from pptx import Presentation

from pptx.util import Inches

# Create a presentation object

prs = Presentation()

# Title Slide

slide = prs.slides.add\_slide(prs.slide\_layouts[0])

title = slide.shapes.title

subtitle = slide.placeholders[1]

title.text = "Graph Algorithms Quiz"

subtitle.text = "Test your knowledge on Graphs, MST, BFS, DFS, and more!\nAuthor: Your Name\nDate: Today's Date"

# Instructions Slide

slide = prs.slides.add\_slide(prs.slide\_layouts[1])

title = slide.shapes.title

content = slide.placeholders[1]

title.text = "Instructions"

content.text = (

"Answer the following questions to the best of your ability.\n"

"Each question has multiple-choice answers.\n"

"Select the correct answer before moving to the next question."

)

# Questions and Answers

questions = [

{

"question": "Which algorithm is used to find the Minimum Spanning Tree of a graph?",

"options": ["A. Dijkstra's Algorithm", "B. Prim's Algorithm", "C. Floyd-Warshall Algorithm", "D. Bellman-Ford Algorithm"],

"answer": "B. Prim's Algorithm"

},

{

"question": "What is the time complexity of Kruskal's algorithm?",

"options": ["A. O(V^2)", "B. O(E log E)", "C. O(V log V)", "D. O(V + E)"],

"answer": "B. O(E log E)"

},

{

"question": "In BFS, which data structure is used to keep track of the vertices to be explored?",

"options": ["A. Stack", "B. Queue", "C. Priority Queue", "D. Linked List"],

"answer": "B. Queue"

},

{

"question": "Which of the following is a property of a DFS traversal?",

"options": ["A. It uses a queue", "B. It explores all neighbors at the present depth first", "C. It finds the shortest path", "D. It always finds the minimum spanning tree"],

"answer": "B. It explores all neighbors at the present depth first"

},

{

"question": "What is the purpose of the union-find data structure in Kruskal's algorithm?",

"options": ["A. To sort the edges", "B. To manage connected components and detect cycles", "C. To find the shortest path", "D. To keep track of node degrees"],

"answer": "B. To manage connected components and detect cycles"

},

{

"question": "In Prim's algorithm, which data structure is commonly used to select the next edge with the minimum weight?",

"options": ["A. Stack", "B. Queue", "C. Priority Queue", "D. Linked List"],

"answer": "C. Priority Queue"

},

{

"question": "Which of the following graph representations is typically used for sparse graphs?",

"options": ["A. Adjacency Matrix", "B. Adjacency List", "C. Incidence Matrix", "D. Edge List"],

"answer": "B. Adjacency List"

},

{

"question": "What is the time complexity of BFS for a graph with V vertices and E edges?",

"options": ["A. O(V^2)", "B. O(V + E)", "C. O(E log V)", "D. O(V log V)"],

"answer": "B. O(V + E)"

},

{

"question": "Which algorithm can detect a cycle in an undirected graph?",

"options": ["A. Kruskal's Algorithm", "B. Prim's Algorithm", "C. DFS", "D. BFS"],

"answer": "C. DFS"

},

{

"question": "In an adjacency matrix representation of a graph, what does a non-zero value at matrix[i][j] indicate?",

"options": ["A. There is no edge between vertex i and vertex j", "B. There is an edge between vertex i and vertex j", "C. Vertex i and vertex j are the same", "D. Vertex i has the minimum degree"],

"answer": "B. There is an edge between vertex i and vertex j"

},

{

"question": "What is the purpose of path compression in the union-find algorithm?",

"options": ["A. To reduce the depth of the tree", "B. To keep track of the number of edges", "C. To find the shortest path", "D. To sort the vertices"],

"answer": "A. To reduce the depth of the tree"

},

{

"question": "In which case does Kruskal's algorithm become less efficient compared to Prim's algorithm?",

"options": ["A. When the graph is dense", "B. When the graph is sparse", "C. When the graph has many cycles", "D. When the graph is directed"],

"answer": "A. When the graph is dense"

},

{

"question": "Which data structure is used to implement DFS?",

"options": ["A. Stack", "B. Queue", "C. Priority Queue", "D. Linked List"],

"answer": "A. Stack"

},

{

"question": "What is the primary difference between Prim's and Kruskal's algorithms?",

"options": ["A. Prim's algorithm builds the MST one vertex at a time, Kruskal's one edge at a time", "B. Prim's algorithm is for directed graphs, Kruskal's for undirected", "C. Prim's algorithm uses a queue, Kruskal's uses a stack", "D. Prim's algorithm sorts edges first, Kruskal's doesn't"],

"answer": "A. Prim's algorithm builds the MST one vertex at a time, Kruskal's one edge at a time"

},

{

"question": "Which algorithm is typically more efficient for a graph with a large number of edges (dense graph)?",

"options": ["A. Prim's Algorithm", "B. Kruskal's Algorithm", "C. BFS", "D. DFS"],

"answer": "A. Prim's Algorithm"

},

{

"question": "In an adjacency list representation, what does each vertex store?",

"options": ["A. A list of all other vertices", "B. A list of edges with weights", "C. A list of adjacent vertices", "D. A list of visited vertices"],

"answer": "C. A list of adjacent vertices"

},

{

"question": "Which algorithm uses a priority queue to find the shortest path in a graph?",

"options": ["A. Prim's Algorithm", "B. Kruskal's Algorithm", "C. BFS", "D. Dijkstra's Algorithm"],

"answer": "D. Dijkstra's Algorithm"

},

{

"question": "What is the time complexity of DFS for a graph with V vertices and E edges?",

"options": ["A. O(V^2)", "B. O(V + E)", "C. O(E log V)", "D. O(V log V)"],

"answer": "B. O(V + E)"

},

{

"question": "Which of the following is a key step in Kruskal's algorithm?",

"options": ["A. Select the minimum edge and add it to the MST without checking for cycles", "B. Select the minimum edge that does not form a cycle", "C. Select the maximum edge and add it to the MST", "D. Select the maximum edge that does not form a cycle"],

"answer": "B. Select the minimum edge that does not form a cycle"

},

{

"question": "In BFS, what happens when a vertex is visited?",

"options": ["A. It is added to a stack", "B. It is added to a queue", "C. It is marked as visited and its adjacent vertices are added to the queue", "D. It is removed from the graph"],

"answer": "C. It is marked as visited and its adjacent vertices are added to the queue"

},

{

"question": "What is the main difference between an adjacency list and an adjacency matrix?",

"options": ["A. An adjacency list is used for dense graphs, an adjacency matrix for sparse graphs", "B. An adjacency list stores edges, an adjacency matrix stores vertices", "C. An adjacency list uses more space than an adjacency matrix", "D. An adjacency list is more space-efficient for sparse graphs, an adjacency matrix is more space-efficient for dense graphs"],

"answer": "D. An adjacency list is more space-efficient for sparse graphs, an adjacency matrix is more space-efficient for dense graphs"

},

{

"question": "Which algorithm can be used to find connected components in an undirected graph?",

"options": ["A. Kruskal's Algorithm", "B. Prim's Algorithm", "C. DFS", "D. Dijkstra's Algorithm"],

"answer": "C. DFS"

},