```
typedef struct  // 图的定义
{
    int edges[maxsize][maxsize];     // 邻接矩阵定义，如果是有权图，则在词句中将int改为float
    int n, e;    // 分别为顶点数和边数
    int vex[maxsize];     // 存放结点信息
}MGraph;

typedef struct ArcNode
{
    int adjvex;                 // 改变所指向的结点的位置
    struct ArcNode *nextarc;    // 指向下一条边的指针
    int info;                   // 该边的相关信息(如权值) 若无要求 可以不写
}ArcNode;

typedef struct
{
    char data;       // 顶点信息
    ArcNode *firstArc;  // 指向第一条边的指针
}VNode;

typedef struct
{
    VNode adjlist[maxsize]; // 邻接表
    int n, e;    // 顶点数和边数
}AGraph;     // 图的邻接表类型

// 图的深度优先遍历
int visited[maxsize];
void DFS(AGraph *G, int v){
    ArcNode *p;
    visited[v] = 1;
    Visit(v);
    p = G->adjlist[v].firstArc;
    while(p){
        if(visited[p->adjvex] == 0)
            DFS(G, p->adjvex);
        p = p->nextarc;
    }
}
```

```
// 图的广度优先遍历
int visited[maxsize];
void BFS(AGraph *G, int v){
    ArcNode* p;
    int que[maxsize], front = 0, rear = 0;
    int j;
    Visit(v);
    visited[v] = 1;
    rear = (rear + 1) % maxsize;
    que[rear] = v;
    while (front != rear)
    {
        front = (front + 1) % maxsize;
        j = que[front];
        p = G->adjlist[j].firstArc;
        while(p){
            if(visited[p->adjvex] == 0){
                Visit(p->adjvex);
                visited[p->adjvex] = 1;
                rear = (rear + 1) % maxsize;
                que[rear] = p->adjvex;
            }
            p = p->nextarc;
        }
    }

}
```