HW5

Shane Drafahl

31 October,2017

1. This algorithm goes over every node in the graph and checks the neighbors of the neighbors and populates a adjacency list for nodes of distance 2 from each other. After this the adjacency list is then set to what the new adjacency list is used.

```
// Given a graph G createG2 converts G to G2 graph
function createG2(G) {
    foreachNode(G as N) {
        foreachNeighbor(N as N1) {
            foreachNeighbor(N1 as N2) {
                append(N.N2, N2);
            }
        }
    }
    foreachNode(G as N) {
        N.N = N.N2;
    }
}

Node {
    visited // by default this is false
    N[] // adjacency list
    N2[] // adjacency list for G squared
}

Graph {
    N[] // List of nodes
```

```
}
```

```
// Given a adjacency matrix where (x, y) and x is the domain and the y is th
// there is a function between different vertices to other vertices.
function createG2(M) { // suppose that M is a adjacency matrix.
       M2 // new Matrix of equal size and width of M
       x = 0
       for(x in range(M.height)) {
           y = 0
           for(y in range(M.width)) {
               if(M[x][y] == 1) {
                   a = 0
                   for(a in range(M.width)) {
                       if(M[y][a] == 1) {
                           M2[x][a] = 1
                       }
                   }
               }
           }
       }
       return M2
}
```

This algorithm goes over every index of the matrix which is $O(n^2)$ where n is the number of vertices. If it finds a connection it has another for loop an then updates the initial row. Worst case scenario this is a fully connected graph so $O(n^3)$ best case the graph has no edges or connections it would be $O(n^2)$.