LAPORAN PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

MODUL 6
“RELASI ANTAR KELAS”

Disusun oleh

Bunga Laelatul Muna

NIM : 21102010

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2023

BAB 1 – TUJUAN PRAKTIKUM

Tujuan dari praktikum instalasi Java (IntelliJ IDEA) adalah untuk memberikan pengetahuan dan keterampilan kepada mahasiswa tentang Relasi Antar Kelas

- Mahasiswa dapat memahami tentang konsep utama pbo yaitu Relasi Antar Kelas
- Mahasiswa dapat mengerti berbagai jenis relasi kelas dalam java

BAB II- DASAR TEORI

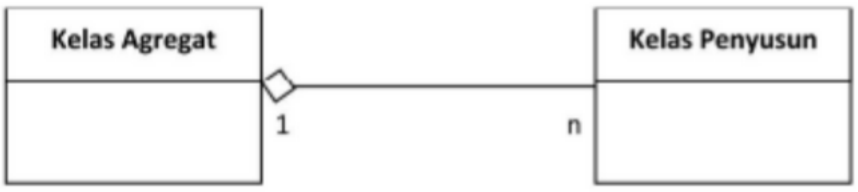
Pemrograman berorientasi objek (Object-Oriented-Programming) mengambil realita masalah dalam kehidupan sehari-hari. Dalam OOP, sering terjadi relasi antara satu objek dengan objek yang Iain.

Ada beberapa relasi mungkin terjadi antara suatu kelas dengan kelas yang Iain, yaitu:

a. Agregasi

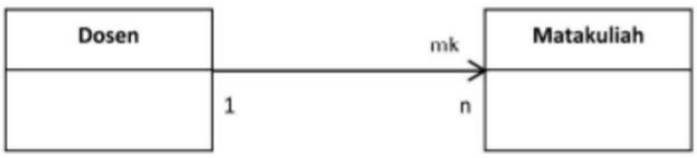
Agregasi adalah relasi antara dua buah objek dengan mengatakan bahwa satu objek memiliki atau mengandung atau berisi objek yang Iain. Misalnya, sebuah mobil memiliki mesin, roda, body; sebuah rumah memiliki dapur, kamar mandi, kamar makan. Apabila suatu objek tertentu tersusun atas objek-objek Iain, maka objek tersebut dikatakan sebagai objek agregat atau objek komposit (agregate, composite Object). Relasi ini sering disebut juga dengan relasi 'has-a' atau relasi 'whole-part'. Dalam diagram UML, relasi agregat ini digambarkan dengan simbol diamond. Simbol ini menunjukkan adanya inklusi struktural sebuah objek terhadap objek yang

Iain yang diwujudkan dengan cara membuat kelas agregat yang memiliki atribut yang bertipe kelas penyusun.



b. Asosiasi

Relasi asosiasi menyatakan suatu hubungan struktural antara beberapa objek yang menggambarkan objek dari suatu kelas dihubungkan dengan objek Iain. Relasi ini menunjukkan variabel dalam suatu kelas yang menyimpan rujukan bertipe kelas Iain. Diagram berikut menunjukkan relasi asosiasi antara kelas Dosen dan kelas Matakuliah, dengan arah panah asosiasi dari kelas Dosen menuju kelas Matakuliah yang menunjukkan bahwa Dosen menyimpan rujukan ke Matakuliah.



Nama yang berada di anak panah (mk) merupakan role name yang kelak akan menjadi atribut dari kelas Dosen.

Pada saat kita akan merancang relasi kelas, ada beberapa hal yang perlu diperhatikan, yaitu :

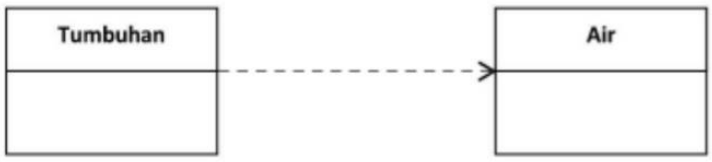
- Ada berapa objek dari masing-masing kelas yang terlibat dalam relasi.
- Apakah relasi tersebut bersifat wajib (mandatory) atau optional.

Dalam implementasi, asosiasi secara sintaks tidak memiliki perbedaan dengan implementasi agregasi, kecuali asosiasi bersifat dua arah.

c. Dependency

Relasi kebergantungan (dependency) menyatakan bahwa suatu kelas bergantung pada kelas yang lain, maka perubahan pada kelas yang menjadi ketergantungan dari kelas lain menyebabkan perubahan terhadap kelas yang tergantung tersebut. Misalnya, kelas tumbuhan membutuhkan kelas air, jika kelas air mengalami perubahan, maka menyebabkan perubahan pada kelas tumbuhan.

Relasi dependency dapat digambarkan dengan diagram UML sebagai berikut:



Perwujudan relasi ini dapat dilakukan dalam 3 bentuk:

1. Penggunaan kelas B sebagai parameter pada fungsi di kelas A.
2. Penggunaan kelas B sebagai nilai balikan pada fungsi di kelas A.
3. Penggunaan kelas B sebagai variabel lokal pada fungsi di kelas A.

Keterangan : Kelas A adalah kelas yang bergantung pada kelas B.

BAB III – GUIDED

Guided 1

- **Source Code (Manusia.Java)**

```
//Bunga Laelatul Muna
// 21102010
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Guided;
class Manusia {
    private  String nama;
    private int umur;
    private Manusia ibu;
    private Manusia anak;
    //Letak asosiasi, kelas anak jdi variabel
    public  Manusia(){}
    public  Manusia(String nm, int umr)
    {
        nama    = nm;
        umr     = umr;
        ibu     = new Manusia();
        anak    = new Manusia();
        ibu     = null;
        anak    = null;
    }
    public Manusia(String nm, int umr, Manusia ibu_angkat)
    {
        nama = nm;
        umur = umr;
        ibu  = new Manusia();
        anak = new Manusia();
        ibu  = ibu_angkat;
        ibu_angkat.anak = this;
    }
    //Relasi yang menunjukan asosiasi
    public void adopsi (Manusia anak_angkat)
    {
        anak = anak_angkat;
        anak_angkat.ibu = this;
    }
    public void cetak()
    {
        System.out.println("\n- Data Pribadi -");
        System.out.println("Nama : " + nama);
        System.out.println("Umur : " + umur);
        if (ibu!= null)
            System.out.println("Nama ibu : " + ibu.nama);
        else if (anak!=null)
            System.out.println("Nama anak : "+ anak.nama);
    }
}
```

- **Source Code (IbuAnak.java)**

```
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Guided;
public class IbuAnak {
    public static void main(String[] args) {
        Manusia ibu1 = new Manusia("Budi", 30);
```

```
        Manusia anak1 = new Manusia("Ani", 4);
        Manusia ibu2 = new Manusia("Diana", 40);
        Manusia anak2 = new Manusia("Andi", 5, ibu2);

        //Relasi Antara manusia dengan manusia
        System.out.println("==================\n");
        ibu1.cetak();
        anak1.cetak();
        System.out.println("====================\n");
        ibu1.adopsi(anak1);
        ibu1.cetak();
        anak1.cetak();
        System.out.println("======================\n");
        ibu2.cetak();
        anak2.cetak();

    }
}
```

- **SS Output**

```
====================


- Data Pribadi -
Nama : Budi
Umur : 0

- Data Pribadi -
Nama : Ani
Umur : 0
====================


- Data Pribadi -
Nama : Budi
Umur : 0
Nama anak : Ani
```

```
- Data Pribadi -
Nama : Budi
Umur : 0
Nama anak : Ani

- Data Pribadi -
Nama : Ani
Umur : 0
Nama ibu : Budi
======================


- Data Pribadi -
Nama : Diana
Umur : 0
Nama anak : Andi

- Data Pribadi -
Nama : Andi
Umur : 5
Nama ibu : Diana
```

- **Penjelasan**

  Kelas Manusia:

  - Kelas ini memiliki atribut nama dan umur yang mewakili nama dan umur seseorang.
  - Terdapat juga atribut ibu yang bertipe Manusia, yang digunakan untuk menunjukkan hubungan antara anak dengan ibunya.

- Terdapat konstruktor untuk membuat objek Manusia dengan parameter nama dan umur.
- Terdapat metode adopsi yang menerima objek Manusia sebagai parameter, yang digunakan untuk menetapkan hubungan ibu-anak antara dua objek Manusia.
- Terdapat metode cetak untuk mencetak informasi nama, umur, dan status hubungan ibu-anak dari objek Manusia.

Kelas IbuAnak:

- Kelas ini merupakan kelas utama yang memiliki metode main sebagai entry point program.
- Pada metode main, objek Manusia dibuat dengan nama dan umur yang diberikan.
- Beberapa objek Manusia kemudian diinisialisasi dan dihubungkan melalui hubungan ibu-anak menggunakan metode adopsi.
- Setelah itu, informasi dari objek Manusia dicetak menggunakan metode cetak

Guided 2

- **Source Code (Pegawai.Java)**

```
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Guided2;

class Pegawai {
    private String nama;
    private  String NIP;
    public Pegawai()
    {
        System.out.println("Konstruktor pegaia
dijalankan...");
    }
    public Pegawai (String NIP, String nama)
    {
        this.NIP = NIP;
        this.nama = nama;
        System.out.println("Konstruktor pegawai
dijalankan....");
    }
    public void tampilpeg(){
        System.out.println("NIP: " + NIP + " , Nama : " +
nama);
    }
}
```

- **Source Code (Perusahaan.java)**

```java
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Guided2;

class Perusahaan {
    private String namaper;
    private  Pegawai peg[]; //Agregasi antara pegawai dan
perusahaan
    private int counter;

    public Perusahaan (String namaper)
    {
        this.namaper = namaper;
        counter = 0;
        peg = new Pegawai[3];
        System.out.println("Konstruktor perusahaan
dijalankan");

    }
    //Relasi agresasi antara pegawai dan perusahaan
    public void insertPegawai (Pegawai p)
    {
        peg[counter] = p;
        counter++;
    } public void tampilPer()
    {
        System.out.println("Perusahaan " + namaper + "
memiliki pegawai : " ) ;
        for (int i=0; i<counter; i++)
        {
            peg[i].tampilpeg();
        }
    }

}
```

- **Relasi.Java**

```java
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Guided2;

public class Relasi {
    public static void main(String[] args) {
        Perusahaan Per = new Perusahaan("Nusantara Jaya");
        Pegawai Peg1, Peg2, Peg3;
        Peg1 = new Pegawai("P001", "Rudi");
        Peg2 = new Pegawai("P002", "Agus");
        Peg3 = new Pegawai("P003", "Andi");

        //Relasi antara class peg dan per

        Per.insertPegawai(Peg1);
        Per.insertPegawai(Peg2);
        Per.insertPegawai(Peg3);
        System.out.println();
        Per.tampilPer();
    }
```

- **SS Output**

```
Konstruktor perusahaan dijalankan
Konstruktor pegawai dijalankan....
Konstruktor pegawai dijalankan....
Konstruktor pegawai dijalankan....

Perusahaan Nusantara Jaya memiliki pegawai :
NIP: P001 , Nama : Rudi
NIP: P002 , Nama : Agus
NIP: P003 , Nama : Andi

Process finished with exit code 0
```

- **Penjelasan**
  - Kelas Perusahaan memiliki atribut namaPerusahaan dan daftar pegawai (ArrayList<Pegawai> pegawais). Terdapat juga beberapa metode, seperti insertPegawai(Pegawai pegawai) untuk menambahkan pegawai ke daftar pegawai perusahaan, dan tampilPer() untuk menampilkan informasi perusahaan beserta daftar pegawainya.
  - Kelas Pegawai memiliki atribut idPegawai dan namaPegawai. Terdapat juga konstruktor untuk menginisialisasi nilai atribut.
  - Dalam program ini, objek Perusahaan dengan nama "Nusantara Jaya" dibuat. Kemudian, tiga objek Pegawai (Peg1, Peg2, dan Peg3) juga dibuat dengan id dan nama masing-masing.
  - Kemudian, melalui pemanggilan metode insertPegawai(Pegawai pegawai), ketiga objek pegawai tersebut ditambahkan ke daftar pegawai di objek Perusahaan
  - Terakhir, dengan pemanggilan tampilPer(), program mencetak informasi perusahaan beserta daftar pegawainya.
  - Dengan menggunakan konsep agregasi, objek Perusahaan memiliki asosiasi dengan objek-objek Pegawai melalui daftar pegawainya. Hal ini memungkinkan perusahaan untuk mengelola dan memiliki koleksi pegawai yang terkait dengannya.

BAB IV – UNGUIDED

UnGuided 1

- **Source Code(Mahasiswa.Java)**

```java
//Bunga Laelatul Muna
//21102010
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Tugas;
class Mahasiswa {
    private String nim;
    private String nama;

    public Mahasiswa(String nim, String nama) {
        this.nim = nim;
        this.nama = nama;
    }

    public String getNim() {
        return nim;
    }

    public String getNama() {
        return nama;
    }
}
```


```java
//Bunga Laelatul Muna
//21102010
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Tugas;
class Mahasiswa {
    private String nim;
    private String nama;

    public Mahasiswa(String nim, String nama) {
        this.nim = nim;
        this.nama = nama;
    }

    public String getNim() {
        return nim;
    }

    public String getNama() {
        return nama;
    }
}
```


- **Source Code (Jurusan.java)**

```java
//Bunga Laelatul Muna
//21102010
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Tugas;
class Jurusan {
    private String nama;
    private Mahasiswa[] mahasiswas;

    public Jurusan(String nama, Mahasiswa[] mahasiswas) {
        this.nama = nama;
        this.mahasiswas = mahasiswas;
    }

    public String getNama() {
        return nama;
    }

    public Mahasiswa[] getMahasiswas() {
        return mahasiswas;
    }
}
```


```java
//Bunga Laelatul Muna
//21102010
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Tugas;
class Jurusan {
    private String nama;
    private Mahasiswa[] mahasiswas;

    public Jurusan(String nama, Mahasiswa[] mahasiswas) {
        this.nama = nama;
        this.mahasiswas = mahasiswas;
    }

    public String getNama() {
        return nama;
    }

    public Mahasiswa[] getMahasiswas() {
        return mahasiswas;
    }
}
```


- **Source Code (Main.Java)**

```java
//Bunga Laelatul Muna
//21102010
package com.Bunga_Laelatul_Muna.PBO.Pertemuan6.Tugas;
public class Main {
    public static void main(String[] args) {
        // Membuat objek mahasiswa
        Mahasiswa mhs1 = new Mahasiswa( nim: "21102010",  nama: "Bunga Laelatul Muna");
        Mahasiswa mhs2 = new Mahasiswa( nim: "21102011",  nama: "Madu Sahara");

        // Membuat objek jurusan dengan asosiasi ke mahasiswa
        Mahasiswa[] mahasiswas = {mhs1, mhs2};
        Jurusan jurusan = new Jurusan( nama: "Teknik Informatika", mahasiswas);

        // Menampilkan informasi mahasiswa dalam jurusan
        System.out.println("Jurusan: " + jurusan.getNama());
        System.out.println("Daftar Mahasiswa:");
        for (Mahasiswa mahasiswa : jurusan.getMahasiswas()) {
            System.out.println("NIM: " + mahasiswa.getNim() + ", Nama: " + mahasiswa.getNama());
        }
    }
}
```

- **SS Output**

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaag
Jurusan: Teknik Informatika
Daftar Mahasiswa:
NIM: 21102010, Nama: Bunga Laelatul Muna
NIM: 21102011, Nama: Madu Sahara
```

- **Penjelasan**

  Terdapat dua kelas utama yaitu Mahasiswa dan Jurusan. Kelas Mahasiswa merepresentasikan objek mahasiswa dengan atribut NIM dan nama. Kelas Jurusan merepresentasikan objek jurusan dengan atribut nama dan array mahasiswas yang merupakan kumpulan objek Mahasiswa dalam jurusan tersebut.

- Pada Main class, objek Mahasiswa dibuat dan kemudian dimasukkan ke dalam array. Array tersebut kemudian digunakan untuk membuat objek Jurusan yang memiliki asosiasi terhadap objek-objek Mahasiswa. Kemudian, program menampilkan informasi mahasiswa dalam jurusan dengan menggunakan metode getMahasiswas() untuk mengakses array mahasiswas dan mendapatkan informasi individu dari setiap objek Mahasiswa.

- Dengan demikian, program ini memanfaatkan konsep asosiasi/agregasi untuk menghubungkan objek-objek Mahasiswa dengan objek Jurusan.

A. REFERENSI

1. PetaniKode
2. Java Documentation
3. Modul 6