题目不难, 当热身了。

## 连续自然数和

因为是连续,所以构成等差数列,设首项为 $a_1$ ,末项为 $a_n$ .

则这一段的和为
$$\frac{(a_1+a_n)\times(a_n-a_1+1)}{2}$$
,我们要让这个等于 $m$ ,

$$\displaystyle \lim \frac{(a_1+a_n)\times (a_n-a_1+1)}{2}=m.$$

不妨设 $x = a_1 + a_n, y = a_n - a_1 + 1$ ,则原式可化为 $x \times y = 2m$ .

同时又有 $(x-y)+1=2\times a_1$ ,将两式联立,可得 $\frac{2m}{y}-y+1=2\times a_1$ .

所以我们可以枚举y,进而求出 $a_1$ 和 $a_n$ ,然后验证一下式子是否成立就行了。

# 复杂度O(m).

```
#include <bits/stdc++.h>
   #define int long long
   using namespace std;
   signed main() {
        ios::sync with stdio(0);
7
        cin.tie(0);
8
        int m; cin >> m;
10
        vector<array<int, 2> > ans;
11
        auto calc = [](int a1, int an) {
12
            return (a1 + an) * (an - a1 + 1) / 2LL;
13
14
        };
        for (int y = 1; y \le 2 * m; ++y) {
15
            if (2 * m % y) continue;
16
17
            int tem = 2 * m / y - y + 1;
18
            if (tem <= 0 || tem % 2) continue;
19
            int a1 = tem / 2;
            int an = v + a1 - 1;
20
21
            if (a1 == an) continue;
            if (calc(a1, an) == m) {
22
23
                ans.push back({a1, an});
24
            }
25
26
        sort(ans.begin(), ans.end());
        for (auto &&[x, y] : ans) cout << x << ' ' << y << '\n';
27
        return 0;
28
```

## 笨小猴

按题意模拟即可。

```
1 #include <bits/stdc++.h>
    #define int long long
   using namespace std;
 4
    signed main() {
 5
 6
        ios::sync with stdio(0);
 7
        cin.tie(0);
 8
 9
        string s; cin >> s;
        vector<int> cnt(26);
10
11
        for (int i = 0; i < s.size(); ++i) {
12
            cnt[s[i] - 'a']++;
13
        int maxx = -1, minn = (int)(1e18);
14
        for (int i = 0; i < 26; ++i) {
15
            if (cnt[i] == 0) continue;
16
17
            maxx = max(maxx, cnt[i]);
18
            minn = min(minn, cnt[i]);
19
20
        auto check = [] (int x) -> bool {
21
            if (x \le 1) return 0;
22
23
            if (x == 2) return 1;
24
            for (int i = 2; i \le sqrt(x); ++i) {
                if (x % i == 0) return 0;
25
26
27
            return 1;
28
        };
        if (check(maxx - minn)) {
29
            cout << "Lucky Word\n" << maxx - minn << '\n';</pre>
31
           return 0;
33
        cout << "No Answer\n0\n";</pre>
        return 0;
34
35
```

#### 洣宫

数据很小, 所以直接dfs即可。

```
1
    #include <bits/stdc++.h>
 2
    #define int long long
 3
   using namespace std;
 4
 5
   int dx[] = \{1, -1, 0, 0\};
    int dy[] = \{0, 0, 1, -1\};
 6
7
8
    signed main() {
9
        ios::sync with stdio(0);
10
        cin.tie(0);
11
12
        int n, m, ; cin >> n >> m >> ;
13
        array<int, 2> st, ed; cin >> st[0] >> st[1] >> ed[0] >> ed[1];
        vector pass (n + 1, vector(m + 1, 1));
14
15
        vector vis (n + 1, vector(m + 1, 0));
16
        while ( --) {
17
            int x, y; cin >> x >> y;
18
            pass[x][y] = 0;
19
20
21
        int ans = 0;
22
        auto dfs = [\&] (auto dfs, int x, int y) -> void {
23
            if (x == ed[0] && y == ed[1]) {
24
                ans++;
25
                return;
26
            }
27
28
            for (int i = 0; i < 4; ++i) {
29
                int tox = x + dx[i];
                 int toy = y + dy[i];
31
                 if (1 \le tox \&\& tox \le n \&\& 1 \le toy \&\& toy \le m \&\&
    pass[tox][toy] && !vis[tox][toy]) {
32
                    vis[tox][toy] = 1;
33
                     dfs(dfs, tox, toy);
34
                     vis[tox][toy] = 0;
35
36
            }
37
        };
38
        vis[st[0]][st[1]] = 1;
39
        dfs(dfs, st[0], st[1]);
        cout << ans << '\n';
40
41
        return 0;
42 }
```

## 合并果子

贪心,每次把最小的两堆合并,这样累加的时候一定加的是最小的。用优先队列实现即可。

```
#include <bits/stdc++.h>
 1
 2
    #define int long long
 3
   using namespace std;
 4
 5
   int dx[] = \{1, -1, 0, 0\};
    int dy[] = \{0, 0, 1, -1\};
 7
 8
    signed main() {
 9
        ios::sync with stdio(0);
10
        cin.tie(0);
11
12
        int n; cin >> n;
13
        priority queue<int, vector<int>, greater<int> > q;
        for (int i = 1; i \le n; ++i) {
14
15
            int x; cin >> x;
16
            q.push(x);
17
        }
18
        int ans = 0;
19
        while (q.size() >= 2) {
20
            int x = q.top(); q.pop();
21
            int y = 0;
22
            if (!q.empty()) {
23
                y = q.top(); q.pop();
24
25
            ans += x + y;
26
            q.push(x + y);
27
28
        cout << ans << '\n';
        return 0;
29
30
```

## 产生数

数据比较小,所以可以跑一遍floyd,求出一个数能够转换成其他数的个数,然后乘法原理计数即可。

```
1 #include <bits/stdc++.h>
   #define int long long
   using namespace std;
4
   using i128 = int128;
5
   int dx[] = \{1, -1, 0, 0\};
7
   int dy[] = \{0, 0, 1, -1\};
8
   void print(i128 x) {
9
10
        if (x < 0) {
11
            putchar('-');
```

```
12
           X = -X;
13
14
        if (x > 9)
15
            print(x / 10);
        putchar(x % 10 + '0');
16
17
18
19
    signed main() {
20
        ios::sync with stdio(0);
21
        cin.tie(0);
22
23
        string n;
24
        int k; cin >> n >> k;
25
        vector pass(11, vector(11, 0));
26
        vector<int> a(10);
27
        for (int i = 1; i \le k; ++i) {
28
            int u, v; cin >> u >> v;
29
            pass[u][v] = 1;
30
31
        for (int i = 1; i \le 9; ++i) {
32
            for (int 1 = 0; 1 \le 9; ++1) {
33
                for (int r = 1; r \le 9; ++r)
34
                    if (pass[l][i] && pass[i][r]) pass[l][r] = 1;
35
36
37
        for (int i = 0; i \le 9; ++i) {
38
           pass[i][i] = 1;
39
            for (int j = 0; j \le 9; ++j)
40
                if (pass[i][j]) a[i]++;
41
42
        i128 \text{ ans} = 1;
43
        for (int i = 0; i < n.size(); ++i) {
44
            ans *= a[n[i] - '0'];
45
        print(ans);
46
47
        return 0;
48
```