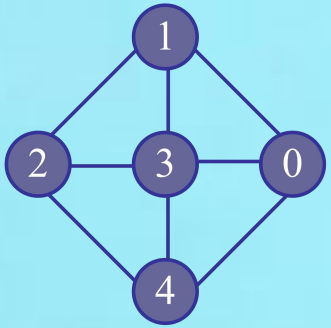
本节主题：

图的遍历

图的遍历

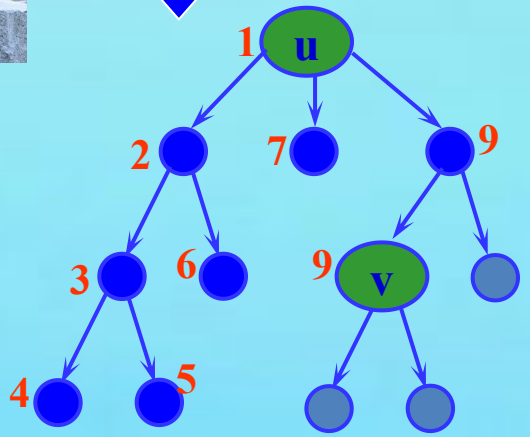
- 定义：从给定图中任意指定的顶点（称为**初始点**）出发，按照某种搜索方法沿着图的边**访问图中的所有顶点**，使**每个顶点仅被访问一次**，这个过程称为**图的遍历**。
- 结果：如果给定图是连通的无向图，或者是强连通的有向图，则遍历过程一次就能完成，并可按访问的先后顺序得到由该图**所有顶点组成的一个序列**。
- 方法
  - 深度优先搜索法（DFS）
    Deep-First-Search
  - 广度优先搜索法（BFS）
    Breadth-First-Search



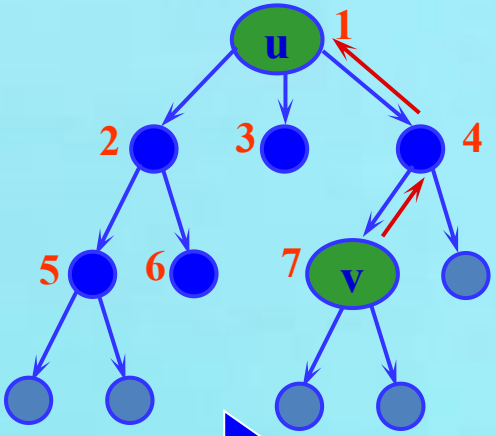
0 - 3 - 2 - 1 - 4

遍历策略

DFS
深度优先

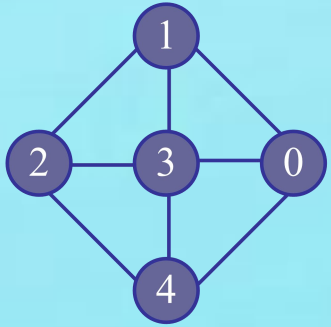


BFS
广度优先



DFS和BFS遍历图

- DFS算法过程
  - （1）首先访问初始顶点v。
  - （2）选择与顶点v相邻且没被访问过的顶点w为下一个顶点，再从w出发进行深度优先搜索。
  - （3）直到图中与当前顶点v邻接的所有顶点都被访问过为止。
- 例
  - 0 1 2 3 4
  - 0 3 2 1 4
- 实现
  - 递归/栈



- 共同问题：
  如何知道都/未被访问过？

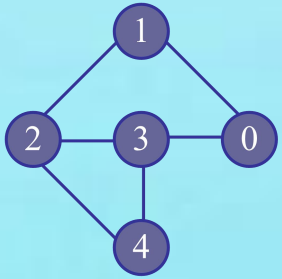
int visited[N];

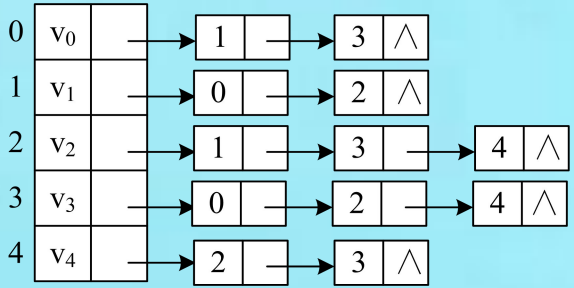
[0]	1
[1]	0
[2]	0
[3]	1
[4]	0

- BFS算法过程
  - （1）访问初始点v。
  - （2）接着访问v的所有未被访问过的邻接点$v_1,v_2,\ldots,v_t$；按照$v_1,v_2,\ldots,v_t$的次序，访问每一个顶点的所有未被访问过的邻接点。
  - （3）直到图中所有和初始点v有路径相通的顶点都被访问过为止。
- 例
  - 0 1 3 4 2
  - 1 2 3 0 4
- 实现
  - 队列

DFS实现

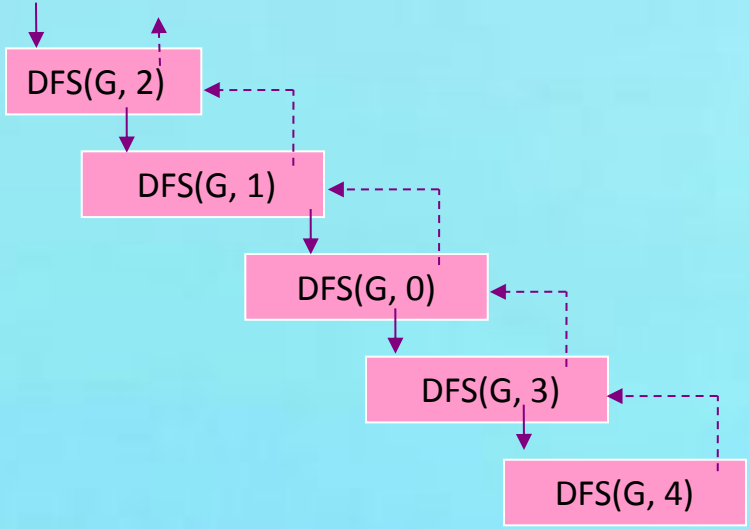
```
int visited[N];  //算法执行前全置0
void DFS(ALGraph *G , int v)
{
    ArcNode *p;
    int w;
    visited[v]=1;
    printf("%d ", v);
    p=G->adjlist[v].firstarc;
    while (p!=NULL)
    {
        w=p->adjvex;
        if (visited[w]==0)
            DFS(G,w);
        p=p->nextarc;
    }
}
```





int visited[N];

[0]	1
[1]	1
[2]	1
[3]	1
[4]	1



访问序列 2 1 0 3 4

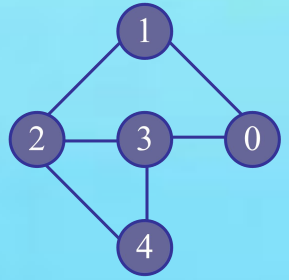
BFS实现

```
void BFS(ALGraph *G，int v)
{
   ArcNode *p;
   int w，i;
   int queue[MAXV]，front=0，rear=0;
   //定义存放节点的访问标志的visited数组
   //访问第一个顶点并入队
   while (front!=rear)
   {
      //取出队中顶点，访问未访问的邻接点并使之入队
   }
   printf("\n");
}
```

```
int visited[MAXV];
for (i=0; i<G->n; i++)
   visited[i]=0;
```

```
printf("%2d"，v);
visited[v]=1;
rear=(rear+1)%MAXV;
queue[rear]=v;
```

```
front=(front+1)%MAXV;
w=queue[front];
p=G->adjlist[w].firstarc;
while (p!=NULL)
{
   if (visited[p->adjvex]==0)
   {
      printf("%2d"，p->adjvex);
      visited[p->adjvex]=1;
      rear=(rear+1)%MAXV;
      queue[rear]=p->adjvex;
   }
   p=p->nextarc;
}
```



int queue[ ];

[0]	
[1]	
[2]	
[3]	
[4]	
[5]	

int visited[ ];

[0]	0
[1]	0
[2]	0
[3]	0
[4]	0

思考题

- 用栈求解迷宫问题，有DFS算法有什么关联？
- 用队列求解迷宫问题，有BFS算法有什么关联？