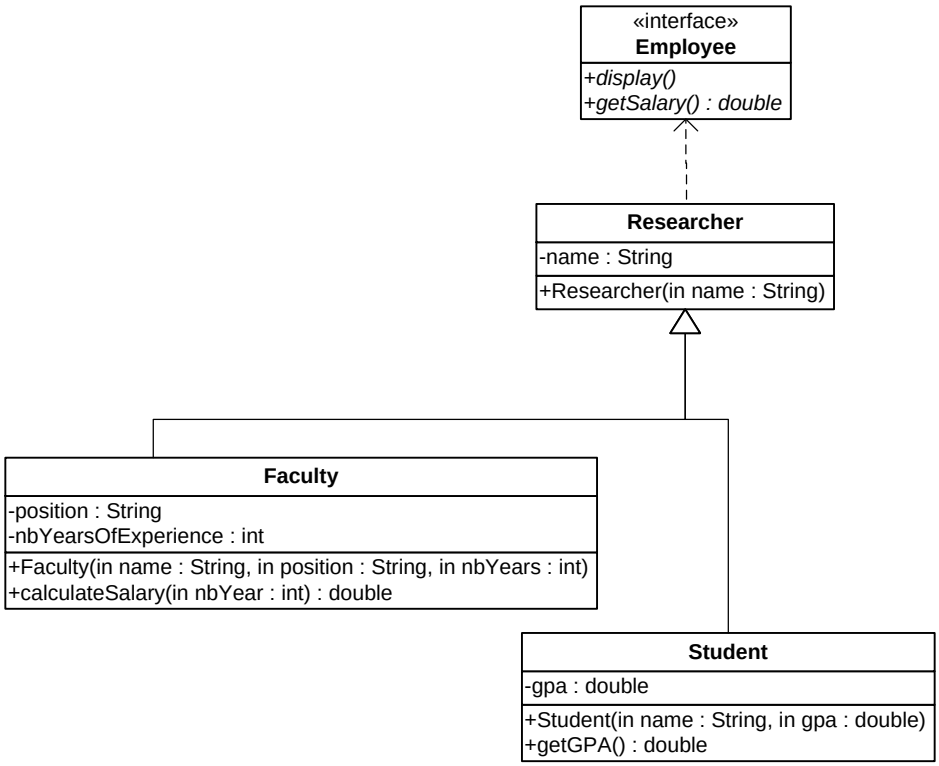
# King Saud University
# College of Computer and Information Sciences
# Department of Computer Science
# CSC113 – Computer Programming II – Midterm 2 Exam – Fall 2017

## Exercise1:




***Employee*** interface***:***

- Methods:
  - ***Display()***: displays all attributes of the researcher.
  - ***getSalary()***: this method does the following:

    - ***For Faculty***: it returns the he salary which is computed by the method ***calculateSalary*** using the number of years of experience.

    - ***For Student***: it calculates as returns the salary which is computed as follows: salary = 1000 * gpa.

***Researcher*** class***:***

- Attributes:
  - ***name***: the name of the researcher.
- Methods:
  - ***Researcher (name: String)***: constructor

***Faculty*** class

- Attributes:
  - ***position***: the position of the faculty.
  - ***nbYearsOfExperience:*** the number of years of experience of the faculty.
- Methods:
  - ***Faculty (name: String, position: String, nbYears: int)***: constructor.
  - ***calculateSalary (nbYears: int)***: it calculates and returns the salary of the faculty as follows:
    - ***For those without any experience ( nbYears is equal to 0), then the salary is 5600 SAR.***
    - ***For those who have 10 years of experience or less, then the salary of the current year = 1.05 * salary of the past year.***
    - ***For those who have more than 10 years of experience, then the salary of the current year = salary of the past year.***

***Student*** class:

- Attributes:
  - ***gpa***: the gpa of the student.
- Methods:
  - ***Student (name: String, gpa: double):*** constructor.
  - ***getGPA():*** returns the gpa of the student.

**QUESTION**: Translate into Java code the following interface and classes:

- ***Employee***
- ***Researcher.***
- ***Faculty.***

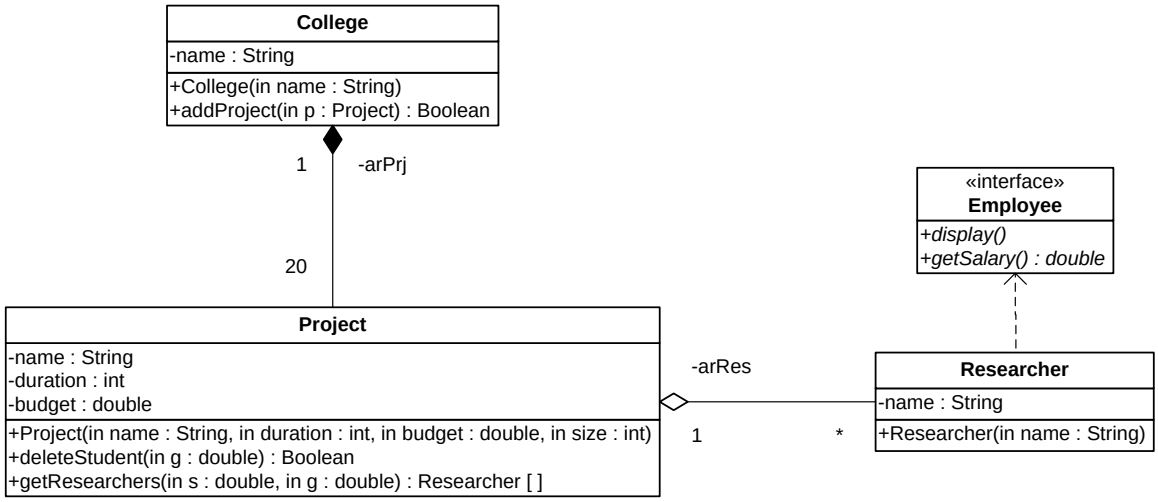
*Notice: The method **calculateSalary** should be implemented in a recursive way.*

## Answer of Exercise1: ……./ 16

```
public interface Employee {  ……./ 2
	public void	display();……./ 1
	public double	getSalary();……./ 1
}


public abstract class Researcher implements Employee { ……./ 1 + 1    ……./ 4

	private String name;

	public Researcher(String s) {  ……./ 1
		name = s;
	}

	public void display() {  ……./ 1
		System.out.println(name);
	}
}

public class Faculty extends Researcher { ……./ 1        ……./ 10
	private String	position;
	private int	nbYearsOfExperience;

	public Faculty(String name, String p, int nbE) {  ……./ 1
		super(name); ……./ 1
		position = p;
		nbYearsOfExperience = nbE;
	}

	public void display() {  ……./ 2
		super.display();……./ 1
		System.out.println(position + "---" + nbYearsOfExperience );…./ 1
	}

	public double calculateSalary(int n) {  ……./ 5

		if (n == 0) ……./ 1
			return 5600; ……./ 1
		else {
			if (n <= 10) ……./ 1
				return (1.05 *  calculateSalary(n - 1)); ……./ 1
			else
				return calculateSalary(n - 1); ……./ 1
		}
	}

	public double getSalary() {  ……./ 1
		return calculateSalary(nbYearsOfExperience); ……./ 1
	}
}
```

## Exercise 2:

Let's consider the same Researcher class and its subclasses described in exercise 1.




***Project*** class***:***

- Attributes:
  - ***name***: the name of the project.
  - ***duration***: the duration of the project.
  - ***Budget***: the budget of the project
- Methods:
  - ***Project(name: String, duration: int, budget: double, size: int)***: constructor
  - ***deleteStudent (g: double):*** this method deletes the first Student having a gpa less than ***g***.
  - ***getResearchers ():*** this method returns an array containing all Faculty members having a salary greater than ***s***, and all Students having a gpa less than ***g***.

**QUESTION**: Translate into Java code the class ***Project.***

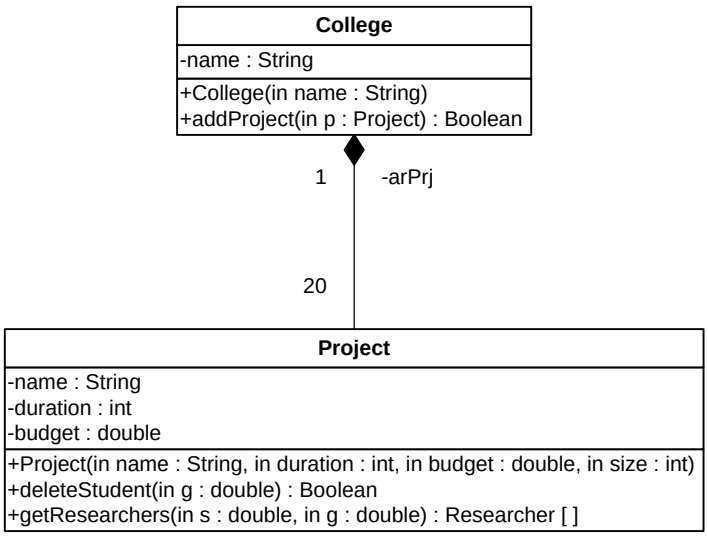
## Answer of Exercise2: ……./ 25

```
public class Project { ……./ 25
	private String name;
	private int duration;
	private double budget;

	private Researcher arRes[];……./ 1
	private int nbRes;	……./ 1

	public Project(String s, int d, double b,int size) { ……./ 2
		name = s;
		duration = d;
		budget = b;

		arRes = new Researcher[size]; ……./ 1
		nbRes = 0;  ……./ 1
	}

	public Project(Project p) { ……./ 4
		name = p.name;
		duration = p.duration;
		budget = p.budget;

		arRes = new Researcher[p.arRes.length]; ……./ 1
		for (int i = 0 ; i < p.nbRes; i ++) ……./ 1
			arRes[i] = p.arRes[i]; ……./ 1
		nbRes = p.nbRes; ……./ 1
	}

	public boolean deleteStudent(double g) { ……./ 6

		for (int i = 0; i < nbRes; i++) { ……./ 1
			if (  arRes[i] instanceof Student && ……./ 1
				((Student)arRes[i]).getGpa() < g ) { ……./ 1
				arRes[i] = arRes[nbRes -1]; ……./ 1
				nbRes --; ……./ 1
				return true; ……./ 0.5
			}
		}

		return false; ……./ 0.5

	}
```

```
	public Researcher[] getResearchers (double s, double g) {  ……./ 11

		Researcher [] res = new Researcher[nbRes]; ……./ 1
		int j = 0; ……./ 1

		for (int i = 0 ; i < nbRes; i++) { ……./ 1
			if (  (	arRes[i] instanceof Faculty && ……./ 1
				arRes[i].getSalary() > s ……./ 1 ) ||
				(	arRes[i] instanceof Student && ……./ 1
				((Student) arRes[i]).getGpa() < g …/ 1 + 1)
					) {
				res[j] = arRes[i]; ……./ 1
				j++; ……./ 1
			}
		}

		return res; ……./ 1
	}
}
```

## Exercise 3:

Let's consider the same Project class described in exercise 2.




***College*** class***:***

- Attributes:
  - ***name***: the name of the project.
- Methods:
  - ***College(name: String)***: constructor
  - ***addProject (p: Project*)*:*** this method adds the project ***p*** to the college. It returns true if the p is added correctly. It returns false otherwise.

**QUESTION**: Translate into Java code the class ***College.***

## Answer of Exercise3: ……./ 9

```
public class College { ……./ 9
	private String name;

	private Project arPhases[];……./ 1
	private int nbPrj; ……./ 1

	public College(String s) { ……./ 2
		name = s;

		arPhases = new Project[20]; ……./ 1
		nbPrj = 0; ……./ 1
	}

	public boolean addProject(Project p) { ……./ 5
		if (nbPrj < arPhases.length) {……./ 1
			arPhases[nbPrj] = new Project(p); ……./ 1 + 1
			nbPrj ++; ……./ 1
			return true; ……./ 0.5
		}
		return false; ……./ 0.5
	}
}
```