**LCA**

1. **LCA问题的简介**

LCA问题（Least Common Ancestors，最近公共祖先问题），是指给定一棵有根树T，给出若干个查询LCA(u, v)（通常查询数量较大），每次求树T中两个顶点x和y的最近公共祖先。

询问：LCA(x, y)，即求x与y的最近公共祖先。

题目描述：通常是先给定一棵有根树，然后询问（两种询问方式，下文会提及）。

1. **LCA题目的分类**

在线算法：你问我一次，我立马回答你。

离线算法：等你问完了，我再一次性回答你。

1. **LCA问题的解法**

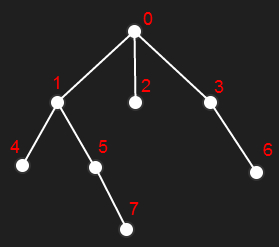


图1

以下以图1为例，分别对两个分类的算法作相应的讲解：

1. 离线算法——Tarjan(后序DFS+并查集)

【算法步骤】

1. 后序遍历访问树（假设现在访问到结点x并且x不是叶子）
2. 将以x为根的子树合并到一个集合中（集合中有共同的特性数值，我称之为ancestor，即x）
3. 然后检查是否有与x有关的查询，假设有并且x对应的y已经访问过，则得到答案（答案为y的ancestor）

应该注意的是，得到答案的顺序和查询的顺序不一样，需要另行处理。

【举例说明】查询LCA(4, 7)， LCA(5，7)：

a.访问完4之后{4}与{1}合并为{4，1}，则ancestor{1,4} = 1。

b.访问7，先遍历与7有关的查询，发现有关于7和4的查询，并且4在前面已经c.访问过了，于是得到LCA(4,7) = ancestor{1,4} = 1。

c.访问完7后{7}与{5}合并为{7，5}，则ancestor{5,7}=5。

d.此时访问5，遍历与5有关的查询，发现有关于5和7的查询，并且7在先前已经访问过，则LCA(5,7) = ancestor{5,7} = 5;

e.此时该访问1了，将{4,1}与{5,7}合并为{4,1,5,7}，则ancestor{4,1,5,7} = 1。

【相关入门题】：hdu2586

1. 在线算法——基于RMQ（先序DFS+ST算法）

【算法步骤】

1. 先序遍历树，得出欧拉序列f、各结点深度序列depth和各结点在f中第一次出现的位置序列pos。
2. 当有一个查询LCA(x，y)时，在f(x，y)中利用RMQ找到depth值最小的z，则z为LCA(x，y)。

【举例说明】查询LCA(4, 7)：

1. 先序遍历树得出有序的 f和depth序列：

f{0->1->4->1->5->7->5->1->0->2->0->3->6->3->0}

depth{0->1->2->1->2->3->2->1->0->1->0->1->2->1->0}

pos{0(0),1(1),9(2),11(3),2(4),4(5),12(6),5(7)}为方便理解，()内为结点。

1. 查询LCA(4,7)，利用RMQ求出pos(4)与pos(7)之间depth{2->1->2->3}值最小的1，其对应的结点为1，即LCA(4，7) = 1。

【相关入门题】poj2763