

| | | |
|---|-------------------|-------------------|
| Nama & NPM | Topik: | Tanggal: |
| HANIFAH AZIZAH G1F024037 | KELAS JAVA | 18/09/2024 |
| [No.1] Identifikasi Masalah: | | |
| UNIT 1 KELAS (CLASS) Contoh 1: <pre> public class Manusia { // deklarasi kelas // deklarasi variabel String nama; String rambut; // deklarasi constructor tanpa parameter public Manusia() { System.out.println("Kelas Manusia tanpa nama"); } } </pre> <p>Dari kode tersebut, didapatkan soal berupa:</p> <ol style="list-style-type: none"> Analisa ciri-ciri umum Kelas Manusia yang dapat menjadi <ol style="list-style-type: none"> atribut variabel, dan perilaku/ behavior untuk method! | | |
| [No.1] Analisis dan Argumentasi | | |
| <ol style="list-style-type: none"> Dalam java, variable adalah tempat penyimpanan yang diberi nama dan dapat diisi dengan data, setelah itu dapat dimanipulasi. Sedangkan atribut adalah variable yang didefinisikan di dalam kelas. Atribut ini biasa dikenal dengan tipe data primitive seperti char, int, dan string. Atribut menyimpan deklarasi dan inisialisasi ciri-ciri dari objek. Contoh atribut variable dalam kode tersebut adalah nama, rambut. Ini bertindak sebagai konstruktor. Perilaku atau behavior untuk method merupakan kegiatan atau aksi yang dilakukan oleh objek. Contoh method dalam kode adalah seperti berbicara, makan. | | |
| [No.1] Penyusunan Algoritma dan Kode Program | | |
| Algoritma <ol style="list-style-type: none"> Mulai. Deklarasi dan inisialisasi nama kelas 'manusia'. Deklarasi dan inisialisasi variable 'String nama' Deklarasi dan inisialisasi variable 'String rambut' Deklarasi constructor tanpa parameter Tampilkan "kelas manusia tanpa nama" Akhiri program | | |
| [No.1] Kesimpulan | | |
| <ol style="list-style-type: none"> Evaluasi <ol style="list-style-type: none"> Apa konsekuensi dari skenario pemrograman ini? Kode yang disajikan tidak memiliki parameter untuk konstruktor yang dibuat sehingga luaran yang dihasilkan tidak dapat menangkap data yang dimasukkan. Method utama juga tidak dideklarasikan sehingga dapat menyebabkan error. | | |

| | | |
|-------------------------------------|-------------------|-------------------|
| Nama & NPM | Topik: | Tanggal: |
| HANIFAH AZIZAH G1F024037 | KELAS JAVA | 18/09/2024 |

[No.2] Identifikasi Masalah:

UNIT 2 OBJEK

Contoh 2: Salin dan tempel kode program berikut ke Eclipse atau JDoodle.

```
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

Dari kode tersebut, didapatkan soal berupa:

1. Susun kembali kode di contoh 2 dengan menambahkan data ciri-ciri Anda di dalam variabel constructor!
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. Kode yang telah ada perlu disusun dengan penambahan ciri-ciri saya sendiri di dalam kode tersebut. Sehingga, saya menambahkan lebih banyak atribut seperti warna mata dan warna kulit di dalam kode.
2. Jika nantinya memiliki keturunan, maka yang akan diturunkan berupa warna mata hitam, warna rambut hitam, warna kulit kuning langsung dan behavior hobi membaca.

[No.2] Penyusunan Algoritma dan Kode Program

Algoritma

1. Mulai.
2. Deklarasi dan inisialisasi nama kelas 'public class ortu'
3. Deklarasi variable konstruktor nama, rambut, mata, kulit.
4. Buat method main.
5. Buat objek satu dari kelas ortu.
6. Tampilkan parameter yang telah menerima data.
7. Akhiri program.

Main.java

Share

Run

```

1- public class ortu {
2
3+  public ortu(String nama, String rambut, String mata, String kulit) {
4
5-      System.out.println(" Nama saya: "+ nama +
6          "\n Warna Rambut: " + rambut + "\n Warna Mata: " + mata + "\n Warna Kulit: " + kulit
7          );
8  }
9+  public static void main (String[] args) {
10      ortu satu = new ortu("Hanifah Azizah", "hitam", "hitam", "kuning langsung");
11
12  }
13
14 }
```

Output

```

java -cp /tmp/Se8Iy6hiaE/ortu
Nama saya: Hanifah Azizah
Warna Rambut: hitam
Warna Mata: hitam
Warna Kulit: kuning langsung

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

| |
|---|
| |
| [No.2] Kesimpulan |
| <ol style="list-style-type: none">1. Evaluasi<ol style="list-style-type: none">b) Apa konsekuensi dari skenario pemrograman ini?<p>Program ini perlu disusun dengan teliti karena banyak method yang perlu ditambahkan dan banyak variable yang harus dideklarasikan secara tepat. Jika kode program tidak sama antara variable, method, dan parameter maka kode akan menampilkan luaran error.</p> |

| | | |
|-------------------------------------|-------------------|-------------------|
| Nama & NPM | Topik: | Tanggal: |
| HANIFAH AZIZAH G1F024037 | KELAS JAVA | 19/09/2024 |

[No.3] Identifikasi Masalah:

UNIT 3 METHOD

Contoh 3: Salin dan tempel kode program berikut ke Eclipse atau JDoodle.

```
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
```

Dari kode tersebut, didapatkan soal berupa:

1. Analisa perbedaan deklarasi constructor, method, dan method utama!
2. Tentukan kapan Anda perlu menggunakan constructor dan method?
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. Berdasarkan kode pada soal, perbedaan konstruktor, method, dan method utama terdapat pada nama, tipe return, tujuan penggunaan. Konstruktor harus memiliki nama yang sama dengan nama kelas, method memiliki nama yang berbeda dengan kelas, sedangkan method utama harus memiliki nama berupa main.
Untuk tipe return atau return type, konstruktor tidak memiliki tipe, method menggunakan tipe void, method utama menggunakan void.
Tujuan dari konstruktor adalah untuk menginisialisasi objek, method untuk menjalankan suatu tindakan tertentu, method utama untuk sebagai titik awal eksekusi program.
2. Konstruktor sebaiknya digunakan saat ingin menginisialisasi objek, menerima parameter, dan overloading. Sedangkan method digunakan jika ingin mendefinisikan perilaku objek, mengubah nilai atribut, dan membuat kode menjadi lebih terstruktur.
3. Constructor Overloading memungkinkan lebih dari satu constructor dengan parameter yang berbeda. Sedangkan constructor overriding tidak ada dalam konteks kelas.


Method Overloading memungkinkan lebih dari satu method dengan nama yang sama tetapi parameter berbeda. Sedangkan method overriding menggantikan implementasi method dari superclass.

Method yang mengembalikan nilai memberikan hasil setelah eksekusi. Sedangkan method tidak mengembalikan nilai melakukan tindakan tanpa hasil.

[No.3] Penyusunan Algoritma dan Kode Program

Algoritma

1. Mulai.
2. Deklarasi nama kelas 'manusia'
3. Deklarasi atribut manusia dalam variable nama, rambut.
4. Deklarasi konstruktor 'public manusia'
5. Tampilkan data parameter nama, rambut.
6. Deklarasi method 'void sukaNonton'
7. Tampilkan parameter nonton.
8. Deklarasi method utama 'public static void main'
9. Tampilkan objek manusia satu dengan data atribut manusia.
10. Akhiri program.

| Main.java | Run | Output |
|---|---|--|
| <pre>1- public class Manusia { 2 //deklarasi atribut Manusia dalam variabel 3 String nama, rambut; 4 5 //deklarasi konstruktor 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 }</pre> |  | <pre>java -cp /tmp/TfcACX1AP6/Manusia Nama saya : Putri Warna Rambut : hitam Hobi Menonton : Drakor === Code Execution Successful ===</pre> |

[No.3] Kesimpulan

2. Evaluasi

- c) Apa konsekuensi dari skenario pemrograman ini?

Kode program berisi tiga deklarasi dengan masing-masing method yang berbeda untuk fungsi yang berbeda pula. Sehingga penggunaannya perlu disesuaikan agar data yang ada bisa lebih efisien untuk disimpan dan lebih mudah untuk dimodifikasi sesuai dengan method yang digunakan. Luaran yang dihasilkan menampilkan atribut variable dan elemen data yang tersimpan.

| | | |
|-------------------------------------|-------------------|-------------------|
| Nama & NPM | Topik: | Tanggal: |
| HANIFAH AZIZAH G1F024037 | KELAS JAVA | 19/09/2024 |

[No.4] Identifikasi Masalah:

UNIT 4 EXTENDS

Contoh 4: Salin dan tempel kode program berikut ke JDoodle. Kemudian catat waktu eksekusinya.

```
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 objek0 = new Ortu(); // memanggil objek induk
    objek0.sukaMenonton("Berita"); // memanggil sifat spesifik induk
    objek0.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 objek0 = new Ortu(); // memanggil objek induk
    objek0.sukaMenonton("Berita"); // memanggil sifat spesifik induk
    objek0.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

Dari kode tersebut, didapatkan soal berupa:

1. Bandingkan method yang dimiliki `class Anak extends Ortu` dengan method di `class Ortu!`
2. Ubahlah Contoh 4 dengan menambahkan objek anak dengan method yang berbeda!

[No.4] Analisis dan Argumentasi

1. Pada class `Ortu`, terdapat `'void sukaMenonton(String a):'` method ini mencetak string yang menunjukkan jenis tontonan yang merupakan method spesifik. Kemudian terdapat `'void sukaMembaca(String a):'` method ini mencetak string yang menunjukkan jenis bacaan yang merupakan method umum yang bisa diubah oleh subclass. Sedangkan pada class `Anak`, terdapat `'void sukaMenonton(int a, String b):'` method ini berbeda dari method di `Ortu` karena turut mencetak jam setelah jenis tontonan dan merupakan method spesifik untuk class `Anak`. Selanjutnya terdapat `'void sukaMenonton(String a):'` method ini sama dengan method class `Ortu`, tetapi di-override untuk memberikan kemungkinan yang lebih spesifik bagi class `Anak`. Adapula `'void sukaMembaca(String a):'` method ini juga di-override, meskipun fungsinya tetap sama dengan class `Ortu` pada subclass `Anak`.
2. Pada kode awal, ditambahkan lagi method baru yang berisi olahraga kesukaan anak yang berbeda dengan class `Ortu`.

[No.4] Penyusunan Algoritma dan Kode Program

Algoritma

1. Mulai.
2. Deklarasikan nama kelas `Ortu`.
3. Deklarasikan method `void` untuk `suka menonton` pada kelas `Ortu`.
4. Deklarasikan method `void` `suka membaca` pada kelas `Ortu`.
5. Deklarasikan method `public static void` pada kelas `Ortu` untuk atribut sifat `Ortu`.
6. Deklarasikan method baru untuk atribut sifat `Anak`.
7. Deklarasikan kelas `extend` `Anak`.
8. Deklarasikan method `suka menonton` pada class `Anak` dengan atribut `int` dan `string` secara spesifik.
9. Deklarasikan method `suka membaca` pada class `Anak` yang dapat dimodifikasi pada subclass.
10. Deklarasikan method baru yang berbeda untuk class `Anak`.
11. Tampilkan hasil.
12. Akhiri program.

| | |
|--|--|
| <pre>Main.java 1- public class Ortu { 2- void sukallenonton(String a) { 3 System.out.println("Nonton " + a); 4 } 5 6- void sukallenbaca(String a) { 7 System.out.println("Suka Baca " + a); 8 } 9 10- public static void main(String[] args) { 11 System.out.println("Sifat Orang Tua :"); 12 Ortu objekO = new Ortu(); 13 objekO.sukallenonton("Berita"); 14 objekO.sukallenbaca("Koran"); 15 16 System.out.println("\nSifat Anak :"); 17 Anak objekA = new Anak(); 18 objekA.sukallenonton(9, "Film Drakor"); 19 objekA.sukallenbaca("Komik One Piece"); 20 objekA.sukaOlahraga("Sepak Bola"); // Memanggil metode baru 21 } 22 } 23 24- class Anak extends Ortu { 25- void sukallenonton(int a, String b) { 26 System.out.println("Nonton Jam " + a + " Malam " + b); 27 } 28 29- void sukallenbaca(String a) { 30 System.out.println("Suka Baca " + a); 31 } 32 33 // Menambahkan metode baru yang berbeda 34- void sukaOlahraga(String olahraga) { 35 System.out.println("Suka Olahraga: " + olahraga); 36 } 37 }</pre> | <pre>Output java -cp ./tmp/iqo6H86lmW/Ortu Sifat Orang Tua : Nonton Berita Suka Baca Koran Sifat Anak : Nonton Jam 9 Malam Film Drakor Suka Baca Komik One Piece Suka Olahraga: Sepak Bola === Code Execution Successful ===</pre> |
|--|--|

[No.4] Kesimpulan

3. Evaluasi

d) Apa konsekuensi dari skenario pemrograman ini?

Penggunaan metode baru pada class anak akan berpengaruh pada luaran yang dihasilkan setelah atribut dipanggil pada class extend anak. Untuk itu penyusunan kode menggunakan method baru pada class anak tidak berhubungan dengan class ortu tetapi tidak mengganggu class ortu itu sendiri.

Refleksi

Pada minggu pengerjaan tugas ini saya merasa sedikit kebingungan karena penggunaan method yang beragam dan banyak susunan kode yang berisiko menampilkan error jika tidak diteliti dengan baik. Materi yang didapatkan memang berdasarkan kemampuan dasar tetapi menurut saya ini cukup kompleks dan butuh focus penuh dalam pengerjaannya. Soal-soal yang diberikan berkaitan dengan materi class, method, objek, dan extends.