UIT2201: Computer Science and Information Technology Revolution (Spring Semester 2013)

MID TERM TEST (1 hour)

INSTRUCTIONS:

1. This Mid-Term Test is CLOSED BOOK / CLOSED NOTES.
2. Answer ALL questions in this answer book.
3. Make sure you write down your NAME and MATRIC NUMBER

Name: ____

Matric Number: __________________________

QUESTION	POSSIBLE	SCORE
Q1	20	
Q2	15	
Q3	15	
Q4	10	
TOTAL	60+1	

Fun Question: (1 bonus point)

Name the university I visited in Jakarta during the one-week break. Or make a guess.

(a) ____

(Now, please *relax* and *enjoy* the Quiz, OK?)

Question 1: (20 points)

True-False (2 point each)

(z) ***(Sample)*** *The course UIT2201 is taught by Prof. Leong Hon Wai* **_*TRUE*__**

(a) Computer Science is the *study and use* of computers and computer programs and applications. ________

(b) Computing the sum of squares of n numbers ($X_1, X_2, \ldots, X_n$), namely, finding ($X_1^2 + X_2^2 + \ldots + X_n^2$) can be done in $\Theta(n)$ time, namely, in *linear* time. ________

(c) One important virtue of an *algorithm* is that if we can specify an algorithm to solve a problem, then we can *automate* the solution. ________

(d) In Scratch, a sprite A can ask another sprite B to start executing, then wait until B finishes before A executes the next action. ________

(e) An algorithm **A** with running time $2.013n^2$ will be *faster* than an algorithm **B** with running time $2013n(\lg n)$ when n is sufficiently big. ________

(f) The binary search algorithm is so-named because it uses only *binary operations* and *binary numbers*. ________

(g) When answering SQL queries, the *join* operation is usually the most expensive operation. ________

(h) Each *row* of a database table is called an *attribute*. ________

Fill in the blanks: (2 points each)

(i) Name the fun feature/characteristic of Scratch that you personally like the most.

Answer: ____

(j) Name one aspect of Scratch that you like to be improved upon. Give ***short*** description.

Answer: ____

Question 2: (15 points)

In the USP, there are 5 committees, each committee consisting of different faculty and staff members (represented by letters A, B, C, D, E, G, H, J), as given below:

DDC = { A, B, G } ***CRC*** = { B, C, E} ***SLC*** = { C, D, H }
ORC = { A, D, J } ***BWC*** = { A, E, J }

Each committee wants to schedule a 1-hour meeting every Monday morning. However, because some committee members belong to multiple committees, some of the meetings cannot be held concurrently. A simple way is to schedule each meeting at a different time slot (using 5 time slots). However, it is better to finish all the meetings in the *fewest* number of time slots possible (so that the members can have time do other work every Monday morning). Thus, the *Meeting Scheduling Problem* (MSP) is to schedule the 1-hour meetings for all the committees (and each member is able to attend all the committee meetings) with the *fewest* number of time slots.

(i) [3] Show how we can model the conflicts in this problem with a graph $G = (V, E)$. Each vertex in V in the graph models a committee in the USP. Explain clearly how the edges are defined, namely, under what condition do we connect two vertices u and v?

(b) [5] Draw the graph for the example of the 5 committees in the USP.

(c) [3] Explain how to solve the meeting scheduling problem using the graph model.

(d) [4] Give a meeting schedule that uses the minimum number of time slots.

Question 3: (15 points)

A "big" company, **BB** has a (sorted) customer list **`LL`** with **10 names** and uses *binary search algorithm* for searching. A "small" company, **SS**, has an unsorted customer list **`TT`** with **5 names** and uses *sequential search algorithm* for searching. (The numbers are intentionally *kept small* for this Quiz; else these are likely to be 1,000,000 and 1,000, respectively.)

When the companies **BB** and **SS** merged to form **BSBS**, they want to combine their customer lists. You are brought in to develop algorithms for searching the ***combined list***. But, you only have a short time to develop the algorithms, so your initial approach is to "combine" the searches with a quick-and-dirty solution.

Your quick-and-dirty idea, which you call **Hybrid-BB-SS**, is to *first* search list **`LL`** (using BB's binary search algorithm). If it is unsuccessful, *then* search the list **`TT`**. (using SS's sequential search algorithm). The resulting pseudo-code is as follows:

Algorithm Hybrid-BB-SS (`aName`) **(*`aName` is a given name to search*)**
1. Use *binary search* to search for **`aName`** in the list **`LL`**;
2. If unsuccessful, use *sequential search* to search for **`aName`** in the list **`TT`**;

(a) [4] Draw the ***search tree*** that is used to visualize this **Hybrid-BB-SS** algorithm applied to the sorted list **`LL`** of size `10` and unsorted list **`TT`** of size `5`. (Include the "square" nodes for unsuccessful searches).

(b) **[4]** What is the ***number of comparisons*** needed for a *successful* search in (i) the worst-case and (ii) average-case? (Assume that all names are *equally likely* to be searched.)
[You can leave your answers in fractions.]

Worst Case: _____________ comparisons

Average Case: (Hint: sum and divide by 15)

(c) **[2]** What is the ***number of comparisons*** needed, in the ***worst case***, for an ***unsuccessful search***?

Worst Case: _____________ comparisons

(d) **[3]** Now, suppose that you have more time to develop a much better algorithm than the **Hybrid-SS-BB** algorithm given above. Informally, describe your better algorithm or give a *high-level pseudo-code* of the algorithm (similar to pseudo-code for (a).)
[There is *no need to give detailed code* for this problem.]

(e) **[2]** What is the ***number of comparisons*** needed for a successful search ***using your algorithm*** in the worst-case?

Worst Case: _____________ comparisons

Question 4: (10 points)

Given a database with the following 3 tables: `{SI, CI, EN}`. You should use these short table names to save space and writing. (Use the reverse blank pages, if necessary)

SI (STUDENT-INFO)						
Student-ID	**Name**	**NRIC-No**	**Address**	**Tel-No**	**Faculty**	**Major**
---	---	---	---	---	---	---

CI (COURSE-INFO)					
Course-ID	**Name**	**Day**	**Hour**	**Venue**	**Instructor**
---	---	---	---	---	---

EN (ENROLMENT)	
Student-ID	**Course-ID**
---	---

For **(a)** and **(b)** below, give the appropriate (i) **SQL query**, and (ii) a sequence of basic database primitives (using **`e-project, e-select, e-join`**):

(a) [4 pts] List the **Student-ID, Name, Major** of all students whose major is "`UBW`".

(i) SQL Query:

(ii) Using Basic Primitives:

(b) [6 pts] List the **Course-ID, Student-ID, Name** (of student), **Major** of students whose Major are "`SSS`" and have class on "`Wed`".

(i) SQL Query:

(ii) Using Basic Primitives:

~~~ END OF QUIZ ~~~