Latihan 1 Pengenalan Tipe Data

Nama & NPM	Topik:	Tanggal:
Muhammad Arya Nugraha G1F024002	Kelas, Method, Contructor	19 September 2024

[No.1] Identifikasi Masalah:

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

     // deklarasi constructor tanpa parameter
     public Manusia() {
          System.out.println("Kelas Manusia tanpa nama");
     }
  }
```

Latihan 1:

1.1. Analisa ciri-ciri umum Kelas Manusia yang dapat menjadi
   a. atribut variabel, dan
   b. perilaku/ behavior untuk method!

[No.1] Analisis dan Argumentasi

1) Atributnya adalah nama dan rambut.
2) Method yang bisa ditambahkan adalah berjalan() dan bernapas()

[No.1] Penyusunan Algoritma dan Kode Program

1) Algoritma
   - Mulai program.
   - Deklarasi variavel atribut.
   - Deklarasi contructor.
   - Akhiri program.
2) Tuliskan kode program dan luaran

```
1  public class Manusia {  // deklarasi kelas
2      // deklarasi variabel
3      String nama;
4      String rambut;
5
6       // deklarasi constructor tanpa parameter
7      public Manusia() {
8          System.out.println("Kelas Manusia tanpa nama");
9      }
10 }
```

```
java -cp /tmp/2nyc6jcen6/Manusia
ERROR!
Error: Main method not found in class Manusia, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

[No.1] Kesimpulan

a) Analisa

Program tidak memiliki output karena tidak ada main method

Latihan 2 Pengenalan Tipe Data

Nama & NPM	Topik:	Tanggal:
Muhammad Arya Nugraha G1F024002	Kelas, Method, Contructor	19 September 2024

[No.2] Identifikasi Masalah:

```
public class Ortu {
    //deklarasi constructor
    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");
 }    }
```

Luaran 2:

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

Latihan 2:

2.1. Susun kembali kode di contoh 2 dengan menambahkan data ciri-ciri Anda di dalam variabel constructor!

2.2. Apabila nanti Anda akan memiliki keturunan, analisa sifat (atribut), constructor, dan perilaku positif (behavior) apa yang akan diturunkan?

[No.2] Analisis dan Argumentasi

1. Menambahkan warna rambut dan tinggi badan ke kode.
2. Sifat yang akan diturunkan adalah bersikap baik.

[No.2] Penyusunan Algoritma dan Kode Program

1. Rancang desain solusi atau algoritma
    - Mulai program.
    - Buat class ortu.
    - Buat constructor.
    - Buat method.
    - Print di main method.
    - Akgiri program.
2. Tuliskan kode program dan luaran

```
1  public class Ortu {
2      // deklarasi constructor dengan tambahan variabel
3      public Ortu(String nama, String rambut, String tinggi, String kulit) {
4          // nama, rambut, tinggi, dan kulit adalah variabel constructor
5          System.out.println("Nama saya : " + nama +
6          "\nWarna Rambut : " + rambut +
7          "\nTinggi : " + tinggi +
8          "\nWarna Kulit : " + kulit);
9      }
10
11     public void bersikapBaik() {
12         System.out.println("Saya selalu berusaha untuk bersikap baik dan jujur.");
13     }
14
15
16     public static void main(String[] args) {
17         Ortu satu = new Ortu("Arya", "hitam", "163 cm", "sawo matang");
18         satu.bersikapBaik();
19     }
20 }
21
```

```
java -cp /tmp/PGTrJoXzi7/Ortu
Nama saya : Arya
Warna Rambut : hitam
Tinggi : 163 cm
Warna Kulit : sawo matang
Saya selalu berusaha untuk bersikap baik dan juju

=== Code Execution Successful ===
```

[No.2] Kesimpulan

1. Analisa
    a) Program sudah mampu menampilkan data ortu beserta kebiasan yang akan diwariskan.

Latihan 3 Pengenalan Tipe Data

Nama & NPM	Topik:	Tanggal:
Muhammad Arya Nugraha G1F024002	Kelas, Method, Contructor	19 September 2024

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

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

Luaran 3:

```
 Nama saya : Putri
 Warna Rambut : hitam
 Hobi Menonton : Drakor
```

Latihan 3:

3.1. Analisa perbedaan deklarasi constructor, method, dan method utama!
3.2. Tentukan kapan Anda perlu menggunakan constructor dan method?
3.3. Uraikan perbedaan berikut:
   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

1. Contructor dideklarasikan dengan public Manusia, method dengan void sukaMenonton, dan method utama dengan public static void main(String[] args)
2. Contructor digunakan saat ingin menginisialisasikan nilai objek. Dan method dignakan saat ingin memberikan aksi pada objek.
3. Contructor overloading adalah saat kelas memiliki beberapa constructor dengan nama yang sama tetapi parameter yang berbeda satu sama lainnya. Method overloading adalah saat sebuah kelas memiliki beberapa method dengan nama yang sama tetapi parameter yang berbeda. Method overriding adalah saat method mengganti implementasi method dari superclass dalam subclass dengan implementasi yang berbeda. Method overriding harus memiliki nama method yang sama, parameter yang sama, dan tipe pengembalian yang sama. Dan method yang tidak mengembalikan nilai ditandai dengan key word ‘void’.

[No.3] Penyusunan Algoritma dan Kode Program

1. Rancang desain solusi atau algoritma
   - Mulai program.
   - Kelas.
   - Atribut.

- Constructor.
- Method.
- Main method.
- Akhiri program

2. Tuliskan kode program dan luaran

```
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     //deklarasi method utama
17     public static void main( String[] args) {
18             Manusia satu = new Manusia("Putri", "hitam");
19             satu.sukaNonton("Drakor");
20     }
21 }
```

```
java -cp /tmp/LeHaPaKQkf/Manusia
 Nama saya : Putri
 Warna Rambut : hitam
 Hobi Menonton : Drakor

=== Code Execution Successful ===
```

[No.3] Kesimpulan

1. Analisa
   a) Constructor berhasil mencetak informasi nama dan warna rambut.
   b) Method sukaNonton berfungsi dengan baik untuk mencetak hobi menonton yang diberikan sebagai parameter.
   c) Output sesuai dengan apa yang diharapkan berdasarkan kode yang diberikan.

Latihan 4 Pengenalan Tipe Data

Nama & NPM	Topik:	Tanggal:
Muhammad Arya Nugraha G1F024002	Kelas, Method, Contructor	19 September 2024

[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
    }
}
```

Luaran 4:
Sifat Orang Tua :
Nonton Berita
Suka Baca Koran

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

Latihan 4:

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

4.2. Ubahlah Contoh 4 dengan menambahkan objek anak dengan method yang berbeda!

[No.4] Analisis dan Argumentasi

1. Pada kelas `Anak` yang mewarisi dari kelas `Ortu`, terdapat method yang diubah dan ditambahkan dibandingkan dengan method di kelas `Ortu`. Kelas `Anak` memiliki overload pada method `sukaMenonton`, yang menambah varian method dengan parameter `int` dan `String` selain method `sukaMenonton` yang sama dengan kelas `Ortu`. Selain itu, `Anak` mengoverride method `sukaMenonton(String a)` dan `sukaMembaca(String a)` untuk memberikan implementasi yang lebih spesifik. Dengan demikian, `Anak` menambahkan fungsionalitas baru dan memodifikasi perilaku method yang diwarisi dari `Ortu`.
2. Method Baru sukaOlahraga(String olahraga): Ditambahkan ke kelas Anak untuk memberikan fungsi tambahan yang tidak ada di kelas Ortu.

[No.4] Penyusunan Algoritma dan Kode Program

1. Rancang desain solusi atau algoritma
   - Mulai Program.
   - Kelas `Ortu`.
   - Method `main` pada `Ortu`.
   - Kelas `Anak` (extends `Ortu`).
   - Method `main` pada `Anak`.
   - Akhiri Program.
2. Tuliskan kode program dan luaran

```java
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
    }
}
```

```
java -cp /tmp/bPlZ8CzTXs/Ortu
Sifat Orang Tua :
Nonton Berita
Suka Baca Koran

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

=== Code Execution Successful ===
```

[No.4] Kesimpulan

1. Analisa
    1. Pewarisan dan Overriding: Kelas Anak mewarisi method dari kelas Ortu dan mengoverride method sukaMenonton(String a) dan sukaMembaca(String a). Ini memungkinkan Anak untuk memperbarui perilaku method yang diwarisi.
    2. Overloading: Kelas Anak juga menambahkan overload baru untuk method sukaMenonton dengan parameter (int a, String b), yang memungkinkan method ini dipanggil dengan tipe parameter yang berbeda.

Refleksi

(Saya Belajar Menggunakan kelas, method, dan contructor di Java)