```cpp
#include"MGraph.h"

#define pathMatrix bool
#define ShortPathTable int

bool finl[MAX_VERTEX_NUM];
ShortPathTable D[MAX_VERTEX_NUM];
pathMatrix p[MAX_VERTEX_NUM][MAX_VERTEX_NUM];

//用DijStra算法求又向网G的v0顶点到其余顶点v的最短路径p[v];及其带权长度D[v]
void ShortestPath_DIJ(MGraph G, int v0){
    //p[v][w]==true 则w是从v0到v当前求得最短路径上的顶点
    //finl[v]==true 当且仅当v属于s集即已经求得v0到v的最短路径

    //初始化
    for(int v=0; v<G.vexnum; v++){
        finl[v] = false;
        D[v] = G.arcs[v0][v].adj;
        for(int w=0; w<G.vexnum; ++w)    //设置空路径
            p[v][w] = false;
        if(D[v]<INFINITY){
            p[v][v0] = true;
            p[v][v] = true;
        }
    }//for

    //初始化，v0顶点属于s集
    D[v0] = 0;  finl[v0] = true;
    int min;
    int v = v0; //用v标识当前离v0最近的顶点
    //开始主循环，每次求得v0到某个v顶点的最短路径，并将其添加到s集
    for(int i=1; i<G.vexnum; i++){  //其余G.vexnum-1个顶点
        min = INFINITY;             //当前所知距离v0的最短路径
        for(int w=0; w<G.vexnum; ++w)
            if(!finl[w])            //w在V-S集中
                if(D[w]<min){       //w离v0顶点更近
                    v=w; min=D[w];}
        finl[v] = true;             //离v0最近的顶点v添加到S
        for(int w=0; w<G.vexnum; w++)//更新当前最短路径及距离
            if(!finl[w] && (min+G.arcs[v][w].adj < D[w]) ){ //修改D[w]和p[w], w属于V-S
                D[w] = min + G.arcs[v][w].adj;
                //p[w] = p[v] + p[w]
                for(int j=0; j<G.vexnum; j++)
                    p[w][j] = p[v][j];
                p[w][w] = true;
            }//if
    }//for
}//ShortestPath_DIJ

void demo_DIJ(MGraph G){
    VertexType v;
    cout << "请输入要查询的顶点:\t";
    cin >> v;
    cout << endl;
    ShortestPath_DIJ(G,LocateVex(G,v));
    cout << "从 " << v << " 出发到达图中个顶点的路径是:" << endl;
    for(int i=0; i<G.vexnum; i++){
        for(int j=0; j<G.vexnum; j++)
            cout << p[i][j] << "  ";
        cout << "\t\t" << D[i]<< endl;
    }
}

int main(){
    MGraph G;
    CreateGraph(G);

    PrintAdjMatrix(G);

    demo_DIJ(G);

    return 0;
}
```