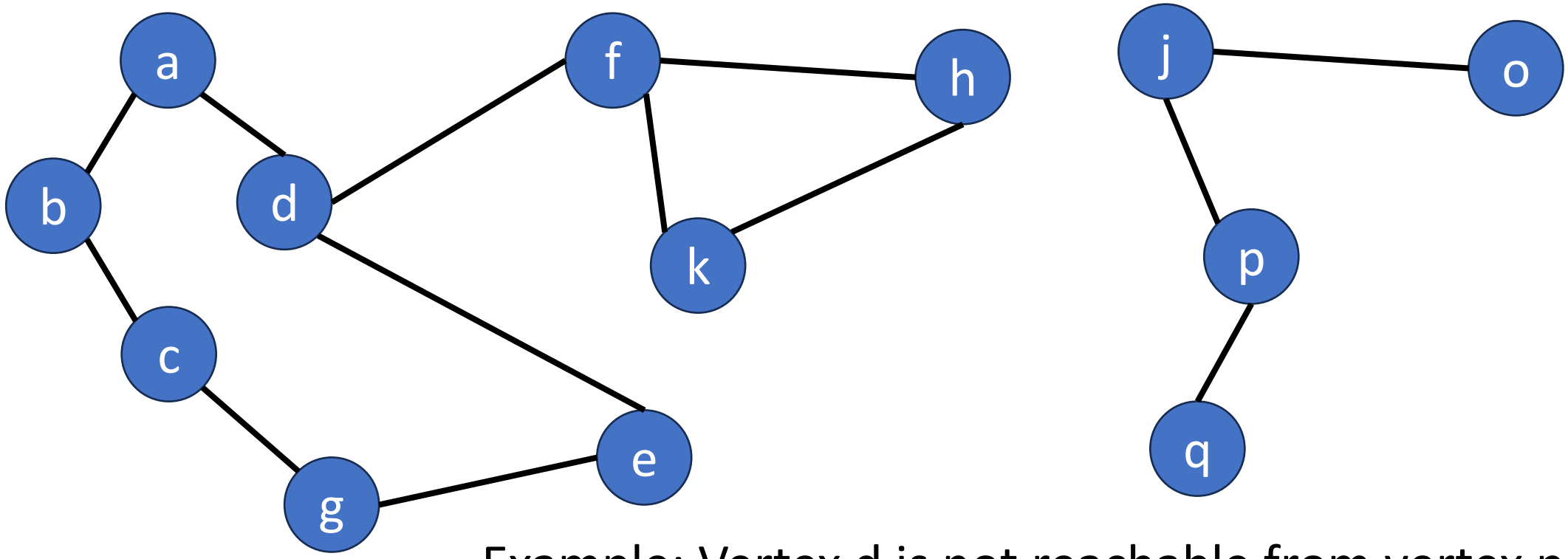
Graph search methods

Ch. 6.2

Graph search methods

- A vertex *u* is reachable from vertex *v* iff there is a path from *v* to *u*.
- A search method starts at a given vertex *v* and visits/labels/marks every vertex that is reachable from *v*.



Example: Vertex d is not reachable from vertex p.

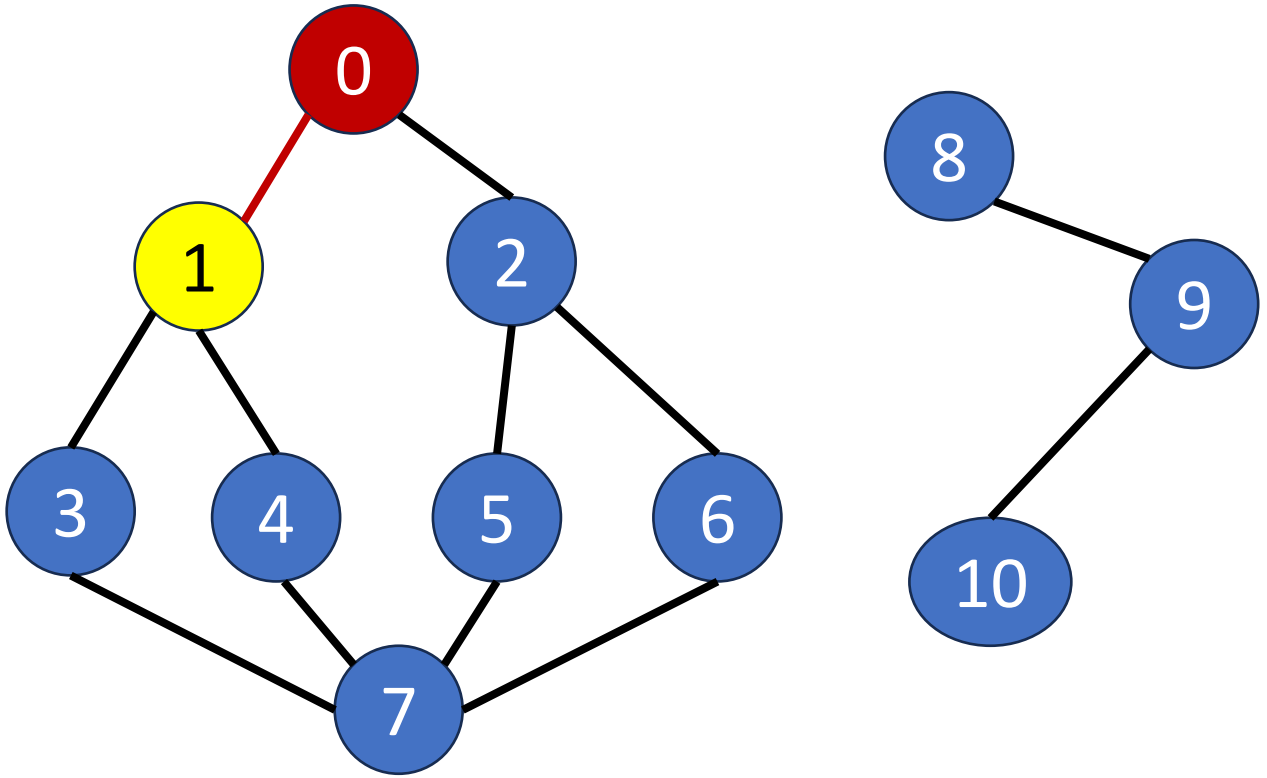
Graph search methods

- Commonly used search methods
  - Depth First Search
    - Similar to a preorder tree traversal
  - Breath First Search
    - Similar to a level-order tree traversal

Depth first search: method

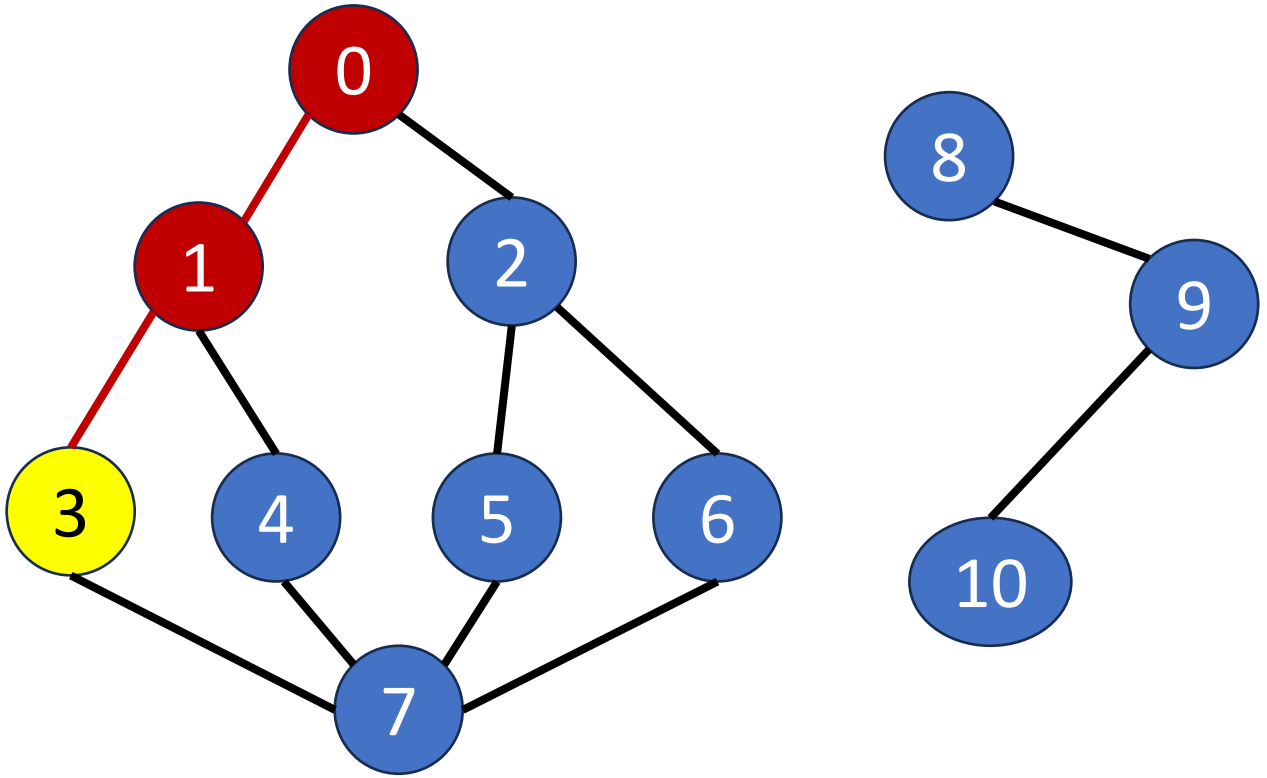
```
short int visited[MAX_VERTICES];
/*Depth first search of a graph beginning at v.*/
dfs(v)
{
   /*Label vertex v as reached.*/
   visited[v] = TRUE;
   /*Explore the adjacent unvisited vertices.*/
   for (each unreached vertex u adjacent from v)
      dfs(u);
}
```

Depth first search: example



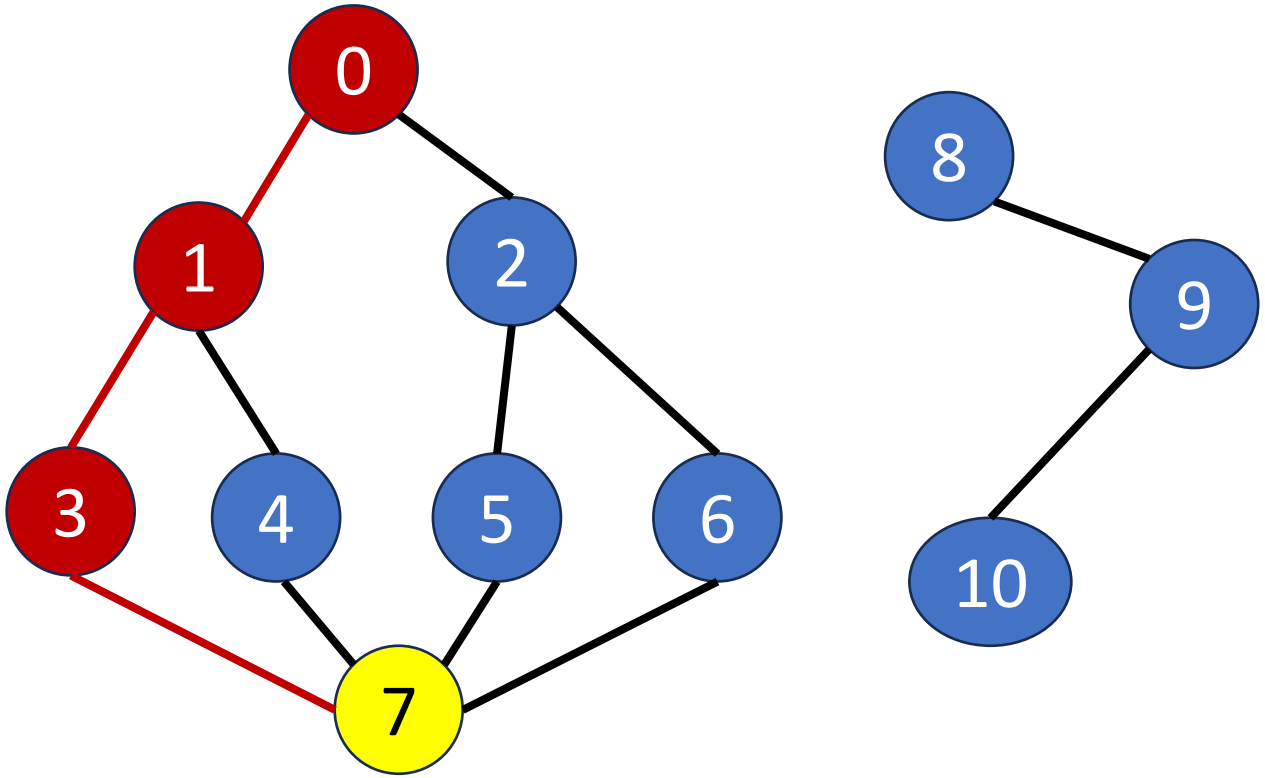
- dfs(v_0): $[v_0]$
  - Start search at vertex 0.
  - Label vertex 0 and do a dfs from either vertex 1 or vertex 2.
  - In this example, vertex 1 is selected. Do dfs(v_1).

Depth first search: example



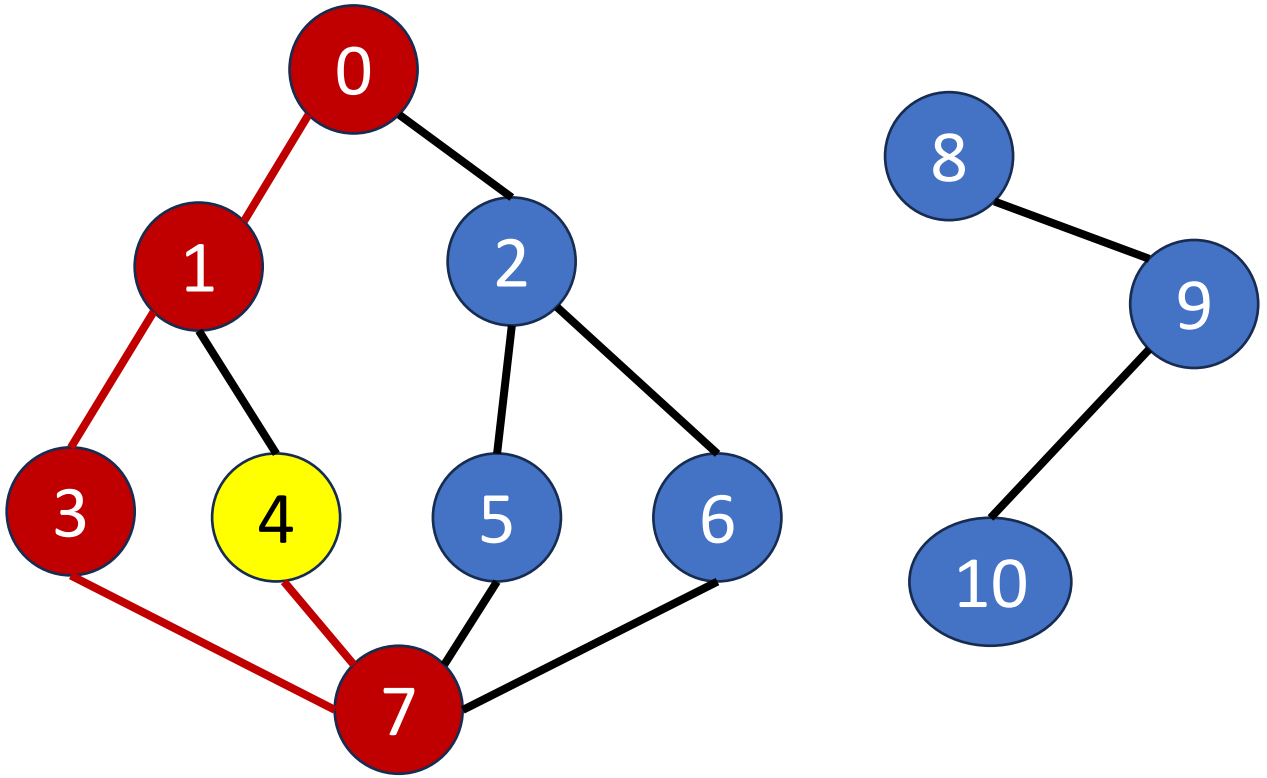
- dfs(v_1): $[v_0, v_1]$
  - Start search at vertex 1.
  - Label vertex 1 and do a dfs from either vertex 3 or vertex 4.
  - In this example, vertex 3 is selected. Do dfs(v_3).

Depth first search: example



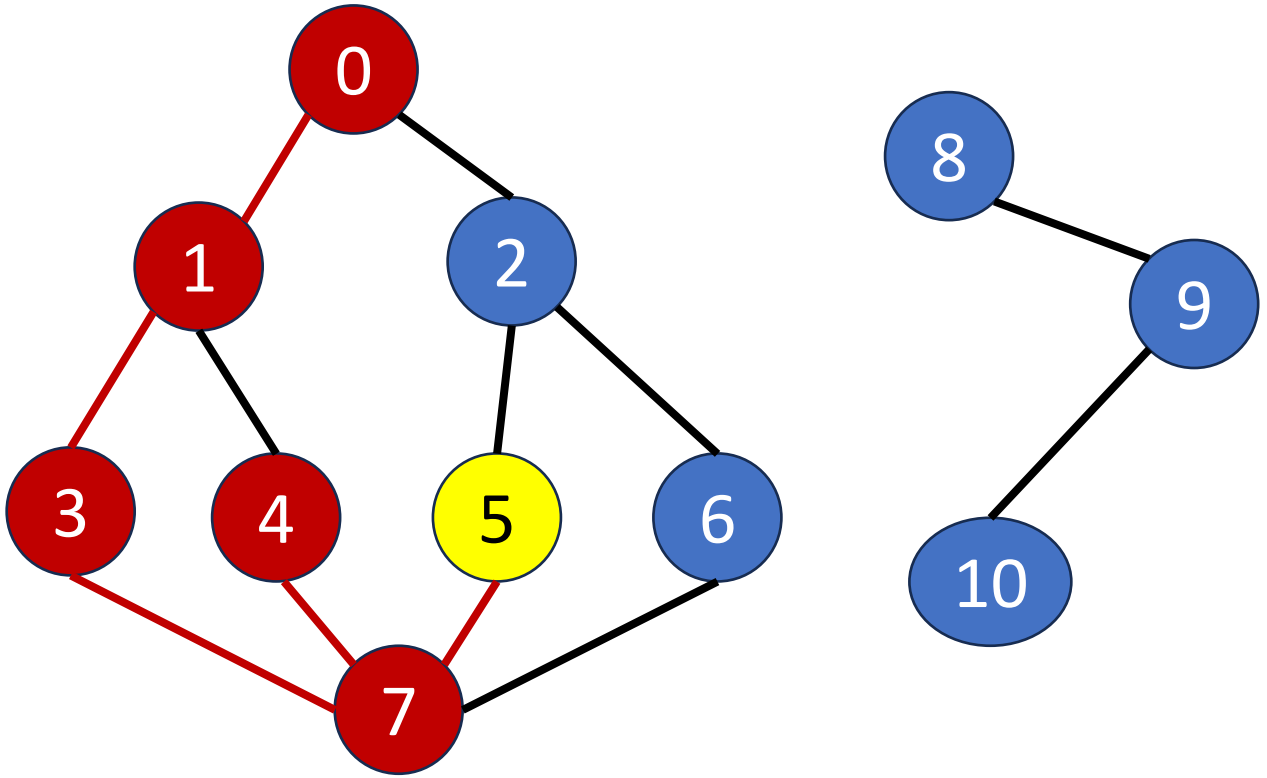
- dfs(v_3): $[v_0, v_1, v_3]$
  - Start search at vertex 3.
  - Label vertex 3 and do a dfs from vertex 7.
  - Do dfs(v_7).

Depth first search: example



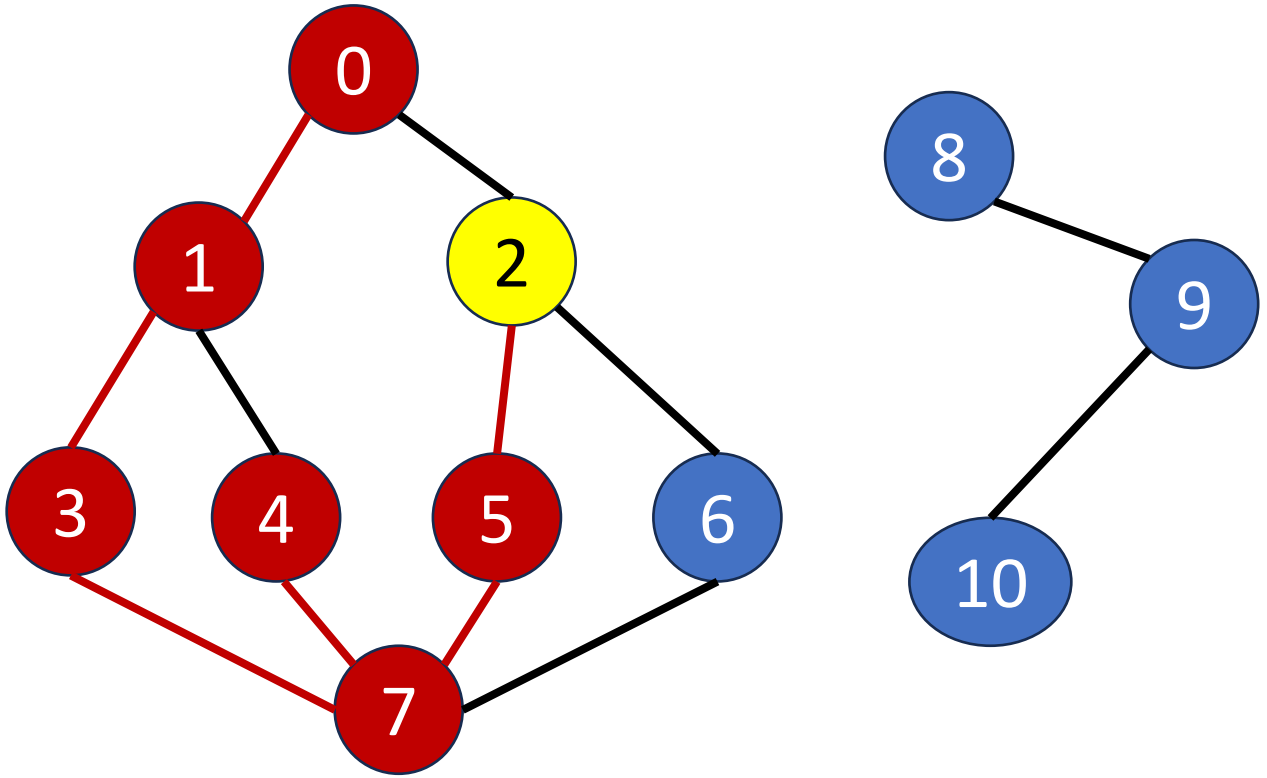
- dfs(v_7): $[v_0, v_1, v_3, v_7]$
  - Start search at vertex 7.
  - Label vertex 7 and do a dfs from vertices 4, 5, or 6.
  - In this example, vertex 4 is selected. Do dfs(v_4).

Depth first search: example



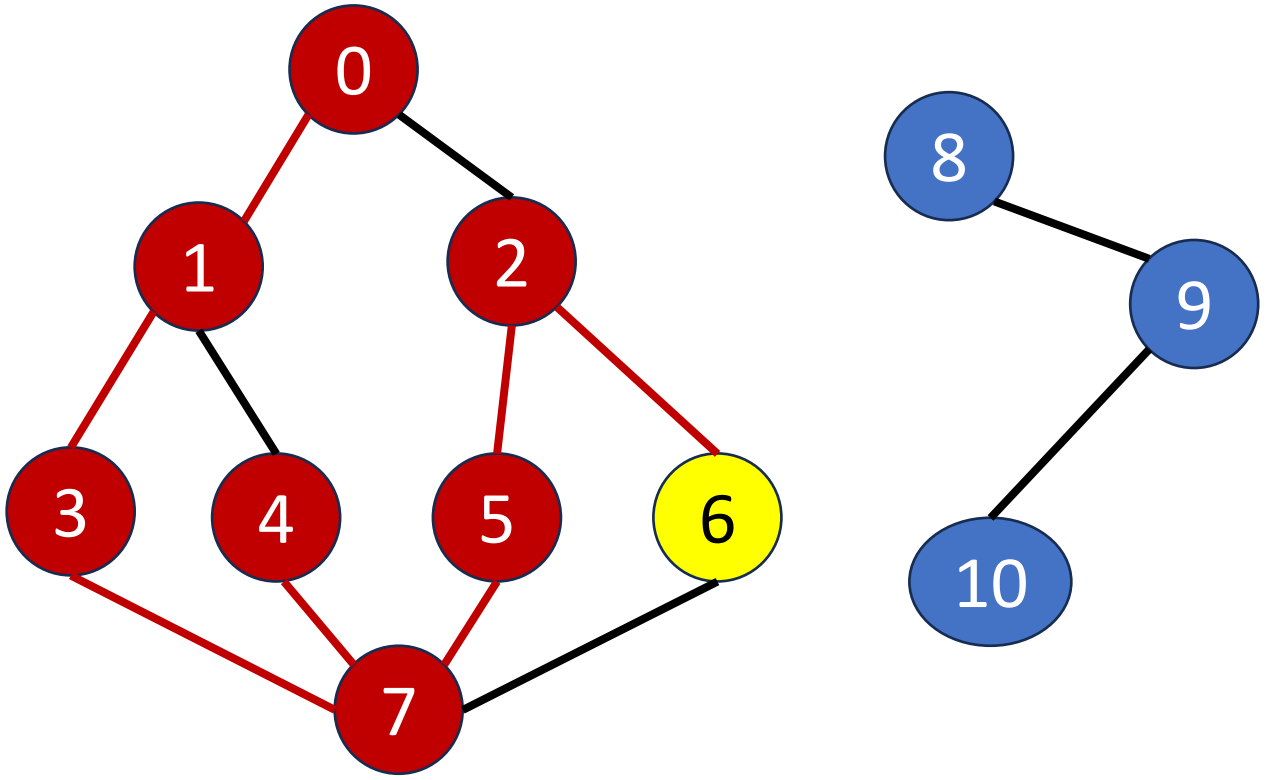
- dfs(v_4): $[v_0, v_1, v_3, v_7, v_4]$
  - Start search at vertex 4.
  - No unvisited adjacent vertex. Return to vertex 7.
- do a dfs from vertices 5 or 6.
- In this example, vertex 5 is selected. Do dfs(v_5).

Depth first search: example



- dfs(v_5): $[v_0, v_1, v_3, v_7, v_4, v_5]$
  - Start search at vertex 5.
  - Label vertex 5 and do a dfs from vertex 2.
  - Do dfs(v_2).

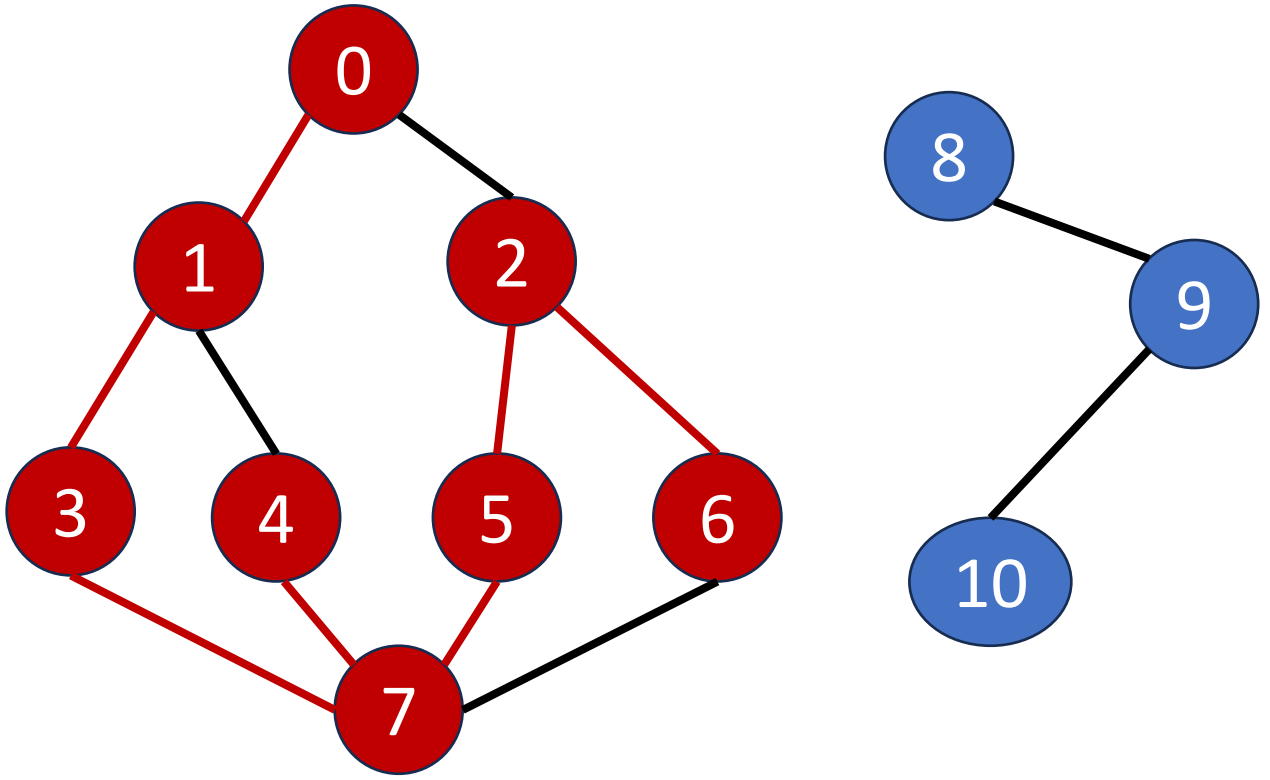
Depth first search: example



- dfs(v_2):
  - Start search at vertex 2.
  - Label vertex 2 and do a dfs from vertex 6.
  - Do dfs(v_6).

$[v_0, v_1, v_3, v_7, v_4, v_5, v_2]$

Depth first search: example



- dfs(v_6): $[v_0, v_1, v_3, v_7, v_4, v_5, v_2, v_6]$
  - Start search at vertex 6.
  - Label vertex 6. No unvisited adjacent vertex.
- Return to vertices 2, 5, 7, 3, 1, 0 sequentially.

Time complexity

- When adjacency matrix is used

$$O(n^2)$$

  - For a node, search the corresponding row to find adjacent vertices. ➔ $O(n)$
  - Visit at most n nodes. ➔ $O(n \times n)$

- When adjacency lists are used

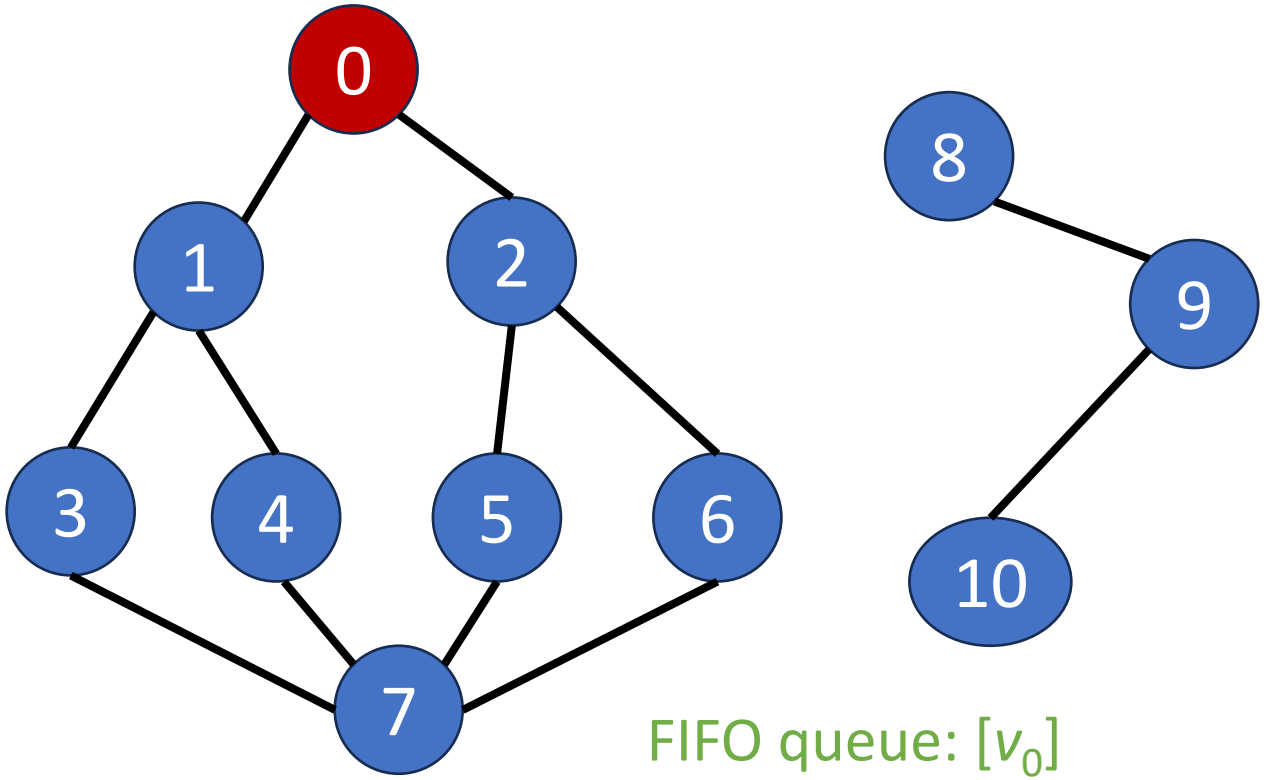
$$O(n+e)$$

  - Search at most e edges and n nodes.

Breadth first search: method

- Visit start vertex and put into a FIFO queue.
- Repeatedly remove a vertex from the queue, visit its unvisited adjacent vertices, put newly visited vertices into the queue.

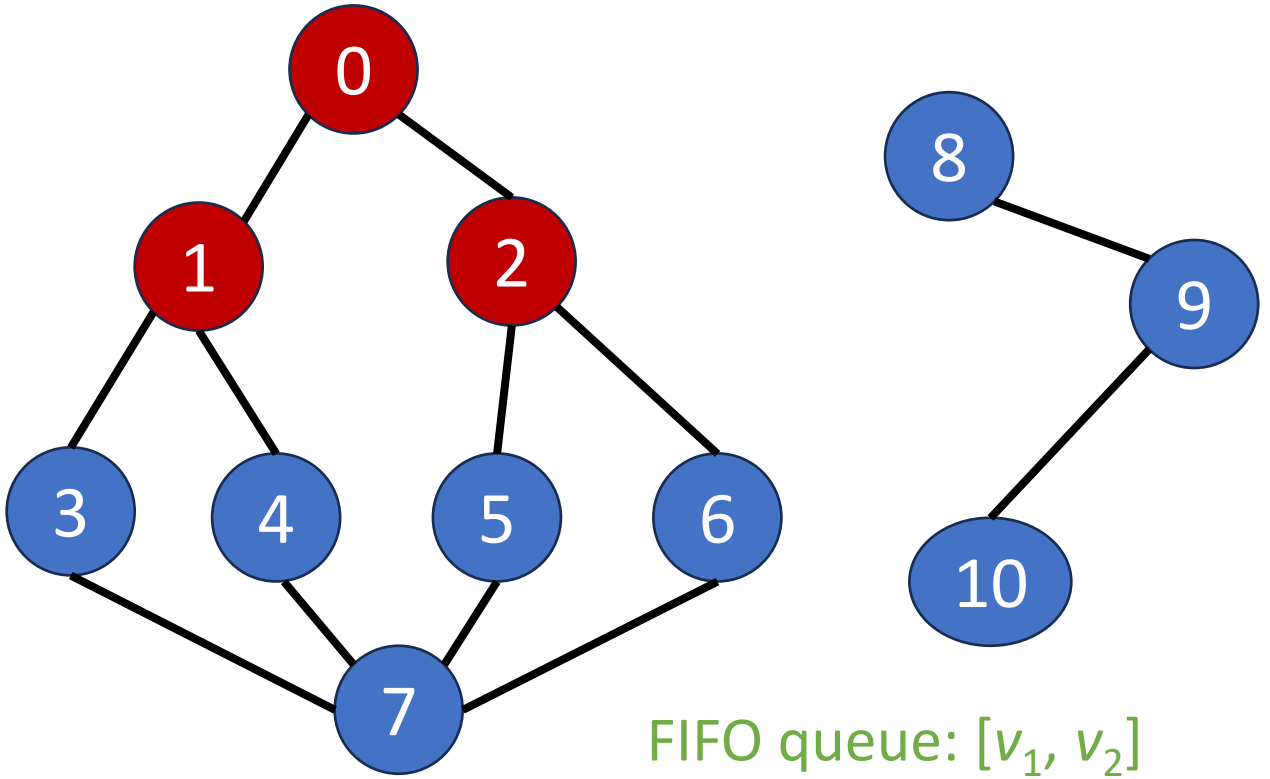
Breadth first search: example



FIFO queue: $[v_0]$

- Start search at vertex 0.
- Label vertex 0 and add vertex 0 to a FIFO queue.

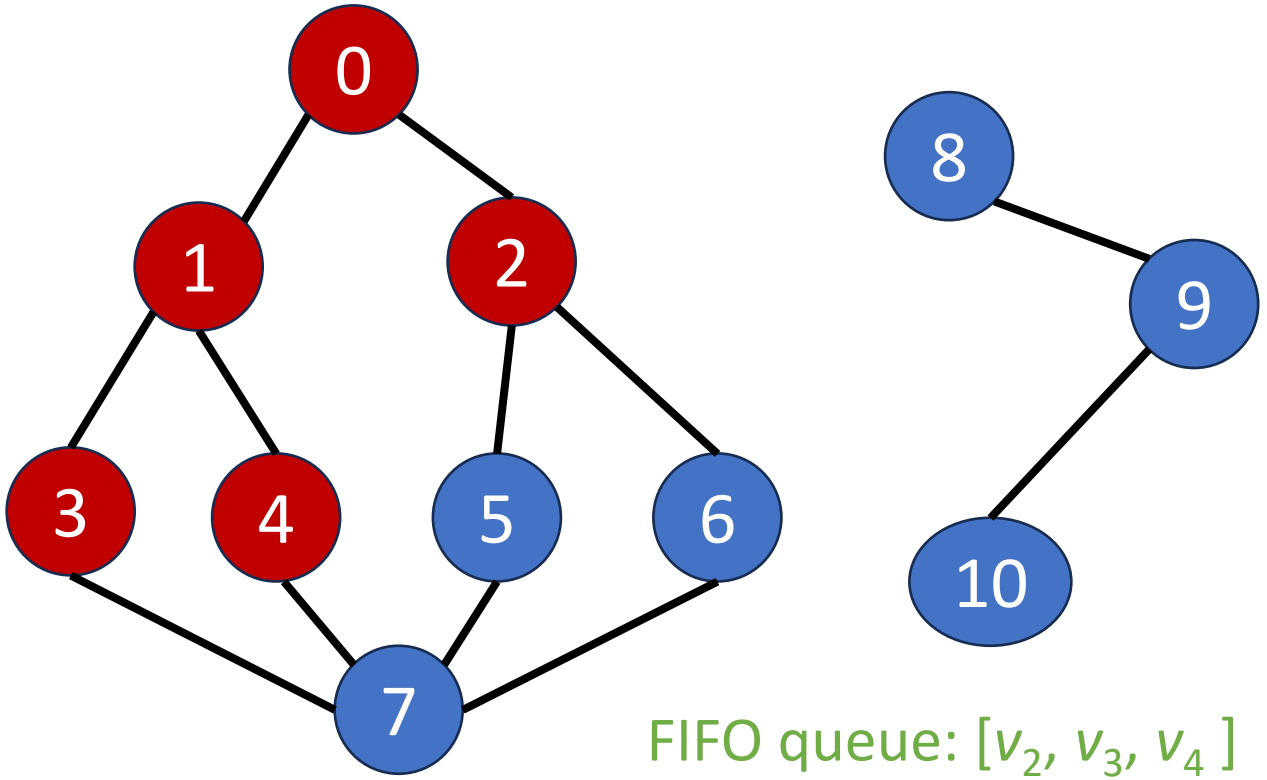
Breadth first search: example



FIFO queue: [v_1, v_2]

- Remove vertex 0 from queue.
- Visit adjacent unvisited vertices 1 and 2 and put into FIFO queue.

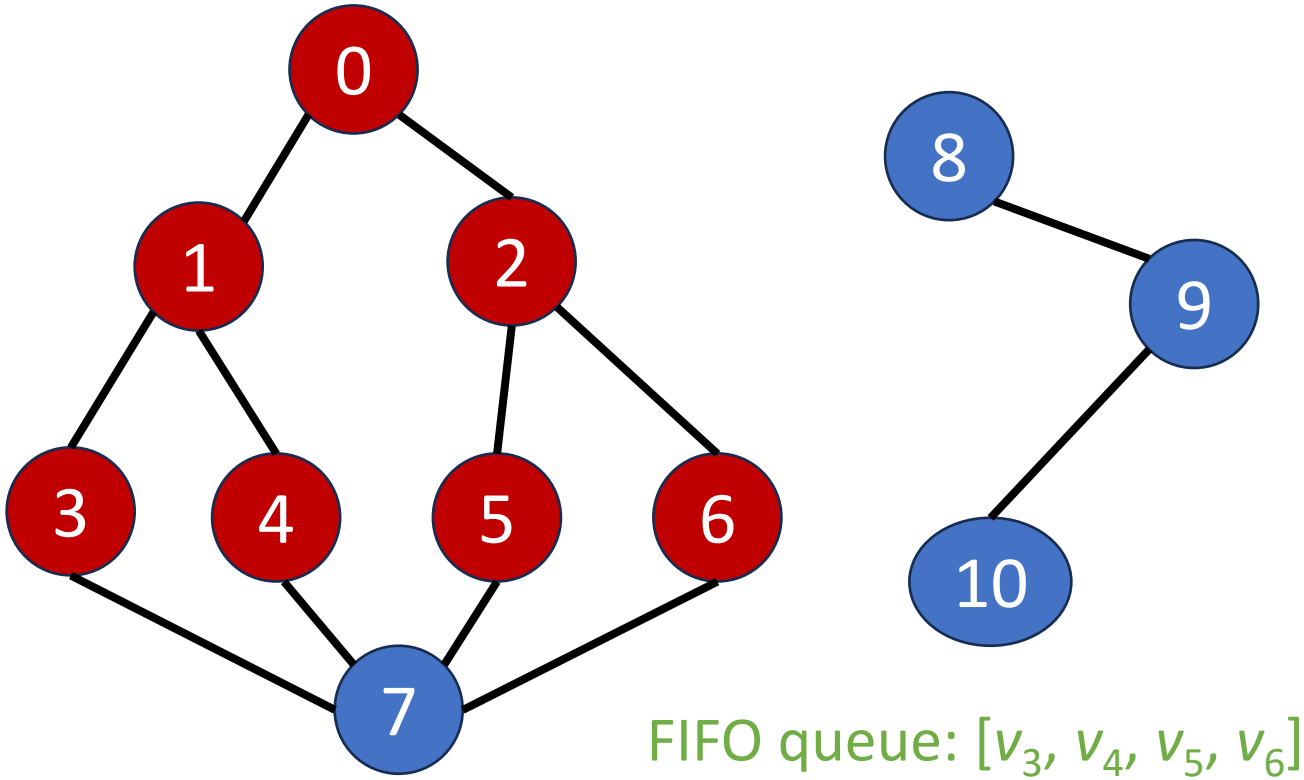
Breadth first search: example



FIFO queue: [v_2, v_3, v_4]

- Remove vertex 1 from queue.
- Visit adjacent unvisited vertices 3 and 4 and put into FIFO queue.

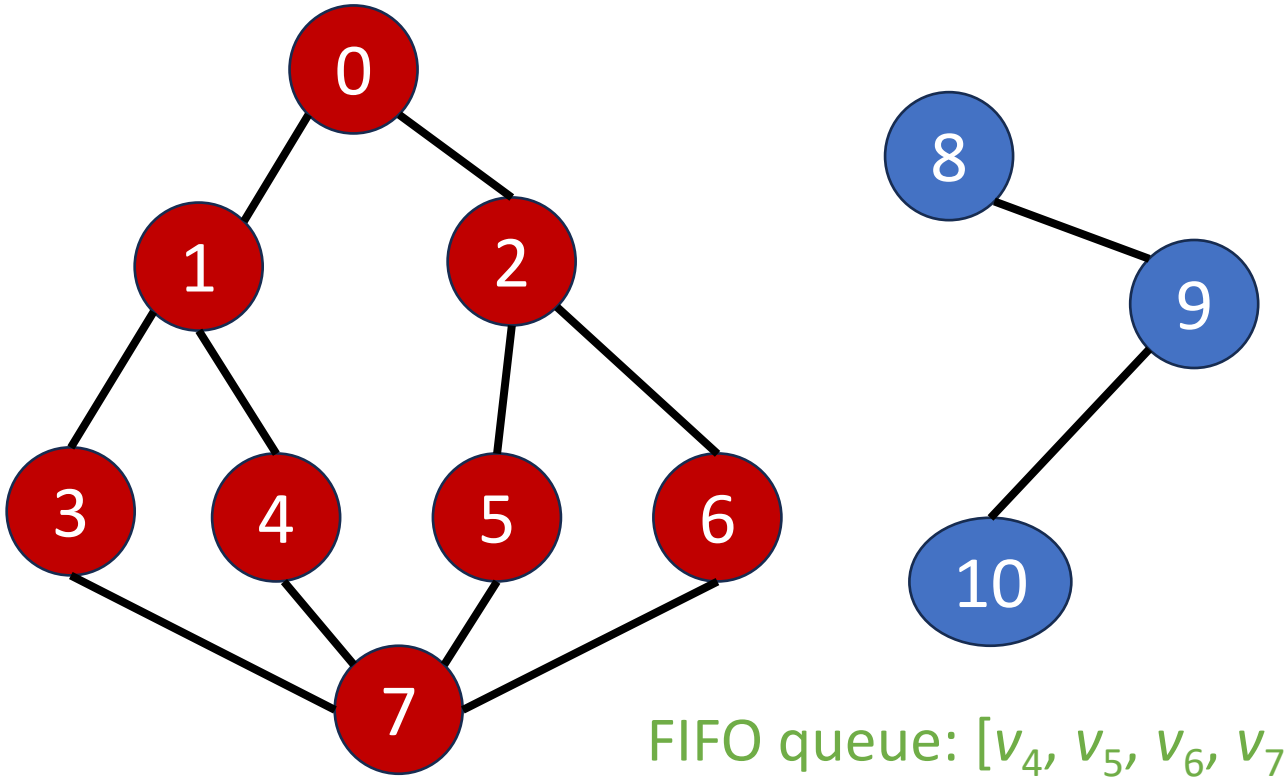
Breadth first search: example



FIFO queue: [v_3, v_4, v_5, v_6]

- Remove vertex 2 from queue.
- Visit adjacent unvisited vertices 5 and 6 and put into FIFO queue.

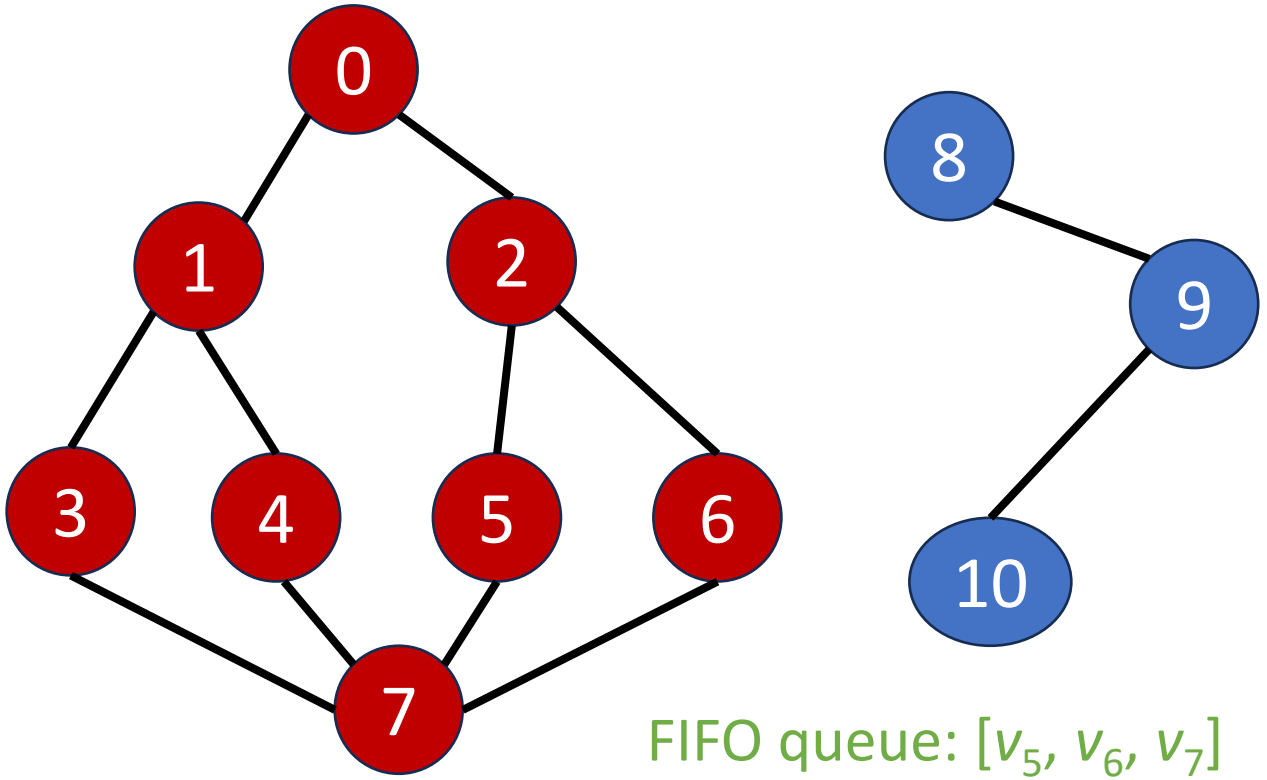
Breadth first search: example



FIFO queue: [v_4, v_5, v_6, v_7]

- Remove vertex 3 from queue.
- Visit adjacent unvisited vertex 7 and put into FIFO queue.

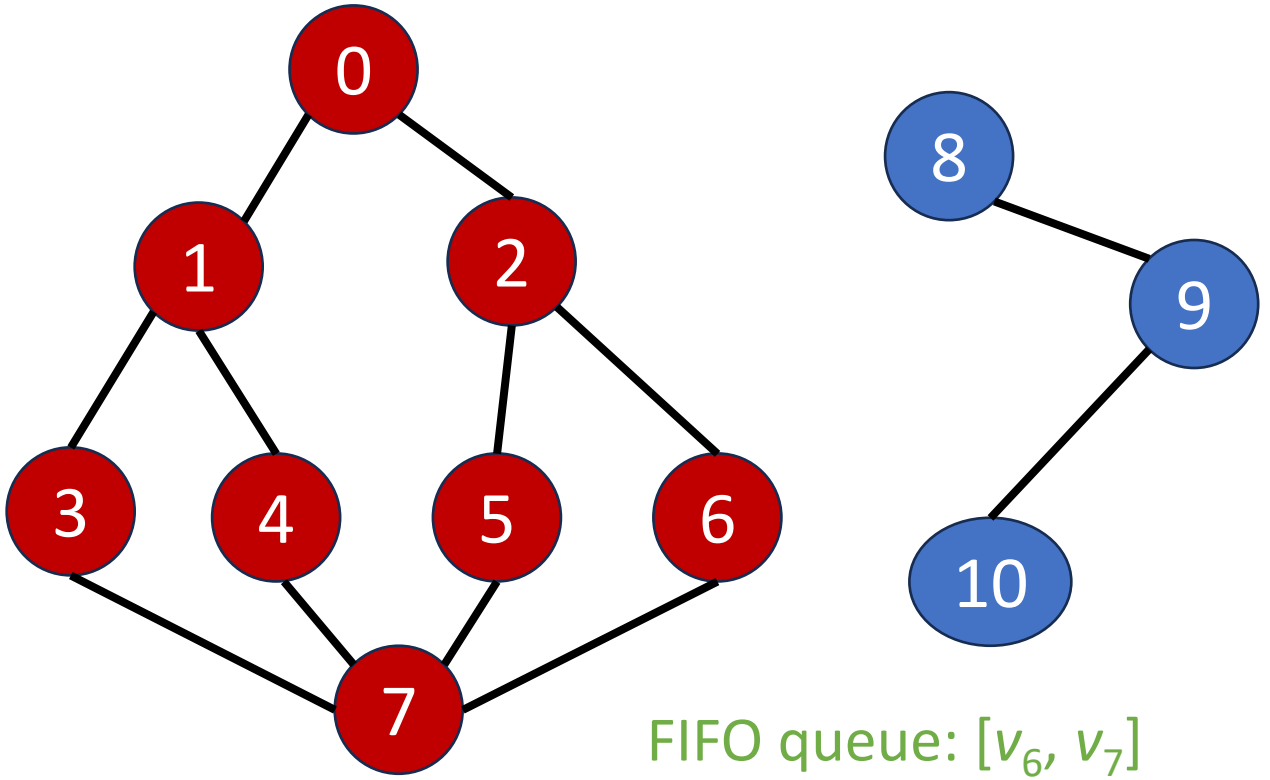
Breadth first search: example



FIFO queue: [v_5, v_6, v_7]

- Remove vertex 4 from queue.
- Visit adjacent unvisited vertex and put into FIFO queue.

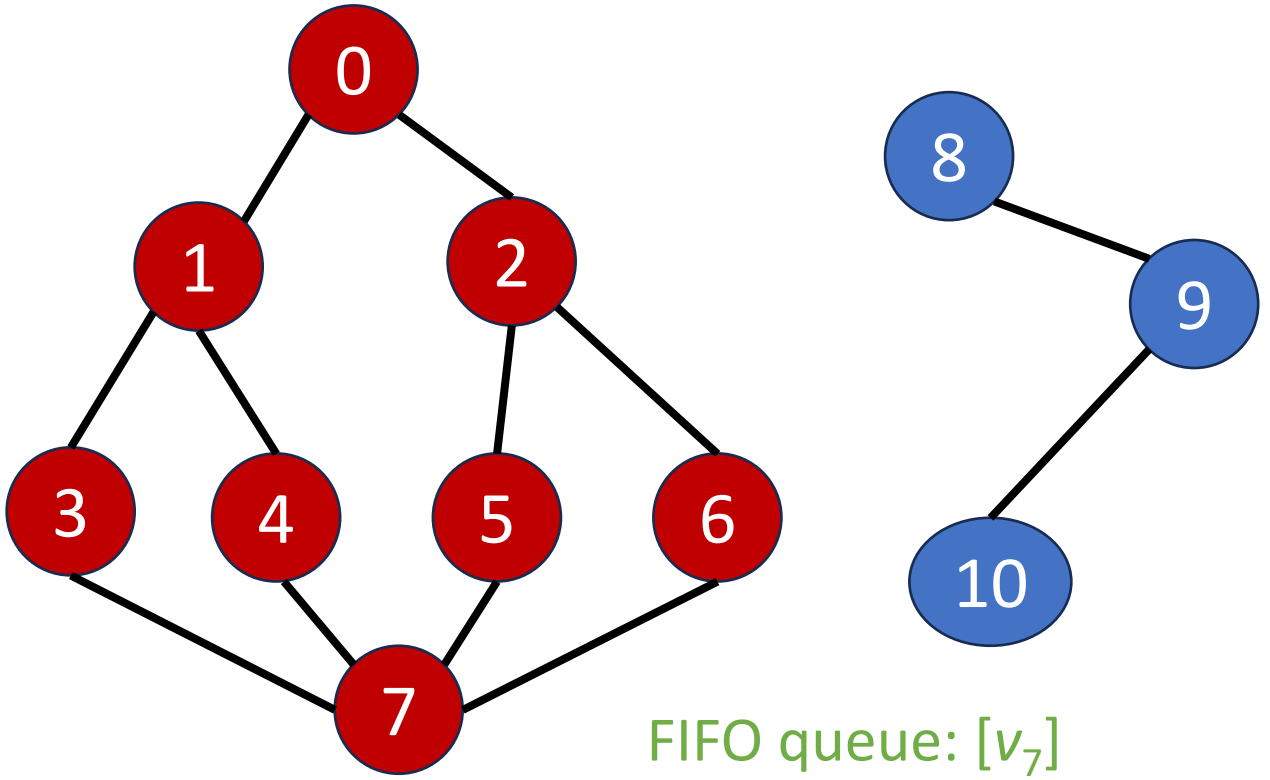
Breadth first search: example



FIFO queue: [v_6, v_7]

- Remove vertex 5 from queue.
- Visit adjacent unvisited vertex and put into FIFO queue.

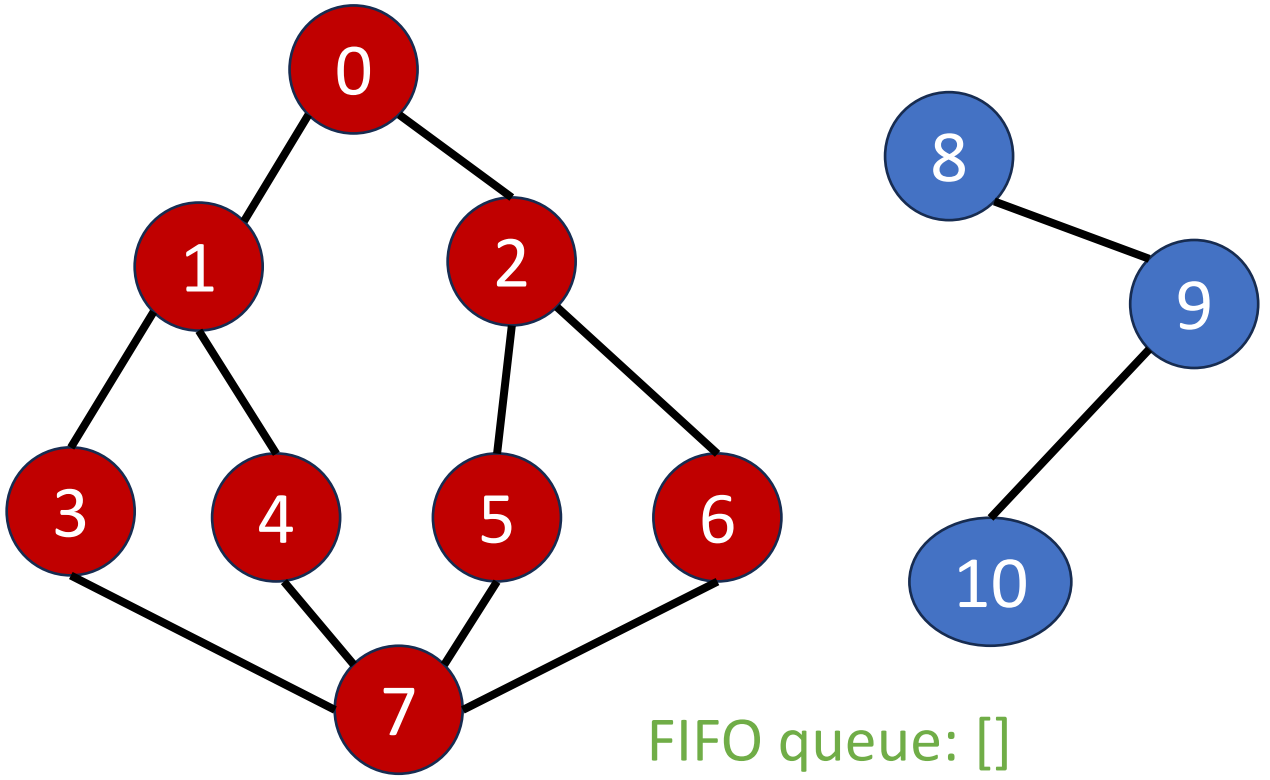
Breadth first search: example



FIFO queue: [v_7]

- Remove vertex 6 from queue.
- Visit adjacent unvisited vertex and put into FIFO queue.

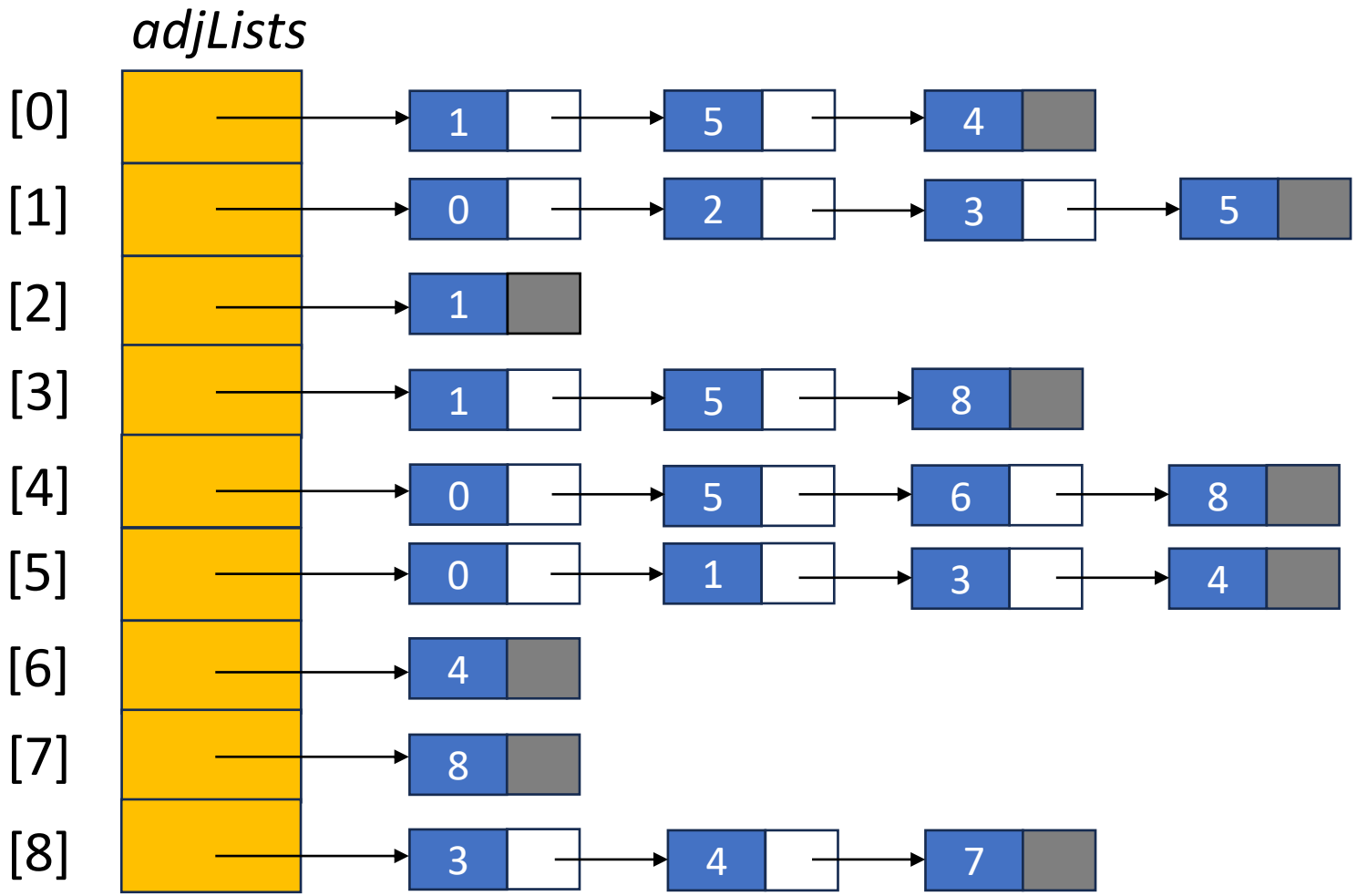
Breadth first search: example



FIFO queue: []

- Remove vertex 7 from queue.
- Visit adjacent unvisited vertex and put into FIFO queue.

Exercise

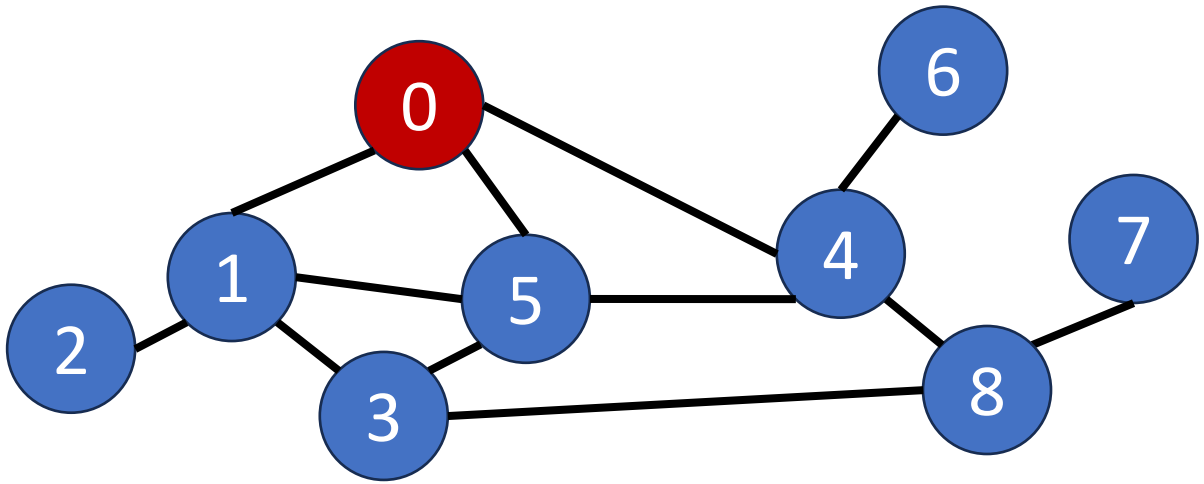


Please reply your answers of Q1-Q2 via the following link:

Group members: 1~3 people

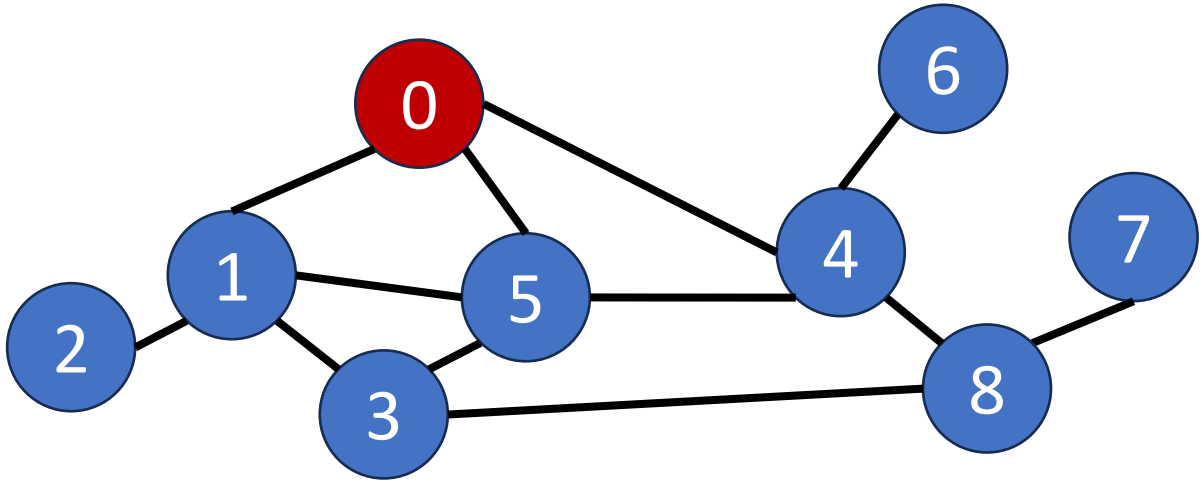
- Q1: Write out the results of DFS starting at vertex 0.
- Q2: Write out the results of BFS starting at vertex 0.

Exercise



DFS(0)

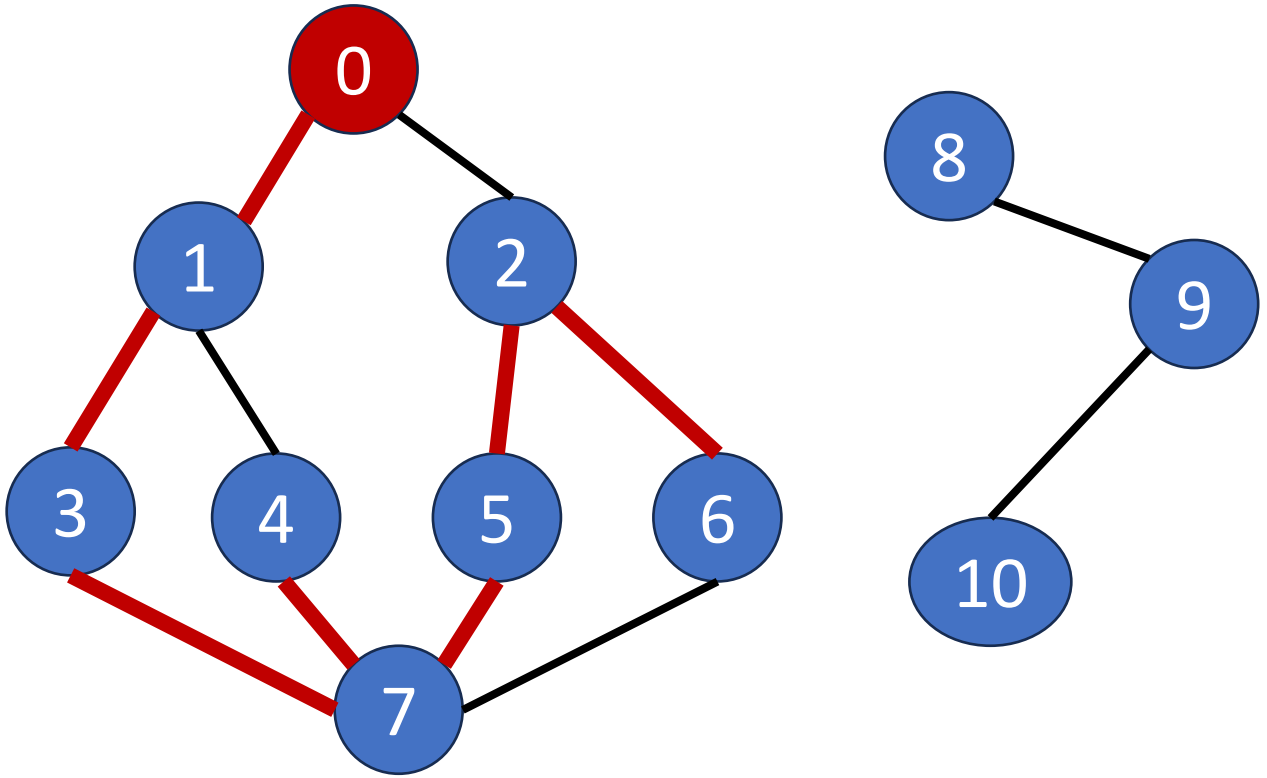
0,1,2,3,5,4,6,8,7



BFS(0)

0,1,4,5,2,3,6,8,7

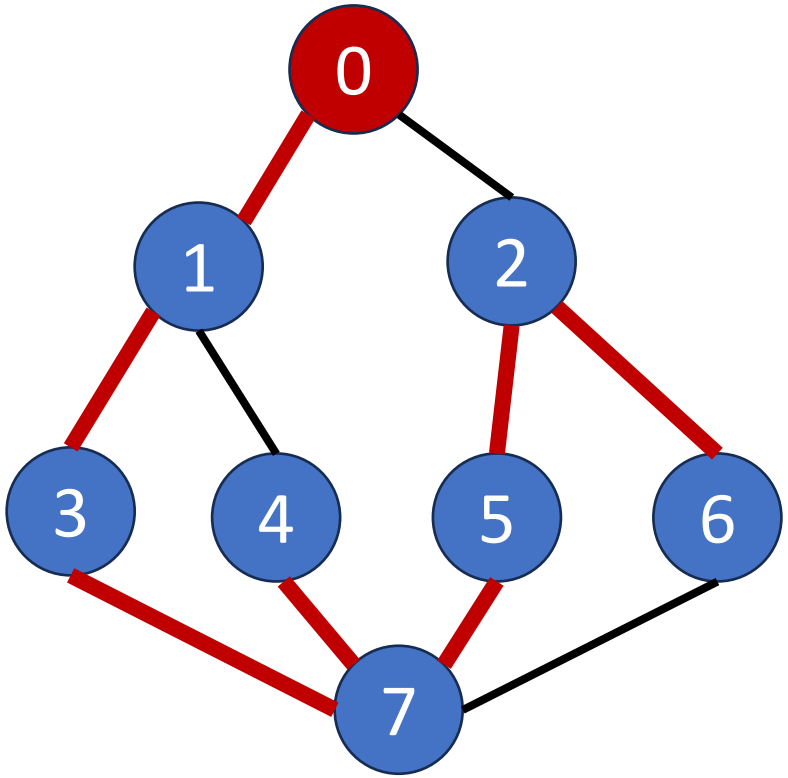
Applications: connected components



Depth-first search (or breadth-first search) from vertex 0.

Vertices 8, 9, and 10 are unvisited. ➔ The graph is not connected.

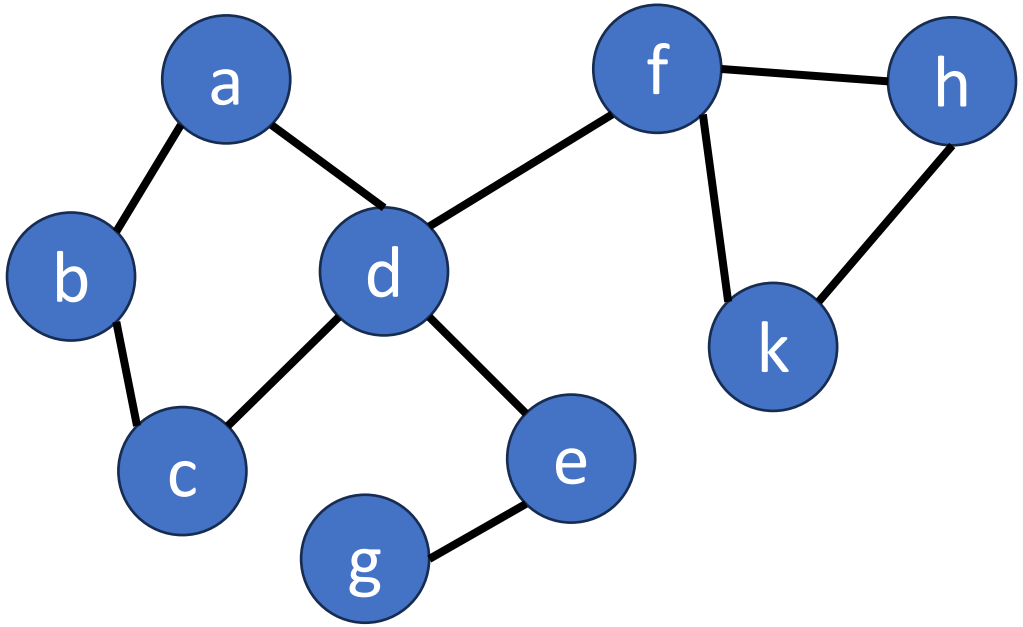
Applications: spanning tree



Depth-first search from vertex 0.

Generating a depth-first spanning tree.

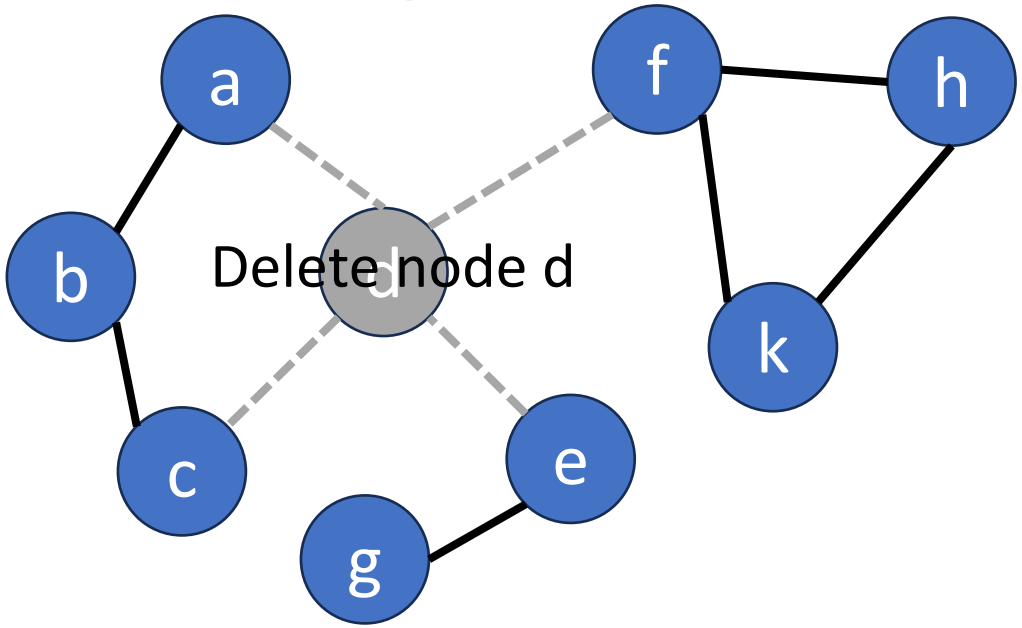
Applications: articulation points

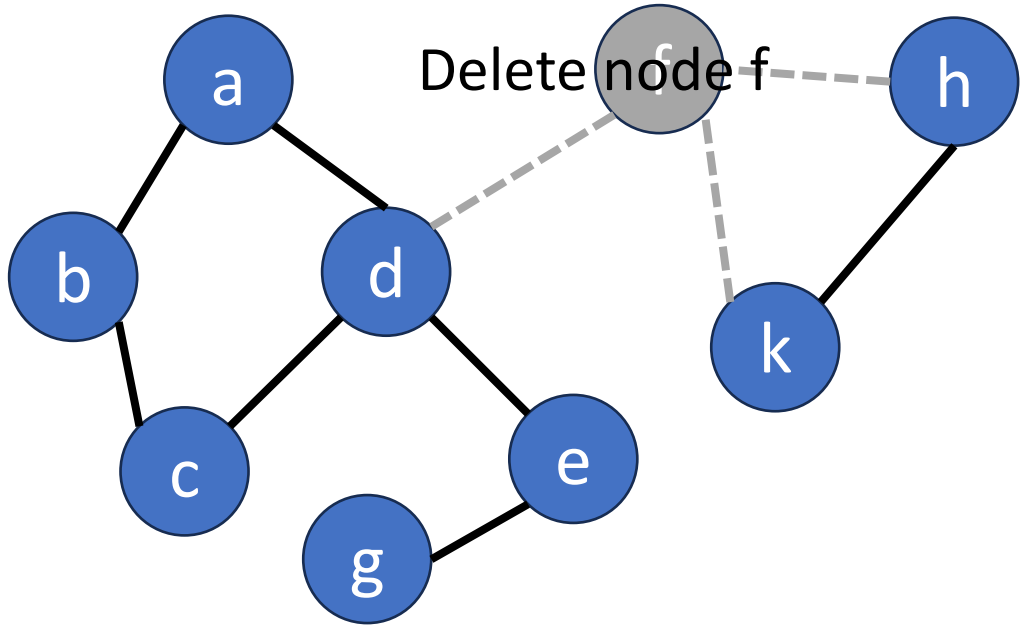


Articulation point:

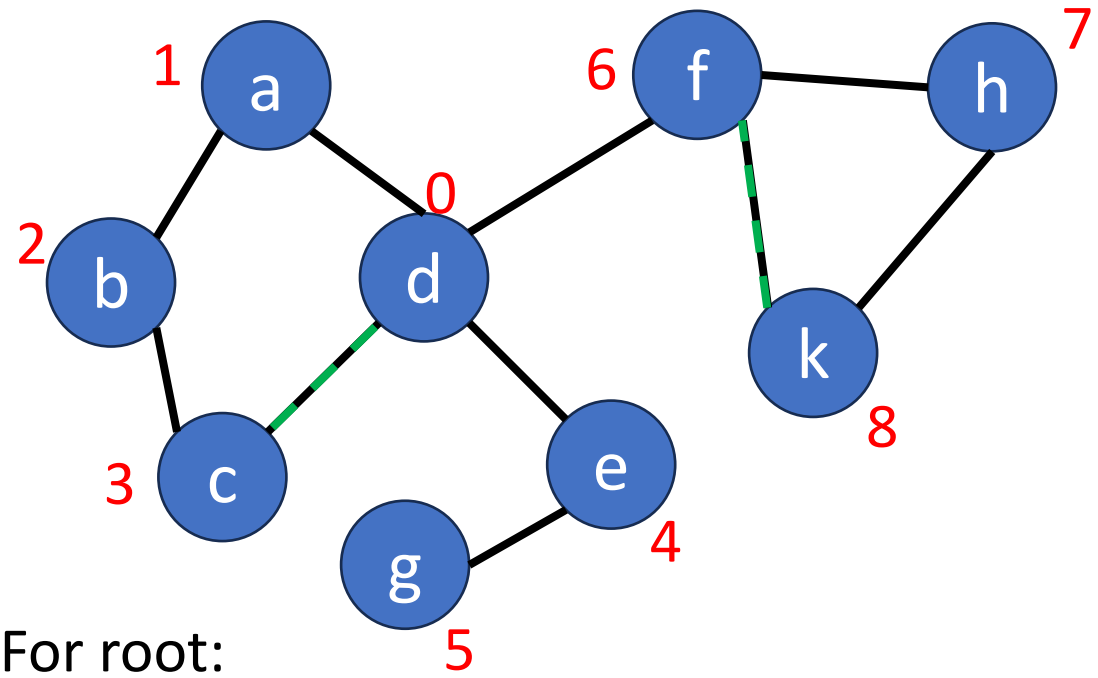
If a vertex is deleted, at least 2 connected components are produced.

For example, node d and node f.

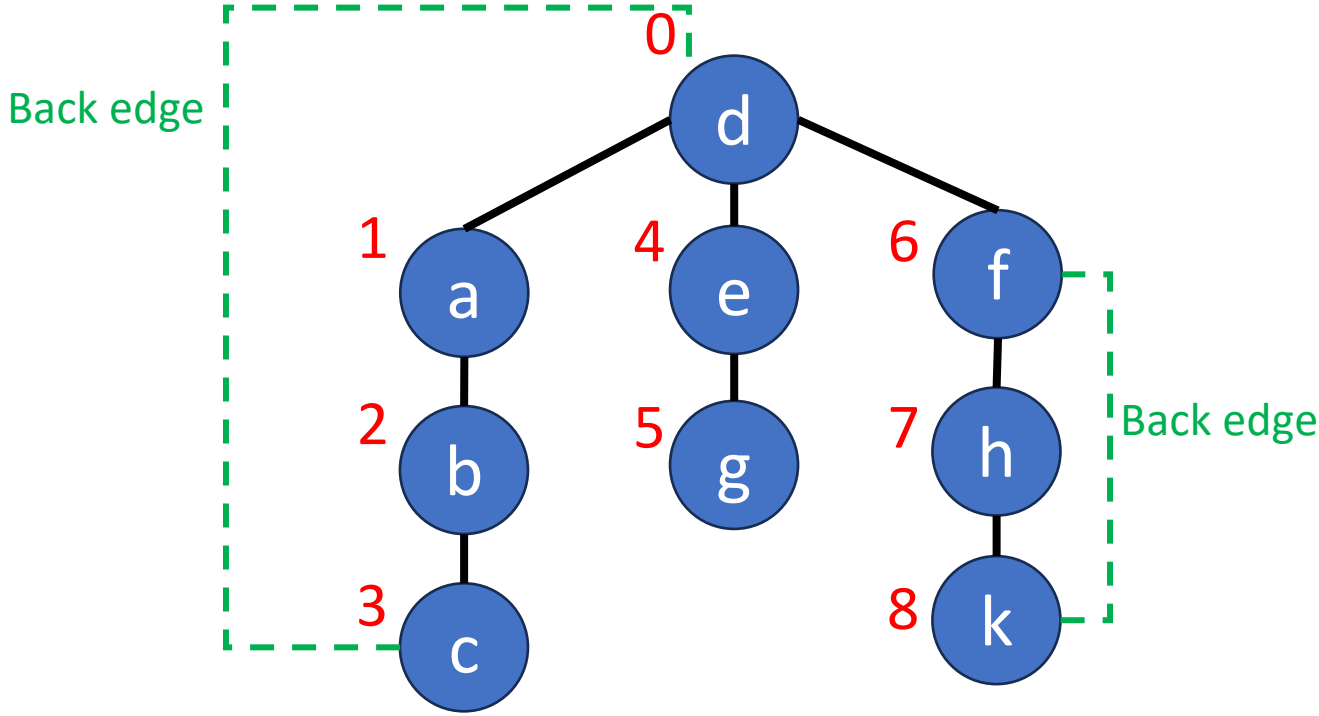




Applications: articulation points



Generating a depth-first search spanning tree. Ex: dfs(d)



- For root:

  Number of children of root ≥ 2

  → Root is an articulation point.

- For a non-root vertex v:

  A child of vertex v cannot reach any ancestor of vertex v via other paths.

  → Vertex v is an articulation point.

Summary

- Depth first search
- Breadth first search
- Applications:
  - spanning tree, connected components, and articulation points