UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

Department of Computing

B. Sc. Examination 2018

IS52038A/B
Algorithms & Data Structures

Duration: 2 hours 15 minutes

Date and time:

This paper is in two parts: part A and part B. You should answer ALL questions from part A and TWO questions from part B. Part A carries 40 marks, and each question from part B carries 30 marks. The marks for each part of a question are indicated at the end of the part in [.] brackets.

There are 100 marks available on this paper.

Calculators may be used in this examination; however, calculators which display graphics, text or algebraic equations are not allowed.

Part A

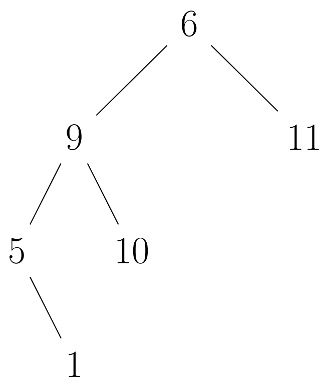
Attempt all questions

Question 1

(a) Define the term *ordered collection*, and give one example of a data structure that can be used as a ordered collection. [3]

(b) Write in pseudocode an algorithm to compute and return the nth term of the series u_n, where $u_0 = -2$, $u_1 = 4$ and $u_{k+2} = u_{k+1} + 2 \times u_k - 2$. [5]

(c) Consider the following binary tree:



i. list the contents of the nodes in this tree visited in *breadth-first* order; [2]

ii. give the name of the order for visiting nodes in the order [6, 9, 5, 1, 10, 11]; [2]

iii. give pseudocode for visiting the nodes of this tree with an *in-order* traversal. [4]

(d) The function F is defined as follows:

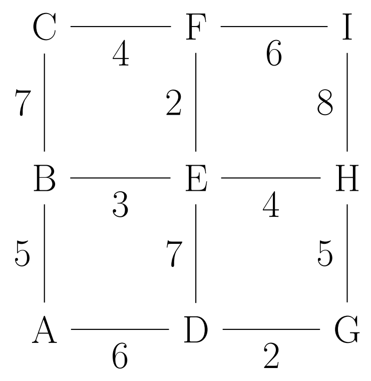
```
Require: L :: linked list
Require: n :: non-negative integer
  function F(L,n)
      if n = 0 then
          return NIL
      else
          return CONS(FIRST(L), F(REST(L), n−1))
      end if
  end function
```

i. What is the return value of F([6, 3, 17], 0)? [1]

ii. What is the return value of F([6, 3, 17], 1)? Explain your reasoning. [3]

iii. Describe what F does for general arguments L and n. (You may wish to consider cases n ≤ LENGTH(L) and n > LENGTH(L) separately). [4]

(e) For the following graph, execute Dijkstra's algorithm to find the shortest path between A and I and its path cost, showing your working. [9]



(f) A binary min-heap is represented implicitly in the following array:

[2 4 8 5 6 17 15 12]

i. draw the tree representation of this binary min-heap; [3]

ii. state, showing your working, the contents of the array after EXTRACT-MIN! has been called on this binary min-heap. [4]

Part B

Attempt two questions

Question 2

(a) A hash table H consists of 11 buckets, and stores positive integers using the hash function $h(x) = 2x + 5$ and standard linear probing for collision resolution.

i. What would be a suitable reduction function for this situation? [2]

ii. What are the contents of the bucket table after the data

[17 51 55 42 82 24 89 92 59]

are inserted (one by one, in the order given)? [5]

iii. How should efficient hash tables be modified before inserting elements which would exceed the capacity of their bucket table? In your answer, be as specific as you can about the changes to the data structure and any associated functions. [5]

iv. Explain the benefits of Robin Hood linear probing over standard linear probing? [5]

(b) Describe in detail an algorithm that identifies whether or not a given linked list is cyclic. Include with your description a statement of the algorithm's time and additional space complexities in terms of the number of distinct list nodes N and the cycle length (if any) L. [6]

(c) How might you adapt your cycle-finding algorithm from part (b) to determine whether a directed graph contains cycles or not? For your adaptation, identify the worst case situation for time complexity, and suggest how your adaptation scales in this worst case with respect to the number of vertices V in the graph. [7]

Question 3

(a) The following function is intended to merge two linked lists sorted in ascending order and return a single sorted list with the contents of the two input lists.

```
Require: L1, L2 :: sorted linked lists
  function MERGE(L1, L2)
      if NULL?(L1) then
          return A
      else if NULL?(L2) then
          return B
      else if FIRST(L1) $\leq$ FIRST(L2) then
          return CONS(C, D)
      else
          return CONS(E, F)
      end if
  end function
```

i. What pseudocode expressions should replace **A**, **B**, **C**, **D**, **E** and **F** for this function to operate as desired? [6]

ii. Give, as precisely as possible, the worst-case time complexity of MERGE in terms of the total number N of elements in the return value. Explain your answer. [5]

(b) Mergesort operates by recursively sorting two half-sized sublists of its input, and then combining the two sorted sublists using MERGE from part (a).

i. State the base case for mergesort, being as specific as possible on the inputs treated as the base case and the corresponding return value. [2]

ii. Write down the recurrence relation for the time taken to do merge sort $T(N)$ on a list with N elements; you will need to incorporate your answer to part a.(ii). [3]

iii. Using the master theorem, or otherwise, solve the recurrence relation in part b.(ii) to express the asymptotic time complexity of mergesort. [3]

(c) Describe in detail how to sort a collection of records corresponding to living people into ascending order of birth year, in time proportional to the size of the collection. [6]

(d) "A comparison sort cannot have worst-case performance better than $\Theta(N \log N)$". Justify this statment, and explain how your answer to part (c) is not a counterexample. [5]

Question 4

The following is a set of equal-length strings S:

{ “`care`”, “`cart`”, “`scam`”, “`scar`” }

(a) Draw the trie corresponding to the set of strings S. [3]

(b) For the trie in part (a), explain in detail the operations that a process will perform to determine that:

i. the string “`core`” is not a member of the set; [3]

ii. the string “`scar`” is a member of the set; [3]

iii. the string “`car`” is not a member of the set. [3]

(c) State and justify the time complexity of using a trie to establish string set membership, in terms of the length m of the string and the number of elements N in the set. [3]

(d) Explain how you could use the trie in part (a) in a modification of naïve string matching to search for the first index at which *any* of the elements of S is present in a string. Include with your explanation a statement about the worst-case time complexity of your modified algorithm in terms of: n, the size of the text; N, the number of elements in the set; and m, the size of each element in the set. [9]

(e) How could you speed up your answer to part (d) with the use of rolling hash functions? [6]