Tugas Praktikum PBO Kelas D

**Enkapsulasi & Relasi Antar Objek**

Departemen Informatika membutuhkan sebuah sistem untuk melakukan pendataan kegiatan akademik. Sistem yang dibuat dapat mencatat mengenai course (kelas) apa saja yang tersedia pada Departemen Informatika, siapa saja lecture (dosen) yang ada pada departemen dan apa saja course yang ia ampu, serta siapa saja student (mahasiswa) yang ada pada departemen dan apa saja course yang ia ambil. Buatlah sebuah management system tersebut dengan ketentuan sebagai berikut:

1. Kelas
   1. Student
      1. Properties
         1. name
         2. age
         3. address
         4. studentID
         5. coursesEnrolled (List of Course objects)
      2. Methods
         1. getDetails()
         2. enrollCourse(Course c)
         3. viewEnrolledCourses()
   2. Lecture
      1. Properties
         1. name
         2. age
         3. address
         4. employeeID
         5. coursesTaught (List of Course objects)
      2. Methods
         1. getDetails()
         2. teachCourse(Course c)
         3. viewTaughtCourses()
   3. Course
      1. Properties
         1. courseCode
         2. courseName
         3. lecture (Lecture object)
         4. studentsEnrolled (List of Student objects)
      2. Methods
         1. addStudent(Student s)
         2. removeStudent(Student s)
         3. viewEnrolledStudents()
2. Relasi
   1. Kelas Course memiliki banyak objek Student
   2. Kelas Student memiliki banyak objek Course
   3. Kelas Lecture memiliki banyak objek Course
3. Enkapsulasi

Berikan enkapsulasi untuk properties dari setiap kelas dengan membuatnya tidak bisa diakses di luar kelas tersebut dan berikan fungsi untuk mendapatkan nilai serta mengubah nilai dari properties tersebut.

Jalankan program tersebut pada file driver berupa Main.java

Objek yang dibuat cukup di-define langsung di Main.

Misal dibuat interaktif command line nilai ++++++++

Operasi pada Main dibebaskan selama minimal mencakup semua method yang terdapat pada kriteria di atas.

Feel free jika ingin membuat kreasi tambahan selama telah memenuhi semua kriteria di atas :D

Untuk memudahkan pengerjaan berikut kerangka code yang bisa digunakan (ga wajib, yang mau mau saja)

* Student class

|  |
| --- |
| import java.util.List;  import java.util.ArrayList;  public class Student {  /\* TODO: implementasi enkapsulasi dengan  berikan akses yang sesuai  \*/  String name;  int age;  String address;  int studentID;  List<Course> coursesEnrolled;  Student(String name, int age, String address, int studentID) {  // TODO: buatlah fungsi konstruktor  this.coursesEnrolled = new ArrayList<>();  }  void getDetails() {  /\* TODO: buat fungsi untuk print detail dari Student,  menampilkan nama, umur, alamat, dan ID  \*/  }  void enrollInCourse(Course course) {  // TODO: buatlah fungsi untuk menambah sebuah course  }  void viewEnrolledCourses() {  /\* TODO: buatlah fungsi untuk menampilkan seluruh course  yang diambil oleh mahasiswa.  Hint: gunakan loop dan method getDetails dari Course  \*/  }  // Other methods...  } |

* Lecture class

|  |
| --- |
| import java.util.List;  import java.util.ArrayList;  public class Lecture {  /\* TODO: implementasi enkapsulasi dengan  berikan akses yang sesuai  \*/  String name;  int age;  String address;  int employeeID;  List<Course> coursesTaught;  Lecture(String name, int age, String address, int employeeID) {  // TODO: buatlah fungsi konstruktor  this.coursesTaught = new ArrayList<>();  }  void getDetails() {  /\* TODO: buat fungsi untuk print detail dari Lecture,  menampilkan nama, umur, alamat, dan ID  \*/  }  void teachCourse(Course course) {  // TODO: buatlah fungsi untuk menambah sebuah course  }  void viewTaughtCourses() {  /\* TODO: buatlah fungsi untuk menampilkan seluruh course  yang diampu oleh dosen.  Hint: gunakan loop dan method getDetails dari Course  \*/  }  // Other methods...  } |

* Course class

|  |
| --- |
| import java.util.List;  import java.util.ArrayList;  public class Course {  /\* TODO: implementasi enkapsulasi dengan  berikan akses yang sesuai  \*/  String courseCode;  String courseName;  Lecture lecture;  List<Student> studentsEnrolled;  Course(String courseCode, String courseName, Lecture lecture) {  // TODO: buatlah fungsi konstruktor  this.studentsEnrolled = new ArrayList<>();  }  void addStudent(Student student) {  // TODO: buatlah fungsi untuk menambah seorang mahasiswa  }  void removeStudent(Student student) {  // TODO: buatlah fungsi untuk menghapus seorang mahasiswa  }  void viewEnrolledStudents() {  /\* TODO: buatlah fungsi untuk menampilkan seluruh  student yang mengambil course.  Hint: gunakan loop dan method getDetails dari Student  \*/  }  void getDetails() {  /\* TODO: buat fungsi untuk print detail dari Course,  menampilkan kode course, nama course, dan dosen pengampu  \*/  }  // Other methods...  } |

* Main class

|  |
| --- |
| public class Main {  public static void main(String[] args) {  Lecture agus = new Lecture("Agus", 35, "Tembalang", 177013);  Lecture bambang = new Lecture("Bambang", 35, "Banyumanik", 412873);    Student cahyo = new Student("Cahyo", 20, "456 Elm St", 723751);  Student dani = new Student("Dani", 20, "123 Oak St", 734621);  Student edi = new Student("Edi", 20, "789 Pine St", 733451);    Course cs101 = new Course("CS101", "Intro to Computer Science");  Course cs102 = new Course("CS102", "Data Structures");  Course cs103 = new Course("CS103", "Algorithms");    agus.teachCourse(cs101);  agus.teachCourse(cs102);  bambang.teachCourse(cs103);  cs101.addStudent(cahyo);  cs101.addStudent(dani);  cs101.addStudent(edi);  cs102.addStudent(cahyo);  cs102.addStudent(dani);  cs103.addStudent(cahyo);  cahyo.viewEnrolledCourses();  agus.viewTaughtCourses();  cs101.viewEnrolledStudents();  }  } |