最近公共祖先——
欧拉序RMQ算法代码实现

主讲人：邓哲也

欧拉序RMQ算法代码实现

- 树的存储结构：
- struct edge{
- int v, next;
- }e[M];
- int h[N], etot;

- 采用邻接链表来存树，其中h[]是头数组，e[]是边数组

欧拉序RMQ算法代码实现

- 添加边
- void add_edge(int x, int y) {
- e[etot].v = y;
- e[etot].next = h[x];
- h[x] = etot ++;
- }

- h[]初始化为-1，etot初始化为0

欧拉序RMQ算法代码实现

- dfs，得到欧拉序列、每个点第一次出现在欧拉序列的位置，和每个点的深度信息。

```
void dfs(int u, int fa, int dep) {
    ++ cnt;
    f[cnt][0] = dep;
    g[cnt][0] = u;
    fir[u] = cnt;
    for (int i = h[u]; i != -1; i = e[i].next) {
        if (e[i].v != fa) {
            dfs(e[i].v, u, dep + 1);
            ++ cnt;
            f[cnt][0] = dep;
            g[cnt][0] = u;
        }
    }
}
```

欧拉序RMQ算法代码实现

- 初始化ST表，注意这里不仅要记录最小值f[][]，还要记录最小值出现的位置g[][]。

```
void init_rmq(int x, int y) {
    for (int j = 1; (1 << j) <= cnt;j ++)
        for (int i = 1; i + (1 << j) - 1 <= cnt; i ++) {
            int k = i + (1 << j);
            if (f[i][j - 1] < f[k][j - 1]) {
                f[i][j] = f[i][j - 1];
                g[i][j] = g[i][j - 1];
            } else {
                f[i][j] = f[k][j - 1];
                g[i][j] = g[k][j - 1];
            }
        }
}
```

欧拉序RMQ算法代码实现

- 找到u和v两个第一次在欧拉序列中出现的位置，用ST表找到最小值对应的节点。

```
int lca(int u, int v) {
    u = fir[u];
    v = fir[v];
    if (u > v) swap(u, v);
    int j = lg[v - u + 1];
    int k = v - (1 << j) + 1;
    if (f[u][j] < f[k][j])
        return g[u][j];
    else
        return g[k][j];
}
```

下节课再见