Bipartite Graphs and Matchings

The purpose of this project is for you to write code that tests whether or not a given bipartite graph has a perfect matching. In addition, you will write code that determines whether or not a graph is bipartite. The ONLY IMPORT ALLOWED is 'copy', and you are ONLY allowed to use the copy.deepcopy() method from this package. All of your code should be 'from scratch.'

Objectives

You are to write a function `power(list)` that takes in a list as its input, and then returns the power set of that list. You may assume that the input will not have any duplicates (i.e., it's a set).

- `power`(["A", "B"]) should return [[], ["A"], ["B"], ["A", "B"]].
- `power`(["A", "B", "C"]) should return [[], ["A"], ["B"], ["C"], ["A", "B"], ["A", "C"], ["B", "C"], ["A", "B", "C"]]

You are to write a function `partite_sets(graph)` that takes in a BIPARTITE graph as its input, and then returns the partite sets of the graph (the order of the sets outputed does not matter). For ease, output these sets as lists.

- `partite_sets`({"A" : ["B", "C"], "B" : ["A"], "C" : ["A"]}) should return ["A"], ["B", "C"] (or ["B", "C"], ["A"])
- `partite_sets`({"A" : ["B", "C"], "B" : ["A", "D"], "C" : ["A", "D"], "D" : ["B", "C"]}) should return ["A", "D"], ["B", "C"] (or ["B", "C"], ["A", "D"])

You are to write a function `is_bipartite(graph)` that takes in a graph as its input, and then determines whether or not the graph is bipartite. In other words, it returns True if it is, and False if it is not.

- `is_bipartite`({"A" : ["B", "C"], "B" : ["A"], "C" : ["A"]}) should return True
- `is_bipartite`({"A" : ["B", "C"], "B" : ["A", "C"], "C" : ["A", "B"]}) should return False

You are to write a function `is_perfect(graph)` that takes in a BIPARTITE graph as its input, and then determintes whether or not the graph has a perfect matching. In other words, it will return the Boolean value True if it has one, and False if it does not.

- `is_perfect`({"A" : ["B", "C"], "B" : ["A", "D"], "C" : ["A", "D"], "D" : ["B", "C"]}) should return True
- `is_perfect`({"A" : ["B", "C"], "B" : ["A"], "C" : ["A"]}) should return False

Grading Rubric

Your functions will be tested using a collection of pre-made test cases that I will create. Your grade will be based on how often your code produces correct results and on the quality of the descriptions that you provide for your functions. Full credit will be given to a notebook whose functions work 100% of the time. Also each function has a description that clearly explains how the function works.