CSE1325 OOP Final

August 8 or 12, 2014

Name: Key UTA ID: 1000

Instructions:

1. Read all of the instructions for each question and answer what is asked. Do not write down random stuff if you don’t know the answer.
2. The test is worth 100 points and there will be 10 points extra credit available.
3. If you have a question during the test, raise your hand and ask the proctor. You may or may not get an answer, but you won’t know unless you ask.
4. Check for bonus questions.

1. Assume there is a small GUI class to do the following: The class should create a panel that displays a question and two buttons, YES and NO. Based on the user's selected button, the panel should display a message for YES and a different message for NO. All items should be on the same panel. Use the drawings below as the three versions of the panel. The all yellow version A is what is produced when the program begins running. Version B, with the green band of text at the bottom, is produced if the user clicks the Yes button and version C, with the blue band of text at the bottom, is produced if the user clicks the No button. Using these three panels, answer the questions below.

Have you stopped waiting until the last minute to start your lab assignments?

Yes, I have

No, I have not

Testing QuesStop - YN

Well, it's good that you stopped!

Have you stopped waiting until the last minute to start your lab assignments?

Yes, I have

No, I have not

Testing QuesStop - YN

**You should think about stopping!**

Have you stopped waiting until the last minute to start your lab assignments?

Yes, I have

No, I have not

Testing QuesStop - YN

A.

B.

C.

1.a. What type of Swing component would you use to implement the buttons and why would you choose that component for this window? {4 points}

JRadioButton in a ButtonGroup

So that only one button can be selected at a time and each choice is discrete

1.b For the Swing components in question 1.a. what interface would be needed to add functionality to the buttons? Name the specific interface to use and the name of the method to implement. {4 points}

ActionListener or ItemListener (or more specific listener)

isSelected or ActionPerformed // ActionPerformed is very generic

1.c. Write only the Java statements that would be needed to create the buttons, place them on a specified panel labQues, and connect them to the interface discussed in question 1.b. Assume you already have a panel named labQues defined and that all needed imports have been done. {9 points}

JRadioButton yes = new JRadioButton("Yes");

JRadioButton no = new JRadioButton("No");

ButtonGroup waiting = new ButtonGroup();

waiting.add(yes);

waiting.add(no);

labQues.add(yes);

labQues.add(no);

yes.addActionListener(buttonHandler);

no.addActionListener(buttonHandler);

// assumes that buttonHandler is defined as a class to implement the

// ActionListener interface for yes and no

1.d. Connecting Swing components to their interface is called

 registering the action listener

 {4 points}

1.e. Given that the three windows shown are the only possible configurations for the GUI in question 1, answer the following True / False questions about the behavior of the GUI. {2 pts each; 8 total}

( Use T for True and F for False. If I cannot tell what the letter is, then you will get no points. )

A. False The Yes button sets the GUI to version A.

B. True The version C GUI changes the formatting of the background and/or answer text.

C. True The No button changes the GUI to version C.

D. False The version B GUI is the initial state of the GUI.

2. Define an "abstract class" and an "interface" in Java and explain the differences between the two concepts. Your definition and explanation should be written in words. Also give code examples of both. {12 points}

An abstract class is a class that has methods that are not concretely implemented. It can only be used as a superclass for defining other classes

 Example of abstract class

An interface is list of methods that must be defined by any class that implements the interface

<a skeleton that holds methods. The methods can be defined or undefined.>

 Example of interface

3.a. Name and describe two actions that are part of the wrapper class *Double* that are not part of the primitive type *double*. {6 points}

Many answers

3.b.Give two examples where using *Double* class would allow you to manipulate your data usefully that could not be done with the *double* class. Write in sentences, not in Java, and make sure your examples are different from the actions you described in question 3.a. {6 points}

The bubble sort is an O(n2) algorithm. The algorithm pseudocode to bubble sort from “smallest” element to “largest” element is stated in the four lines 1. – 4. below.

Given an unsorted array A of *n* elements indexed from [0] to [n-1]:

1. For the number of elements in the array, starting at j = 0, and incrementing until j = n-1,

2. For the first element A[0] to the last element A[n-1] where the current element is A[k],

3. Compare to see if element A[k] is larger than A[k+1]. If so,

4. Swap the values of A[k] and A[k+1]

 // k = n at this point

// j = n at this point

4.a. How many times must line 1. be executed to complete this algorithm? How did you determine this number? {5 points}

Line 1 would be executed n + 1 times. On the first execution j = 0. On the last successful execution j = n – 1. When j = n, the line would be executed but would fail. Therefore it would execute 1 time more than the n times needed for the successful executions.

4.b. Which line or lines are executed approximately n2 times? Is it possible to determine exactly how many times those lines are run? If so, calculate that number. If not, explain why not. {7 points}

Line 3 would be executed n2 times. This can be determined exactly.

Line 1 executes successfully n times (as described above)

 Each time Line 1 runs, Line 2 executes successfully n times (using the same logic as for Line 1)

 Each time Line 2 runs, Line 3 is run once.

Therefore Line 1 execution *n* \* Line 2 execution *n* \* Line 3 execution *1* = ***n \* n \* 1***  *= n2*

4.c. There are two modifications that are commonly made to improve the efficiency of bubble sort. Rewrite the pseudocode for bubble sort below, changing it so that the two efficiency modifications are included. (If you’ve forgotten your bubble sorting, one mod reduces the number of times that previously sorted elements are looked at and the second mod looks at the number of swaps done and uses that info.) {6 points}

Given an unsorted array A of *n* elements indexed from [0] to [n-1]:

Set swapCounter = 1

1. For the number of elements in the array, starting at j = 0, and

incrementing until j = n-1 or until swapCounter = 0;

Set swapCounter = 0

2. For the first element A[0] to the last element A[n-1-j] where the current element is A[k],

3. Compare to see if element A[k] is larger than A[k+1]. If so,

4. Swap the values of A[k] and A[k+1]

 Increment the swapCounter

 // k = n at this point

// j = n at this point

4.d. After making the changes in the previous question, does the overall efficiency of the algorithm change? For each line of your modified algorithm, indicate how many times that line would be executed in terms of *n* for an array of size *n*. {8 points}

No Worst Case Executions

Set swapCounter = 1 1

1. For the number of elements in the array, starting at j = 0, and n+1

incrementing until j = n-1 or until swapCounter = 0;

Set swapCounter = 0 n

1. For the first element A[0] to the last element A[n-1-j] where the current is A[k], n \* (n+1)

3. Compare to see if element A[k] is larger than A[k+1]. If so, n \* n

4. Swap the values of A[k] and A[k+1] <= n\*n

 Increment the swapCounter

 // k = n at this point

// j = n at this point

5.a. Write Java code to implement a class that represents the following information:

Create a class to hold all data for a student's advising record. This class may use other classes to hold data and you may define enums. Make sure to define any unique classes you use. A student advising record will have first name, last name, and middle initial for a given student along with the student ID value (same format as UTA ID values). The advising record should also hold information, for each semester that the student has been active at UTA, that represents that specific semester and year, whether the student was required to see an advisor for that semester, where the student is advised (choice of UCOL or one of the Eng. departments), who their advisor is by name, the date they received advising for the semester, and notes and additional data fields to record any major changes made by the student during advising. {This would include professional program status, changes in major, addition of transfer classes, reason why a student withdrew for a semester, etc.} Use a collection class to hold information within the advising record. Only include the class(es)' data fields for this question. {7 points}

public enum Semester {FALL, SPRING, SUMMER, SUM1, SUM2, WINTER, MAY}

public enum AdvisingLocations {UCOL, UENG, MAE, BE, CE, CSE, EE, IE, MSE}

public class SemesterData

{

 private Semester currSemester;

 private int currYear;

 private boolean mustSeeAdvisor;

 private AdvisingLocations whereAdvised;

 private String advisorLastName;

 private GregorianCalendar advisingDate;

 private boolean inProfessionalProgram;

 private AdvisingLocations previousMajor;

 private String advisingNotes;

}

public class StudentAdvisingRecord

{

 private String firstName;

 private String lastName;

 private char middleInitial;

 private long studentID; // could be String or char array or int array but will need error check

 private ArrayList studentSemesterData; // will be an arraylist of semester objects

 private SemesterData aSemester;

}

5.b. Given the student advising record class with the data members listed above, give all the expected method signatures needed for the class including at least two constructors and a toString method. (You do not need to implement the methods listed.) {6 points}

public String getFirstName()

public String getLastName()

public char getMiddleInitial()

public long getStudentID() // could be String or char array or int array but will need error check

public SemesterData getSemester()

public ArrayList getStudentSemesterData(); // how to handle ArrayList

public String setFirstName(String first)

public String setLastName(String last)

public char setMiddleInitial(char init)

public long setStudentID(long idNum) // will need error check

public SemesterData setSemester(Semester currSemester, int currYear, boolean mustSeeAdvisor, AdvisingLocations whereAdvised, String advisorLastName, GregorianCalendar advisingDate, boolean inProfessionalProgram, AdvisingLocations previousMajor, String advisingNotes)

public ArrayList setStudentSemesterData(SemesterData semester) // arraylist

public StudentAdvisingRecord()

public StudentAdvisingRecord(String first, String last, char init, long idNum, ArrayList SemData )

public String toString()

5.c. Given the student advising record class with the data members and methods listed above, and assuming that your program has access to the students' transcript also (see below), pseudocode a method for the advising record class that determines the status of the student with regard to the professional program based on the following criteria:

 i. Student has completed at least 15 hours at UTA

 ii. All prerequisite courses for the professional program are completed. The prerequisite courses are in a SET listed by 4-digit course number.

 iii. All prerequisite courses are passed with a grade of A, B, or C

You may assume that student's transcript info is stored as an arraylist of completed courses where a "course" is a class containing a 4 letter dept. abbreviation, a 4 digit course number, an integer number of credit hours, an enum semester (FALL, SPRING, SUMMER, WINTER, MAY, SUM1STFIVE, SUM2NDFIVE, TRANSFER), a 4 digit year, and a final grade for the class. {8 points}

import java.util.ArrayList;

import java.util.Collections;

public class Transcript

{

 public ArrayList<Course> myClasses = new ArrayList<Course>();

 public Course mine = new Course(1230, Semester.SUMMER, 2012, 'A', "BE");

}

Assume there is a Set of professional program courses for each department per the question

Assume a Course has the info described in the question

 4 letter dept. abbreviation,

 a 4 digit course number,

 an integer number of credit hours,

 an enum semester ,

 a 4 digit year, and

 a final grade for the class

To determine if a student can be in the professional program, three criteria must be met.

1. Student has at least 15 hours credit at UTA
2. All the courses in the Set for the student's department are also in the student's transcript
3. All the courses in the Set and in the transcript are passed with A, B, or C grades.

// To check any of these criteria requires iterating through the courses in the transcript. So

// pseudocode must include some iteration:

Method to determine if a student is eligible to be moved into the professional program in their department:

Create empty student prerequisite Set

Create running total hours = 0

For each course in the student's transcript arraylist,

 If course is not TRANSFER,

 Sum the credit hours to the running credit hour total

 If course is in department professional program prerequisite class Set,

 Then If course grade is A, B, or C

 Add course to student's prerequisite class Set

If student's prerequisite class Set matches the department prerequisite class Set

AND If the credit hours total is >= 15,

 Student should be moved into the professional program

Extra Credit questions

XC1. What is one thing that would need to be defined about Course before you could run your method described above? Tell what would need to be defined and tell why this is needed. {2 points}

How to determine number of credit hours for a class. Cannot determine if student meets credit hour minimum unless you have credit hours.

XC2. What logical answer to the question CANNOT be given with the choices defined for the windows in question 1? Describe this unavailable answer in your own words. {3 points}

I never waited until the last minute

XC3. Define static and final in your own words and describe one way they are the same and one way they are different. {3 points}

Static -

Final -

XC4. What superpower would you like to have in order to be a better object oriented software designer? {Any answer will receive 2 points}