# AGREGASI DAN KOMPOSISI

## Tujuan

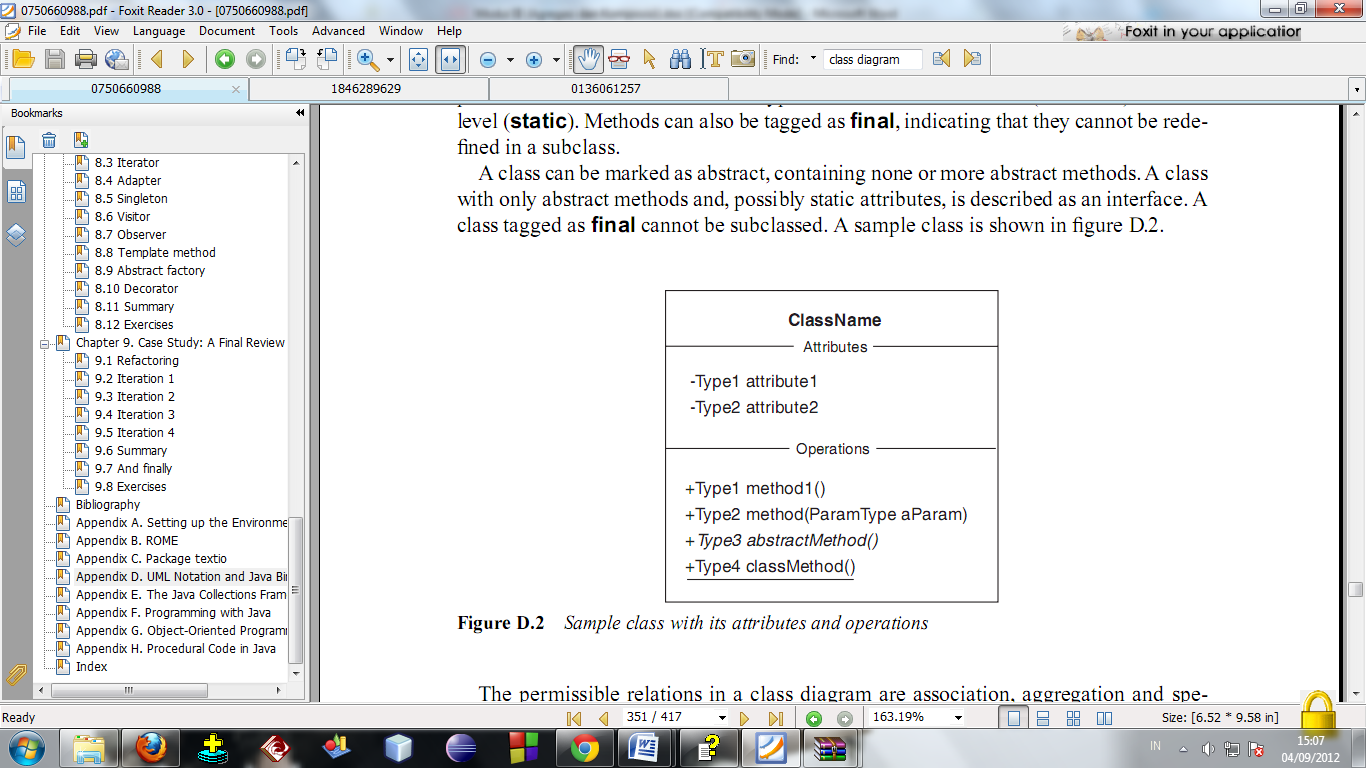
Setelah mengikuti praktikum ini mahasiswa diharapkan dapat :

1. **Mengerti konsep** dasar *object oriented programming*
2. Memahami **penggunaan kelas** dalam pemrograman java
3. Memahami **hubungan antar** kelas baik **asosiasi, agreasi** dan **komposisi**.

## Diagram Kelas

Diagram kelas merupakan sebuah **diagram** yang digunakan untuk **memodelkan** kelas-kelas yang digunakan di dalam **sistem** beserta **hubungan antar kelas** dalam sistem tersebut.

Beberapa **elemen penting** dalam diagram kelas adalah **kelas** dan **relasi antar kelas**. Kelas digambarkan dengan simbol kotak seperti gambar di bawah ini:



Gambar 3‑1 Diagram Kelas

**Baris pertama** dari simbol diagram kelas menandakan **nama** dari **kelas** yang bersangkutan. **Baris** di **bawahnya** menyatakan **atribut-atribut** dari kelas tersebut apa saja, dan **baris setelahnya** menyatakan ***method-method***yang terdapat pada **kelas** tersebut. Adapun simbol untuk ***access modifer*** adalah sebagai berikut:

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
| + | simbol access modifier ***public*** |
| - | simbol access modifier ***private*** |
| # | simbol access modifier ***protected*** (untuk simbol ini tidak terdapat pada gambar diatas) |

Sedangkan untuk menggambarkan **hubungan antar kelas** digunakan simbol **garis antara dua kelas** yang **berelasi**. Simbol garis tersebut antara lain:

- Class1 **berasosiasi** dengan Class2 , digambarkan sebagai berikut:



Gambar 3‑2 Hubungan Asosiasi

- Class2 merupakan **elemen part-of** dari Class1 (Class1 berelasi **agregasi** dengan Class2), digambarkan sebagai berikut:



Gambar 3‑3 Hubungan Agregasi

- Class1 dengan Class2 berelasi **komposisi**, digambarkan sebagai berikut:



Gambar 3‑4 Hubungan Komposisi

- Class1 merupakan **turunan** dari Class2, digambarkan sebagai berikut:



Gambar 3‑5 Hubungan Generalisasi

## Hubungan Antar Kelas

Dalam Obyek Oriented Programming, kelas-kelas yang terbentuk dapat memiliki hubungan satu dengan yang lainnya, sesuai dengan kondisi dari kelas-kelas yang bersangkutan. Ada beberapa jenis hubungan yang dapat terjadi antara kelas yang satu dengan kelas yang lainnya, antara lain:

1. Asosiasi
2. Agregasi
3. Komposisi
4. Generalisasi dan Spesialisasi (terkait dengan pewarisan)

### Asosiasi

Asosiasi merupakan **hubungan** antara **dua kelas tidak konseptual** **terkait** tetapi dalam konteks masalah perlu **memanfaatkan layanan** masing-masing . Contoh dari hubungan asosiasi ini adalah:



Gambar 3‑6 Contoh hubungan asosiasi

Pada diagram kelas di atas terlihat hubungan bahwa kelas dosen mengajar beberapa mahasiswa. Bentuk implementasi dari diagram kelas tersebut di Java adalah sebagai berikut:

|  |
| --- |
| //mahasiswa.java  public class mahasiswa {  private String nim;  private String nama;  public void setnama (String nama) {  this.nama = nama;  }  public void setnim (String nim) {  this.nim = nim;  }  public String getnim () {  return this.nim;  }  public String getnama () {  return this.nama;  }  }  //dosen.java  public class dosen {  private String Kddosen;  private String[] nimMHS;  private int JmlMahasiswa;  public void setKddosen (String Kddosen) {  this.Kddosen = Kddosen;  }  public void setNimMahasiswa (String nimMhs) {  if (JmlMahasiswa<5) {  nimMHS[JmlMahasiswa] = nimMhs;  JmlMahasiswa++;  }  }  public int getJmlMahasiswa () {  return this.JmlMahasiswa;  }  public String getKddosen () {  return this.Kddosen;  }  public String getmahasiswa (int i) {  return (nimMHS[i]);  }  } |

Pada **implementasi** terlihat bahwa **tidak ada relasi yang kuat** antara kelas dosen dan kelas mahasiswa, hanya ada **atribut** dari kelas **dosen** yang **serupa** dengan **atribut** dari kelas **mahasiswa** yang menandakan bahwa kedua kelas itu **berasosiasi**, yaitu atribut nimMahasiswa pada kelas dosen dan atribut nim pada kelas mahasiswa.

### Agregasi

Agregasi merupakan **hubungan** antara **dua kelas** yang **lebih ketat** dan terdapat hubungan **bagian** (part) dan **keseluruhan** (whole). Contoh hubungan agregasi ini adalah:



Gambar 3‑7 Contoh hubungan agregasi

Pada diagram kelas di atas, terlihat **hubungan** antara kelas **Jurusan** dengan kelas **Mahasiswa**. Kelas **mahasiswa** merupakan **bagian dari** kelas **jurusan**, tetapi **kelas jurusan** dan **kelas mahasiswa** **dapat** **diciptakan sendiri-sendiri**.

Implementasi dari diagram kelas tersebut dalam Java adalah sebagai berikut:

|  |
| --- |
| //mahasiswa.java  public class mahasiswa {  private String NIM, Nama;  public mahasiswa(String no, String nm) {  this.NIM = no;  this.Nama = nm;  }  public String GetNIM() {  return (NIM);  }  public String GetNama() {  return (Nama);  }  }  //jurusan.java  public class Jurusan {  private String KodeJurusan, NamaJurusan;  private Mahasiswa[] Daftar = new Mahasiswa[10];  public Jurusan(String kode, String nama) {  this.KodeJurusan = kode;  this.NamaJurusan = nama;  }  private static int JmlMhs = 0;  public void AddMahasiswa(Mahasiswa m) {  this.Daftar[JmlMhs] = m;  this.JmlMhs++;  }  public void DisplayMahasiswa() {  int i;  Sistem.out.println("Kode Jurusan : "+this.KodeJurusan);  Sistem.out.println("Nama Jurusan : "+this.NamaJurusan);  Sistem.out.println("Daftar Mahasiswa :");  for (i=0;i<JmlMhs;i++)  Sistem.out.println(Daftar[i].GetNIM()+" "+Daftar[i].GetNama());  }  } |

Pada implementasi terlihat bahwa kelas **jurusan memiliki atribut** yang memiliki **tipe kelas mahasiswa**, sehingga kelas **mahasiswa** merupakan **bagian** dari kelas **jurusan**.

### Komposisi

Komposisi merupakan **bentuk khusus** dari **agregasi** dimana terdapat hubungan **antar kelas** yang **lebih** **kuat** , karena yang menjadi **part (bagian)** **tidak** **akan ada tanpa** adanya kelas yang menjadi **whole**. Contoh hubungan komposisi adalah sebagai berikut:





🡪

Gambar 3‑8 Contoh hubungan komposisi

Dari diagram kelas di atas terlihat bahwa kelas **CPU, Monitor,** dan **Mouse** semuanya merupakan **bagian** dari kelas **Komputer** dan ketika kelas **Komputer** musnah maka kelas **CPU, Monitor, dan Mouse** akan ikut **musnah**.

Implementasi dari diagram kelas tersebut dalam Java adalah sebagai berikut:

Pada implementasi di atas, terlihat bahwa kelas **CPU, Monitor,** dan **Printer** merupakan **atribut** dari kelas **Komputer** dan baru **diciptakan** pada saat **instansiasi obyek** dari kelas **Komputer**.

|  |
| --- |
| //CPU.java  public class CPU {  private String Merk;  private int Kecepatan;  public CPU(String m, int k) {  this.Merk = m;  this.Kecepatan = k;  }  public void DisplaySpecCPU() {  Sistem.out.println(this.Merk + ", " + this.Kecepatan);  }  }  //Monitor.java  public class Monitor {  private String Merk;  public Monitor(String m) {  this.Merk = m;  }  public void DisplaySpecMonitor() {  Sistem.out.println(this.Merk);  }  }  //Printer.java  public class Mouse {  private String Merk, Type;  public Printer(String m, String t) {  this.Merk = m;  this.Type = t;  }  public void DisplaySpecMouse() {  Sistem.out.println(this.Merk + ", " + this.Type);  }  } |