|  |  |  |
| --- | --- | --- |
| **Nama & NPM** | **Topik:** | **Tanggal:** |
| Melisa Yunita Sari (G1F024026) | Tugas Individu kelas Java | **18 September 2024** |

**[No. 1] Identifikasi Masalah:**

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");  
    }

1.Perbaiki pesan kesalahan Contoh 1!  
2. Analisa ciri-ciri lain Kelas Manusia yang dapat menjadi  
           a. atribut variabel, dan  
           b. perilaku/ behavior!

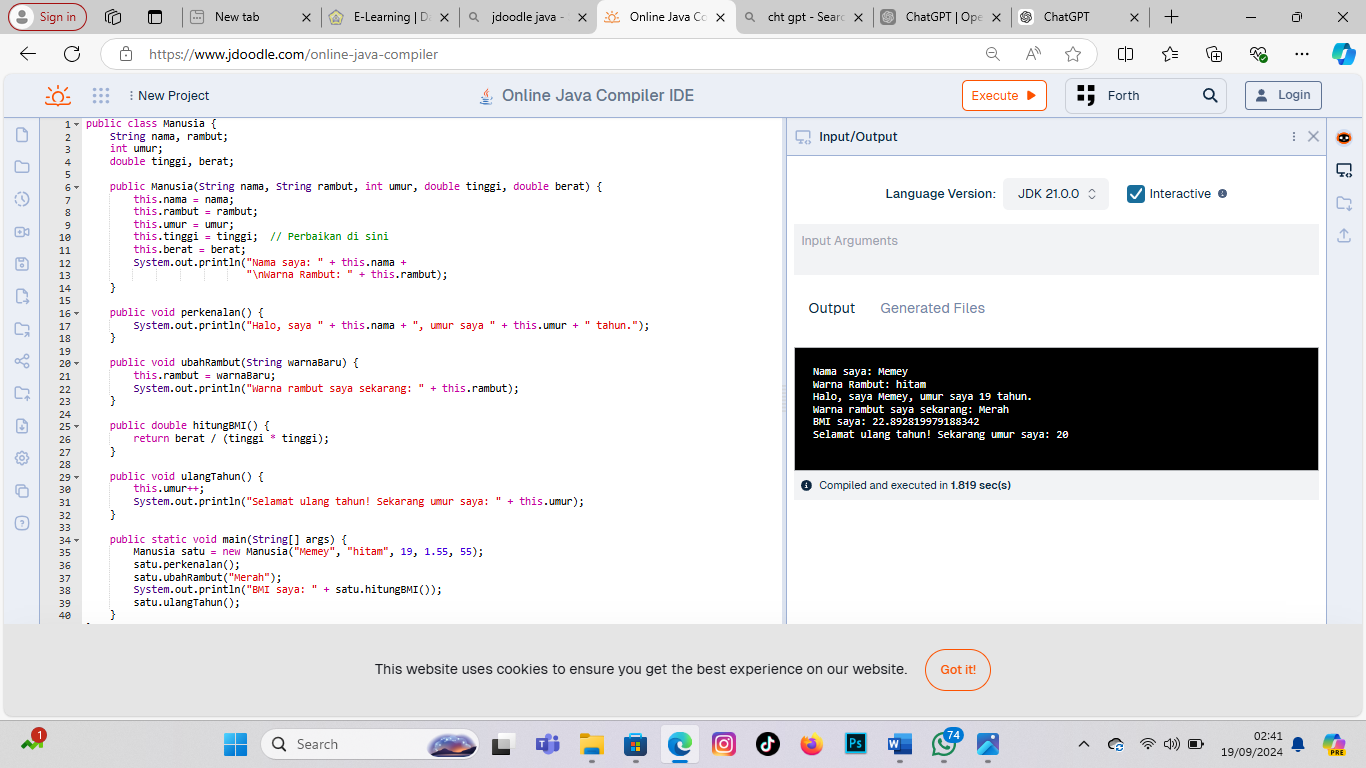
**[No.1] Analisis dan Argumentasi**

1. Saya mengusulkan permasalahan ini dapat diatasi dengan cara menambahkah metode dan atribut yang relevan.
2. Alasan solusi ini karena dengan menambahkan atribut kelas manusia menjadi lebih informatif dan interaktif, menciptakan objek yang lebih realistis dan berguna.
3. Perbaikan kode program dengan cara cara mengimplementasikan atribut tambahan dan metode baru yang memungkinkan objek Manusia untuk melakukan lebih banyak fungsi, seperti memperkenalkan diri dan menghitung BMI**.**

**[****No.1 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai
3. Deklarasi kelas
4. Deklarasi atribut
5. Deklarasi Constructor
6. Deklarasi mode utama
7. Selesai

2) Kode program dan luaran



* Analisa luaran yang dihasilkan

program ini menunjukkan bagaimana objek dari kelas Manusia dapat digunakan untuk merepresentasikan data dan perilaku individu. Luaran yang dihasilkan memberikan informasi lengkap tentang atribut dan metode yang diterapkan pada objek. Kelas manusia memiliki atribut yang mendefenisikan karakteristik individu dan metode yang menggambarkan perilaku merka.

**[No.1] Kesimpulan**

Kode program ini memungkinkan penyimpanan informasi tentang individu secara teratur, mendukung interaksi pengguna, dan menerapkan prinsip OOP, tetapi juga harus diperhatikan potensi kesalahan dan validasi data.

**[No. 2] Identifikasi Masalah:**

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");  
     }  
}

**1.  Evaluasi penyebab kesalahan dan perbaiki kode tersebut!  
2. Apabila Ortu memiliki data variabel umur = 25 dan jenis kelamin = P (untuk Perempuan), rekomendasikan constructor dengan parameter yang baru untuk ditambahkan dalam program!**

**[No.2] Analisis dan Argumentasi**

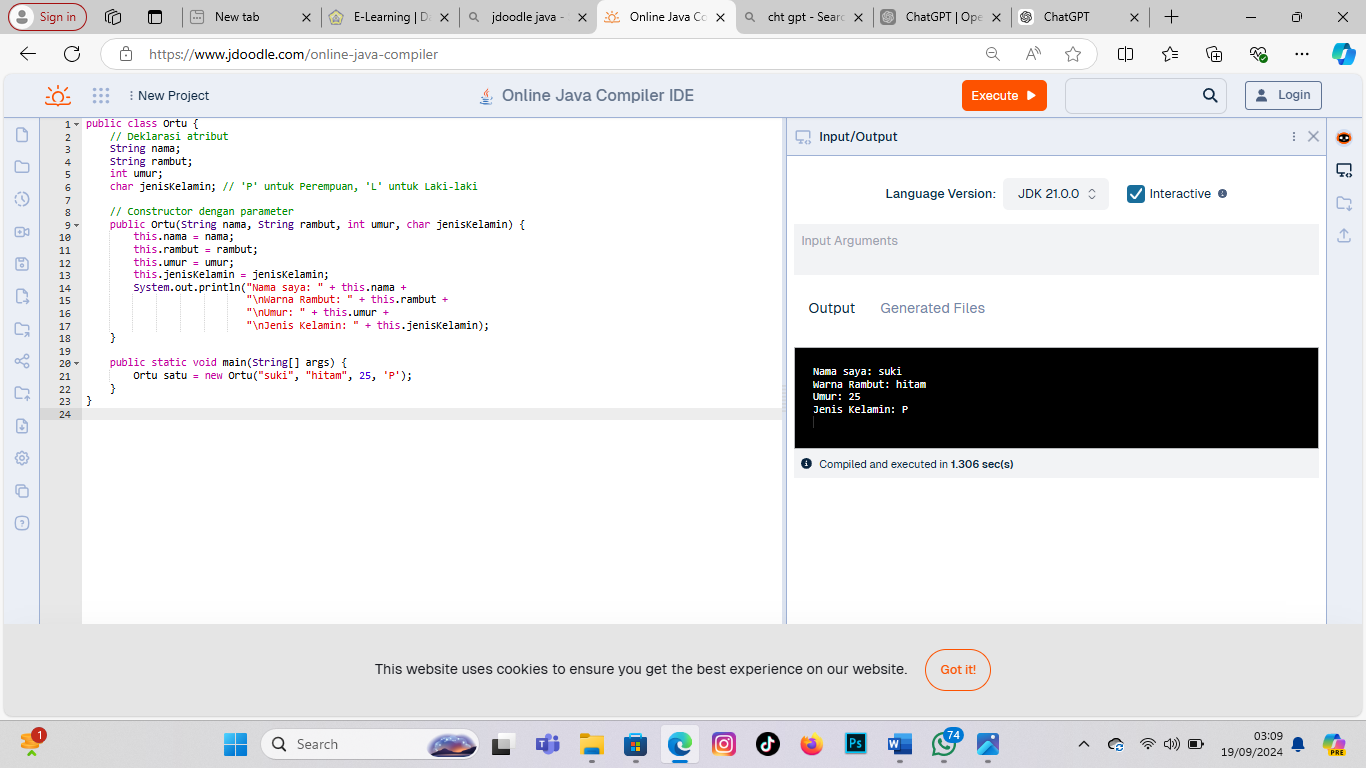
1 Saya mengusulkan permasalahan ini dapat diatasi dengan cara menggunakan constructor yang lengkap untuk menginisialisasi semua atribut kelas.

2 Alasan solusi ini karena constructor yang lengkap memungkinkan kita untuk membuat objek dengan semua data yang diperlukan sekaligus, sehingga lebih mudah untuk memahami dan mengelola objek yang dibuat.

3. Perbaikan kode program dengan cara menambahkan parameter untuk umur dan jenisKelamin dalam constructor, sehingga ketika objek dibuat, semua atribut dapat diatur dengan jelas dan langsung.

**[No.2] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai
3. Deklarasi kelas
4. Deklarasi atribut
5. Deklarasi Constructor
6. Method main
7. Eksekusi
8. Kode program dan luaran



* Analisa luaran yang dihasilkan

Program berhasil menginisialisasi objek dengan data yang diberikan dan mencetak informasi secara akurat.

Semua atribut berhasil ditampilkan sesuai dengan input yang dimasukkan, menunjukkan bahwa constructor dan method main berfungsi dengan baik.

**[No.2] Kesimpulan**

Diperlukan sebuah kelas untuk menyimpan dan menampilkan informasi mengenai atribut seseorang.

Program ini menunjukkan kemampuan dasar pemrograman beriteraksi objek dalam java.

**[No. 3] Identifikasi Masalah**

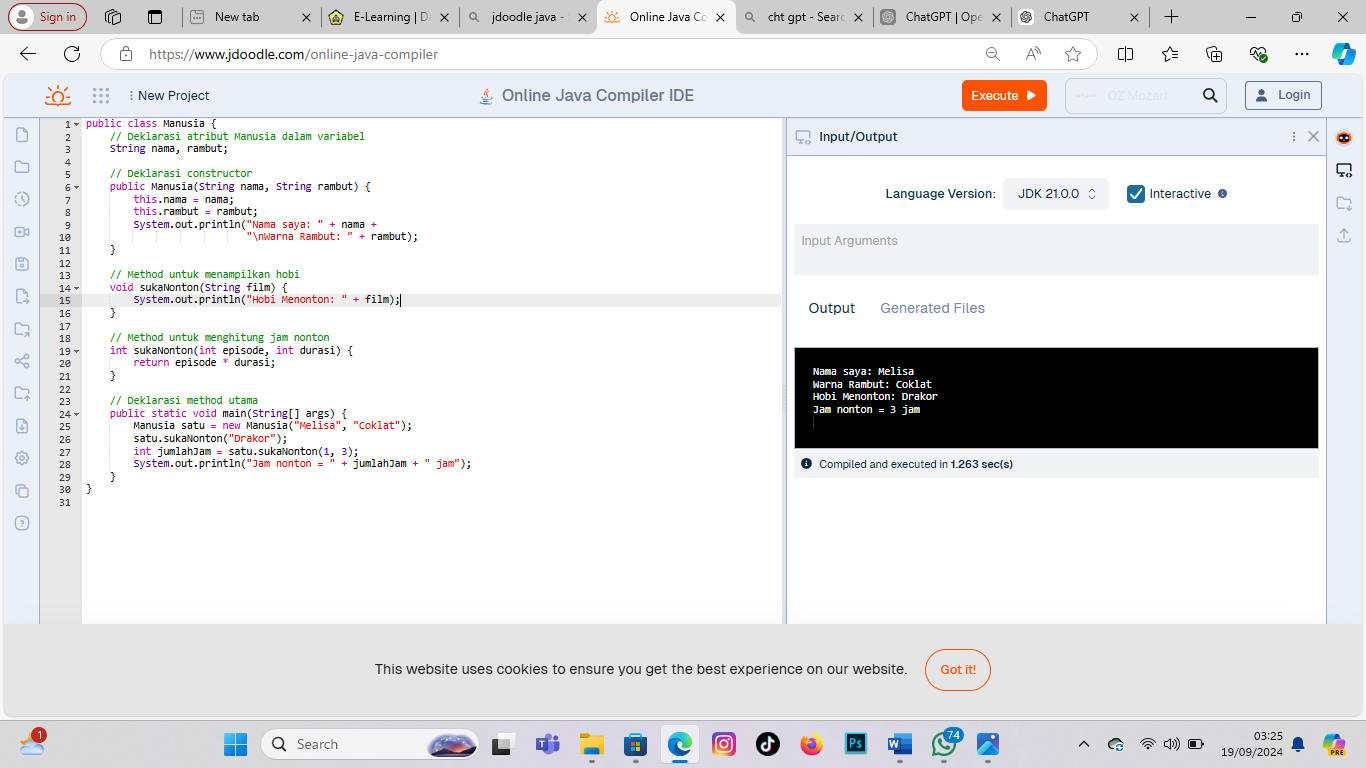
public class Manusia {  
    //deklarasi atribut Manusia dalam variabel  
    String nama, rambut;  
      
    //deklarasi constructor  
    public Manusia1(String nama, String rambut) {  
            System.out.println(" Nama saya : "+ nama +   
            "\n Warna Rambut : " + rambut);      
    }  
      
    //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");  
    }  
}

1.  Evaluasi penyebab kesalahan dan perbaiki kode tersebut!  
2.  Ubahlah method dan constructor Contoh 3 sesuai dengan perilaku/ behavior anda  
3.  Berdasarkan Contoh 3 dan Latihan 3.2. simpulkan perbedaan:  
        a) constructor overloading dan overriding  
        b) method overloading, dan method overriding  
        c) method yang mengembalikan nilai dan method tidak mengembalikan nilai

**[No. 3] Penyusunan Algoritma dan Kode Program**

1. Algoritma
2. Mulai
3. Deklarasi
4. Selesai

2) kode program dan luaran



* Analisa luaran yang dihasilkan

rogram ini berhasil menampilkan informasi tentang objek Manusia, termasuk nama, warna rambut, hobi, dan menghitung jam menonton berdasarkan input yang diberikan. Semua metode berfungsi dengan baik dan memberikan output yang diharapkan.

**[No.3] Kesimpulan**

Program ini menunjukkan penggunaan prinsip pemrograman berorientasi objek (OOP) dalam Java dengan baik, termasuk penggunaan constructor, method overloading, dan atribut. Kode tersebut dapat dijalankan tanpa kesalahan dan memberikan hasil yang sesuai dengan yang diharapkan. Program ini dapat diperluas lebih lanjut dengan menambahkan fitur atau atribut lain, seperti umur atau hobi lainnya.

**[No. 4] Identifikasi Masalah:**

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 objekO = new Ortu();     // memanggil objek induk  
    objekO.sukaMenonton("Berita");    // memanggil sifat spesifik induk  
    objekO.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 objekO = new Ortu();     // memanggil objek induk  
    objekO.sukaMenonton("Berita");        // memanggil sifat spesifik induk  
    objekO.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  
}   }

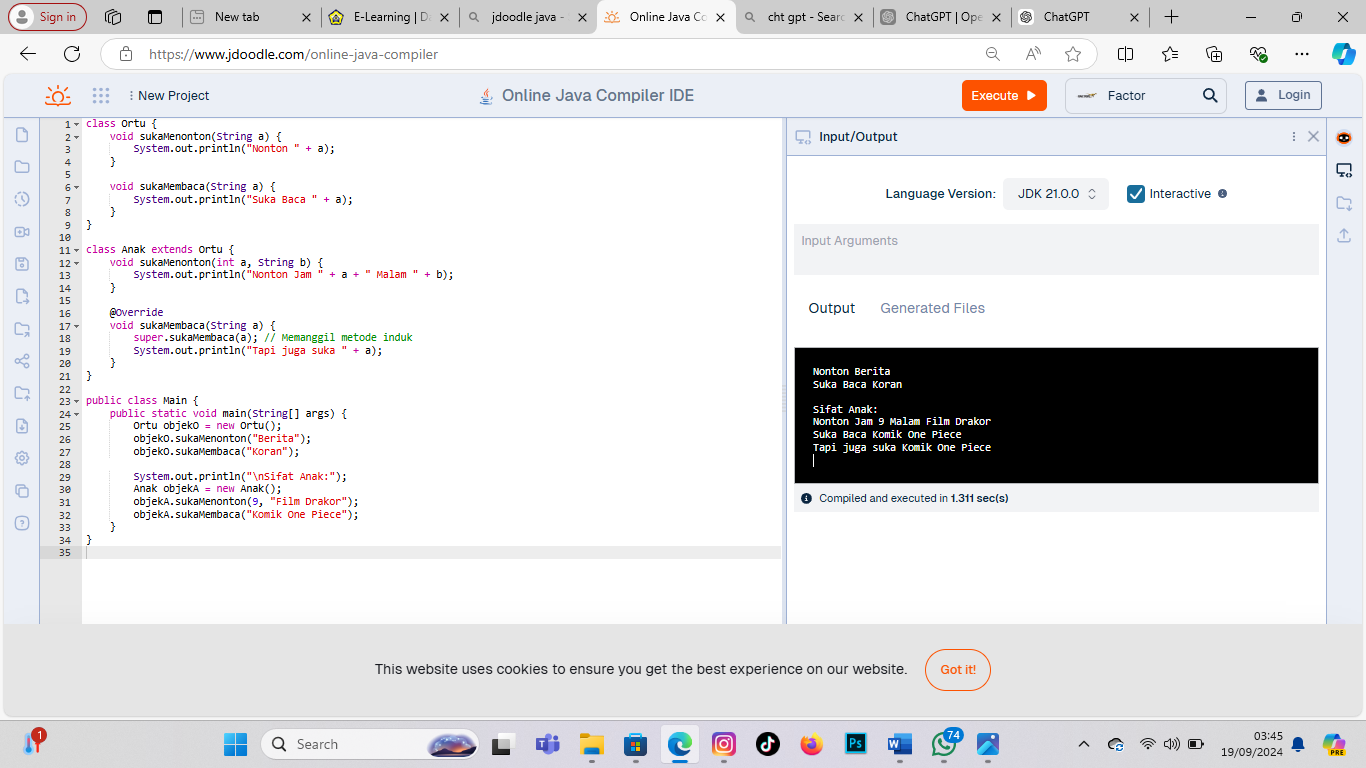
1) Evaluasi method yang dimiliki class Anak extends Ortu dengan method di class Ortu!  
       Apakah penulisan method ini sudah efisien?  
2. Setelah dirunning di JDoodle, catat waktu eksekusinya.   
       Rekomendasikan perbaikan penulisan kode method untuk dapat mengefisienkan waktu eksekusi

**[No.4 ] Penyusunan Algoritma dan Kode Program**

1. Algoritma

* Mulai
* Deklarasi
* Selesai

2) kode program dan luaran



* Analisa luaran yang dihasilkan

**ewarisan dan Overriding:** Program menunjukkan penggunaan pewarisan dan overriding dengan baik. Metode yang diubah di kelas Anak memungkinkan fleksibilitas dalam perilaku objek.

**Output yang Konsisten:** Output mencerminkan dengan tepat apa yang diharapkan, mengindikasikan bahwa metode bekerja sesuai dengan tujuan yang ditetapkan. **Desain Kode yang Baik:** Kode yang disusun mencerminkan struktur yang jelas dan mudah dipahami, memisahkan tanggung jawab antara kelas induk dan anak.