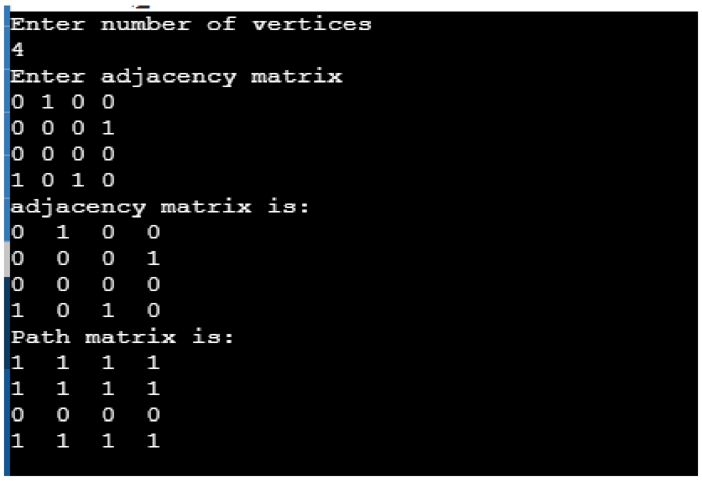
ADA-LAB TEST-2

Vakamalla keerthi priya
(1BM19CS176)

2. Implement Warshall’s algorithm using dynamic programming.

Modification:By using path matrix obtained detect the cycle in the graph.

```
#include<stdio.h>
void warshall(int a[10][10], int p[10][10],int n)
{

    int i,j,k;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            p[i][j]=a[i][j];
        }
    }
    for(k=0;k<n;k++)
    {
        for(i=0;i<n;i++)
        {
            for(j=0;j<n;j++)
            {
                if(p[i][j]!=1 && p[i][k]==1 && p[k][j]==1)
                p[i][j]=1;
            }
        }
    }
}

int main()
{
    int a[10][10],p[10][10],n,i,j;
    printf("Enter number of vertices\n");
    scanf("%d",&n);
    printf("Enter adjacency matrix\n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        scanf("%d",&a[i][j]);
    }
    printf("adjacency matrix is: \n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        printf("%d  ",a[i][j]);
        printf("\n");
    }
    warshall(a,p,n);
    printf("Path matrix is: \n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        printf("%d  ",p[i][j]);
        printf("\n");
    }
    return 0;
}
```



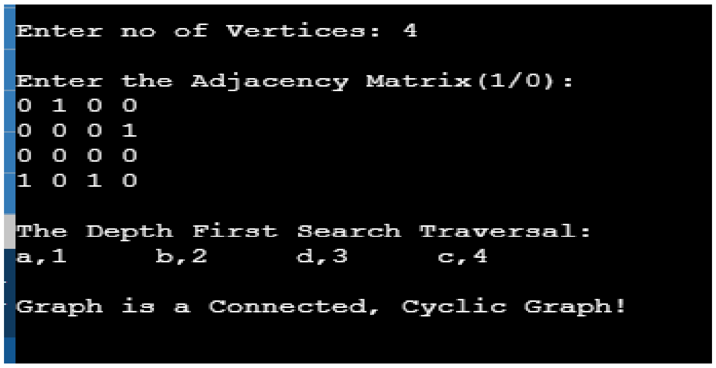
```
Enter number of vertices
4
Enter adjacency matrix
0 1 0 0
0 0 0 1
0 0 0 0
1 0 1 0
adjacency matrix is:
0  1  0  0
0  0  0  1
0  0  0  0
1  0  1  0
Path matrix is:
1  1  1  1
1  1  1  1
0  0  0  0
1  1  1  1
```

MODIFIED-

```
#include<stdio.h>
#include<stdlib.h>

int A[20][20],visited[20],count=0,n;
int seq[20],connected=1,acyclic=1;

void DFS();

void DFSearch(int cur);
```

```
int main()
  {
   int i,j;

   printf("\nEnter no of Vertices: ");
   scanf("%d",&n);

   printf("\nEnter the Adjacency Matrix(1/0):\n");
   for(i=1;i<=n;i++)
          for(j=1;j<=n;j++)
      scanf("%d",&A[i][j]);

   printf("\nThe Depth First Search Traversal:\n");

   DFS();

   for(i=1;i<=n;i++)
      printf("%c,%d\t",'a'+seq[i]-1,i);

   if(connected && acyclic)    printf("\n\nIt is a Connected, Acyclic Graph!");
   if(!connected && acyclic)   printf("\n\nIt is a Not-Connected, Acyclic Graph!");
   if(connected && !acyclic)   printf("\n\nGraph is a Connected, Cyclic Graph!");
   if(!connected && !acyclic)  printf("\n\nIt is a Not-Connected, Cyclic Graph!");

   printf("\n\n");
   return 0;
  }

void DFS()
   {
   int i;
   for(i=1;i<=n;i++)
      if(!visited[i])
        {
      if(i>1) connected=0;
      DFSearch(i);
           }
   }

void DFSearch(int cur)
   {
   int i,j;
   visited[cur]=++count;
```

```
        seq[count]=cur;
        for(i=1;i<count-1;i++)
                if(A[cur][seq[i]])
                   acyclic=0;

    for(i=1;i<=n;i++)
        if(A[cur][i] && !visited[i])
           DFSearch(i);
}
```



```
Enter no of Vertices: 4

Enter the Adjacency Matrix(1/0):
0 1 0 0
0 0 0 1
0 0 0 0
1 0 1 0

The Depth First Search Traversal:
a,1      b,2      d,3      c,4

Graph is a Connected, Cyclic Graph!
```