**CSC142, Computer Science II, Project 7 assignment**

Submit all required java files *Time, Records*, and *Calculator* that are explained in the below. Later submission is not accepted.

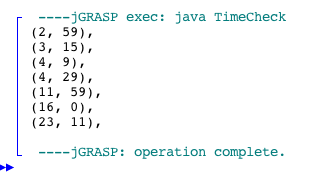
1. (10 points) Below is a class without using any extra import. It stores the hour and minute in private attributes. The constructor is also given here.

public class Time implements Comparable {  
private int hour =0;  
private int minute =0;  
public Time (int h, int m){  
 if (h<0||h>23) return;  
 if(m<0||m>59) return;  
 hour = h;  
 minute = m;  
}  
// additional (instance) methods will go here.  
…

}

Please complete and submit the above **Time** class with methods compareTo and toString, in order to support the following application class and reach the desired result:

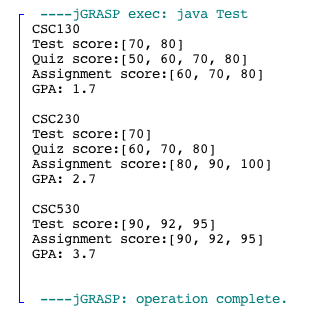
import java.util.Arrays;  
  
public class TimeCheck{  
public static void main(String args[]){  
 Time [] t = {new Time(3, 15), new Time(16, 0), new Time(4, 29),  
 new Time(11, 59), new Time(23, 11), new Time(2, 59), new Time (4, 9) };  
  
 Arrays.sort(t); // need for compareTo  
 for (Time f: t)  
 System.out.println(f+",");   
}  
}



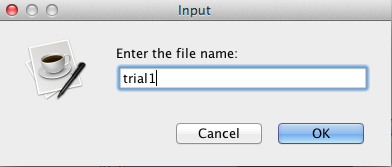
The desired class *Time* should **NOT** import anything and must use the above start part as we displayed here. Please do not change, otherwise you will lose all points of the entire program. Refer to Fruit.java for the use of Object type parameter when Comparable interface is implemented.

1. (10 points) Given the super class Student, please complete and submit the following class **Records**, in order to support the weighted score grading by the Test program and reach the desire result by using the input file “trial1”.

import javax.swing.JOptionPane;  
import java.io.\*;  
import java.util.\*;  
  
public class Test{  
public static void main(String [] args){  
Records [] t = {  
 new Records("CSC130", 2, 4, 3, 60, 10, 30),   
 // 2 tests, 4 quizzes, and 3 assignments  
 new Records("CSC230", 1, 3, 3, 25, 15, 60),   
 // 1 test, 3 quizzes, and 3 assignments  
 new Records("CSC530", 3, 0, 3, 60, 0, 40)};   
 // 3 tests, 0 quiz, and 3 assignments  
  
 String filename =   
 JOptionPane.showInputDialog("Enter the file name: ");  
 File inputFile = new File (filename);  
 try {   
 Scanner input = new Scanner(inputFile);  
 String val;  
 while(input.hasNext()){  
 val = input.nextLine();  
 String [] data = val.split(" ");  
   
 int index = Integer.parseInt(data[0]);  
 char opt = data[1].toLowerCase().charAt(0);  
 int [] s = new int[data.length-2];  
 for(int i = 0; i<s.length; i++)  
 s[i] = Integer.parseInt(data[i+2]);  
   
 switch(opt){  
 case 't':  
 t[index].set\_test(s);  
 break;  
 case 'q':  
 t[index].set\_quiz(s);  
 break;  
 case 'a':  
 t[index].set\_assignment(s);  
 break;  
 }  
 }  
 input.close();  
 }  
 catch (FileNotFoundException e)  
 {  
 System.out.println("file reading fails.");  
 }   
 for(int i = 0; i<t.length; i++){  
 Records tmp = t[i];  
 tmp.grading();  
 System.out.println(tmp);  
 }  
}  
}



“



In detail, the class *Records* (**NOT** using any extra import) should have the private attributes and the constructor as follows:

public class Records extends Student{  
private int [] test = null;  
private int [] quiz = null;  
private int [] assignment = null;  
  
public Records (String name, int tn, int qn, int an, int tw, int qw, int aw){  
 super(name, tw, qw, aw);  
 test = new int [tn];  
 quiz = new int [qn];  
 assignment = new int [an];  
}  
…

}

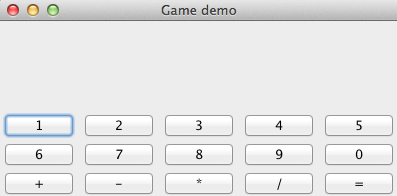
This class *Records* should also have the following methods:

* 1. set\_test: that will accept a group of test scores in an integer array. The values will copy to the private attribute array test.
  2. set\_quiz: that will accept a group of quiz scores in an integer array. The values will copy to the private attribute array quiz.
  3. set\_assignment: that will accept a group of assignment scores in an integer array. The values will copy to the private attribute array assignment.
  4. get\_test: will calculate and return the average of test scores in attribute array test.
  5. get\_quiz: will calculate and return the average of quiz scores in attribute array quiz.
  6. get\_assignment: will calculate and return the average of assignment scores in attribute array assignment.
  7. grading: will call the method “letter\_grading” of the super class to set the GPA score (i.e., attribute of super class).
  8. toString: will return the information in a String, which carries both the course name information and the scores of tests, quizzes, and assignments. GPA information is also needed.

1. (Bonus, 10 pts) Complete the given java class **Calculator** with **JavaFx** (**NOT** JFrame as you see in GUI\_Demo.java inside GUI zip file).

public class Calculator extends Application

# The program should create a GUI frame that contains 15 buttons and one label. See the feature as demonstrated here (note that the size or color does not matter).



Many calculators (e.g., Microsoft Calculator) follow the immediate execution mode. The

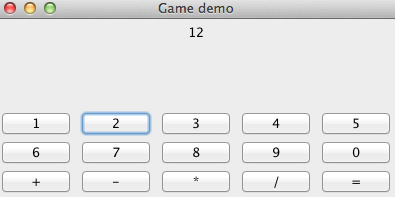
operation is executed as soon as the next operator is pressed. For instance,

12 + 23 \* 34 = (12 + 23) \* 34 = 35 \* 34

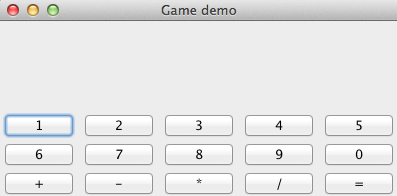
will yield 1190.

This program will implement a calculator in the immediate execution mode. For instance,

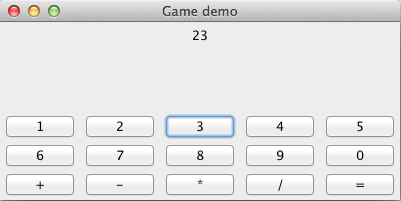
Step 1: click on button 1 and 2.



Step 2: click on button +.



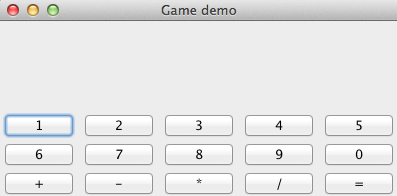
Step 3: click on buttons 2 and 3.



Step 4: click on button =. (Without using =, your process can be more complicated)



Step 5: click on button \*.



Step 6: click on buttons 3 and 4.



Final step: click on button =.



Note that if you use PC windows, the feature might have slight difference but the content and the functionality must fully be implemented. Due to the use of JavaFx, your program should prepare a lot of setOnAction, instead of using actionPerformed (that you see in Demo\_GUI.java).