CN LAB 9

Pranay Kumar Karvi

Reg No – 23BDS1137

Aim – Implement Distance Vector Algorithm using C

Code –

Distance Vector Algorithm

```
#include <stdio.h>

#define INF 9999

#define MAX_ROUTERS 20

int cost[MAX_ROUTERS][MAX_ROUTERS];

int dist[MAX_ROUTERS][MAX_ROUTERS];

int next_hop[MAX_ROUTERS][MAX_ROUTERS];

void print_routing_table(int num_routers, int router_index) {

   printf("Routing Table for Router %d:\n", router_index);

   printf("Destination | Cost | Next Hop\n");

   for (int i = 0; i < num_routers; i++) {

      if (cost[router_index][i] != INF) {

         printf("%d          | %d   | %d\n", i, dist[router_index][i], next_hop[router_index][i]);

      } else {

         printf("%d          | INF  | -\n", i);
```

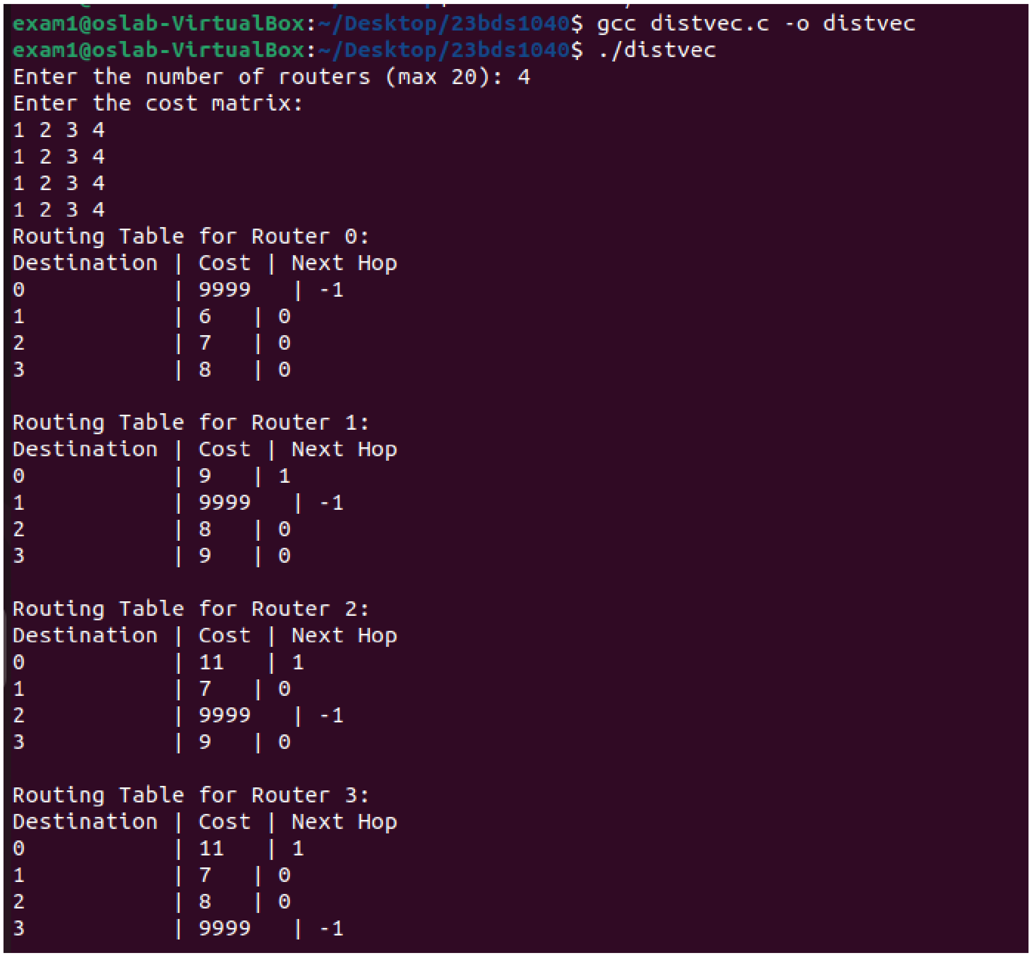
```
        }
    }
    printf("\n");
}

int update_routing_table(int num_routers) {
    int updated = 0;
    for (int i = 0; i < num_routers; i++) {
        for (int j = 0; j < num_routers; j++) {
            if (i != j) {
                int min_cost = INF;
                int next_hop_router = -1;

                for (int k = 0; k < num_routers; k++) {
                    if (cost[i][k] != INF && dist[k][j] != INF) {
                        int total_cost = cost[i][k] + dist[k][j];
                        if (total_cost < min_cost) {
                            min_cost = total_cost;
                            next_hop_router = k;
                        }
                    }
                }
                if (min_cost != dist[i][j]) {
                    dist[i][j] = min_cost;
                    next_hop[i][j] = next_hop_router;
                    updated = 1;
                }
            }
        }
    }
    return updated;
}
void link_failure(int router1, int router2) {
    cost[router1][router2] = INF;
    cost[router2][router1] = INF;
}
void change_link_cost(int router1, int router2, int new_cost) {
    cost[router1][router2] = new_cost;
    cost[router2][router1] = new_cost;
}
int main() {
    int num_routers;
    printf("Enter the number of routers (max 20): ");
    scanf("%d", &num_routers);

    if (num_routers > MAX_ROUTERS) {
        printf("Error: Maximum number of routers is 20.\n");
        return 1;
```

```
}

printf("Enter the cost matrix:\n");
for (int i = 0; i < num_routers; i++) {
    for (int j = 0; j < num_routers; j++) {
        scanf("%d", &cost[i][j]);
        if (cost[i][j] != INF && i != j) {
            dist[i][j] = cost[i][j];
            next_hop[i][j] = j;
        } else {
            dist[i][j] = INF;
            next_hop[i][j] = -1;
        }
    }
}

int converged = 0;
while (!converged) {
    converged = 1;
    for (int i = 0; i < num_routers; i++) {
        converged &= update_routing_table(num_routers);
    }
}
    for (int i = 0; i < num_routers; i++) {
```

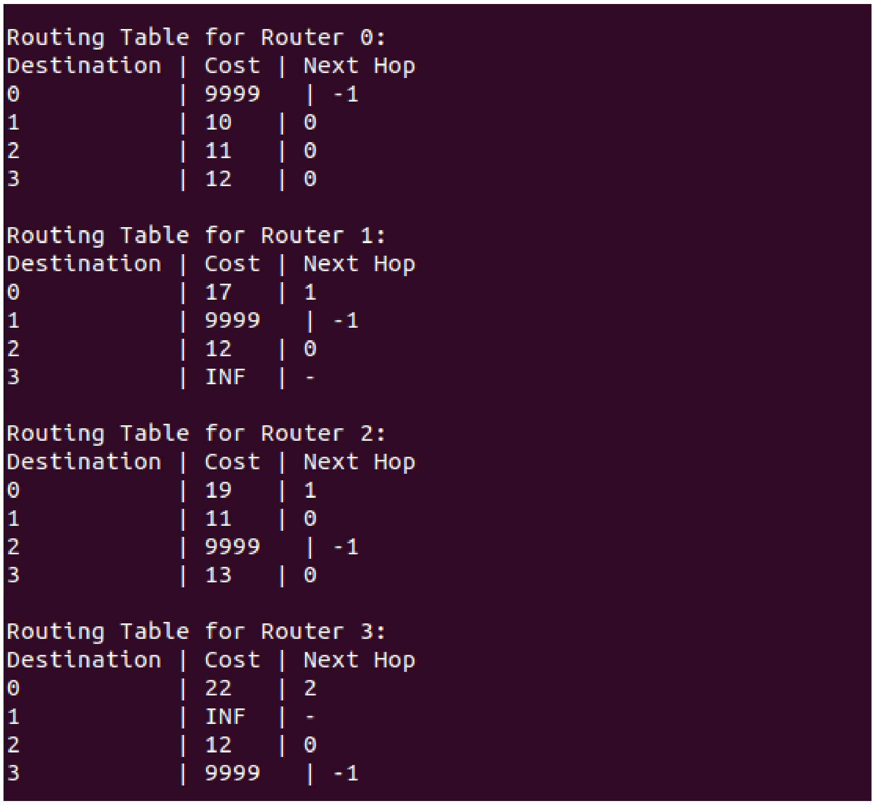
```
    print_routing_table(num_routers, i);
}
link_failure(1, 3);
converged = 0;
while (!converged) {
    converged = 1;
    for (int i = 0; i < num_routers; i++) {
        converged &= update_routing_table(num_routers);
    }
}
for (int i = 0; i < num_routers; i++) {
    print_routing_table(num_routers, i);
}

change_link_cost(2, 4, 1);
converged = 0;
while (!converged) {
    converged = 1;
    for (int i = 0; i < num_routers; i++) {
        converged &= update_routing_table(num_routers);
    }
}
for (int i = 0; i < num_routers; i++) {
    print_routing_table(num_routers, i);
```

```
  }

  return 0;

}
```

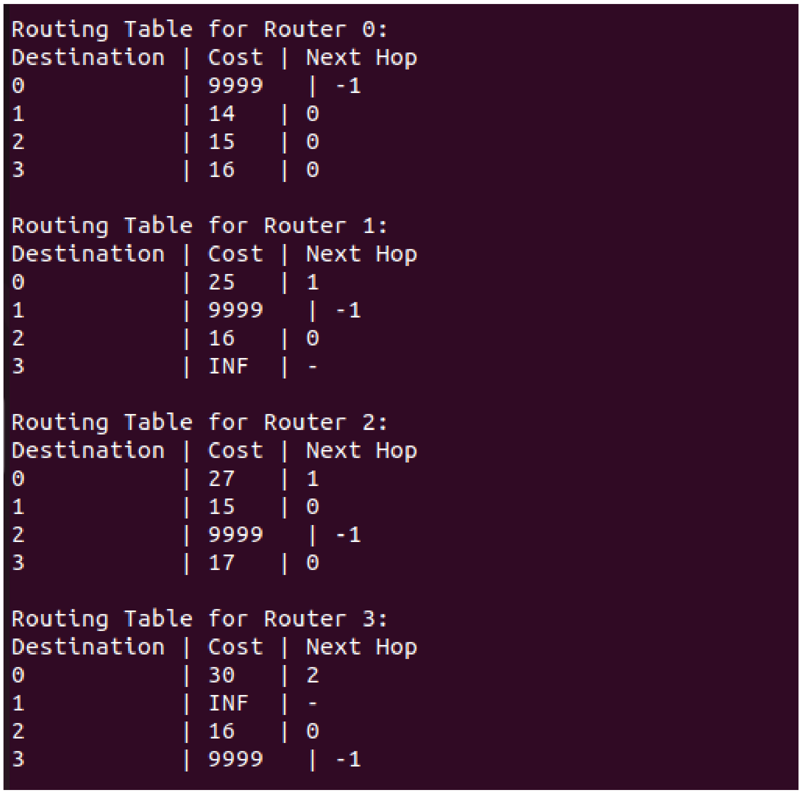
Output –



```
exam1@oslab-VirtualBox:~/Desktop/23bds1040$ gcc distvec.c -o distvec
exam1@oslab-VirtualBox:~/Desktop/23bds1040$ ./distvec
Enter the number of routers (max 20): 4
Enter the cost matrix:
1 2 3 4
1 2 3 4
1 2 3 4
1 2 3 4
Routing Table for Router 0:
Destination | Cost | Next Hop
0           | 9999   | -1
1           | 6   | 0
2           | 7   | 0
3           | 8   | 0

Routing Table for Router 1:
Destination | Cost | Next Hop
0           | 9   | 1
1           | 9999   | -1
2           | 8   | 0
3           | 9   | 0

Routing Table for Router 2:
Destination | Cost | Next Hop
0           | 11   | 1
1           | 7   | 0
2           | 9999   | -1
3           | 9   | 0

Routing Table for Router 3:
Destination | Cost | Next Hop
0           | 11   | 1
1           | 7   | 0
2           | 8   | 0
3           | 9999   | -1
```



```
Routing Table for Router 0:
Destination | Cost | Next Hop
0           | 9999   | -1
1           | 10   | 0
2           | 11   | 0
3           | 12   | 0

Routing Table for Router 1:
Destination | Cost | Next Hop
0           | 17   | 1
1           | 9999   | -1
2           | 12   | 0
3           | INF  | -

Routing Table for Router 2:
Destination | Cost | Next Hop
0           | 19   | 1
1           | 11   | 0
2           | 9999   | -1
3           | 13   | 0

Routing Table for Router 3:
Destination | Cost | Next Hop
0           | 22   | 2
1           | INF  | -
2           | 12   | 0
3           | 9999   | -1
```



```
Routing Table for Router 0:
Destination | Cost | Next Hop
0           | 9999   | -1
1           | 14   | 0
2           | 15   | 0
3           | 16   | 0

Routing Table for Router 1:
Destination | Cost | Next Hop
0           | 25   | 1
1           | 9999   | -1
2           | 16   | 0
3           | INF  | -

Routing Table for Router 2:
Destination | Cost | Next Hop
0           | 27   | 1
1           | 15   | 0
2           | 9999   | -1
3           | 17   | 0

Routing Table for Router 3:
Destination | Cost | Next Hop
0           | 30   | 2
1           | INF  | -
2           | 16   | 0
3           | 9999   | -1
```