Homework 5 - Problem 3

This algorithm runs through a set of classes containing all its prerequisite courses and counts the number of them.

```
Let int V = num Vertices
Let set Adj be a graph of all the values (2D Array of Values and their pairs)

findCount:
    Let set visited = new set of booleans of size V
    Let set maxValues = new set of numbers of size V

    for every value < V in visited:
        If value was NOT in visited:
                DFS(v, visited, maxValues)
    Let maxValue = 0
    for every value v in MaxVlue:
        Let maxValue = MAX between MaxValue and value v

DFS:
    Given v is the vertex number
    Given set visited (from above)
    Given set maxValues (from findCount)

    Set value in visited at v to true;
    If value in adj at column 0 is 0, change maxValues at value v to 1

    For every value w in V (from above):
        Let neighbour = value in adj
        If neighbour = 0 then break
        Else if visited at neighbour is false:
                DFS(neighbour, visited, maxValues)
        Set maxValues = MAX between MaxValue at v and MaxValues at neighbour + 1

main:
    Given number of classes n

    Let set adj = new set of size n * n
    For every class inputted, add all prerequisites to adj

    findCount()
```

This works because:

It finds a maximum for every single class, this meaning that each class will return the proper among of prereqs

Time Complexity: O(m+n)

This is because you use DFS to find the results, DFS is m+n and when DFS is done, the algorithm to find the max number is O(n)