Class Course

```java
public class Course {
	private String name;
	private double grade;
	public Course(String name, double grade) {
		this.name = name;
		this.grade = grade;
	}
	public Course(Course c){
		this.name = c.name;
		this.grade = c.grade;
	}
	public String getName() { return name; }
	public double getGrade() { return grade; }
	public String toString(){ return name + " : " + grade; }
}
```

Class Student

```java
public abstract class Student {

	private int id;
	private String name;
	protected Course courses[];
	protected int nbCourses;
	public Student(int id, String name, int size) {
		this.id = id;
		this.name = name;
		courses = new Course[size];
		nbCourses = 0;
	}
	public Student(Student s){
		this.id = s.id;
		this.name = s.name;
		courses = new Course[s.courses.length];
		for(int i = 0; i < s.nbCourses; i++){
			this.courses[i] = s.courses[i];
		}
		this.nbCourses = s.nbCourses;
	}
```

```
	public boolean addCourse(Course c){
		if(nbCourses == courses.length)
			return false;
		courses[nbCourses++] = new Course(c);
		return true;
	}
	public int getId() {
		return id;
	}
	public String getName() {
		return name;
	}
	public abstract double calcGPA();

	public double getAverage(){
		if(nbCourses == 0) return -1;
		double sum = 0;
		for(int i = 0; i < nbCourses; i++)
			sum += courses[i].getGrade();
		return sum / nbCourses;
	}
	public String toString(){
		return "ID: " + id + ", Name: " + name;
	}

}
```

Class UnderGrad

```
public class UnderGrad extends Student{

	public UnderGrad(int id, String name, int size) {
		super(id, name, size);
	}
	public UnderGrad(UnderGrad ug){
		super(ug);
	}
	public double calcGPA(){
		double average = getAverage();
		if (average == -1) return -1;
		return average / 20;
	}
}
```

Class Graduate

```
public class Graduate extends Student{
	private int researchHours;

	public Graduate(int id, String name, int size, int researchHours) {
		super(id, name, size);
		this.researchHours = researchHours;
	}
	public Graduate(Graduate g){
		super(g);
		this.researchHours = g.researchHours;
	}
	public int getResearchHours() {
		return researchHours;
	}
	public String toString(){
		return super.toString() + ", Research hours completed: "
							+ researchHours;
	}
	public double calcGPA(){
		double average = getAverage();
		if (average == -1) return -1;
		return average / 25 + researchHours * 0.05;
	}
}
```

Class University

```
public class University {
	private String name;
	private Student [] students;
	private int nbStudents;

	public University(String name, int size) {
		this.name = name;
		students = new Student[size];
		nbStudents = 0;
	}
	public int search(Student s){
		for(int i = 0; i < nbStudents; i++)
			if(students[i].getId() == s.getId())
				return i;
		return -1;
	}
```

```
	public boolean addStudent(Student s){
		if(search(s) != -1 || nbStudents >= students.length)
			return false;
		if(s instanceof UnderGrad)
			students[nbStudents++] = new UnderGrad((UnderGrad) s);
		else
			students[nbStudents++] = new Graduate((Graduate) s);
		return true;
	}
	public boolean removeStudent(Student s){
		int index = search(s);
		if(index == -1) return false;
		students[index] = students[nbStudents-1];
		students[nbStudents-1] = null;
		nbStudents--;
		return true;
	}
	public Student getMaxGPA(){
		if(nbStudents == 0) return null;
		Student max = students[0];
		for(int i = 1; i < nbStudents; i++)
			if(students[i].calcGPA() > max.calcGPA())
				max = students[i];
		return max;
	}
	public int getNumberOfGrad(){
		int count = 0;
		for(int i = 0; i < nbStudents; i++)
			if(students[i] instanceof Graduate)
				count++;
		return count;
	}
	public void splitStudents(Graduate [] grad, UnderGrad [] underGrad){
		int countG = 0, countUG = 0;
		for(int i = 0; i < nbStudents; i++)
			if(students[i] instanceof Graduate)
				grad[countG++] = (Graduate) students[i];
			else
				underGrad[countUG] = (UnderGrad) students[i];
	}
	public Student[] getStudents(int hours){
		Student temp[] = new Student[getNumberOfGrad()];
		int count = 0;
		for(int i = 0; i < nbStudents; i++)
			if(students[i] instanceof Graduate
			&& ((Graduate)students[i]).getResearchHours() > hours)
				temp[count++] = students[i];
		return temp;
	}
}
```