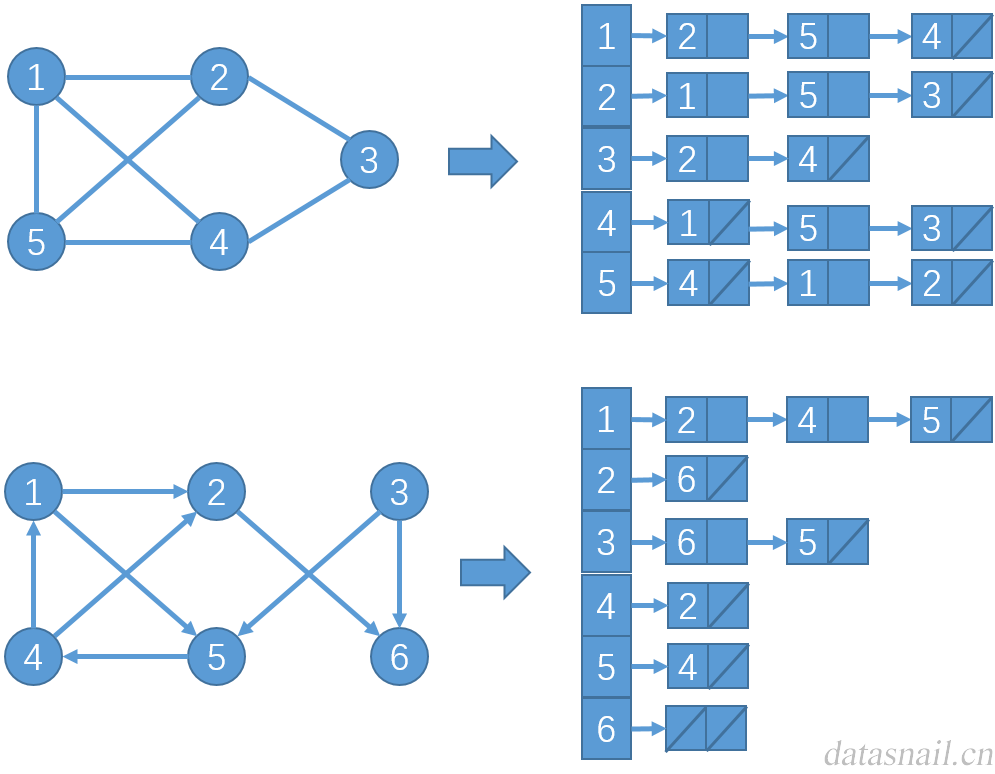
# 实验6 图的深度优先遍历和广度优先遍历

## 题目

要求图用邻接表方式存储，编写能够通过键盘输入一个无向图并存储，实现图的深度优先遍历及广度优先遍历的程序。

## 测试用例



深度优先遍历序列：略，自己补

广度优先遍历序列：略，自己补

## 算法概述

本题目要求图信息从键盘输入，也可以将图信息以文件形式存放，程序从文件获取图信息，通过文件重定向，二者可以兼容。在对图进行操作的时候，一般使用邻接矩阵表示，例如要存储如上的无向图，可以直接用一个csv或者txt文件，存储内容如下：

5,14

1,3,2,5,4

2,3,1,5,3

3,2,2,4

4,3,1,5,3

5,3,4,1,2

上述文件第一行存储了总共的节点数量、边的数量；接下来的每一行存储顶点id和其邻接点id，并且第2位数字存储了该节点总共有多少邻接节点（没有边的节点，其邻接点个数为0）。有向图的存储可以类似写出来。

## 3.1深度优先遍历

 对于当前节点i; 若节点j与节点i有连接，且j没有被遍历过，则遍历j;

DFS(G)

for each vertex u属于 G.V

u.color=WHITE

for each vertex u 属于G.V

if u.color==WHITE

DFS-VISIT(G,u)

DFS-VISIT(G,u)

for each v 属于G.Adj[u]

if v.color==WHITE

DFS-VISIT(G,v)

u.color=BLACK

## 广度优先遍历

 逐层遍历图

(1) 从源节点出发，源节点为第一层；

   (2) 对于第i层的队列

    遍历第i层的所有元素，其下一届点全部放入第i+1层队列

BFS(G,s)

for each vertex u 属于G.V-{s}

u.color=WHITE //未访问过的节点

Q=NIL //Q为广度优先遍历的节点队列

E=ENQUEUE(Q,s)

While Q!=NIL

u=DELQUEUE(Q)

for each v 属于 G.Adj[u]

if v.color==WHITE

ENQUEUE(Q,v)

u.color=BLACK