COMP 26120

One and a half hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Algorithms and Imperative Programming

Date: Tuesday 15th January 2019

Time: 09:45 - 11:15

Please answer BOTH Questions.

Use a SEPARATE answerbook for each QUESTION

This is a CLOSED book examination

The use of electronic calculators is permitted provided they are not programmable and do not store text

[PTO]

1. Complexity Analysis and Recurrence.

   a) **(Asymptotic Analysis)** Given $f_1(n) \in O(g_1(n))$ and $f_2(n) \in O(g_2(n))$, for each of the following statements use the O-notation to either show that the statement is valid or otherwise give a counterexample.

   (i) $f_1(n).f_2(n) \in O(g_1(n).g_2(n))$
   (ii) $f_1(n)^{f_2(n)} \in O(g_1(n)^{g_2(n)})$

   No marks will be given for simply stating whether the statement is true or false without an explanation. (4 marks)

   b) **(Sums)** You are given a program with two loops. The first loop iterates i from 1 to n and performs $log\ i$ operations per step. The second loop iterates i from 1 to n-1 and performs $\frac{i}{i\cdot(i+1)}$ iterations per step. Using either proof by induction or integrals solve the following two sums to find the complexities of each loop.

   (i) $\sum_{i=1}^{n} log\ i$
   (ii) $\sum_{i=1}^{n-1} \frac{i}{i\cdot(i+1)}$

   (4 marks)

   c) **(Recurrence)** Solve the following recurrence by changing variables and then applying the master method.

$$T(n) = \begin{cases} 2T(n^{0.25}) + 1, & \text{if } n > 1 \\ 1, & \text{if } n = 1. \end{cases}$$

   (6 marks)

   d) **(Complexity Analysis)** Design a binary-search algorithm to find an arbitrary number k in a **sorted** list of n integers, which breaks the instance into two parts: one with 1/3 of the elements and another one with 2/3. You can assume that the list is sorted and should present your algorithm using pseudocode. Compute the complexity and make a comparative analysis with the traditional binary-search algorithm (which breaks the instance in half). (6 marks)

2. Trees

a) Describe the main difference in structure of binary search trees and AVL trees. Based on this, give the complexity of the operation `FindKey(k)` in both cases as a function of the number of keys n that a tree currently holds. Discuss the best and the worst case scenarios. (6 marks)

b) Describe a linked data structure that is suitable for storing trees. In your answer make a distinction between the cases of general trees and binary trees. (4 marks)

c) Consider the following sequence of keys:

```
25,38,11,26,31,46,57
```

We wish to insert these keys in the given order (from left to right) into an AVL tree. Show the process of insertion, step by step. (10 marks)

END OF EXAMINATION