**Rubric**

| **Building adjacency matrix** | **3** |
| --- | --- |
| **Taking correct parameters** | **1** |
| **Correctly traversing the adjacency matrix** | **4** |
| **Building proper logic with handling corner case** | **6** |
| **Returning correct output** | **1** |
| **Total** | **15** |

**SET A**

| def helper(graph, start, nextNode, destination):  if nextNode == len(graph):  if graph[start][destination] == 1:  return True  return False   if graph[start][nextNode] != 1:  return False   return helper(graph, start+1, nextNode+1, destination)  def wantToReturnStart(graph):  return helper(graph, 0, 1, 0)  graph = [  [0, 1, 0, 0, 0],  [0, 0, 1, 0, 0],  [0, 0, 0, 1, 0],  [0, 0, 0, 0, 1],  [1, 0, 0, 0, 0] ]  print(want\_to\_return\_start(graph)) # Output: True | public class GraphHelper {    public static boolean helper(int[][] graph, int start, int nextNode, int destination) {  if (nextNode == graph.length) {  return graph[start][destination] == 1;  }   if (graph[start][nextNode] != 1) {  return false;  }   return helper(graph, start + 1, nextNode + 1, destination);  }   public static boolean wantToReturnStart(int[][] graph) {  return helper(graph, 0, 1, 0);  }   public static void main(String[] args) {  int[][] graph = {  {0, 1, 0, 0, 0},  {0, 0, 1, 0, 0},  {0, 0, 0, 1, 0},  {0, 0, 0, 0, 1},  {1, 0, 0, 0, 0}  };  System.out.println(wantToReturnStart(graph)); // Output: true  } } |
| --- | --- |

**SET B**

| def helper(graph, start, nextNode, destination):  if nextNode < len(graph):  if graph[start][destination] == 1:  return True   if graph[start][nextNode] != 1:  return False   return helper(graph, start+1, nextNode+1, destination)  return False  def reachingToFinalDestination(graph):  return helper(graph, 0, 1, 4)  graph = [  [0, 1, 0, 0, 0],  [0, 0, 1, 0, 0],  [0, 0, 0, 1, 0],  [0, 0, 0, 0, 1],  [0, 0, 0, 0, 0] ]  print(reaching\_to\_final\_destination(graph)) # Output: True | public class GraphHelper {   public static boolean helper(int[][] graph, int start, int nextNode, int destination) {  if (nextNode < graph.length) {  if (graph[start][destination] == 1) {  return true;  }   if (graph[start][nextNode] != 1) {  return false;  }   return helper(graph, start + 1, nextNode + 1, destination);  }  return false;  }   public static boolean reachingToFinalDestination(int[][] graph) {  return helper(graph, 0, 1, 4);  }   public static void main(String[] args) {  int[][] graph = {  {0, 1, 0, 0, 0},  {0, 0, 1, 0, 0},  {0, 0, 0, 1, 0},  {0, 0, 0, 0, 1},  {0, 0, 0, 0, 0}  };   System.out.println(reachingToFinalDestination(graph)); // Output: true  } } |
| --- | --- |