UNIVERSITY OF EAST ANGLIA

School of Computing Sciences

Main Series UG Examination 2015-16

DATA STRUCTURES AND ALGORITHMS

CMP-5014Y

Time allowed: 3 hours

Section A (Attempt any 4 questions: 60 marks)

Section B (Attempt any 2 questions: 60 marks)

Notes are not permitted in this examination.

Do not turn over until you are told to do so by the Invigilator.

SECTION A

1. (a) Outline a general strategy for determining the complexity of an algorithm. [4 marks]

   (b) Use the strategy you have described to determine the worst-case run-time complexity of the following sorting algorithm.

```
for i from 1 to n-1 do
  //Find the item next, that should be in
  //position i in the sorted array:
  next := a[i];
  indexOfNext := i;
  for j from i+1 to  n do
    if ( a[j] < next ) then
      next := a[j];
      indexOfNext := j;
    endif
  endfor
  swap the positions of item a[i] and next in the array;
  // items in positions 1, ..., i are now in sorted order.
endfor;
```

   Show your working. [11 marks]

2. (a) Describe how a FIFO queue may be represented using a "circular" array. [4 marks]

   (b) Give algorithms for the `enqueue` and `dequeue` methods for a queue represented in this way.

   You may assume that a method, `doubleCircularArray`, is available for doubling the length of the underlying array when necessary. [8 marks]

(c) State the run-time complexity of each of your algorithms. [3 marks]

3. (a) Describe a linear-time heap construction algorithm, illustrating your answer by constructing a max heap for the following numbers.

$$37, 25, 31, 39, 36, 38, 42, 26, 44, 38, 42, 45, 53, 60, 46.$$

[7 marks]

(b) Determine the maximum number of swap operations your algorithm could require to construct a heap from thirty one numbers (in a swap operation, a parent value is swapped with one of its child values). [2 marks]

(c) Describe an algorithm for inserting an additional element into a heap, assuming that the underlying data structure used to represent the heap is a one-dimensional array, `a`. [6 marks]

4. (a) Define what is meant by a B-tree of order m. [5 marks]

(b) Draw diagrams to illustrate the insertion of the following keys, in the order given, into a 2-3 tree.

$$39, 62, 43, 27, 14, 60, 35, 24, 82, 51, 33.$$

[10 marks]

5. (a) In the context of hashing, briefly describe what is meant by each of the following terms:

   (i) open addressing;

   (ii) probe increment;

   (iii) quadratic probing;

   (iv) double hashing. [8 marks]

   (b) Let $h(k) = k\%11$, where $\%$ is the remainder operator (e.g. $25\%11 = 3$). Insert the following set of keys into a hash table of size 11, using double hashing to resolve collisions, with the probe increment defined as follows:

$$p(k) = 5 - (k\%5).$$

Give the sequence of locations probed during the insertion of each key.

$$17, 9, 26, 34, 25, 31, 37, 48.$$

[7 marks]

SECTION B

6. (a) Explain what is meant by a “divide-and-conquer" algorithm. [4 marks]

(b) Describe both informally and formally in pseudo-code the divide-and-conquer algorithms *merge sort* and *Quicksort* for sorting an array of comparable items into ascending order. [12 marks]

(c) Illustrate the use of the *merge sort* algorithm and the *Quicksort* algorithm (using the median of three pivot) to sort the following array into ascending order:

$$A = [12, 28, 14, 12, 30, 11, 8, 35, 11, 12].$$

In each case, show all working and count the number of comparisons performed. [6 marks]

(d) Describe informally and formally in pseudo-code a partitioning algorithm for *Quicksort* that allows an in-place implementation. Demonstrate how the algorithm works by partitioning the following array

$$A = [12, 28, 14, 12, 30, 11, 8, 35, 11, 12]$$

using the pivots you obtained in part (c). [6 marks]

(e) State the worst-case and average case time complexity of the *Quicksort* algorithm and the *merge sort* algorithm. [2 marks]

7. (a) In the context of binary trees, explain what is meant by:

   (i) the *height* of the tree; [2 marks]

   (ii) a *height-balanced* binary tree. [6 marks]

   (b) Draw diagrams to illustrate the effect in a binary tree of

   (i) a *right* rotation; [2 marks]

   (ii) a *left* rotation. [2 marks]

   (c) (i) Insert the following integers in the order given into a binary search tree

$$33, 40, 38, 24, 45, 19, 34, 57, 35, 25, 49, 39, 42.$$

   [5 marks]

   (ii) Write the balance of each node next to it in your tree. State which node(s) is (are) unbalanced. [3 marks]

   (iii) Explain how the tree may be re-balanced, drawing diagram(s) to illustrate the process. [6 marks]

   (iv) Explain what happens when 33 is deleted from your balanced tree. What is the complexity of deletion in a height-balanced binary search tree. Justify your answer. [4 marks]

8. (a) Describe what is meant by a trie. [4 marks]

   (b) Given the following set of keys:

$$S = \{346, 354, 355, 3553, 3558, 357, 380, 510, 513, 5134, 546, 549\}.$$

   (i) Draw the trie corresponding to S. [6 marks]

   (ii) Explain how a trie may be represented as a binary tree; represent the trie from (i) as a binary tree. [10 marks]

   (iii) Explain how a trie may be represented as a two-dimensional array; represent the trie from (i) using this method. [10 marks]

9. Consider the following set of disjoint sets:

$$S = \{\{2,5,7\},\{3,11,6\},\{1,8,10,12\},\{4\}\}.$$

(a) Represent S by an appropriate forest of trees. [4 marks]

(b) Explain how your forest may in turn be represented by a one-dimensional array and give the array. [5 marks]

(c) Suppose that it is required to merge the set containing 11 with the set containing 10. Assuming no ranking rule is used in deciding how to do this merge, draw the two possible forests of trees that could result and give the corresponding one-dimensional arrays.

Which of the two forests is to be preferred and why? [8 marks]

(d) Give an algorithm that uses the array representation of a set of disjoint sets to find the set containing item i, where i is specified. Illustrate the application of this algorithm to find element 10 in each of the arrays given in part (c). [7 marks]

(e) Describe how a ranking rule based on the count of the number of nodes in a tree can be used when implementing the `merge` operation so as to reduce the average number of levels traversed during applications of the `find` operation. [6 marks]

END OF PAPER