NATIONAL UNIVERSITY OF SINGAPORE

CS2103/T – SOFTWARE ENGINEERING

(Semester 2 AY2014/2015)

Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. Please write your Student Number only. Do not write your name.
2. This assessment paper contains **FIVE** questions and comprises **TEN** printed pages.
3. Answer **ALL** questions within the space in this booklet.
4. This is an OPEN BOOK assessment.

STUDENT NO: _____

This portion is for examiner's use only

Question	Marks	Remarks
Q1	/7	
Q2	/10	
Q3	/11	
Q4	/8	
Q5	/4	
Total	/40	

The **problem description** used for all questions in this assessment paper is given in the appendices (last two pages of this assessment paper).

Q1 [5+2=7 marks]:
Please read *Appendix A* of the problem description. When answering this question, you may consider the requirements specified in the problem description as well as requirements that you would like to add. Assume that you have categorized PointTracker user stories into categories *must-have*, *nice-to-have*, and *unlikely-to-have*.

a) Give one *nice-to-have* user story for each of the four most important user types. Three user types to include are Lecturer, Tutor, and Student. You may decide the fourth user type.

b) Give a non-functional requirement for PointTracker. Choose a requirement that is more specific to PointTracker, as opposed to a requirement that is applicable to most other software.

Q2 [2+8=10 marks]:

(a) Consider the two architecture choices given in *Appendix B*. Which one is better? Justify your answer.

(b) Assume that when a tutor runs the PointTracker executable, it initializes the components, downloads the relevant data from the remote server, and shows the student list of the class. Use suitable UML diagram(s) to explain the interactions between architectural components required during that scenario. Remember to include the API functions taking part in the interactions.

Q3 [8+3=11marks] (a) Use suitable UML diagrams to propose an object-oriented design for the `Data` component. Include navigabilities, multiplicities, association roles/labels etc. when they add value to the diagram. Methods and attribute details need not be shown. Your design should include the following class.

> Assessment: An object of this class represents the assessment by a tutor for a specific tutorial task for a specific team.

(b) Consider the sample data given in *Appendix C* and your proposed design above. Use a suitable UML diagram to show the object structure in the `Data` component for the given sample data.

Q4 [3+5=8 marks]:

(a) Consider testing of the assignPoints function given in *Appendix D*. Give equivalence partitions for each parameter.

Parameter	Equivalence partitions
teamID	{belongs to a team in the class}
tutorUsername	
points	

(b) Assume you have been asked to unit test the assignPoints function and the software has been loaded with the sample data given in *Appendix C*. Add 7 more unit tests to the example given. Try to maximize the efficiency and effectiveness of testing when you design the test cases.

#	teamID	tutorUsername	points	expected
1	team1	tutor1	5	true
2				
3				
4				
5				
6				
7				
8				

Q5 [4 marks]:

Now, assume there is a need for the Data component to notify the Connector component when there is a change to the data, without adding a direct dependency from Data component to the Connector. Illustrate a way to achieve that using appropriate diagrams.

Hint: Use a suitable design pattern.

---Use this page if you need extra space for any of the questions---

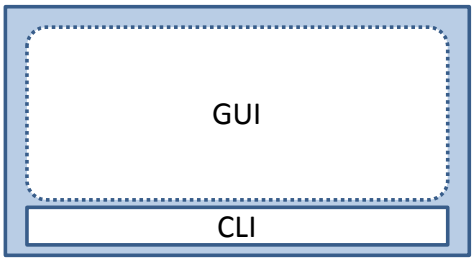
---Use this page if you need extra space for any of the questions---

--- End of Paper---

Given below is the **problem description** used for all questions in this assessment paper. You may detach this page from the assessment paper. There is no need to submit this page.

Appendix A: PointTracker product description

Your team has been asked to build a software application called **PointTracker** to keep track of CS2103 participation points. It has a desktop application and a remote server. Here is the expected behavior of PointTracker.



1. Users access PointTracker using the desktop application. The remote server does not have a user interface.

2. PointTracker desktop application has a GUI and a Command Line Interface (CLI).

3. Tutors use the PointTracker desktop application to enter participation points students earned for each tutorial task. After the lecturer publishes the points for a particular tutorial task, students can use the PointTracker desktop application to view their points.

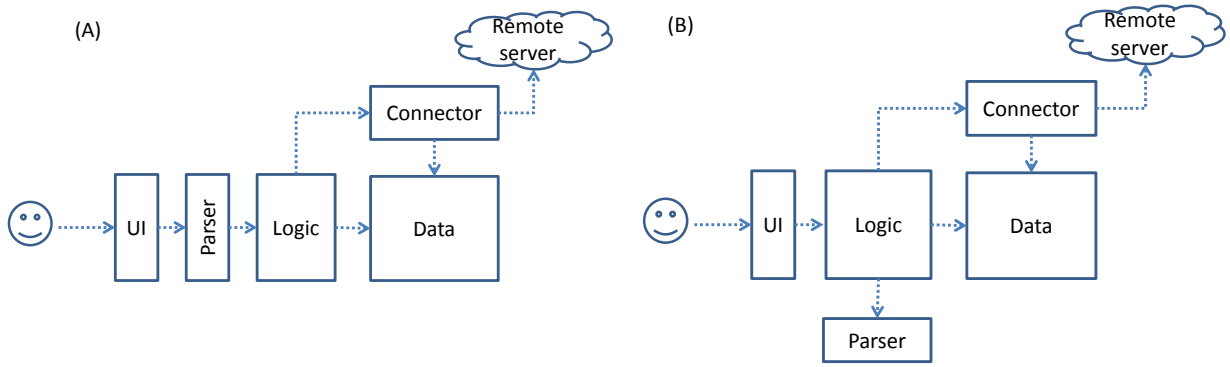
4. PointTracker desktop application does not store data locally (i.e. in the user's Computer). Instead, it connects to the PointTracker remote server to read/write data.

5. Authentication to the remote server is done using the login credentials the user used to log in to the computer. PointTracker picks up the username and password from the Operating System. There is no need for the user to type in username and password when using PointTracker.

6. The proposed PointTracker system has the following simplifications:
    a. It deals with data of the current semester only. Data of the past semesters are deleted at the end of the semester.
    b. It deals with team-based tutorial tasks only. It will not deal with individual tutorial tasks.
    c. The tutor allocation can change over time. A tutor may assess a different set of teams each week. However, each team (comprising 2 to 5 members) is allocated a tutorial slot (e.g. Wed 9am) which does not change for the entire semester.
    d. PointTracker keeps track of which tutor gave how many points to which tutorial task of which team.

Appendix B: Architecture choices

Given below are two architecture choices for PointTracker. The difference is in how the Parser component is connected.



An overview of the components is given below. It applies to both diagrams.

- `Parser` : Parses the CLI commands typed by the user
- `Data` : The in-memory container for objects representing PointTracker data. E.g. Tutors, Students, Tutorial Tasks, Assessments, etc.
- Remote Server: Represents an online server that stores PointTracker data. It is accessed by PointTracker using a remote API, which includes the following function.

  getData(username, password): String

  Returns all relevant data of the user in a String format (e.g. XML or JSON) if the username and the password are correct.
- `Connector`: Handles the connection with the remote server API.
- UI, Logic: As the names suggest.

Appendix C: Sample data

Given below are a set of sample data that exists in PointTracker at a particular point of time.

- Usernames of the tutors in the class: tutor1, tutor2
- Usernames of the students in team1: s1, s2
- tutor1 gave 5 points for task1 for team1

Appendix D: Testing the assignPoints function

assignPoints(string teamID, string tutorUsername, int points):boolean
Returns true if all the following conditions are true.

- teamID belongs to a team in the class
- tutorUsername belongs to a tutor of the class
- points are in the range 0..10 (both inclusive)
- tutor indicated by the tutorUsername is assigned to assess the team indicated by teamID

-- End of problem description--