Trevon and Gabon Williams
Project 3

<u>Useful information</u>

- For the routing tables, we used dictionaries
- We randomly generated a number to choose when we would change the cost to a node and randomly selected which node to apply that cost
- Python v3 was used for this good and should be backwards compatible

For this assignment, we used python to implement a distance vector algorithm to find the shortest path to a given router based upon the routing tables given from the sending router's neighbors routing table.

We ran into a couple of problems when implementing this algorithm which resulted in us not be able to query to the cost to the correct nearest hop for some cases (it works 60% of the time). The next node query also has some mistakes (works 80% of the time). Since we did not build a graph to optimize the program and eliminate the recursive updating problem, our code uses the tables sent over a UDP socket to find the shortest path to a specified node.The shortest path cost is optimized for each run if the shortest path is not already found.

You will see when running the code that for example when node A is looking for the shortest path to node F, if node E does not start broadcasting the route is not the shortest path, but updates when all nodes broadcast.

<u>Notes</u>

We felt that honestly was the best policy, and we worked really hard trying to wrap our mind around the problem without rewriting the code a third time. We have a prototype that builds a graph to find the shortest path, but we found that we were stuck in a recursive loop when trying to implement the challenge.

Running the program:

```
python main
```