树的直径

Diameter of Tree

Concept：树的直径是指树中最长的一条简单路径的长度

两次 dfs

Theorem：从任意一点开始 dfs 一遍，找到距离起点最远的点 x，则 x 一定是直径的一个端点；再从x 开始 dfs 一遍，找到距离 x 最远的点 y，则 x 和 y 就是树的直径。

Complexity：$O(n)$

Code：

```cpp
int dis[N];
void dfs(int x, int f, int d, int &p){
    dis[x] = d;
    if(!p || dis[p] < d)    p = x;
    for(int i = head[x]; i; i = edge[i].nxt){
        if(edge[i].to == f) continue;
        dfs(edge[i].to, x, d + edge[i].dis, p);
    }
}

int main(){
    //...
    int x = 0, y = 0;
    memset(dis, 0, sizeof dis); dfs(1, 0, 0, x);
    memset(dis, 0, sizeof dis); dfs(x, 0, 0, y);
    // now x, y are ends of the diameter
    printf("%d\n", dis[y]);
    return 0;
}
```

树形 dp

Idea：设 $dp[i]$ 表示以 i 为根的子树中，一个端点在 i 的最长路径长度，则在 dp 过程中记录子树的最大、次大长度进行转移和更新答案即可。

Complexity：$O(n)$

Code：

```
int dp[N], ans;
void dfs(int x, int f){
    int mx1 = 0, mx2 = 0;
    for(int i = head[x]; i; i = edge[i].nxt){
        if(edge[i].to == f) continue;
        dfs(edge[i].to, x);
        if(mx1 < dp[edge[i].to] + edge[i].dis)
            mx2 = mx1, mx1 = dp[edge[i].to] + edge[i].dis;
        else if(mx2 < dp[edge[i].to] + edge[i].dis)
            mx2 = dp[edge[i].to] + edge[i].dis;
    }
    dp[x] = mx1;
    ans = max(ans, mx1 + mx2);
}

int main(){
    //...
    dfs(1, 0);
    printf("%d\n", ans);
    return 0;
}
```