第七章作业答案

7.27

```
bool visited[MAX]={false};
bool FindKPath(ALGraph G, int i, int j, int k){
    for(int v=0;v<G.verxnum;v++)
        visited[v]=FALSE;
    return DFS(G,i,j,k)
}

bool DFS(ALGraph G,int i,int j,int k){
    visited[i]=TRUE;
    if(i==j&&k==0)
        return TRUE;
    else if(k==0)
        return FALSE;
    for(p=G.vertices[i].firstarc;p!=NULL;p=p->nextarc){
        if(!visited[p->adjvex]){
            if(DFS(G,p->adjvex,j,k-1))
                return TRUE;
        }
    }
    visited[i]=FALSE;
    return FALSE;
}
```

7.28

```
bool visited[MAX]={false};
char Path[MAX_VERTEX_NUM];
void DFS(ALGraph G,char u,char v,int n){
    visited[u]=TRUE;
    if(u==v){
        for(int i=0;i<n;i++){
            printf("%c",Path[i]);
        }
        printf("%c\n",v);
        visited[v]=FALSE;
        return;
    }
    else{
        for(p=G.vertices[u].firstarc;p!=NULL;p=p->nextarc){
            if(!visited[p->adjvex]){
                Path[n]=u;
                DFS(G,p->adjvex,v,n+1);
                visited[p->adjvex]=FALSE;
            }
        }
    }
}

void FindPath(ALGraph G,char u,char v){
    for(int i=0;i<verxnum;i++)
        visited[i]=FALSE;
    DFS(G,u,v,0);
}
```

7.29

```
bool visited[MAX]={false};
int ans=0;
void DFS(MGraph G,int i,int j,int k){
    visited[i]=TRUE;
    if(i==j&&k==0){
        ans++;
        visited[j]=FALSE;
        return;
    }
    else if(k==0){
        visited[i]=FALSE;
        return;
    }
    else{
        for(int m=1;m<=G.verxnum;m++){
            if(G.arcs[i-1][m-1]==1){
                DFS(G,m,j,k-1);
            }
        }
        visited[i]=FALSE;
        return;
    }
}


int PathNum(MGraph G,int i,int j,int k){
    for(int m=0;m<G.verxnum;m++)
        visited[m]=FALSE;
    DFS(G,i,j,k);
    return n;
}



//这题本质上就是一个拓扑排序
//所以直接按照拓扑排序照抄即可
Status problem_7_34(ALGraph G, int Topo[])
{
    SqStack S;
    ArcNode *p;
    int i, k, count, indegree[MAX_VERTEX_NUM + 1];
    FindInDegree(G, indegree);          //对各顶点求入度
    InitStack_Sq(&S);                   //初始化栈
    for (i = 1; i <= G.vexnum; i++)         //建立入度为0的顶点栈
        if (!indegree[i])
            Push_Sq(&S, i);             //入度为0者进栈
    count = 0;
    while (!StackEmpty_Sq(S))            //出栈输出部分
    {
        Pop_Sq(&S,& i);
        count++;
        Topo[count] = i;                //对节点进行编号
        for (p = G.vertices[i].firstarc; p; p = p->nextarc)
        {
            k = p->adjvex;
            if (!(--indegree[k]))
                Push_Sq(&S, k);
        }
    }
    if (count<G.vexnum)
        return ERROR;
    else
        return OK;
}




//7.37题本质就是7.36题 所以先给出7.36的答案
//因为7.37要存储路径节点，所有这里存储的路径长度实际上是路径上的节点数
//这里默认节点编号从1开始，到G.vexnum
//MPL[k][0]存储k出发的最长路径顶点数，MPL[k][1…]存储顶点
```

```
Status problem_7_36(ALGraph G, int MPL[][MAX_VERTEX_NUM + 1])
{
    int Topo[MAX_VERTEX_NUM + 1];              //存储拓扑序列
    int i, j, k, max, tmp;
    ArcNode *r;

    if (problem_7_34(G, Topo))                 //拓扑排序
    {
        for (i = 0; i <= MAX_VERTEX_NUM; i++) //初始化最长路径数组
            for (j = 0; j <= MAX_VERTEX_NUM; j++)
                MPL[i][j] = 0;
        MPL[0][0] = G.vexnum;                  //总路径数
        for (k = G.vexnum; k >= 1; k--)            //逆拓扑序列求各顶点最长路径
        {
            r = G.vertices[Topo[k]].firstarc;
            MPL[Topo[k]][1] = Topo[k];
            if (!r)
                MPL[Topo[k]][0] = 1;
            else
            {
                max = tmp = 0;
                while (r)
                {
                    if (MPL[r->adjvex][0]>max)
//若某顶点有多条最长路径，则只取首先遇见的一条
                    {
                        max = MPL[r->adjvex][0];
                        tmp = r->adjvex;
                    }
                    r = r->nextarc;
                }
                MPL[Topo[k]][0] = max + 1;
//当前顶点出发的最长路径中顶点个数
                for (i = 1; i <= max; i++)
                    MPL[Topo[k]][i + 1] = MPL[tmp][i];
            }
        }
        return OK;
    }
    return ERROR;
}
```

```
//在完成7.36之后 7.37就迎刃而解了
Status problem_7_37(ALGraph G, int MPL[][MAX_VERTEX_NUM + 1], int Path[])
{
    int k, tmp;
    if (problem_7_36(G, MPL))
    {
        for (k = 1, Path[0] = tmp = 0; k< = G.vexnum; k++)
        {
            if (MPL[k][0]>Path[0])
//若有向无环图有多条最长路径，则只取首先遇见的一条
            {
                Path[0] = MPL[k][0];
                tmp = k;
            }
        }
        for (k = 1; k <= Path[0]; k++)
            Path[k] = MPL[tmp][k];
        return OK;
    }
    return ERROR;
}
```