Manager

public void addStudent(String name, String email, int ID)
public int getStudent(String name);
public void addInstructor(String name, String email, String
course)
public int getInstructor(String name);
public void addCourse(String name, String instructor, String

roomNumber)
public int getCourse(String name);

Course

private String courseName private String instructor private String roomNumber private String schedule private ArrayList<Student> roster

Grade (Static?)

public String getLetterGrade(int numGrade) public void getClassGrades() public void getStudentGrades()

Assignment

private String name
private String course
private int maxScore
private ArrayList<Integer> studentIDs
private ArrayList<Integer> studentGrades

public Assignment(String name, String course, int maxScore)
public void printDetails()
// setters & getters for maxScore, name, course public void setStudent(int ID, int grade)
public int getStudentGrade(int ID)

Abstract Person

private String name private String email

public Person(String name, String email)
 public void print(Details()
 public void setName(String name)
 public String getName()
 public void setEmail(String email)
 public String getEmail()

-Student & Instructor Inherit Person-

Student

public int studentID

public Student(String name, String email, int ID)
public void printDetails()
public void setID(int ID)
public int getID()
public void exportStudent()

Instructor

private String courseName

public Instructor(String name, String, email, String course) public void printDetails()

Explanation **Polymorphism Encapsulation** Inheritance **Abstraction Description & Justification** Student and Instructor both inherit Abstract class Person inherited by Student and Instructor both have Frequent use of setters and getters Using a manager class with to handle all objects, as well as dynamic ArrayLists to contain them. methods with the same names (EX rather than directly manipulating Person Student and Instructor This makes it simple to interact with all of the classes. setName() or printDetails()) but they variables will be executed differently