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 2016/2017

ALGORITHM DESIGN AND DATA STRUCTURES [CMPU2001]

MR. RICHARD LAWLOR
DR. DEIRDRE LILLIS
MR. PAUL COLLINS

FRIDAY 12TH 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 THE 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) Using diagrams, show the detailed workings for the first two iterations of the outer loop of a bubble sort on the following array.

9	8	5	2	7	3	4	1	0

(4 marks)

(b) Explain what is meant by *Tortoises and Hares* with reference to bubble sort. Provide a simple example to illustrate.

(4 marks)

(c) Write an adaptation of bubble sort in pseudocode which helps with the problem of *Tortoises and Hares*.

(10 marks)

(d) Show how your algorithm from part (c) sorts the above array.

(5 marks)

(e) Using diagrams, show the array in part (a) can be converted to a heap.

(5 marks)

(f) Using diagrams, show how heap sort works on the heapified array from part (d).

(5 marks)

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

(8 marks)

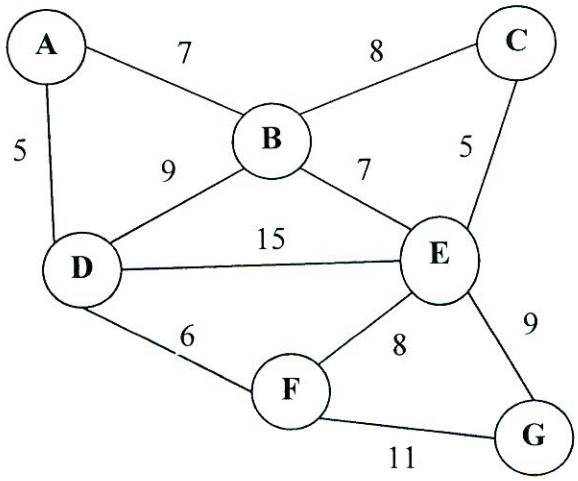
(b) What representation of a graph is appropriate for implementing Kruskal's algorithm? What is the complexity of creating this data structure and removing the edges as required from it?

(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)

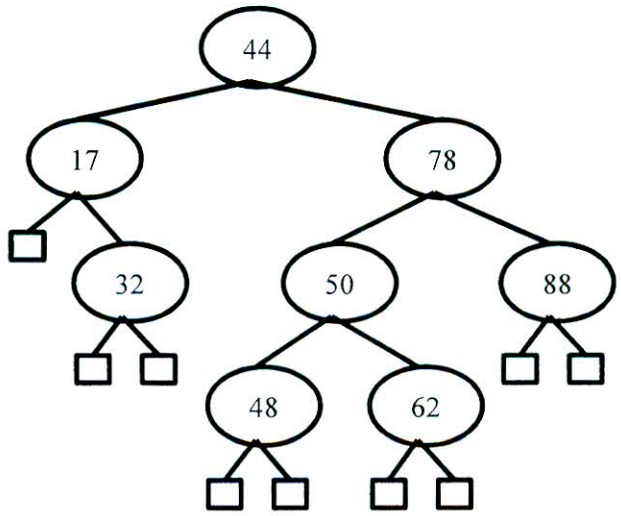
(b) What is a binary search tree (BST)? Mention any specific advantage or possible disadvantage. What is the complexity of searching a BST?

(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)