Problem 1) Depth-first Search
Run a Depth-first Search on the following tree.

Assume the adjacency lists are sorted alphabetically on the node labels. For example, traverse node b before node e, because ‘b’ comes earlier in the alphabet than ‘e’.

Indicate which edges are traversed by drawing them with bold lines. Indicate the order in which the vertices are visited. Mark the starting node with a ‘1’, the second node with a ‘2’ and so on.

1a. (5pts) Start the search at node a.
1b. (5pts) Start the search at node l.

Problem 2) Breadth-first Search
Run a Breadth-first Search on the tree used in Problem 1.

Assume the adjacency lists are sorted alphabetically on the node labels. For example, traverse node b before node e, because ‘b’ comes earlier in the alphabet than ‘e’.

Indicate which edges are traversed by drawing them with bold lines. Indicate the order in which the vertices are visited. Mark the starting node with a ‘1’, the second node with a ‘2’ and so on. Note: This is NOT the same as d[v]. The nodes should be marked 1 through 12 based on the order they became “visited”.

2a. (5pts) Start the search at node a.
2b. (5pts) Start the search at node l.
Problem 3) Use Dijkstra’s algorithm to find the shortest path between two nodes. Indicate which edges are traversed by drawing them with bold lines. Indicate the order in which the vertices are visited. Mark the starting node with a ‘1’, the second node with a ‘2’ and so on. In this question ‘visited’ means that the vertex is the “active” search node – in the pseudocode presented in class, that would be node \( u \).

![Graph Image](image)

3a. (10pts) Find the shortest distance between \( a \) and \( j \)
3b. (10pts) Find the shortest distance between \( e \) and \( d \)

Problem 4) (10 pts) Represent the white regions in this map as a graph.
Problem 5) (50 pts) There are two types of professional wrestlers: “good guys” and “bad guys”. Between any pair of professional wrestlers, there may or may not be a rivalry. Suppose that we have $n$ professional wrestlers and we have a list of $r$ pairs of wrestlers for which there are rivalries. Give an $O(n + r)$ algorithm that determines whether it is possible to designate some of the wrestlers as good guys and the remainder as bad guys such that each rivalry is between a good guy and a bad guy. If it is possible to perform such a designation, your algorithm should produce it.

Generate pseudocode for this.

Think about how you would represent this problem as a graph. Consider how you would use one of the graph traversal techniques to solve this problem. Once you have described how you represent the problem as a graph, in plain language, you may use graph operations over this structure in your pseudocode.