香 港 中 文 大 學
The Chinese University of Hong Kong

二〇一七至一八年度下學期科目考試
Course Examination 2nd Term, 2017-18

科目編號及名稱 Course Code & Title :	CSCI2100A/ESTR2102 Data Structures			
時間 Time allowed :	2	小時 hours	00	分鐘 minutes
學號 Student I.D. No. :		座號 Seat No. :		

Please note the following:

1. The written examination is an close-book and open-notes examination. You may bring your notes on one sheet of A4 paper.
2. Please put your student ID on both the answer sheet and the examination question script.
3. You have 120 minutes on this 100/120 points examination. This means that you should spend about 1 minute for each 1 point.
4. More answer scripts are available upon request.
5. For regular students, answer questions only up to 4(e). Your maximum score is 100. You will not receive any additional bonus points for answering questions for Elite students.
6. For Elite students, answer all questions, your maximum score is 120.
7. No calculator is needed. Turn off your phone.

1. **Hashing (20)**

   Given the input $\{16, 36, 2, 48, 6, 84, 77, 26, 96, 12\}$ and a hash function $h(x) = x \bmod 10$, show what happens when each element in the list is inserted into the hash table of size 10 using the following collision policy.

   (a) (5) Open hash table. Explain what method of insertion you are using.

   (b) (5) Closed hash table using linear probing (assuming that the increment is 1).

   (c) (5) What are the advantages and disadvantages of the various collision strategies in the first two questions? Justify your answer.

   (d) (5) Which hash function is a better hash function for the above sequence: (1) $h_1(x) = x \bmod 12$ or (2) $h_2(x) = x \bmod 13$? Justify your answer. Note that the size of hash table will change according to hash function.

2. **Sorting (26)**

   (a) (5) Show all the inversion pairs in the list $(3, 5, 4, 6, 2, 1, 7)$? List them out in an organized manner, i.e., sort them in an ascending order by the first element and then by the second element.

   (b) (5) Given the sequence (11,19,12,3,9,6,18,4), sort the sequence into ascending order using merge sort. Illustrate the result after each pass. How many comparisons have you performed?

   (c) (8) Given an array with n elements to sort, what is the *average* and *worst* time complexity when using bubble sort, selection sort, heap sort and quick sort in terms of big-O notation.

   (d) (4) What is the loop invariant condition during one pass of the Quicksort that was demonstrated in the class note? (Hint: illustrate your answer using p as the pivot element.

   (e) (4) You have a list of elements that could either be b or w to be sorted, e.g., $(b, b, w, b, w, b, w, w, b)$. You only allowed to sort the elements in $O(n)$ time. What type of sorting algorithm will you use?

3. **Short Answers (28)** (Please give only concise and short answers!)

   (a) (2) Evaluate the following recurrence relation:

   $$T(n) = T(n/2) + n \log n,\ T(1) = 1.$$

   (b) (4) The follwoings are the results of a binary tree's in-order traversal and post-order traversal. Please draw the corresponding tree. Note that the tree has 8 nodes in total, which are from A to G.
   In-order: D H B E A F C G Post-order: H D E B F G C A

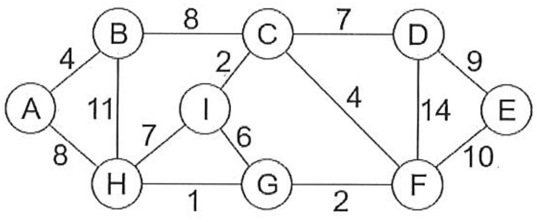
   (c) (6) Given an array of $n+1$ integers from 0 to $n-1$, where one of these numbers is duplicated. Design an algorithm by writing a pseudo code to find this duplicate in time $O(n)$ and space $O(1)$. For example, given an array of (0,1,2,3,4,2) your algorithm should return 2. Note that the array is not necessarily sorted.

   (d) (6) Given an array of n integers, design an algorithm by writing a pseudo code to find the majority element n time $O(n)$ and space $O(1)$. The majority element is the element that appears more than $n/2$ times. For example, given an array of 1,0,0,2,0 your algorithm should return 0. You may assume that the array is non-empty and the majority element always exists in the array.

   (e) (10) What is the average-case and worst-case run times of a single `DeleteMin` operation on the following types of data structures? The answers should be expressed in the big-O notation in terms of n, the number of elements stored in the data structure.

Data Structure	Average-Case	Worst-Case
Binary Search Tree		
AVL Tree		
Min-Heap		
Descending-Sorted Array		
Descending-Sorted Linked List		

4. **Graph (26/36)**

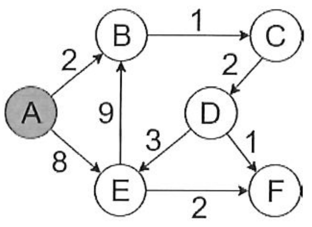


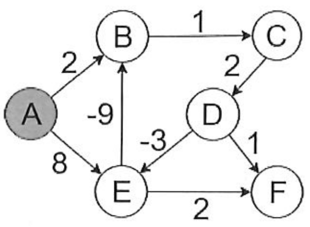
3-1

(a) (4) Draw the corresponding adjacency matrix for graph 3-1.

(b) (4) For graph 3-1, please list the breadth-first search sequence starting from vertex A.

(c) (8) Suppose that you have access to the sorted weight list in an ascending order in $O(1)$, which of the minimum cost spanning tree algorithms (Prim or Kruskal) would you use to achieve a better running time performance for graph 3-1? Justify why your selected algorithm is better suited for the above graph. Show the final result of the minimum cost spanning tree that you have obtained.





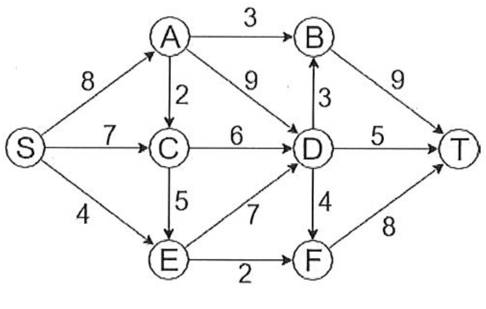
3-2 3-3

(d) (5) For graph 3-2, Find the shortest weighed path from vertex A to F using the Dijkstra's algorithm. Illustrate intermediate steps with a table.

(e) (5) Figure out whether there exists a topological sorting in graph 3-2, justify your answer. If yes, write out one valid topological sorting result. If your answer is no, please remove the least number of edges to construct a valid topological sorting, write down the topological sorting result after removing the edges.

(f) **(For Elite)**(5) For graph 3-3, Find the shortest weighed path from vertex A to all the other vertices using the Bellman-Ford algorithm. Illustrate intermediate steps with a table.

(g) **(For Elite)**(5) Given a graph 3-4, as shown below, with each number on an arc indicating a capacity, and S is the source vertex and T is the target vertex. You are asked to find a flow of maximum value. (You need to draw the graph in the answer book, give the flow of each edge, show the minimum cut in the graph, and compute the maximum value)



3-4

5. **(For Elite) Dynamic Programming (10)**
The Longest Increasing Subsequence problem is defined as follows. Given a sequence S with length n, output the length of the longest subsequence in S such that all numbers in the subsequence are sorted in ascending order. For example, if $S = [5, 11, 3, 1, 50]$, you need to output 3, because the longest increasing subsequence is $[5, 11, 50]$.

   (a) (2) What is the advantage of a dynamic programming algorithm over a pure recursive algorithm?

   (b) (6) Write down the optimal substructure for the Longest Increasing Subsequence problem.

   (c) (2) Based on the optimal substructure, we can devise a dynamic programming algorithm to solve the Longest Increasing Subsequence problem. Write down the time complexity and the space complexity of this dynamic programming algorithm in terms of big-O.

-End-