**QUESTION 1**

**Initial)** Unknown: A(∞), B(∞), C(∞), D(∞), E(∞), F(∞), G(∞), S(∞)

**Step I)**

Known: S(S,0)

Unknown: A(S,3), B(S,2), C(∞), D(∞), E(∞), F(∞), G(∞)

**Step II)**

Known: S(S,0), B(S,2)

Unknown: A(S,3), C(∞), D(B,3), E(∞), F(∞), G(∞)

**Step III)**

Known: S(S,0), B(S,2), A(S,3)

Unknown: C(∞), D(B,3), E(A,5), F(∞), G(∞)

**Step IV)**

Known: S(S,0), B(S,2), A(S,3), D(B,3)

Unknown: C(∞), E(D,4), F(D,5), G(D,5)

**Step V)**

Known: S(S,0), B(S,2), A(S,3), D(B,3), E(D,4)

Unknown: C(∞), F(D,5), G(D,5)

**Step VI)**

Known: S(S,0), B(S,2), A(S,3), D(B,3), E(D,4), F(D,5)

Unknown: C(F,6), G(D,5)

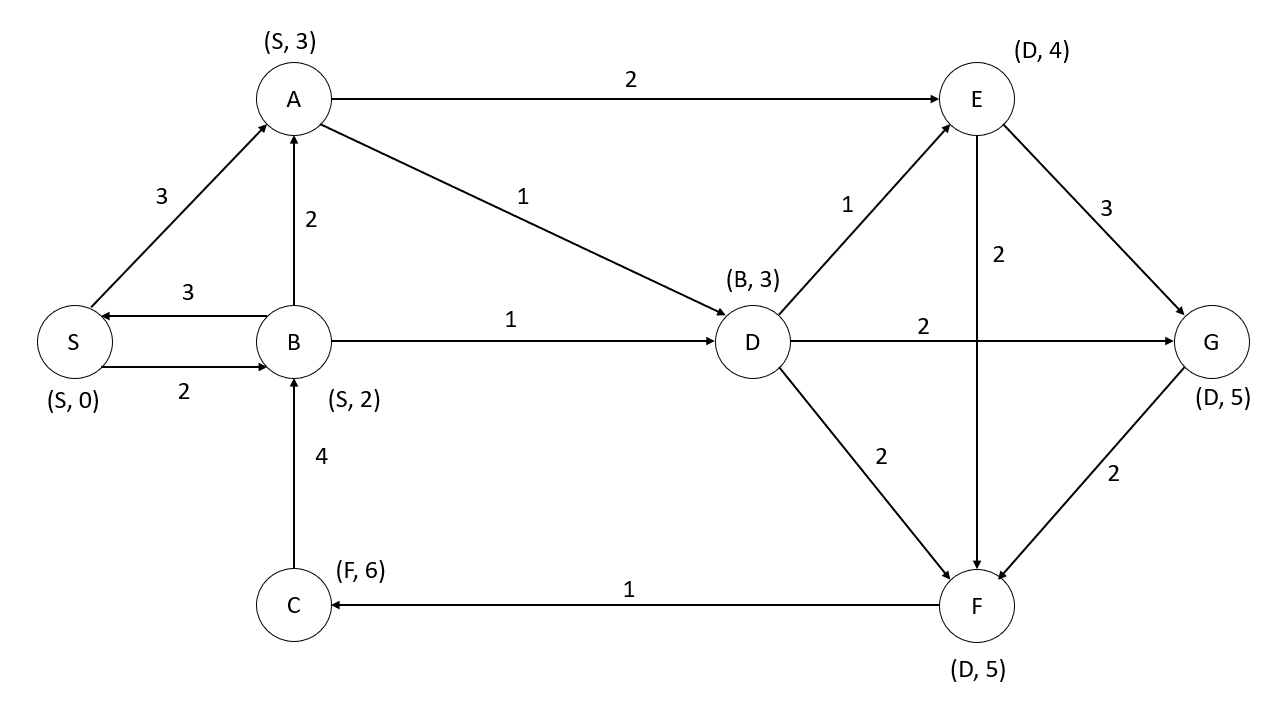
**Step VII)**

Known: S(S,0), B(S,2), A(S,3), D(B,3), E(D,4), F(D,5), G(D,5)

Unknown: C(F,6)

**Step VIII)**

Known: S(S,0), B(S,2), A(S,3), D(B,3), E(D,4), F(D,5), G(D,5), C(F,6)

****

**QUESTION 2**

**Initial:** (True, false stands for v.known of that specific vertices, and number is distance to tree)

Vertices = {S(True,0), A(False,3), B(False,2), C(False,∞), D(False,∞), E(False,∞), F(False,∞), G(False,∞)}

Edges = {}

**Step I)**

Vertices = {S(True,0), A(False,2), B(True,2), C(False,∞), D(False,1), E(False,∞), F(False,∞), G(False,∞)}

Edges = {SB}

**Step II)**

Vertices = {S(True,0), A(False,2), B(True,2), C(False,∞), D(True,1), E(False,1), F(False,2), G(False,2)}

Edges = {SB, BD}

**Step III)**

Vertices = {S(True,0), A(False,2), B(True,2), C(False,∞), D(True,1), E(True,1), F(False,2), G(False,2)}

Edges = {SB, BD, DE}

**Step IV)**

Vertices = {S(True,0), A(False,2), B(True,2), C(False,∞), D(True,1), E(True,1), F(False,2), G(True,2)}

Edges = {SB, BD, DE, DG}

**Step V)**

Vertices = {S(True,0), A(False,2), B(True,2), C(False,1), D(True,1), E(True,1), F(True,2), G(True,2)}

Edges = {SB, BD, DE, DG, DF}

**Step VI)**

Vertices = {S(True,0), A(False,2), B(True,2), C(True,1), D(True,1), E(True,1), F(True,2), G(True,2)}

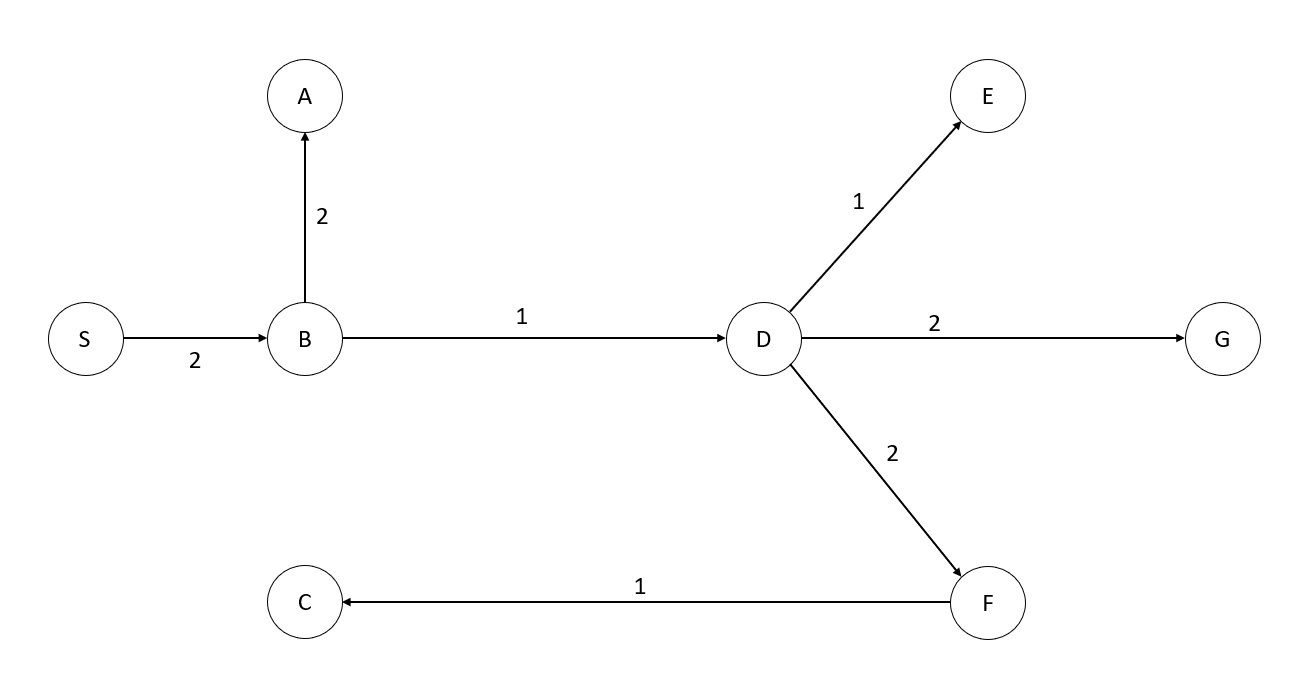
Edges = {SB, BD, DE, DG, DF, FC}

**Step VII)**

Vertices = {S(True,0), A(True,2), B(True,2), C(True,1), D(True,1), E(True,1), F(True,2), G(True,2)}

Edges = {SB, BD, DE, DG, DF, FC, BA}

*FINAL IMAGINATION OF THE TREE*



**QUESTION 3**

**Initial:** Forest = {{S}, {A}, {B}, {C}, {D}, {E}, {F}, {G}}

**Step I)** Add C to F it does not cause a circle applicable

Forest = {{S}, {A}, {B}, {C, F}, {D}, {E}, {G}}

**Step II)** Add B to D it does not cause a circle applicable

Forest = {{S}, {A}, {B, D}, {C, F}, {E}, {G}}

**Step III)** Add A to D it does not cause a circle applicable

Forest = {{S}, {A, B, D}, {C, F}, {E}, {G}}

**Step IV)** Add D to E it does not cause a circle applicable

Forest = {{S}, {A, B, D, E}, {C, F}, {G}}

**Step V)** Add S to B it does not cause a circle applicable

Forest = {{S, A, B, D, E}, {C, F}, {G}}

**Step VI)** Add B to A it does cause a circle not applicable

Forest = {{S, A, B, D, E}, {C, F}, {G}}

**Step VII)** Add A to E it does cause a circle not applicable

Forest = {{S, A, B, D, E}, {C, F}, {G}}

**Step VIII)** Add D to G it does not cause a circle applicable

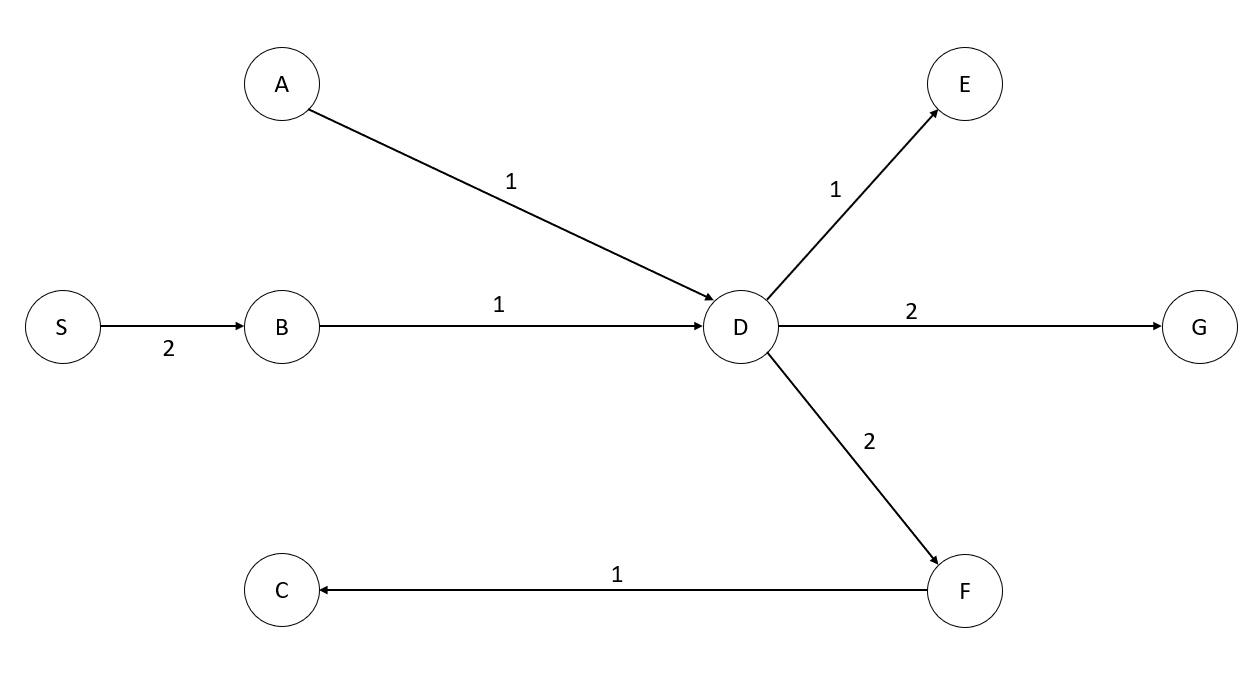
Forest = {{S, A, B, D, E, G}, {C, F}}

**Step IX)** Add G to F it does cause a circle not applicable

Forest = {{S, A, B, D, E, G}, {C, F}}

**Step X)** Add D to F it does not cause a circle applicable

Forest = {{S, A, B, D, E, G, C, F}} All vertices in one tree, terminate.



**QUESTION 4**

**Initial:** Since there is no weight of edges, numerate S as 0 (distance from S to S is 0)

Distances = {S=0, A=∞, B=∞, C=∞, D=∞, E=∞, F=∞, G=∞}

**Step I)** Unvisited vertices 1 away from S: (A and B)

Distances = {S=0, A=1, B=1, C=∞, D=∞, E=∞, F=∞, G=∞}

**Step II)** Unvisited vertices 2 away from S: ,1 away from A: (D and E)

Distances = {S=0, A=1, B=1, C=∞, D=2, E=2, F=∞, G=∞}

**Step III)** Unvisited vertices 2 away from S: ,1 away from B: (no vertices, D is already visited)

Distances = {S=0, A=1, B=1, C=∞, D=2, E=2, F=∞, G=∞}

**Step V)** Unvisited vertices 3 away from S: ,1 away from D: (F and G)

Distances = {S=0, A=1, B=1, C=∞, D=2, E=2, F=3, G=3}

**Step VI)** Unvisited vertices 3 away from S: ,1 away from E: (no vertices, G is already visited)

Distances = {S=0, A=1, B=1, C=∞, D=2, E=2, F=3, G=3}

**Step VII)** Unvisited vertices 4 away from S: ,1 away from F: (C)

Distances = {S=0, A=1, B=1, C=4, D=2, E=2, F=3, G=3}

**Step VII)** Unvisited vertices 4 away from S: ,1 away from G: (no vertices, G and F is already visited)

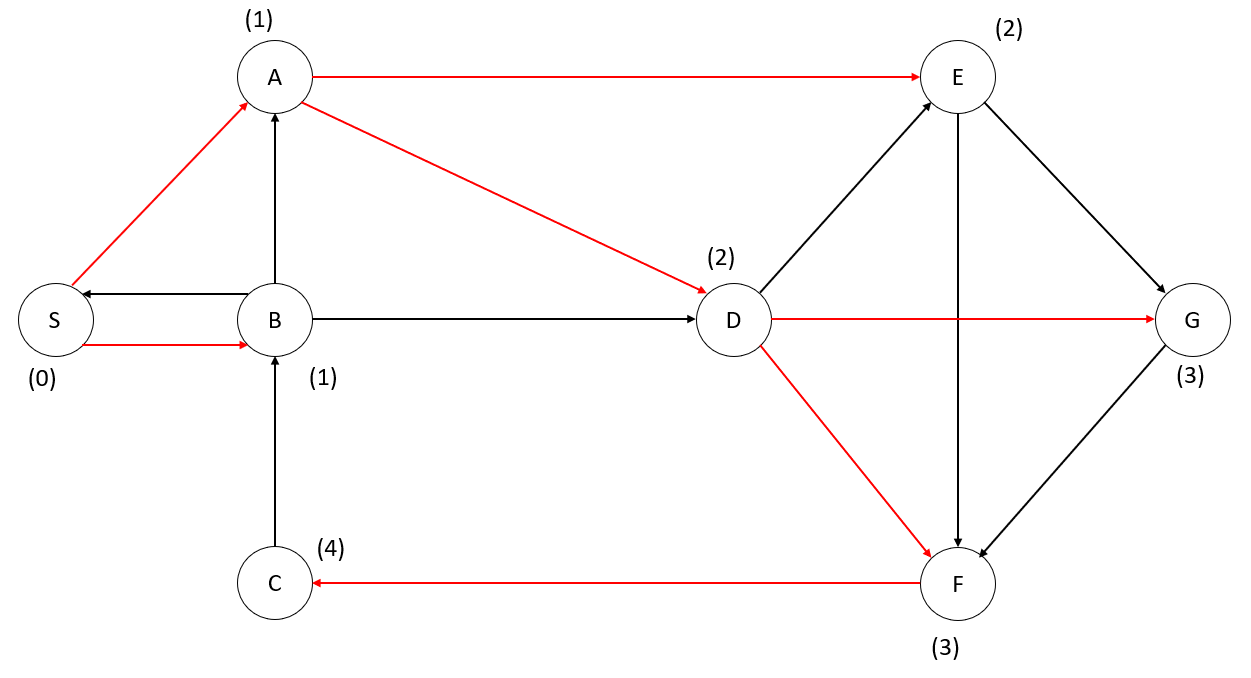
Distances = {S=0, A=1, B=1, C=4, D=2, E=2, F=3, G=3}

**Step VII)** Unvisited vertices 5 away from S: ,1 away from C: (no vertices, B is already visited)

Distances = {S=0, A=1, B=1, C=4, D=2, E=2, F=3, G=3}

There is no distance greater than 4, checks are immediately failing (terminating).

*FINAL VERSION OF THE BREADTH-FIRST TRAVERSAL*



**QUESTION 5**

**Initial:** Since there is no weight of edges, numerate S as 1 (depth of S is 1)

**Step 1)** Call the adjacent DFS’s from S, dfs(A), dfs(B):

A is unvisited:

Depths = {S=1, A=2, B=∞, C=∞, D=∞, E=∞, F=∞, G=∞}

**Step 1.1)** Call the adjacent DFS’s from A, dfs(D), dfs(E):

D is unvisited:

Depths = {S=1, A=2, B=∞, C=∞, D=3, E=∞, F=∞, G=∞}

**Step 1.1.1)** Call the adjacent DFS’s from D, dfs(E), dfs(F), dfs(G):

E is unvisited:

Depths = {S=1, A=2, B=∞, C=∞, D=3, E=4, F=∞, G=∞}

**Step 1.1.1.1)** Call the adjacent DFS’s from E, dfs(F), dfs(G):

F is unvisited:

Depths = {S=1, A=2, B=∞, C=∞, D=3, E=4, F=5, G=∞}

**Step 1.1.1.1.1)** Call the adjacent DFS’s from F, dfs(C):

C is unvisited:

Depths = {S=1, A=2, B=∞, C=6, D=3, E=4, F=5, G=∞}

**Step 1.1.1.1.1.1)** Call the adjacent DFS’s from C, dfs(B):

B is unvisited:

Depths = {S=1, A=2, B=7, C=6, D=3, E=4, F=5, G=∞}

B will not call dfs(S) nor dfs(A) nor dfs(D), because all of them are visited

**Step 1.1.1.1.2)** Call the adjacent DFS’s from E (continuing) dfs(G):

G is unvisited:

Depths = {S=1, A=2, B=7, C=6, D=3, E=4, F=5, G=8}

G will not call dfs(F) because it is already visited

**Step 1.1.1.2)** Call the adjacent DFS’s from D (continuing) dfs(F), dfs(G):

F and G are visited, dfs for them are not called

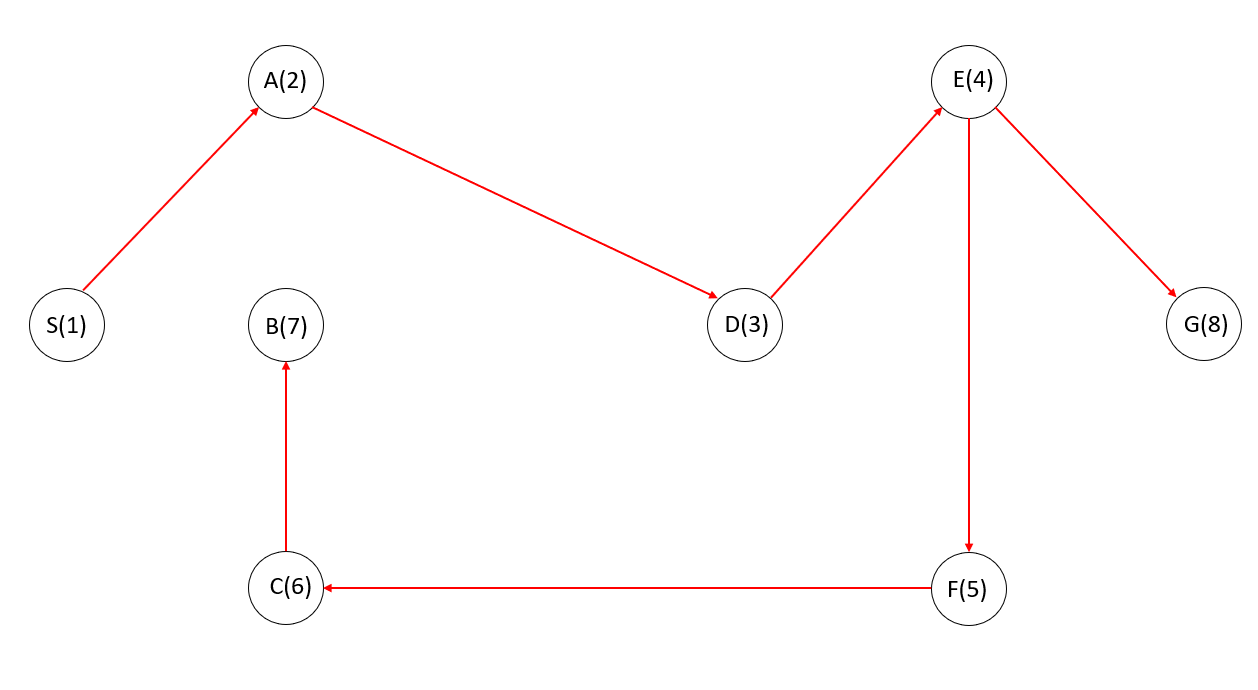
**Step 1.1.2)** Call the adjacent DFS’s from A (continuing) dfs(E):

E is visited, dfs for that is not called.

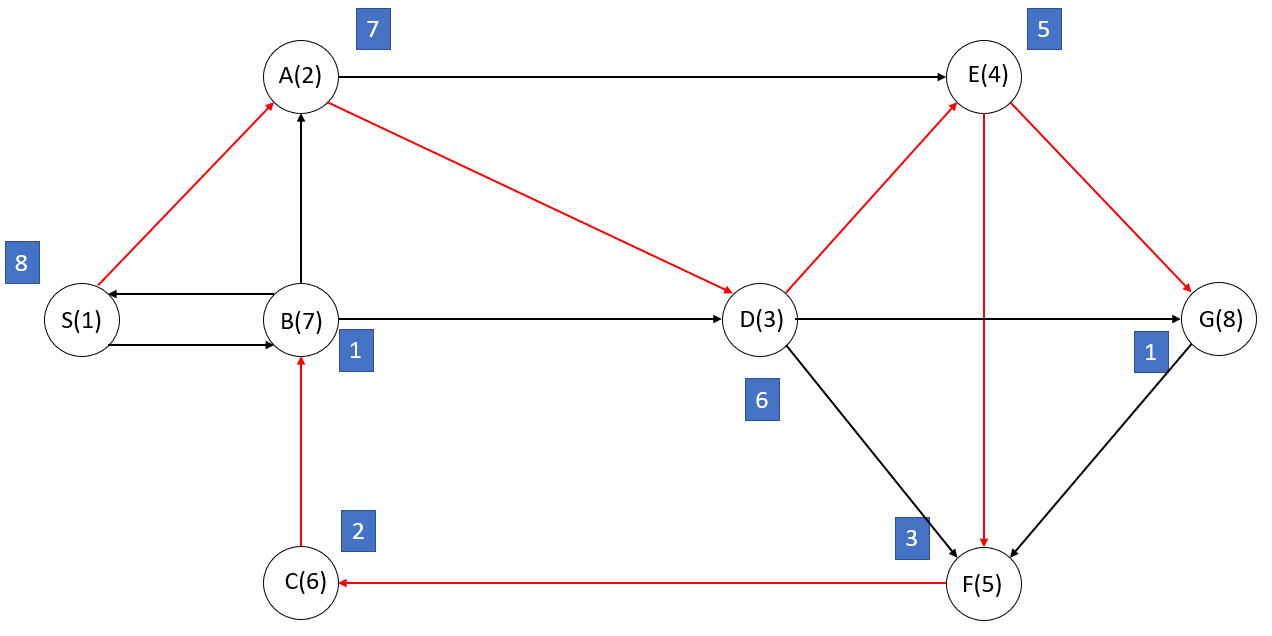
**Step 1.2)** Call the adjacent DFS’s from S (continuing) dfs(B):

B is visited, dfs for that is not called.

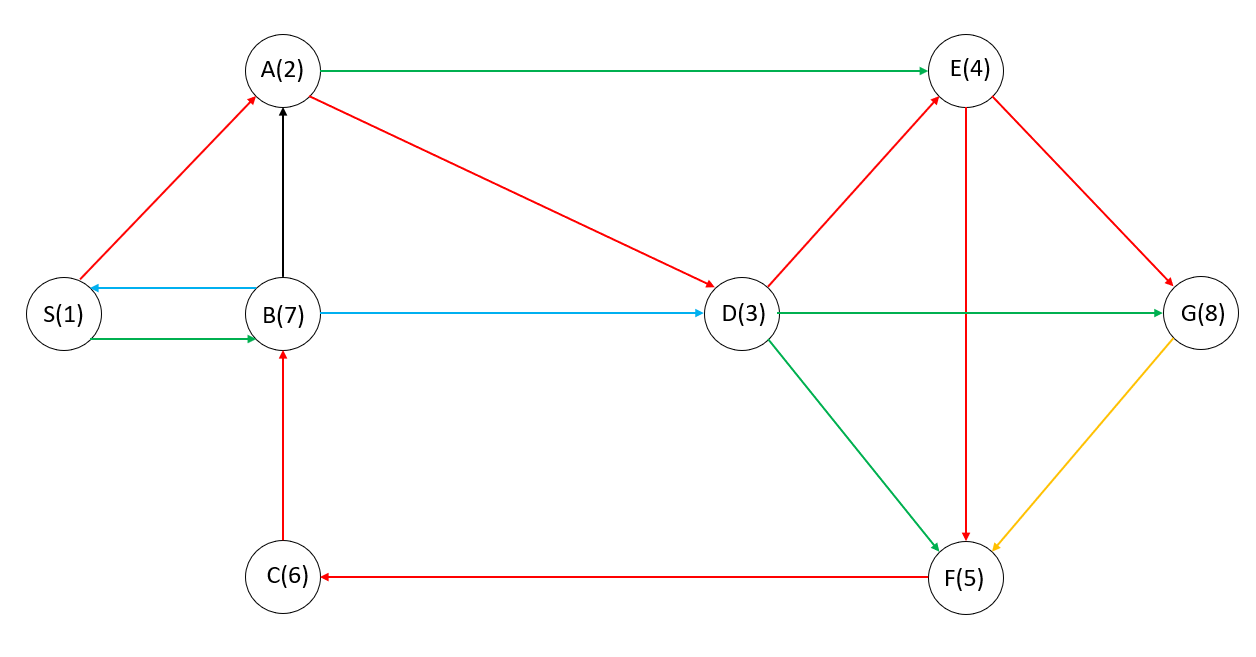
*FINAL VERSION OF DEPTH-FIRST TREE:*



*POST-ORDER NUMBERS* (blue numbers)and *PRE-ORDER NUMBERS* (inside of the circles)



*ARC TYPES* (orange: cross arc, green: forward arc, blue: backward arc, red: tree arcs)



**QUESTION 6**

**Initial:** Vertex with in-degree 0 is “D”.

Step 1: After removing D, vertex with in-degree 0 is B

Step 2: After removing D and B, vertices with in-degree 0 is A and E (E is chosen first)

Step 3: After removing D, B and E, vertex A has still 0 in-degree, so it is printed

Step 4: After removing D, B, E and A, F has 0 in-degree

Step 5: There are only two vertices left (C and G) and they both have no degrees, order does not matter (C is chosen first)

D 🡪 B 🡪 E 🡪 A 🡪 F 🡪 C 🡪 G