Josh Barber Assignment 11 March 29, 2023

Pseudo code:

```
CliqueCheck(connections, adjacentConnections):
    for u in connections
        for v in adjacentConnections
            if u = v
                CliqueArray push u

    return CliqueArray

MaxCliques(Graph)
    max = 0
    for u in Graph
        connections = all adjacent vertices to u + u
        connections.sort()
        count = 0
        for v in connections
            if v != u:
                adjacentConnections = all adjacent vertices to v + v
                count++
                CliqueArray = CliqueCheck(connections, adjacentConnections)
                if (count = len(CliqueArray) and max < len(CliqueArray)
                        max = len(CliqueArray)
                        CliqueVertices = CliqueArray
    return CliqueVertices
```

I brute force the max clique problem. For every vertice within the graph, the algorithm checks all adjacent vertices for that vertice, and creates an array (collection) of all adjacent vertices, plus the vertice checked against. With all adjacent vertices tracked (collection), the algorithm tracks all adjacent vertices to the already tracked adjacent vertices and throws it into an array (adjCollection). It then compares the two arrays, collection and adjCollection. If there are say 3 matches of vertices in the two arrays, no matter the size. There would also need to be 3 adjacent arrays created with the same vertice matches. If this is true, a clique has formed. The algorithm then keeps track of the biggest clique. The runtime analysis is $O(2^n \cdot n^2)$