Eskil Digernes, Sondre With, Torjus With

Assignment 6

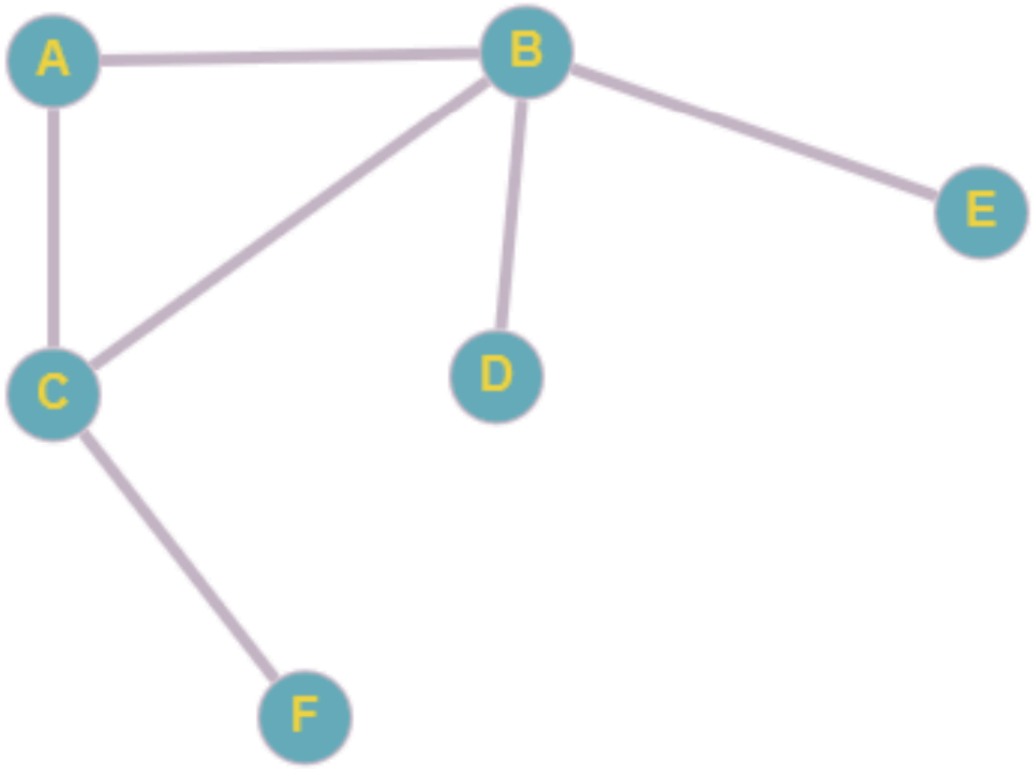
Exercise 1

1 → 2 → 3
2 → 1 → 4 → 5
3 → 1 → 6 → 7
4 → 2
5 → 2
6 → 3
7 → 3

Give an equivalent adjacency-matrix representation.

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	1	0	0	0	0
2	1	0	0	1	1	0	0
3	1	0	0	0	0	1	1
4	0	1	0	0	0	0	0
5	0	1	0	0	0	0	0
6	0	0	1	0	0	0	0
7	0	0	1	0	0	0	0

Exercise 2



Note: Consider node A as a start node; the alphabetical order is prioritized in visiting nodes.

2.1

Traverse each node in the graph using Breadth First Search traversal. Show the detailed steps using a Queue. (1 point)

Step 1: Start with node A. Enqueue A and mark as visited. Queue: [A]
Step 2: Dequeue A, and enqueue its neighbors B and C in alphabetical order. Queue: [B, C]
Step 3: Dequeue B, and enqueue its neighbor D and E. Mark B as visited. Queue: [C, D, E]
Step 4: Dequeue C, and enqueue its neighbors F. Queue: [D, E, F]
Step 5: Dequeue D. D has no unvisited neighbors. Queue: [E, F]
Step 6: Dequeue E. E has no unvisited neighbors. Queue: [F]
Step 7: Dequeue F. F has no unvisited neighbors.

Queue is now empty.

Traversal order: A, B, C, D, E, F

2.2

Traverse each node in the graph using Depth First Search traversal. Show the detailed steps using a Stack. (1 point)

Step 1: Start with node A. Push A onto the stack and mark it as visited. Stack: [A]
Step 2: Pop A, push its neighbors B, C, and F onto the stack. Stack: [B, C, F]

Step 3: Pop B, and push its neighbors D and E onto the stack. Stack: [C, F, D, E]
Step 4: Pop C. C has no unvisited neighbors. Stack: [F, D, E]
Step 5: Pop F. F has no unvisited neighbors. Stack: [D, E]
Step 6: Pop D. D has no unvisited neighbors. Stack: [E]
Step 7: Pop E. E has no unvisited neighbors.

Stack is now empty.

Traversal order: A, B, C, F, D, E