THE UNIVERSITY OF TRINIDAD & TOBAGO

ALTERNATE
FINAL ASSESSMENT/EXAMINATION
SEPTEMBER – DECEMBER 2012

Course Code and Title: DSAL3001, Algorithm Analysis and Design

Programme: Bachelor of Applied Science in Computer Engineering

Date and Time: [Please insert in accordance to the timetable] **Duration:** [Please ensure accordance with time slot]

PLEASE READ ALL INSTRUCTIONS CAREFULLY BEFORE YOU BEGIN THIS EXAMINATION

Instructions to Candidates

This paper has 5 pages and 7 questions.

1. **You are required to answer ALL questions.**

Key Examination Protocol

1. Students please note that academic dishonesty (or cheating) includes but is not limited to plagiarism, collusion, falsification, replication, taking unauthorised notes or devices into an examination, obtaining an unauthorised copy of the examination paper, communicating or trying to communicate with another candidate during the examination, and being a party to impersonation in relation to an examination.

2. The above mentioned and any other actions which compromise the integrity of the academic evaluation process will be fully investigated and addressed in accordance with UTT's academic regulations.

3. Please be reminded that speaking without the Invigilator's permission is **NOT** allowed.

Instructions: Answer all questions

1. Given the following numbers below:

13	28	14	16	12	8	20	44	26	19

   a. Show the step by step configuration of the list after a Quick Sort algorithm has been applied to the list. [5]

2. T(n) is the running time of the algorithm on an input size n, Given that

   $T(1) = 1$

   $T(n) = 2\,T(n + 2) + c$, for $n > 1$ and a constant c

   a. Derive the formula for T(n) and determine the order of the algorithm, state any assumptions made. [5]

3. Given the following Algorithm determine its asymptotic order
Let k be some positive integer [5]

```
read(n)
a = 0;
while (n >= 1)
{
        for (j=1; j<=n; j++)
                a++;
        n = n/2;
}
```

4. The *n*-queens problem is to place *n* queens on an *n*-by-*n* chessboard so that no two queens attack each other by being in the same row or in the same column or on the same diagonal. Write a Java code that uses the backtracking strategy to solve the 4-queens problem. [10]

5. In a treasure hunt, a neighborhood is divided into a rectangular field of m rows and n columns. Starting from a house on the top column, a player must move to a house to the bottom collecting 'treasure' as he goes. At each step a player can moved diagonally to either of the following.
    a. The house to his left in the next row provided he is not already at the extreme left
    b. The house to his right in the next row provided he is not already at the extreme right

   Each house has a treasure point assigned to it, p where 0<=p<=50. When a player moves to a square, he must add its point value to his score.

   Assume that the rows of the houses are numbered 1 to m, going from the top to the bottom and columns are numbered 1 to n, from left to right. T[i,j] contains the treasure point value of the house in the i^{th} row and j^{th} column.

    a. Describe how a binary counter can be used to solve this problem. [5]

    b. Write Java code to determine the maximum score a player can get as he moves from any square in the left column to a square in the right column. [10]

    c. Write Java code which uses an efficient algorithm to solve this problem. [10]

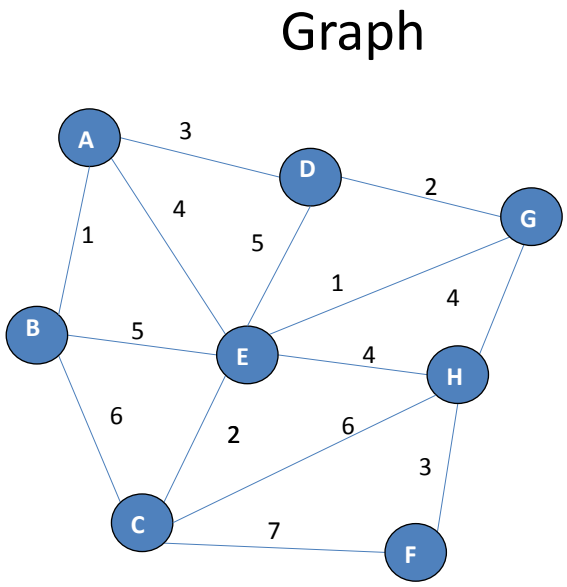
6. A forensic investigator has discovered DNA material at a crime scene. The DNA needs to be matched with other samples collected in order to determine whether there were additional suspects involved. The investigator needs to determine the longest common subsequence in order to determine the probability of a match. The patterns are stored in 2 arrays A[1..m] and B[1..n].

    a. Write a Java method to compute a table L[i,j] which tracks the length of the longest common subsequence. [10]

    b. Construct the table L[i,j] using the data provided in the table above [5]

    c. Write a Java method that traverses the table and outputs the longest common subsequence. [10]

7. The diagram below shows an undirected acyclic graph.



a. Using Dijkstra's algorithm, compute the shortest path to all other nodes starting from node A. [8]

Clearly show the state of the minimum priority queue, the distance and parent nodes for each pass of the algorithm.

b. Write Java code that outputs the shortest part from node A to every other node using the information in part (a). [7]

c. Given the adjacency matrix below show the derivation of a topological sort on the modified graph. Clearly show the parent nodes, queue and state during construction [10]

A	B	D	E
B	C	E	
C	F		
D	E	G	
E	D	H	
F			
G	H		
H	F		