CS101 Quiz 4

Name:

You are given a class named **Student**, for which properties (private) and methods are specified on the right hand side.

```
public class Student implements Comparable{
   private String name;
   private int ID;
   // Constructor
   public Student (String name, int ID){
    …..
   }
   // All methods including Getters,
   // toString and any other required methods
    ……
}
// This class is given to you!
// Do NOT implement this class.
```

Your assignment is to write a class named **Course**, for which properties and methods are specified as follows:

Properties (5 points):

courseName : A string variable, containing the course name

quota: An integer denoting the maximum number of students which may enroll to the course

noOfStudentsEnrolled : An integer variable, denoting the numberof students enrolled to the course

students : An **ArrayList** of type **Student**, when a student is enrolled to the course it is added to this list

Note: All of the properties should be *private*

Constructor (10 points): Constructor of this class takes two parameters, one for course name and one for the quota, rest of the properties are initialized based on these two parameters.

Methods:

- **isFull (10 points):** A method that checks whether the course is full or not
- **isEnrolled (15 points):** A method that checks if a student is enrolled to the course
- **enrollStudent (15 points):** A method that enrolls a student to the course if the student is not already enrolled. Note that, you need to compare the two students to decide for this. But, you do not know the details of the comparison rule due to **encapsulation**.
- **findStudent (15 points):** This method takes student id as its parameter and searches the *students* for a student having the given id. If a student with the given id is present then it returns that student, otherwise it prints out an error message.
- **findMinStudent (15 points):** This method returns the student with minimum ID among all the *students.*
- **toString (15 points):** A method that returns the course information, first it should have the course name, course quota and the number of students enrolled then it should have the ids' of the students enrolled to the course.

Note: Code reuse is important!

```
public class CourseRunner{
  public static void main(String[] args){
    Course c1 = new Course("CS224",4);
    c1.enrollStudent(new Student("Ali",12));
    c1.enrollStudent(new Student("Zerrin",7));
    c1.enrollStudent(new Student("Ahmet",7));
    c1.enrollStudent(new Student("Ayse",8));
    c1.enrollStudent(new Student("Veli",16));
    c1.enrollStudent(new Student("Ahmet",5));
    c1.enrollStudent(new Student("Asya",12));
    System.out.println(c1);
    System.out.println("The minimum student is: " + c1.findMinStudent());
  }
}
```

Sample Run

```
This Course entitled CS224 with a quota of 4 contains 4 students.
The students are as follows:
Name: Ali, ID: 12
Name: Zerrin, ID: 7
Name: Ayse, ID: 8
Name: Veli, ID: 16

The minimum student is: Name: Zerrin, ID: 7
```

```
import java.util.ArrayList;
public class Course {
	private String courseName;// A string variable, containing the course name
	private int quota;// An integer denoting the maximum number of students which may enroll to the course
	private int noOfStudentsEnrolled;// An integer variable, denoting the numberof students enrolled to the course
	private ArrayList<Student> students;// An ArrayList of type Student, when a student is enrolled added
	public Course (String courseName, int quota){ //Constructor
		this.courseName = courseName;
		this.quota = quota;
		noOfStudentsEnrolled = 0;
		students = new ArrayList<Student>();
	}
	public boolean isFull(){	//A method that checks whether the course is full or not
		return (noOfStudentsEnrolled==quota);
	}
	public boolean isEnrolled(Student s){		//A method that checks if a student enrolled to the course
		for(Student temp:students){
			if(s.compareTo(temp)==0)
				return true;
		}
		return false;
	}
	public void enrollStudent(Student s){		//A method that enrolls a student to the course
		if(!isFull() && !isEnrolled(s)){
			students.add(s);
			noOfStudentsEnrolled++;
		}
	}
	//This method takes student id as its parameter and searches the students for a student having the given id.
	public Student findStudent(int ID){
		for(Student s:students){
			if(s.getID()==ID)
				return s;
		}
		System.out.println("The student with ID "+ID+ " does not exist in this course.");
		return null;
	}
	public Student findMinStudent(){	// This method returns the student with minimum
		int min = Integer.MAX_VALUE;
		Student minStudent = null;
		for(Student s:students){
			if(s.getID()<min){
				min=s.getID();
				minStudent=s;
			}
		}
		return minStudent;
	}
	public String toString(){	//A method that returns the course information
		String output = "This Course entitled " + courseName + " with a quota of " + quota;
		output +=" contains " + noOfStudentsEnrolled + " students.\n";
		if(noOfStudentsEnrolled>0){
			output += "The students are as follows:\n";
			for(Student s:students)
				output += s + "\n";
		}
	return output;
  }
}
```