IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2017

MSc in Computing Science
for Internal Students of the Imperial College of Science, Technology and Medicine

PAPER C580

ALGORITHMS

Friday 5 May 2017, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1 a An encryption routine, using Algorithm X, has a running time of $T_X(N) = 100\sqrt{N} + 4$ for an input containing N characters.

i) Using the formal definition of $O(f(N))$, show that $T_X(N) = O(2N)$.

ii) You are implementing an application to encrypt social media messages. Each message has a maximum of 144 characters. If the running time using Algorithm Y is $T_Y(N) = 2N$, for a message with N characters, would you build your application using Algorithm X or Algorithm Y, and why?

b Sort the following table of student data by degree, according to the ordering $a5 < r5 < s5 < v5$, using the Counting sort algorithm.

Student Id	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Degree	s5	r5	s5	v5	s5	a5	v5	v5	a5	v5	r5	s5	v5	v5	s5	a5

Your answer should show: the state of the counting table after the first pass through the input; the state of the counting table after the second pass through the input; the state of the counting table and the output table when the output table is half full; the state of the counting table and the output table when the output table is full.

c Mary Startupperberg is travelling to meet an important investor. The investor is available all day, so Mary can choose any time to meet. The train takes 2 hours and the earliest Mary could return would be 2 hours after she arrives, although she can stay longer if it is cheaper to return later. The train fares are contained in the arrays Out and Rtn, both of length N, such that $Out[i]$ and $Rtn[i]$ are the fares for the ith outbound and return trains respectively, which both depart at the same time. There is one train per hour in both directions, so the first possibility is to take the first outbound train and the *fifth* return train. Mary wants to find the lowest total fare she can pay.

i) Explain how to decompose this problem into N subproblems. Your answer should say what the subproblems are (what a solution to a subproblem means), how they can be solved, and how they combine to give the overall solution.

ii) Can the problem be solved using dynamic programming? Explain your answer with reference to your decomposition of the problem.

iii) Write a procedure that solves the problem. Your procedure should iterate through the input data at most once.

The three parts carry, respectively, 30%, 25%, and 45% of the marks.

2 a i) What is the time taken to insert a new key k into a binary search tree (BST) containing N keys, where k is not already in the tree, in the best case and the worst case? Explain your answer.

ii) What is the time taken for the operation described in (i), in the case of a BST that has the same shape (exactly the same parent–child relationships) as a red-black tree containing the same keys? Explain your answer.

iii) Given a red-black tree containing N keys and an identically shaped BST containing the same keys, as in (ii) above, and a new key k as in (i), would you expect it to take longer to insert k into the binary search tree or into the red-black tree? Assume that all operations common to both insert procedures take the same time. Your answer should explain why the longer insertion takes longer, and how much extra time is taken, in Θ notation.

b i) What is *collision resolution* in hash tables and why is it necessary?

ii) What is the difference between the result of a hash function used in a table with collision resolution by chaining and one used in a table with probing?

iii) Describe how objects are added into a hash table that uses collision resolution by chaining, and one that uses probing. What is the time taken, in the worst case and the average case, to add a new object to each type of table if the table contains N objects and is of size m?

iv) Given a hash table containing N objects, briefly describe how you would return all the objects, ordered by key. What would be the time taken by your scheme? State any assumptions that you make.

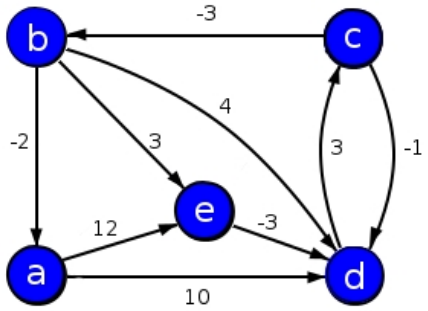
The two parts carry, respectively, 45% and 55% of the marks.

3 a You are implementing a game in which a player attempts to escape from a maze. The maze exists within a $M \times N$ grid, has one exit, and is defined by a set of *connections*. From their current location the player can move horizontally or vertically to a neighbouring point, but only if a connection exists to that point. So, only if a connection $[[1,2],[1,3]]$ is part of the maze's definition, is it possible to take a step between points $[1,2]$ and $[1,3]$ in either direction.

i) An *escape route* is a way to reach the exit in the fewest steps. Starting with the list $C = [C_1, \ldots, C_k]$ of connections and the location $[e_x, e_y]$ of the exit, describe the operations you would perform before the game is played so that an escape route from any point $[x, y]$ can be obtained during the game within $\Theta(s)$ time, where s is the number of steps in the route. No coded procedures are required, but you should give details of each stage of the process.

ii) Assume that it is possible to reach the exit from every point in the maze. What is the time taken by your pre-game procedure? Explain your answer.

b The Bellman–Ford algorithm is applied to the following graph, with source vertex a. The order followed when looping through edges is $a \rightarrow$, $b \rightarrow$, $c \rightarrow$, $d \rightarrow$, $e \rightarrow$, where $v \rightarrow$ is the set of all edges *from* v to any other vertex.



i) Why is Dijkstra's algorithm not appropriate for this graph? Describe the execution up to the first point at which the solution becomes incorrect.

ii) Write out a table showing all changes to the shortest path estimate for each vertex during execution of the Bellman–Ford algorithm.

iii) What would be the result of the Bellman–Ford algorithm if the weight of the edge (d, c) is changed to 1?

The two parts carry, respectively, 55% and 45% of the marks.

4 a The *Heapsort* algorithm sorts an array A using $\Theta(1)$ memory, by making A heap-ordered, and then moving values to their correctly sorted positions.

i) Write a procedure that implements *Heapsort*, sorting its input array into *ascending* order. Procedures to add and remove values from a heap can be called in HEAPSORT without giving code for them. Describe the properties of the subarrays created within A and how they are maintained.

ii) Given an input array of size N, what will be the running time of your HEAPSORT procedure, in Θ notation? Explain your answer. Include an explanation of the performance of the heap operations.

b The Quicksort algorithm presented in this course uses the *Lomuto* partitioning scheme, which divides the data into three parts: elements less than the pivot, the pivot element, and elements greater than or equal to the pivot. The following algorithm implements the alternative *Hoare* partitioning scheme.

```
1:  procedure HOAREPARTITION(A = [A_1, ..., A_N])
2:      pivot = A_N, i = 0, j = N + 1
3:      while true do
4:          repeat
5:              i = i + 1
6:          until A_i ≥ pivot
7:          repeat
8:              j = j − 1
9:          until A_j ≤ pivot
10:         if i ≥ j then
11:             return i
12:         else
13:             SWAP(A, i, j)
```

i) After carefully considering the properties of the partitions created by the code above, write a procedure SORT that implements Quicksort using HOAREPARTITION.

ii) Describe a type of input for which Quicksort is asymptotically faster using Hoare partitioning rather than Lomuto partitioning. Explain your answer, stating the running time for each algorithm, in Θ notation.

iii) What is the worst case input, and its running time, for your SORT procedure? Explain your answer.

The two parts carry, respectively, 45% and 55% of the marks.