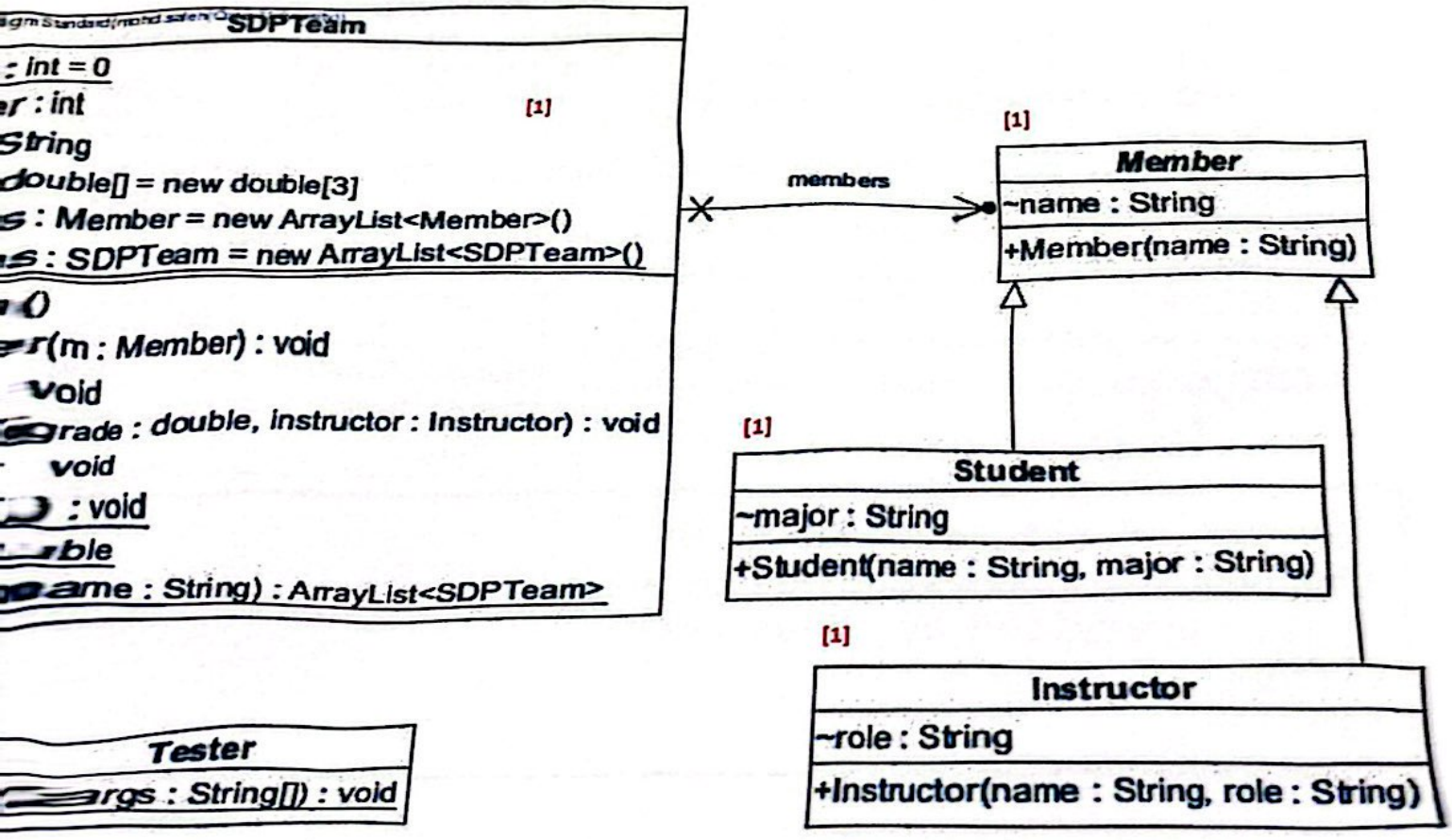
me:	QUID:

POINTS] Implement the following class diagram.

- The members of an **SDPTeam** consists of students and two instructors: one with a **role** of a supervisor, and the other with the **role** of examiner. The grade is an array of three integers: the second integer of it is the grade given by the supervisor, the third integer is the grade given by the examiner. The first integer of the grade array is the average of second and third integers of the grade array which is calculated by the method **calcGrade** whenever the method **gradeTeam** is called.
- A student **major** is either CS or CE. An **SDPTeam** has a **number** which is automatically generated starting with 1. Depending on the **major** of the students in the group being all CS, CE, or mixed of CE and CS, the **code** is the string CS, CE, or CECS followed by the value of **number** respectively, see **Tester** output. The method **setCode** sets the value of the **code** whenever the method **addMember** is called. The method **addMember** also adds a member to **members**.
- The class method **showHighest** finds the group **code** and **grade** of the group having the highest grade and display them on the screen, see **Tester** output.
- The class method **average** returns the average of all SDP teams grades.
- The class method **SupervisedBy** returns an ArrayList of all SDP teams supervised by the instructor with the given **name**.
- The **Tester** class and its output might give a better understanding of this diagram and expected functionality.



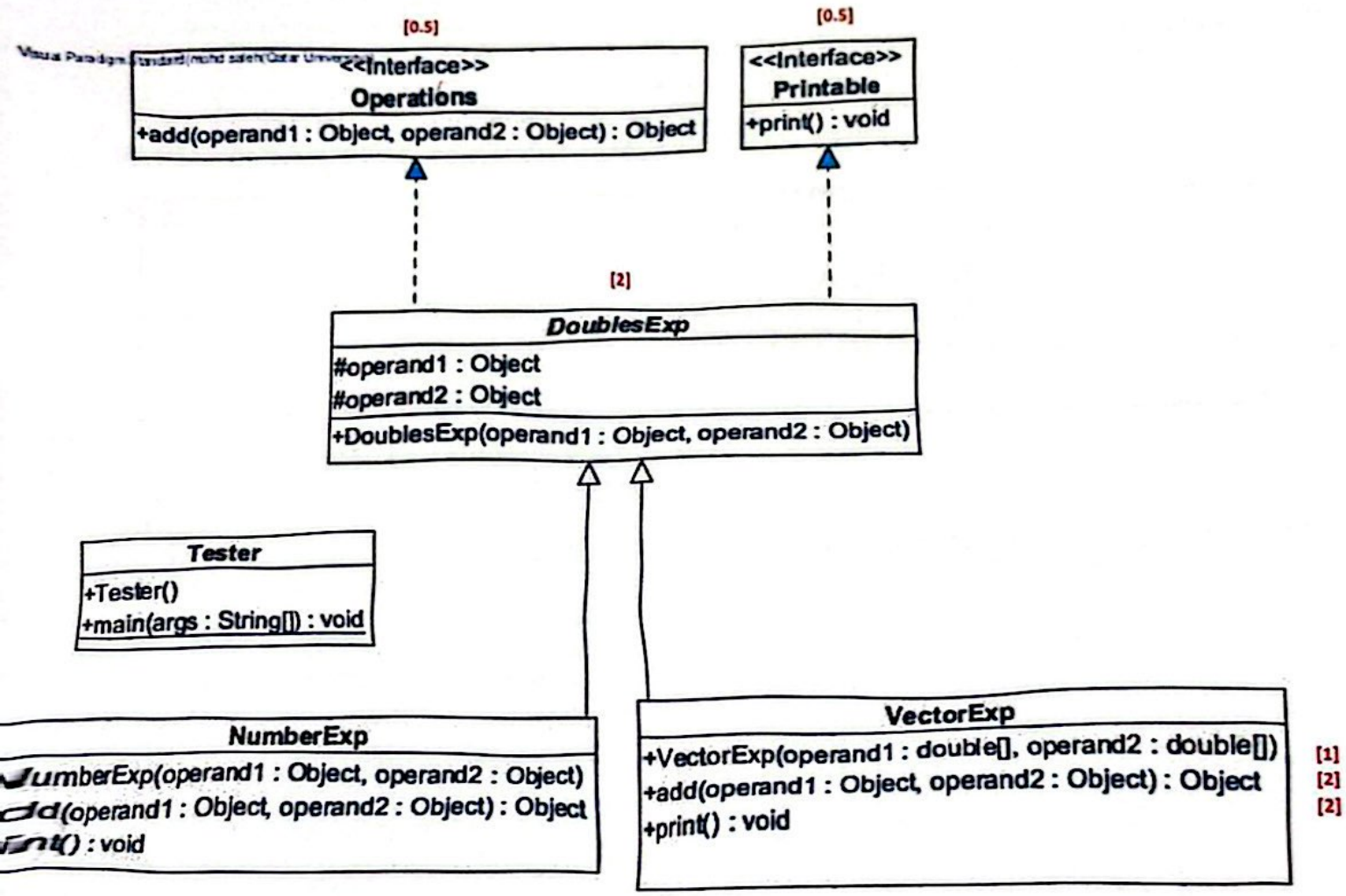
```
t java.util.ArrayList;
c class Tester {
  public static void main(String[] args) {
        Student s1 = new Student("Imran Hasan","CS");
        Student s2 = new Student("Ismael Qasim","CE");
        Student s3 = new Student("Ali Abdullah","CS");
        Student s4 = new Student("Husam Ahmed","CS");
        Student s5 = new Student("Zaid Awad","CE");
        Student s6 = new Student("Firas Majdi","CE");
        Instructor i1 = new Instructor("Abbas Omer","Supervisor");
        Instructor i2 = new Instructor("Abbas Omer","Examiner");
        Instructor i3 = new Instructor("Jihad Tawfiq","Supervisor");
        Instructor i4 = new Instructor("Jihad Tawfiq","Examiner");

        SDPTeam team1 = new SDPTeam();
        team1.addMember(s1);team1.addMember(s2);
        team1.addMember(i1);team1.addMember(i4);

        SDPTeam team2 = new SDPTeam();
        team2.addMember(s3);team2.addMember(s4);
        team2.addMember(i1);team2.addMember(i4);

        SDPTeam team3 = new SDPTeam();
        team3.addMember(s5);team3.addMember(s6);
        team3.addMember(i2);team3.addMember(i3);

        team1.gradeTeam(90, i1);team1.gradeTeam(80, i4);
        team2.gradeTeam(70, i1);team2.gradeTeam(60, i4);
        team3.gradeTeam(96, i2);team3.gradeTeam(90, i3);

        System.out.println("Average of all teams is "+SDPTeam.average());
        SDPTeam.showHighest();
        ArrayList<SDPTeam> sdps = SDPTeam.supervisedBy("Abbas Omer");
        System.out.println("Teams supervised by Abbas Omer:");
        for(SDPTeam sdp : sdps) {
              System.out.println(sdp.code);
        }
  }
```

```
Average of all teams is 81.0
CE3 has the highest grade of 93.0
Teams supervised by Abbas Omer:
CECS1
CS2
```

2. [11 POINTS] Implement the following class diagram.

An add operation of a double expression adds two operands that could be both double numbers or both double vectors. A double vector is a one-dimensional array of double numbers. The print method prints the addition expression and its result in formats that depends on what being added numbers or vectors, see the **Tester** class and its output.

The **Tester** class and its output might give a better understanding of this diagram and expected functionality.



```
import java.util.ArrayList;
public class Tester {
	public Tester() {
		ArrayList<DoublesExp> exps = new ArrayList<DoublesExp>();
		exps.add(new NumberExp(3.0, 5.0));
		double[] a = { 7, 1, 0, 5 };
		double[] b = { 5, 6, 9, 3 };
		exps.add(new VectorExp(a, b));
		for (DoublesExp e : exps)
			e.print();
	}
	public static void main(String[] args) {
		new Tester();
	}

}
```

```
3.0 + 5.0 = 8.0
[7.0, 1.0, 0.0, 5.0] +
[5.0, 6.0, 9.0, 3.0] =
[12.0, 7.0, 9.0, 8.0]
```