# CSC 142

# Computer Science II

# Fall 2020

# Final Exam

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |
| --- | --- | --- | --- |
| Page | Java file to submit through D2L | Points | Score |
| 2 | Student | 20 |  |
| 4 | MyArray, Friends | 10+5 |  |
| Total |  | 35/30 |  |

1. Implement Student.java, which includes the following method development:
2. Implement an accessor **get\_average** that can return the average of the grades in the records.
3. Implement an accessor **get\_name** that can return the name of the calling student.
4. Implement the **toString** method that can return the letter grade “A”-“F”. The grading is based on the following Numeric-to-letter scale: [90 .. 100] A, [80 .. 90) B, [65 .. 80) C, [60 .. 65) D, and [0 .. 60) F, where the bracket [ ] indicates an interval including both end points and the parentheses ( ) indicates the interval not including any end point.
5. Implement the accessor **compareTo** that uses an Object type variable as the parameter, but can accept a Student object as an argument. It returns an integer (-1, 0, and 1) to indicate whether the average of this object is less, equal, or larger than that of the parameter. Refer to the attached Fruit.java for the Object type parameter.

Student class must have the following start part without any change, which includes the private attributes name and grade to store the records for a student. Please do not change, otherwise you will lose all points of the entire program.

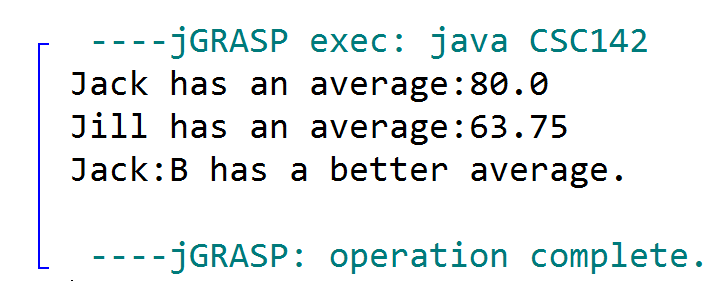
public class Student implements Comparable{  
private String name;  
private int grade[];  
// No more any other attribute.

public Student(String n, int[] g){  
name = n;  
grade = new int [g.length];  
for(int i = 0; i<g.length; i++)  
 grade[i] = g[i];  
}

// additional (instance) methods will go here.  
} // end of class Student.

The development should support the following class CSC142 and achieve the desired result, where main method constructs two students, “Jack” and “Jill”. Jack’s grades on three tests for the class are 70, 80, and 90. Jill’s grades are 40, 60, 65, and 90. After constructing the objects, the main method uses the above instance methods to determine who has a better average score and then print out this student’s name and letter grade.

public class CSC142 {  
public static void main(String [] args){  
int [] g1 = {70, 80, 90};  
int [] g2 = {40, 60, 65, 90};  
Student a = new Student ("Jack", g1);  
Student b = new Student ("Jill", g2);  
System.out.println(a.get\_name()+" has an average:" +a.get\_average());  
System.out.println(b.get\_name()+" has an average:" +b.get\_average());  
if(a.compareTo(b)<0) System.out.println(b.get\_name()+":"+b+" has a better average.");  
else System.out.println(a.get\_name()+":"+a+" has a better average.");   
}  
}



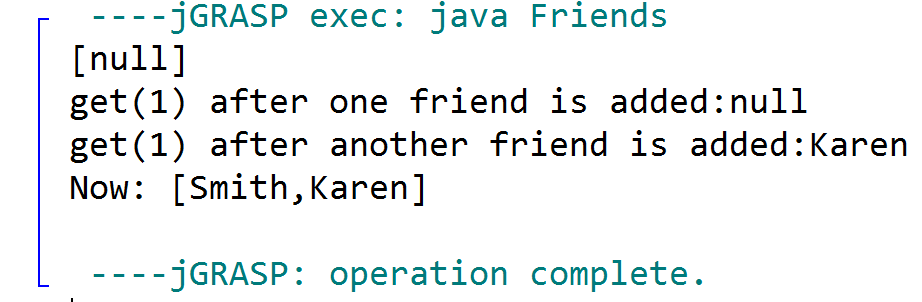
1. Given interface StringArrayInterface:

public interface StringArrayInterface{  
public void add(String item);  
// add a new record from the tail.  
public String toString();  
// toString  
public String get(int i);  
// return the value of record at position i  
}

Complete the class MyArray with a *private* array attribute “private String [] head” and support the following program for the desired result without throwing any exception:

public class Friends extends MyArray{  
 public Friends (){  
 super();  
 }  
   
public static void main(String [] args){  
 Friends r = new Friends();  
 System.out.println(r);  
 r.add("Smith");  
 System.out.println("get(1) after one friend is added:" + r.get(1));  
 r.add("Karen");  
 System.out.println("get(1) after another friend is added:" + r.get(1));  
 System.out.println("Now: "+r);  
}

…   
}



Your job has two parts:

1. Complete the abstract class MyArray without changing anything that has already been provided:

public abstract class MyArray implements StringArrayInterface {  
private String [] head;  
  
public MyArray(){ head = null;}

public MyArray(String [] str){  
 if (str==null) {  
 head = null; // or head = str;  
 return;  
 }  
 head = new String [str.length]; // deep copy is needed.  
 for(int i = 0; i<str.length; i++)  
 head[i] = str[i];  
}  
public int size(){  
 if(head==null||head.length==0)  
 return 0;  
 else  
 return head.length;  
}  
public abstract String toString();  
// Your development will go here.

} // End of class MyArray

1. Complete the class Friends which extends the (abstract) class MyArray:

public class Friends extends MyArray{  
 public Friends (){  
 super();  
 }   
 public static void main(String [] args){  
 Friends r = new Friends();  
 System.out.println(r);  
 r.add("Smith");  
 System.out.println("get(1) after one friend is added:" + r.get(1));  
 r.add("Karen");  
 System.out.println("get(1) after another friend is added:" + r.get(1));  
 System.out.println("Now: "+r);  
}

// Your development will go here. Hint—an abstract method needs implementation.  
  
}