Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Algorithms and Imperative Programming

Date: Wednesday 20th January 2016

Time: 14:00 - 16:00

Please answer any TWO Questions from the THREE Questions provided

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. Algorithm design.

   For each of the computational tasks (a), (b) and (c) below,

   (i) describe an algorithm for the task. Your description may be a program in a standard language, in pseudocode, or a clear and precise step-by-step description. You should explain your algorithm and why it works. Marks are awarded for a correct algorithm. Some marks are also awarded for efficiency: the more efficient your algorithm is, the more marks it will be awarded.

   (ii) give the worst-case time complexity of your algorithm in terms of the size of the input and the number of operations required. Use the O-notation to express the complexity and explain how you calculated your answer.

   a) Remove duplicates from a list of integers, i.e. given a list of integers, return as output a list of the same integers, but where each integer occurs only once in the output (the order of elements in the output list is not important). (6 marks)

   b) The intersection of lists: Given two lists of integers, compute a list of integers which consists of those integers which appear in both of the two lists. The order of the resulting list does not matter and, if numbers appear several times in the lists, the result need not reflect this multiplicity - though it may. Thus, given lists $[2,5,3,8,2,4,7]$ and $[6,7,2,4,9,1]$, one possible intersection list is $[2,4,7]$. (7 marks)

   c) A *majority element* in a list of integers of length N, is an element that occurs strictly more than $N/2$ times in the list. Determine whether or not any list of integers has a majority element. (7 marks)

2. Sorting

a) Suppose that you are given a procedure `merge`, which, given two lists of integers, `left` and `right`, each sorted in ascending order, produces the result of ordering their concatenation in ascending order, thus:

```
merge([1, 4, 7], [2, 7, 9, 10])  =  [1, 2, 4, 7, 7, 9, 10].
```

The sorting algorithm `merge-sort` employs `merge` to sort lists of integers in ascending order. Give pseudo-code for `merge-sort`. (You need not give pseudo-code for `merge`.) (4 marks)

b) Assuming that `merge` runs in time $O(n)$ in the combined length of its arguments, and that comparison of two integers runs in time $O(1)$, show informally that `merge-sort` runs in time $O(n \log n)$. You may use a diagram. (6 marks)

c) The *height of a tree* is defined to be the number of edges on a longest path from the root node to a leaf node. What is the height of a full binary tree with M leaves? (You need not show any working.) (2 marks)

d) Show that for all $n \geq 1$, $n! > (n/2)^{n/2}$. (2 marks)

e) Explain why a sorting algorithm based on comparisons has time complexity $\Omega(n \log n)$. (6 marks)

3. Rates of growth and complexity

   a) Let $f : \mathbb{N} \to \mathbb{N}$ be a function. Define *exactly* what is meant by the expression $O(f)$. (Hint: $O(f)$ is a set of functions.) (4 marks)

   b) If `P` is some algorithm, it is sometimes said that “the asymptotic running time of `P` is $O(f)$.” What does this mean? (4 marks)

   c) Consider the following algorithm for raising a positive integer a to a non-negative integral power b:

```
Pow1(int a, int b)
    ans:= 1;
    for i from 1 to b
       ans:= ans*a
    return ans
```

   Using the O-notation, give the asymptotic running time of `Pow1` in terms of the size of a and b, assuming multiplication of integers is performed in constant time. Warning: the size of (= number of bits in) an integer c is approximately $\log_2 c$. In other words, if c has size n, then c may be as large as $2^n - 1$. (6 marks)

   d) Using pseudocode, give a more efficient algorithm, `Pow2`, for raising a positive integer a to a non-negative integral power b, and give its asymptotic running time. Again, assume that multiplication of integers is performed in constant time. Hint: use the trick of ‘repeated squaring’.

   (6 marks)

END OF EXAMINATION