Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Algorithms and Imperative Programming (CSE)

Date: Friday 1st June 2012
Time: 14:00 - 16:00

Please answer THREE Questions from the FOUR Questions provided

For full marks your answers should be concise as well as accurate.
Marks will be awarded for reasoning and method as well as being correct.

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 following computational tasks

(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. Marks are awarded for a clear and correct algorithm. Some marks will also be 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. Explain how you calculated your answer.

a) Remove duplicates from a list of integers, i.e. given a list of integers, return a list of integers whose elements occur only once in the result and are exactly the elements of the given list (the order of elements in the result list is not important). For example, for the list [2,3,6,1,2,3,7,1], one possible result would be [2,3,6,1,7].

   (6 marks)

b) Given a list of integers as input, determine whether or not two integers (not necessarily distinct) in the list have a sum \( k \). For example, for list [2,10,5,3,7,4,8] and \( k = 17 \), there is a pair, 10 and 7 such that \( 10 + 7 = 17 \).

   (7 marks)

c) Counting the number of times each element occurs in a list, returning a list of pairs, each pair consisting of an element of the list and the number of its occurrences in the list. Each element of the list should appear in the result once but their order is not important. Thus, for the list [2,1,5,2,1,7,3,5,1], one possible result is the list [(5,2),(2,2),(7,1),(3,1),(1,3)], saying that 5 occurs twice in the original list, etc.

   (7 marks)
2. a) The following four expressions give the number of basic operations that four different programs, A, B, C, and D, perform in terms of an input size $n$.

(A) $n + n + n + n^3$
(B) $4 + n \times n + 2n$
(C) $3n + 6 + n^{n/2}$
(D) $\sqrt{n} + \log(n) \times n + 10n$

i) Put each expression in Big O form and simplify as much as possible. (4 marks)

ii) Which expression has quadratic complexity? (1 mark)

iii) Which expressions are bounded by a polynomial? (State ALL that apply) (1 mark)

iv) Which expressions would generally be considered intractable? (State ALL that apply) (1 mark)

v) Which expressions would not tend toward a (perfectly) straight line on a log-log plot for large $n$ (plotting log number of operations against log $n$)? (State ALL that apply) (1 mark)

b) Given that $f(n)$ is $O(n)$ and $g(n)$ is $O(\log n)$, express the worst-case time complexity of the following code fragments (A), (B) and (C) using the big-O notation in terms of two inputs $n$ and $m$. (6 marks)

——— (A) ———
$c = m;$
while($c > 0$)
    $g(n)$
    $c--$
end while

——— (B) ———
if($j < n$)
    for($i = 0; i < n/2; i++$)
        $f(n)$;
else
    $g(m)$
end if
if(j < m)
    for(i=0;i<n/2;i++)
        f(n);
else
    g(n)
end if
3. Consider the following heap, where the boxed node containing the element 16 is the last element of the heap.

![Heap Diagram]

a) Assuming a total order relation on the keys of a dataset is given (e.g. by a comparator), define the heap property of a complete binary tree.

(3 marks)

b) Show the steps of removing the minimal element from the heap given above (picture the heap at each step of this process).

(5 marks)

c) Give the asymptotic time complexity of the following heap operations: `insert(a)`, `remove_min()`, and `replace(a,b)`. Your answer should be given in terms of Big O notation.

(6 marks)

d) Give an efficient algorithm (using a pseudo-code or a clear step-by-step explanation) for finding all the keys in a heap that are smaller or equal to a given key $k$. The value of $k$ is not necessarily equal to any element in the heap. In your answer do not use `remove_min()` operation, i.e. you should leave the initial heap structure unchanged. What is the asymptotic time complexity of the proposed algorithm? Explain your answer.

(6 marks)
4. Graphs and graph algorithms.

a) Consider the following two representations of finite directed graphs

- Adjacency lists
- Adjacency matrices

i) Explain clearly what these representations are.

Show how the following example of a directed graph is presented using these two representations. (2 marks)

ii) For each of the following operations on finite directed graphs, explain clearly how it may be implemented using adjacency lists and using adjacency matrices. Give the worst-case time complexities of your algorithms, in terms of the number of nodes \( N \) and the number of edges \( E \), for each representation and explain how you calculate this. (6 marks)

- Finding a node with maximum in-degree in the graph (the in-degree of a node \( n \) is the number of edges in the graph whose target is \( n \)).
- Finding whether the graph contains cycles of length 2.

b) Explain what is meant by a Depth-First Search (DFS) of a finite directed graph. (2 marks)

Give an explicit algorithm for performing DFS of finite directed graphs which numbers the nodes in the order that they are first encountered. You may either present a program or express the algorithm in pseudocode. (4 marks)

c) A topological sorting of the nodes of a finite directed graph is a list of the nodes, each occurring exactly once in the list, such that, if there is an edge from node \( s \) to node \( t \), then \( s \) is before \( t \) in the list.

Explain clearly why a directed graph with a topological sorting must be acyclic. (2 marks)

Given an acyclic finite directed graph, explain how, using a DFS or otherwise, a topological sorting may be constructed. (4 marks)