# 第 1 章 atcode 066

## 1.1 arc066

### 1.1.1 c-Lining Up

首先不管怎么样每个数都不能出现超过两次（左边一个右边一个然后就没了），否则就是不合法状态对于长度为偶数的情况，显然 0 不会出现
对于长度为奇数的情况，显然 0 只会出现一次
所以把不合法状态判一下，答案就是 2n/2（每对数两种排列）

```
#include <bits/stdc++.h>
using namespace std;
const int N = 1e5 + 10;
const int mod = 1e9 + 7;
#define ll long long
int n;
int a[N];
int cnt[N];
ll ans = 0;
ll power(int a, int b)
{
    ll ans = 1, base = a;
    while (b) {
        if (b & 1)
            ans = ans * base % mod;
        base = base * base % mod;
        b >>= 1;
    }
    return ans;
}
int main()
{
    scanf("%d", &n);
    for (int i = 1; i <= n; i++)
        scanf("%d", &a[i]), cnt[a[i]]++;
    for (int i = 0; i < n; i++) {
        if (cnt[i] > 2)
            return puts("0"), 0;
    }
    if (n & 1 && cnt[0] != 1)
        return puts("0"), 0;
    if (!(n & 1) && cnt[0])
        return puts("0"), 0;
    printf("%lld\n", power(2, n / 2));
}
```

### 1.1.2 d- Xor Sum

暴力枚举，也就是每次都加 1，然后判断一下是不是能过，能过就是最小的，没错。。

```cpp
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll mod = 1e9 + 7;
map<ll, ll> dp;
ll Dp(long long x)
{
    if (dp[x])
        return dp[x];
    else {
        return dp[x] = ((Dp(x / 2) + Dp((x - 1) / 2) + Dp((x - 2) / 2))) % mod;
    }
}
int main()
{
    dp[0] = 1;
    dp[1] = 2;
    long long n;
    cin >> n;
    cout << Dp(n);
    return 0;
}
```

# 第 2 章 atcode 067

## 2.1 arc067

### 2.1.1 c-Factors of Factorial

求 n 的阶乘的因子个数将这个数表示成质因子的乘积 $n! = 2^a + 3^b + 5^c + 7^d$.....
因子个数就是 $(a+1)*(b+1)*(c+1)*$..... 即对 2 来说可以选择 0 到 a 个

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll MOD = 1e9 + 7;
const int INF = 1e5 + 100;
int f[INF];
int main()
{
    int n, a;
    cin >> n;
    memset(f, 0, sizeof(f));
    for (int i = 2; i <= n; i++) {
        a = i;
        for (int j = 2; j <= a; j++) {
            while (a % j == 0) {
                f[j]++;
                a = a / j;
            }
        }
        if (a != 1)
            f[a]++;
    }
    ll ans = 1;
    for (int i = 1; i <= n; i++) {
        if (f[i] != 0)
            ans = (ans * (f[i] + 1)) % MOD;
    }
    cout << ans % MOD << endl;
    return 0;
}
```

### 2.1.2 d-Iroha and a Grid

城镇按升序排列，因此最好按顺序访问它们.
此外，如果你想传送，你应该从一开始就去下一个城市
为了在城镇和城镇或传送之间行走
你可以看到两者中哪一个不那么疲惫

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
ll solve()
{
    ll N, A, B;
    cin >> N >> A >> B;
```

```
    vector<ll> X(N);
    for (auto& in : X)
        cin >> in;
    ll res = 0;
    for (int i = 0; i < N - 1; i++) {
        res += min(A * (X[i + 1] - X[i]), B);
    }
    return res;
}
int main(void)
{
    cin.tie(0);
    ios::sync_with_stdio(false);
    cout << solve() << endl;
    return 0;
}
```

# 第 3 章 atcode 068

## 3.1 arc068

### 3.1.1 c-X: Yet Another Die Game

一个色子，前后左右的翻转，问翻转几次，可以到达 x（开始位置任意）。

```
#include <bits/stdc++.h>
using namespace std;
#define ll long long
using namespace std;
#define N 1500
int a[N];
int main()
{
    ll n;
    scanf("%lld", &n);
    if (n <= 6) {
        cout << 1 << endl;
    } else {
        ll ans = 0;
        ll x = n / 11;
        ll y = n % 11;
        ans += 2 * x;
        if (y <= 6 && y >= 1) {
            ans++;
        } else if (y > 6) {
            ans += 2;
        }
        cout << ans << endl;
    }
    return 0;
}
```

### 3.1.2 d-An Invisible Hand

假设 a，b 和 c 是不同的数字 (a<b<c) 这个问题的可能模式是 a，b，c => b a，a，b => a a，a，a => a 在这里考虑重叠数字的模式至于 a，如果有两个 a，则可以通过使用两个来制作一个，如上所述另外，当存在三个或更多个时，可以通过使所有相同的值最终使两个或一个。因此，最好考虑结果有两个相同数字的情况，

如果有两个相同的数字，则意味着一个单独的数字就足够了。这里，为了优化行为，使用的另一个数字是尽可能重叠的数字。只是使用它

```
#include <bits/stdc++.h>
using namespace std;
#define ll long long
int main(void)
{
    cin.tie(0);
    ios::sync_with_stdio(false);
    ll N;
    cin >> N;
```

```
    vector<ll> A(N);
    for (auto& in : A)
        cin >> in;
    const int MAX_N = 1e5 + 1;
    vector<ll> C(MAX_N, 0);
    for (auto& v : A)
        C[v]++;
    sort(C.begin(), C.end(), greater<ll>());
    for (int i = 0; i < (int)C.size(); i++) {
        if (C[i] >= 3) {
            if (C[i] % 2 == 1)
                C[i] = 1;
            else
                C[i] = 2;
        }
    }
    sort(C.begin(), C.end(), greater<ll>());
    for (int i = 0; i < (int)C.size(); i++)
        if (C[i] == 2) {
            C[i] = 1;
            C[i + 1]--;
        }
    cout << accumulate(C.begin(), C.end(), 0LL) << endl;
    return 0;
}
```

# 第 4 章 atcode 069

## 4.1 arc069

### 4.1.1 c- Scc Puzzle

贪心题目，意思是相邻；两个相加的和不能超过 x, 答案是最少减去多少。

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
const int MAXN = 500000 + 5;
ll n, m;
int main()
{
    scanf("%lld%lld", &n, &m);
    if (2 * n >= m) {
        cout << min(n, m / 2) << endl;
    } else {
        ll cha = m - 2 * n;
        cout << n + cha / 4 << endl;
    }
    return 0;
}
```

### 4.1.2 d-An Ordinary Game

考虑将图像从所有山的石头状态恢复到给定的山状态。因为字典顺序最小的条件，我想尽可能从最小数量的山恢复在这里，修复过程中山峰的最佳选择是什么
事实证明，相对于给定山地州的数量，相对于从山 1 开始的山脉数量选择单调增加就足够了。那就是如果山的状态如下（样本 2）
i 1 2 3 4 5 6 7 7 8 9
a 1 2 1 3 2 4 2 5 8 1
我应该按顺序 1-> 2-> 4-> 6-> 6-> 8-> 9

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
typedef pair<ll, ll> pll;
void solve()
{
    ll N;
    cin >> N;
    vector<ll> a(N);
    for (int i = 0; i < N; i++) {
        cin >> a[i];
    }
    vector<ll> xa = a;
    sort(xa.begin(), xa.end());
    xa.erase(unique(xa.begin(), xa.end()), xa.end());
    for (int i = 0; i < N; i++) {
```

```
        a[i] = (ll)(lower_bound(xa.begin(), xa.end(), a[i]) - xa.begin());
    }
    vector<ll> imos(xa.size() + 1);
    for (int i = 0; i < N; i++) {
        imos[0]++;
        imos[a[i] + 1]--;
    }
    for (int i = 0; i < xa.size(); i++)
        imos[i + 1] += imos[i];
    vector<ll> ans(N);
    vector<pll> p;
    for (int i = 0; i < N; i++) {
        if (p.empty())
            p.push_back({ a[0], i });
        if (p[p.size() - 1].first < a[i])
            p.push_back({ a[i], i });
    }
    ll sum = 0;
    ll idx = 0;
    for (int i = 0; i < xa.size(); i++) {
        if (i == 0) {
            sum += xa[i] * imos[i];
        } else {
            sum += (xa[i] - xa[i - 1]) * imos[i];
        }

        if (p[idx].first == i) {
            ans[p[idx].second] = sum;
            sum = 0;
            idx++;
        }
    }
    for (int i = 0; i < N; i++) {
        cout << ans[i] << endl;
    }
}
int main(void)
{
    cin.tie(0);
    ios_base::sync_with_stdio(false);
    solve();
    return 0;
}
```

# 第 5 章 atcode 070

## 5.1 arc065

### 5.1.1 c-Go Home

$\sum_{i=1}^{n} \geq x$ 的 n 的最小值

```
#include <bits/stdc++.h>
using namespace std;
int x;
int main()
{
    scanf("%d", &x);
    int sum = 0;
    for (int i = 1;; i++) {
        sum += i;
        if (sum >= x) {
            printf("%d\n", i);
            return 0;
        }
    }
}
```

### 5.1.2 d-Connectivity

并查集

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
ll N, K;
vector<ll> a;
ll solve()
{
    ll res = 0;
    cin >> N >> K;
    a.resize(N);
    for (auto& in : a)
        cin >> in;
    sort(a.begin(), a.end());
    reverse(a.begin(), a.end());
    ll ans = -1;
    for (int i = 0; i < N; i++) {
        ll sum = 0;
        for (ll j = i; j < N; j++) {
            if (sum + a[j] < K) {
                sum += a[j];
                continue;
            }
            ans = max(ans, j);
        }
    }
    return res = N - ans - 1;
}
int main(void)
```

```
{
    cin.tie(0);
    ios::sync_with_stdio(false);
    cout << solve() << endl;
    return 0;
}
```