CITY UNIVERSITY OF HONG KONG

Course code & title	:	CS4335 Design and Analysis of Algorithms
Session	:	**MOCK EXAM**
Time allowed	:	Two hours

This paper has FOUR pages (including this cover page).

---

1. This paper consists of SIX questions.

2. Answer ALL questions.

---

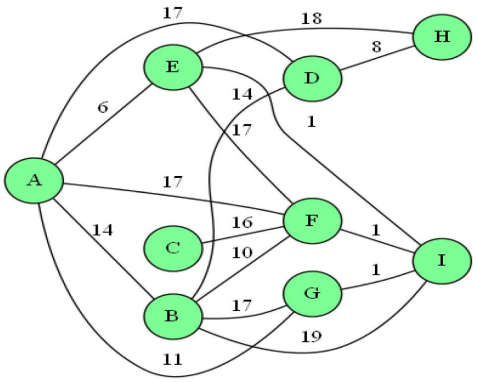
*This is a **closed-book** examination.*
This question paper should NOT be taken away.

No materials or aids are allowed during the whole examination. If any unauthorized materials or aids are found on a candidate during the examination, the candidate will be subject to disciplinary action.

WARNING: There are NO guarantees that this mock paper shares any similarities with the final exam paper.

Question 1. (15 points)

(a) (6 points) Use Prim's algorithm to find a minimum spanning tree for the following graph. Intermediate steps are required, you should indicate the order that the edges are selected.



(b) (9 points) There are 8 lectures with starting time and finish time as follows: (7, 12), (11, 14), (12, 14), (4, 7), (7, 11), (1, 6), (2, 5), and (8, 11). Give the schedule that minimizes the number of classrooms used.

Question 2. (20 points)

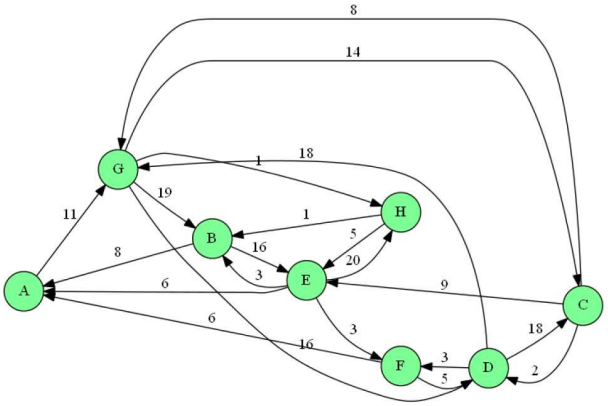
(a) (5 points) Compute the SCS for the two sequences s1 and s2: s1= abbabbabaa and s2=ababbababa. Backtracking process is required. What is the running time of the algorithm?

(b) (10 points) Write the pseudo-code of weighted interval schedule problem. Backtracking process is required.

(c) (5 points) Using the linear time method to find all the occurrences of **abcabc** in **ab*cabcabcabab***. Intermediate steps are required. You do not need to show the intermediate steps for failure function calculations.

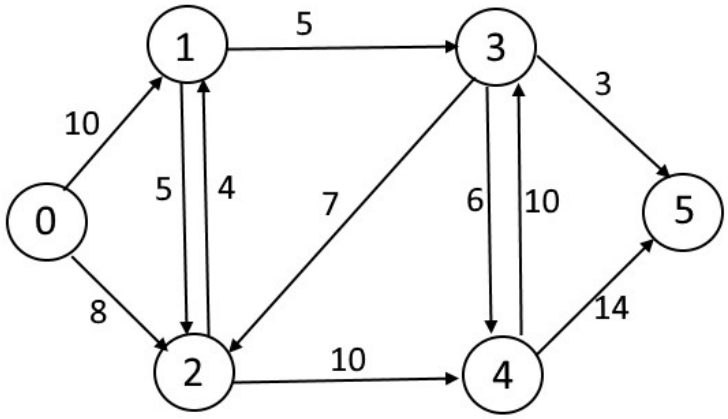
Question 3. (10 points)

(a) (10 points) Consider the following undirected graph. Find the shortest path from A to E in the following graph using Dijkstra's algorithm. Intermediate steps are required. The backtracking for finding the shortest path is required.



Question 4. (20 points)

(a) (6 points) Compute the maximum flow for the following graph (source *0* and sink *5*) using Edmonds-Karp's algorithm. Intermediate steps are required.



Source : 0
Sink : 5

(b) (10 points) A matching in a Bipartite Graph is a set of the edges chosen in such a way that no two edges share an endpoint. A maximum matching is a matching of maximum size (maximum number of edges). How can you use Edmonds-Karp's algorithm to solve the maximum matching problem for bipartite graph? What is your running time? State your algorithm using English or the pseudo code.

(c) (4 points) Write the pseudocode for the 0-1 knapsack problem.

(d) (6 points) What is the running time for the algorithm to solve the 0-1 knapsack problem? Is the running time in polynomial to the input size? Justify your answer.

Question 5. (10 points)

(a) (3 points) Design a divide and conquer approach to find the maximum element from an array in integers. You can use either English or pseudo code to describe your algorithm. 1

(b) (7 points) The following is a high-level version of the quicksort algorithm. Suppose S is a set of numbers.

```
    function quicksort(S)
1.    if |S| ≤ 1
2.      then return(S)
3.      else
4.        Choose an element a from S
5.        Let S1, S2, S3 be the elements of S that are respectively <, =, > a
6.        return(quicksort(S1),S2,quicksort(S3))
```

Suppose a is always the median (the median of a set is half of elements are larger than it, and half are less) of the input set. What is the running time for quicksort? Write the recurrence.

Question 6. (25 points)

(a) (10 points) Word Break Problem: Given a string and a dictionary of words, determine if the string can be segmented into a space-separated sequence of one or more dictionary words. **Design** an algorithm to solve the problem. You can use either English or pseudo code to describe your idea. State the running time of the method.

For example,

Input:

dict[] = { this, th, is, famous, Word, break, b, r, e, a, k, br, bre, brea, ak, problem };

word = Wordbreakproblem

Output:

Word b r e a k problem
Word b r e ak problem
Word br e a k problem
Word br e ak problem
Word bre a k problem
Word bre ak problem
Word brea k problem
Word break problem

(b) (10 points) Find a subsequence of a given sequence such that the subsequence sum is as high as possible and the subsequence's elements are sorted in ascending order. This subsequence is not necessarily contiguous or unique. Please note that the problem specifically targets subsequences that need not be contiguous, i.e., subsequences are not required to occupy consecutive positions within the original sequences. For example, consider subsequence {0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11}. The maximum sum increasing subsequence is {8, 12, 14} which has sum 34. **Design** an algorithm to solve the problem. You can use either English or pseudo code to describe your idea. State the running time of the method.

(c) (5 points) Given a string, count the number of times a given pattern appears in it as a subsequence. Please note that the problem specifically targets subsequences that need not be contiguous, i.e., subsequences are not required to occupy consecutive positions within the original sequences. **Design** an algorithm to solve the problem. You can use either English or pseudo code to describe your idea. State the running time of the method. For example,

Input:

string = "subsequence"
pattern = "sue"

Output: 7

subsequence
subsequence
subsequence
subsequence
subsequence
subsequence
subsequence

- END -