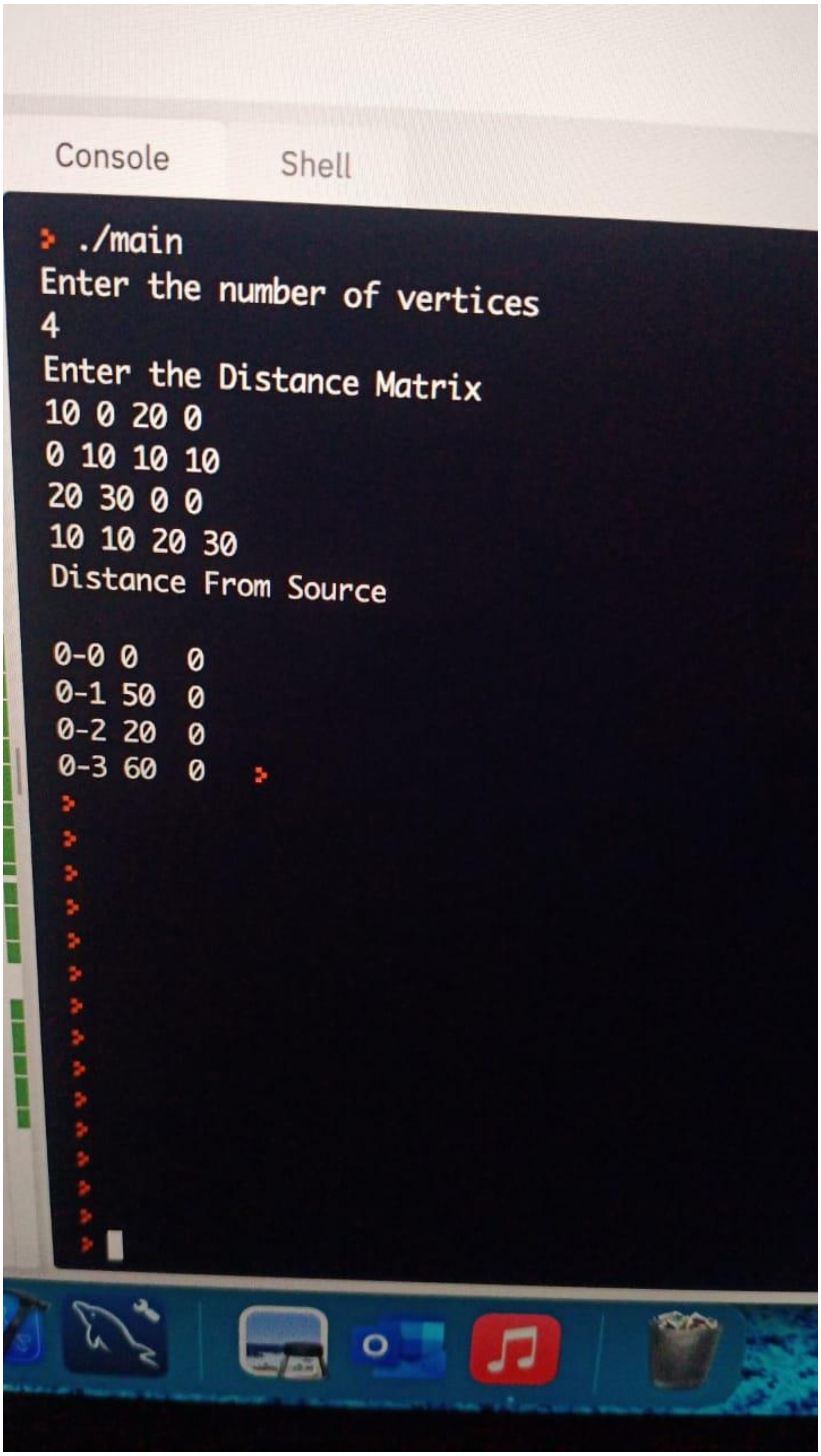
ADA LABTEST-2

NAME-ASHUTOSH UPADHYAY
USN-1BM19CS027
SEC-4 ‘A’ CSE

Q: **From a given vertex in a weighted connected graph, find shortest paths to other vertices usingDijkstra’s algorithm.**

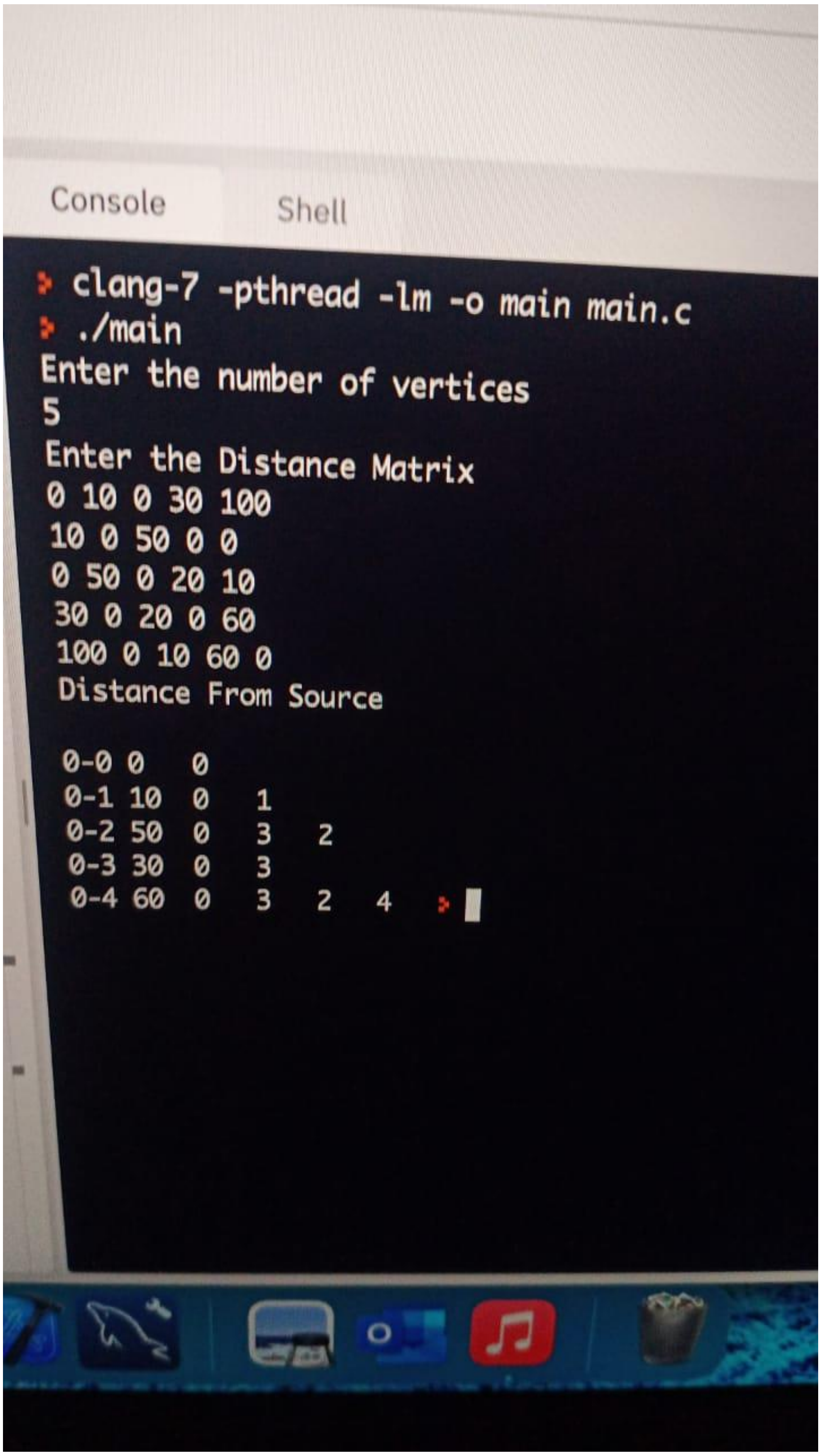
```
#include<stdio.h>

#include<limits.h>

#include<stdbool.h>

int V;

int minKey(int key[],bool mstset[])

{

    int min,minIndex;

    min=INT_MAX;

    for(int i=0;i<V;i++)

    {

        if(mstset[i]==false && key[i]<min)

        {

            min=key[i];

            minIndex=i;

        }

    }

    return minIndex;

}


void printmst(int key[],int parent[])

{

    int sum=0;

    printf("Distance From Source\n");
    for(int i=0;i<V;i++)

    {

        printf("\n%d-%d\t%d\t%d\t",0,i,key[i],0);


    }

}

void primst(int graph[V][V])

{

    int parent[V];

    int key[V];

    bool mstset[V];

    int sum=0;

    for(int i=0;i<V;i++)

    {

        parent[i]=0;

        key[i]=INT_MAX;

        mstset[i]=false;

    }

    key[0]=0;

    parent[0]=-1;

    for(int count=0;count<V-1;count++)

    {
        int u=minKey(key,mstset);

        mstset[u]=true;

        for(int v=0;v<V;v++)

        {

            if(graph[u][v] && mstset[v]==false && key[u]!=INT_MAX && key[u]+graph[u][v]<key[v])

            {

                parent[v]=u;

                key[v]=key[u] + graph[u][v];

            }

        }

    }

    printmst(key,parent);

}

int main()

{

    printf("Enter the number of vertices\n");

    scanf("%d",&V);

    int graph[V][V];

    printf("Enter the Distance Matrix\n");

    for(int i=0;i<V;i++)

    {

        for(int j=0;j<V;j++)
        {
            scanf("%d",&graph[i][j]);
        }
    }
    primst(graph);
}
```



```
> ./main
Enter the number of vertices
4
Enter the Distance Matrix
10 0 20 0
0 10 10 10
20 30 0 0
10 10 20 30
Distance From Source

0-0 0   0
0-1 50  0
0-2 20  0
0-3 60  0    >
>
>
>
>
>
>
>
>
>
>
>
>
>
>
```

MODIFICATION: Print number of nodes along the shortest paths.

```
#include<stdio.h>
#include<limits.h>
#include<stdbool.h>
int V;
int minKey(int key[],bool mstset[])
{
    int min,minIndex;
    min=INT_MAX;
    for(int i=0;i<V;i++)
    {
        if(mstset[i]==false && key[i]<min)
        {
            min=key[i];
            minIndex=i;
        }
    }
    return minIndex;
}
void printpath(int parent[],int j)
{
    if(parent[j]==-1)
    return;

    printpath(parent,parent[j]);
    printf("%d\t",j);
}
void printmst(int key[],int parent[])
{
    int sum=0;
    printf("Distance From Source\n");
    for(int i=0;i<V;i++)
    {
        printf("\n%d-%d\t%d\t%d\t",0,i,key[i],0);
        printpath(parent,i);

    }
}
void primst(int graph[V][V])
{
    int parent[V];
    int key[V];
    bool mstset[V];
    int sum=0;
    for(int i=0;i<V;i++)
    {
        parent[i]=0;
        key[i]=INT_MAX;
        mstset[i]=false;
    }
    key[0]=0;
    parent[0]=-1;
    for(int count=0;count<V-1;count++)
    {
        int u=minKey(key,mstset);
        mstset[u]=true;
        for(int v=0;v<V;v++)
        {
            if(graph[u][v] && mstset[v]==false && key[u]!=INT_MAX && key[u]+graph[u][v]<key[v])
            {
                parent[v]=u;
                key[v]=key[u] + graph[u][v];
            }
        }
    }
    printmst(key,parent);
}
int main()
{
    printf("Enter the number of vertices\n");
    scanf("%d",&V);
    int graph[V][V];
    printf("Enter the Distance Matrix\n");
    for(int i=0;i<V;i++)
    {
        for(int j=0;j<V;j++)
        {
            scanf("%d",&graph[i][j]);
        }
    }
    primst(graph);
}
```



```
> clang-7 -pthread -lm -o main main.c
> ./main
Enter the number of vertices
5
Enter the Distance Matrix
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0
Distance From Source

 0-0 0   0
 0-1 10  0   1
 0-2 50  0   3   2
 0-3 30  0   3
 0-4 60  0   3   2   4   >
```