Experiment 3: Neighbor Table Determination

Aim: To create neighbor table for a given network topology

Objective: After carrying out this experiment, students will be able to:

- Generate neighbor table for all the nodes in a given topology.
- Analyze how this is useful in the process of routing data

Problem statement: You are required to write a program that calculates neighbor table for all the nodes in a given network. Consider a network with 10 nodes that is deployed in an area of 500 m^2. Your program should initially determine the distance between each node and all other nodes. Then the range of the nodes is given as input to the user. Using this range information, determine the neighbors of all the nodes.

Analysis: While analyzing your program, you are required to address the following points:

- How this is useful in the process of routing data?
- For a 3D topology, how would your program need to be changed?

MARKS DISTRIBUTION

Component	Maximum Marks	Marks Obtained
Algorithm/Flowchart	7	
Program	7	
Results/ Documentation	6	
Total	**20**	

Submitted by: **Harshit Kumar**

Register No: **21ETMC412011**

1. Algorithm/Flowchart
   a) Import the Random and Math packages to perform the particular operations.
   b) Prompt the user for the range of X and Y co-ordinates.
   c) Create two list for X and Y and generate 10 random set of numbers in both the list.
   d) Print the random generated numbers in a form of table.
   e) Now calculate the distance between all the nodes using Euclidian Distance formula.
   f) Now create an empty list to store the values of the distance in that.
   g) Append the distance values in another empty list.
   h) Now represent the distance values in a table.
   i) Prompt the user to enter the range for the Neighbor table.
   j) Now **temp[i][j] <=the range entered by the user then**, print the corresponding nodes

2. Program

```
import random   #Package for generating random numbers in given range
import math     #Package for using mathematical formulas
x =int(input("enter the range of X co-ordinates"))
y = int(input("enter the range of Y co-ordination"))
XL = random.sample(range(0,50),10)     #Generates 10 random numbers in 0 to 50 range
YL = random.sample(range(0,10),10)     #Generates 10 random numbers in 0 to 10 range
print(XL)
print(YL)
temp = []    # Temporary list to store the values of distance formulas
for i in range(len(XL)):   # For loop to display the XL values in a tabular form
    print(i+1,end="            ")
print("\n")
for i in range(len(XL)):  # For loop to display the YL values in a tabular form
    print(i+1,end=" ")
    d = []   # d list stores the values of distances between all the nodes
    for j in range(len(YL)):
        d.append(math.sqrt((XL[i]-XL[j])**2 + (YL[i]-YL[j])**2))      # Euclidian Distance formula between two nodes
    temp.append(d)
    for k in d:
        print("%.5f"%k,end="     ")
    print("\n")
a = float(input("Enter the range for the Neighbour: "))       #Prompting the user to enter the value from which Neighbour table will be constructed
b = 0
for i in range(len(XL)):                #Making the Neighbour Table
    print("\n")
    print("The Neighbours"" "+str(i+1)+" ""are :",end=" ")
    for j in range(len(YL)):
        if temp[i][j]<=a:
            b = temp[i][j]
            print("{0}{1} ".format(j+1,b))
```


3. Results

Debug I/O | Exceptions | Python Shell | Messages | OS Commands

Debug I/O (stdin, stdout, stderr) appears below

```
enter the range of X co-ordinates50
enter the range of Y co-ordination10
[0, 31, 33, 20, 11, 40, 2, 42, 1, 41]
[1, 2, 9, 0, 7, 8, 3, 6, 4, 5]
```

	1	2	3	4	5	6	7	8	9	10
1	0.00000	31.01612	33.95585	20.02498	12.52996	40.60788	2.82843	42.29657	3.16228	41.19466
2	31.01612	0.00000	7.28011	11.18034	20.61553	10.81665	29.01724	11.70470	30.06659	10.44031
3	33.95585	7.28011	0.00000	15.81139	22.09072	7.07107	31.57531	9.48683	32.38827	8.94427
4	20.02498	11.18034	15.81139	0.00000	11.40175	21.54066	18.24829	22.80351	19.41649	21.58703
5	12.52996	20.61553	22.09072	11.40175	0.00000	29.01724	9.84886	31.01612	10.44031	30.06659
6	40.60788	10.81665	7.07107	21.54066	29.01724	0.00000	38.32754	2.82843	39.20459	3.16228
7	2.82843	29.01724	31.57531	18.24829	9.84886	38.32754	0.00000	40.11234	1.41421	39.05125
8	42.29657	11.70470	9.48683	22.80351	31.01612	2.82843	40.11234	0.00000	41.04875	1.41421
9	3.16228	30.06659	32.38827	19.41649	10.44031	39.20459	1.41421	41.04875	0.00000	40.01250
10	41.19466	10.44031	8.94427	21.58703	30.06659	3.16228	39.05125	1.41421	40.01250	0.00000

```
Enter the range for the Neighbour: 10

The Neighbours 1 are : 10.0
72.8284271247461903
93.1622776601683795

The Neighbours 2 are : 20.0
37.280109889280518

The Neighbours 3 are : 27.280109889280518
30.0
67.0710678118654755
89.486832980505138
108.94427190999916

The Neighbours 4 are : 40.0

The Neighbours 5 are : 50.0
79.848857801796104

The Neighbours 6 are : 37.0710678118654755
60.0
82.8284271247461903
103.1622776601683795

The Neighbours 7 are : 12.8284271247461903
59.848857801796104
70.0
91.4142135623730951

The Neighbours 8 are : 39.486832980505138
62.8284271247461903
80.0
101.4142135623730951

The Neighbours 9 are : 13.1622776601683795
71.4142135623730951
90.0

The Neighbours 10 are : 38.94427190999916
63.1622776601683795
81.4142135623730951
100.0
```


4. Analysis and Discussions

   [Answer:-1.] The above code is used to creating a Neighbor Table (or) A Routing Table which determines whether the given nodes are in the range or not with respect to all the nodes within an specified area or specified grid. This is achieved by using Euclidian Distance Formula which is:

   $\boldsymbol{D} = \sqrt{(\boldsymbol{x2} - \boldsymbol{x1})^2 + (\boldsymbol{y2} - \boldsymbol{y1})^2}$ (This is for the points in 2 dimension)

[Answer:-2] In the context of a 3D topology, where nodes are distributed in three-dimensional space, the program needs to be adapted to account for the additional dimension. Key considerations for modifying the program include:

- **Coordinate System Extension:** Nodes in a 3D topology would have coordinates (x, y, z). The program should be adjusted to handle and calculate distances in three dimensions.
- **Topological Representation:** Data structures representing the network's connections may need to be extended to accommodate the third dimension. This could involve using 3D arrays or more complex data structures to represent connectivity.
- **Traversal Algorithm Modification:** Algorithms for determining distances and identifying neighbors may need to be modified to traverse the 3D space. Adjustments to pathfinding algorithms should consider three-dimensional paths.

5. Conclusions

   In this we learnt what is a Routing Table and its implementation

6. Comments

   a. Limitations of the experiment
   b. Limitations of the results obtained

      The table is created on the given static data.
   c. Learning

      Implementation of Routing table.
   d. Recommendations