Bai CRIMINAL:

Bài này mang tư tưởng của thuật toán Tarjan.

Vẫn khởi tạo những dữ liệu tương tự thuật toán Tarjan : low[], num[].

Dùng tính chất thời gian vào và ra một đỉnh khi duyệt dfs để xác định quan hệ tiền bối – hậu duệ của hai đỉnh u, v : mảng tin[] là thời gian vào và tout[] là thời gian ra (tương tự như LCA). Từ đó tạo hàm thuoc(v,u) có nghĩa là v thuộc u nếu hàm trả về true và ngược lại.

Tạo thêm mảng pa[u][i] là cha thứ 2^i của u trên cây dfs. Từ đó tạo hàm FindParent(p,v) trả về nút con u của p mà trong cây dfs gốc u chứa v. (tức là tìm đỉnh cao nhất chứa v nhưng nằm trong nội bộ của p).

Còn thuật kiểm tra xem hai truy vấn đó là phụ thuộc bạn đọc tự vẽ hình và xét các trường hợp thôi, chứ không tiện nói cụ thể ở đây được (vì nó quá dài dòng, chủ yếu là sự hiểu biết của bạn đọc về thuật toán Tarjan đến đâu thôi). (Các bạn xem code để hiểu L). Xét càng nhiều trường hợp thì càng chặt, dư điều kiện thì cũng chẳng sao, miễn đúng là được.

Mình nói thêm là những dữ liệu trên tại sao mình nghĩ ra được thực chất là từ cách mình nghĩ ra thuật rồi mới hình thành chúng. Giống như tại sao mình tạo mảng pa[u][i], thực tế ban đầu không có, nhưng sau khi xét đến trường hợp cần tìm đỉnh cao nhất chứa v nằm trong nội bộ p, thì mình mới liên tưởng đến thuật tìm LCA trong O(log).

Theo mình nhận xét thì bài này khó, kết hợp nhiều thuật toán.