LAPORAN RESMI
MODUL V
POLIMORFISME
PEMROGRAMAN BERBASIS OBJEK

NAMA	**:**	**ANISYAFAAH**
N.R.P	**:**	**220441100105**
DOSEN	**:**	**FIRMANSYAH ADIPUTRA, ST., M.Cs.**
ASISTEN	**:**	**KUKUH COKRO WIBOWO**
TGL PRAKTIKUM	**:**	**05 MEI 2023**

Disetujui : Mei 2023
Asisten

KUKUH COKRO WIBOWO
21.04.411.00102

LABORATORIUM BISNIS INTELIJEN SISTEM
PRODI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pesatnya kecanggihan teknologi di zaman sekarang, membuat teknologi menjadi salah satu hal yang sangat penting dalam kehidupan sehari-hari, terutama pemrograman komputer. Dalam membuat program komputer, seseorang membutuhkan proses pembelajaran yang cukup lama untuk menguasainya. Terlebih lagi, dalam dunia pemrograman, tentunya harus menggunakan bahasa pemrograman dan perlu menggunakan konsep penting di dalam programnya, seperti konsep polimorfisme.

Polimorfisme dalam OOP merupakan sebuah konsep OOP dimana class memiliki banyak bentuk method yang berbeda, meskipun namanya sama. Maksud dari bentuk adalah isinya yang berbeda, namun tipe data dan juga berbeda. Polimorfisme dapat diartikan sebagai teknik programming yang mengarahkan kita untuk memprogram secara general. Penggunaan method dengan nama yang sama dapat melalui atau menggunakan override dan overloading. Meskipun demikian, keduanya memiliki perbedaan. Berikut ini adalah contoh pembuatan program Java menggunakan konsep polimorfisme.

1.2 Tujuan

- Mahasiswa mampu memahami konsep Polimorfisme dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.

BAB II
DASAR TEORI

2.1 Pengertian Polimorfisme

Polimorfisme adalah suatu konsep dasar java. Polimorfisme berasal dari dua kata yaitu poli dan morphs, poli berarti banyak morphs berarti bentuk. Polimorfisme juga dapat diartikan sebagai metode dengan penamaan yang sama namun memiliki respon yang berbeda - beda. Implementasi dari polimorfisme sendiri adalah Override, Overloading, dan Polimorfisme Runtime.

a) Override :

Teknik ini telah disebutkan pada bagian sebelumnya teknik ini adalah teknik menempa atau menulis ulang method superclass pada subclass. Penggunaannya adalah dengan menulis ulang nama method pada bagian subclass.

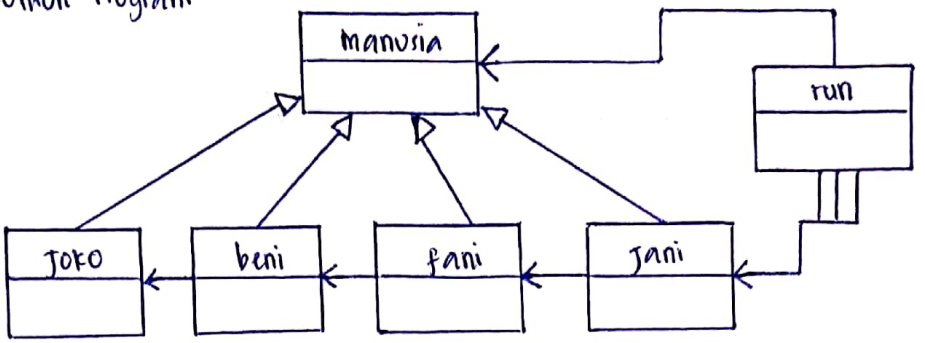
b) Overloading :

Overloading adalah kemampuan sebuah class memiliki beberapa method dengan nama yang sama, overloading bisa juga disebut Polimorfisme static. Teknik ini dapat digunakan apabila variablenya berbeda, teknik ini akan dibahas lebih lanjut nanti.

c) Polimorfisme Runtime :

Polimorfisme Runtime seperti namanya polimorfisme ini terjadi ketika program dijalankan, dalam implementasinya menggunakan metode inheritance dan override. Dalam membuat object biasa "manusia a = new manusia" manusia yang pertama berfungsi sebagai variable dan yang berikutnya adalah object, namun sekarang object dibuat dengan variable dan object yang berbeda "makhluk_hidup a = new manusia" makhluk hidup disini adalah superclass sedangkan manusia adalah subclassnya.

2.2 Contoh Program



- Class manusia sebagai superclass dari class joko, beni, fani, dan jani
- Run membuat object dari semua subclass manusia.

Class manusia :

```
public class manusia
{
   //membuat method
   public void profesi() {
      System.out.println("masukkan profesi");
   }
   //membuat method
   public void tugas() {
      System.out.println("masukan tugas");
   }
   //membuat method
   public void peralatan() {
      System.out.println("masukan peralatan yang dibutuhkan");
   }
}
```

Class ini berisikan 3 method yang nantinya akan diturunkan pada subclassnya.

Class joko, beni, fani, dan jani :

```
//inheritence
//membuat class yang terhubung dengan manusia
```

```
public class joko extends manusia
{
   //override
   public void profesi() {
     System.out.println("profesi : tentara");
   }
   //override
   public void tugas() {
     System.out.println("tugas : menjaga negara");
   }
   //override
   public void peralatan() {
     System.out.println("peralatan : senjata");
   }
}

//inheritence
//membuat class yang terhubung dengan manusia
public class beni extends manusia
{
   //override
   public void profesi() {
     System.out.println("profesi : petani");
   }
   public void tugas() {
     System.out.println("tugas : bercocok tanam");
   }
   //override
   public void peralatan() {
     System.out.println("peralatan : cangkul");
   }
}
```

```
//inheritence
//membuat class yang terhubung dengan manusia
public class fani extends manusia
{
  //overide
  public void profesi() {
    System.out.println("profesi : dokter");
  }
  //override
  public void tugas() {
    System.out.println("tugas : merawat pasien");
  }
  //override
  public void peralatan() {
    System.out.println("peralatan : stetoskop");
  }
}


//inheritence
//membuat class yang terhubung dengan manusia
public class jani extends manusia
{
  //overide
  public void profesi() {
    System.out.println("profesi : siswa");
  }
  //overide
  public void tugas() {
    System.out.println("tugas : belajar");
  }
  //overide
  public void peralatan() {
    System.out.println("peralatan : buku");
  }
```

Pada ketiga class di atas semuanya adalah subclass dari class manusia namun semuanya dilakukan proses inheritence dan disesuaikan dengan kebutuhannya masing-masing.

Class run :

```
public class run
{
    public static void running() {
        // Mendeklarasikan array dengan tipe manusia
        // Dengan kata lain sebagai array yang memegang object dari
        // tipe orang
        manusia [] manusia = new manusia [4] ;

        // Perhatikan, kita bisa menyimpan subclass apa saja dari
        // tipe manusia di dalam array manusia.
        manusia[0] = new joko ();
        manusia[1] = new beni ();
        manusia[2] = new jani ();
        manusia[3] = new jani ();

        // Sekarang, kita melakukan perulangan
        // terhadap array tersebut, dan memanggil method dari
        // class manusia maka tiap object akan melakukan hal yang
        // benar atau object akan menggunakan method yang ada
        // di classnya masing-masing!
    for (int i = 0; i < manusia.length ; i++) {
        manusia[i].prosesi ();
        manusia[i].tugas ();
        manusia[i].perilaku ();
        System.out.println ();
    }
}
```

Class ini digunakan untuk menjalankan program awalnya membuat array dengan jumlah subclass yang akan disampaikan. Selanjutnya,

pembuatan object pada array yang menggunakan metode Polimorfisme Runtime. Kemudian melakukan perulangan sebanyak jumlah array dari manusia, di dalam array itu kita panggil semua method yang ada pada subclass manusia.

Running Program :

profesi : tentara
tugas : menjaga negara
peralatan : senjata

profesi : petani
tugas : bercocok tanam
peralatan : cangkul

profesi : dokter
tugas : merawat pasien
peralatan : stetoskop

profesi : siswa
tugas : belajar
peralatan : buku

Dapat menampilkan semua anggota subclass dari class manusia dengan kaidah polimorfisme.

BAB III
TUGAS PENDAHULUAN

3.1 Soal

Jelaskan perbedaan dari ketiga implementasi polimorfisme (Override, Overloading, dan Polimorfisme Runtime)!

3.2 Jawab

- Override

  Override adalah teknik menempa method superclass pada subclass. Override memiliki nama yang sama dan parameter yang sama (antara subclass dan superclass) namun menjalankan perintah dan fungsi yang berbeda.

- Overloading

  Overloading adalah kemampuan sebuah class memiliki beberapa method dengan nama yang sama namun berbeda parameter (perintah yang dijalankan bisa jadi sama maupun tidak antara satu dengan yang lainnya)

- Polimorfisme Runtime

  Polimorfisme Runtime yaitu proses yang menyediakan panggilan ke metode yang diganti saat waktu proses. Prosesnya melibatkan penggunaan variabel referensi dari superclass untuk memanggil metode yang diganti. Prosesnya terjadi ketika program dijalankan dan menggunakan konsep inheritance dan override.

BAB IV

IMPLEMENTASI

4.1 Soal

Buatlah aplikasi penjumlahan, pengurangan, perkalian, dan pembagian menggunakan konsep polimorfisme dengan memasukkan parameter bilangan A = 10.5, bilangan B = 0.5, dan bilangan C = 1.25!

4.2 Jawaban

a) Hasil

PENJUMLAHAN

Bilangan A : 10.5
Bilangan B : 0.5
Bilangan C : 1.25
Hasil A + B + C : 12.25

PENGURANGAN

Bilangan A : 10.5
Bilangan B : 0.5
Bilangan C : 1.25
Hasil A - B - C : 8.75

PERKALIAN

Bilangan A : 10.5
Bilangan B : 0.5
Bilangan C : 1.25
Hasil A * B * C : 6.5625

```
*PEMBAGIAN
-------------
Bilangan A  : 10.5
Bilangan B  : 0.5
Bilangan C  : 1.25
Hasil A/B/C : 16.8
BUILD SUCCESSFUL (total time : 0 seconds)
```

b) Source Code

1) OperasiBilangan

```
package Projects;
public class OperasiBilangan {
    protected double a, b, c, d;
    protected void set_A (double a) {
      this.a = a;
    }
    protected void set_B (double b) {
      this.b = b;
    }
    protected void set_C (double c) {
      this.c = c;
    }
    protected void set_D (double d) {
    }
    protected double get_A() {
      return this.a;
    }
    protected double get_B() {
      return this.b;
    }
    protected double get_C() {
      return this.c;
    }
```

```
    protected double get_D() {
      return this.d;
    }
    protected void tampil() {
    }
}
```

2) OperasiPenjumlahan

```
package Project5;
public class OperasiPenjumlahan extends OperasiBilangan {
    @Override
    protected void set_D() {
      this.d = a + b + c;
    }
    @Override
    protected void tampil() {
      System.out.println("PENJUMLAHAN");
      System.out.println("--------------------");
      System.out.println("Bilangan A    : "+this.get_A());
      System.out.println("Bilangan B    : "+this.get_B());
      System.out.println("Bilangan C    : "+this.get_C());
      System.out.println("Hasil A+B+C : "+this.get_D());
    }
}
```

3) OperasiPengurangan

```
package Project5;
public class OperasiPengurangan extends OperasiBilangan {
    @Override
    protected void set_D() {
      this.d = A - b - c;
    }
```

```
    @Override
    protected void tampil() {
      System.out.println("PENGURANGAN");
      System.out.println("-----------------");
      System.out.println("Bilangan A   : " + this.get_A());
      System.out.println("Bilangan B   : " + this.get_B());
      System.out.println("Bilangan C   : " + this.get_C());
      System.out.println("Hasil A-B-C : " + this.get_D());
    }
}
```

4) OperasiPerkalian

```
package Projects;
public class OperasiPerkalian extends OperasiBilangan {
    @Override
    protected void set_D() {
      this.d = a*b*c;
    }
    @Override
    protected void tampil() {
      System.out.println("PERKALIAN");
      System.out.println("-----------------");
      System.out.println("Bilangan A   : " + this.get_A());
      System.out.println("Bilangan B   : " + this.get_B());
      System.out.println("Bilangan C   : " + this.get_C());
      System.out.println("Hasil A*B*C : " + this.get_D());
    }
}
```

5) OperasiPembagian

```
package Projects;
public class OperasiPembagian extends OperasiBilangan {
    @Override
```

```
    protected void set_D() {
      this.d = A / b / c;
    }
    @Override
    protected void tampil() {
      System.out.println("PEMBAGIAN")
      System.out.println("----------");
      System.out.println("Bilangan A : " + this.get_A());
      System.out.println("Bilangan B : " + this.get_B());
      System.out.println("Bilangan C : " + this.get_C());
      System.out.println("Hasil A/B/C : " + this.get_D());
    }
}
```

c) OperasiBilanganCetak

```
package Projects;
public final class OperasiBilanganCetak {
  private static void Cetak(OperasiBilangan[] OB, double a,
  double b, double c) {
  OB[0] = new OperasiPenjumlahan();
  OB[1] = new OperasiPengurangan();
  OB[2] = new OperasiPerkalian();
  OB[3] = new OperasiPembagian();

  for (int i = 0; i < OB.length; i++) {
     OB[i].set_A(a);
     OB[i].set_B(b);
     OB[i].set_C(c);
     OB[i].set_D();
     OB[i].tampil();
  }
}
```

```
    public static void main (String[] args) {
        OperasiBilangan[] A = new OperasiBilangan[4];
        Cetak ( A, 10.5, 0.5, 1.25);
    }
}
```

c) Penjelasan

Pada pembuatan program aplikasi ini, terdapat enam file berbeda yang disesuaikan dengan fungsinya masing-masing. Program ini menggunakan konsep polimorfisme dengan banyak bantuan penulisan set, get, operator aritmatika, dan penggunaan override. Operator yang digunakan tentunya disesuaikan dengan nama class atau nama filenya.

BAB V
PENUTUP

5.1 Analisa

Dari hasil praktikum, praktikan menganalisa bahwa dalam membuat sebuah aplikasi penjumlahan, pengurangan, perkalian, dan pembagian menggunakan konsep polimorfisme dengan parameter maka diperlukan beberapa class yang didalamnya memiliki masing-masing fungsi dan peran. Parameter menggunakan tipe data double dan untuk memprosesnya menggunakan perintah set, get, penggunaan inheritance, dan penggunaan override. Program yang dibuat bersifat statis karena nilai parameternya sudah ditentukan.

5.2 Kesimpulan

1. Polimorfisme adalah metode dengan penamaan yang sama namun memiliki respon yang berbeda-beda.
2. Implementasi dari polimorfisme terdiri dari override, overloading, dan polimorfisme runtime.
3. Program yang telah diimplementasikan di atas menggunakan konsep polimorfisme, perintah set, get, perulangan, penggunaan inheritance dan override.