[CS21003 : Algorithms-I] Online-Quiz/Test

Date: 08-April-2020 (Wednesday)
Time: 10:00 am - 12:00pm (2-hours)
Total Marks: 20 -- Three Questions [Q1 = 6-marks ++ Q2 = 7-marks ++ Q3 = 7-marks]
Course: CS21003 -- Algorithms-I
Session: Spring - 2020

INSTRUCTIONS: (Please read carefully!)
-- There are THREE questions. You are asked to answer ALL of these.
-- It is advised that you first read the question, solve it fully in a rough-paper before going on entering the results (all together for that question) in the portal and finally cross-check your answers with the entered values.
-- Within the permitted 2-hours time of online-test, you can update/modify/correct your answers and re-submit with your log-in, as many times as you require.
-- You are free to consult any resources you want, but plagiarisms will be severely penalized (as per institute norms).
-- *comma-separated format* means providing values of an array/list in sequence separated by only comma(,) and nothing else.
[To enter an example 4-valued array, say ARR[ ] = {4, 7, 5, -2}, in comma-separated format, it will be written as ONLY 4, 7, 5, -2 (i.e. 4 values in sequence + 3 commas in between and NOTHING ELSE)]

NOTE: You NEED to SIGN-IN to your GOOGLE account to participate in this Online Quiz/Test.
* Required

1. Email address *

2. Name *

3. Roll-Number *

4. Department *

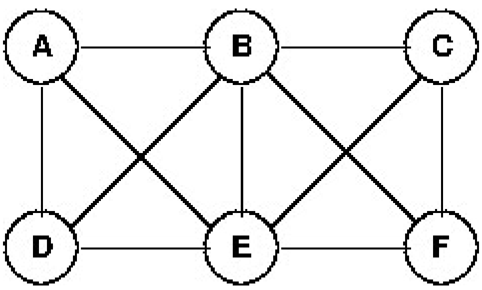
5. Disclaimer *

Check all that apply.

☐ I have read the INSTRUCTIONS and understood the same. I hereby indicate my participation in the Online-Quiz/Test.

Question-1: Minimum Spanning Tree [Marks: 6]	Given a weighted undirected graph G =<V, E, W>, where the vertices are V = {A,B,C,D,E,F} (|V| = 6), the edges are E = {(A,B), (A,D), (A,E), (B,C), (B,D), (B,E), (B,F), (C,E), (C,F), (D,E), (E,F)} (|E| = 11), the weights form the set W (not specified and you have to enter). PLEASE REFER TO THE IMAGE (Figure-1) GIVEN BELOW. Your are asked to do the following: [i] Define the weights (as a positive number) for each edge to make the example (wighted undirected graph) complete. Remember, every edge-weight should be a distinct number. [ii] Show the step-wise running (as asked below) of Prim's algorithm over the example graph that you constructed. Assume that, 'A' be the starting vertex/node to initiate this algorithm. [Marks = 3] More precisely, you shall be notifying the next edge to be added at every step. [iii] Show the step-wise running (as asked below) of Kruskal's algorithm over the example graph that you constructed. [Marks = 3] More precisely, you shall be notifying the next edge to be added at every step.

Figure-1 (for Question-1)



6. Are you Attempting Question-1? *

Mark only one oval.

- ( ) YES
- ( ) NO

7. Enter Weight of Edge-(A,B)

8. Enter Weight of Edge-(A,D)

9. Enter Weight of Edge-(A,E)

10. Enter Weight of Edge-(B,C)

11. Enter Weight of Edge-(B,D)

12. Enter Weight of Edge-(B,E)

13. Enter Weight of Edge-(B,F)

14. Enter Weight of Edge-(C,E)

____

15. Enter Weight of Edge-(C,F)

____

16. Enter Weight of Edge-(D,E)

____

17. Enter Weight of Edge-(E,F)

____

18. Prim's Algorithm: Added Edge at Step-1

Let, 'A' be the starting vertex/node to initiate the algorithm.

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

19. Prim's Algorithm: Added Edge at Step-2

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

20. Prim's Algorithm: Added Edge at Step-3

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

21. Prim's Algorithm: Added Edge at Step-4

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

22. Prim's Algorithm: Added Edge at Step-5

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

23. Prim's Algorithm: What is the total cost of the Minimum Spanning Tree that you formed?

____

24. Kruskal's Algorithm: Added Edge at Step-1

Mark only one oval.

- ( ) Edge-(A,B)
- ( ) Edge-(A,D)
- ( ) Edge-(A,E)
- ( ) Edge-(B,C)
- ( ) Edge-(B,D)
- ( ) Edge-(B,E)
- ( ) Edge-(B,F)
- ( ) Edge-(C,E)
- ( ) Edge-(C,F)
- ( ) Edge-(D,E)
- ( ) Edge-(E,F)

25. Kruskal's Algorithm: Added Edge at Step-2

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

26. Kruskal's Algorithm: Added Edge at Step-3

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

27. Kruskal's Algorithm: Added Edge at Step-4

Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

28. Kruskal's Algorithm: Added Edge at Step-5

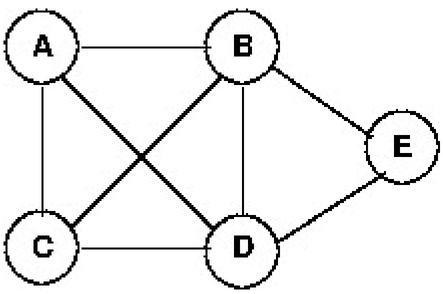
Mark only one oval.

- Edge-(A,B)
- Edge-(A,D)
- Edge-(A,E)
- Edge-(B,C)
- Edge-(B,D)
- Edge-(B,E)
- Edge-(B,F)
- Edge-(C,E)
- Edge-(C,F)
- Edge-(D,E)
- Edge-(E,F)

29. Kruskal's Algorithm: What is the total cost of the Minimum Spanning Tree that you formed?

Question-2: Single-source Shortest Path [Marks: 7]	Five cities, {A, B, C, D, E}, are connected via roads, {(A,B), (A,C), (A,D), (B,C), (B,D), (B,E), (C,D), (D,E)}, in the following manner AS SHOWN IN THE IMAGE (Figure-2) BELOW. There is a travel-cost to travel from one city to another (any direction) via the road connections. Let, all the travel-costs are unique and the travel-cost from City-X to City-Y is the same as the travel-cost from City-Y to City-X. Moreover, there is a tourist-cost (associated with every city) which one has to pay if (s)he touches any city while travelling. Your are asked to do the following: [i] Define all the travel-costs and tourist-cost (as a positive number) to make the above example (wighted undirected graph) complete. Remember that, all the travel-cost between cities are distinct positive values and the tourist-cost for the cities are also distinct positive values (though, some travel-cost may match with some tourist-cost). [ii] You start from the City-A and want to estimate the minimum cost route to visit all the other destination cities, {B, C, D, E}. Present the modified version Dijkstra's algorithm to solve this problem. [Marks = 2] [iiI] Show the step-wise running (as asked below) of a modified Dijkstra's algorithm over the example graph that you constructed. [Marks = 5] More precisely, you shall be updating the two 5-element 1-D arrays, cost[ ] and parent[ ]; where -- --> cost[Z] indicates the minimum-cost values to reach City-Z from City-A. Initially, cost[A] = tourist-cost(A), cost[B] = infinity, cost[C] = infinity, cost[D] = infinity, cost[E] = infinity. --> parent[Z] indicates the previous city from which City-Z is being reached. Initially, parent[A] = null, parent[B] = null, parent[C] = null, parent[D] = null, parent[E] = null.

Figure-2 (for Question-2)



30. Are you Attempting Question-2? *

Mark only one oval.

- ( ) YES
- ( ) NO

31. Enter Travel-Cost between (A,B)

32. Enter Travel-Cost between (A,C)

33. Enter Travel-Cost between (A,D)

34. Enter Travel-Cost between (B,C)

35. Enter Travel-Cost between (B,D)

36. Enter Travel-Cost between (B,E)

37. Enter Travel-Cost between (C,D)

38. Enter Travel-Cost between (D,E)

39. Enter Tourist-Cost for City-A

40. Enter Tourist-Cost for City-B

41. Enter Tourist-Cost for City-C

42. Enter Tourist-Cost for City-D

43. Enter Tourist-Cost for City-E

44. Solution: Modified Version of Dijkstra's Algorithm

Assume the following notation while writing the algorithm: The travel-cost between City-X and City-Y is C[X,Y], the tourist-cost for City-Z is T[Z], and the minimum-cost to a reach City-W from City-A (start-city) is given by cost[W].

45. Step-0 (Initiation): Enter initial 5-values (in comma-separated format) of the cost[ ] array (indexed as cost[A, B, C, D, E]).

For "infinity", write INFY for that entry

46. Step-0 (Initiation): Enter initial 5-values (in comma-separated format) of the parent[ ] array (indexed as parent[A, B, C, D, E]).

For "null", write NULL for that entry

47. Step-1: Starting (First Chosen/Marked) City (Vertex)

Mark only one oval.

- City-A
- City-B
- City-C
- City-D
- City-E

48. Step-1: Enter modified 5-values (in comma-separated format) of the cost[ ] array (indexed as cost[A, B, C, D, E]).

For "infinity", write INFY for that entry

49. Step-1: Enter modified 5-values (in comma-separated format) of the parent[ ] array (indexed as parent[A, B, C, D, E]).

For "null", write NULL for that entry

50. Step-2: Next Chosen/Marked City (Vertex)

Mark only one oval.

- City-A
- City-B
- City-C
- City-D
- City-E

51. Step-2: Enter modified 5-values (in comma-separated format) of the cost[ ] array (indexed as cost[A, B, C, D, E]).

For "infinity", write INFY for that entry

52. Step-2: Enter modified 5-values (in comma-separated format) of the parent[ ] array (indexed as parent[A, B, C, D, E]).

For "null", write NULL for that entry

53. Step-3: Next Chosen/Marked City (Vertex)

Mark only one oval.

- City-A
- City-B
- City-C
- City-D
- City-E

54. Step-3: Enter modified 5-values (in comma-separated format) of the cost[ ] array (indexed as cost[A, B, C, D, E]).

For "infinity", write INFY for that entry

55. Step-3: Enter modified 5-values (in comma-separated format) of the parent[ ] array (indexed as parent[A, B, C, D, E]).

For "null", write NULL for that entry

56. Step-4: Next Chosen/Marked City (Vertex)

Mark only one oval.

- City-A
- City-B
- City-C
- City-D
- City-E

57. Step-4: Enter modified 5-values (in comma-separated format) of the cost[ ] array (indexed as cost[A, B, C, D, E]).

For "infinity", write INFY for that entry

58. Step-4: Enter modified 5-values (in comma-separated format) of the parent[ ] array (indexed as parent[A, B, C, D, E]).

For "null", write NULL for that entry

59. Step-5: Final Chosen/Marked City (Vertex)

Mark only one oval.

- City-A
- City-B
- City-C
- City-D
- City-E

60. Step-5: Enter final 5-values (in comma-separated format) of the cost[ ] array (indexed as cost[A, B, C, D, E]).

For "infinity", write INFY for that entry

61. Step-5: Enter final 5-values (in comma-separated format) of the parent[ ] array (indexed as parent[A, B, C, D, E]).

For "null", write NULL for that entry

62. Costliest-City: Which city requires the MOST cost to be visited from City-A?

Mark only one oval.

- City-B
- City-C
- City-D
- City-E

63. Cheapest-City: Which city requires the LEAST cost to be visited from City-A?

Mark only one oval.

- City-B
- City-C
- City-D
- City-E

Question-3: All-pairs Shortest Path [Marks: 7]

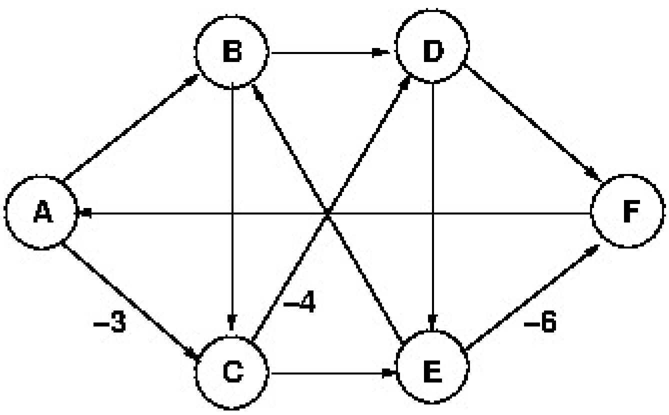
Given a weighted directed graph G =<V, E, W>, where the vertices are V = {A,B,C,D,E,F} (|V| = 6), the directed edges are E = {(A,B), (A,C), (B,C), (B,D), (C,D), (C,E), (D,E), (D,F), (E,B), (E,F), (F,A)} (|E| = 11), the weights form the set W (specified partially -- only negative edge-weights are given, that is, W(A,C)=-3, W(C,D)=-4 and W(E,F)=-6 and rest will be filled by you). PLEASE REFER TO THE IMAGE (Figure-3) GIVEN BELOW.

Your are asked to do the following:

[i] Define only the positive weights (as a positive number) for all remaining edges to make the example (wighted directed graph) complete.
Remember, each edge-weights should all be distinct positive numbers only (negative edge-weights are already given).

[ii] Show the step-wise running (as asked below) of Floyd-Warshall's algorithm over the example graph that you constructed. [Marks = 7]
More precisely, you shall be providing (in row-wise manner for all-6 rows) the 2-D cost calculation matrix (memoized) values F[ ][ ] at every step (you may refer to the lecture-slides!) and also indicate the number of negative entries in F[ ][ ].

Figure-3 (for Question-3)



64. Are you Attempting Question-3? *

Mark only one oval.

- [ ] YES
- [ ] NO

65. Enter Weight of Directed Edge-(A,B)

____

66. Enter Weight of Directed Edge-(B,C)

67. Enter Weight of Directed Edge-(B,D)

68. Enter Weight of Directed Edge-(C,E)

69. Enter Weight of Directed Edge-(D,E)

70. Enter Weight of Directed Edge-(D,F)

71. Enter Weight of Directed Edge-(E,B)

72. Enter Weight of Directed Edge-(F,A)

73. Step-0: Initial 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

74. Step-0: Initial 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

75. Step-0: Initial 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

76. Step-0: Initial 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

77. Step-0: Initial 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

78. Step-0: Initial 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[6][ ]

For "infinity", write INFY for that entry

79. Step-1: Updated 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

80. Step-1: Updated 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

81. Step-1: Updated 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

82. Step-1: Updated 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

83. Step-1: Updated 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

84. Step-1: Updated 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[6][ ]

For "infinity", write INFY for that entry

85. Step-2: Updated 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

86. Step-2: Updated 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

87. Step-2: Updated 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

88. Step-2: Updated 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

89. Step-2: Updated 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

90. Step-2: Updated 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[6][ ]

For "infinity", write INFY for that entry

91. Step-3: Updated 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

92. Step-3: Updated 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

93. Step-3: Updated 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

94. Step-3: Updated 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

95. Step-3: Updated 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

96. Step-3: Updated 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[6][ ]

For "infinity", write INFY for that entry

97. Step-4: Updated 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

98. Step-4: Updated 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

99. Step-4: Updated 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

100. Step-4: Updated 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

101. Step-4: Updated 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

102. Step-4: Updated 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[6][ ]

For "infinity", write INFY for that entry

103. Step-5: Updated 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

104. Step-5: Updated 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

105. Step-5: Updated 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

106. Step-5: Updated 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

107. Step-5: Updated 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

108. Step-5: Updated 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[6][ ]

For "infinity", write INFY for that entry

109. Step-6: Final 6-values of Row-1 (in comma-separated format) for Vertex-A in the 2-D cost-matrix, i.e. F[1][ ]

For "infinity", write INFY for that entry

110. Step-6: Final 6-values of Row-2 (in comma-separated format) for Vertex-B in the 2-D cost-matrix, i.e. F[2][ ]

For "infinity", write INFY for that entry

111. Step-6: Final 6-values of Row-3 (in comma-separated format) for Vertex-C in the 2-D cost-matrix, i.e. F[3][ ]

For "infinity", write INFY for that entry

112. Step-6: Final 6-values of Row-4 (in comma-separated format) for Vertex-D in the 2-D cost-matrix, i.e. F[4][ ]

For "infinity", write INFY for that entry

113. Step-6: Final 6-values of Row-5 (in comma-separated format) for Vertex-E in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

114. Step-6: Final 6-values of Row-6 (in comma-separated format) for Vertex-F in the 2-D cost-matrix, i.e. F[5][ ]

For "infinity", write INFY for that entry

115. Finally, how many entries have negative values in your (memoized) cost-calculation matrix, F[ ][ ] (having 6x6 dimension)?

Mark only one oval.

- 0
- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8
- 9
- 10
- 11
- 12
- 13
- 14
- 15
- 16
- 17
- 18
- 19
- 20
- 21
- 22
- 23
- 24
- 25
- 26
- 27
- 28
- 29
- 30

- 31
- 32
- 33
- 34
- 35
- 36