```
#include<stdio.h>
#define MAX 50
#define N 5
#define M 1000

typedef int itemtype;
typedef struct {
    itemtype item[MAX];
    int qfront;
    int qrear;
    int qcount;
} Queue;

void inisialisasi_queue(Queue *q)
{
    q->qfront = q->qrear = q->qcount = 0;
}

int QPenuh(Queue *q)
{
    return(q->qcount==MAX);
}

int QKosong(Queue *q)
{
    return(q->qcount==0);
}

void Enqueue (Queue *q, itemtype x)
{
    if(QPenuh(q))
        printf("Queue penuh\n");
    else{
        q->item[q->qrear]=x;
        q->qrear=(q->qrear+1)%MAX;
        q->qcount++;
    }
}

itemtype Dequeue (Queue *q)
{
    itemtype temp;
    if(QKosong(q)){
        printf("Queue kosong\n");
        temp=-1;
    }
    else{
        temp=q->item[q->qfront];
        q->qfront=(q->qfront+1)%MAX;
        q->qcount--;
    }
    return temp;
}

void inisialisasi_djikstra(int Q[], int R[], int asal)
{
    int i;
    for(i=0;i<N;i++){
        if(i==asal-1)
            Q[i]=0;
        else
            Q[i]=M;
        R[i]=0;
    }
}

int ada_di_queue(Queue *q, int node)
{
    int i=q->qfront, ada=0;
    while(i!=q->qrear && !ada){
        if(q->item[i]==node)
            ada=1;
        else
            i=(i+1)%MAX;
    }
    return ada;
}

void cetak(int A[], char *judul)
{
    int i;
    printf("%s: ", judul);
    for(i=0;i<N;i++){
        if(A[i]<M)
            printf("%d ", A[i]);
        else
            printf("M ");
    }
    printf("\n");
}

main()
{
    int input[N][N]={M,1,3,M,M,
                     M,M,1,M,5,
                     3,M,M,2,M,
                     M,M,M,M,1,
                     M,M,M,M,M};
    int Q[N], R[N], asal, tujuan, CN, i;
    Queue qu;
    printf("Masukkan titik asal: ");scanf("%d",&asal);
    printf("Masukkan titik tujuan: ");scanf("%d",&tujuan);
    inisialisasi_djikstra(Q,R,asal);
    inisialisasi_queue(&qu);
    Enqueue(&qu,asal);
    while(!QKosong(&qu)){
        CN=Dequeue(&qu)-1;
        for(i=0;i<N;i++)
            if(input[CN][i]!=M)
                if(Q[CN]+input[CN][i]<Q[i]){
                    Q[i]=Q[CN]+input[CN][i];
                    R[i]=CN+1;
                    if(i!=(asal-1) && i!=(tujuan-1) && !ada_di_queue(&qu,i+1))
                        Enqueue(&qu,i+1);
                }
    }
    cetak(Q,"Beban");
    cetak(R,"Rute");
}
```