```
******SHARED MEMORY: WRITESHM******

Key of the shared memory is: 98304
Enter data to wrtie to shared memory: Wussup World?

The data written is : Wussup World?

s4csbs@rajagiri-OptiPlex-5090:~/Desktop/S4 OS/Expt9 -
 - Shared Memory/"shared

******SHARED MEMORY: SHARED DATA******
Key of shared memory is: 98304
Data from shared memory: Wussup World?
```

```
******MESSAGE QUEUS: WRITING DATA******
Write message to send: Goodbye World
The data is : Goodbye World
s4csbs@rajagiri-OptiPlex-5090:~/Desktop/S4 OS/
/Expt8 - Message Queues/"readdata

******MESSAGE QUEUS: READING DATA DATA******
Data received is Goodbye World
```

```
******PIPES******

Parent pass value to child
Enter data: Hello World

Child received data
Buffer: Hello World
```

```
Enter no of vertices: 6
Enter adjacency matrix:
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 1 0 0
0 1 0 0 0 0
1 1 0 0 0 0
1 0 1 0 0 0
printing adjacency matrix:
0       0       0       0       0       0
0       0       0       0       0       0
Enter adjacency matrix:
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 1 0 0
0 1 0 0 0 0
1 1 0 0 0 0
1 0 1 0 0 0
printing adjacency matrix:
0       0       0       0       0       0
0       0       0       0       0       0
0       0       0       1       0       0
0       1       0       0       0       0
1       1       0       0       0       0
1       0       1       0       0       0

which node to start from?: 5

Result:
4       5       2       3       1       0
```

```c
#include<stdio.h>
#include<stdlib.h>
void main()
{
    int v,i,j,k=0,top=-1;
    printf("Enter no of vertices: ");
    scanf("%d",&v );
    int adj[v][v];
    printf("Enter adjacency matrix:\n");
    for(int i=0;i<v;i++)
        for(int j=0;j<v;j++)
            scanf("%d",&adj[i][j]);
    printf("printing adjacency matrix: \n");
    for(i=0;i<v;i++)
    {
        for(j=0;j<v;j++)
        {
            printf("%d\t",adj[i][j]);
        }
        printf("\n");
    }
    int visited[v];
    int stack[v];
    for( i=0;i<v;i++)
        visited[i]=0;
    void push(int x)
    {
        top++;
        stack[top]=x;
    }
    void pop()
    {
        printf("%d\t",stack[top]);
        top--;

    }
    void topsort(int n)
    {
            for(int j=0;j<v;j++)
                if(adj[n][j]!=0 && visited[j]==0)
```

```
                    topsort(j);

            push(n);
            visited[n]=1;
    }
    int z;
    printf("\nwhich node to start from?: ");
    scanf("%d",&z);
    topsort(z);
    printf("\nResult: \n");
    z=0;
    for(i=0;i<v;i++)
    {   if(visited[i]==0)
        {   printf("%d\t", i);
            z++;
        }
    }
    for(i=0;i<v-z;i++)
    {
        pop();
    }
    printf("\n");
}
```