# Discrete Mathematics

# LECTURE 11

# Shortest Path & Distance

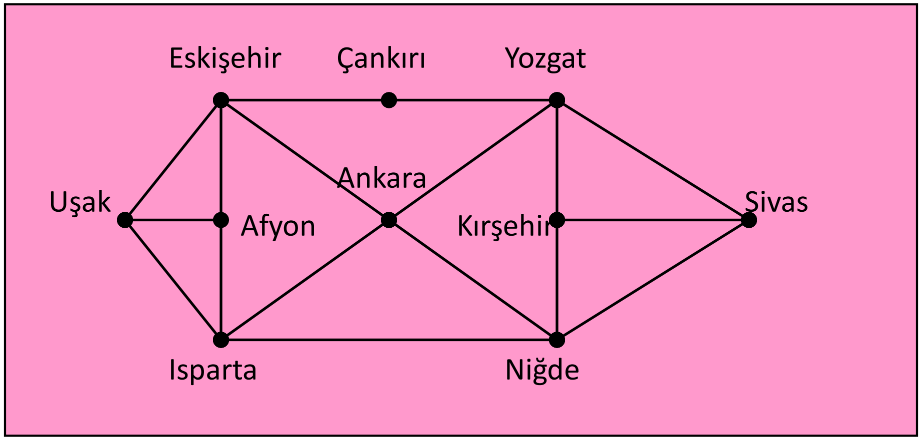
**Assistant Professor Gülüzar ÇİT**

# Outline

- Shortest Path & Distance
  - Breath First Search Algorithm
  - Dijkstra Algorithm
- References

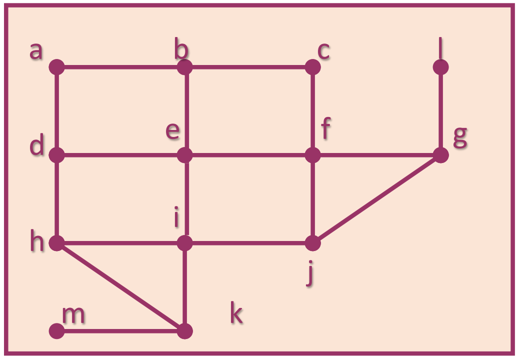
# Shortest Path & Distance

- The path with the minimum edges between any S and T nodes in a graph is called the **shortest path**.
- The total number of edges on this S-T path is called the **distance** from node S to node T.

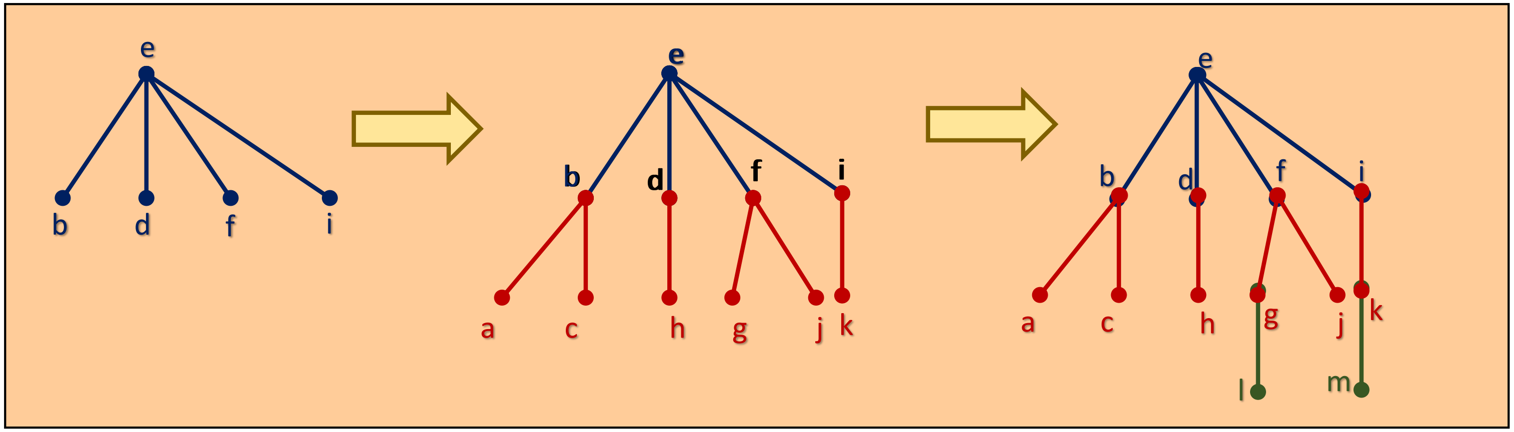


# Shortest Path & Distance…

➢ **Example:** Fin the shortest path for the graph on the left from node e to all other nodes




➢ **Solution:**



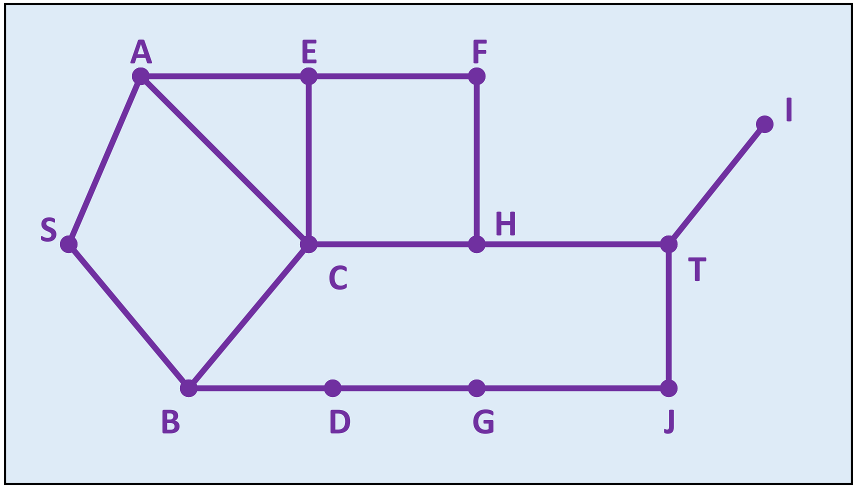
# Shortest Path & Distance…

## Breath First Search Algorithm…

```
STEP 1 ( label S )
   a) Give S label 0, S has no prior
   b) L ={S} and k=0
STEP 2 ( label nodes)
   REPEAT
      STEP 2.1 ( increase label)
          k = k + 1
      STEP 2.2 (expand labeling)
          WHILE (L has a node V labeled as k-1 adjacent to a node W not in L)
            a) assign label k to node W
            b) assing node V as the prior of node W
            c) Add node W to L
          END WHILE
    UNTIL (T is inside L or none of the nodes not in L are neighbors to the nodes in L)
STEP 3 (create the shortest path to T)
    IF (T is in L)
          Create the shortest path until it reaches node S by using the prior to node T,
          prior of predecessor, etc.,
          The label of T will show the distance from S to T.
    ELSE
          There is no path from S to T
    END IF
```

# Shortest Path & Distance…

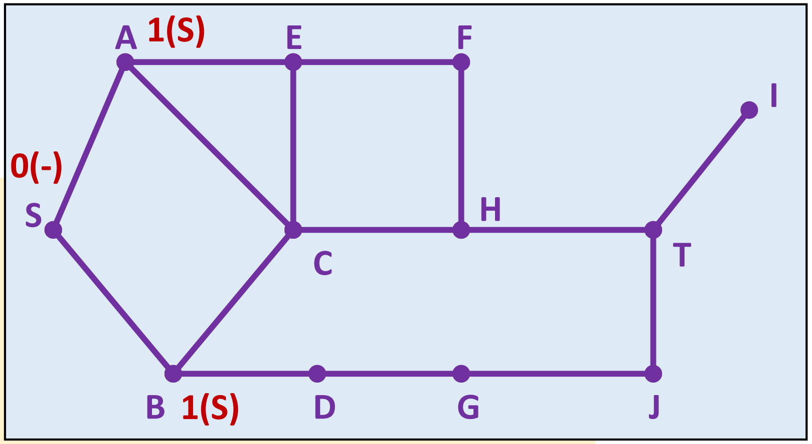
➢ **Example:** Apply the BFS algorithm step by step to find the shortest path from node S to node T in the multigraph shown on the figure.



# Shortest Path & Distance…

## BFS Algorithm…

### Solution:…

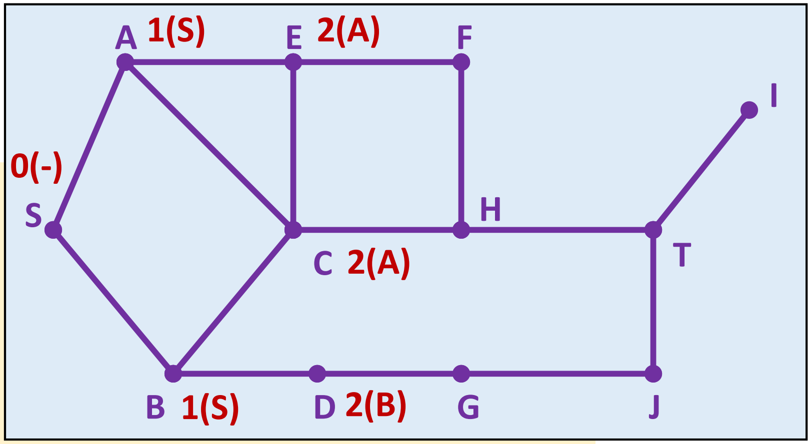



```
STEP 1 (label S)
    a) Give S label 0
    b) K = {S}, k ← 0
STEP 2 (label nodes)
   REPEAT (1)
     STEP 2.1 (increase label)
          k ← 1
     STEP 2.2 (expand labeling)
          WHILE (1) (L has a node V labeled as 0 adj. to a node A not in L)
             a) Give A label 1
             b) Assign prior of A as S
             c) L = {S,A}
           ----------------------------------------------------------------------
          WHILE (2) (L has a node V labeled as 0 adj. to a node B not in L)
             a) Give A label 1
             b) Assign prior of A as S
             c) L = {S,A}
          ----------------------------------------------------------------------
          WHILE (3) (L has not a node labeled as 0 adj. to a node not in L)
          END WHILE
          ----------------------------------------------------------------------
```

# Shortest Path & Distance…

## BFS Algorithm…

### Solution:…

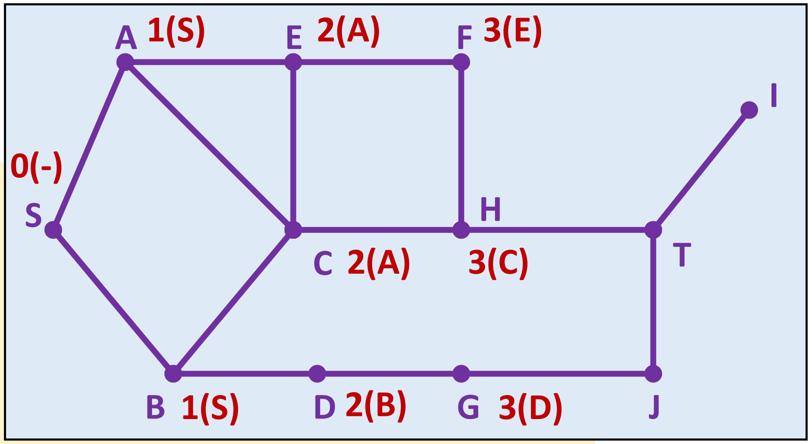



```
REPEAT (2)
  STEP 2.1 (increase label) k ← 2
  STEP 2.2 (expand labeling)
       WHILE (1) (L has a node A labeled as 1
                  adj. to a node C not in L)
          a) Give C label 2
          b) Assign prior of C as A
          c) L = {S,A,B,C}
       ----------------------------------------------------------------------
       WHILE (2) (L has a node A labeled as 1 adj. to a node E not in L)
          a) Give E label 2
          b) Assign prior of E as A
          c) L = {S,A,B,C,E}
       ----------------------------------------------------------------------
       WHILE (3) (L has a node B labeled as 1 adj. to a node D not in L)
          a) Give D label 2
          b) Assign prior of D as B
          c) L = {S,A,B,C,E,D}
       ----------------------------------------------------------------------
       WHILE (4) (L has not a node labeled as 1 adj. to a node not in L)
       END WHILE
       ----------------------------------------------------------------------
```

# Shortest Path & Distance…

## BFS Algorithm…

### Solution:…

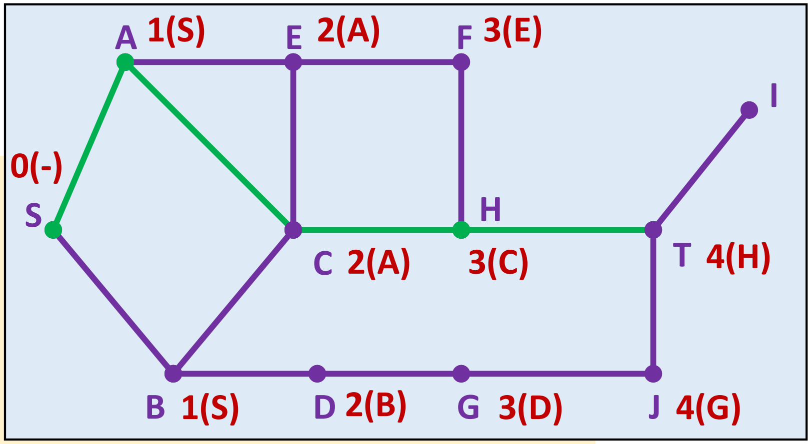



```
REPEAT (4)
  STEP 2.1 (increase label) k ← 4
  STEP 2.2 (expand labeling)
       WHILE (1) (L has a node C labeled as 2
                  adj. to a node H not in L)
          a) Give H label 3
          b) Assign prior of H as C
          c) L = {S,A,B,C,E,D,H}
       ----------------------------------------------------------------------
       WHILE (2) (L has a node E labeled as 2 adj. to a node F not in L)
          a) Give F label 3
          b) Assign prior of F as E
          c) L = {S,A,B,C,E,D,H,F}
       ----------------------------------------------------------------------
       WHILE (3) (L has a node G labeled as 2 adj. to a node D not in L)
          a) Give G label 3
          b) Assign prior of G as D
          c) L = {S,A,B,C,E,D,H,F,G}
       ----------------------------------------------------------------------
       WHILE (4) (L has not a node labeled as 2 adj. to a node not in L)
       END WHILE
       -----------------------------------------------------------------------
```

# Shortest Path & Distance…

## BFS Algorithm…

### Solution:…




```
    REPEAT (3)
      STEP 2.1 (increase label) k ← 3
      STEP 2.2 (expand labeling)
           WHILE (1) (L has a node H labeled as 3
                      adj. to a node T not in L)
              a) Give T label 4
              b) Assign prior of T as H
              c) L = {S,A,B,C,E,D,H,F,GT}
           ----------------------------------------------------------------------
           WHILE (2) (L has a node G labeled as 3 adj. to a node J not in L)
              a) Give J label 4
              b) Assign prior of J as G
              c) L = {S,A,B,C,E,D,H,F,G,T,J}
           ----------------------------------------------------------------------
           WHILE (4) (L has not a node labeled as 3 adj. to a node not in L)
           END WHILE
           -----------------------------------------------------------------------
    UNTIL (T is inside L)
STEP 3 (create the shortest path to T)
    IF (T is in L)
           prior of T is H, prior of H is C, prior of C is A and prior of A is S
           distance from S to T is 4
    END IF
```

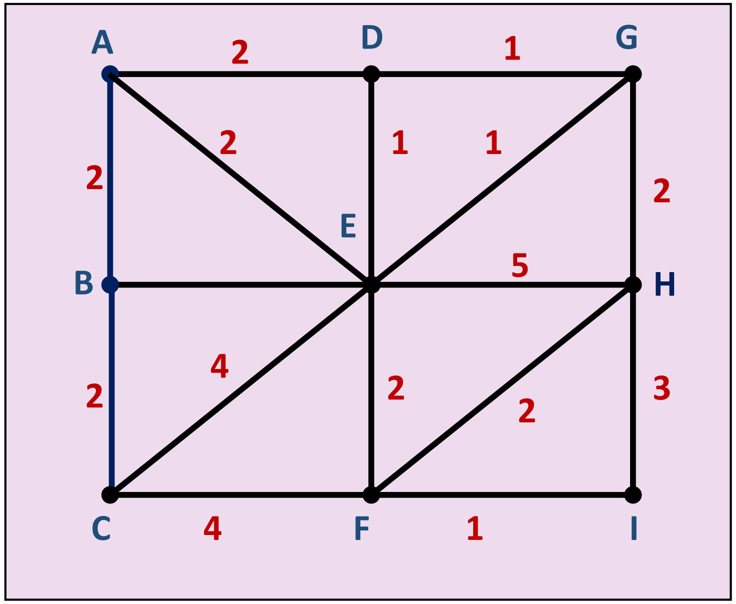
# Shortest Path & Distance…

- **Weighted Graphs**
  - often, when graphs are used to describe relationships between objects, each edge is assigned a number.
  - for example, if a graph is a graph showing city roads, distances are written on the edges.
  - a weighted graph is a graph with a number called weight on each side.
  - the weight of a road is the sum of the weights of the sides on that relevant road.

# Shortest Path & Distance...

## ➢ Weighted Graphs...

- ➢ The weight of A-E-H-I path is 2+5+3 = 10
- ➢ The weight of CFEDG is 4+2+1+1=8



# Shortest Path & Distance…

- **Dijkstra Algorithm**
  - one of the algorithm that finds the shortest path between two vertices in a weighted graph
  - a greedy algorithm discovered by the Dutch mathematician Edsger Dijkstra in 1959
  - solves the shortest path problem in undirected weighted graphs where all the weights are positive.
  - easy to adapt it to solve shortest-path problems in directed graphs.

# Shortest Path & Distance…

- **Dijkstra Algorithm…**
  - let G be a weighted graph with more than one node with all positive weights.
  - the algorithm finds the shortest path and distance from node S to another node in G.
  - in the algorithm, P represents the set of labeled nodes.
  - the prior of node A is the node used to label A in P.
  - $W(U,V)$ is the weight of the edge between nodes $U$ and $V$.
  - if there is no edge between $U$ and $V$, namely they are not adjacent, then $W(U,V) = \infty$.

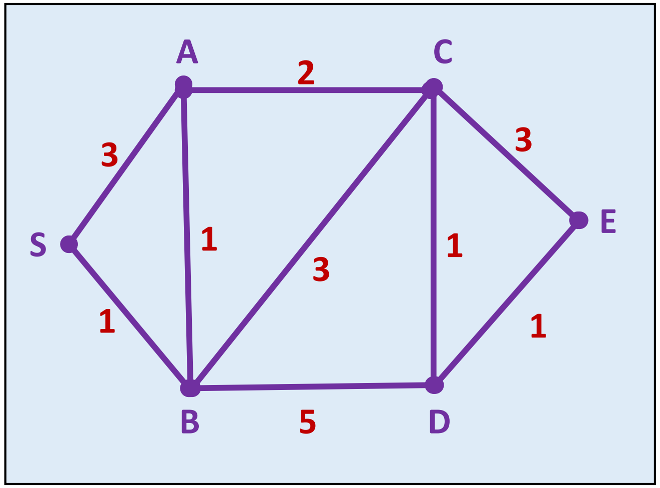
# Shortest Path & Distance…

## ➢ Dijkstra Algorithm…

```
STEP 1 ( label S )
   a) Give S label 0, S has no prior
   b) P ={S}
STEP 2 ( label nodes)
   Assign the label W(S,V) to each node V that is not in P (maybe temporary) and let V have
   prior of S
STEP 3 ( expand P and review)
   REPEAT
      STEP 3.1 ( make another label fixed)
          Add the smallest labeled node U not in P to P. (If there is more than one such
          node, choose one arbitrarily.)
      STEP 3.2 (revise temporary labels)
          For every node X adjacent to U that is not in P, replace the label of X with
          the old label of X and the label of U, whichever is less than the sum of W(U,X).
          If the label of X has changed, make the node of U the prior of X.
    UNTIL (P has all the nodes in G)
STEP 4 (find the shortest path and distance)
    IF (the label of Y is ∞)
          there is no path
    ELSE
          the path from S to Y is the prior of Y, prior of the next and etc. is obtained
          by going in reverse order from Y to S.
    END IF
```

# Shortest Path & Distance…

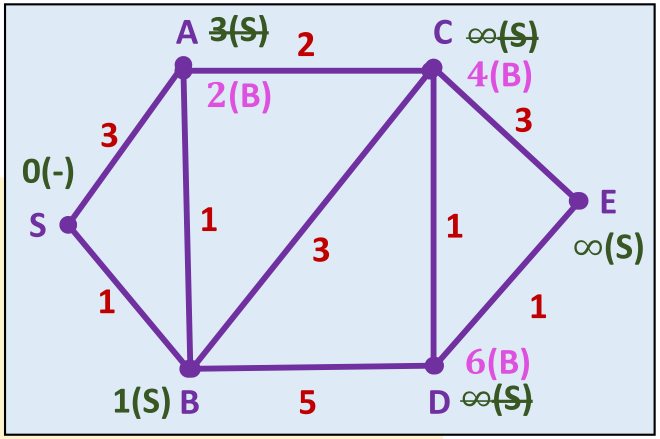
➢ Example: Apply the Dijkstra algorithm step by step to find the shortest path from node S node to E node in the multigraph given on the figure.



# Shortest Path & Distance…

## Dijkstra Algorithm

### Solution:…

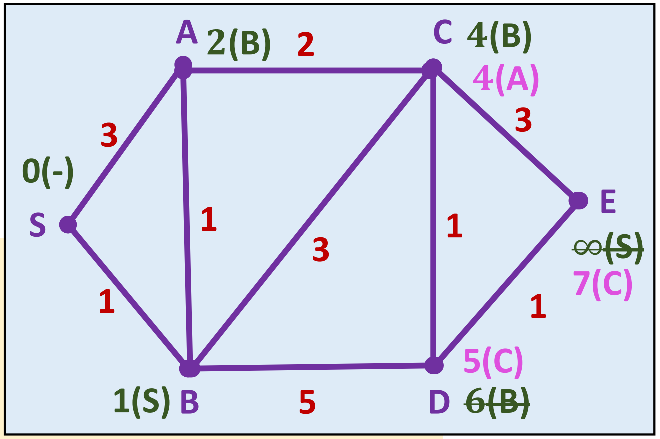



```
STEP 1 (label S)
    a) Give S label 0, S has no prior
    b) P = {S}
STEP 2 (label nodes)
    Assign W(S,V) to each node V that is not in P
    and let V the prior of S
    A ← 3(S), B ← 1(S), C ← ∞(S), D ← ∞(S), E ← ∞(S),
STEP 3 (expand P and review)
   REPEAT (1)
     STEP 3.1 (make another label fixed)
          the smallest node not in P is B, P={S,B}
     STEP 3.2 (revise temporary labels)
          The adjacent nodes of B are A,C and D
          Node X     Old label  Label of B + W(B,X)   Minimum    New label
          ----------------------------------------------------------------
          A             3            1+1=2               2          2(B)
          C             ∞            1+3=4               4          4(B)
          D             ∞            1+5=6               6          6(B)
          ----------------------------------------------------------------
```

# Shortest Path & Distance…

## Dijkstra Algorithm…

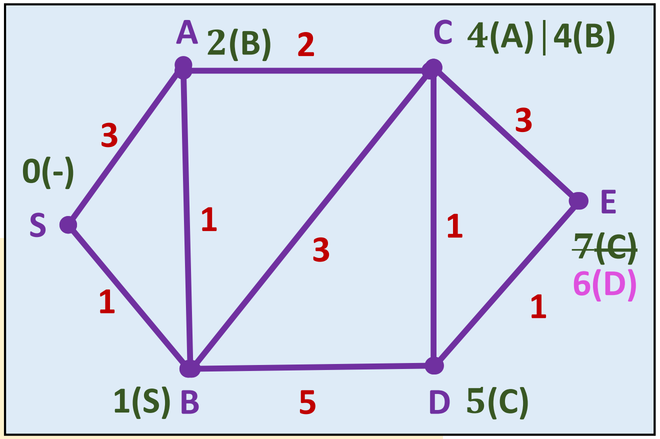
### Solution:…




```
REPEAT (2)
   STEP 3.1 (make another label fixed)
         the smallest node not in P is A, P={S,B,A}
   STEP 3.2 (revise temporary labels)
         The adjacent nodes of A is C
         Node X     Old label  Label of B + W(B,X)   Minimum    New label
         ----------------------------------------------------------------
         C             4            2+2=4              4          4(B)|4(A)
         ----------------------------------------------------------------
REPEAT (3)
   STEP 3.1 (make another label fixed)
         the smallest node not in P is C, P={S,B,A,C}
   STEP 3.2 (revise temporary labels)
         The adjacent nodes of C are D and E
         Node X     Old label  Label of B + W(B,X)   Minimum    New label
         ----------------------------------------------------------------
         D             6            4+1=5              5          5(C)
         E             ∞            4+3=7              7          7(C)
         ----------------------------------------------------------------
```

# Shortest Path & Distance…

- **Dijkstra Algorithm…**
  - Solution:…



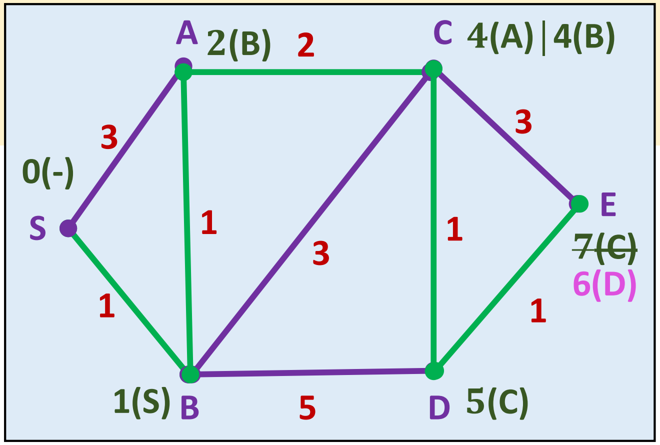

```
REPEAT (4)
   STEP 3.1 (make another label fixed)
        the smallest node not in P is D, P={S,B,A,D}
   STEP 3.2 (revise temporary labels)
        The adjacent nodes of D is E
        Node X     Old label  Label of B + W(B,X)   Minimum    New label
        ----------------------------------------------------------------
        E             7           5+1=6                6          6(D)
        ----------------------------------------------------------------
REPEAT (5)
   STEP 3.1 (make another label fixed)
        the smallest node not in P is E, P={S,B,A,C,E}
   STEP 3.2 (revise temporary labels)
        There is no adjacent node of E
        ----------------------------------------------------------------
UNTIL (all nodes are in P)
```

# Shortest Path & Distance…

## ➢ Dijkstra Algorithm…

### ➢ Solution:…

```
STEP 4 (find the shortest path and distance)
    IF (the label of Y is ∞)
    ELSE
           the prior of E is D, the prior of S is C, the prior of C is A (if 4(A)
           is selected), the prior of A is B and prior of B is S ⟹ the path is
           S-B-A-C-D-E
           the legth from S to E is 6
    END IF
```

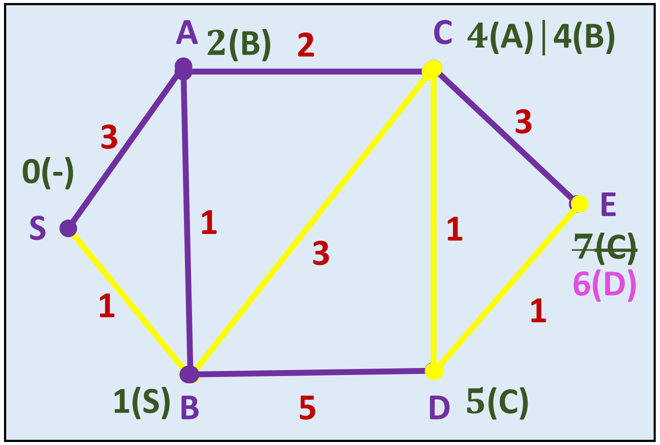


# Shortest Path & Distance…

## Dijkstra Algorithm…

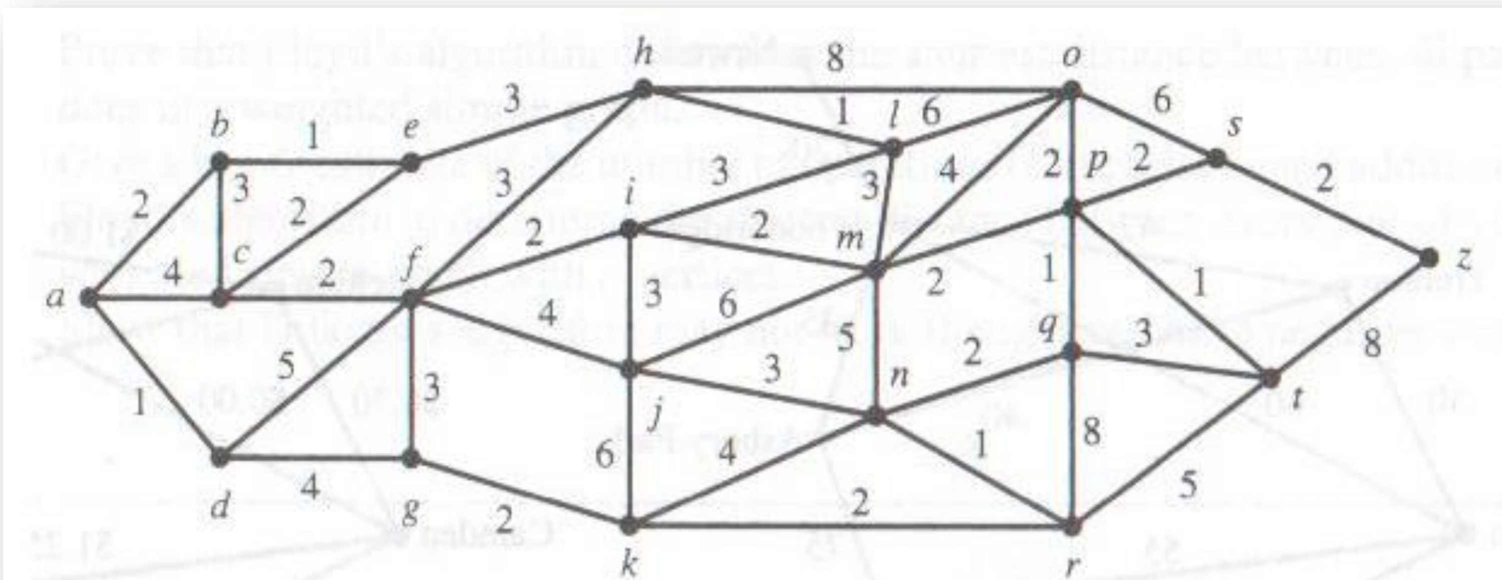
### Solution:

an alternative solution is below, if 4(B) is selected for node C ✓



# Shortest Path & Distance...

➢ **Example:** Apply the Dijkstra algorithm step by step to find one of the the shortest paths from node S node to E node in the multigraph given on the figure.



# References


- K.H. Rosen, Discrete Mathematics and Its Applications, Seventh Edition, Mc Graw Hill, 2012.
- R.P. Grimaldi, Discrete and Combinatorial Mathematcics, An Applied Introduction, Fifth Edition, Pearson, 2003.
- S.S. Epp, Discrete Mathematics with Applications, Fouth Edition, 2010.
- N. Yurtay, "Ayrık İşlemsel Yapılar" Lecture Notes, Sakarya University.