Student Number:

The University of Melbourne

COMP20007: Design of Algorithms

Semester 1 Sample Assessment 2019

Reading Time 15 minutes.

Writing Time Three hours.

This paper has 35 pages including this cover page.

Authorised Materials: None.

Instructions to Invigilators:
Students will write all of their answers on this examination paper. Students may not remove any part of the examination paper from the examination room.

Instructions to Students:
This paper counts for 60% of your final grade. All questions must be answered in the indicated answer boxes provided on the examination paper. Answer each of the following questions by writing a brief response or explanation (no essays please!). Only material written inside the boxes will be marked. If you need to make rough notes, or prepare draft answers, you may do so on the reverse of any page. If you need additional space for your answers, you may use the overflow section on the last page.

Paper to be held by Baillieu Library: No.

Examiner use only:

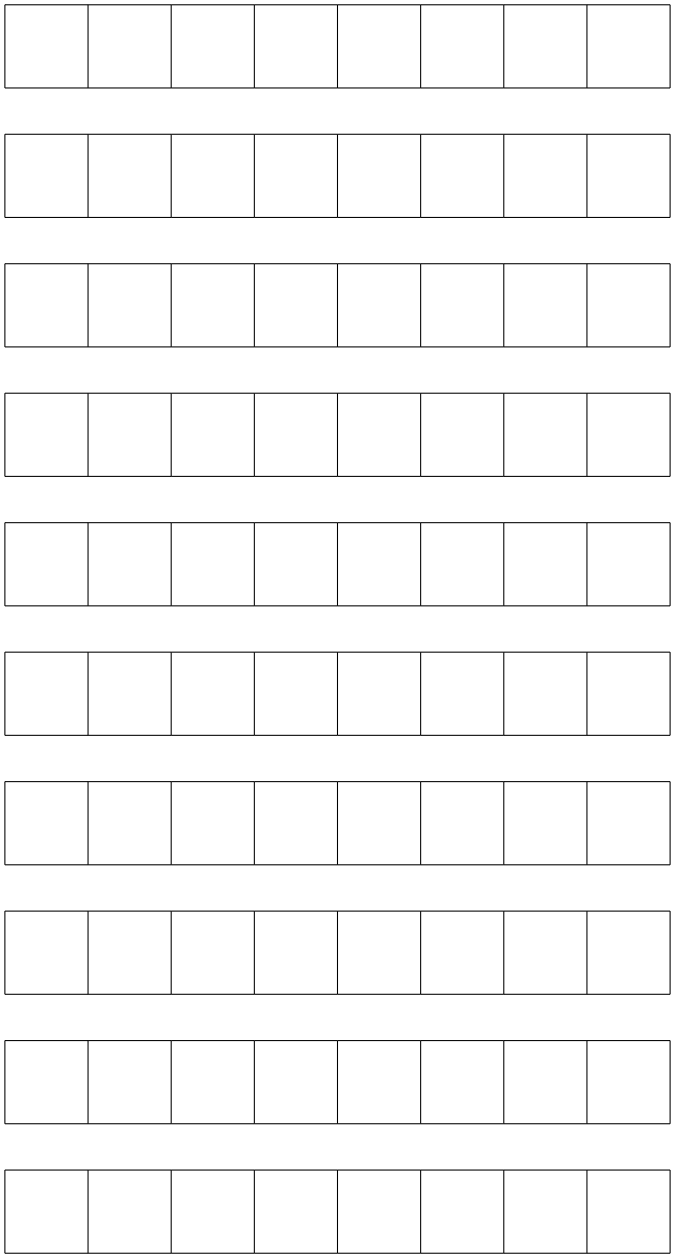
Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8

This page is blank (almost)

Question 1 (8 marks).

a. Perform mergesort on the array $[25, 23, 17, 29, 7, 16, 33, 21]$, showing the result after each recursive call. Indicate which range of elements are being considered during each recursive call by outlining those elements. Only fill the arrays which are needed.

[2 marks]



b. What are the two operations which a queue must implement?

[1 mark]

This page is blank (almost)

c. Describe how you would implement a queue using a singly linked list to ensure that both of these operations run in $O(1)$ time.

[1 mark]

d. Insert the characters `C, O, M, P, L, E, X, I, T, Y` into an initially empty 2-3 tree. Show the state of the tree after inserting `O, L, X` and `Y`. Fill the boxes from left to right first.

[2 marks]

After inserting `O`:

After inserting `L`:

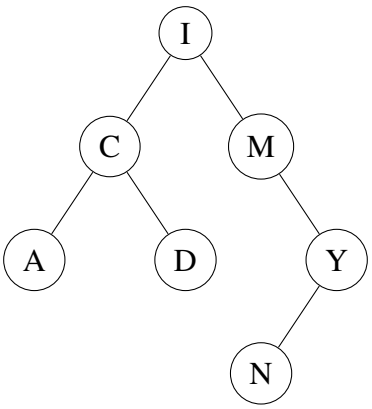
After inserting `X`:

After inserting `Y`:

This page is blank (almost)

e. Which two rotations must be done to balance the following tree? Give the node and the direction of the rotation (left or right).

[1 mark]



f. Give the in-order traversal of this tree after these two rotations.

[1 mark]

This page is blank (almost)

Question 2 (9 marks).

a. How many comparisons does Horspool's algorithm perform to search for the pattern `EAR` in the text `STRINGSEARCH`?

[1 mark]

b. A pair of strings are *anagrams* of each other if they contain the exact same letters, for example `"DESSERTS"` and `"STRESSED"` are anagrams.

Write an algorithm which takes as input two strings, S and T, and determines whether or not they are anagrams. Your algorithm should run in $O(n)$ time where $n := \max\{|S|, |T|\}$. You may assume that all characters in S and T are uppercase alphabetic characters.

[4 marks]

This page is blank (almost)

c. Write an algorithm which takes as input an array of n strings, of length no longer than m characters long, and outputs the largest set of strings which are all anagrams of each other.

For example, given the array ["ABC", "ABBA", "CAB", "BABA", "CBA", "BAC", "BAD"] your algorithm would output "ABC", "CAB", "CBA", "BAC" (not necessarily in that order).

Your algorithm must run in $O(mn \log n)$ time. You may again assume that all characters are uppercase alphabetic characters.

[4 marks]

This page is blank (almost)

Question 3 (6 marks).

a. Which of the following statements are true?

(i) $\log(n) \in \Omega(\log(\log(n)))$
(ii) $\log_3(n) \in \Theta(\log_2(n^2))$
(iii) $100\sqrt{n} + 0.01n^2 + 0.001n^3 \in O(n)$
(iv) $\log n + n^2 \in \Omega(n)$

[2 marks]

b. Solve, using the method of repeated substitutions, the following recurrence relation:

$$T(n) = 2T(n-1) + 7, \quad T(2) = 0$$

[2 marks]

This page is blank (almost)

c. Consider the following sorting algorithm:

```
function ChunkSort(A[0...n-1])
    if n == 2 and A[0] > A[1] then
        swap A[0] and A[1]
    else
        ChunkSort(A[0...⌊n/2⌋-1])
        ChunkSort(A[⌊n/2⌋...n-1])
        ChunkSort(A[⌊n/4⌋...⌊3n/4⌋-1])
        ChunkSort(A[0...⌊n/2⌋-1])
        ChunkSort(A[⌊n/2⌋...n-1])
        ChunkSort(A[⌊n/4⌋...⌊3n/4⌋-1])
```

Write the recurrence relation (including the base case) for the number of comparisons of ChunkSort.

[1 mark]

d. Recall, the master theorem states that if T is a recurrence relation such that $T(n) = aT(\frac{n}{b}) + \Theta(n^c)$ and $T(1) = \text{constant}$ then,

$$T(n) \in \begin{cases} \Theta(n^c) & \text{if } c > \log_b(a) \\ \Theta(n^c \log n) & \text{if } c = \log_b(a) \\ \Theta\left(n^{log_b(a)}\right) & \text{if } c < \log_b(a) \end{cases}.$$

Use the master theorem to find the time complexity of ChunkSort.

[1 mark]

This page is blank (almost)

Question 4 (4 marks).

a. Consider the decision problems PROBLEMA and PROBLEMB. PROBLEMA is known to be NP-Complete.

Your friend has found a polynomial time reduction from PROBLEMB to PROBLEMA. Another friend of yours has found an algorithm to solve any instance of PROBLEMB in polynomial time.

Have your friends proved that P=NP? Explain your answer.

[2 marks]

b. Consider the following algorithm, which computes the value of the nth triangle number.

```
function TRIANGLENUMBER(n)
    result ← 0
    for i in 1...n do
        result ← result + i
    return result
```

Is TRIANGLENUMBER a polynomial time algorithm in terms of the size of its inputs? Explain your answer.

[2 marks]

This page is blank (almost)

Question 5 (10 marks).

a. Consider a hash table with 8 slots and a hash function $h(k) = k \mod 8$, which uses open addressing with a step size of 1. Show the contents of the table after inserting 16, 23, 7, 10 and 12.

[2 marks]

0	1	2	3	4	5	6	7

b. How many comparisons are required to lookup the key 7? You can assume that we can check if a slot is empty without making any comparisons.

[1 mark]

c. How many comparisons are required to lookup the key 15? You can assume that we can check if a slot is empty without making any comparisons.

[1 mark]

d. Consider again a hash table with 8 slots and a hash function $h_1(k) = k \mod 8$. This time, the hash table uses open addressing and double hashing, with the step size determined by the hash function $h_2(k) = k \mod 3 + 1$.

Explain why $h_2(k)$ must contain the $+1$ term?

[1 mark]

This page is blank (almost)

e. Show the state of the hash table after using this double hashing scheme (*i.e.*, the scheme described in part d.) to insert 9, 17, 4 and 11.

[1 mark]

0	1	2	3	4	5	6	7

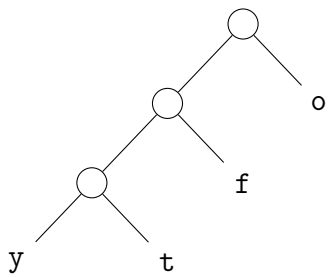
f. Use Huffman's algorithm to construct a huffman tree for the string `"free-coffee"`.

[2 marks]

g. What is the encoded version of this string, using your Huffman tree? Let each left child be 0 and each right child be 1.

[1 mark]

h. Use the following Huffman tree (again, using 0 for left and 1 for right) to decode 0111001000.



[1 mark]

This page is blank (almost)

Question 6 (10 marks).

a. A string of parentheses (*i.e.*, `"("` and `")"`) is *valid* if no pair of parentheses is "closed" before it is "opened", *e.g.*, `"(())()"` is valid while `"())("` is not.

The 5 valid strings of 6 parentheses are:

`"()()()"`, `"(())()"`, `"()(())"`, `"(()())"` and `"((()))"`.

Give 3 (three) examples of valid strings of parentheses containing 8 (eight) parentheses.

[1 mark]

b. Write a recursive formula and base case(s) for the subproblem N_i, where N_i is defined as the number of different valid strings containing exactly i parentheses.

[4 marks]

This page is blank (almost)

c. Using the recursive formula you derived in part b. write pseudocode for an algorithm which computes the number of different valid strings containing n parentheses.

[3 marks]

d. What is the space complexity of your algorithm? Give your answer in Big-Oh notation.

[1 mark]

e. What is the time complexity of your algorithm? Give your answer in Big-Oh notation.

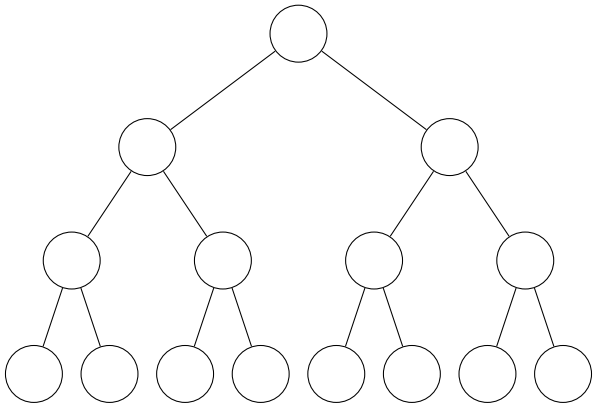
[1 mark]

This page is blank (almost)

Question 7 (13 marks).

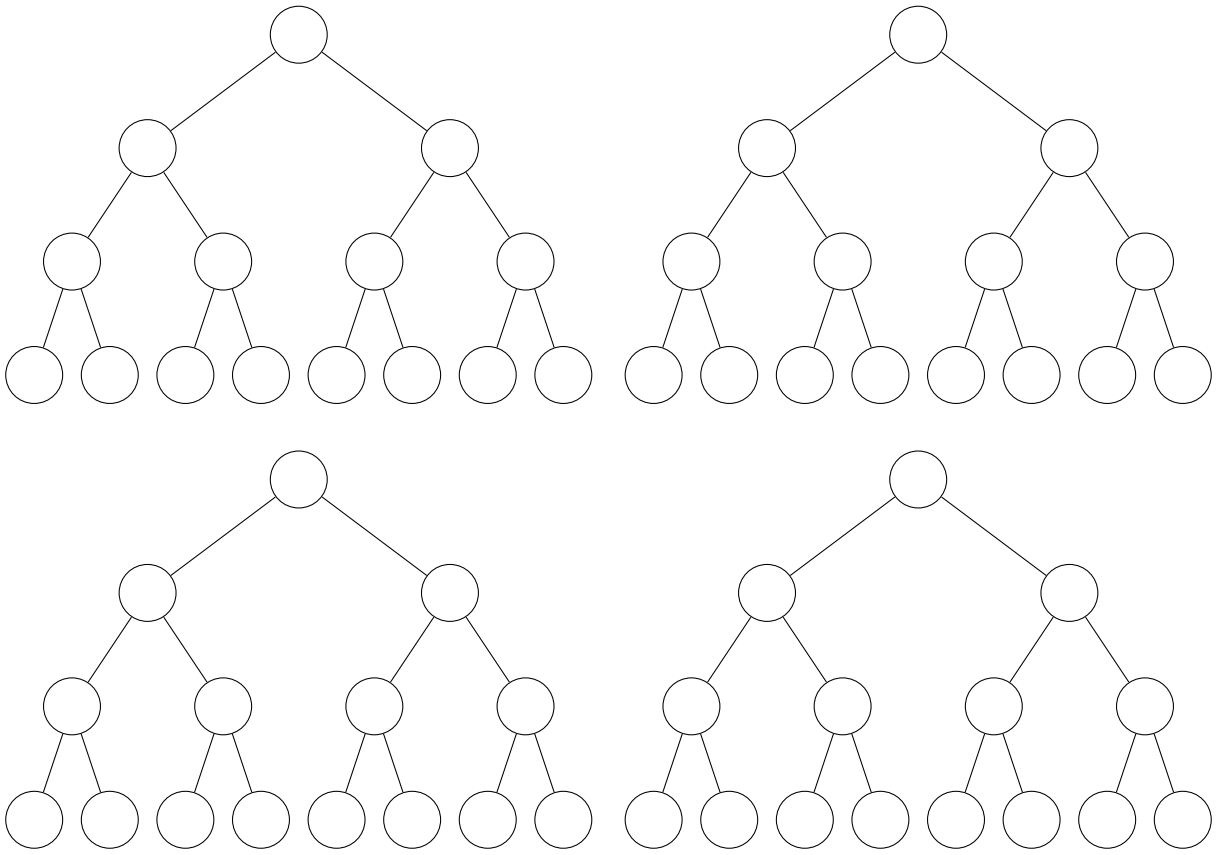
a. Show how we interpret the array $[\texttt{null}, 8, 4, 3, 5, 2, 13, 6, 4]$ as a tree with the indexing scheme used for heaps. Note that some of the nodes will not be used.

[1 mark]



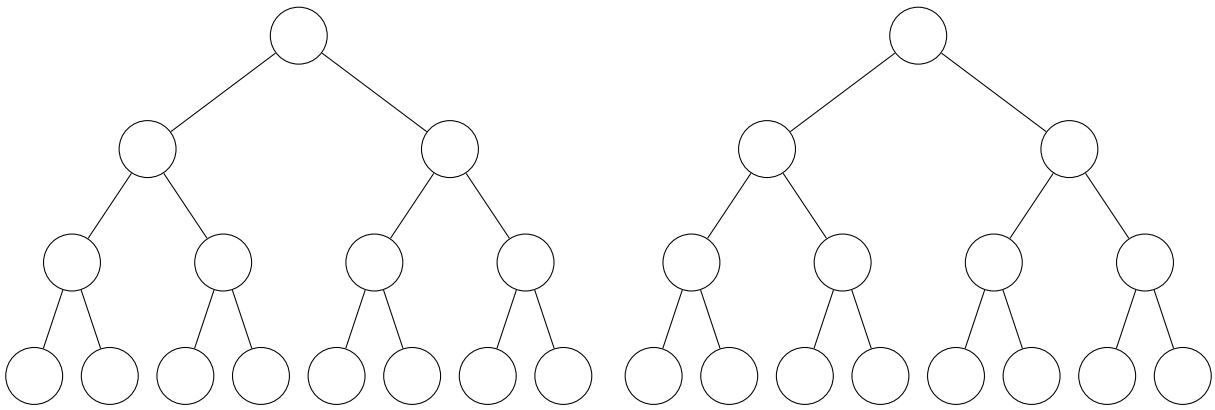
b. Run the bottom-up construct heap algorithm to convert this array into a *max-heap*. Show all steps. Fill the trees from left to right first.

[2 marks]



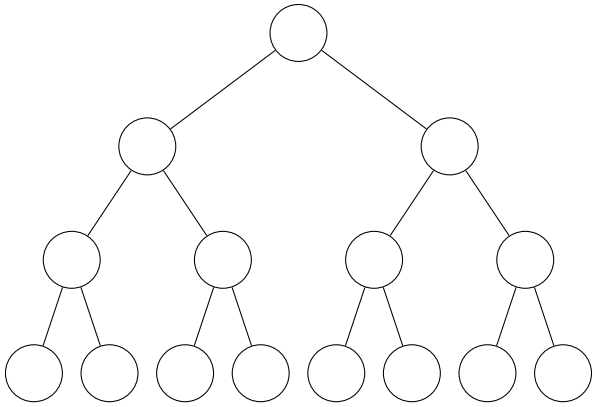
This page is blank (almost)

Part b. continued.



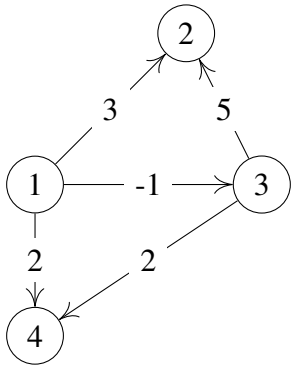
c. Show the state of the heap after one REMOVEMAX operation.

[1 mark]



d. Give the weights matrix for the following graph.

[1 mark]



This page is blank (almost)

e. Run Floyd's algorithm to find the shortest paths between each pair of vertices in the graph from part d. Write the matrices after each step of the algorithm in the following boxes. Fill the boxes from left to right first.

[3 marks]

This page is blank (almost)

f. What is the time complexity of Floyd's algorithm (assume the graph has n vertices).

[1 mark]

g. Using DIJKSTRA(G, u, v) as a helper function which returns the cost of the shortest path between u and v in a graph G, write an algorithm in pseudocode for computing the *all pairs shortest paths* in a graph G with non-negative edge weights. The graph G has n vertices and m edges and is sparse, *i.e.*, $m \in O(n)$.

Your algorithm must have a time complexity better (*i.e.*, smaller) than Floyd's algorithm.

[3 marks]

h. What is the time complexity of your algorithm?

[1 mark]

This page is blank (almost)

Overflow Answers

The boxes here are for emergency use only. If you do need to use extra space for any answers, indicate CLEARLY in your previous answer that you have continued onto this page. Without such an indication, it is possible that this part of your answer will be overlooked.