Name: Marinna Ricketts-Uy

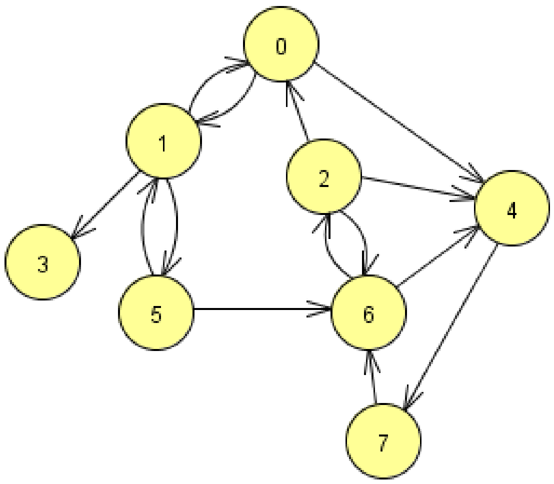
Section: 02 – Budhraja

HW #: 6

Version: A

Username: pd12778

1. Complete the BFS given below. Show each step and explain why. Start at vertex 2.

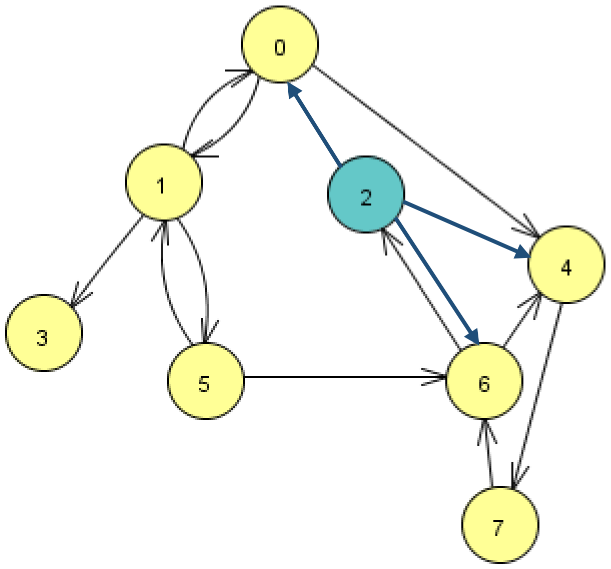


Step 1: Starting at vertex 2, we add vertex 2 to the BFS Queue. We set its parent to -1.

Parent	
0	
1	
2	-1
3	
4	
5	
6	
7	

Visited	
0	F
1	F
2	T
3	F
4	F
5	F
6	F
7	F

BFS Queue
2

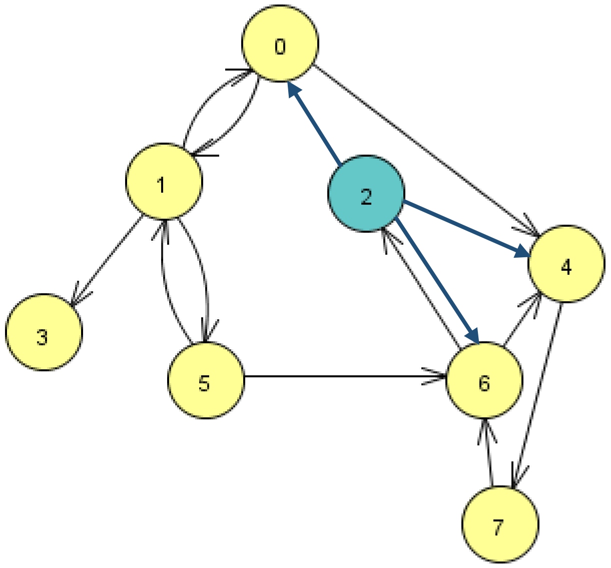


Step 2: We see that vertex 2 points to 0, 4, and 6. Add 0, 4, and 6 to the BFS Queue and pop 2 off. We set 0, 4 and 6's parents to 2 and visited to True.

Parent	
0	2
1	
2	-1
3	
4	2
5	
6	2
7	

Visited	
0	T
1	F
2	T
3	F
4	T
5	F
6	T
7	F

BFS Queue
0
4
6

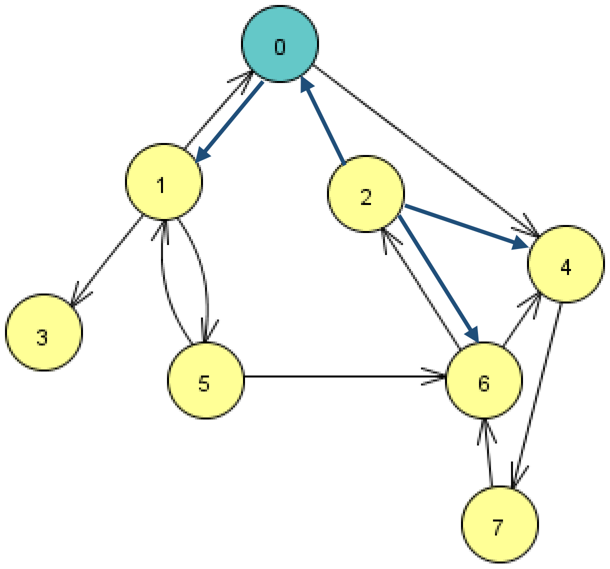


Step 3: Next we look at vertex 0.We see that 0 points to 1 and 4. Vertex 1 is added to the BFS Queue and vertex 0 is popped off. We set 1's parent to 0 and visited to True.

Parent	
0	2
1	0
2	-1
3	
4	2
5	
6	2
7	

Visited	
0	T
1	T
2	T
3	F
4	T
5	F
6	T
7	F

BFS Queue
4
6
1

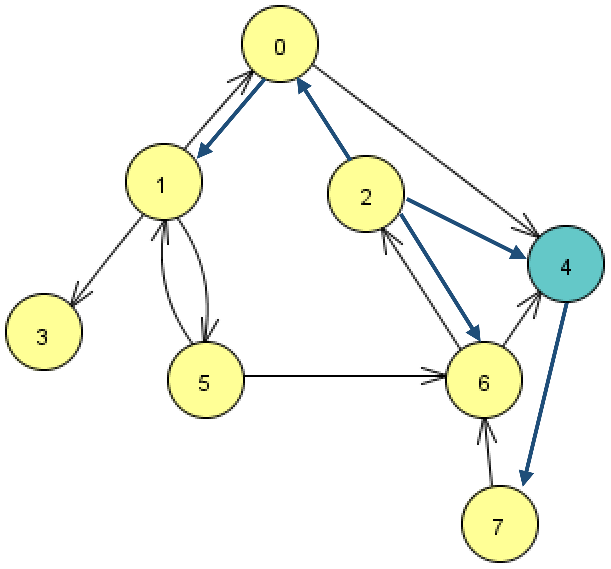


Step 4: Next we look at vertex 4. We see that 4 points to 7. Vertex 7 is added to the BFS Queue and vertex 4 is popped off. We set 7's parent to 4 and visited to True.

Parent	
0	2
1	0
2	-1
3	
4	2
5	
6	2
7	4

Visited	
0	T
1	T
2	T
3	F
4	T
5	F
6	T
7	T

BFS Queue
6
1
7

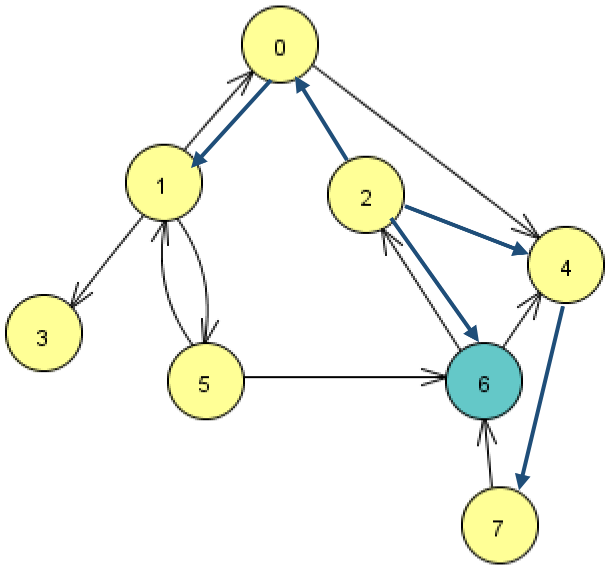


Step 5: Next we look at vertex 6. We see that 6 points to 2 and 4. They have already been taken care of. Vertex 6 is popped off the BFS Queue.

Parent	
0	2
1	0
2	-1
3	
4	2
5	
6	2
7	4

Visited	
0	T
1	T
2	T
3	F
4	T
5	F
6	T
7	T

BFS Queue
1
7

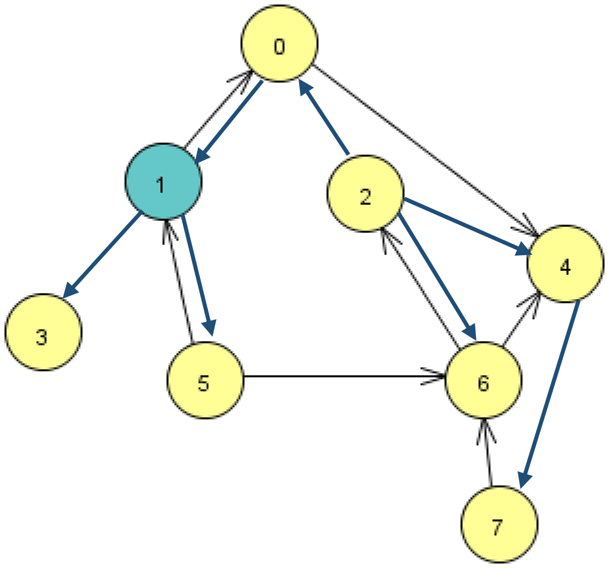


Step 6: Next we look at vertex 1. We see that 1 points to 0, 3 and 5. Vertices 3 and 5 are added to the BFS Queue and vertex 1 is popped off. We set the parents of 3 and 5 to 1 and set the visted to True.

Parent	
0	2
1	0
2	-1
3	1
4	2
5	1
6	2
7	4

Visited	
0	T
1	T
2	T
3	T
4	T
5	T
6	T
7	T

BFS Queue
7
3
5

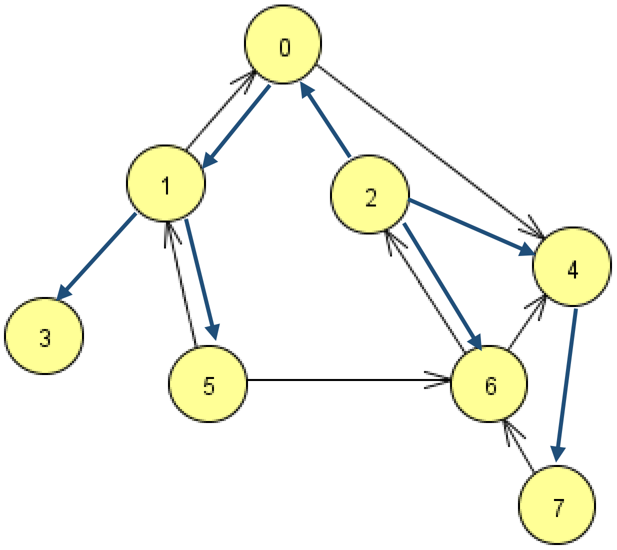


Step 7: Next we look at vertex 7. We see that 7 points to 6. Vertex 6 has already been taken care off. So we just pop vertex 7 off the BFS Queue.

Parent	
0	2
1	0
2	-1
3	1
4	2
5	1
6	2
7	4

Visited	
0	T
1	T
2	T
3	T
4	T
5	T
6	T
7	T

BFS Queue
3
5

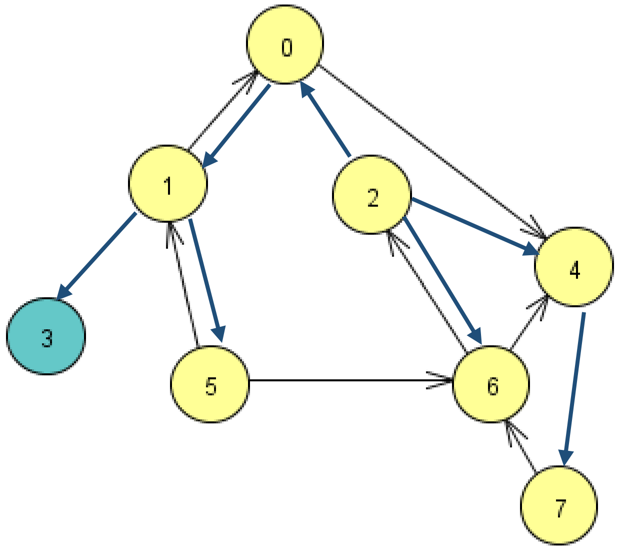


Step 8: Next we look at vertex 3. We see that 3 doesn't point anywhere. So we just pop vertex 3 off the BFS Queue.

Parent	
0	2
1	0
2	-1
3	1
4	2
5	1
6	2
7	4

Visited	
0	T
1	T
2	T
3	T
4	T
5	T
6	T
7	T

BFS Queue
5

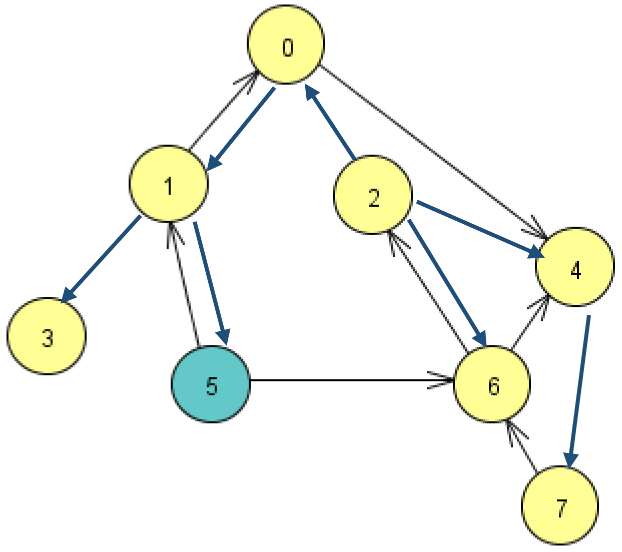


Final Step: Next we look at vertex 5. We see that 5 points to 1 and 6. Vertices 1 and 6 have already been taken care off. So we just pop vertex 5 off the BFS Queue.

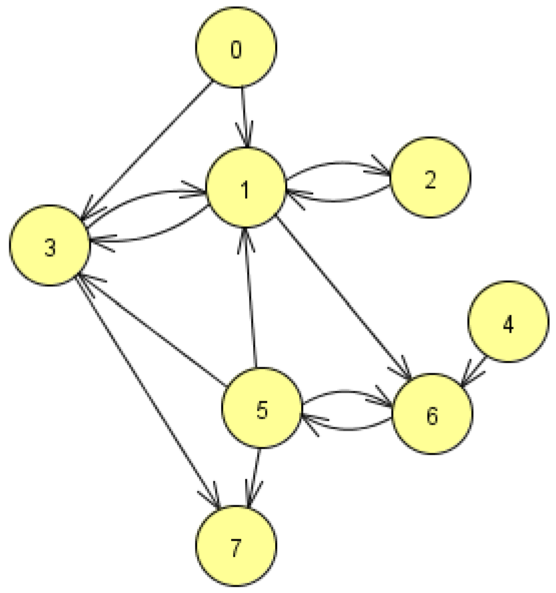
Parent	
0	2
1	0
2	-1
3	1
4	2
5	1
6	2
7	4

Visited	
0	T
1	T
2	T
3	T
4	T
5	T
6	T
7	T

BFS Queue



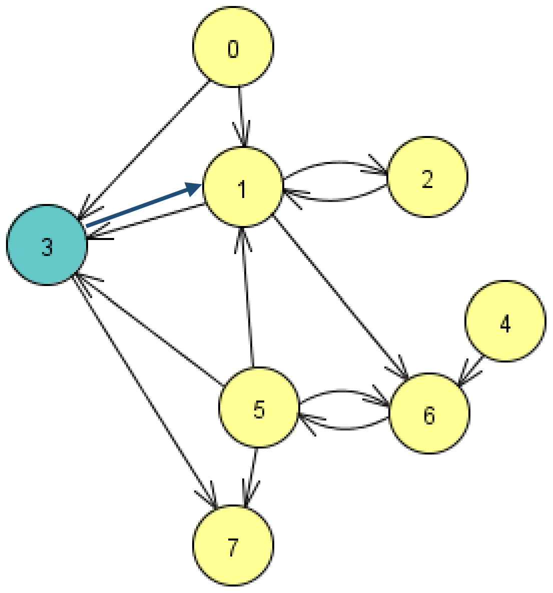
2. Complete the DFS given below. Show EACH step and explain why. Start at vertex 3.



Step 1: We start at 3. The parent of 3 is set to -1. Vertex 3 points to 1 and 7. We go to vertex 1.

Parent	
0	
1	
2	
3	-1
4	
5	
6	
7	

Visited	
0	F
1	F
2	F
3	T
4	F
5	F
6	F
7	F

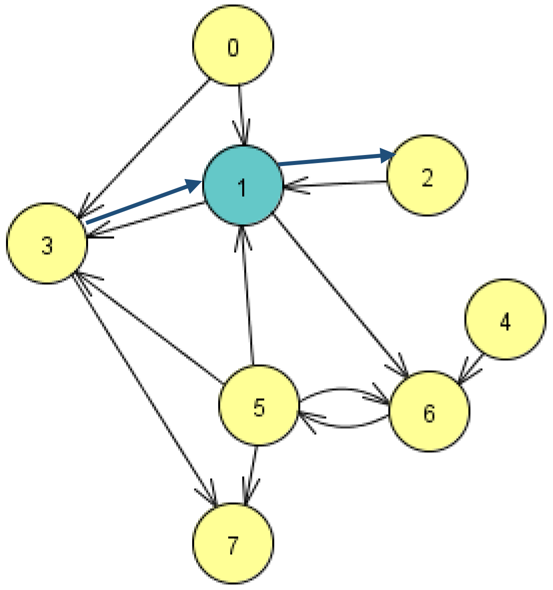


DFS(3)

Step 2: Set the parent of vertex 1 to 3. Vertex 1 points to 2, 3, and 6. So we move to vertex 2.

Parent	
0	
1	3
2	
3	-1
4	
5	
6	
7	

Visited	
0	F
1	T
2	F
3	T
4	F
5	F
6	F
7	F



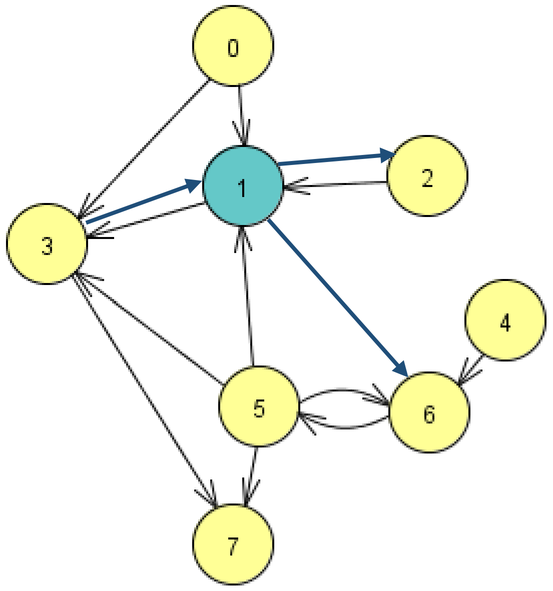
DFS(3)

DFS(1)

Step 3: Set the parent of vertex 2 to 1. Vertex 2 points back to 1. Then we move to vertex 6.

Parent	
0	
1	3
2	1
3	-1
4	
5	
6	
7	

Visited	
0	F
1	T
2	T
3	T
4	F
5	F
6	F
7	F



DFS(3)

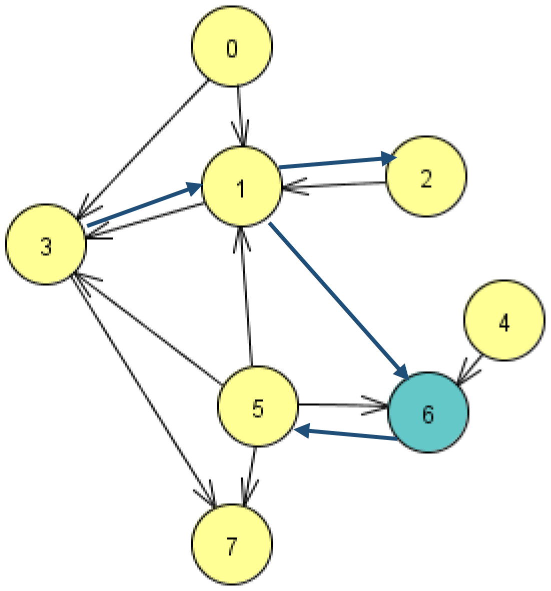
DFS(1)

DFS(2)

Step 4: Set the parent of vertex 6 to 1. Vertex 6 points to 5. Then we move to vertex 5.

Parent	
0	
1	3
2	1
3	-1
4	
5	
6	1
7	

Visited	
0	F
1	T
2	T
3	T
4	F
5	F
6	T
7	F



DFS(3)

DFS(1)

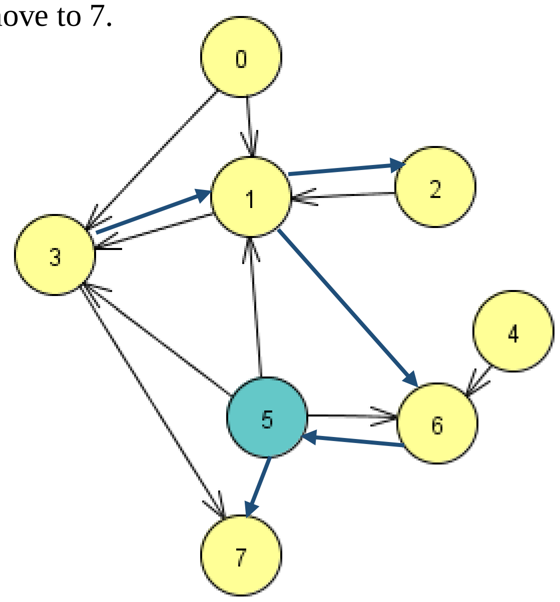
DFS(2)

DFS(6)

Step 5: Set the parent of vertex 5 to 6. Vertex 5 points 6, 7, 3, and 1. Vertices 1, 3 and 6 have already been taken care of so we move to 7.

Parent	
0	
1	3
2	1
3	-1
4	
5	6
6	1
7	

Visited	
0	F
1	T
2	T
3	T
4	F
5	T
6	T
7	F



DFS(3)

DFS(1)

DFS(2)

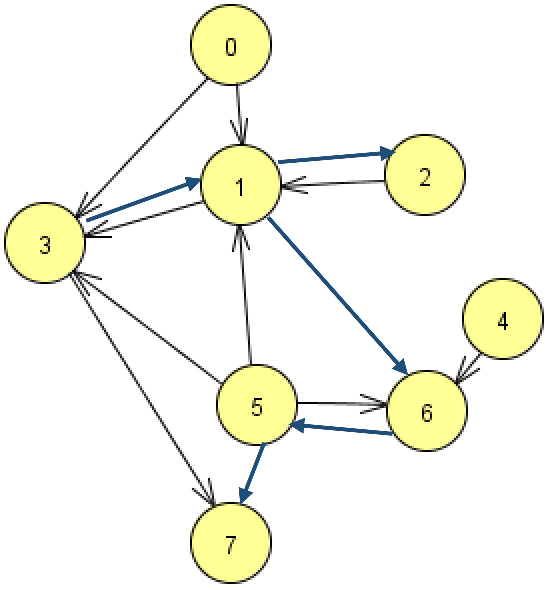
DFS(6)

DFS(5)

Final Step: Set the parent of vertex 7 to 5. Vertex 7 points to nothing.

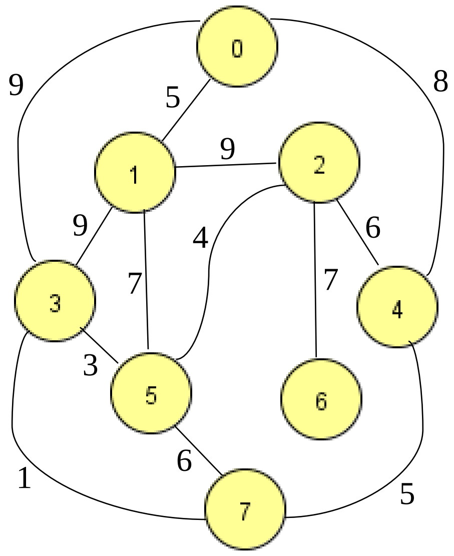
Parent	
0	
1	3
2	1
3	-1
4	
5	6
6	1
7	5

Visited	
0	F
1	T
2	T
3	T
4	F
5	T
6	T
7	T



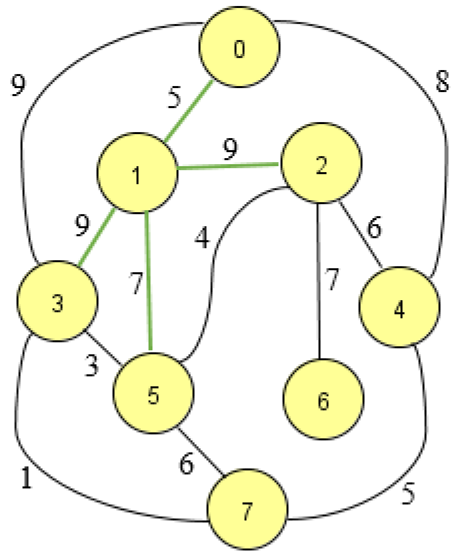
```
DFS(3)
    DFS(1)
        DFS(2)
        DFS(6)
            DFS(5)
                DFS(7)
```

3. Complete the Dijkstra's Shortest Path algorithm on the graph given below. Show EACH step and explain why. Start at vertex 1.



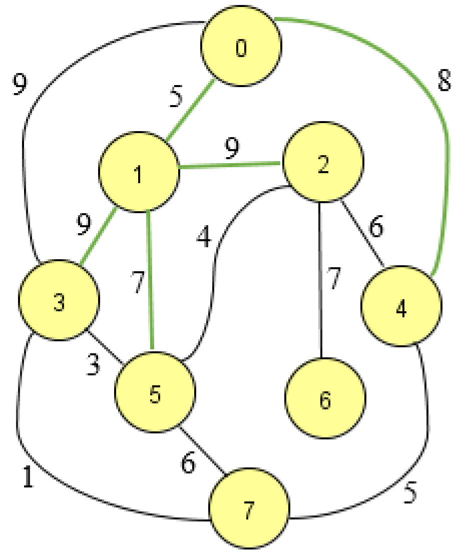
Step 1: Starting at vertex 1, change the known for vertex 1 to True and cost to 0. Then we see that 1 is connected to vertices 0, 2, 3, 5. We update the costs and paths for those vertices since the knowns are False.

Vertex	Known	Cost	Path
0	F	5	1
1	**T**	0	-1
2	F	9	1
3	F	9	1
4	F	INF	-1
5	F	7	1
6	F	INF	-1
7	F	INF	-1



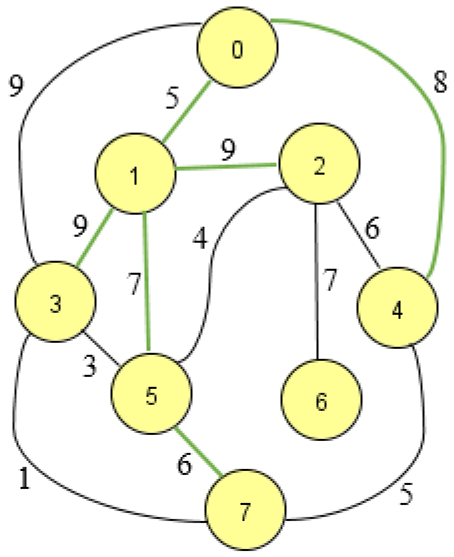
Step 2: The next vertex to look at is vertex 0 since its known is False in step 1 and its cost is the smallest, 5. So we set its known to True now. Then we see that vertex 0 is connected to 1, 3, and 4. Vertex 1 is already known, but 3 and 4 are still false, so we update their costs and paths.

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	F	9	1
3	F	9	1
4	F	13	0
5	F	7	1
6	F	INF	-1
7	F	INF	-1



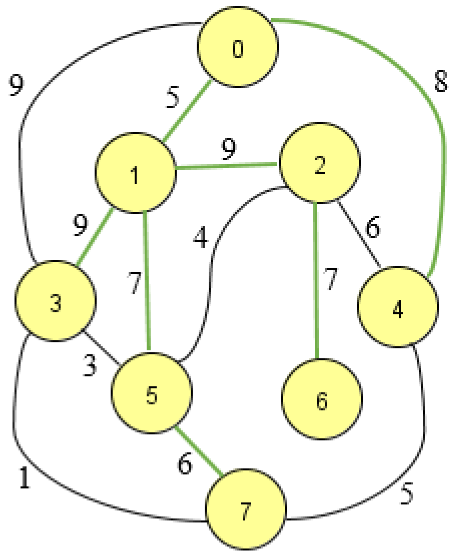
Step 3: The next vertex to look at is vertex 5 since its known is False in the previous step and its cost is the smallest, 7. So we set its known to True now. Then we see that vertex 5 is connected to 1, 2, 3, and 7. Vertex 1 is already known, but 2, 3, and 7 are still false, so we update their costs and paths.

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	F	9	1
3	F	9	1
4	F	13	0
5	**T**	7	1
6	F	INF	-1
7	F	13	5



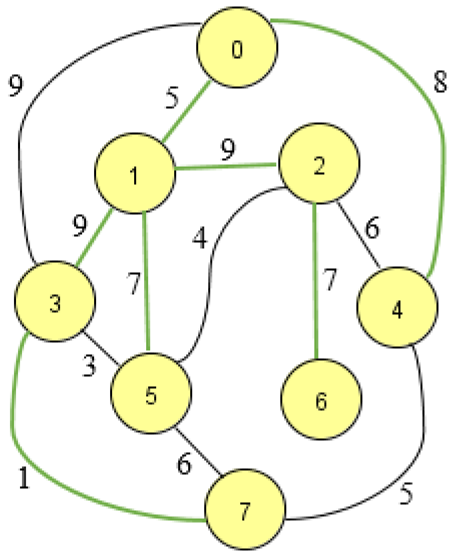
Step 4: The next vertex to look at is vertex 2 since its known is False in the previous step and its cost is the smallest, 9. So we set its known to True now. Then we see that vertex 2 is connected to 1, 4, 5 and 6. Vertices 1 and 5 are already known, so we just update vertex 4 and 6.

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	**T**	9	1
3	F	9	1
4	F	13	0
5	**T**	7	1
6	F	16	2
7	F	13	5



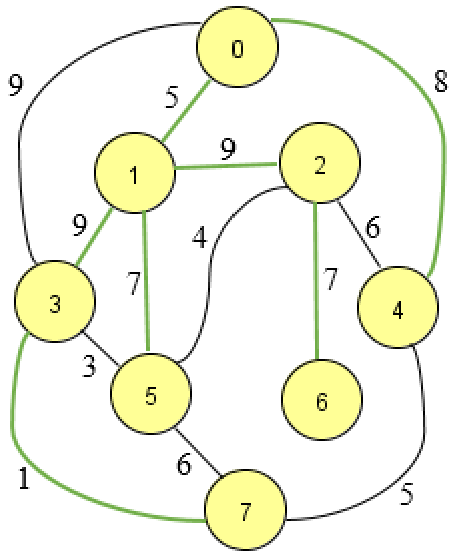
Step 5: The next vertex to look at is vertex 3 since its known is False in the previous step and its cost is the smallest, 9. So we set its known to True now. Then we see that vertex 3 is connected to 0, 1, 5, and 7. Vertices 0, 1, and 5 are already known. So we just update vertex 7.

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	**T**	9	1
3	**T**	9	1
4	F	13	0
5	**T**	7	1
6	F	16	2
7	F	10	3



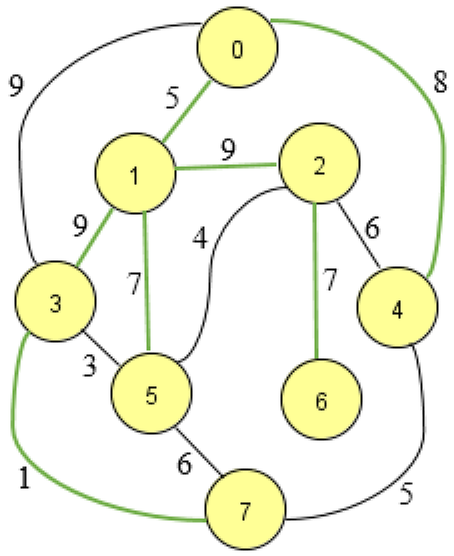
Step 6: The next vertex to look at is vertex 7, since its known is False in the previous step its cost is the smallest, 10. So we set its known to True now. Then we see that vertex 7 is connected to 3, 4 and 5. Vertices 3 and 5 are already known. So we look at 4 to update.

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	**T**	9	1
3	**T**	9	1
4	F	13	0
5	**T**	7	1
6	F	16	2
7	**T**	10	3



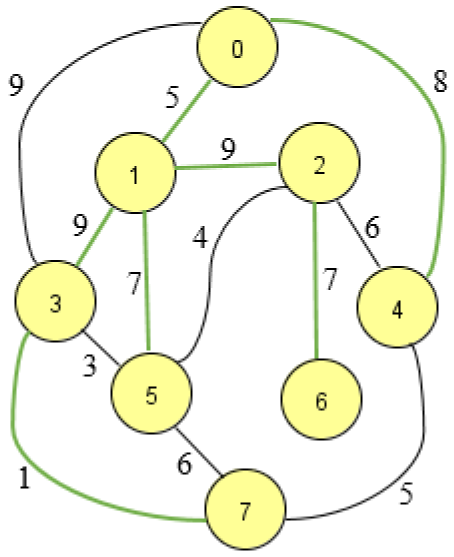
Step 7: The next vertex to look at is vertex 4 since its known is False in the previous step and its cost is the smallest, 13. So we set its known to True now. Then we see that vertex 4 is connected to 0 and 7. Vertices 0 and 7 are already known so no changes occur.

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	**T**	9	1
3	**T**	9	1
4	**T**	13	0
5	**T**	7	1
6	F	16	2
7	**T**	10	3

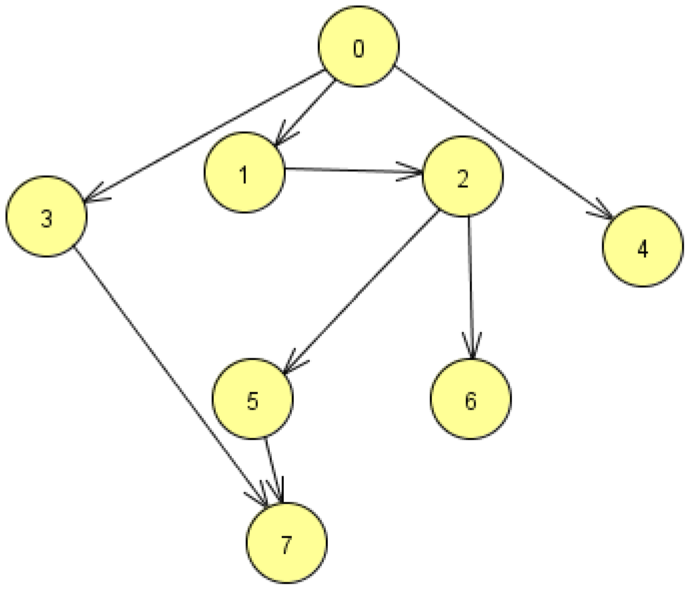


Final Step: The last vertex to look at is vertex 6. So we set its known to True now. Then we see that vertex 6 is connected to 2. Vertex 2 is already known, so no changes are made. All vertices are known so we are DONE!

Vertex	Known	Cost	Path
0	**T**	5	1
1	**T**	0	-1
2	**T**	9	1
3	**T**	9	1
4	**T**	13	0
5	**T**	7	1
6	**T**	16	2
7	**T**	10	3



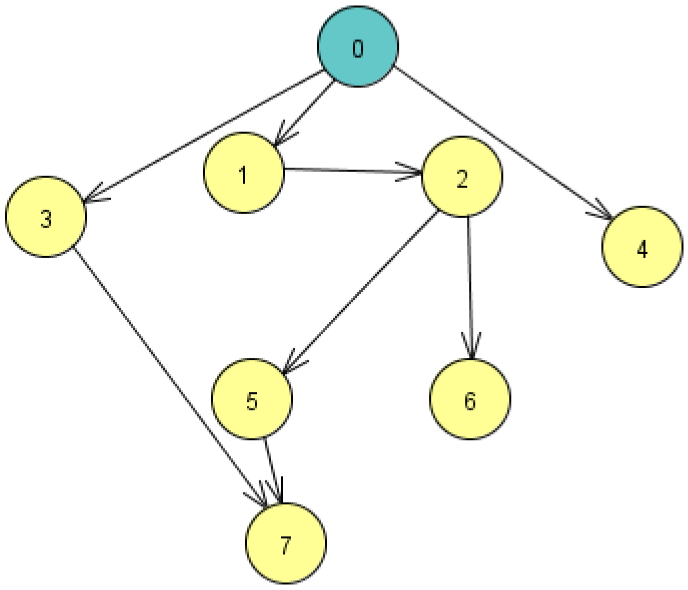
4. Complete the Topological sort on the graph given below. Show EACH step and call stack and explain why.



Step 1: Start at vertex 0 because it is the lowest value. We see that vertex 0 points to 1, 3 and 4.

DFS(0)

Topological Order

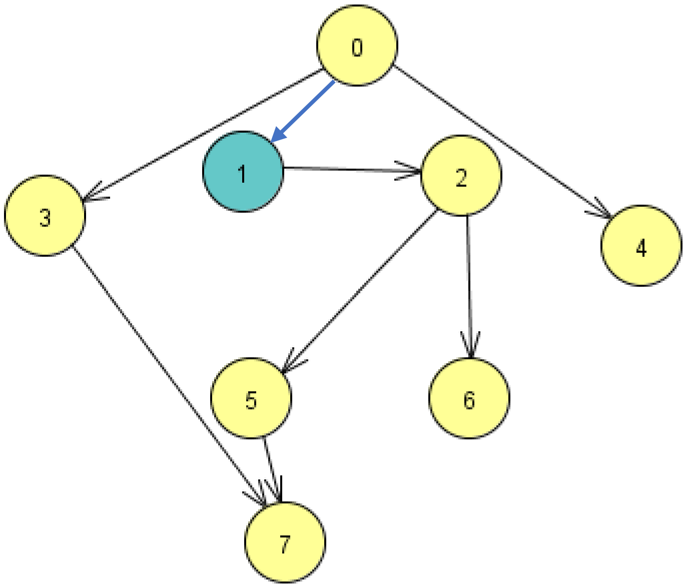


Step 2: We move the current node to vertex 1 because it is the lowest value that vertex 0 points to and vertex 1 is added to the stack. We see that vertex 1 points to 2.

DFS(0)

DFS(1)

Topological Order



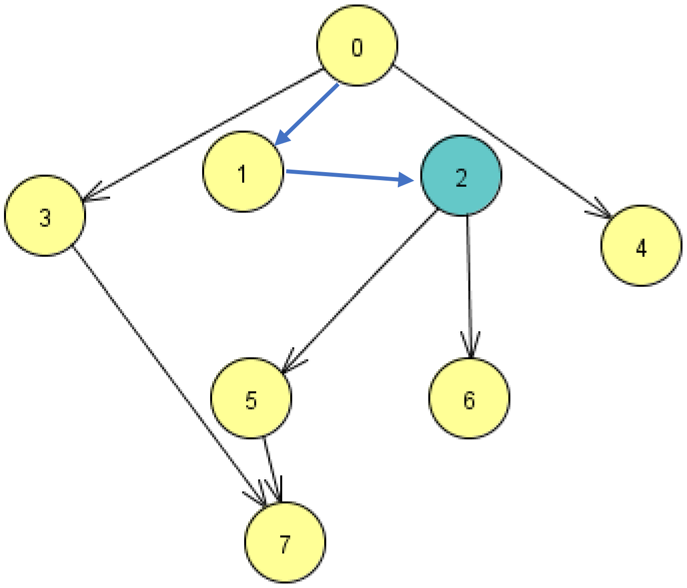
Step 3: We move the current node to vertex 2 since that's the only node that vertex 1 points to. Vertex 2 is added to the stack. Vertex 2 points to 5 and 6.

DFS(0)

DFS(1)

DFS(2)

Topological Order



Step 4: We move the current node to vertex 5 since it's the lowest value that vertex 2 points to. Vertex 5 is added to the stack. Vertex 5 points to only vertex 7.

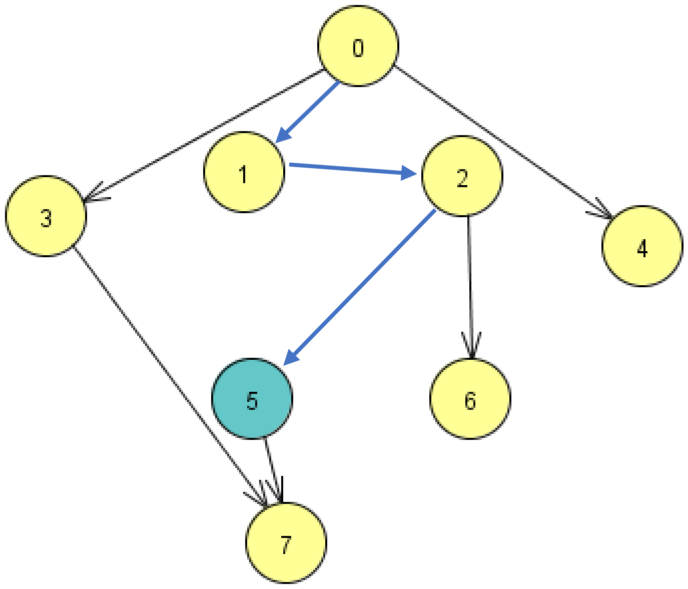
DFS(0)

DFS(1)

DFS(2)

DFS(5)

Topological Order



Step 5: We move the current node to vertex 7 since it's the only thing vertex 5 points to.Vertex 7 is added to the stack. Since vertex 7 doesn't point to anything, it is added to the Topological Order.

DFS(0)

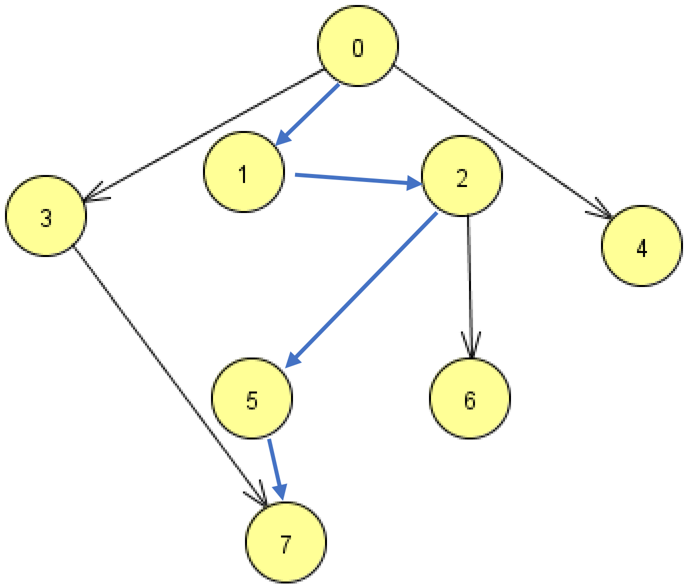
DFS(1)

DFS(2)

DFS(5)

DFS(7)

Topological Order
7



Step 6: Since vertex 7 doesn't point to anything, we move up the stack. We move up to vertex 5 and see that it doesn't point to anything else, so it's added to the Topological Order. Then we move up to vertex 2 and see that it points to vertex 6.

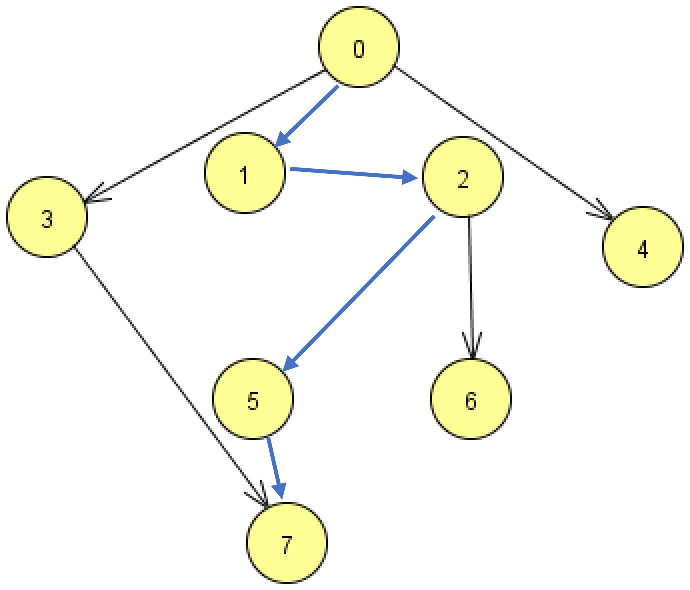
DFS(0)

DFS(1)

DFS(2)

DFS(5)

DFS(7)



Topological Order
5
7

Step 7: We set the current node to vertex 6. Vertex 6 is added to the stack. We see that vertex 6 doesn't point to anything, so vertex 6 is added to the Topological Order. We move back up to vertex 2 and see that it doesn't point to anything else, so it's added to the Topological Order. Same thing occurs with vertex 1. Then we move up to vertex 0.

DFS(0)

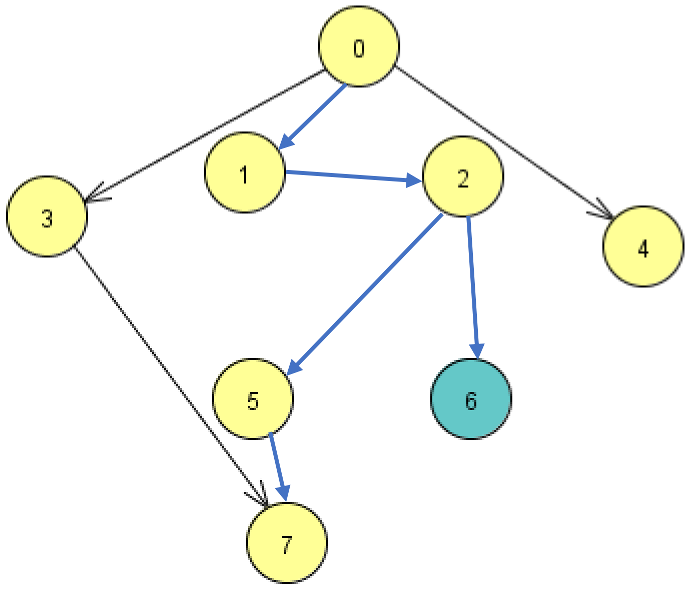
DFS(1)

DFS(2)

DFS(5)

DFS(7)

DFS(6)



Topological Order
1
2
6
5
7

Step 8: We set the current node to vertex 0. We see that vertex 0 points to 3 and 4. We move to vertex 3 since it is the lowest value.

DFS(0)

DFS(1)

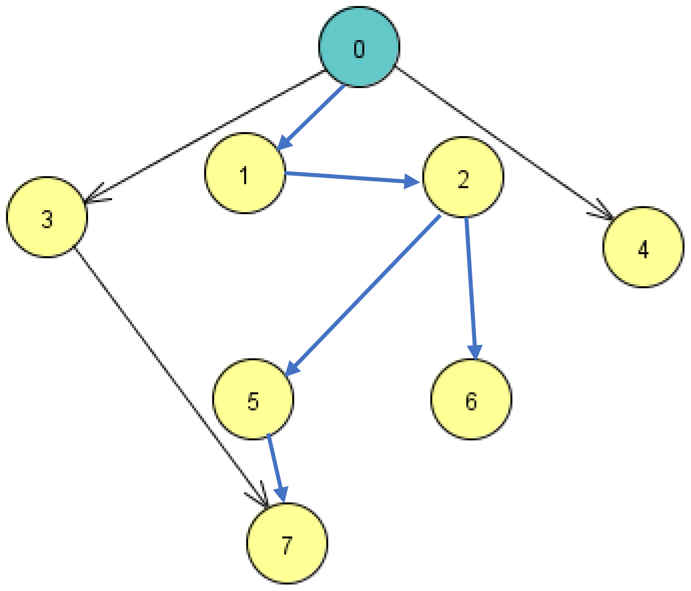
DFS(2)

DFS(5)

DFS(7)

DFS(6)

Topological Order
1
2
6
5
7



Step 9: We set the current node to vertex 3. Vertex 3 is added to the stack. Vertex 3 points to vertex 7, but it has already been taken care of. So vertex 3 is added to the Topological Order. We move back up to vertex 0.

DFS(0)

DFS(1)

DFS(2)

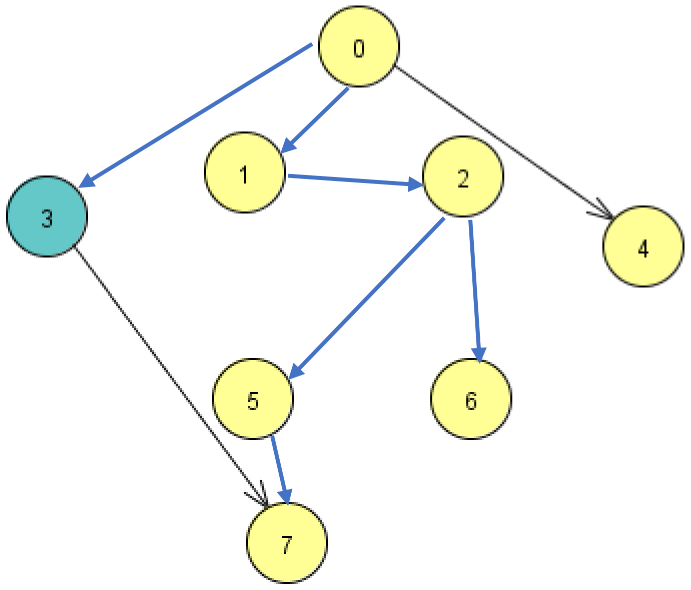
DFS(5)

DFS(7)

DFS(6)

DFS(3)

Topological Order
3
1
2
6
5
7



Step 10: We set the current node to vertex 0. Vertex 0 points to vertex 4. We move to vertex 4.

DFS(0)

    DFS(1)

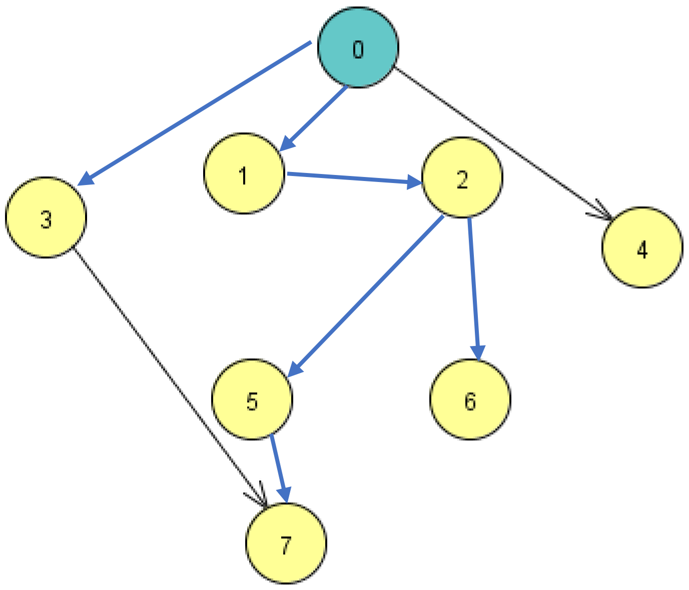
        DFS(2)

            DFS(5)

                DFS(7)

            DFS(6)

    DFS(3)



Topological Order
3
1
2
6
5
7

Step 10: We set the current node to vertex 4. Vertex 4 is added to the stack. Vertex 4 doesn’t point to anything, so its added to the Topological Order. We move back up to vertex 0. Vertex 0 points to everything that has already been taken care of. Vertex 0 is added to the Topological Order.

DFS(0)

DFS(1)

DFS(2)

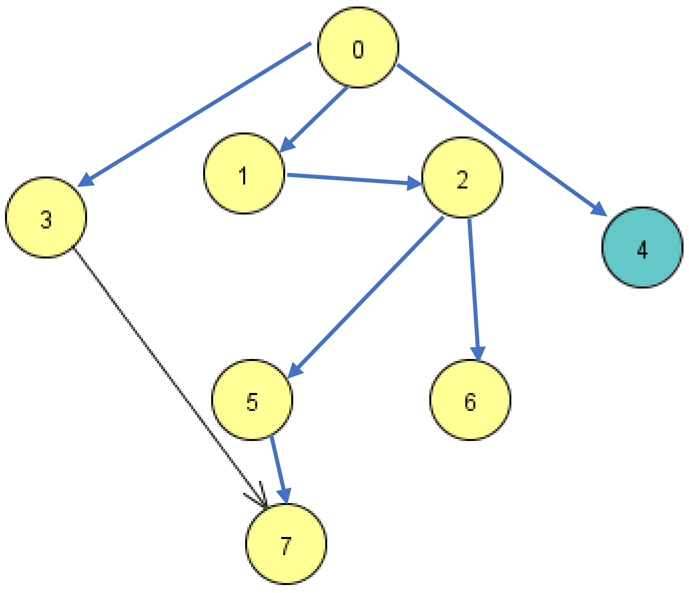
DFS(5)

DFS(7)

DFS(6)

DFS(3)

DFS(4)



Topological Order
0
4
3
1
2
6
5
7

Final Step: The final step is to make sure the Topological Sort is correct. So do this, we take the Topological Order and draw the lines that connect each vertex. If the lines are all in one direction, then it was done correctly.