DUBLIN INSTITUTE OF TECHNOLOGY

DT228 BSc. (Honours) Degree in Computer Science

Year 2

DT282 BSc. (Honours) Degree in Computer Science (International)

Year 2

SUMMER EXAMINATIONS 2015/2016

ALGORITHMS & DATA STRUCTURES [CMPU2001]

MR RICHARD. LAWLOR

FRIDAY 13TH MAY 4.00 P.M. – 6.00 P.M.

TWO HOURS

ATTEMPT THREE OUT OF FOUR QUESTIONS
ALL QUESTIONS CARRY EQUAL MARKS
ONE COMPLEMENTARY MARK FOR PAPER

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 partial circular buffer implementation of ADT *Queue*. Show the data structure used and the code for *deQueue()* only.

(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)** 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)

3. (a) Provide a brief explanation on how Kruskal's MST algorithm works and then write the algorithm in pseudocode.

(8 marks)

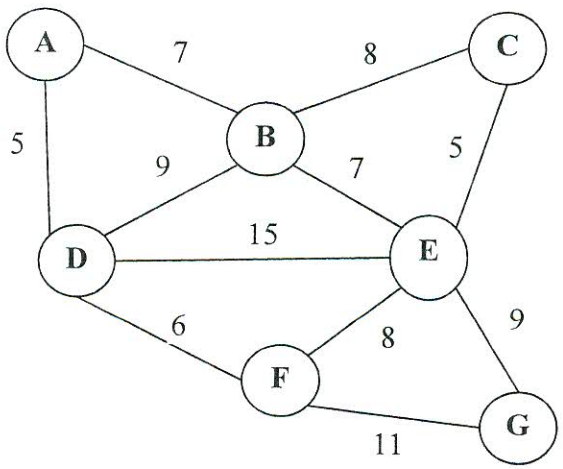
(b) What representation of a graph is appropriate for implementing Kruskal's algorithm? What is the complexity of creating this data structure and removing the edges as required from it?

(5 marks)

(c) 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.

(10 marks)

(d) Illustrate in detail how Kruskal's algorithm computes a MST for the graph below showing the contents of the union-find sets at each stage.



(10 marks)

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

1	5	6	9	10	12	14	17	21

(5 marks)

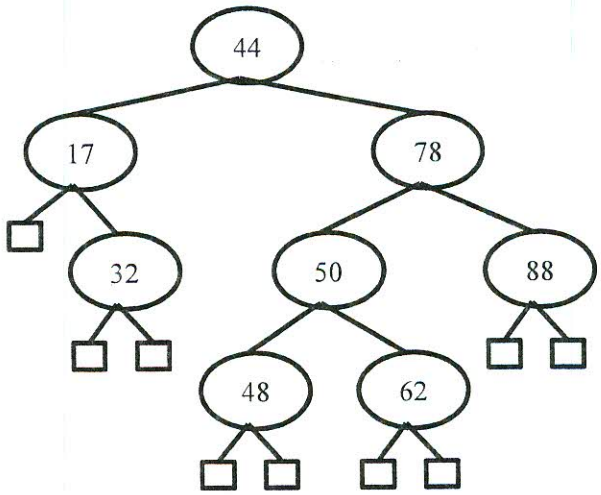
(b) What is a binary search tree (BST)? Mention any specific advantage or possible disadvantage. What is the complexity of searching a BST?

(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.

(11 marks)