Nama & NPM	Topik:	Tanggal:
Rivan Alfatoni G1F024047	**Kelas, Objek, Method**	**12 September 2024**

[No. 1] Identifikasi Masalah:

1.1. Perbaiki pesan kesalahan Contoh 1!

Contoh :

```
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)
```

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

[No.1] Analisis dan Argumentasi

1. Pada soal 1.1 kita mencari kesalahan pada kode program tersebut. Kode tersebut memiliki kesalahan pada constructor Manusia1 yang seharusnya Manusia seperti aturannya bahwa nama constructor harus sama dengan nama class, kode yang diperbaiki :

```
public class Manusia {
   String nama, rambut; //deklarasi variabel nama dan rambut

   //deklarasi constructor dengan parameter nama dan rambut
   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");
   }
}
```

   Maka akan menghasilkan luaran seperti:

```
Nama saya : Putri
 Warna Rambut : hitam
```

2. Saya membuat kode untuk membuat atribut variable dan perilaku behavior karena ingin menganalisa ciri ciri lain manusia seperti tinggi badan,berat badan,dan lain lain.

[No.1] Penyusunan Algoritma dan Kode Program

1.1.1 Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.
Algoritma Kelas Manusia

1. Mulai
2. Deklarasikan kelas Manusia dengan atribut: nama dan rambut yang bertipe data String.
3. membuat constructor Manusia yang menerima dua parameter yaitu nama dan rambut
4. Mengeksekusi Method main dan tampilkan output
5. selesai

1.1.2 Kode program dan luaran

a) Screenshot/ Capture potongan kode dan hasil luaran

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

Luaran:

```
Nama saya : Putri
Warna Rambut : hitam
```

b) Analisa luaran yang dihasilkan

Program menciptakan objek Manusia dengan nama “Putri” dan warna rambut “hitam”. Constructor (metode khusus yang dipanggil saat objek dibuat) dijalankan dengan nilai-nilai yang diberikan (“Putri” dan “hitam”).Pesan yang dicetak menggambarkan informasi tentang nama dan warna rambut dari objek Manusia.

1.2.1. Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

Algoritma Kelas Manusia

a. Mulai
b. Deklarasikan kelas Manusia dengan atribut: nama dan rambut yang bertipe data String.
c. membuat constructor Manusia yang menerima dua parameter yaitu nama dan rambut
d. Mengeksekusi Method main dan tampilkan output
e. selesai

1.2.2. Kode program dan luaran

a. Screenshot/ Capture potongan kode dan hasil luaran

```java
public class Manusia {
    // deklarasi variabel nama, rambut, jenisKelamin, usia, tinggiBadan
    String nama, rambut, jenisKelamin;
    int usia;
    double tinggiBadan;

    // constructor manusia
    public Manusia(String nama, String rambut, String jenisKelamin, int usia, double tinggiBadan) {
        // cetak hasil constructor
        System.out.println("Nama saya : " + nama +
                           "\nWarna Rambut : " + rambut +
                           "\nJenis Kelamin : " + jenisKelamin +
                           "\nUsia : " + usia +
                           "\nTinggi Badan : " + tinggiBadan + " cm");
    }

    // method untuk berjalan
    public void berjalan() {
        // cetak hasil berjalan
        System.out.println("sedang berjalan.");
    }

    // method untuk makan
    public void makan() {
        // cetak hasil makan
        System.out.println("sedang makan.");
    }

    // method main untuk menjalankan program
    public static void main(String[] args) {
        // buat objek manusia dengan nama Rivan Alfatoni
        Manusia satu = new Manusia("Rivan Alfatoni", "hitam", "Laki-laki", 18, 161.6);
        // jalankan method berjalan dan makan
        satu.berjalan();
        satu.makan();
    }
}
```

Luaran kode program:

```
Nama saya : Rivan Alfatoni
Warna Rambut : hitam
Jenis Kelamin : Laki-laki
Usia : 18
Tinggi Badan : 161.6 cm
sedang berjalan.
sedang makan.
```

b. Analisa luaran yang dihasilkan
Program menciptakan objek Manusia dengan nama Rivan alfatoni ,warna rambut hitam, jenis kelamin laki laki, usia 18 dan tinggi 161.6 .

[No.1] Kesimpulan

Evaluasi

Apa konsekuensi/dampak dari kode program yang dibuat?

Pada program itu saya membuat contructor manusia sesuai dengan nama kelas karena ingin membuat objek manusia. Setelah itu saya membuat perilaku/bahvior lain yaitu berjalan dan makan.

[No. 2] Identifikasi Masalah:

2.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

Kode program:

```
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");
```

}
}

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!

[No.2] Analisis dan Argumentasi

1. Pada soal 2.1 kita mencari kesalahan pada kode program tersebut. Kesalahan terjadi pada deklarasi contructor yang seharusnya nama contructor harus sama dengan nama kelas dan tidak ada parameter nilai yang dikirim ke dalam contructor.
2. Saya membuat kode program untuk mencari data variable umur dan jenis kelamin. Merekomendasikan tipe data yang tepat untuk variable tersebut.

[No.2] Penyusunan Algoritma dan Kode Program

2.1.1 Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

Algoritma

a. Mulai
b. Membuat class Ortu
c. Membuat constructor dalam class Ortu
d. Mendeklarasikan method main
e. Selesai

2.1.2 Kode program dan luaran

a. Screenshot/ Capture potongan kode dan hasil luaran

```
1  public class Ortu {
2      //deklarasi constructor  (variabel constructor)
3      public Ortu(String nama, String rambut)   {
4              //nama dan rambut adalah variabel constructor
5           System.out.println(" Nama saya : "+ nama +
6          "\n Warna Rambut : " + rambut);
7  }
8      public static void main (String[] args) {
9          Ortu satu = new Ortu("Putri", "hitam");
10      }
11 }
```

Luaran:

```
Nama saya : Putri
Warna Rambut : hitam
```

b. Analisa Luaran yang dihasilkan

Luaran yang dihasilkan sudah sesuai dengan kode program, karena saya memperbaiki constructor Ortu dan menambah variabel constructor yang berisi nama dan rambut. Maka kode tersebut menghasilkan Luaran Nama saya: Putri Warna Rambut : hitam.

2.2.1. Algoritma

Algoritma adalah langkah-langkah penyelesaian masalah.

Algoritma

a. Mulai
b. Membuat class Ortu
c. Membuat constructor dalam class Ortu
d. Mendeklarasikan method main
e. Selesai

2.2.2. Kode program dan luaran

a. Screenshot/ Capture potongan kode dan hasil luaran

```java
public class Ortu {
    //deklarasi constructor  (variabel constructor)
    public Ortu(String nama, String rambut, int umur, char jenisKelamin)   {
            //nama dan rambut adalah variabel constructor
         System.out.println(" Nama saya : "+ nama +  //cetak nama
        "\n Warna Rambut : " + rambut +  //cetak warna rambut
        "\n Umur saya : " + umur +  //cetak umur
        "\n Jenis Kelamin : " + jenisKelamin);    //cetak jenis kelamin
    }

    public static void main (String[] args) {
            //buat objek dengan nama Putri
            Ortu satu = new Ortu("Putri", "hitam", 25, 'P');
        }
    }

```

Luaran:

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

b. Analisa Luaran yang dihasilkan

Luaran yang dihasilkan sudah sesuai dengan kode program, karena saya memperbaiki constructor Ortu dan menambah variabel constructor yang berisi nama dan rambut. Maka kode tersebut menghasilkan Luaran.

[No.2] Kesimpulan

Jika mendeklarasi method nama yang digunakan harus sama dengan nama class utama, jika nama yang digunakan berbeda dengan nama class utama maka program terssebut akan error.

[No. 3] Identifikasi Masalah

3.1. Evaluasi penyebab kesalahan dan perbaiki kode tersebut!

Kode:

```
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 3:

```
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)
```

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

[No.3] Analisis dan Argumentasi

3.1 Pada kode di atas, terdapat beberapa kesalahan yang mengakibatkan error dalam kompilasi dan eksekusi program. Berikut adalah evaluasi penyebab kesalahan dan cara memperbaikinya:

1. **Nama Kelas dan Konstruktor Tidak Konsisten**:
   - Kelas dideklarasikan dengan nama Manusia, tetapi konstruktornya diberi nama Manusia1. Dalam Java, nama konstruktor harus sama dengan nama kelas.
   - Solusi: Ubah nama konstruktor dari Manusia1 menjadi Manusia.

     ```
     public Manusia(String nama, String rambut) {
             System.out.println("Nama saya : " + nama + "\nWarna Rambut : " + rambut);
     }
     ```
2. **Method sukaNonton Tidak Didefinisikan dengan Benar**:
   - Terdapat dua deklarasi method sukaNonton yang tidak lengkap. Pertama, method sukaNonton yang seharusnya menerima parameter String (untuk judul film), tidak memiliki parameter yang dideklarasikan. Selain itu, ada dua tanda kurung {} yang salah, karena tidak dituliskan sebagai method yang valid.
   - Pada deklarasi kedua, sukaNonton yang mengembalikan jumlah jam (dari episode dan durasi) juga tidak dideklarasikan sebagai method yang benar.
   - **Solusi**: Perbaiki deklarasi method sukaNonton dengan menambahkan parameter dan return value yang sesuai.

     ```
     // Method yang menerima parameter film

     void sukaNonton(String film) {

        System.out.println("Hobi Menonton : " + film);

     }

     // Method yang menerima dua parameter dan mengembalikan nilai integer

     int sukaNonton(int episode, int durasi) {

        return episode * durasi;

     }
     ```
3. **Kesalahan di Main Method:**

- Pada main method, objek dibuat dari kelas Manusia, tetapi method sukaNonton("Drakor") dan sukaNonton(2, 2) tidak dikenali karena belum dideklarasikan dengan benar pada kelas Manusia.
- **Solusi**: Setelah memperbaiki method seperti di atas, main method akan berfungsi dengan benar.

```
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");
}
```

4. **Kesalahan Penulisan Atribut atau Variabel film:**

- Dalam method sukaNonton, variabel film tidak dideklarasikan. Harus dipastikan bahwa variabel ini dideklarasikan dengan benar sebagai parameter method.
- **Solusi**: Deklarasikan film sebagai parameter method seperti yang dijelaskan sebelumnya.

3.2 Untuk memodifikasi method dan konstruktor agar lebih sesuai dengan behavior, misalnya jika ingin menambahkan kebiasaan menonton film dan durasi yang sering dihabiskan:

```
public class Manusia {
   String nama, rambut;
   String hobiMenonton;
   // Konstruktor yang memuat nama dan rambut
   public Manusia(String nama, String rambut, String hobiMenonton) {
      System.out.println("Nama saya : " + nama +
                  "\nWarna Rambut : " + rambut +
                  "\nHobi saya : " + hobiMenonton);
   }
   // Method yang menerima parameter film
   void sukaNonton(String film) {
      System.out.println(nama + " suka menonton film: " + film);
   }
   // Method yang mengembalikan total waktu menonton
   int sukaNonton(int episode, int durasi) {
      return episode * durasi;
   }
   public static void main(String[] args) {
      Manusia satu = new Manusia("Putri", "hitam", "Menonton Film");
      satu.sukaNonton("Drakor");
      int jumlahJam = satu.sukaNonton(5, 1); // contoh 5 episode, durasi 1 jam tiap episode
      System.out.println("Jam nonton = " + jumlahJam + " jam");
   }
}
```

3.3 a) **Constructor Overloading dan Constructor Overriding:**

Constructor Overloading: Merupakan konsep di mana sebuah kelas dapat memiliki lebih dari satu konstruktor dengan nama yang sama, tetapi berbeda parameter (baik jumlah maupun tipe

parameter). Fungsinya untuk memberikan fleksibilitas dalam inisialisasi objek dengan cara yang berbeda.

Contoh:

```
public Manusia(String nama, String rambut, String hobiMenonton) {
    System.out.println("Nama saya : " + nama +
                "\nWarna Rambut : " + rambut +
                "\nHobi saya : " + hobiMenonton);
}
```

Constructor Overriding: Ini sebenarnya bukan konsep yang tepat dalam Java, karena konstruktor tidak dapat di-override (tidak diwarisi). Namun, jika kita membicarakan method yang diwarisi dan diubah perilakunya pada kelas turunan, itu dinamakan method overriding.

b) **Method Overloading dan Method Overriding:**

Method Overloading: Adalah ketika dua atau lebih method dalam satu kelas memiliki nama yang sama tetapi berbeda dalam parameter (jumlah atau tipe parameter). Ini memungkinkan method dengan nama yang sama untuk melakukan fungsi yang berbeda tergantung pada parameter yang diberikan.

```
void sukaNonton(String film) { ... }
int sukaNonton(int episode, int durasi) { ... }
```

Method Overriding: Merupakan proses mendefinisikan ulang method pada kelas anak yang sudah ada pada kelas induk. Tujuannya untuk memberikan implementasi spesifik pada kelas anak yang berbeda dari kelas induk.

Contoh:

```
@Override
void sukaNonton(String film) {
    // implementasi baru
}
```

c) **Method yang Mengembalikan Nilai dan Tidak Mengembalikan Nilai:**

Method yang Mengembalikan Nilai: Method ini menggunakan keyword return dan harus memiliki tipe pengembalian (seperti int, String, dll.).

Contoh:

```
int sukaNonton(int episode, int durasi) {
    return episode * durasi;
}
```

Method yang Tidak Mengembalikan Nilai: Method ini menggunakan keyword void dan tidak mengembalikan nilai apa pun.

Contoh:

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

[No.3] Penyusunan Algoritma dan Kode Program

1. Algoritma
   - Membuat class
   - Mendeklarasi constructor
   - Mendeklarasi metthod utama
2. Kode program dan luaran
   a. Screenshot/ Capture potongan kode dan hasil luaran

(Coding 3.1) :

```
1  public class Manusia {
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
12     void sukaNonton(String film) {
13         System.out.println(" Hobi Menonton : " + film);
14     }
15
16     int sukaNonton(int episode, int durasi) {
17         return episode*durasi;
18     }
19
20     //deklarasi method utama
21     public static void main( String[] args) {
22             Manusia satu = new Manusia("Putri", "hitam");
23             satu.sukaNonton("Drakor");
24             int jumlahJam = satu.sukaNonton(2, 2);
25             System.out.println("Jam nonton = " +jumlahJam + " jam");
26     }
27 }
```

(Luaran 3.1)

```
 Nama saya : Putri
 Warna Rambut : hitam
 Hobi Menonton : Drakor
Jam nonton = 4 jam
```

(Coding 3.2)

```java
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 sukaMain(String film) {
        System.out.println(" Hobi Main Game : " + film);
    }

    int sukaMain(int match, int durasi) {
        return match*durasi;
    }

    //deklarasi method utama
    public static void main( String[] args) {
            Manusia satu = new Manusia("Rivan", "hitam");
            satu.sukaMain("EPEP");
            int jumlahJam = satu.sukaMain(2, 10);
            System.out.println("Total Durasi Main = " +jumlahJam + " Menit");
    }
}
```

(Luaran 3.2)

```
Nama saya : Rivan
Warna Rambut : hitam
Hobi Main Game : EPEP
Total Durasi Main = 20 Menit
```

(Coding 3.3)

b. Luaran sesuai dengan program yang dijalankan

[No.3] Kesimpulan

Analisa

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

Permasalahan yang dihadapi adalah bagaimana membuat sebuah kelas Java yang dapat menampilkan informasi tentang seseorang, seperti nama dan hobi mereka. Kode juga perlu menghitung total aktivitas berdasarkan parameter tertentu, jumlah episode yang ditonton atau jumlah bungkus makanan yang dimakan. Tantangan lainnya adalah menggunakan metode dengan parameter yang berbeda, sehingga konsep method overloading dapat diterapkan.

[No. 4] Identifikasi Masalah

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!

[No.4] Analisis dan Argumentasi

1. Rincian Solusi yang Diusulkan

4.1. Method dalam Kelas Ortu:

- Kelas Ortu memiliki dua fungsi utama:

- sukaMenonton(String a), yang menampilkan aktivitas menonton yang sedang dilakukan.
- sukaMembaca(String a), yang menampilkan aktivitas membaca tertentu.

Method dalam Kelas Anak:

- Kelas Anak yang mewarisi dari kelas Ortu memiliki beberapa method berikut:
  - sukaMenonton(int a, String b), yang menambahkan parameter angka (int a) untuk menentukan jam spesifik aktivitas menonton.
  - sukaMenonton(String a), yang meng-*override* method yang sama dari kelas induk, tetapi sebenarnya hanya mengulang fungsi yang sudah ada dalam kelas Ortu.
  - sukaMembaca(String a), juga meng-*override* method dari kelas induk, tetapi hanya menampilkan pesan yang sedikit berbeda tanpa menambahkan fungsionalitas baru.

4.2. Penyederhanaan Metode:

- Hilangkan metode-metode yang tidak memberikan tambahan fungsi baru di kelas Anak. Jika method dalam kelas induk (Ortu) sudah sesuai, tidak perlu dilakukan overriding di kelas Anak. Ini akan merampingkan kode dan meningkatkan efisiensi program.

2. Analisis Solusi dan Keterkaitannya dengan Masalah

- Solusi ini menangani masalah inheritance dengan lebih efisien. Kelas Anak dapat mewarisi method-method dari kelas Ortu tanpa mengulang kode yang sudah ada, kecuali dalam kasus tertentu di mana perubahan dibutuhkan, seperti pada method sukaMenonton dengan parameter tambahan. Dengan cara ini, pengulangan kode yang tidak diperlukan dihindari, membuat program lebih mudah untuk dikembangkan dan lebih optimal dalam eksekusinya.

[No.4] Penyusunan Algoritma dan Kode Program

1. Rancang desain solusi atau algoritma
   - Membuat kelas
   - Mendeklarasi construktor
   - Mendeklarasi method utama
2. Kode program dan luaran

a. Screenshot/ Capture potongan kode dan hasil luaran

```
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("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 {
    @Override
    void sukaMenonton(String a) {           // method induk spesifik
        System.out.println("Nonton " + a);
    }
    @Override
    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("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:

Luaran yang dihasilkan sudah sesuai dengan perintah , waktu terpangkas dari 1.8s ke 1.2s

[No 4] Kesimpulan

Peningkatan Efisiensi dengan menghapus method redundant sukaMenonton(String a) di class Anak, kita mengurangi duplikasi kode dan membuat program lebih sederhana dan mudah dipelihara.

Refleksi

Praktikum kali ini mungkin sedikit susah karena sudah mulai banyak tugas, coding sudah mulai ke level medium. Mungkin praktikum selanjutnya untuk di jelaskan lebih dalam mengenai kelas, objek dan method.