Solution 8.4

T1 字符串

签到题。反转串后按字符开头 sort，然后相同字符开头的按评分 sort。
或者读入时按字符结尾归类，然后一类的按评分 sort。0(1) 查询输出。

写法一：

```
#include <bits/stdc++.h>
using namespace std;

const int A = 1e5 + 5;
int n, m;
struct node {
	int val, id;
	string x;
	inline void reverse() {
	int len = x.size() - 1;
		for (int i = 0; i <= len / 2; i++) swap(x[i], x[len - i]); return;
	}
} a[A];
int w[A];

inline bool cmp1(node u, node v) { return u.x < v.x; }

inline bool cmp2(node u, node v) {
	if (u.val != v.val) return u.val > v.val;
	return u.id < v.id;
}

signed main() {

	cin >> n >> m;
	for (int i = 1; i <= n; i++) {
		cin >> a[i].x >> a[i].val;
		a[i].id = i;
		a[i].reverse();
	}
	sort(a + 1, a + 1 + n, cmp1);
	for (int i = 1; i <= n; i++) {
```

```
            w[a[i].x[0] - 'a'] = i;
            if (a[i].x[0] != a[i - 1].x[0]) {
            int pos = i;
            while (a[pos + 1].x[0] == a[pos].x[0]) pos++;
                sort(a + i, a + pos + 1, cmp2);
                i = pos;
            }
        }

        w[26] = n + 1;
        for (int i = 26; ~i; i--) if (!w[i]) w[i] = w[i + 1];

        for (int i = 1; i <= n; i++) a[i].reverse();
        while (m--) {
            char x; cin >> x;
            int k; cin >> k;
            int num = x - 'a';
            if (w[num + 1] - w[num] < k)
                puts("Orz YYR tql");
            else
                cout << a[w[num] + k - 1].x << '\n';
        }
        return 0;
}
```

写法二：

```
#include <bits/stdc++.h>
#define ll long long
using namespace std;
int n, m;
struct pr {
        int id, sc;
        string c;
};
vector<pr> a[28];

char s[55];
bool cmp(pr x, pr y) { return x.sc != y.sc ? x.sc > y.sc : x.id < y.id; }
int main() {
        scanf("%d%d", &n, &m);
        for (int x, i = 1; i <= n; ++i) {
        scanf("%s%d", s, &x);
        a[s[strlen(s) - 1] - 'a'].push_back(pr{i, x, s});
}
for (int i = 0; i < 26; ++i) sort(a[i].begin(), a[i].end(), cmp); int p;
```

```
    while (m--) {
        scanf("%s%d", s, &p);
        s[0] -= 'a';
        if (p > a[s[0]].size())
            printf("Orz YYR tql\n");
        else
            cout << (a[s[0]][p - 1].c) << endl;
    }
    return 0;
}
```

T2 公约数

首先枚举 a, b，预处理出每个 gcd 的出现次数， 然后枚举每个 gcd 与 [1, n] 中的数，将贡献相加即可。

【核心代码】

```
for(int i = 1; i <= n; i++)
    for(int j = 1; j <= n; j++)
        ++tong[gcd(i, j)];
for(int i = 1; i <= n; i++)
    for(int j = 1; j <= n; j++)
        ans += tong[i] * gcd(i, j);
```

T3 数独游戏

爆搜 + 合理的剪枝，类似 NOIP 靶形数独

T4 入侵攻击

并查集，我们可以对 d 数组排序，离线处理这个问题。

n 最大 1000，因此我们可以预处理 n^2 条边然后排序，把边从小到大依次加入并查集中。

对于当前 d 询问和 (0, 0, 0) 点相连的连通块大小

【核心代码】

```
    cin >> n >> m;
    for (int i = 1; i <= n; i++) {
        cin >> p[i].x >> p[i].y >> p[i].z;
    }
    p[++n] = (point){ 0, 0, 0 };

    for (int i = 1; i <= n; i++) {
        for (int j = 1; j < i; j++) {
```

```
            insert(i, j, dist(p[i], p[j]));
        }
    }

    sort(e + 1, e + cnt + 1, cmp);
    for (int i = 1; i <= n; i++) {
        father[i] = i;
        sz[i] = 1;
    }

    for (int i = 1; i <= m; i++) {
        cin >> q[i].d;
        q[i].id = i;
    }

    sort(q + 1, q + m + 1, cmpr);

    int now = 1;
    for (int i = 1; i <= m; i++) {
        while (now <= cnt && e[now].w <= q[i].d) {
            int x = find(e[now].u);
            int y = find(e[now].v);
            if (x != y) {
                father[x] = y;
                sz[y] += sz[x];
            }
            now++;
        }

        ans[q[i].id] = sz[find(n)] - 1;
    }

    for (int i = 1; i <= m; i++) {
        printf("%d\n", ans[i]);
    }
```