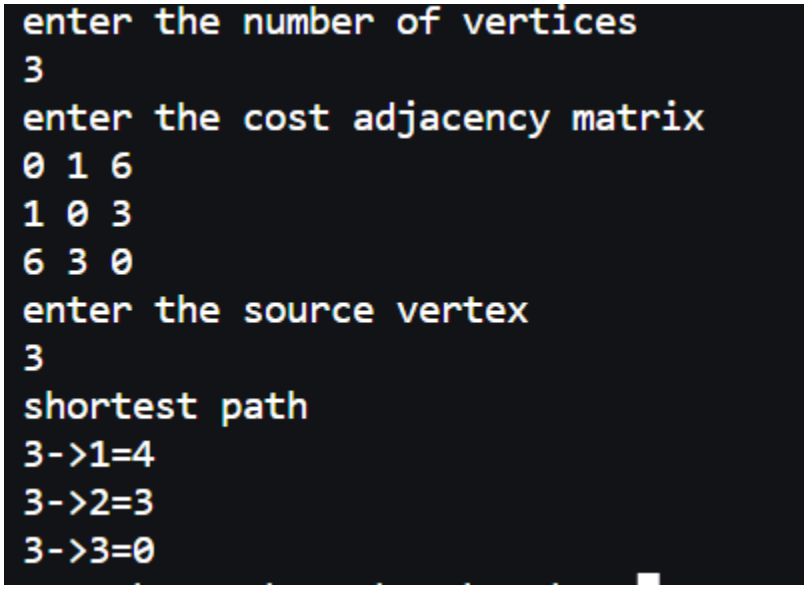
DJIKSTRAS ALGORITHM

```
#include <stdio.h>

void dijkstra(int n, int cost[10][10], int src)
{
    int i, j, u, dis[10], vis[10], min;
    for (i = 1; i <= n; i++)
    {
        dis[i] = cost[src][i];
        vis[i] = 0;
    }
    vis[src] = 1;
    for (i = 1; i <= n; i++)
    {
        min = 999;
        for (j = 1; j <= n; j++)
        {
            if (vis[j] == 0 && dis[j] < min)
            {
                min = dis[j];
                u = j;
            }
        }
        vis[u] = 1;
        for (j = 1; j <= n; j++)
        {
            if (vis[j] == 0 && dis[u] + cost[u][j] < dis[j])
            {
                dis[j] = dis[u] + cost[u][j];
            }
        }
    }

    printf("shortest path\n");
    for (i = 1; i <= n; i++)
        printf("%d->%d=%d\n", src, i, dis[i]);
```

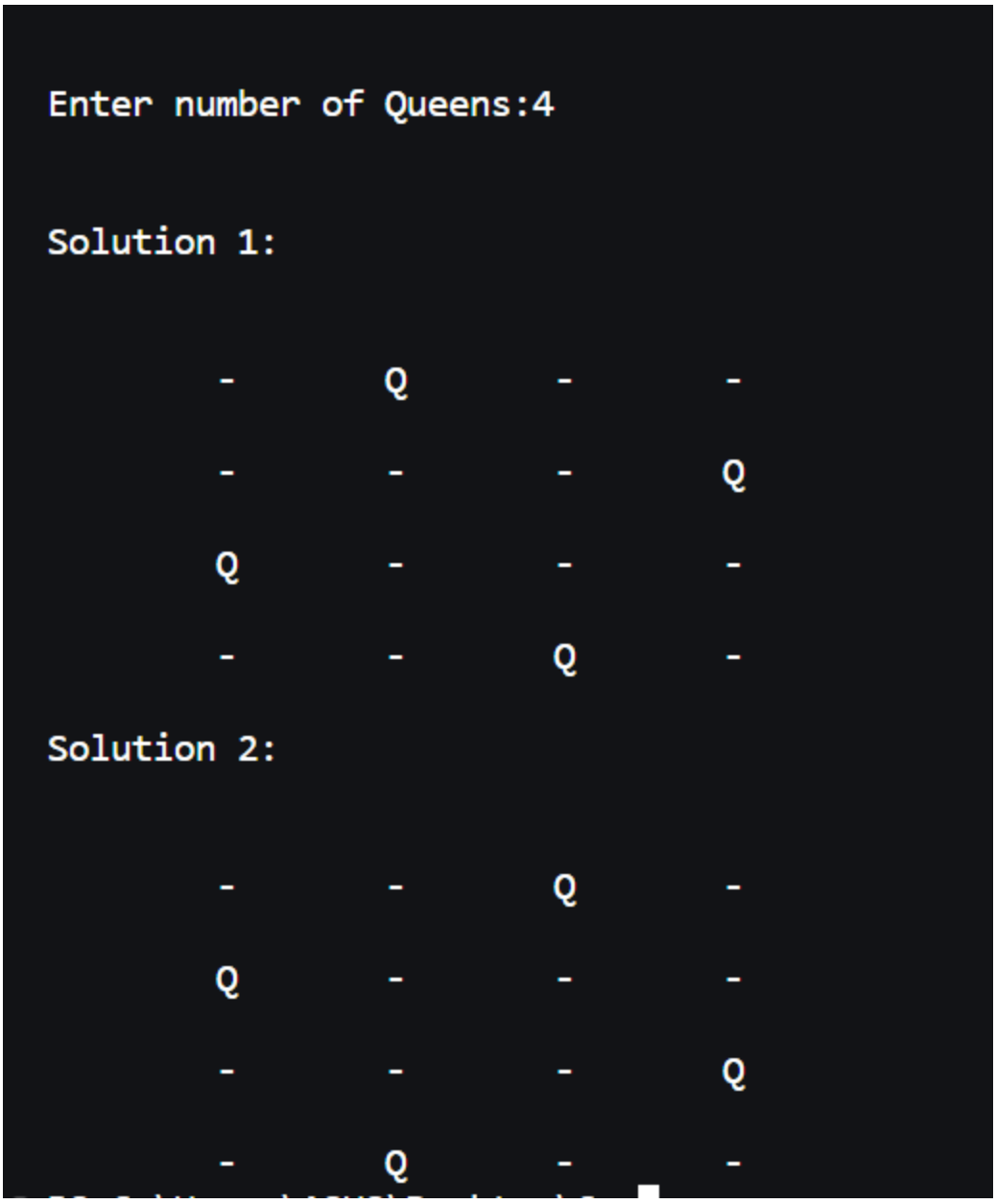
```
}

void main()
{
    int src, j, cost[10][10], n, i;

    printf("enter the number of vertices\n");
    scanf("%d", &n);
    printf("enter the cost adjacency matrix\n");
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
            scanf("%d", &cost[i][j]);
    printf("enter the source vertex\n");
    scanf("%d", &src);
    dijkstra(n, cost, src);
}
```



N-QUEEN PROBLEM

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

int board[20], count;

int main()
{
   int n, i, j;
   void queen(int row, int n);
   printf("\n\nEnter number of Queens:");
   scanf("%d", &n);
   queen(1, n);
   return 0;
}

void print(int n)
{
   int i, j;
   printf("\n\nSolution %d:\n", ++count);
   for (i = 1; i <= n; ++i)
   {
      printf("\n\n");
      for (j = 1; j <= n; ++j)
      {
         if (board[i] == j)
            printf("\tQ");
         else
            printf("\t-");
      }
   }
}

int place(int row, int column)
{
   int i;
```

```
    for (i = 1; i <= row - 1; ++i)
    {
        if (board[i] == column)
            return 0;
        else if (abs(board[i] - column) == abs(i - row))
            return 0;
    }
    return 1;
}

void queen(int row, int n)
{
    int column;
    for (column = 1; column <= n; ++column)
    {
        if (place(row, column))
        {
            board[row] = column;
            if (row == n)
                print(n);
            else
                queen(row + 1, n);
        }
    }
}
```


Enter number of Queens:4
Solution 1:
- Q - -
- - - Q
Q - - -
- - Q -
Solution 2:
- - Q -
Q - - -
- - - Q
- Q - -