第三次作业

1. 对于图8.49所示的无向图，给出以顶点0作为初始点的所有深度优先遍历序列和广度优先遍历序列

深度优先遍历序列： 014523 ,015423 014532,015432

021453,021543,023145,023154

032145,032154,031452,031542

广度优先遍历序列：012345,012354,013245,013254

021345,021354,031245,031254

032145,032154,032145,032154

1. 给出利用Prim算法（从顶点0开始构造）和Kruskal算法构造初的最小生成树的结果，要求结果按构造边的顺序列出。

Prim算法：(0,1),(0,2),(1,2),(2,5),(5,4)

Kruskal算法：(0,1),(0,3),(1,2),(4,5),(2,5)

1. 求图中所有顶点对之间的最短路径

以0为源点的dist数组[0,1,4,2,8,10]

以1为源点的dist数组[-1,0,-1,3,7,9](-1代表无法抵达)

0到1最短路径长度：1 路径：0，1

0到2最短路径长度：4 路径：0，1，2

0到3最短路径长度：2 路径：0，3

0到4最短路径长度：8 路径：0，1，4

0到5最短路径长度：10 路径：0，1，2，5

1到2最短路径长度：3 路径：1，2

1到4最短路径长度：7 路径：1，4

1到5最短路径长度：9 路径：1，2，5

2到5最短路径长度：6 路径：2，5

3到5最短路径长度：8 路径：3，5

实验题14，利用深度优先遍历算法输出从入口(1,1)到出口(M,N)的所有迷宫途径，迷宫用图对应的邻接表表示，写出算法求解思路

下为迷宫(起点以3表示，终点以4表示,0为通路，1为障碍)

3011

1000

0010

0140

0001

建立图的结构和邻接表如上，首先要有一个和迷宫同等大小的二维数组visit记录访问过的序列，创建一个Find函数，参数里包含起点的位置s和图的邻接表G和记录路径path的数组结构体。Find函数刚进去时将visit数组里s的位置置为1表示已访问，并将路径信息传入到path结构体中。判断当前位置是不是终点，如果是终点，打印path里存储的路径。定义指针p指向G的s头结点对应的第一条边结点。当p不等与空时循环判断边结点的是否被访问，如果没有，Find函数的起点s换成边结点的位置递归访问，然后p指下一个边节点。最后将visit里起点s的位置置为0.即调用Find前当前点标记为已访问，Find执行结束后当前点标记为未访问：前者防止递归调用时重复访问当前点走回头路，后者是为探索其他路径时能重复访问该点。如此就能找到所有的迷宫路径。