*Owen Lindsey*

*Professor Demland, David*

*CST-201 Exercise 6*

*10/13/2024*

**Exercise 3.1 - 5**

*Algorithm Explanation*

1. *Take a boolean matrix A[0..n-1, 0..n-1] as an input.*
2. *Keep minimum size requirement n > 3.*
3. *Create separate methods to verify topology named* ***IsRing****,* ***IsStar****, and* ***IsFullyConnectedMesh****.*
4. ***IsRing method****:*  
    *1. Initialize a counter for each node's connections.*

*2. Iterate through each row of the matrix:*

*- Count the number of 'true' values (connections) in the row.*

*- If the count is not exactly 2, return false.*

*- If all rows have 2 connections, return true.*

*3. If all nodes have 2 connections, check for a single cycle:*

*- Start from any node and follow connections.*

*- Use a visited array to track traversed nodes.*

*- If all nodes are visited exactly once and we return to the start, it's a ring.*

*5.* ***IsStar method:***  
 *1. Initialize counters for center node and leaf nodes.*   
 *2. Iterate through each row of the matrix.*

*- Count the number of ‘true’ values (connections) in the row.*

*- Look for one row with n-1 connections (center node).*

*- Ensure all other rows have exactly one connection.*

*- If these conditions are met, it's a star.*

***Exercise 3.1 - 5***

*Algorithm Explanation*

*6.* ***IsFullyConnectedMesh method:***

1. *For each element A[ i ] [ j ] :*  
   *- if i ==j, check that A[ i ] [ j ] is false (no self-connections).*

*- If i != j, check that A[ i ] [ j ] is true (connected to all others).*

*2. If all these conditions are met, it's a fully connected mesh*

*7. If none of the above checks pass, the matrix doesn't represent any of the three topologies.*

***Exercise 3.1 - 5***

*Time Efficiency Explanation*

*The time efficiency class of this brute-force algorithm is O(n^2), where is the number of nodes in the network.*

* *For each topology check, we potentially examine every element of the n x n adjacency matrix.*
* *The worst-case scenario involves checking for all three topologies, each requires O(n^2) time.*

**Exercise Sorting “EXAMPLE”: Bubble Sort**  
  
*Algorithm Explanation*

1. *The starting string is a scrambled array “XAMPLE E”.*
2. *Repeatedly step through the list, compare elements and swap them if they are in the wrong order based on their position.*
3. *Example visualization of a bubble sort:*   
   *1. XAMPLE E*

*2. AMPLE EX*

*3.MPLE EXA*  
*4. PLE EXAM*  
*5. LE EXAMP*  
*6. E EXAMPL*  
*7. EXAMPLE (SOLVED)*

**Exercise Sorting “EXAMPLE”: Selection Sort**  
*Algorithm Explanation*

1. *The starting string is a scrambled array “XAMPLE E”*
2. *Find the first occurrence of that character in the remaining unsorted portion of the array.*
3. *Swap it with current position if necessary.*
4. *Example visualization of a selection sort:*  
   *1. XAMPLE E*

*2. EAMPLX E (E Moved to front)*

*3. EXMPLA E (X moved to second position)*

*4.* EXAPLM E (A moved to third position)

5. EXAMPL E (M moved to fourth position)

6. EXAMPL E (P already in correct position)

7. *EXAMPL E (L already in correct position)*

*8. EXAMPLE (E moved to last position SOLVED)*