1. Consider the following undirected graph:

(a) Give the adjacency MATRIX representation of the graph in Figure 1;
(b) Give the adjacency LIST representation of the graph in Figure 1.

2. Consider the following directed graph:

(a) Draw the depth first search trees produced when depth first search is applied to the vertices of the graph in Figure 1 in the order 1,2,3,4,5,6,7,8,9,10,11,12.
(b) Give the start and finishing times when depth first search is applied to the vertices of the graph in Figure 1 in the order 1,2,3,4,5,6,7,8,9,10,11,12.
(c) List all the forward (non-tree) edges, all the back edges and all the cross edges when depth first search is applied to the vertices of the graph in Figure 1 in the order 1,2,3,4,5,6,7,8,9,10,11,12.
(d) Draw the depth first search trees produced when depth first search is applied to the vertices of the graph in Figure 1 in the order 12,11,10,9,8,7,6,5,4,3,2,1.
(e) Give the start and finishing times when depth first search is applied to the vertices of the graph in Figure 1 in the order 12,11,10,9,8,7,6,5,4,3,2,1.
(f) List all the forward (non-tree) edges, all the back edges and all the cross edges when depth first search is applied to the vertices of the graph in Figure 1 in the order 12,11,10,9,8,7,6,5,4,3,2,1.
3. Can the vertices of the graph in Figure 3 be topologically sorted?
   - If your answer is yes, give the graph vertices in a topologically sorted order.
   - If your answer is no, explain why not.

![Figure 3](image)

4. Can the vertices of the graph in Figure 4 be topologically sorted?
   - If your answer is yes, give the graph vertices in a topologically sorted order.
   - If your answer is no, explain why not.

![Figure 4](image)

5. The total weight of a vertex \( v_i \in G \) is the sum of the edge weights of edges incident on \( v_i \). The following algorithm computes the total weight of the vertices of a graph. Assume the graph is represented as an adjacency list. Analyze the running time of this algorithm in terms of the number of vertices \( n \) and the number of edges \( m \). (Justify your answer. Do not simply give a running time.)

   ```
   procedure ComputeTotalWeight(G)
   1   foreach \( v_i \in V(G) \) do
   2       TotalWeight[\( v_i \)] \leftarrow 0;
   3   end
   4   foreach \( v_i \in V(G) \) do
   5       foreach edge \((v_i, v_j)\) incident on \( v_i \) do
   6           TotalWeight[\( v_i \)] \leftarrow TotalWeight[\( v_i \)] + \text{weight}(v_i, v_j);
   7       end
   8   end
   ```

6. Consider the following procedure whose input is a graph \( G \). Edges of \( G \) are represented by the adjacency MATRIX \( G.ADJ[i,j] \).
   (a) What is the maximum number of elements in queue \( Q \)? Give an exact number and justify your answer. (Give your answer in terms of the number of vertices \( n \) and the number of edges \( m \) in \( G \).)
   (b) Analyze the the asymptotic running time of Function \text{Func1} \text{ in terms of the number of vertices n and the number of edges m in G. (Justify your answer. Do not simply give a running time.)}
procedure Func1(G) /* Q is a priority queue implemented as a heap */
1 Q.Init();
2 foreach vertex v_i ∈ V(G) do
3     foreach vertex v_j ∈ V(G) do
4         if (G.Adj[i, j] == 1) then
5             x ← i * n + j;
6             Q.Insert(x);
7         end
8     end
9 end
10 while (Q.IsNotEmpty()) do
11     x ← Q.DeleteMin();
12     Print x;
13 end