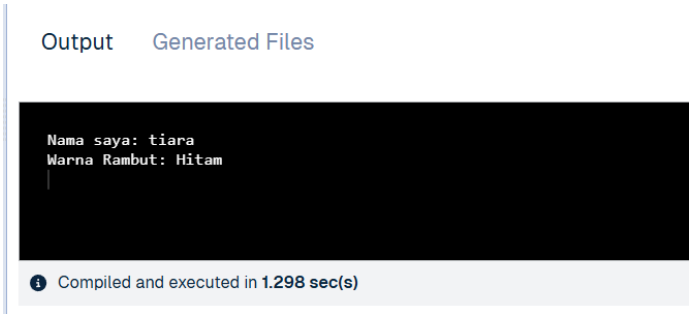
- Buat objek `Ortu` dengan data ciri-ciri diri sendiri.

2) Kode program dan luaran:

```java
public class Ortu {
    public Ortu(String nama, String rambut) {
        System.out.println("Nama saya: " + nama);
        System.out.println("Warna Rambut: " + rambut);
    }

    public static void main(String[] args) {
        Ortu saya = new Ortu("tiara", "Hitam");
    }
}
```

a) Screenshot/Capture potongan kode dan hasil luaran:

- Hasil luaran:



b) Analisa luaran yang dihasilkan:

Luaran sudah sesuai dengan program yang disusun. Data diri ditampilkan sesuai dengan parameter yang dimasukkan ke dalam constructor.

[No. 2] Kesimpulan

1) Analisa:

Saya menggunakan constructor untuk menginisialisasi atribut `nama` dan `rambut`. Program berhasil menampilkan data sesuai dengan ciri-ciri saya.

[No. 3] Identifikasi Masalah

1) Uraikan permasalahan dan variabel:

- Tuliskan kembali soal:

Analisa perbedaan deklarasi constructor, method, dan method utama.

- Diketahui dari soal:

Constructor digunakan untuk inisialisasi, method untuk perilaku, dan method utama untuk eksekusi program.

[No. 3] Analisis dan Argumentasi

1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara:

Menjelaskan fungsi dari masing-masing elemen kode (constructor, method, dan method utama).

2) Alasan solusi ini karena:

Kode program harus memiliki struktur yang jelas dengan constructor untuk inisialisasi, method untuk operasi, dan method utama untuk menjalankan program.

3) Perbaikan kode program dengan cara:

Menambah constructor, method, dan method utama yang berfungsi sesuai.

[No. 3] Penyusunan Algoritma dan Kode Program

1) Algoritma:

- Buat constructor untuk inisialisasi atribut.

- Buat method untuk menggambarkan perilaku objek.

- Tambahkan method utama untuk menjalankan program.

2) Kode program dan luaran:

```java
public class Manusia {
    String nama;
    String rambut;

    public Manusia(String nama, String rambut) {
        this.nama = nama;
        this.rambut = rambut;
    }

    void sukaNonton(String film) {
        System.out.println("Hobi Menonton: " + film);
    }

    public static void main(String[] args) {
        Manusia satu = new Manusia("tiara", "hitam");
        satu.sukaNonton("Drakor");
    }
}
```

a) Screenshot/Capture potongan kode dan hasil luaran:

- Hasil luaran:

b) Analisa luaran yang dihasilkan:

Luaran sudah sesuai dengan program yang disusun. Constructor berhasil menginisialisasi atribut, method menjalankan operasi, dan method utama mengeksekusi program.

[No. 3] Kesimpulan

1) Analisa:

Pada program ini, saya menggunakan constructor untuk inisialisasi data, method untuk menggambarkan perilaku, dan method utama untuk eksekusi program.

[No. 4] Identifikasi Masalah

1) Uraikan permasalahan dan variabel:

- Tuliskan kembali soal:

Bandingkan method yang dimiliki class `Anak` dengan method di class `Ortu`.

- Diketahui dari soal:

Kelas `Anak` adalah turunan dari kelas `Ortu` dan memiliki method yang sama serta beberapa method yang berbeda.

[No. 4] Analisis dan Argumentasi

1) Saya mengusulkan permasalahan ini dapat diatasi dengan cara:

Menjelaskan perbedaan antara method yang di-override dan method yang di-overload.

2) Alasan solusi ini karena:

Dengan membandingkan kedua kelas, kita bisa memahami bagaimana inheritance berfungsi dalam OOP.

3) Perbaikan kode program dengan cara:

Mengimplementasikan method yang sama di kelas `Anak` untuk memperlihatkan overriding dan menambahkan method baru.

[No. 4] Penyusunan Algoritma dan Kode Program

1) Algoritma:

- Definisikan kelas `Anak` yang mewarisi `Ortu`.

- Override method dari `Ortu` dan tambahkan method baru.

2) Kode program dan luaran:

```java
public class Ortu {
    void sukaMenonton(String a) {
        System.out.println("Nonton " + a);
    }

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

class Anak extends Ortu {
    // Override method dari Ortu
    @Override
    void sukaMenonton(String a) {
        System.out.println("Anak suka nonton " + a);
    }

    // Method baru dengan parameter yang berbeda (overloading)
    void sukaMenonton(int a, String b) {
        System.out.println("Nonton film dengan rating " + a + ": " + b);
    }

    void sukaMembaca(String a) {
        System.out.println("Anak suka membaca " + a);
    }
}

class Main { // Kelas baru sebagai entry point
    public static void main(String[] args) {
        Anak anak = new Anak();
        anak.sukaMenonton("Komedi"); // Memanggil method yang di-overridden
        anak.sukaMembaca("Buku Cerita");
        anak.sukaMenonton(5, "Action"); // Memanggil method yang di-overloaded
    }
}
```

n("Anak suka

a) Screenshot/Capture potongan kode dan hasil luaran:

- Hasil luaran:

```
Anak suka nonton Komedi
Anak suka membaca Buku Cerita
Nonton film dengan rating 5: Action
```

b) Analisa luaran yang dihasilkan:

Luaran sudah sesuai dengan program yang disusun. Method di kelas `Anak` berhasil mengoverride method dari kelas `Ortu` dan menambah method baru.

[No. 4] Kesimpulan

1) Analisa:

Di kelas `Anak`, saya mengoverride method `sukaMembaca` dan mendefinisikan method baru `sukaMenonton` dengan parameter berbeda. Ini menunjukkan konsep overriding dan overloading dalam inheritance.

Refleksi

Pada latihan ini, saya belajar cara mengimplementasikan constructor, method, dan inheritance (extends). Tantangan utama adalah memahami perbedaan method overloading dan overriding, tetapi saya sudah memahaminya dengan mencoba beberapa contoh implementasi.