1. **试设计一个算法：将一个无向图的邻接矩阵转换为邻接表。**

解答：先设置一个空的邻接表，然后在邻接矩阵上查找值不为零的元素，找到后在邻接表的对应单链表中插入相应的边表结点。

void MatToList(AdjMatrix &A, AdjList &B)

{

B.vexnum = A.vexnum;

B.edgenum = A.edgenum;

for (i = 0; i < A.vexnum; i++)

{

B.vexs[i].vex = A.vexs[i];

B.vexs[i].firstedge = NULL;

}

for ( i = 0; i < A.vexnum;i++)//双循环找邻接矩阵所有边

{

for (j = 0; j < A.vexnum; j++)

{

if (A.edges[i][j] != 0){

p->adjvex = j;

p->nextedge = B.vexs[i].firstedge;//头插法

B.vexs[i].firstedge = p;

}

}

}

}

1. **试设计一个算法：基于深度优先遍历算法，判断以邻接表存储的有向图中是否存在由顶点vi到vj的路径（i≠j）。**

解答：

bool DFSPath(ALGraph G, int vi, int vj)

{//如果vi到vj有路径返回1，否则返回0;G为有向图的邻接表

for (i = 0; i < G.vexnum; ++i ) visited[i]=0;

visited[vi]=1;

for(p=G.vexs[vi].firstedge;p;p=p->nextedge)

{

j=p->adjvex;

if(!visited[j]) {

visited[j]=1;

if(j==vj) return 1; //找到路径

else return(DFSPath(G,j,vj)) ;

}

}

return 0;

}

**3．试设计一个算法：判断无向图G是否连通；如果连通，则返回1；否则返回0。**

解答：

bool connect(MGraph G)

{

int i,flag=1;

for (i=0;i<G.vexnum;i++)

visited[i]=0;

DFS(G,0);

for (i=0;i<G.vexnum;i++)

if (visited[i]==0)

flag=0;

break;

return flag;

}

void DFS(MGraph G,int v)

{//从定点v出发深度优先遍历以邻接表存储的图G

visited[v]=1;

for(p=G.vexs[v].firstedge;p;p=p->nextedge){

j=p->adjvex;

if(!visited[j]) DFS(G,j);

}

}