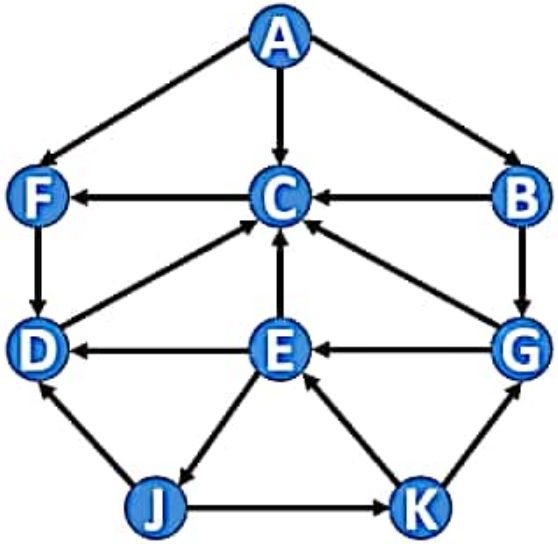
Breadth First Search (BFS)



Activate Windows
Go to Settings to activate Windows.

Algorithm

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the ready state (STATUS = 1), and change their status to the waiting state (STATUS = 2).

   [End of Step 3 loop.]
6. Exit.

Activate Windows
Go to Settings to activate Windows.

> Consider the graph G in Fig. A. (The adjacency lists of the nodes appear in Fig. B.)
> Suppose G represents the daily flights between cities of some airline, and suppose we want to fly from city A to city J with the minimum number of stops.
> In other words, we want the minimum path P from A to J (where each edge has length 1)

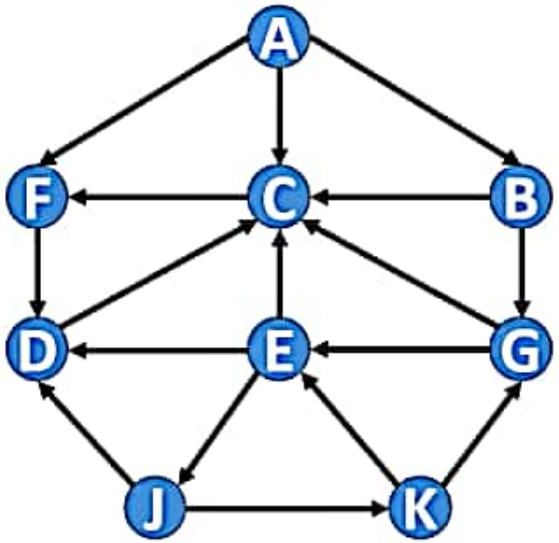


Fig. A

Adjacency list	
A	F, C, B
B	G, C
C	F
D	C
E	D, C, J
F	D
G	C, E
J	D, K
K	E, G

Fig. B

> The minimum path P can be found by using a breadth-first search beginning at city A and ending when J is encountered.
> During the execution of the search, we will also keep track of the origin of each edge by using an array ORIG together with the array QUEUE. The steps of our search follow.

Go to Settings to activate Windows.

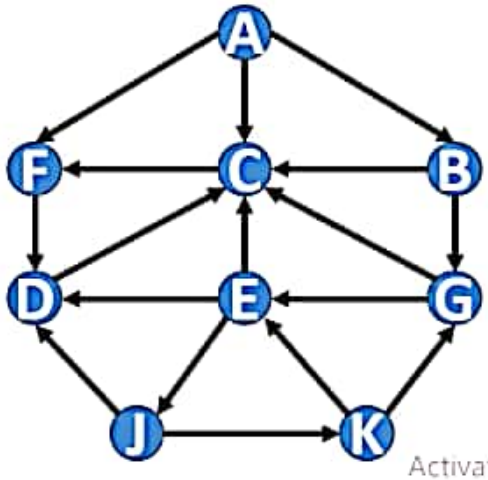
(A)

- Initially, add A to QUEUE and add NULL to ORIG as Follows:

FRONT = 1 QUEUE : A

REAR = 1 ORIG : Ø

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2).
   [End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

(B)

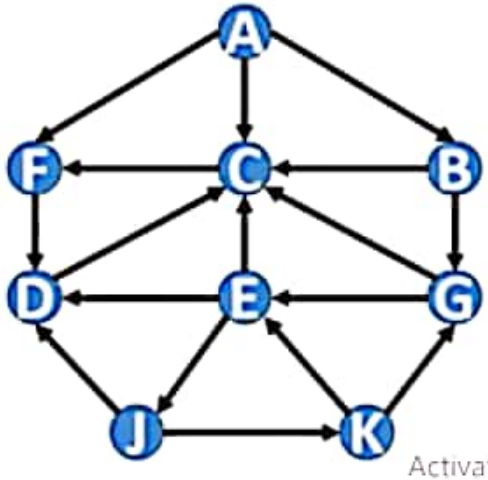
- Remove the front element A from QUEUE by setting FRONT := FRONT + 1, and add to QUEUE the neighbors of A as follows:

FRONT = 2 QUEUE : A, F, C, B

REAR = 4 ORIG : Ø, A, A, A

- Note that the origin *A* of each of the three edges is added to ORIG.

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2). [End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

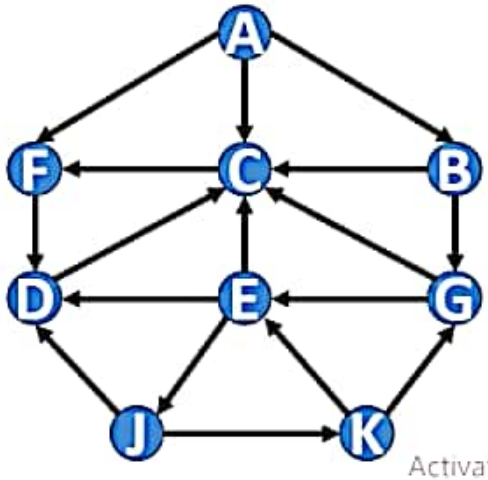
(C)

- Remove the front element F from QUEUE by setting FRONT := FRONT + 1, and add to QUEUE the neighbors of F as follows:

FRONT = 3 QUEUE : A, F, C, B, D

REAR = 5 ORIG : Ø, A, A, A, F

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2). [End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

(D)

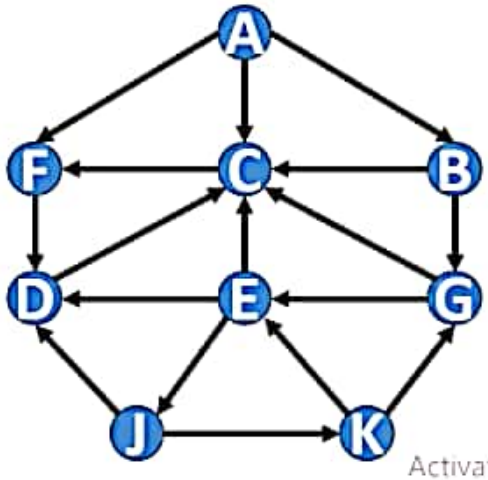
- Remove the front element C from QUEUE, and add to QUEUE the neighbors of C (which are in the ready state) as follows:

FRONT = 4 QUEUE : A, F, C, B, D

REAR = 5 ORIG : Ø, A, A, A, F

- Note that the neighbor F of C is not added to QUEUE, since F is not in the ready state (because F has already been added to QUEUE).

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2).
[End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

(E)

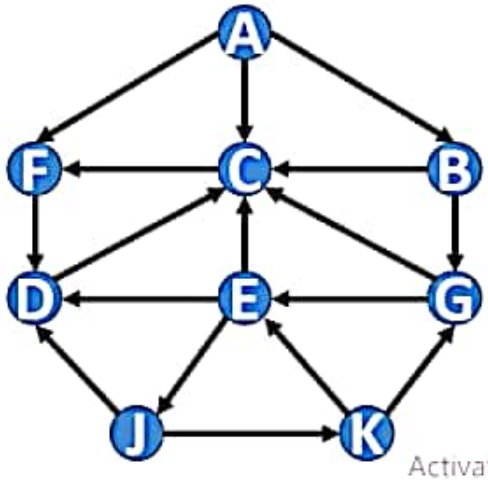
- Remove the front element B from QUEUE, and add to QUEUE the neighbors of B (the ones in the ready state) as follows:

FRONT = 5 QUEUE : A, F, C, B, D, G

REAR = 6 ORIG : Ø, A, A, A, F, B

- Note that only G is added to QUEUE, since the other neighbor, C is not in the ready state.

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2).
   [End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

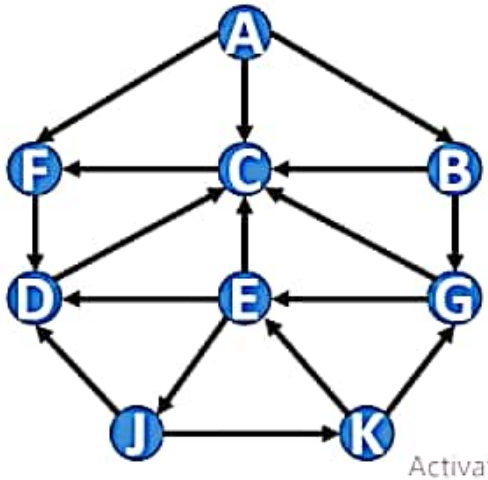
(F)

- Remove the front element D from QUEUE, and add to QUEUE the neighbors of D (the ones in the ready state) as follows:

FRONT = 6 QUEUE : A, F, C, B, D, G

REAR = 6 ORIG : Ø, A, A, A, F, B

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2).
   [End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

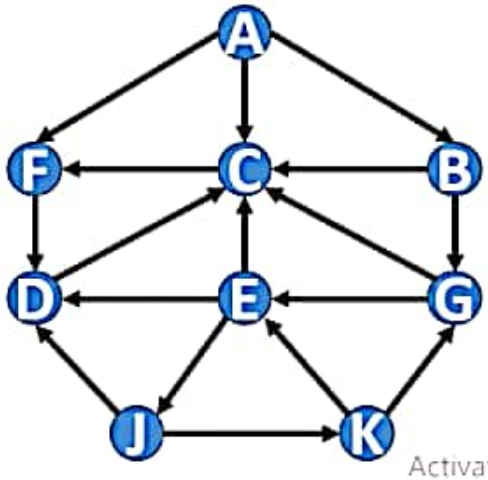
(G)

- Remove the front element G from QUEUE, and add to QUEUE the neighbors of G (the ones in the ready state) as follows:

FRONT = 7 QUEUE : A, F, C, B, D, G, E

REAR = 7 ORIG : Ø, A, A, A, F, B, G

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2).
   [End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.

(H)

- Remove the front element E from QUEUE, and add to QUEUE the neighbors of E (the ones in the ready state) as follows:

FRONT = 8 QUEUE : A, F, C, B, D, G, E, J

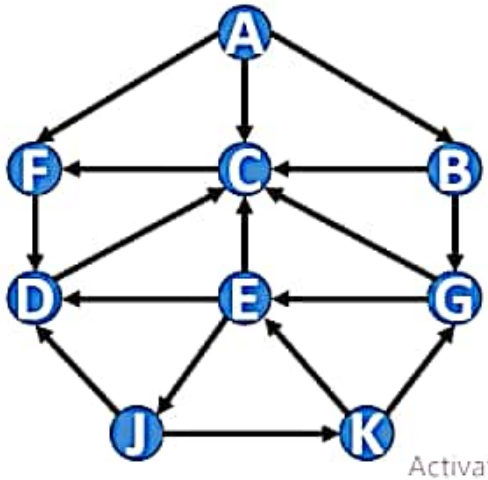
REAR = 8 ORIG : Ø, A, A, A, F, B, G, E

- We stop as soon as J is added to QUEUE, since J is our final destination. We now backtrack from J, using the array ORIG to find the path P. Thus

J<E<G<B<A

is the required path P.

1. Initialize all nodes to ready state (STATUS = 1)
2. Put the starting node A in QUEUE and change its status to waiting state (STATUS = 2)
3. Repeat Steps 4 and 5 until QUEUE is empty:
4. Remove the front node N of QUEUE. Process N and change the status of N to the processed state (STATUS = 3)
5. Add to the rear of QUEUE all the neighbors of N that are in the steady state (STATUS = 1), and change their status to the waiting state (STATUS = 2).
[End of Step 3 loop.]
6. Exit.



Activate Windows
Go to Settings to activate Windows.