```cpp
#include<iostream>
using namespace std;

//-----------------------------------------------无向图的邻接多重表
#define MAX_VERTEX_NUM 20
#define VisitIf bool
#define InfoType int
#define VerTexType char
#define INFINITY 16 //INT_MAX

typedef enum{ERROR,OK}Status;    //枚举型，函数状态变量

typedef struct EBox{
    VisitIf mark;                       //访问标记
    int ivex, jvex;                     //该边依附的两个顶点的位置
    struct EBox *ilink, *jlink;         //分别指向依附这两个顶点的下一条边
    InfoType *info;                     //该边信息指针
}EBox;  //边的结构
typedef struct VexBox{
    VerTexType data;                //顶点信息
    EBox *firstedge;                //指向第一条依附该顶点的边
}VexBox;    //顶点结构
typedef struct{
    VexBox adjmulist[MAX_VERTEX_NUM];
    int vexnum, arcnum;
}AMLGraph;      //邻接多重表

//若G中存在顶点u，则返回u在G中的位置，若没找到则返回INFINITY
int LocateVex(AMLGraph G, VerTexType u){
    int i=0;
    for(i=0; i<G.vexnum; i++){
        if(G.adjmulist[i].data==u)
            break;
    }
    if(i<G.vexnum)
        return i;
    else
        return INFINITY;
}

Status CreateUDG(AMLGraph &G){
    char IncInfo;   //0/1用来表示弧边是否含有辅助信息
    cout <<"Please input: vexnum(no more than 20) arcnum(no more than vexnum*vexnum) IncInfo(default 0)" << endl;
    cin >> G.vexnum >> G.arcnum >> IncInfo;                 //IncInfo=0表示弧边不含辅助信息
    cout << "构造顶点向量" << endl;
    for(int i=0; i<G.vexnum; i++){   //构造并初始化顶点向量
        cin >> G.adjmulist[i].data;
        G.adjmulist[i].firstedge = NULL;
    }

    EBox *p;
    char v1,v2; //顶点
    int v1_int, v2_int;
    cout << "以'起点 终点'的方式依次输入每一条边(例如：ab\t起点：a，终点：b)：" << endl;
    for(int k=0; k<G.arcnum; ++k){  //构造邻接矩阵
        cin >>v1>>v2;
        //定位v1,v2在G中的位置
        v1_int = LocateVex(G, v1);
        v2_int = LocateVex(G, v2);

        p = new EBox;
        //对边p赋值{mark, ivex, ivex, ilink, ilink, info}
        *p = {false, v1_int, v2_int, G.adjmulist[v1_int].firstedge, G.adjmulist[v2_int].firstedge,NULL};
        G.adjmulist[v1_int].firstedge = p;
        G.adjmulist[v2_int].firstedge = p;
            // if(IncInfo)
            //     Inuput(*G.arcs[i][j].info);
            //若弧边上有辅助信息，则输入
    }
    return OK;
}

int main(){
    AMLGraph G;
    CreateUDG(G);

    for(int i=0; i<G.vexnum; i++)
        cout << G.adjmulist[i].data << "\t";
    cout << endl;

    EBox *p;
    for(int i=0; i<G.vexnum; i++){
```

```
83            cout << G.adjmulist[i].data << " |  ";
84            p = G.adjmulist[i].firstedge;
85            while(p){
86                cout << G.adjmulist[p->ivex].data << "_" << G.adjmulist[p->jvex].data << "    ";
87                p=p->ilink;
88            }
89            cout << endl;
90        }
91
92        return 0;
93    }
94
```