```
///////////////////////
//2005级信息安全2班  张文
//    200532530040
///////////////////////
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

#define MAXV 10//最大顶点数
#define WEIGHT 1//边的权重
#define SP 0//起始的结点(i)的编号
#define EP 1//结束的节点(j)的编号
///////////////////////////////////////邻接表
typedef struct ANode
{
        int adjvex;
        struct ANode *nextarc;
}ArcNode;

typedef struct VNode_type
{
        int node;
        ArcNode *firstarc;
}VNode;

typedef struct VNode_type  AdjList[MAXV];

typedef struct
{
        AdjList adjlist;
        int n,e;
}ALGraph;

/////////////////////////BFS所需的队列等数据结构
typedef struct
{
        int PreNode;
        int CurrentNode;
        int cost;
}QNode_type;

typedef struct
{
        QNode_type NodeInfo;
        int visited;
}VisitedStatus;

/////////////////////////////////////////函数申明
void initGraph(ALGraph *);
void ShortestPath(ALGraph *,QNode_type *,int);
void PrintPath(QNode_type *,int,int,int);

/////////////////////////////////////////主程序
int main()
{
        ALGraph MyGraph;
        QNode_type MyAns[MAXV];
        initGraph(&MyGraph);
        ShortestPath(&MyGraph,MyAns,SP);
        PrintPath(MyAns,SP,EP,MyGraph.n);
        return 0;
}

void initGraph(ALGraph *ThisGraph)//建立邻接表
{
        int Node[2];
        int i,j;
```

```c
	ArcNode *ThisNode,*PreNode;

	printf("输入节点总数：");
	scanf("%d",&(ThisGraph->n));

	for(i=0;i<MAXV;i++)
	{
		ThisGraph->adjlist[i].firstarc=NULL;
		ThisGraph->adjlist[i].node=i;
	}

	i=0;
	while(1)
	{
		printf("输入每条边的两个端点，用空格间隔（输入两个－1结束输入）：");
		scanf("%d%d",&Node[0],&Node[1]);
		if(Node[0]==-1 && Node[1]==-1)break;

		for(j=0;j<2;j++)
		{
			PreNode=ThisGraph->adjlist[Node[j]].firstarc;
			ThisNode=(ArcNode *)malloc(sizeof(ArcNode));
			ThisNode->adjvex=Node[(j+1)%2];
			ThisNode->nextarc=NULL;

			if(PreNode==NULL)ThisGraph->adjlist[Node[j]].firstarc=ThisNode;
			else
			{
				while(PreNode->nextarc!=NULL)PreNode=PreNode->nextarc;
				PreNode->nextarc=ThisNode;
			}
		}

		i++;
	}
	ThisGraph->e=i;
}

void ShortestPath(ALGraph *ThisGraph,QNode_type *Head,int StartPoint)//利用BFS思想求最短路
{
	QNode_type Queue[MAXV];
	VisitedStatus Status[MAXV];
	ArcNode *ThisArc;

	int open,closed,i;

	memset(Queue,0,sizeof(Queue));
	memset(Status,0,sizeof(Status));

	open=0;
	closed=-1;

	Queue[0].PreNode=-1;
	Queue[0].CurrentNode=0;
	Queue[0].cost=0;

	memcpy(&(Status[0].NodeInfo),&Queue[0],sizeof(int)*3);
	Status[0].visited=1;

	while(open>=closed)
	{
		closed++;
		ThisArc=ThisGraph->adjlist[Queue[closed].CurrentNode].firstarc;
		while(ThisArc!=NULL)
		{
			if(!Status[ThisArc->adjvex].visited)
			{
```

```
                Status[ThisArc->adjvex].visited=1;
                open++;
                Queue[open].CurrentNode=ThisArc->adjvex;
                Queue[open].PreNode=Queue[closed].CurrentNode;
                Queue[open].cost=Queue[closed].cost+WEIGHT;
                memcpy(&(Status[ThisArc->adjvex].NodeInfo),&Queue[open],sizeof(int)*3);
            }
            ThisArc=ThisArc->nextarc;
        }
    }

    for(i=0;i<ThisGraph->n;i++)memcpy(&(Head[i]),&(Status[i].NodeInfo),sizeof(int)*3);

}

void PrintPath(QNode_type *ThisNode,int StartPoint,int EndPoint,int MapScale)//输出结果
{
    int CurrentPoint;
    int path[MAXV];
    int i,j;

    printf("从结点%d到节点%d的最短路径长度为 %d\n\n",SP,EP,ThisNode[EndPoint].cost);

    for(CurrentPoint=0;CurrentPoint<MapScale;CurrentPoint++)
        if(CurrentPoint!=SP)
        {
            printf("从结点%d到节点%d的最短路径长度为%d\n",SP,CurrentPoint,ThisNode[CurrentPoint].cost);
            i=-1;j=CurrentPoint;
            while(j!=-1)
            {
                path[++i]=j;
                j=ThisNode[j].PreNode;
            }
            for(;i>0;i--)printf("%d->",path[i]);
            printf("%d\n\n",path[i]);
        }
}
```