Examination Paper

Examination Session:	Year:	Exam Code:
May/June	2021	COMP1081-WE01

Title: Algorithms and Data Structures

Release Date/Time	9.30am 02/06/2021
Deadline Date/Time	9.30am 03/06/2021
Format of Exam	Take home exam
Duration:	2 hours
Word/Page Limit:	None
Additional Material provided:	None
Expected form of Submission	A SINGLE PDF file (handwritten or typed) submitted onto Gradescope. Upload your submission with a file name comprising of your student ID and the Exam Code e.g 00001234532 COMP1081-WE01.
Submission method	Gradescope

Instructions to Candidates:	Answer ALL questions.

Section A Part I
(Prof Matthew Johnson)

Question 1

(a) The input to the algorithm A below is a string x. Describe the output string y.

```
A(x)
  stack S
  for character in x do
    S.push(character)
  end for
  string y
  y = ""
  while S.isEmpty() = False do
    y = y + S.pop()
  end while
  return y
```

[3 Marks]

(b) Let L be a singly linked list in which each piece of data is an integer. Suppose that L has been sorted so that the integer at the head of the list is the smallest in L and every other integer in L is equal to or greater than the one before it. Let a be an integer. Write pseudocode to add a to L such that L remains sorted. It is possible that L is empty initially. You do not need to include code to amend the value of L.size. **[9 Marks]**

this question is continued on the next page

(c) Consider the function L(n) below whose input is a positive integer n. What is printed when L(n) is run? Justify your answer. **[4 Marks]**

```
L(n)
  queue Q
  Q.enqueue("1")
  for i = 1 to n do
    x = Q.dequeue()
    print(x)
    y = x + "0"
    z = x + "1"
    Q.enqueue(y)
    Q.enqueue(z)
  end for
```

(d) Suppose that you are given an array $A = [a_1, a_2, \ldots, a_n]$ of n integers. A subarray of A is a sequence of integers that appear consecutively in A and the weight of the subarray is the sum of the integers it contains. For example if A is $[1, -3, 4, -6, 2, 3]$ then $[1, -3, 4]$ and $[-6, 2, 3]$ are subarrays with weights of 2 and -1 respectively.

Design an algorithm that decides whether an array A has a subarray of weight 0. The algorithm should read each integer in the array only once.

Write pseudocode for your algorithm. **[9 Marks]**

Section B Part II
(Dr Konrad Dabrowski)

Question 2

(a) In the QuickSort algorithm, suppose you have a partition function that, given n numbers:

- always chooses the median (i.e. the $\lceil n/2 \rceil$-th smallest element) as the pivot,
- keeps the elements less than the pivot in the same order with respect to each other and
- keeps the elements greater than the pivot in the same order with respect to each other.

i. Apply the QuickSort algorithm from lectures, but with this partition function instead, to sort the following array:

$$90, 19, 88, 75, 61, 98, 64$$

Show the state after every time the partition function is called.

[5 Marks]

ii. Suppose the partition function above runs in $\Theta(n^2)$ time. Give tight upper and lower bounds for the runtime of QuickSort with this partition function (you may assume that n is one less than a power of 2 and that no two elements of the input array are equal). Justify your answer. **[5 Marks]**

(b) Consider functions $f, g : \mathbb{N} \to \mathbb{R}^+$. Is it true that $4^{f(n)} = O(3^{g(n)})$ implies that $f(n) = O(g(n))$? Prove or disprove your claim. **[5 Marks]**

this question is continued on the next page

(c) The function `3mod(x)` takes a positive integer x as input and returns 0, 1 or -1 if x is congruent to 0, 1 or -1 modulo 3, respectively. The input for algorithm C is an array A of positive integers $A[1], ..., A[n]$.

```
C(A)
  n = length(A)
  swapped = True
  while swapped = True do
    swapped = False
    for i = 1 to n-1 do
      if (3mod(A[i-1])*A[i-1]) > (3mod(A[i])*A[i]) then
        swap(A[i-1], A[i])
        swapped = True
      end if
    end for
  end while
  return A
```

Describe what this algorithm does. Find tight upper and lower bounds on the asymptotic runtime of this algorithm (i.e. find the best possible $f(n)$ and $g(n)$ such that the algorithm always uses $O(f(n))$ comparisons and $\Omega(g(n))$ comparisons). Justify your answers. **[10 Marks]**

Section C Part III
(Prof Andrei Krokhin)

Question 3

(a) Manually run the RadixSort algorithm with base-2 representation on the following array (where all numbers are given in base-2):

$$11001_2, 11101_2, 101_2, 10000_2, 1011_2, 11110_2.$$

[5 Marks]

(b) For $n \geq 3$, let T_n denote the AVL tree obtained by inserting the numbers $1, 2, 3, \ldots, n$, in this order, into an empty AVL tree.

A rooted binary tree is called perfect if every non-leaf node in it has 2 children and all the leaves are on the same level.

i. Let a be a node in T_n such that a is the left child of its parent. Prove, by induction on n or otherwise, that the subtree of T_n rooted at a is perfect. **[6 Marks]**

ii. Prove that the number stored in the root of T_n is a power of 2, i.e. it is of the form 2^k for some integer k. **[3 Marks]**

iii. Prove that the number stored in the root of T_n is equal to $2^{\lfloor \log_2 \frac{n}{3} \rfloor + 1}$. **[6 Marks]**

(c) Consider the task of finding (indices of) all the smallest elements an unsorted array $[a_1, \ldots, a_n]$ containing not necessarily distinct integers. Use the comparison-based model, which, when comparing two elements a_i and a_j, returns one of the three possible answers: $<$, $=$, $>$. When analysing an algorithm in this model, only comparisons are counted and everything else is ignored. For the task described above and in the model described above, find a lower bound and an algorithm matching this bound. Justify your answers.

Partial credit will be given for non-matching lower bound and algorithm. **[5 Marks]**

End of Section C **continued**

Section D Part IV
(Dr George Mertzios)

Question 4

(a) Tamaru University participates in a large consortium of n universities across their country in a joint project on multimedia. All these universities are connected via a pre-existing computer network, which can be modeled by a graph G with n vertices. The consortium decided to install a central file server at one of the universities such that they can all share the same multimedia data. The transmission time from the server to any university is proportional to the number of edges in a shortest path of G connecting them, so it is desirable to choose a central vertex in the "middle" of the graph G to host the server.

Given a graph G, the remoteness of a vertex v is the distance from v to the vertex u that is farthest from v in G. That is, the shortest path from v to u is as long as possible. A vertex of G with the smallest remoteness is called a middle vertex of G.

i. Design an efficient algorithm that, given a graph G with n vertices and m edges, computes a middle vertex of G. Prove the correctness of your algorithm. What is the asymptotic complexity of your algorithm? You do not need to write any pseudocode and can use algorithms introduced in lectures. **[5 Marks]**

ii. Assuming that G is a tree, design an algorithm that computes a middle vertex in $O(n+m)$ time. Prove its correctness and its running time. **[10 Marks]**

(b) Let T be an MST of a weighted undirected graph G. Let (u, v) be an edge of T, and let T_1 and T_2 be the two trees obtained if (u, v) is removed from T. Let G_1 be the subgraph of G induced be the vertices of T_1.

Show that T_1 is an MST for G_1, and that (u, v) is a least weight edge from a vertex of T_1 to a vertex of T_2. **[4 Marks]**

this question is continued on the next page

(c) Consider the following algorithm for finding an MST of a graph G.

- Divide the vertices of G into two roughly equal sets V_1 and V_2, i.e. the sizes of V_1 and V_2 differ at most by 1.
- Let G_1 be the subgraph of G containing the vertices of V_1 and all the edges of G among those vertices.
- Let G_2 be the subgraph of G containing the vertices of V_2 and all the edges of G among those vertices.
- Find MSTs of G_1 and G_2. Let these two trees be T_1 and T_2.
- Find a least-weight edge (u, v) that connects a vertex in V_1 and a vertex in V_2.
- The output of the algorithm is the tree formed from T_1, T_2 and (u, v).

Is the algorithm correct? Justify your answer. **[6 Marks]**

END OF PAPER