Nama & NPM	Topik:	Tanggal:
Fherta Afrisidenta G1F024003 **Michelia Erza Annadhira G1F024035** **Bagas Satrio Winata G1F024059**	**Atribut, Method, dan** **Construktor**	**14 September 2024**

1.a Identifikasi Masalah:

1) Uraikan permasalahan dan variabel

1. Apabila diketahui kelas induk adalah Mahasiswa dan Kelas anak adalah turunan dari mahasiswa maka:

- (a) Analisa atribut, method, dan constructor dari kelas induk Mahasiswa dan kelas anak (turunan dari Mahasiswa)!
- (b) Evaluasi perbedaan kelas induk Mahasiswa dan kelas anak (turunan dari Mahasiswa)!
- (c) Rekomendasi atribut, method, dan constructor yang bisa digunakan bersama kelas induk dan kelas anak!
- (d) Desain kode program Java yang berisi atribut, method, dan constructor dari kelas induk Mahasiswa dan kelas anak (turunan dari Mahasiswa)!

2) Rincikan sumber informasi yang relevan
https://www.youtube.com/watch?v=60IdOc8m8Es
https://www.youtube.com/watch?v=6qULMlcv-eg

1.b Analisis dan Argumentasi

a) Analisa atribut, method, dan constructor dari kelas induk Mahasiswa dan kelas anak (turunan dari Mahasiswa)

Pada kode program ini, kami tidak memasukkan atribut (variabel) yang didefinisikan baik di kelas induk UniversitasBengkulu maupun di kelas anak SistemInformasi. Biasanya atribut adalah variabel yang digunakan untuk menyimpan data, seperti nama mahasiswa atau jurusan. Dari segi method :

- Organisasi(String a), mencetak pesan tentang organisasi apa yang mahasiswa senang pelajari.
- sukaBelajar(String a), mencetak pesan tentang hal apa yang mahasiswa senang pelajari

Kelas anak SistemInformasi mewarisi kedua method tersebut, tetapi di sini kita mengoverride method-method tersebut, meskipun method dari induk diwariskan, kelas anak bisa memodifikasinya sesuai kebutuhan. Dalam contoh yang kami buat ini, method organisasi dan sukaBelajar di kelas anak sebenarnya tidak berubah sama sekali, hanya ditulis ulang

Constructornya, baik kelas induk maupun kelas anak tidak memiliki constructor yang eksplisit, jadi Java otomatis menyediakan constructor default. Yang berarti, kita tetap bisa membuat objek dari kedua kelas ini meskipun tidak mendefinisikan constructor secara manual.

b) Evaluasi perbedaan kelas induk Mahasiswa dan kelas anak (turunan dari Mahasiswa)!

Perbedaan utamanya ada pada method overriding. Ketika kita memanggil method organisasi atau sukaBelajar dari objek kelas anak, Java akan menjalakan versi method yang ada di kelas anak, bukan yang di kelas induk. Ini adalah konsep overriding, di mana kelas anak dapat menyediakan versi sendiri dari method yang diwariskan.

Dalam hal pewarisan, kelas anak otomatis mewarisi semua method non-private dan kelas induk. Dalam kode program ini, kelas anak SistemInformasi mewarisi Organisasi dan sukaBelajar,

tapi karena di overriding, hasilnya akan sesuai dengan difinisi di kelas anak.

c) Rekomendasi atribut, method, dan constructor yang bisa digunakan bersama kelas induk dan kelas anak!

Agar lebih rapi dan bisa digunakan bersama antara kelas induk dan kelas anak, berikut beberapa rekomendasi yang dapat kami berikan :

- Atribut bersama, kalian bisa menambahkan atribut di kelas induk yang sifatnya umum untuk semua mahasiswa, seperti nama,NPM, dan jurusan.
- Method bersama, method sukaBelajar sebaiknya dibiarkan saja di kelas induk, tanpa di-override di kelas anak, kecuali ada alasan khusus untuk mengubah perilakunya. Ini akan mengurangi duplikasi kode.
- Constructor, digunakan untuk menginisialisasi objek, dalam kode program yang kami buat ini, tidak ada constructor khusus yang dibuat di kelas induk atau anak. Secara default, Java akan menggunakan constructor tanpa parameter (default constructor) jika tidak ada constructor yang didefinisikan.

d) Desain kode program Java yang berisi atribut, method, dan constructor dari kelas induk Mahasiswa dan kelas anak (turunan dari Mahasiswa)!

- Desain programnya akan kami jelaskan lebih lanjut di bagian penyusunan algoritma dan kode program

1.c Penyusunan Algoritma dan Kode Program

Disini kami akan menjelaskan point D, tentang desain kode program Java yang berisi atribut, method, dan constructor dari kelas induk Mahasiswa dan kelas anak

1) Algoritma
   - Mulai
   - Buat nama kelas menjadi Universitas Bengkulu
   - Buat method induk yaitu Organisasi dan sukaBelajar
   - Buat objek kelas induk menjadi BEM dan Speakup
   - Buat objek kelas anak menjadi HIMASIF dan Coding
   - Pada public static void dibuat sesuai objek kelas induk maupun kelas anak
   - Run kode program
   - Selesai

2) Tuliskan kode program dan luaran
   a) Beri komentar pada kode

```
public class UniversitasBengkulu {      // membuat kelas induk
  void Organisasi(String a) {    // method induk spesifik
    System.out.println("Ikut Organisasi " + a);
  }
  void sukaBelajar(String a) {     // method induk umum bisa diubah anak
    System.out.println("Suka Belajar " + a);
  }

public static void main(String [] args) {
    System.out.println("Kegiatan Mahasiswa Universitas Bengkulu :");
    UniversitasBengkulu objekO = new UniversitasBengkulu();     // memanggil objek
induk
    objekO.Organisasi("BEM");    // memanggil sifat spesifik induk
    objekO.sukaBelajar("Speakup");     // memanggil method dengan variabel dapat
diubah

    System.out.println("\nKegiatan Mahasiswa Sistem Informasi :");
```

```
    SistemInformasi objekA = new SistemInformasi();    //memanggil objek anak
    objekA.Organisasi("HIMASIF");       //memanggil sifat spesifik anak yang diturunkan
induk
    objekA.sukaBelajar("Coding"); //memanggil method ke induk yang otomatis
diturunkan tanpa deklarasi ulang di anak
}   }

class SistemInformasi extends UniversitasBengkulu {
  void Organisasi(String a) {          // method induk spesifik
      System.out.println("Ikut Organisasi " + a);
  }
  void sukaBelajar(String a) {     // method induk umum bisa diubah anak
      System.out.println("Suka Belajar " + a);
  }

public static void main(String [] args) {
    System.out.println("Kegiatan Mahasiswa Universitas Bengkulu :");
    UniversitasBengkulu objekO = new UniversitasBengkulu();     // memanggil objek
induk
    objekO.Organisasi("BEM");       // memanggil sifat spesifik induk
    objekO.sukaBelajar("Speakup");     // memanggil method dengan variabel dapat
diubah

    System.out.println("\n Sifat Anak :");
    SistemInformasi objekA = new SistemInformasi();    //memanggil objek anak
    objekA.Organisasi("HIMASIF");       //memanggil sifat spesifik anak yang diturunkan
induk
    objekA.sukaBelajar("Coding"); //memanggil method ke induk yang otomatis
diturunkan tanpa deklarasi ulang di anak
    }
}
```

b) Uraikan luaran yang dihasilkan

Kegiatan Mahasiswa Universitas Bengkulu :
Ikut Organisasi BEM
Suka Belajar Speakup

Kegiatan Mahasiswa Sistem Informasi :
Ikut Organisasi HIMASIF
Suka Belajar Coding

c) Screenshot/ Capture potongan kode dan hasil luaran

- Kode program

```java
public class UniversitasBengkulu {       // membuat kelas induk
  void Organisasi(String a) {    // method induk spesifik
    System.out.println("Ikut Organisasi " + a);
  }
  void sukaBelajar(String a) {     // method induk umum bisa diubah anak
    System.out.println("Suka Belajar " + a);
  }

public static void main(String [] args) {
    System.out.println("Kegiatan Mahasiswa Universitas Bengkulu :");
    UniversitasBengkulu objekO = new UniversitasBengkulu();     // memanggil objek induk
    objekO.Organisasi("BEM");    // memanggil sifat spesifik induk
    objekO.sukaBelajar("Speakup");     // memanggil method dengan variabel dapat diubah

    System.out.println("\nKegiatan Mahasiswa Sistem Informasi :");
    SistemInformasi objekA = new SistemInformasi();    //memanggil objek anak
    objekA.Organisasi("HIMASIF");        //memanggil sifat spesifik anak yang diturunkan induk
    objekA.sukaBelajar("Coding"); //memanggil method ke induk yang otomatis diturunkan tanpa d
}    }

class SistemInformasi extends UniversitasBengkulu {
  void Organisasi(String a) {            // method induk spesifik
       System.out.println("Ikut Organisasi " + a);
  }
  void sukaBelajar(String a) {     // method induk umum bisa diubah anak
       System.out.println("Suka Belajar " + a);
  }

public static void main(String [] args) {
    System.out.println("Kegiatan Mahasiswa Universitas Bengkulu :");
    UniversitasBengkulu objekO = new UniversitasBengkulu();     // memanggil objek induk
    objekO.Organisasi("BEM");        // memanggil sifat spesifik induk
    objekO.sukaBelajar("Speakup");     // memanggil method dengan variabel dapat diubah

    System.out.println("\n Sifat Anak :");
    SistemInformasi objekA = new SistemInformasi();    //memanggil objek anak
    objekA.Organisasi("HIMASIF");        //memanggil sifat spesifik anak yang diturunkan induk
    objekA.sukaBelajar("Coding"); //memanggil method ke induk yang otomatis diturunkan tanpa d
    }
}
```

- Output

```
Kegiatan Mahasiswa Universitas Bengkulu :
Ikut Organisasi BEM
Suka Belajar Speakup

Kegiatan Mahasiswa Sistem Informasi :
Ikut Organisasi HIMASIF
Suka Belajar Coding
```

1.d Kesimpulan

1) Analisa

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

Dari masalah yang kami hadapi dalam kode program ini, focus utamanya adalah bagaimana cara kerja hubungan antara kelas induk UniversitasBengkulu dan kelas anak SistemInformasi. Kelas anak mewarisi method dari kelas induk dan dalam beberapa kasus, method tersebut bisa diubah atau disesuaikan melalui overriding.

Pewarisan dan overriding, kelas anak bisa menggunakan method dari kelas induk, tapi tetap dapat mengubah perilaku method tersebut jika dibutuhkan.

Penggunaan ulang kode, dengan menggunakan pewarisan, kami menghindari pengulangan kode. Kode yang umum dan berlaku untuk semua kelas anak bisa diletakkan di kelas induk. Hasilnya, kami hanya menulis satu kali untuk hal-hal yang sama dan menyesuaikan yang berbeda di kelas anak

Algoritma yang kami gunakan sederhana, yaitu dengan membuat objek dari kelas induk dan kelas anak, lalu memanggil method yang ada sesuai input. Kode berjalan secara berurutan dan menampilkan output sesuai dengan method yang dijalankan.

b) Apakah dasar alasan pengambilan keputusan Anda untuk kasus ini?

Alasan mengapa kami menggunakan overriding di kelas anak adalah karena konsep polimorfisme dalam pemprograman berorientasi objek. Dengan overriding, kita bias mengubah method yang diwarisi dari kelas induk untuk menyesuaikannya dengan kebutuhan spesifik kelas

anak, seperti menampilkan pesan yang berbeda saat memanggil Organisasi() atau sukaBelajar().

Penggunaan pewarisan ini dipilih karena kita ingin menjaga kode tetap rapid an efisien. Kode yang bersifat umum, seperti method sukaBelajar, bisa ditaruh di kelas induk, sehingga tidak perlu diulang di setiap kelas anak. Dengan begitu, kami bisa focus menulis kode yang spesifik untuk kelas anak.

Refleksi

Selama minggu ini, kami mempelajari konsep pewarisan (Inheritance) dalam Java. Awalnya konsep ini terasa cukup sulit dan rumit karena melibatkan hubungan antara kelas induk dan kelas anak . Namun, setelah mendiskusikan dan mencoba beberapa contoh kode, kami mulai melihat bagaimana pewarisan bisa membuat kode lebih efisien dan tidak berulang. Kami menemukan bahwa atribut dan method dari kelas induk dapat digunakan kembali oleh kelas anak dan ini sangat membantu dalam membuat program yang lebih terstruktur. Salah satu hal baru yang kami pelajari adalah bagaimana method di kelas induk bisa diubah oleh kelas anak melalui proses yang disebut method overriding yang sebelumnya belum kami pahami sepenuhnya.

Singkatnya, kami merasa tugas ini membuat kami untuk memahami dasar pewarisan dan bagaimana cara mengimplementasikannya dalam kode bahasa Java.