

综合练习

人员

阮文璋、褚锦轩、王承周 到课

上周作业检查

上周作业链接: <https://cppoj.kids123code.com/contest/2455>

#	用户名	姓名	编程分	时间	A	B	C	D
1	wangchengzhou	王承周	400	41929	100	100	100	100
2	ruanwenzhang	阮文璋	300	40741	100	100	100	
3	xuruiqian	许睿谦	200	32213	100		100	
4	wangyibo	王毅博	197	558	100		97	
5	chujinxuan	褚锦轩	100	713	100			

本周作业

<https://cppoj.kids123code.com/contest/2585> (课上讲了 A ~ C 题, 课后作业是 C 题)

课堂表现

今天给同学们找了几道 trie 树、hash、树状数组的练习题, 许多同学已经把之前的知识遗忘很多了, 第一题都做的不太好, 还需要多复习。

课堂内容

[POI 2002] 商务旅行 (上周作业)

直接用 LCA 维护即可

```
#include <bits/stdc++.h>
#define int long long

using namespace std;

const int N = 2e5 + 5, M = 20;
vector<int> vec[N];
int dis[N], f[N][M];

void dfs(int u, int fa) {
    dis[u] = dis[fa]+1, f[u][0] = fa;
    for (int i = 1; i < M; ++i) f[u][i] = f[f[u][i-1]][i-1];
```

```

    for (int i : vec[u]) {
        if (i != fa) dfs(i, u);
    }
}

int lca(int a, int b) {
    if (dis[a] < dis[b]) return lca(b, a);

    for (int i = M-1; i >= 0; --i) {
        if (dis[f[a][i]] >= dis[b]) a = f[a][i];
    }

    if (a == b) return a;

    for (int i = M-1; i >= 0; --i) {
        if (f[a][i] != f[b][i]) a = f[a][i], b = f[b][i];
    }

    return f[a][0];
}

int get_value(int a, int b) {
    int id = lca(a, b);
    return dis[a] + dis[b] - 2*dis[id];
}

signed main()
{
    ios::sync_with_stdio(false);
    cin.tie(0);

    int n; cin >> n;
    for (int i = 1; i <= n-1; ++i) {
        int a, b; cin >> a >> b;
        vec[a].push_back(b), vec[b].push_back(a);
    }

    dfs(1, 0);

    int m, last = 1, res = 0; cin >> m;
    while (m-- ) {
        int x; cin >> x;
        res += get_value(last, x);
        last = x;
    }
    cout << res << endl;
    return 0;
}

```

Karuta

trie 树, 把所有字符串插入到 trie 树中, 最后看每个字符串能匹配多深

```

#include <bits/stdc++.h>

using namespace std;

const int N = 5e5 + 5, M = 26 + 5;
int tr[N][M], idx = 0;
int cnt[N];
string str[N];

void tr_insert(string s) {
    int p = 0;
    for (char i : s) {
        int u = i - 'a';
        if (!tr[p][u]) tr[p][u] = ++idx;
        p = tr[p][u];
        ++cnt[p];
    }
}

int tr_query(string s) {
    int p = 0, res = 0;
    for (char i : s) {
        int u = i - 'a';
        p = tr[p][u];
        if (cnt[p] == 1) break;
        ++res;
    }
    return res;
}

int main()
{
    int n; cin >> n;
    for (int i = 1; i <= n; ++i) cin >> str[i], tr_insert(str[i]);

    for (int i = 1; i <= n; ++i) cout << tr_query(str[i]) << endl;
    return 0;
}

```

贪婪大陆

求 $l \sim r$ 这一段被多少个区间加过，就是用 r 前面左端点的数量 减去 $l-1$ 前面右端点的数量，可以用两个树状数组维护

```

#include <bits/stdc++.h>

using namespace std;

const int maxn = 1e5 + 5;
int tr_1[maxn], tr_2[maxn];

```

```

int lowbit(int x) { return x&(-x); }
void update(int tr[], int x, int k) {
    while (x < maxn) tr[x] += k, x += lowbit(x);
}
int query(int tr[], int x) {
    int res = 0;
    while (x) res += tr[x], x -= lowbit(x);
    return res;
}

int main()
{
    int n, m; cin >> n >> m;
    while (m -- ) {
        int op, l, r; cin >> op >> l >> r;
        if (op == 1) update(tr_1, l, 1), update(tr_2, r, 1);
        else cout << query(tr_1, r) - query(tr_2, l-1) << endl;
    }
    return 0;
}

```

[加油武汉] 体温调查

二分答案 + LCA

先做一遍 dfs, 把所有叶子按顺序找出来, 然后二分答案求最小值即可

```

#include <bits/stdc++.h>
#define int long long

using namespace std;

const int N = 2e5 + 5, M = 20;
struct node {
    int to, value;
    bool operator < (const node& p) const { return to < p.to; }
};
vector<node> vec[N];
int dis[N], f[N][M];
int p[N];
int deg[N], w[N], c[N], L = 0;
int n, K;

void dfs(int u, int fa, int len) {
    if (deg[u] == 1 && u != 1) w[++L] = u;

    dis[u] = dis[fa]+1, f[u][0] = fa, p[u] = p[fa] + len;
    for (int i = 1; i < M; ++i) f[u][i] = f[f[u][i-1]][i-1];

    for (node it : vec[u]) {

```

```
        if (it.to != fa) dfs(it.to, u, it.value);
    }

int lca(int a, int b) {
    if (dis[a] < dis[b]) return lca(b, a);

    for (int i = M-1; i >= 0; --i) {
        if (dis[f[a][i]] >= dis[b]) a = f[a][i];
    }

    if (a == b) return a;

    for (int i = M-1; i >= 0; --i) {
        if (f[a][i] != f[b][i]) a = f[a][i], b = f[b][i];
    }

    return f[a][0];
}

int get_value(int a, int b) {
    int u = lca(a, b);
    return p[a] + p[b] - 2*p[u];
}

bool check(int mid) {
    for (int i = 1; i <= L; ++i) {
        if (p[w[i]]*2 > mid) return false;
    }

    int cnt = 1, sum = p[w[1]];
    for (int i = 2; i <= L; ++i) {
        if (sum + c[i] + p[w[i]] <= mid) sum += c[i];
        else ++cnt, sum = p[w[i]];
    }
    return cnt <= K;
}

signed main()
{
    cin >> n >> K;
    for (int i = 1; i <= n-1; ++i) {
        int a, b, c; cin >> a >> b >> c; ++deg[a], ++deg[b];
        vec[a].push_back({b,c}), vec[b].push_back({a,c});
    }
    for (int i = 1; i <= n; ++i) sort(vec[i].begin(), vec[i].end());

    dfs(1, 0, 0);
    for (int i = 2; i <= L; ++i) c[i] = get_value(w[i-1], w[i]);

    int l = 1, r = 1e18;
    while (l <= r) {
        int mid = (l + r) / 2;
        if (check(mid)) r = mid-1;
    }
}
```

```
    else l = mid+1;
}
cout << l << endl;
return 0;
}
```