TW 3: Implement the distance vector routing algorithm

#include <stdio.h>

#define NODES 10

#define NO\_ROUTE 999

#define NO\_HOP 1000

int no;

struct node {

int a[NODES][4];

}router[NODES];

void init(int r) {

int i;

for (i = 1; i <= no; i++) {

router[r].a[i][1] = i;

router[r].a[i][2] = NO\_ROUTE;

router[r].a[i][3] = NO\_HOP;

}

router[r].a[r][2] = 0;

router[r].a[r][3] = r;

}

void inp(int r) {

int i;

printf("\nEnter distance from node %d to other nodes\n", r);

printf("Enter 999 if there is no direct route\n");

for (i = 1; i <= no; i++) {

if (i != r) {

printf("Enter distance to node %d: ", i);

scanf("%d", &router[r].a[i][2]);

router[r].a[i][3] = i;

}

}

}

void display(int r) {

int i;

printf("\nThe routing table for node %d is as follows", r);

for (i = 1; i <= no; i++) {

if (router[r].a[i][2] == 999)

printf("\n%d \t no link \t no hop", router[r].a[i][1]);

else

printf("\n%d \t %d \t %d", router[r].a[i][1], router[r].a[i][2], router[r].a[i][3]);

}

}

void dv\_algo(int r) {

int i, j, z;

for (i = 1; i <= no; i++) {

// r → source router

// i → step taken (via which router to reach the dest router)

// j → destination router

// cannot jump from the source router or to a router which is not reachable or from the source router

if (router[r].a[i][2] != 999 && router[r].a[i][2] != 0) {

for (j = 1; j <= no; j++) {

z = router[r].a[i][2] + router[i].a[j][2];

if (z < router[r].a[j][2]) {

router[r].a[j][2] = z;

router[r].a[j][3] = i;

}

}

}

}

}

int main() {

int i, j, x, y;

char choice = 'y';

printf("Enter the number of nodes: ");

scanf("%d", &no);

for (i = 1; i <= no; i++) {

init(i);

inp(i);

}

printf("\nThe routing tables of nodes after initialization is as follows");

for (i = 1; i <= no; i++)

display(i);

printf("\n\nComputing shortest paths...\n");

for (i = 1; i <= no; i++)

dv\_algo(i);

printf("\nThe routing tables of nodes after computation of shortest paths is as follows");

for (i = 1; i <= no; i++)

display(i);

printf("\n");

while (choice != 'n'){

printf("\nEnter the nodes between which shortest distance is to be found: ");

scanf("%d %d", &x, &y);

getchar();

printf("The length of the shortest path between nodes %d and %d is %d\n", x, y, router[x].a[y][2]);

printf("Continue? (y/n): ");

scanf("%c", &choice);

}

return 0;

}

## Output

NP-Lab/TW-3$ gcc dvalgo.c

NP-Lab/TW-3$ ./a.out

Enter the number of nodes: 5

Enter distance from node 1 to other nodes

Enter 999 if there is no direct route

Enter distance to node 2: 1

Enter distance to node 3: 999

Enter distance to node 4: 999

Enter distance to node 5: 999

Enter distance from node 2 to other nodes

Enter 999 if there is no direct route

Enter distance to node 1: 1

Enter distance to node 3: 3

Enter distance to node 4: 4

Enter distance to node 5: 5

Enter distance from node 3 to other nodes

Enter 999 if there is no direct route

Enter distance to node 1: 999

Enter distance to node 2: 2

Enter distance to node 4: 3

Enter distance to node 5: 999

Enter distance from node 4 to other nodes

Enter 999 if there is no direct route

Enter distance to node 1: 999

Enter distance to node 2: 4

Enter distance to node 3: 3

Enter distance to node 5: 999

Enter distance from node 5 to other nodes

Enter 999 if there is no direct route

Enter distance to node 1: 999

Enter distance to node 2: 5

Enter distance to node 3: 999

Enter distance to node 4: 999

The routing tables of nodes after initialization is as follows

The routing table for node 1 is as follows

1 0 1

2 1 2

3 no link no hop

4 no link no hop

5 no link no hop

The routing table for node 2 is as follows

1 1 1

2 0 2

3 3 3

4 4 4

5 5 5

The routing table for node 3 is as follows

1 no link no hop

2 2 2

3 0 3

4 3 4

5 no link no hop

The routing table for node 4 is as follows

1 no link no hop

2 4 2

3 3 3

4 0 4

5 no link no hop

The routing table for node 5 is as follows

1 no link no hop

2 5 2

3 no link no hop

4 no link no hop

5 0 5

Computing shortest paths...

The routing tables of nodes after computation of shortest paths is as follows

The routing table for node 1 is as follows

1 0 1

2 1 2

3 4 2

4 5 2

5 6 2

The routing table for node 2 is as follows

1 1 1

2 0 2

3 3 3

4 4 4

5 5 5

The routing table for node 3 is as follows

1 3 2

2 2 2

3 0 3

4 3 4

5 7 2

The routing table for node 4 is as follows

1 5 2

2 4 2

3 3 3

4 0 4

5 9 2

The routing table for node 5 is as follows

1 6 2

2 5 2

3 8 2

4 9 2

5 0 5

Enter the nodes between which shortest distance is to be found: 1 5

The length of the shortest path between nodes 1 and 5 is 6

Continue? (y/n): y

Enter the nodes between which shortest distance is to be found: 1 4

The length of the shortest path between nodes 1 and 4 is 5

Continue? (y/n): n