Programming Assignment 3

Due Date: 11:59 PM 03/14/2020

How to submit your solutions:

- **You will create three .java files** for this assignment.
- Compress your project folder to a zip file and please upload it on Blackboard.

Note: Codes will be compared against each other using "document similarity" tools. Any evidence of code duplication will result in an 'F' in this assignment or an 'F' in this course.

You are required to implement three classes: `Student` and `StudentDatabase` to store the details of a set of students, and A `StudentDatabaseDemo` which will contain the main method for testing purposes.

1. The Student Class: This class should represent a student. The class's responsibilities are as follows:
   - To know the student's name, GPA, and ID number. Member variables: `String Name, double gpa, int idNumber.`
   - Required methods: one copy constructor, two constructors, accessors and mutators. The constructors are given below. The Default constructor is written for you. You need to complete the other two constructors.

```
public Student()
{
 Name = "";
 gpa = 0;
 idNumber = 0;
}
public Student(String newname, double newgpa, int newidNumber)
{

}

public Student(Student S)
{

}
```

2. The `StudentDatabase` Class: The object of this class should store a list of up to 10 objects of the `Student class`. This class should have private `Student[] Students` and private `int NumStudents` as member variables. The class should have only one constructor (given below)

```
public StudentDatabase(int num)
  {
        NumStudents = num;
        Students = new Student[numStudents];
        /*you need to update NumStudents
        when you add,remove Students.
        Also, you need to make sure NumStudents doesn't exceed 10.
        */

  }
```

This class should support the following methods (See below). You can add additional methods if you need to. Keep in mind; the maximum number of Students is 10. In the methods, you need to make sure that the database doesn't allow the user to add more than 10 Students. To make things easier let us assume no two Students have the same name. In any of the search methods, if the Student cannot be found, you should display that the Student wasn't found in the database.

- Add a new Student to the `StudentDatabase` through a series of inputs (from keyboard) from the user.

```
public void AddStudent()
{
 // not more than 10 Students in the database allowed
}
```

- Add a set of Students to the `StudentDatabase` through a file.

```
public void AddStudents(String StudentsFile)
{

 // Max of 10 Students in the database allowed
 // Hence, read up to max of 10 lines in the file.
}
```

  Assume that the File `StudentsFile` contains several lines, where each line is of the format `StudentName,gpa,idNumber`

  A Sample Students File is given below:

```
John,4.00,9002
Walter,3.40,9100
David,2.00,9010
Sam,3.60,9800
Little,2.56,9000
```

- THIS method should always display the Students' information in **descending order** of their gpa.

```
public void DisplayStudents()
{

}
```

- Allow the user to enter a Student's name, this method finds the Student with that specific name in the database and returns a copy of it. This method should return a null object if the Student was not found in the database.

```
public Student SearchStudentsByName(String StudentName)
{

}
```

- Allow the user to enter a Student's id number, this method finds the Student with that specific id number in the database and returns a copy of it. This method should return a null object if the Student was not found in the database.

```
public Student SearchStudentsByIdNumber(int StudentIdNumber)
{
}
```

- Allow the user to enter a Student's gpa `double StudentGpa`, this method finds ALL THE Students with gpa greater than or equal to the entered gpa in the database and then returns copies of all the matching Students. This method should return a null object if matching Student was not found in the database.

```
public Student[] SearchStudentsByGpa(double StudentGpa)
{
}
```

- Allows the user to enter a Student's name and remove the Student matching that name from the database. This method should give an error message if no Students were found.

```
public void RemoveStudentByName()
{

}
```

3. A `StudentDatabaseDemo` Class to test the above two classes. This class will only have the main method. In the main method you will test the two classes – `Student` and `StudentDatabase` A description of how the main method should be designed is given below.

```
public class StudentDatabaseDemo
{
      public static void main(String[] args)
         {
             /*
              Testing the copy constructor of Student
              Student s1 = new Student("Sam",3.40,1900);
              Student s2 = new Student(p1);
              //setName() is a mutator method
              s1.setName("John");
              //getName() is an accessor method
              System.out.println(s1.getName() == s2.getName()); //should be false

              */

              /*
              StudentDatabase Sim= new StudentDatabase();

            1. Use AddStudents(String StudentsFile) to load Students
            2. Display number of Students in the database
            3. You should display all the Students in the database- their names, gpa and
               id numbers (one Student per line) in descending order of their gpas.

            4. Here you need to create a menu system that allows the user to select
               which option or method to invoke from the Class StudentDatabase. Also,
               provide an option to allow the user to exit from the menu. Everytime, the
               user choose to add or remove students, the details of the students should
               be displayed.
```

```
            5. If the user choose to exit from the menu system, display number of
               remaining Students in the database (their names, gpa and id numbers in the
               database) in descending order of their gpas.

               */

        }
}
```