COMP 26120

One and a half hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Algorithms and Imperative Programming

Date: Thursday 26th January 2017

Time: 09:45 - 11:15

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 and analysis

   Consider the following three algorithmic tasks, a(i), a(ii) and (b). In each case, you are asked to 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 well-presented correct algorithm. Some marks are also awarded for efficiency: the more efficient your algorithm is, the more marks it will be awarded.

   a) Consider lists of non-negative integers of length N of two types, (i) and (ii) below. We wish to sort them into ascending order. Describe two sorting algorithms, one for each type of list. The algorithms you describe should be appropriate for the type of list in the sense that they have a good worst-case time complexity on these types of lists. Give this worst-case time complexity and show how you calculate it.

      Warning: No marks will be given for naming and describing a standard sorting algorithm, even one which is efficient on general lists. You should either devise your own algorithm or, if using a standard algorithm, describe it and explain clearly why it is appropriate for the type of list involved.

      i. Lists in which there is a small fixed number, n, of distinct integers with many repetitions, i.e. n is much smaller than the length of the list, N. (6 marks)

      ii. Lists which consist of a small fixed number, n, of segments connected end-to-end, each segment being already in ascending order.

      (6 marks)

   b) Consider the problem of choosing k items from a list of N distinct integers *at random and without duplication* (i.e. an item must not be chosen more than once). Assume you are given a function *random(m)* which chooses an integer at random between 1 and m. Describe an algorithm for this task and give its worst-case time complexity, explaining how you derive this complexity. (8 marks)

2. Discrete logarithms

a) Let p be a prime number and g a number $1 \leq g < p$. What does it mean to say that g is a *primitive root with respect to p*? (2 marks)

b) Let p be a prime number and g a primitive root with respect to p. Let y be a number $(1 \leq y < p)$. What is the *discrete logarithm of y with base g modulo p*? (2 marks)

c) Give an algorithm in pseudocode for computing the discrete logarithm of some number y with base g modulo p, and show that it runs in exponential time in the number of binary digits in p. (6 marks)

d) In the El Gamal public-key cryptosystem, a recipient's private key is a triple (p,g,x) where p is a prime, g a primitive root with respect to p, and x is any number $(1 \leq x < p)$. The corresponding public key is the triple (p,g,y), where $y = g^x \bmod p$. Explain, using pseudocode, how the recipient can compute a public key from a private key in time bounded by a polynomial function of the number of binary digits in p. (8 marks)

e) Why can the private key not, in practice, be recovered from the public key when p is large? (2 marks)

3. Dictionary look-up

a) Consider the following algorithm for determining whether a given integer X is in a (1-dimensional) array of integers L. The elements of L are in ascending order.

```
lookup(X, L)
    for i = 0 to length(L) − 1
        if L(i) = X
            return Yes
    return No
```

What is the time complexity of this algorithm as a function of the length N of L?
(2 marks)

b) Using pseudocode, give an alternative algorithm for the same problem, but with running time bounded by $O(\log N)$. Explain why your algorithm has the required time complexity. (6 marks)

c) Let Σ be a (finite) alphabet of symbols, ordered in some fixed way. Denoting by Σ^+ the set of non-empty strings (finite sequences) of characters from Σ. Say what is meant by the *lexicographic ordering* on Σ^+. (2 marks)

d) Denote the lexicographic order on Σ^+ by $\preceq$. Using pseudocode, give an algorithm to determine whether, for elements s, t of Σ^+, $s \preceq t$. Assume that strings are implemented as linked lists of characters. Give the time complexity of your algorithm as a function of the lengths of s and t. (4 marks)

e) Suppose the algorithms to parts b) and d) of this question are combined to yield an algorithm for determining whether some given non-empty string s occurs in an array of non-empty strings, L, whose elements are arranged in ascending order. Give the time complexity of this algorithm as a function of the length of s and the total length $\|L\|$ of all the strings in L together. What is time complexity of your algorithm as a function of the length of s, the number of elements N in L and the maximum length m of any string in L?

(6 marks)

END OF EXAMINATION