

KARIKARAN.V

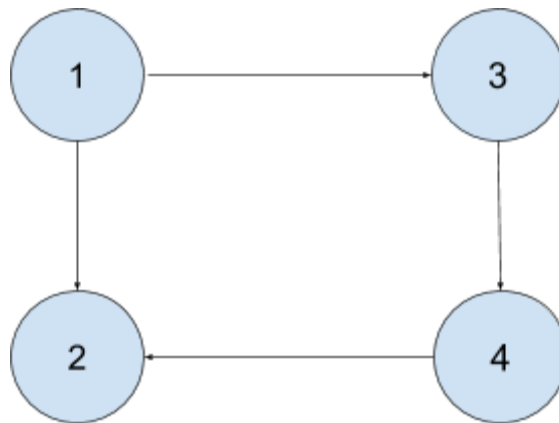
E/16/172

## CO322 – LAB 05

1. Find out what is the Transitive Closure of a graph.

Transitive Closure is the reachability matrix to reach from vertex  $u$  to vertex  $v$  of a graph. One graph is given, we have to find a vertex  $v$  which is reachable from another vertex  $u$ , for all pairs of vertices  $(u, v)$ . The final matrix is the Boolean type. When there is a value 1 for vertex  $u$  to vertex  $v$ , it means there is at least one path from  $u$  to  $v$ .

2. Manually compute the Transitive Closure for the following graph:



Transitive Closure matrix for the above graph:

1	1	1	1
0	1	0	0
0	1	1	1
0	1	0	1

3. Based on the Graph Traversal algorithm discussed in the class, write a C program to compute and print the Transitive Closure of a given graph. Use the following graph to test your program:

```

1 #include<stdio.h>
2
3 // define no of nodes in the graph
4 #define V 7
5
6 void print_mat(int reach[][V])
7 {
8     printf ("\nTransitive closure of the given graph\n\n");
9     for (int i = 0; i < V; i++)
10     {
11         for (int j = 0; j < V; j++)
12             printf ("%d ", reach[i][j]);
13         printf("\n");
14     }
15     printf("\n");
16 }
17
18 void transitiveClosure(int graph[][V])
19 {
20     int reach[V][V], i, j, k;
21
22     for (i = 0; i < V; i++)
23         for (j = 0; j < V; j++)
24             reach[i][j] = graph[i][j];
25
26     for (k = 0; k < V; k++) // checking the position of all to reach
27     {
28         for (i = 0; i < V; i++) // travel the rows
29         {
30             for (j = 0; j < V; j++) // travel the columns
31             {
32                 if(i==j){
33                     reach[i][j] = 1; // the vertex its self can reach condition
34                 }else{
35                     // checking the allredy is there any possible way or k th positional way to reach condition
36                     reach[i][j] = reach[i][j] || (reach[i][k] && reach[k][j]);
37                 }
38             }
39         }
40     }
41
42     print_mat(reach); // printing final reach array
43 }
44
45 int main()
46 {
47
48 }

```

```

harishbaana@harishbaana: ~/5th sem/co322 DataStr/Lab5/Lab05
harishbaana@harishbaana:~/5th sem/co322 DataStr/Lab5/Lab05$ gcc -o e16172_lab05 e16172_lab05.c
harishbaana@harishbaana:~/5th sem/co322 DataStr/Lab5/Lab05$ ./e16172_lab05

Transitive closure of the given graph

1 1 1 1 1 1 1
0 1 0 1 0 1
0 1 1 1 1 1
0 1 0 1 1 0 1
0 1 0 0 1 0 1
0 1 0 1 1 1 1
0 1 0 0 1 0 1

harishbaana@harishbaana:~/5th sem/co322 DataStr/Lab5/Lab05$

```

```

#include<stdio.h>

// define no of nodes in the graph
#define V 7

void print_mat(int reach[][V])
{
    printf ("\nTransitive closure of the given graph\n\n");
    for (int i = 0; i < V; i++)
    {
        for (int j = 0; j < V; j++)
            printf ("%d ", reach[i][j]);
        printf("\n");
    }
}

```



```
};  
  
    // Print the solution  
    transitiveClosure(graph);  
    return 0;  
}
```