# 图论基础

1.什么是图：图（Graph）是一种非线性数据结构。可以说它是一种比较复杂的数据结构，它比树还要复杂。因为图没有层的概念，它们之间的任意元素都可能产生关系。

2.图的基本知识：

（1）顶点：

（2）边：

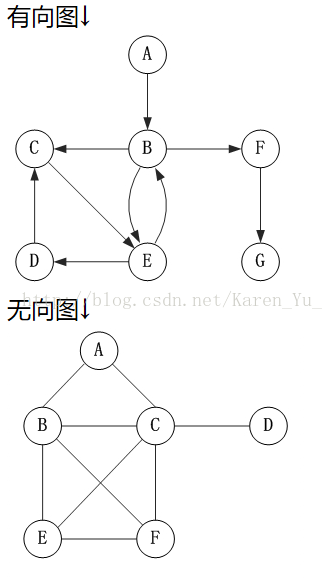
（3）顶点的度：

（4）出度、入度

（5）有向图

（6）无向图

存储:数组+链表

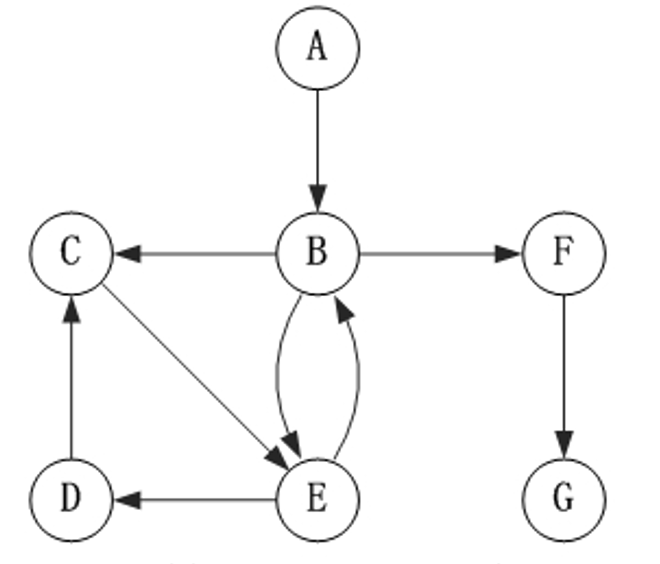


3.图这种结构我们应该如何来存储呢？

（1）邻接矩阵：

图里有x个点就是 x\*x的矩阵。

7\*7的矩阵，有0和1。



A[7][7]:巧妙的应用数组的 下标

A[1][1]:表示从1到1的情况，A[1][2]就表示1到2的情况，有边的就是1,A[1][2]=1,没有的就是0 A[1][3]=0

[null ,1 ,0 ,0 ,0 ,0 ,0 ]

[0 ,null ,1 ,0 ,1 ,1 ,0 ]

[0 ,0 ,null ,0 ,1 ,0 ,0 ]

[0 ,0 ,1 ,null ,0 ,0 ,0 ]

[0 ,1 ,0 ,1 ,null ,0 ,0 ]

[0 ,0 ,0 ,0 ,0 ,null ,1 ]

[0, ,0 ,0 ,0 ,0 ,0 ,null ]

稀疏矩阵：

0 0 0

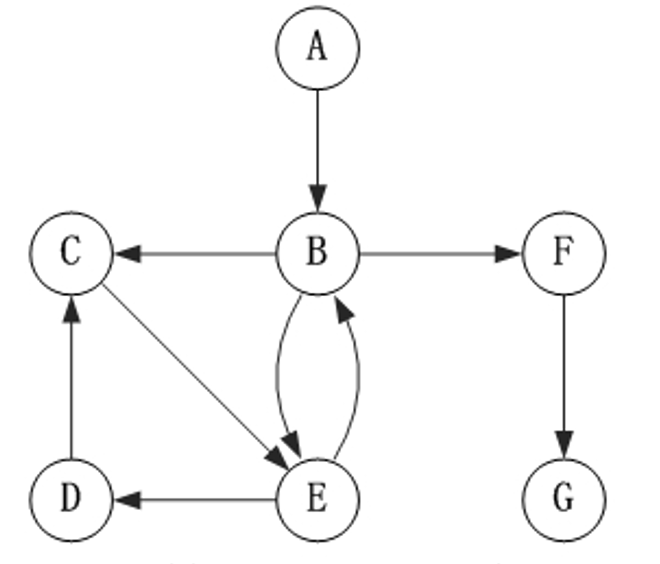
1 1 1

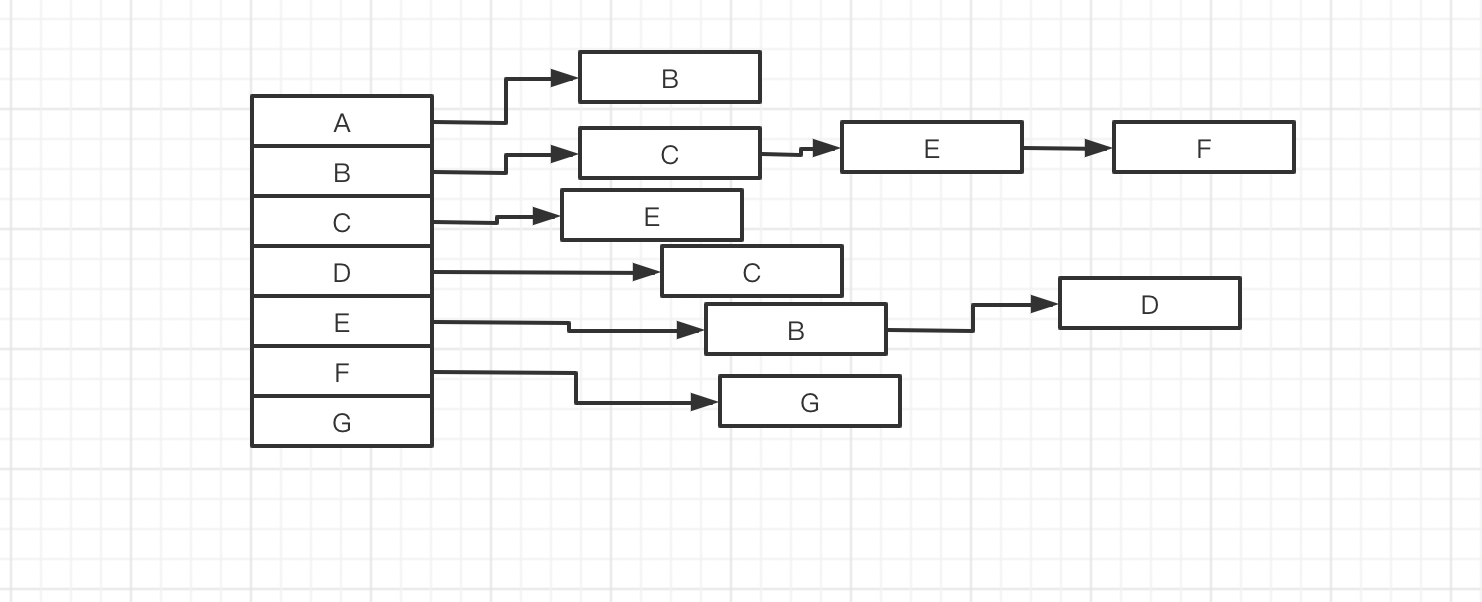
0 1 0

3\*3的矩阵

（2）邻接表：链表

数组+链表的形式：





以上两种对比：不同的情况要不同的解决

数组：浪费空间，但是速度块。数据不大 优先选用数组

链表：节省空间，但是速度慢

4.图形结构我们应该如何遍历呢？

搜索算法：

大家需要掌握最基础两种：

深度优先遍历（DFS）：大家可以想象玩迷宫，是不是选择一个方向走到底，直到不能走了你在返回一步继续试其他的方向，没错这其实就是深度优先遍历。一条路走到底，递归，有回溯。也要标记走过的点

关键的优化:剪枝

广度优先遍历（BFS）：类似于树结构的层次遍历，先找到一个点，然后把该点加入队列，依次找出该点的关联边加入队列，循环操作，一直到队列为空。

开始就把所有的路都给走了

两个关键点：队列，标记数组，加过的点不能在加。

启发式搜索，A\*

