National University of Computer & Emerging Sciences, Karachi
Fall-2017 Department of Computer Science
Mid Term - 2
24th October 2017, 11:00 am– 12:00 noon

Course Code: CS302	**Course Name:** Design and Analysis of Algorithm
Instructor Name / Names: Dr. Muhammad Atif Tahir, Subhash Sagar and Zeshan Khan	
Student Roll No:	**Section:**

Instructions:

- Return the question paper.
- Read each question completely before answering it. There are **4 questions.**
- In case of any ambiguity, you may make assumption. But your assumption should not contradict any statement in the question paper.

Time: 60 minutes. **Max Marks**: 10 points

Question 1:

a) Apply BFS(A), BFS(F), DFS(A), DFS(G) in Graph shown in Figure 1. Show all steps showing Queues and Visit Order [2 Points]

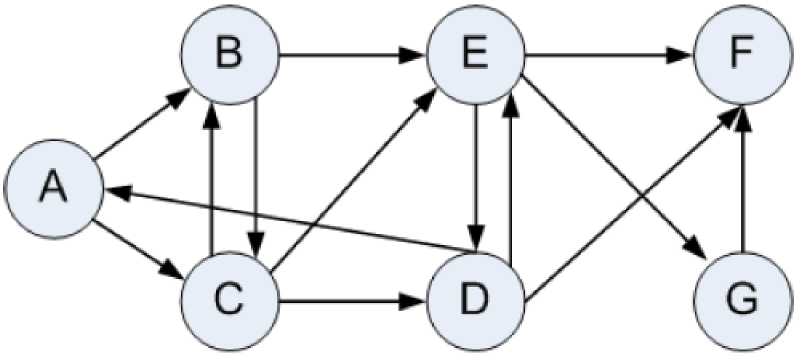


Figure 1: Graph for BFS, DFS and Toplological Sort.

Answer (a) on BFS and DFS

BFS(A): (i) Queue {A}, Visit Order {A}
(ii) Queue {B,C}, Visit Order {A, B}
(iii) Queue {C,E}, Visit Order {A, B, C}

(iv) Queue {E, D}, Visit Order {A, B, C, E}
(v) Queue {D, F, G}, Visit Order {A, B, C, E, D}
(vi) Queue {F, G}, Visit Order {A, B, C, E, D, F}
(vii) Queue {G}, Visit Order {A, B, C, E, D, F, G }
(viii) Queue {}, Visit Order {A, B, C, E, D, F, G }

BFS(F): (i) Queue {F}, Visit Order {F}
(ii) Queue {}, Visit Order {F}

DFS(A): (i) DFS(C) or DFS(B)
(ii) DFS(B) to DFS(C) or DFS(E)
(iii) DFS(C) to DFS(E) or DFS(D)
(iv) DFS(E) to DFS(D) or DFS(F) or DFS(G)
(v) DFS(G) to DFS(F)
(vi) DFS(F)

Visit Order: {F,G,D,E,C,B,A}
DFS(G): (i) DFS(F)

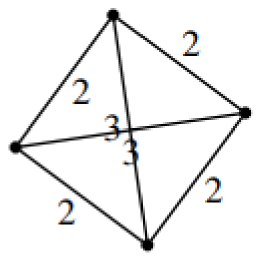
Visit Order: {F,G}

Question 2: **(2 points)**

Given any connected undirected graph G with positive edge weights w, does there always exist a single shortest path tree S such that S is also a minimum spanning tree (MST) of G?(**Yes/No**) Prove your answer by giving an example of graph. (**Note.** Graph must have at least 4 vertices).

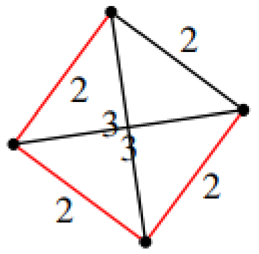
Solution:

No MST does not always lead to shortest path. Here is the counter example

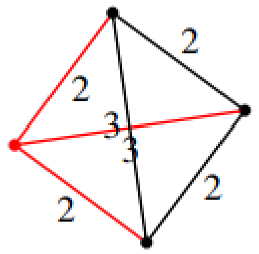


In the above graph, the outside edges each have a weight of 2, and the diagonal inner edges each have a weight of 3.

The minimum spanning trees of this element are any three of the weight-2 edges, as highlighted in red below:



The shortest path tree from each node contains just the edges containing that node, as shown below for the node highlighted in red:



<u>Question 3:</u> **(3 points)**

Travel Agency wants to setup a public transport system between all the cities. The passenger fare in rupees between the cities are shown in the **Figure-2**. How all cities should be linked to maximize the total fare? [Hint: Use Spanning Tree]

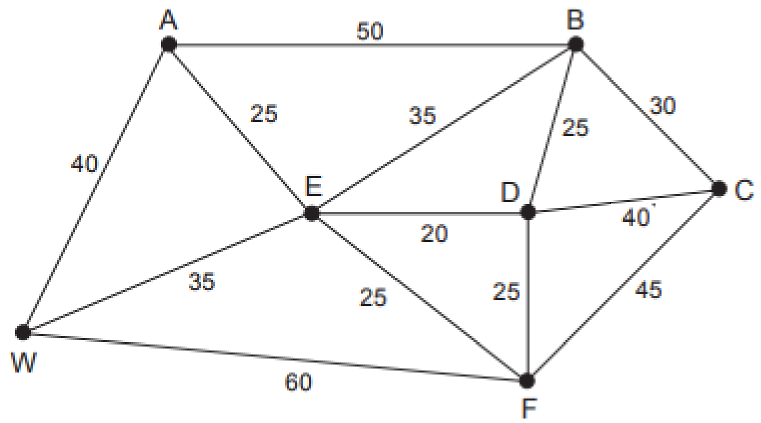
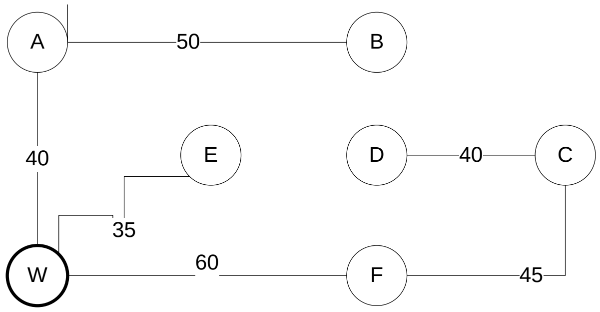


Figure-2

Solution:

Edges to be selected using Kruskal's Algo in the following order:



BE is also possible in place of WE.

Total Cost = 60 + 50 + 45 + 40 +40 + 35 = 270

There are other solutions as well e.g. Prims

Question 4:

(3 points)

Consider the following instance of the 0,1 knapsack problem

Item	1	2	3	4	5
Benefit	2	6	1	8	5
Weight	1	1	2	3	2

The maximum allowable total weight in the knapsack is $W = 5$.

Find an optimal solution for the above problem with the weights and benefits above using Dynamic Programming. Be sure to state both the value of the maximum benefit that you obtain as well as the item(s) that you need to obtain this benefit. Show all steps.

Create Value table

Value[i,w]	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	2	2	2	2	2
2	0	6	8	8	8	8
3	0	6	8	8	9	9
4	0	6	8	8	14	16
5	0	6	8	11	14	16

Maximum Benefit = Value[5][5] = 16

Create keep table for finding items

Keep	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	1	1	1	1
2	0	1	1	1	1	1
3	0	0	0	0	1	1
4	0	0	0	0	1	1
5	0	0	0	1	0	0

- Since keep[5,5] == 0, we do not keep this item
- Now, we look for keep[4,5] = 1. so we keep it and reduce the weight as follows W = 5 - 3 = 2
- keep[3,2] = 0, we do not keep it
- keep[2,2] = 1, we weep it and W = 2 - 1 = 1
- keep[1,1] = 1, we keep it and this will end the back tracing

Item selected 1, 2, 4