Exercise 2

0/10 points (graded)

Consider our representation of permutations of students in a line from Exercise 1. (The teacher only swaps the positions of two students that are next to each other in line.) Let's consider a line of three students, Alice, Bob, and Carol (denoted A, B, and C). Using the Graph class created in the lecture, we can create a graph with the design chosen in Exercise 1: vertices represent permutations of the students in line; edges connect two permutations if one can be made into the other by swapping two adjacent students.

We construct our graph by first adding the following nodes:

nodes = []

nodes.append(Node("ABC")) # nodes[0]

nodes.append(Node("ACB")) # nodes[1]

nodes.append(Node("BAC")) # nodes[2]

nodes.append(Node("BCA")) # nodes[3]

nodes.append(Node("CAB")) # nodes[4]

nodes.append(Node("CBA")) # nodes[5]

g = Graph()

for n in nodes:

g.addNode(n)

Add the appropriate edges to the graph.

[Hint: How to get started?](https://courses.edx.org/xblock/block-v1:MITx+6.00.2x+1T2021+type@vertical+block@d5e598486ce54ba6ac645fc45c62d72a?show_title=0&show_bookmark_button=0&recheck_access=1&view=student_view&format=Lecture%20Sequence)

Write your code in terms of the nodes list from the code above. For each node, think about what permutation is allowed. A permutation of a set is a rearrangement of the elements in that set. In this problem, you are only adding edges between nodes whose permutations are between elements in the set beside each other . For example, an acceptable permutation (edge) is between "ABC" and "ACB" but not between "ABC" and "CAB".

1

Code goes here

g.addEdge(Edge(nodes[0], nodes[1]))

g.addEdge(Edge(nodes[0], nodes[2]))

g.addEdge(Edge(nodes[1], nodes[4]))

g.addEdge(Edge(nodes[2], nodes[3]))

g.addEdge(Edge(nodes[3], nodes[5]))

g.addEdge(Edge(nodes[4], nodes[5]))

# or some variation on this. Obviously, in a Graph,

# g.addEdge(Edge(nodes[0], nodes[1])) functions just as well as

# g.addEdge(Edge(nodes[1], nodes[0])).