UIT2201: Computer Science and Information Technology Revolution
Fall Semester 2014 – Mid-Term Quiz (Solution Sketch)

(NOT TO BE GIVEN TO FUTURE UIT2201 STUDENTS)

Fun Question: To decide (♦♣♥♠), use pigeonhole principle.

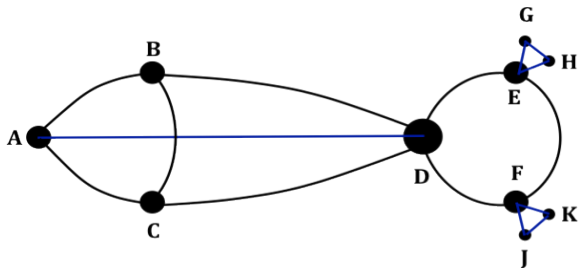
Question 1: (a)--(f) F T T T T F

(i) Having ITEM – IT Enabled Mindset (always be aware of IT and what it can do)
Looking at the real-life problem from an IT perspective
may open up new way of thinking about the problem,
may open up new possible solution that uses IT;
or a combination of traditional methods combined with IT methods.
Solving part or the whole problem with the help of IT,
get more efficient workflows with some or all the tasks automated.
may get good cost savings, more efficient use of manpower.

Question 2: (15 points)

(a) [3] Answer: We connect two vertices *u* and *v* when
the items represented by u and v conflict with each other (*they cannot be put together*).

(b) [5]



Conflict graph $G = (V, E)$

(the inspiration: imbellis)

(c) [3] We colour the vertices in the conflict graph *G* so that no two adjacent vertices have the same colour. And we want to *minimize* the number of colours. The vertices of the same colour do not conflict and so the corresponding items can be put together in the same box. Hence, by minimizing the number of colours, we minimize the number of boxes.

(d) [4] Graph colouring step – DIY. We can pack with 4 boxes. There are many possible graph colouring with 4 colours (using 4 boxes). Below I show one of them.

Box-1: A, E, J
Box-2: B, F, G
Box-3: C, H, K
Box-4: D

Question 3: (15 points)

(a) [3] ***Search Tree:*** DIY: State if *left*-middle-elt (right-heavy) or *right*-mid-elt (left-heavy).

(b) [3] *Successful* search **Worst Case:** 3 comparisons
Average Case: (Assume all names equally likely)
(1 + 2 + 2 + 3 + 3 + 3) / 6 = 14/6 = 7/3 = 2.333

(c) [3] *Unsuccessful* search **Worst Case:** 3 comparisons
Average Case: (Assume all gaps equally likely)
(2 + 3 + 3 + 3 + 3 + 3 + 3) / 7 = 20/7 = 2.857

(d) [3] After iteration k=1, $\boldsymbol{A}$ = [8, 6, 2, 4, 7],
After iteration k=2, $\boldsymbol{A}$ = [8, 6, 2, 4, 7],
After iteration k=3, $\boldsymbol{A}$ = [8, 6, 4, 2, 7],
After iteration k=4, $\boldsymbol{A}$ = [8, 6, 4, 7, 2],
Answer: $\boldsymbol{A}$ = [8, 6, 4, 7, 2]
Output of "Print A[n]": 2

(e) [3] Worst Case running time: $\Theta(n)$

Question 4: (15 points)

(a) [6 pts]
(i) SQL Query:

```
SELECT  SI.Name, SI.Address, SI.Tel-No
FROM    SI, EN
WHERE  (SI.Faculty = "Engg") AND
       (EN.C-ID = "UIT2201") AND
       (SI.S-ID = EN.S-ID);
```

(ii) Using Basic Primitives:

```
G1  ← e-select from EN where (C-ID = "UIT2201");
G2  ← e-select from SI where (Faculty = "Engg");
G3  ← e-join G1 and G2 where (G1.S-ID = G2.S-ID);
Ans ← e-project SI.Name, SI.Address, SI.Tel-No from G3;
```

(b) [4] Putting in two records in CI with Course-ID of "CS1001" violates the fact that Course-ID is the primary key.
In EN, cannot know which lecture-section A222333 is assigned to since it is only identified with the Course-ID. Can only identify all the students enrolled in CS1001, but cannot know wich section.
If we join CI and EN, then result is that A2223333 will be taking both lecture-sections, which is a mistake.

(c) [5] Idea: Create a unique ID for each lecture-section; this should be primary key for CI.

Solution: Create a new field called **`Session-ID`** for table CI. This will become the new ***primary key*** for CI. (Note: **`Course-ID`** is not primary key any more since it is no longer unique.) Then in EN, replace **`Course-ID`** by **`Session-ID`**. Students are then assigned to lecture-sections by their Session-ID. This will solve all the problems mentioned above.

~~~ END OF QUIZ ~~~