# 目录

abc482 ..... 1
E.Diameter Endpoint Colors ..... 1
F.Mod M Divisor ..... 4

# abc482

## E.Diameter Endpoint Colors

### 题目描述

给定一棵有 $N$ 个顶点的树 $T$，顶点编号为 $1, 2, ..., N$。对于 $i = 1, 2, ..., N-1$，第 $i$ 条边连接顶点 $u_i$ 和 $v_i$。

每个顶点都被涂上了某种颜色；顶点 $v$ 被涂成颜色 $C_v$。

求满足以下所有条件的整数对 $(u, v)$（$1 \leq u < v \leq N$）的数量：

- 顶点 $u$ 和顶点 $v$ 的颜色相同；
- 连接顶点 $u$ 和顶点 $v$ 的简单路径的长度等于树 $T$ 的直径。

这里，路径的长度是其所包含的边数，树的直径是树中简单路径的最大长度。

### 约束条件

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq u_i, v_i \leq N$
- 给定的图是一棵树。
- $1 \leq C_i \leq N$
- 所有输入均为整数。

### 输入格式

输入由标准输入给出，格式如下：

```
N
u_1 v_1
u_2 v_2
...
u_(N-1) v_(N-1)
C_1 C_2 ... C_N
```

### 输出格式

输出答案。

### 样例

#### 1.in / 1.out

```
5
1 2
2 3
2 4
4 5
2 1 3 1 2
```

```
1
```

满足条件的整数对有 1 个：$(1, 5)$。

**2.in / 2.out**

```
8
3 2
5 8
2 7
8 1
2 8
1 4
6 5
2 1 5 2 6 5 2 5
```

```
2
```

满足条件的整数对有 2 个：$(3, 6)$ 和 $(4, 7)$。

**3.in / 3.out**

```
9
2 3
4 8
6 1
3 6
1 9
8 6
2 5
7 2
3 2 1 3 3 4 2 3 3
```

```
2
```

满足条件的整数对有 2 个：$(4, 5)$ 和 $(5, 9)$。

**题解**

差一点差一点差一点！要是我再厉害一点点，是不是就能过了呀……

哎，下次加油吧。

这题考察了树直径的性质：

1. 从一个点出发找最远的点，再从这个点出发找最远的点，这两个点就是其中一条直径。
2. 一条树可能有多条直径，但所有的直径一定过任意一条直径的中心（如果直径有奇数个点，就是最中间的一个点；如果直径有偶数个点，就是中间的一条边，或者说两个点都必定过）

所以我们首先找到一个中心，然后以中心为根进行 dfs 查深度。

若现在的深度为 d，我们只需要一个**不在当前子树（根的某个子节点的子树）内**，且深度为 D-d 的点，颜色匹配就可以计入答案。

所以我们用（颜色，深度）两个关键字建立 map，在这个子树处理完之后再把当前子树的结果加入答案集合。

为什么是这个子树？因为只要不在子树内，这条路径就必须过根。

这里是代码归纳的问题了。如果我们有两个中心节点，我们只需要用任意一个作为根进行相同的处理就好了。

为什么？因为直径必定过这个点。虽然我们知道事实上直径必定要过另一个点，但对于其他子树这样处理也是安全的，不会计算多余的答案。

如果我当时能意识到这部分，我可能就能 AC 这道题……

```cpp
#include <map>
#include <stdio.h>
#include <vector>
#define maxn 200007

int li[maxn];
std::vector<int> edge[maxn];
int n, u, v;

int mdp, mdpi;
int fas[maxn], depth[maxn];

void dfs(int index, int fa) {
    depth[index] = depth[fa] + 1;
    fas[index] = fa;
    if (depth[index] > mdp)
        mdp = depth[index], mdpi = index;
    for (auto x : edge[index]) {
        if (x != fa)
            dfs(x, index);
    }
}

typedef long long i64;

i64 ans = 0;
int col[maxn];
int D;

std::map<std::pair<int, int>, int> cts;
std::map<std::pair<int, int>, int> tadd;

void dfs2(int index, int fa) {
    depth[index] = depth[fa] + 1;
    fas[index] = fa;

    for (auto x : edge[index])
        if (x != fa)
            dfs2(x, index);
    ans += cts[{col[index], D - depth[index]}];
    tadd[{col[index], depth[index]}]++;
}
```

```cpp

int main() {
    scanf("%d", &n);
    for (int i = 1; i < n; i++)
        scanf("%d%d", &u, &v), edge[u].push_back(v), edge[v].push_back(u);

    mdp = mdpi = 0;
    dfs(1, 0);
    int a1 = mdpi;
    mdp = mdpi = 0;
    dfs(a1, 0);

    D = depth[mdpi] - 1;

    for (int i = 1; i <= n; i++)
        scanf("%d", col + i);

    int cur = mdpi;
    for (int i = 1; i <= (depth[mdpi] - 1) / 2; i++)
        cur = fas[cur];
    depth[cur] = 0;
    cts[{col[cur], 0}]++;
    for (auto x : edge[cur]) {
        dfs2(x, cur);
        for (auto h : tadd) {
            cts[h.first] += h.second;
        }
        tadd.clear();
    }

    printf("%lld", ans);
    return 0;
}
```

## F.Mod M Divisor

### 题目描述

给定正整数 $N$ 和 $M$。对每个 $k = 0, 1, ..., M-1$，解决以下问题：

- 判断是否存在满足以下条件的正整数 $t$，若存在则求出最小的 $t$：
- 存在 $N^t$ 的一个正约数 $x$，使得 $x \equiv k \pmod{M}$。

### 约束条件

- $1 \leq N \leq 10^{14}$
- $1 \leq M \leq 4 \times 10^5$
- $N$ 和 $M$ 都是整数。

### 输入格式

输入由标准输入给出，格式如下：

```
1 N M
```

**输出格式**

输出 $M$ 行。第 $i$ 行：对 $k = i - 1$，若存在满足条件的正整数 $t$，则输出最小的 $t$；否则输出 $-1$。

**样例**

**1.in / 1.out**

```
1 6 7
```

```
1 -1
2 1
3 1
4 1
5 2
6 2
7 1
```

例如当 $k = 3$ 时，$x = 3$ 是 $N^1 = 6$ 的约数且满足 $x \equiv k \pmod{M}$，所以满足条件的最小正整数 $t$ 为 $t = 1$。

**2.in / 2.out**

```
1 4 8
```

```
1 2
2 1
3 1
4 -1
5 1
6 -1
7 -1
8 -1
```

**3.in / 3.out**

```
1 1 5
```

```
1 -1
2 1
3 -1
4 -1
5 -1
```

**题解**

TODO