

# 树形dp练习1

---

## 人员

---

周子航、高可维、杨青悠、于迦浩、石宇燃、李佳声、王崇宇、孙乐涵 到课

于潇涵、王梓同 请假

## 作业

---

<https://www.luogu.com.cn/contest/158345#problems>

邀请码：bt39

A题要求10分钟默写通过，下节课会要求默写

B/C两题必做，D/E两题选做

## 课堂表现

---

周子航同学做题很快很好，整体上课纪律也不错

## 课堂内容

---

### P5731 【深基5.习6】蛇形方阵

此题目比较简单，不展示代码，要求所有同学可以做到10分钟可以写出来

树形dp：一般是通过记忆化搜索的形式，通过维护子节点信息来更新父节点信息

### P1352 没有上司的舞会

建立状态  $f[i][0/1]$

$f[i][0]$ 代表不选择点 $i$ 时，在子树中的最大值

$f[i][1]$ 代表选择点 $i$ 时，在子树中的最大值

状态转移：

$$f[i][0] = \max(f[j1][0], f[j1][1]) + \backslash$$
$$\max(f[j2][0], f[j2][1]) + \backslash$$
$$\max(f[j3][0], f[j3][1])$$
$$f[i][1] = f[j1][0] + f[j2][0] + f[j3][0]$$

$j1, j2, j3$  是点 $i$ 的子节点

答案：

$$\max(f[\text{root}][0], f[\text{root}][1])$$

```

#include <bits/stdc++.h>

using namespace std;

const int maxn = 6e3 + 5;
int w[maxn], deg[maxn];
vector<int> vec[maxn];
int f[maxn][2]; // f[i][0]: 不选 i 的时候
                // f[i][1]: 选 i 的时候

void dfs(int u) {
    for (int i : vec[u]) dfs(i);

    f[u][1] += w[u];
    for (int i : vec[u]) {
        f[u][1] += f[i][0];
        f[u][0] += max(f[i][0], f[i][1]);
    }
}

int main()
{
    int n; cin >> n;
    for (int i = 1; i <= n; ++i) cin >> w[i];
    for (int i = 1; i <= n-1; ++i) {
        int u, v; cin >> u >> v;
        vec[v].push_back(u);
        deg[u]++;
    }

    for (int i = 1; i <= n; ++i) {
        if (!deg[i]) { // i 是 root
            dfs(i);
            cout << max(f[i][0], f[i][1]) << endl;
            break;
        }
    }
    return 0;
}

```

Cut 'em all!

$f[i]$ : 代表以  $i$  为根时子树中的总节点数量

当节点总数量为奇数时，一定没有合法方案，输出  $-1$

当 $f[i]$ 是偶数时，则可以断开节点 $i$ 与其父亲节点 $u$ 的联系

$++res$ (代表多了一条切割线)

$f[i]=0$ (把 $f[i]$ 设成 $0$ ，不对其父亲节点继续做贡献，也可以不改成 $0$ )

```
#include <bits/stdc++.h>

using namespace std;

const int maxn = 1e5 + 5;
vector<int> vec[maxn];
int f[maxn];
int ans = 0;

void dfs(int u, int fa) {
    f[u] = 1;
    for (int i : vec[u]) {
        if (i == fa) continue;
        dfs(i, u);
        f[u] += f[i];
    }
    if (f[u] % 2 == 0) ++ans;
}

int main()
{
    ios::sync_with_stdio(false);
    cin.tie(0);
    int n; cin >> n;
    for (int i = 1; i <= n-1; ++i) {
        int u, v; cin >> u >> v;
        vec[u].push_back(v), vec[v].push_back(u);
    }
    if (n & 1) { cout << -1 << endl; return 0; }

    dfs(1, -1);

    cout << ans - 1 << endl;
    return 0;
}
```