Official Use Only

Monash University

Semester Two Mid Semester Test 2017

Faculty of Information Technology

EXAM CODES: **FIT2004 (Mid-semester Test T2)**
TITLE OF PAPER: **Algorithms and Data Structures**
TEST DURATION: 45 minutes
READING TIME: 5 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT:

☐ Berwick ✓ Clayton ✓ Malaysia ☐ Off Campus Learning ☐ Open Learning
☐ Caulfield ☐ Gippsland ☐ Peninsula ☐ Enhancement Studies ☐ Sth Africa
☐ Pharmacy ☐ Other (specify)

During an exam, you must not have in your possession, a book, notes, paper, electronic device/s, calculator, pencil case, mobile phone or other material/item which has not been authorised for the exam or specifically permitted as noted below. Any material or item on your desk, chair or person will be deemed to be in your possession. You are reminded that possession of unauthorised materials in an exam is a discipline offence under Monash Statute 4.1.

No examination papers are to be removed from the room.

AUTHORISED MATERIALS

CALCULATORS	**☐ YES**	**✓ NO**
OPEN BOOK	**☐ YES**	**✓ NO**
SPECIFICALLY PERMITTED ITEMS	**☐ YES**	**✓ NO**

STUDENT ID __ __ __ __ __ __ __ __ __

Office use only

This page intentionally left blank, use if needed but it will not be marked.

0

INSTRUCTIONS

- You must answer ALL the questions.
- Answers to each question should be in the space DIRECTLY BELOW the questions and (if required) on the blank page overleaf of each question.
- Script book may be used if ADDITIONAL SPACE is required for answering these questions

General exam technique

Do not throw marks away by **not** attempting all questions. Suppose you get 7/10 on a question for a 20 minutes effort. Spending another half hour on the same question gets *at most* 3 more marks. On the other hand, were you to spend that time on a new question, you might get another 10 marks.

Answer the question that is asked of you. If the question asks for Insertion sort, do not write Quick-sort – this only wastes your time.

Do not write un-necessarily long answers. This wastes your valuable exam time. The question will specifically ask for the information required. Therefore, do not include the information that is not specifically asked for. If asked to justify your answer, provide a clear, logical and concise reasoning.

You do not have to attempt the questions in order. Some questions require less work but may be worth more marks. Carefully read the paper to decide the order in which you should attempt the questions based on the marks associated with each question and whether you know the answer or not.

Best of Luck!

Do not write anything in this table. It is for office use only.

Question	Points	Score
1	10	
2	4	
3	4	
4	4	
Total:	22	

This page intentionally left blank, use if needed but it will not be marked.

0

1. This question is composed of short questions. Write your answers to each of these questions in no more than a few lines.

   (a) (2 marks) What is output-sensitive time complexity? Give an example.

   > The time complexity that also depends on the size of output. E.g., assignment 1 task 2 complexity requirement was $O(klogN + W)$ where W is the output size.

   (b) (2 marks) Show that `lo = hi-1` when while loop terminates.

```
lo = 1
hi = N + 1
while ( lo < hi - 1 )
    mid = floor( (lo+hi)/2 )
    if key >= array[mid]
        lo=mid
    else
        hi=mid
if N > 0 and array[lo] == key
    print(key found at index lo)
else
    print(key not found)
```

   > $lo \geq hi - 1$ when while loop terminates. Let's call it (A).
   > Since $lo < mid < hi$, $lo < hi$ when the while loop terminates. Hence, $lo \leq hi - 1$ when while loop terminates. Let's call it (B).
   > From (A) and (B), we have $lo = hi - 1$.

This page intentionally left blank, use if needed but it will not be marked.

0

(c) (2 marks) What is the worst-case time complexity for searching in a hash table using separate chaining where a sorted array is used for chaining? Give reasoning.

$O(\log N)$. The hash index can have at most $O(N)$ elements and since the elements are in a sorted array, a binary search can be conducted.

(d) (2 marks) Is Selection sort a stable sort? Why or why not?

Select sort is not stable. At each step, the smallest element in the remaining array is found and swapped with the element at the current position. This swapping may change the order of the elements with equal keys.

This page intentionally left blank, use if needed but it will not be marked.

0

(e) (2 marks) Show how the following AVL tree is balanced after 14 is deleted. You need to identify the case (e.g., left-left case) and show how each rotation is done.

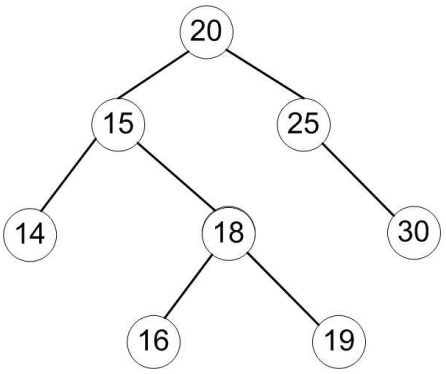
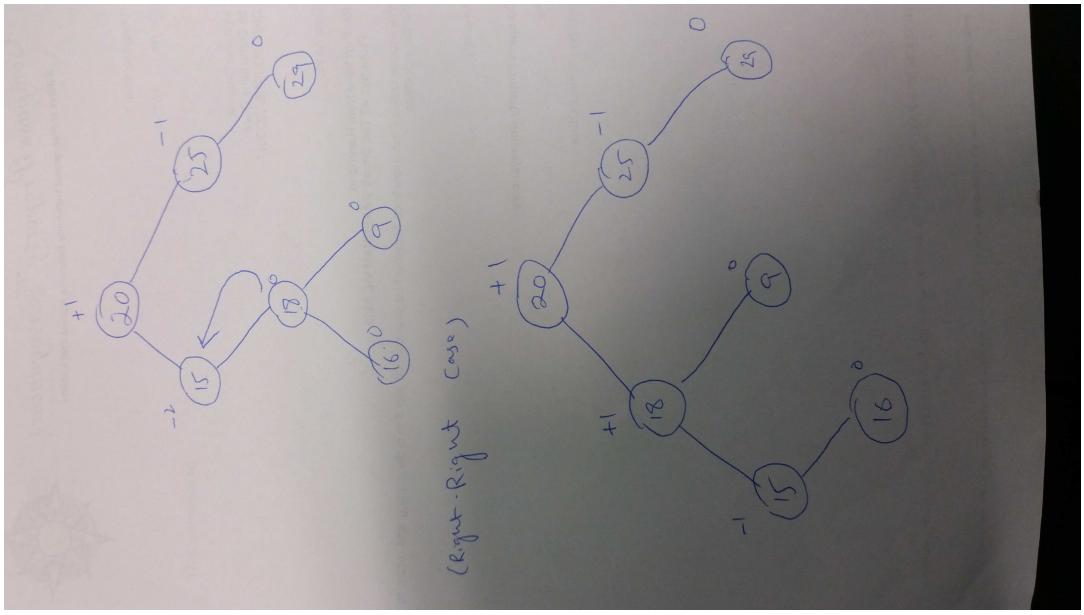


Figure 1: AVL Tree



This page intentionally left blank, use if needed but it will not be marked.

0

2. (4 marks) Consider an array containing N strings where each string has M characters. List the worst-case time complexities to sort the strings for the following alogrithms: quick sort, merge sort, radix sort using stable counting sort in each pass and radix sort using a stable insertion sort at each pass. Give a brief (one line maximum) reasoning of your answer for each algorithm.

Quick sort: $O(MN^2)$ worst-case comparisons is $O(N^2)$ and each comparison takes $O(M)$.

Merge sort: $O(MN \log N)$ comparisons $O(N \log N)$ and each comparison takes $O(M)$.

Radix sort using counting sort: $O(MN)$, each counting sort takes $O(N)$ and we have M calls to counting sort

Radix sort with insertion sort: $O(MN^2)$, each insertion sort takes $O(N^2)$ and we call it M times

4

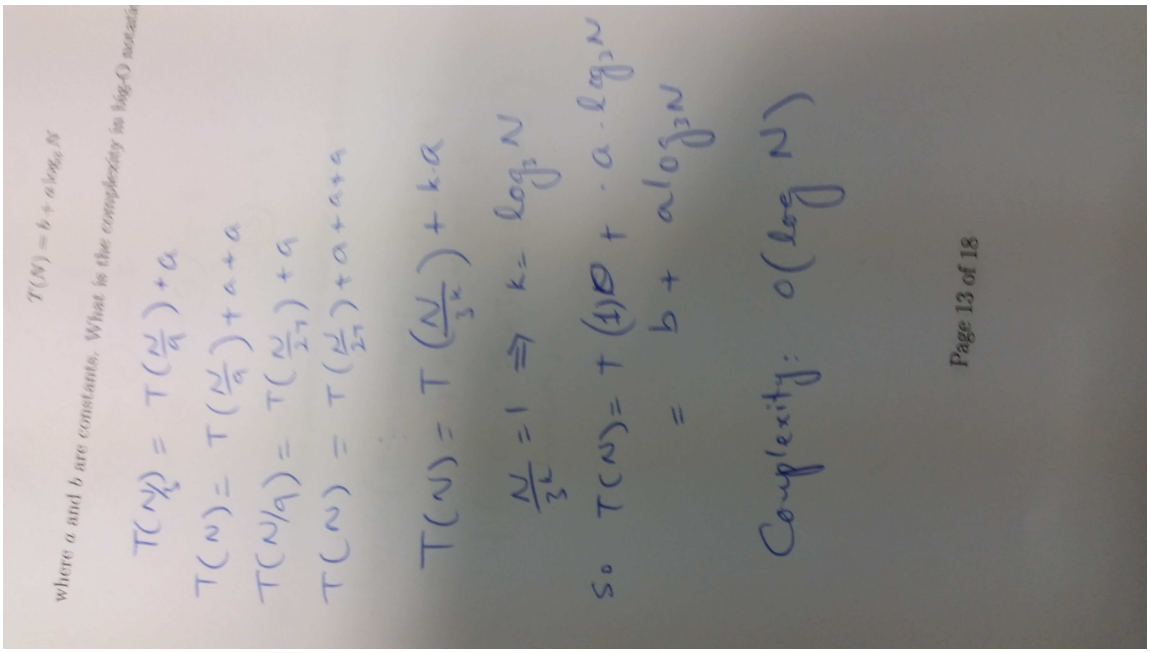
This page intentionally left blank, use if needed but it will not be marked.

0

3. (4 marks) Solve the following **recurrence relationship**:

$$T(N) = \begin{cases} T(N/3) + a, & \text{if } N = 3^k \text{ where } k > 0. \\ b & \text{if } N = 1 \end{cases}$$

where a and b are constants. What is the complexity in big-O notation?

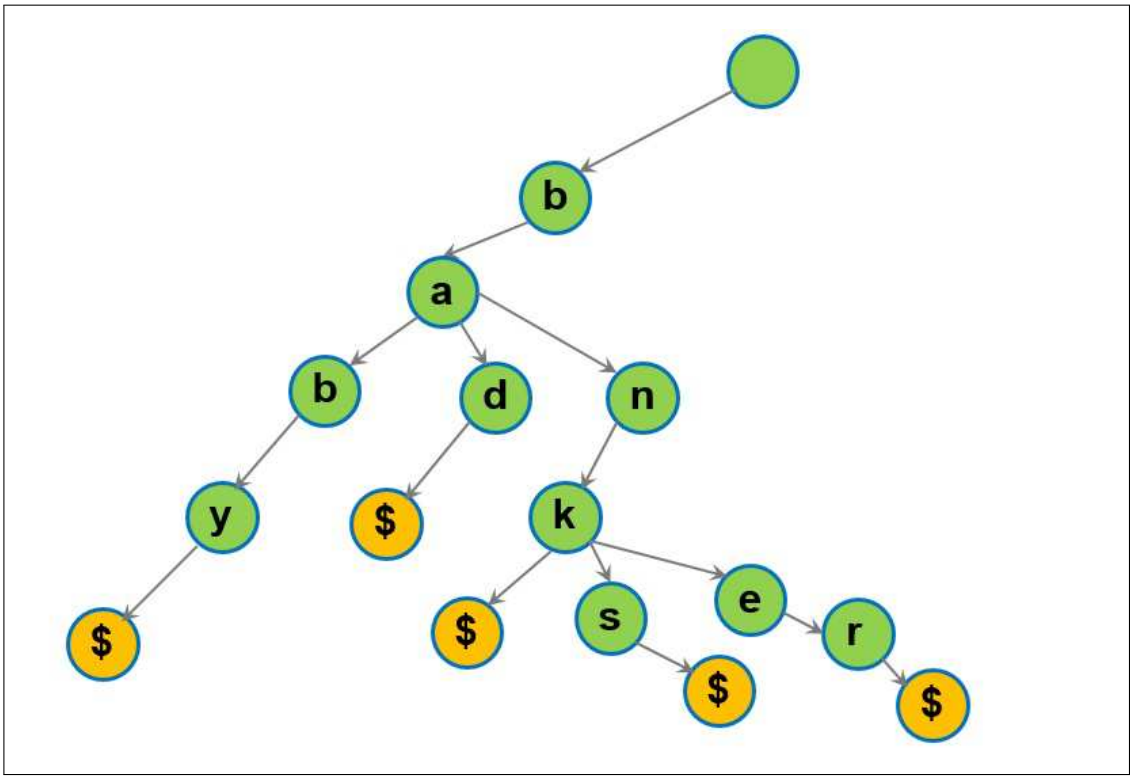


4

This page intentionally left blank, use if needed but it will not be marked.

0

4. (a) (2 marks) Draw a trie containing the following strings: baby, bad, bank, banks, banker.



(b) (2 marks) Write pseudocode for searching a string from a trie.

```
Start from the root node
For each character c in the string (including $)
    If a node containing c exists
        Move to the node
        If c == $
            Return "found"
        Else
            Return "not found"
```


4

This is the end of the test.

0