Nama & NPM	Topik:	Tanggal:
Alif Alfarizi **G1F024069**	**Kelas, Objek, Method**	**18 September 2024**

[Nomor 1] Identifikasi Masalah:

Uraikan permasalahan dan variable

```
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

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

Soal:

1.1. Perbaiki pesan kesalahan Contoh 1!

1.2. Analisa ciri-ciri lain Kelas Manusia yang dapat menjadi
a. atribut variabel, dan
b. perilaku/ behavior!

[Nomor 1] Analisis dan Argumentasi

Uraikan rancangan solusi yang diusulkan.

Kode yang bermasalah ada pada deklarasi constructor yang mana construktor harus mengikuti public class yang ada dan menambahkan tipe data string rambut agar rambut dapat terpanggil dari seperti ini **public Manusia1 (String nama) {** menjadi **public Manusia (String nama, String rambut) {** dan variable yang memakai Manusia harus diganti sesuai public classnya, lalu saya juga menambahkan atribut variable baru dan behavior baru juga seperti umur, tinggi badan dan sedang makan, maka kode programnya seperti ini

```
public class Manusia {  // deklarasi kelas
   //deklarasi konstruktur
   public Manusia(String nama, String rambut, int umur, int tinggiBadan) {
      System.out.println(" Nama saya : "+ nama +
      "\n Warna Rambut : " + rambut +
      "\n Umur : " + umur +
      "\n Tinggi Badan : " + tinggiBadan + " cm");
   }
   //Metode
   public void makan(String makanan) {
      System.out.println("Sedang makan " + makanan);
   }
```

```
  //deklarasi method utama
  public static void main( String[] args) {
     Manusia satu = new Manusia("Putri", "hitam", 18, 160);
     satu.makan("Nasi");
  }
 }
```

Luaran

Nama saya : Putri
Warna Rambut : hitam
Umur : 18
Tinggi Badan : 160 cm
Sedang makan Nasi

[Nomor 1] Penyusunan Algoritma dan Kode Program

Algoritma

1) Membuat public class
2) Deklarasi constructor
3) Menambahkan varibel seperti nama, rambut, umur, dan tinggi badan
4) Deklarasi metode behavior seperti sedang makan
5) Deklarasi main method
6) Luaran
7) Selesai

Hasil pemrograman seperti ini:

```java
public class Manusia {  // deklarasi kelas
    //deklarasi konstruktur
    public Manusia(String nama, String rambut, int umur, int tinggiBadan) {
        System.out.println(" Nama saya : "+ nama +
        "\n Warna Rambut : " + rambut +
        "\n Umur : " + umur +
        "\n Tinggi Badan : " + tinggiBadan + " cm");
    }
    //Metode
    public void makan(String makanan) {
        System.out.println("Sedang makan " + makanan);
    }

    //deklarasi method utama
    public static void main( String[] args) {
        Manusia satu = new Manusia("Putri", "hitam", 18, 160);
        satu.makan("Nasi");
    }
  }
```

```
 Nama saya : Putri
 Warna Rambut : hitam
 Umur : 18
 Tinggi Badan : 160 cm
Sedang makan Nasi
```

Maka luaran yang keluar akan seperti diatas dengan ada nya nama, warna rambut, umur, tinggi badan dan kegiatan yang sedang dilakukan oleh putri.

[Nomor 1] Kesimpulan

Analisa

a) Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!

Pemrograman ini tidak dapat berjanlan dikarenakan beberapa kode yang tidak sesuai dan juga saya menambahkan variable baru dan perilaku dari si putri ini dengan bahavior yang saya taruh dia sedang makan nasi dan akhirnya program berjalan sesuai dengan kode yang telah dibuat.

Nama & NPM	Topik:	Tanggal:
Alif Alfarizi **G1F024069**	**Kelas, Objek, Method**	**18 September 2024**

[Nomor 2] Identifikasi Masalah:

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

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

Soal:

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

[Nomor 2] Analisis dan Argumentasi

Uraikan rancangan solusi yang diusulkan.

Kode pemrograman yang bermasalah pada contruktor ortu yang mana huruf pertamanya tidak kapital seperti public class lalu saya juga menambah variable constructor dangan umur, dan jenis kelamin dengan tipe data int dan char, hingga hasil pemrograman seperti berikut

```
public class Ortu {
   //deklarasi constructor  (variabel constructor)
   public Ortu(String nama, String rambut, int umur, char jenisKelamin) {
   //nama,rambut,umur,jenisKelamin adalah variabel constructor
    System.out.println(" Nama saya : "+ nama +
   "\n Warna Rambut : " + rambut +
   "\n Umur : " + umur +
   "\n Jenis Kelamin : " + jenisKelamin);
}
   public static void main (String[] args) {
      Ortu satu = new Ortu("Putri", "hitam", 25, 'P');
    }
}
```

Luaran

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

[Nomor 2] Penyusunan Algoritma dan Kode Program

Algoritma

1) Membuat public class
2) Deklarasi constructor
3) Menambahkan variable constructor dengan nama, rambut, umur, dan jeniskelamin
4) Membuat main method
5) Luaran
6) Selesai

Hasil pemrograman seperti ini

```java
public class Ortu {
    //deklarasi constructor  (variabel constructor)
    public Ortu(String nama, String rambut, int umur, char jenisKelamin) {
    //nama,rambut,umur,jenisKelamin adalah variabel constructor
     System.out.println(" Nama saya : "+ nama +
    "\n Warna Rambut : " + rambut +
    "\n Umur : " + umur +
    "\n Jenis Kelamin : " + jenisKelamin);
}
    public static void main (String[] args) {
        Ortu satu = new Ortu("Putri", "hitam", 25, 'P');
     }
}
```

```
Nama saya : Putri
Warna Rambut : hitam
Umur : 25
Jenis Kelamin : P
```

Maka luaran seperti diatas dengan beberapa kode yang telah diperbaiki dan menambahkan variable construktor baru.

[Nomor 2] Kesimpulan

Analisa

a) Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!

Pada kode pemrograman ini saya memperbaiki beberapa kode yang eror dan juga menambahkan variable baru ke dalam constructor.

Nama & NPM	Topik:	Tanggal:
Alif Alfarizi **G1F024069**	**Kelas, Objek, Method**	**18 September 2024**

[Nomor 3] Identifikasi Masalah:

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");
   }
}
```

Luaran

Exception in thread "main" java.lang.Error: Unresolved compilation problems:
The method sukaNonton(String) is undefined for the type Manusia1
The method sukaNonton(int, int) is undefined for the type Manusia1
at Manusia1.main(Manusia1.java:23)

Soal:

3.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!
3.2. Ubahlah method dan constructor Contoh 3 sesuai dengan perilaku/ behavior anda
3.3. 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

[Nomor 3] Analisis dan Argumentasi

Uraikan rancangan solusi yang diusulkan.

Kode pemrograman ini memiliki beberapa kesalahan seperti public constructor yang salah dan int sukanonton berada dibawah method lalu beberapa kode yang kurang seperti perhitungan jumlah totaljam animenya disini saya mengubah film menjadi anime sesuai dengan soal yang boleh mengubahnya, setelah dianalisa dan diperbaiki kode program akan seperti ini

```
public class Manusia1 {
   //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
   int sukaNonton(String anime, int episode, int durasi) { {
      System.out.println(" Hobi Menonton : " + anime);
      int totaljam = episode * durasi;
      System.out.println("Total jam menonton: " + totaljam + "jam");
      return totaljam;
   }
   }

   //deklarasi method utama
   public static void main( String[] args) {
         Manusia1 satu = new Manusia1("Putri", "hitam");
         int jumlahJam = satu.sukaNonton("One Piece", 2, 2);
         System.out.println("Jam nonton = " +jumlahJam + " jam");
   }
}
```

Luran
Nama saya : Putri
Warna Rambut : hitam
Hobi Menonton : One Piece
Total jam menonton: 4jam
Jam nonton = 4 jam

Lalu ini soal simpulkan perbedaan

Constructor:

- **Overloading:** Beberapa konstruktor dengan parameter berbeda dalam satu kelas. Tujuannya untuk membuat objek dengan cara yang berbeda-beda.
- **Overriding:** Tidak bisa dilakukan secara langsung pada konstruktor. Konstruktor anak selalu memanggil konstruktor induk.

Method:

- **Overloading:** Beberapa metode dengan nama sama tapi parameter berbeda dalam satu kelas. Tujuannya untuk menyediakan fleksibilitas dalam pemanggilan metode.
- **Overriding:** Metode pada kelas anak mengubah implementasi metode yang sama pada kelas induk. Tujuannya untuk menyesuaikan perilaku metode sesuai dengan kelas anak.

Mengembalikan Nilai:

- **Ya:** Metode menghasilkan suatu nilai (misal: angka, teks).
- **Tidak (void):** Metode melakukan suatu tindakan tapi tidak menghasilkan nilai.

[Nomor 3] Penyusunan Algoritma dan Kode Program

Algoritma

1) Membuat public class
2) Deklarasi constructor
3) Deklarasi method

4) Menambahkan variable anime, episode, durasi
5) Deklarasi main method
6) Luaran
7) Selesai

Hasil pemrograman seperti berikut

```java
public class Manusia1 {
    //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
    int sukaNonton(String anime, int episode, int durasi) { {
        System.out.println(" Hobi Menonton : " + anime);
        int totaljam = episode * durasi;
        System.out.println("Total jam menonton: " + totaljam + "jam");
        return totaljam;
    }
    }

    //deklarasi method utama
    public static void main( String[] args) {
            Manusia1 satu = new Manusia1("Putri", "hitam");
            int jumlahJam = satu.sukaNonton("One Piece", 2, 2);
            System.out.println("Jam nonton = " +jumlahJam + " jam");
    }
}
```

```
 Nama saya : Putri
 Warna Rambut : hitam
 Hobi Menonton : One Piece
Total jam menonton: 4jam
Jam nonton = 4 jam
```

Maka luaran akan seperti diatas setelah saya memperbaiki beberapa kode program yang salah dan menambahkan beberapa variable yang dibutuhkan agar hasil lluaran sesuai dengan keinginan

[Nomor 3] Kesimpulan

Analisa

a) Susunlah kesimpulan berdasarkan permasalahan, algoritma, dan kode program!

Kode pemrograman ini saya memperbaiki dan menganalisa beberapa kode program yang salah dan kurang hingga mengeluarkan luaran yang sesuai.

Nama & NPM	Topik:	Tanggal:
Alif Alfarizi **G1F024069**	**Kelas, Objek, Method**	**18 September 2024**

[Nomor 4] Identifikasi Masalah:

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 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
}  }
```

Luaran

Sifat Orang Tua :
Nonton Berita
Suka Baca Koran

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

Soal:

4.1. Evaluasi method yang dimiliki class Anak extends Ortu dengan method di class Ortu!
Apakah penulisan method ini sudah efisien?

4.2. Setelah dirunning di JDoodle, catat waktu eksekusinya.
Rekomendasikan perbaikan penulisan kode method untuk dapat mengefisienkan waktu eksekusi!

[Nomor 4] Analisis dan Argumentasi

Uraikan rancangan solusi yang diusulkan.

Pada kode pemrograman ini saya hanya menjalankan program ini apakah memilki waktu yang efisien pada saat menjalankan program dan juga menganalisa apakan penulisan method tersebut sudah efisien atau belum

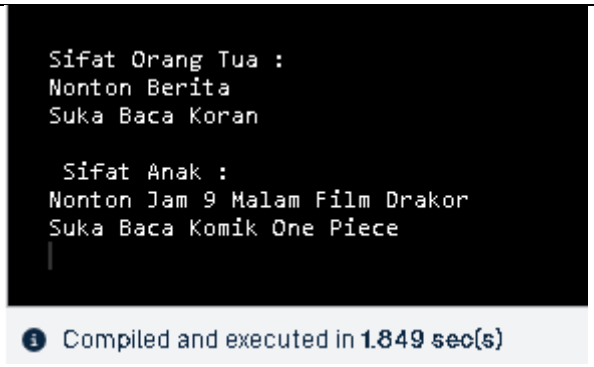
Hal ini saya mendapati bahwa Metode sukaMenonton sudah efisien dalam hal overloading, tetapi pastikan tidak ada kebingungan dalam penggunaannya.

Metode sukaMembaca di kelas Anak tidak perlu diulang jika tidak ada perubahan. Menghapusnya akan membuat kode lebih bersih dan lebih mudah dipelihara. Dan waktu pengeksekusian program tersebut di web jDoodle sya mendapati waktu **1.849 sec(s).**

[Nomor 4] Penyusunan Algoritma dan Kode Program

Screenshot/ Capture potongan kode dan hasil luaran

```
1  public class Ortu2 {       // membuat kelas induk
2    void sukaMenonton(String a) {    // method induk spesifik
3      System.out.println("Nonton " + a);
4    }
5    void sukaMembaca(String a) {     // method induk umum bisa diubah anak
6      System.out.println("Suka Baca " + a);
7    }
8
9  public static void main(String [] args) {
10     System.out.println("Sifat Orang Tua :");
11     Ortu2 objekO = new Ortu2();     // 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 diturunkan induk
18     objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis diturunkan tanpa deklarasi ulang di anak
19 }    }
20
21 class Anak extends Ortu2 {
22   void sukaMenonton(int a, String b) {
23       System.out.println("Nonton Jam " + a + " Malam " + b);
24   }
25
26   @Override
27   void sukaMenonton(String a) {            // method induk spesifik
28       System.out.println("Nonton " + a);
29   }
30
31   @Override
32   void sukaMembaca(String a) {     // method induk umum bisa diubah anak
33     System.out.println("Suka Baca " + a);
34   }
35 public static void main(String [] args) {
36     System.out.println("Sifat Orang Tua :");
37     Ortu2 objekO = new Ortu2();     // memanggil objek induk
38     objekO.sukaMenonton("Berita");        // memanggil sifat spesifik induk
39     objekO.sukaMembaca("Koran");     // memanggil method dengan variabel dapat diubah
40
41     System.out.println("\n Sifat Anak :");
42     Anak objekA = new Anak();    //memanggil objek anak
43     objekA.sukaMenonton(9, "Film Drakor");        //memanggil sifat spesifik anak yang diturunkan induk
44     objekA.sukaMembaca("Komik One Piece"); //memanggil method ke induk yang otomatis diturunkan tanpa deklarasi ulang di anak
45 }    }
```



Saya mendapati luaran seperti atas dengan waktu eksekusi yang ada diatas

[Nomor 4] Kesimpulan

Evaluasi

a) Apa konsekuensi dari skenario pemprograman ini?

Pada kode program ini Kelas Anak mewarisi metode dari Ortu2, memungkinkan penggunaan kembali kode dan juga overloading Mengizinkan variasi metode dengan nama yang sama tetapi parameter berbeda, meningkatkan fleksibilitas.