2023QDEZ男人八题线上同步赛 赛时代码和思路

本文共 7602 字 • 阅读完需 30.5 分钟

比赛链接；赛时答疑；我的博客（此文）

A-std；**B-std**；**C-std**；**D-std**；**E-std**；**F-std**；**G-std**；**H-std**；**Ex-std**

我：$50+50+20+5+30+45+0+0+0=200\text{pts}.$

喜提二中比赛除了二中选手以外的 rk1（大雾

以下是赛时代码和思路，省略快读和头文件等。

PS：因为博客太长了不好看，所以可能把源代码里的空行删去了。

前置代码说明：

```
#include <bits/stdc++.h>
using namespace std;
#define rep(i, x) for (int i = 0; i < (x); ++i)
#define rr read()
inline int read() { // 有的时候换成了 long long
    int num = 0, flag = 1, ch = getchar();
    for (; !isdigit(ch); ch = getchar()) if (ch == '-') flag = -1;
    for (; isdigit(ch); ch = getchar()) num = num * 10 + ch - '0';
    return num * flag;
}
```

T1

一个数减 k 次一，每次操作对答案的贡献为，最后面一个 1 是倒数第几个数。

先打暴力！

```
void solve(ll n, ll k) {
    ll res = 0; while (k--)  {
        auto t = n; --n, ++res;
        ll kk = 1;
        while ((t & 1) == 0) ++res, t >>= 1, ++kk;
        cerr << n + 1 << " - " << n << ": " << kk << endl;
    } printf("%lld\n", res);
} signed main() {
    int T = rr; while (T--) {
        ll n = rr, k = rr; solve(n, k);
    } return 0;
}
```

然后想想，10^{18}，在我的知识范围内，除了数学大概也只有数学加持的动规了。

咋数学呢？先查分吧，好想。

就是 $f(r) - f(l - 1)$ 然后 $f(x)$ 怎么算。

然后发现从后面一位一位的看，倒数第 i 位的 1 似乎需要 2^i 次 -1 操作才能消掉。

发现规律，解决!

```
ll solve(ll x) {
    ll res = 0;
    while (x) res += x, x >>= 1;
    return res;
} signed main() {
    int T = rr; while (T--) {
        ll n = rr, k = rr;
        if (n == k) printf("%lld\n", solve(n));
        else printf("%lld\n", solve(n) - solve(n - k));
    } return 0;
}
```

T2

求本质不同的子序列个数。

题目背景说了是动规，干嘛不想想是不是动规。

想不到，先打暴力。

```
using ll = long long;
using vi = vector<int>;
const ll MOD = 998244353;
int n; vi a;
set<pair<int, vi>> mem;
ll cnt = 0; void dfs(int k, vi now) {
    if (mem.count({k, now})) return;
    mem.insert({k, now});
    if (k == n) { cnt = (cnt + 1) % MOD; return; }
```

```
    dfs(k + 1, now); now.push_back(a[k]);
    dfs(k + 1, now);
} signed main() {
    n = rr; rep(i, n) a.push_back(rr);
    dfs(0, vi()); printf("%lld\n", cnt);
    return 0;
}
```

然后发现每一个数，要么加要么不加，优化（似乎并不快多少

```
using ll = long long;
using vi = vector<int>;
const ll MOD = 998244353;
int n; set<vi> res;
signed main() {
    n = rr; res.emplace(vi());
    rep(i, n) {
        int x = rr; auto bak = res;
        for (auto t : bak) t.push_back(x), res.emplace(t);
    } printf("%lld\n", ll(res.size()) % MOD);
    return 0;
}
```

然后不会优化了，老老实实找规律吧。

不久发现了规律。

设 $f(i)$ 表示以 i 结尾的本质不同的子序列的个数。

先不考虑「本质不同」的要求，从左到右考虑：$f(i)=\sum_{j=1}^{i} f(j)+$

第 i 个是否是该数字第一次出现.

然后考虑这个要求呢？发现如果这个数字不是第一次出现，那么它就不能从「前面与它相同的数字」的前面转移，因为那些东西已经被前面的那个数字计算过了。

特殊的，$f(1)=1$。然后再前缀和优化一下。

答案即是 $\sum_{i=1}^{n} f(i)$。于是，解决。

```
using ll = long long;
using vi = vector<int>;
const int N = 1e6 + 10;
const ll MOD = 998244353;
map<int, int> appear;
int isfirst[N], tolast[N];
ll f[N], s[N];
signed main() {
    int n = rr; for (int i = 1; i <= n; ++i) {
        int a = rr; appear.count(a) ? tolast[i] = appear[a] :
tolast[i] = isfirst[i] = 1;
        appear[a] = i;
    } f[1] = s[1] = 1; for (int i = 2; i <= n; ++i) {
        f[i] = (s[i - 1] - s[tolast[i] - 1] + isfirst[i] + MOD) % MOD;
        s[i] = (s[i - 1] + f[i]) % MOD;
    } printf("%lld\n", (s[n] + 1) % MOD);
    return 0;
}
```

T3

给定一个序列，多组询问，每组询问查询 $[l, r]$ 内极差为 k 的子区间数量。

思考过程：没有，不会，ST 表（打暴力）拿个部分分就完了。

```
const int N = 1e6 + 10;
const int K = 22;
int lg[N];
int f1[N][K], f2[N][K];
int q1(int l, int r) {
    int len = r - l + 1, k = lg[len];
    return max(f1[l][k], f1[r - (1 << k) + 1][k]);
} int q2(int l, int r) {
    int len = r - l + 1, k = lg[len];
    return min(f2[l][k], f2[r - (1 << k) + 1][k]);
} int main() {
    lg[1] = 0; for (int i = 2; i < N; ++i) lg[i] = lg[i >> 1] + 1;
    int n = rr, m = rr, k = rr; for (int i = 1; i <= n; ++i) f1[i][0]
= f2[i][0] = rr;
    for (int j = 1; j < K; ++j) for (int i = 1; i + (1 << j) - 1 <= n;
++i) {
        f1[i][j] = max(f1[i][j - 1], f1[i + (1 << j - 1)][j - 1]);
        f2[i][j] = min(f2[i][j - 1], f2[i + (1 << j - 1)][j - 1]);
    } while (m--) {
        int l = rr, r = rr; ll res = 0;
        for (int i = l; i <= r; ++i) for (int j = i; j <= r; ++j) if
(q1(i, j) - q2(i, j) == k) ++res;
        printf("%lld\n", res);
    } return 0;
```

```
}
```

T4

有 $2n$ 个二元组 (a_i, b_i)，现要两两分组，每一组对答案的贡献为，$a + b$ 较大（如果相等则是 b 较大）的那个二元组的 a.

思路：DFS 暴力偏分呗。

但是我懒，写了个全排列 `next_permutation` 就走人了，$10 \to 5\text{pts}$ 。

```
using ll = long long;
using pii = pair<int, int>;
using vi = vector<int>;
using vp = vector<pii>;
#define sum(x) (x.first + x.second)
int calc(pii a, pii b) {
    if (sum(a) != sum(b)) return sum(a) > sum(b) ? a.first : b.first;
    else return a.second > b.second ? a.first : b.first;
} signed main() {
    int n = rr, _1, _2; vp a; vi p;
    rep(_, 2 * n) _1 = rr, _2 = rr, a.push_back({_1, _2}),
p.push_back(_);
    ll ans = 4e18; do {
        ll res = 0;
        for (int l = 0; l + 1 < p.size(); l += 2) res += calc(a[p[l]],
a[p[l + 1]]);
        if (res < ans) ans = res;
    } while (next_permutation(p.begin(), p.end()));
```

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

T5

有一个排列，还有很多很多的操作，操作种类分为「« 循环右移一位 »、« 翻转 »、« 变为逆排列 »」。

一个排列 p 的逆排列定义为一个排列 p^{-1}，满足 $\forall \mathrm{i} \in \mathbb{N} \cap [1, \mathrm{n}],\ \mathrm{p}_{\mathrm{p}_\mathrm{i}^{-1}} = \mathrm{i}$。

逆排列？没看懂。先模拟一下：$\{4,3,1,2\} \to \{3,4,2,1\}$.

懂啦？没懂。再写个大的。然后才勉强理解（逃

然后，暴力（辣鸡的我。

```
using vi = vector<int>;
void turn(vi &a) {
    reverse(a.begin(), a.end());
} void move(vi &a) {
    int t = a.back();
    for (int i = a.size() - 1; i; --i) a[i] = a[i - 1];
    a[0] = t;
} void iv(vi &a) {
    int n = a.size(); vi t(n);
    for (int i = 0; i < n; ++i) t[a[i] - 1] = i + 1;
    a = t;
} signed main() {
    ios::sync_with_stdio(false), cin.tie(nullptr);
```

```
    int n, q; cin >> n >> q;
    vi a(n); rep(_, n) cin >> a[_];
    while (q--) {
        string s; int c; cin >> s >> c;
        while (c--) for (char op : s)
            if (op == 'S') move(a);
            else if (op == 'F') turn(a);
            else iv(a);
    } for (int _ : a) printf("%d ", _);
    return 0;
}
```

T6

一个有根树，以 1 为根，树上有点集 S，每次点集中的每个元素向上移动，直到根 1 属于这个点集。求每次移动以后的点集大小 $|\mathrm{S}|$。

不会，模拟完就走人。

```
const int N = 2e6 + 10;
vector<int> g[N]; void add(int u, int v) {
    g[u].push_back(v), g[v].push_back(u);
} int f[N]; void dfs(int u, int fa) {
    f[u] = fa; for (int v : g[u]) if (v != fa) dfs(v, u);
} signed main() {
    int n = rr; for (int i = 1; i < n; ++i) add(rr, rr);
    dfs(1, -1); int s = rr;
    set<int> d[2]; for (int i = 1; i <= s; ++i) d[0].emplace(rr);
    for (int k = 1; !d[k ^ 1].count(1); k ^= 1) {
        d[k].clear(); for (int i : d[k ^ 1]) d[k].emplace(f[i]);
        printf("%d ", (int)d[k].size());
```

```
    } return 0;
}
```

然后发现可以每次就把一个节点加到它的父亲上，但似乎一点用也没有。

```
const int N = 2e6 + 10;
vector<int> g[N]; void add(int u, int v) {
    g[u].push_back(v), g[v].push_back(u);
} int f[N]; void dfs(int u, int fa) {
    f[u] = fa; for (int v : g[u]) if (v != fa) dfs(v, u);
} signed main() {
    int n = rr; for (int i = 1; i < n; ++i) add(rr, rr);
    dfs(1, -1); int s = rr; set<int> d[2];
    for (int i = 1; i <= s; ++i) d[0].emplace(f[rr]);
    for (int k = 1; !d[k ^ 1].count(-1); k ^= 1) {
        printf("%d ", (int)d[k ^ 1].size()); d[k].clear();
        for (auto i : d[k ^ 1]) d[k].emplace(f[i]);
    } return 0;
}
```

T7

数论？不知道。看不懂而已（逃

T8

很懵，也没时间了，不写了。

「咱俩到 « 天台 » 逛逛吧」（大雾

T9

我为什么要看一个只有 1pts 的附加题?

总结

« 骗分过样例 »« 暴力出奇迹 »
« 暴搜挂着机 »« 打表出省一 »