The square of a directed G=(V,E) is the graph G2=(V,E2) such that <u,v>∈E2 if and only if G contains a path with at most two edges between *u* and *v*. Describe efficient algorithms for computing G2 from G for the adjacency list and adjacency matrix representations for G . Analyze the running time of your algorithm.

Adjacency Matrix:

In order to compute the adjacency matrix for G^2, you would need to square the adjacency matrix of G.

```
For i = 1 to n
   For j = 1 to n
      G2[i][j] = 0 # Create adjacency matrix G^2
For i = 1 to n
   For j = 1 to n
      If G[i][j] == 1
         For k = 1 to n
            If (g[j][k] == 1)
               G2[i][k] == 1
```

The first 2 loops will run with complexity O(n^2). The next 2 loops will also run in O(n^2), but with the added n iterations to test all edges needed. The final running time of this algorithm will be O(n^3).

Adjacency List:

```
For vertex in Adjacency List U
   For vertex in Adjacency List V
      If edge(u, v) exists in E^2
         Insert v into Adjacency List^2
```

For every edge in the Adjacency List U, we can up to v vertexes. The final running time of this algorithm is O(V*E).