• Essentially I used a for loop with counter variable “i” and counter variable “j” to assign the nodes and figure out which nodes are neighbors with each other. Then I had a do/while loop with another nested for loop that iterates through the list of nodes and figures out which node is the others neighbor. Then it creates a distance vector packet. From there I have a for loop with counter variable “k” that takes the distance vector packet then goes through the list of nodes to determine where the packet needs to go to. Then the for loops with the counter variables “m” and “n” compare the routing tables of the distance vector packet and the node that it was sent to and assigns the cost of the link that is between those two nodes. I have another for loop with the same counter variables (“m” and “n”) that essentially do the same thing however they check to see if the cost is higher or lower so that way we can calculate the shortest path. The primary data structure used would be a linked list since I felt it was most comparable to that of a router. With each node being able to be a spot on the list being connected to with its edges that are used to determine the cost.

Disclaimer:

I believe the destination/cost columns are accurate though my “next hop” is not accurate. For some of the routing tables all 3 columns are correct, however for some of the elements in the “next hop” column they are inaccurate. I believe what is happening is a neighbor node is updating its neighbor and letting it know it has a faster route to another node that is far away - but in actuality the fastest route isn’t what is being displayed. I have tried a few things but unfortunately was unable to turn in a perfectly working program due to work. Will budget my time better for the next project.

To compile my code:

To compile:

g++ main.cpp

To execute:

./a.out

Then you will be prompted to enter the topology file name and its extension

Ex: “topology1.txt”