15/5/2019

14.00 - 16.00pm

CMPU 2001 Algorithms & Data Structures

Basement 3, Kevin Street

Programme Code: DT228 & DT282
Module Code: CMPU 2001
CRN: 22395 & 26458

TECHNOLOGICAL UNIVERSITY DUBLIN

KEVIN STREET CAMPUS

BSc. (Honours) Degree in Computer Science
BSc. Degree International in Computer Science

Year 2

SEMESTER 2 EXAMINATIONS 2018/19

Algorithms & Data Structures

Mr. Richard Lawlor
Dr. Deirdre Lillis
Dr. Martin Crane

Instructions

Attempt three out of four questions
All questions carry equal marks
One complementary mark for paper

Two hour exam

1. (a) Write a simple Java interface to express the services provided by the *Abstract Data Type (ADT) Queue*. You can assume the queue stores *int* values.

(5 marks)

(b) Provide a Java class which implements the *Queue* interface based on a linked list implementation but showing only data structure, constructor and no other methods.

(5 marks)

(c) Give the implementation of the method *deQueue()* for the class in part (b). Explain why you might use or not use a tail reference in your implementation. What is the complexity of *deQueue()* with and without the use of a tail pointer?

(9 marks)

(d) Provide a Java implementation of the *insert(int x)* operation for a sorted linked list. You can assume the list used a sentinel node which points to itself.

(9 marks)

(e) Write down a simple equation for a Stack in terms of *pop()* and *push()* that expresses the last-in-first-out (LIFO) behaviour of the Stack.

(5 marks)

2. (a) Show how binary search works when searching for ***17*** in the following array:

1	5	6	9	10	12	14	17	21

(4 marks)

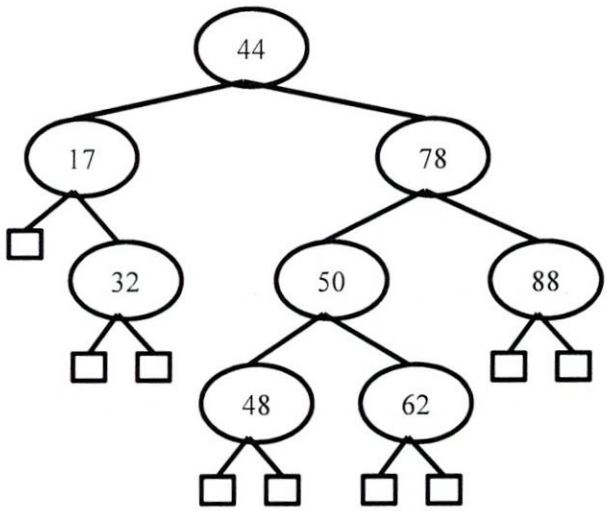
(b) What is a binary search tree (BST)? Mention any specific advantage or possible disadvantage. What is the complexity of searching a BST?

(6 marks)

(c) Write in pseudocode the algorithm for searching a BST.

(6 marks)

(d) Given the following binary search tree, show how it would be modified by inserting 54.



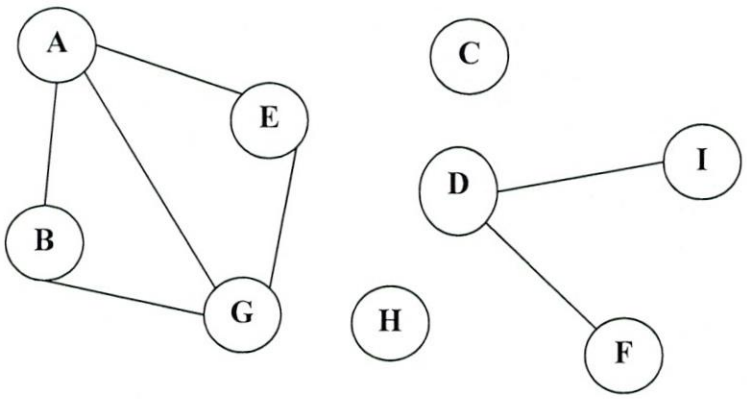
(5 marks)

(e) What is an AVL-tree? Include in your answer the idea of a *rotation*.
Show how the tree that results from inserting 54 in part (d) would be rebalanced if it were an AVL-tree.

(12 marks)

3 (a) Provide in pseudocode or Java a modification of Depth First Graph Traversal so that it counts the number of disconnected components in a graph and labels each vertex with the subgraph component it belongs to.

Then show how the algorithm you described behaves on the following graph.



(11 marks)

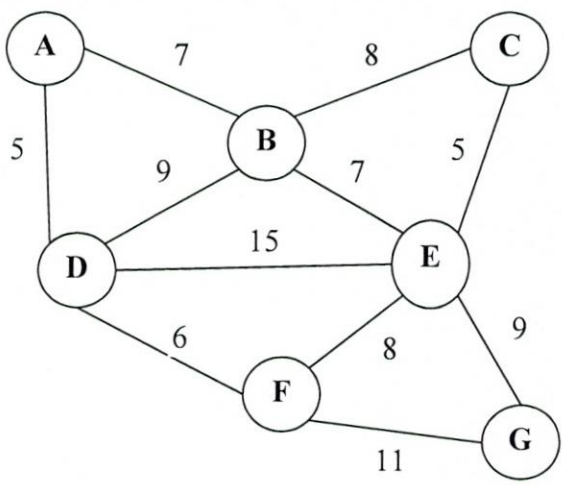
(b) Explain the *Union-Find* data structure and what it is used for in Kruskal's algorithm. Also, with the aid of diagrams, outline a possible implementation of this data structure and give an example showing how its two significant operations work. No code required here.

(10 marks)

(c) Using pseudocode or Java, write down the implementation of the class constructor and the two significant operations of the *Union-Find* data structure described in part (b).

(6 marks)

(d) Illustrate in detail how Kruskal's algorithm computes a Minimum Spanning Tree for the graph below showing the contents of the union-find sets and their representation for the first 4 iterations of the algorithm's while loop.



(6 marks)

4. (a) Explain the terms:

i) priority queue
ii) complete binary tree
iii) heap
iv) heap condition

(5 marks)

(b) Draw the following heap array as a two-dimensional binary tree data structure:

k	0	1	2	3	4	5	6	7	8	9	10	11
a[k]		13	10	8	6	9	5	1				

Also, assuming another array hPos[] is used to store the position of each key in the heap, show the contents of hPos[] for this heap.

(6 marks)

(c) Write in pseudocode the algorithms for the *siftUp()* and *insert()* operations on a heap and show how hPos[] would be updated in the *siftUp()* method if it was to be included in the heap code. Also write down the complexity of *siftUp()*.

(9 marks)

(d) By using tree and array diagrams, illustrate the effect of inserting a node whose key is **12** into the heap in the table of part (b). You can ignore effects on hPos[].

(6 marks)

(e) Given the following array, describe with the aid of text and tree diagrams how it might be converted into a heap.

k	0	1	2	3	4	5	6	7	8
b[k]		2	9	18	6	15	7	3	14

(7 marks)