FACULTY OF SCIENCE

ACADEMY OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

MODULE	CSC03A3/CSC3A10 COMPUTER SCIENCE 3A
CAMPUS	AUCKLAND PARK CAMPUS (APK)
ASSESSMENT	SEMESTER TEST 2

DATE: 2020-05-21	**SESSION:** 14:00 - 16:00
ASSESOR(S):	PROF D.T. VAN DER HAAR MR R. MALULEKA
INTERNAL MODERATOR:	PROF D.A. COULTER
DURATION: 120 MINUTES	**MARKS:** 100

Please read the following instructions carefully:

1. This is a time restricted open book assessment. Answer **all** the questions in a text processor or on paper, which is scanned and submitted.
2. Write *cleanly* and *legibly* on any handwritten parts (if applicable).
3. This paper consists of 4 pages.
4. Ensure that your submission to **Eve** is *complete* and done *before* the cut-off time. Do not forget to fill in the honesty declaration form.

QUESTION 1

(a) Define **total order relations** within the context of the Priority Queue ADT, along with three (3) examples. (6)

(b) Briefly describe two (2) **operations** in an adaptable priority queue, which are **NOT** found in a regular priority queue. (4)

(c) Illustrate the execution of the **bottom-up construction of a heap** on the following sequence. You only need to provide a graphic representation of the heap at each stage in the construction, including any intermediate operations. (10)

4, 6, 1, 10, 7, 57, 43, 23, 15, 9, 8, 32, 2, 35, 36

Total: 20

QUESTION 2

(a) Briefly define a **Map ADT**. Explain the **performance** of the **put** and **get** methods for a list based map. (5)

(b) Given a hash function $h(x) = xmod11$ for a hash table that uses **linear probing**, redraw the hash table below and **insert** the following keys (in this order): (8)

54, 67, 38, 46, 2, 18, 71, 16

0	1	2	3	4	5	6	7	8	9	10	11

(c) Would quadratic probing improve the **worst case** computational complexity of the above insert operation? (2)

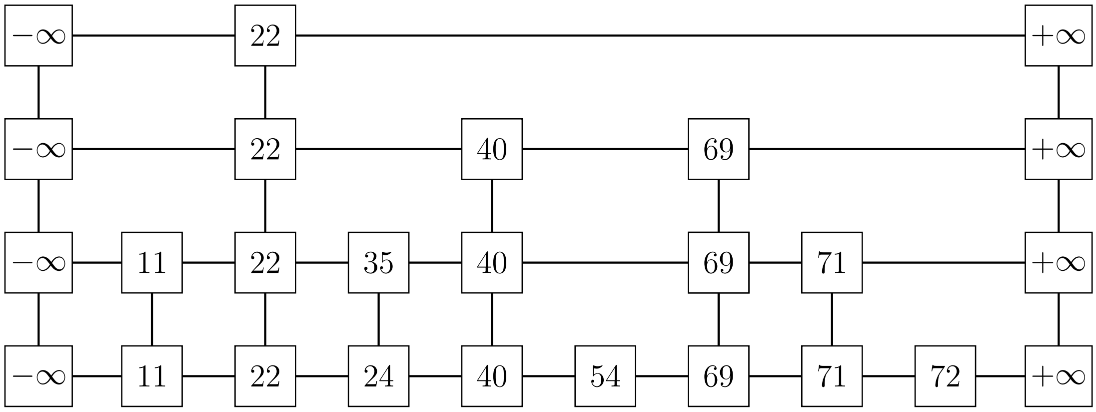
Total: 15

QUESTION 3

(a) Discuss how the **Search Table's** find operation achieves $O(logn)$ worst case performance. Provide an application example where a Search Table can be used effectively. (5)

(b) Would it be a good idea to make a doubly-linked list-based **hash table**? Provide a **justification** for your answer. (4)

(c) Analyse the skip list below and illustrate using diagrams how you would **remove** an entry with a key of 71 (the sentinel sequence is omitted). (6)

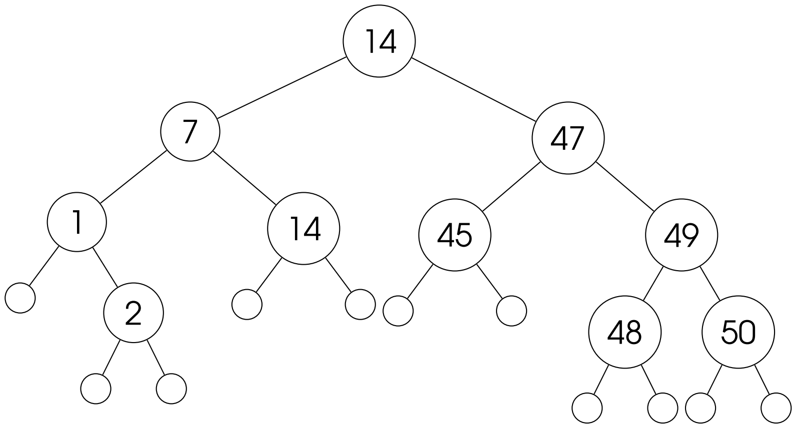


Total: 15

QUESTION 4

(a) For the Binary Search Tree below, **draw** the tree state (and any operations applied) for every step in a **remove** operation for each of the following keys (where each removal follows one another): (6)

7, 47



(b) **Draw** the tree state (and any operations applied) for every step to **insert** the key 48 into the above Binary Search Tree (the one shown in the question 4a, not the answer). *Hint - duplicates are allowed* (4)

(c) Define the two **properties** that make a Multiway Search Tree a (2,4) tree, along with **why** the worst case performance of most operations in a (2,4) tree are $log(n)$. (5)

Total: 15

QUESTION 5

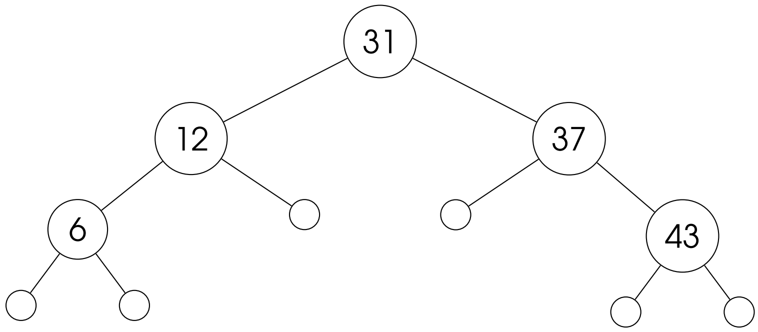
Consider the following AVL tree provided below. Draw the AVL tree state after each of the following operations. If the tree is rebalanced draw the state before and after it being balanced. Removal operations should follow from the tree that resulted from the insertion operations.

1. Insert nodes that contain the following keys: (inserted one-by-one, in the given order)

   35, 12, 43, 42, 20

2. Delete nodes that contain the following keys: (removed one-by-one, in the given order)

   42, 12, 6, 31



Total: 15

QUESTION 6

Consider the following (2,4) tree provided below. Draw the (2,4) tree state after each of the following operations. If the tree is rebalanced draw the state before and after it being balanced.

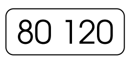
1. Insert nodes that contain the following keys: (inserted one-by-one, in the given order)

   122, 90, 48, 70, 138

2. Delete nodes that contain the following keys: (removed one-by-one, in the given order)

   138, 120, 90

The 2-4 tree is in the current state:



Total: 15

QUESTION 7

Provide the visual **graph representation** for the following matrix:

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{bmatrix} A & B & C & D & E \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 3 & 1 & 0 \\ 1 & 3 & 0 & 3 & 3 \\ 0 & 1 & 3 & 0 & 2 \\ 0 & 0 & 3 & 2 & 0 \end{bmatrix}$$

Total: 5

— End of paper —