Durham University

Examination Paper

Examination Session:	Year:	Exam Code:
May/June	2024	COMP1081-WE01

Title: Algorithms and Data Structures

Time Allowed:	2 hours	
Additional Material provided:	None	
Materials Permitted:	None	
Calculators Permitted:	Yes	Models Permitted: Casio fx-83GT range and Casio fx-85GT range
Visiting Students may use dictionaries:	Yes	

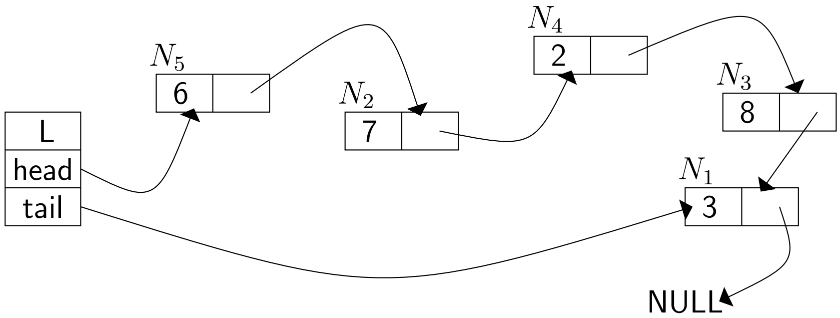
Instructions to Candidates:	Answer ALL questions. Students must use the Computer Science answer booklet.

Section A Fundamental data structures
(Dr Eamonn Bell)

Question 1

(a) i. What features of arrays make them well-suited to the repeated retrieval of data items stored sequentially, assuming that we know the index of the data item to be retrieved? **[2 Marks]**

ii. Consider the singly linked list L illustrated below.



What do the following refer to? **[3 Marks]**

- L.tail
- L.head.next.next.next
- L.head.next.data

iii. Study the following sequence of operations on data structures. A and B are instances of some standard data structures. Their associated operations have also been disguised. At the beginning of the sequence of operations, both A and B are empty. If an operation produces a return value, the value is noted in a comment contained in curly braces ({}) on the same line.

A.spin(3)
A.spin(7)
B.roll(9)
B.roll(A.tilt())
A.handle {returns 3}
B.edge {returns 9}
A.tilt() {returns 3}

For each disguised data structure (A, B) and operation (.spin(), .roll(), .tilt(), .handle, .edge) that appears in the sequence, choose a standard data structure or operation that explains the behaviour in

this question is continued on the next page

the trace above. You may choose from: Stack, Queue, push, pop, enqueue, dequeue, top, front, isEmpty. **[7 Marks]**

(b) i. Complete the hash table that results from applying the hash function,

$$h(k) = (5k + 3) \bmod 9$$

to the keys $35, 37, 17, 25, 49, 26, 50$ and 51, and inserting them into the hash table in that order. Assume that collisions are handled using linear probing.

0	1	2	3	4	5	6	7	8

[3 Marks]

ii. Recall that tombstones are entries with a special meaning that can be used in hash tables to mark the former location of a deleted entry. Explain how the implementation of the following operations must be changed, if at all, when tombstones are used in conjunction with linear probing.

- retrieval from a hash table, given a key known to be in the hash table
- insertion of a given key into a hash table

[3 Marks]

iii. Linear probing is one member of a family of collision resolution policies called open addressing policies. Name a collision resolution policy not in the family of open addressing polices, and one data structure that is frequently used to implement it. **[2 Marks]**

(c) Consider the algorithm $\textsc{Process}$, whose inputs A and B are zero-indexed arrays of characters. For some array L, and some i, j less than the length of the array, let us refer in pseudocode to the subarray containing the items $L[i]$ to $L[j-1]$ using the “slice” notation L[i:j].

this question is continued on the next page

PROCESS(A, B)

```
1: s = A.length
2: t = B.length
3: if s < t then
4:     return 0
5: end if
6: if A[0:t] = B then
7:     return 1 + PROCESS(A[t:s], B)
8: end if
9: return PROCESS(A[1:s], B)
```

i. What is the output of the recursive algorithm PROCESS(A,B) when A = 'CGATATATC' and B = 'ATA'? **[2 Marks]**

ii. Identify the line number or numbers that correspond to the base case in PROCESS. **[1 Marks]**

iii. Describe what PROCESS does in general. **[2 Marks]**

Section B Asymptotic notation and sorting (Prof Thomas Erlebach)

Question 2

(a) Consider the algorithm InsertionSort for sorting n given numbers a_1, $a_2, \ldots, a_n$.

i. Describe briefly how InsertionSort works. You can use English language, pseudocode or python code. **[3 Marks]**

ii. What is the worst-case time complexity of InsertionSort on inputs of size n? On what inputs of size n does the worst case occur? **[2 Marks]**

iii. What is the best-case time complexity of InsertionSort on inputs of size n? On what inputs of size n does the best case occur? **[2 Marks]**

(b) Let f and g be two functions from the natural numbers to the natural numbers. For each of the following statements, state whether it is true or false, and justify your answer.

i. If $f(n)$ is in $o(n)$ and $g(n)$ is in $O(n)$, then $f(n) \cdot g(n)$ is in $o(n^2)$. **[4 Marks]**

ii. If $f(n)$ is in $O(g(n))$ but not in $\Theta(g(n))$, then $f(n)$ must be in $o(g(n))$. **[4 Marks]**

(c) Consider the following recursive algorithm that we call SlowQuickSort for sorting an array containing n numbers $a_1, a_2, \ldots, a_n$ into increasing order (you can assume that no two of the numbers are the same):

If $n \leq 4$, sort the input in constant time using InsertionSort. If $n \geq 5$, proceed as follows:

- Call SlowQuickSort recursively on the first $0.6n$ elements of the input array, and let p be the $0.4n$-smallest value in the sorted order of those elements. (For simplicity, we assume throughout this question that quantities such as $0.6n$ and $0.4n$ are integers.) For example, if $n = 10$, this recursive call will sort the first 6 elements of the input array, and p will be set to the 4-th smallest value among those 6 elements.

this question is continued on the next page

- Rearrange the whole input array in $O(n)$ time so that it contains first the elements smaller than p, then the element p, then the elements larger than p.
- Call SlowQuickSort recursively twice: Once on the part of the array that contains the elements smaller than p, and once on the part of the array that contains the elements larger than p.

Answer the following questions about SlowQuickSort:

i. When SlowQuickSort is called on an input array of size n, what is the largest possible size of a part of the array that may be passed to a recursive call? Justify your answer. **[3 Marks]**

ii. Give a recurrence that captures the time complexity $T(n)$ of the algorithm SlowQuickSort on inputs of size n. It suffices to state the recurrence for the case $n \geq 5$. **[2 Marks]**

iii. Simplify the recurrence from part ii by assuming that each of the three recursive calls is made on a part of the array that has the same worst-case size (which you have identified in part i). **[2 Marks]**

iv. Solve the recurrence from part iii to determine an upper bound on the worst-case time complexity of SlowQuickSort. (This upper bound will be a very loose over-estimate of the worst-case time complexity, due to the simplification of the recurrence in part iii.) In your solution you can use the Master Theorem if you wish, stated below for ease of reference. **[3 Marks]**

Here is a reminder of the statement of the Master Theorem: Suppose a recurrence is of the form $T(n) = aT(n/b) + f(n)$ for constants $a \geq 1$ and $b > 1$.

- **If** $f(n) = O(n^{\log_b(a)-\epsilon})$ for some constant $\epsilon > 0$ **then** $T(n) = \Theta(n^{\log_b(a)})$.
- **If** $f(n) = \Theta(n^{\log_b(a)} \cdot (\log n)^k)$ with $k \geq 0$ **then** $T(n) = \Theta(n^{\log_b(a)} \cdot (\log n)^{k+1})$.
- **If** $f(n) = \Omega(n^{\log_b(a)+\epsilon})$ for some constant $\epsilon > 0$ **and** if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all n large enough **then** $T(n) = \Theta(f(n))$.

End of Section B

Section C Selection and data structures (Dr Anish Jindal)

Question 3

(a) Consider a sequence S of n entries, where each entry's key is an integer in the range $[0, k-1]$. Give the worst-case time complexity of the RADIXSORT algorithm for sorting sequence S. Additionally, for the same sequence S, assuming k is the number of buckets and keys are uniformly distributed among the buckets, what is the time complexity of the standard BUCKETSORT algorithm? **[4 Marks]**

(b) Suppose we have integer values between 1 and 1000 in a binary search tree and the search operation is performed to find the key 437. Which of the following options cannot be the sequence of keys examined? Explain your answer. **[4 Marks]**

- 7, 8, 9, 987, 654, 533, 426, 502, 437
- 523, 466, 123, 201, 455, 453, 202, 302, 437
- 201, 250, 540, 333, 402, 415, 567, 434, 437
- 700, 120, 612, 523, 258, 302, 345, 380, 437

(c) i. What is the maximum height of any AVL tree with 7 nodes? Assume that the height of a tree with a single node is 1. **[3 Marks]**

ii. Discuss the different types of AVL tree rotations and when they occur. **[2 Marks]**

iii. Show the AVL tree that results after each of the following integer keys are inserted, in that order, into an initially empty AVL tree.

9, 27, 50, 15, 2, 21

Clearly show the tree that results after each insertion and indicate any rotations that must be performed. **[4 Marks]**

(d) i. Discuss how a min-heap can be converted into a max-heap. Illustrate the working of this process using the input array A (in the given order) and show the resulting tree at each step.

$$A = [1, 2, 3, 4, 5, 6, 7, 8]$$

[6 Marks]

this question is continued on the next page

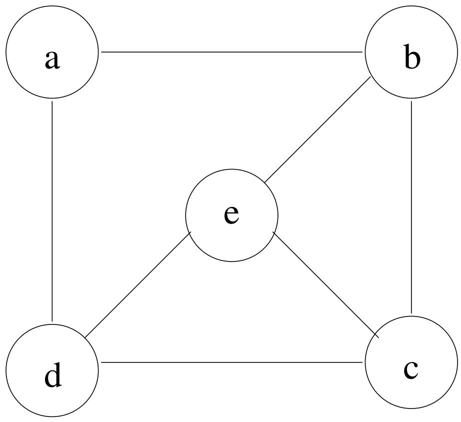
ii. Provide a reason why any correct comparison-based algorithm designed to find the smallest element in an unsorted integer array of length n must make at least $n-1$ comparisons. **[2 Marks]**

End of Section C

Section D Graph algorithms
(Dr Amitabh Trehan)

Question 4

(a) Consider the following graph:

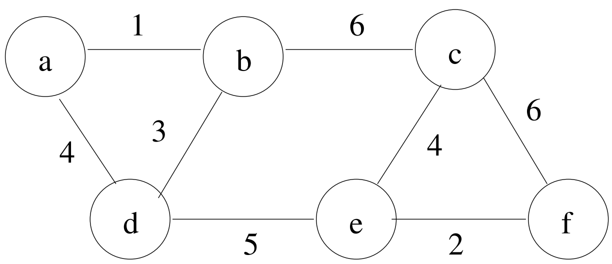


Graph 1

i. State, with justification, if Graph 1 is a **simple graph**. **[2 Marks]**

ii. Graph 1 does not have an Euler cycle. Give the definition of an Euler cycle (or Euler circuit) and give a set of edges which could be added to Graph 1 so that it remains simple and it does have an Euler cycle. **[4 Marks]**

(b) Consider the following graph:



Graph 2

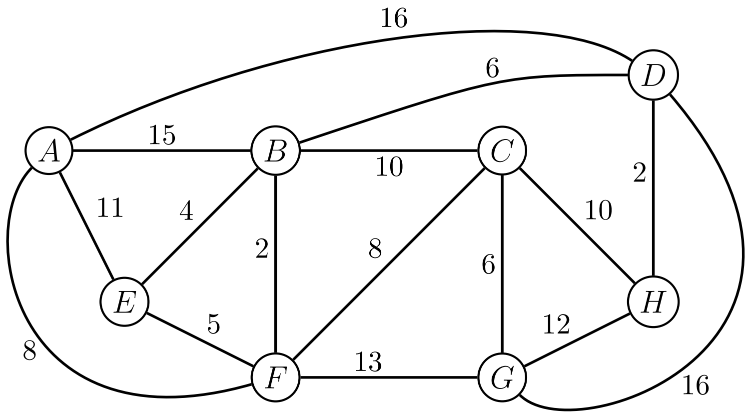
For the following two sets of edges, state with justification whether they are spanning trees and/or minimum spanning trees for Graph 2:

$$E_1 = \{(a,d),(a,b),(b,c),(c,f),(e,f),(e,c)\}$$
$$E_2 = \{(a,b),(e,f),(b,d),(d,e),(c,e)\}$$

[6 Marks]

this question is continued on the next page

(c) Find a minimum spanning tree of the graph below (Graph 3) using Prim's algorithm starting from node F. List the edges of this tree in the order in which they are added to the tree. What is the weight of the constructed tree?



Graph 3

[6 Marks]

(d) Let $G = (V, E)$ be an undirected graph such that every edge $e \in E$ has a positive weight w_e, and let T be a minimum spanning tree of G. Now suppose that we replace every weight w_e by its square w_e^2, thereby creating a new instance of the problem with the same graph G but with the new weights. Is T always a minimum spanning tree in this new instance or not? Justify your answer. **[7 Marks]**

END OF PAPER