In either Java or Python, implement both a breadth-first search algorithm and a depth-first search algorithm. Each of these functions should take as input:

1) A graph such as those in exercise 5-1 (p. 185 in the book). You can use any one of the two graphs as your input.

2) A starting node (for example, A).

Return a list of vertices encountered in the traversal. Break all ties by picking the vertices in alphabetical order. For example, if you input the left graph, from node A, you should traverse node B before node D. Zip your code and submit it in the Canvas.

NOTE:

- You can use either an adjacency matrix or an adjacency list as the graph data structure. You can use the built-in List, ArrayList of Java and Python to implement the adjacency list.
- You can use the built-in Stack or Queue data structure of Java and Python.
- It's not enough to mindlessly type lines of code. You need to make sure you understand what each line of code is doing. You will be quizzed about it.