

# A ICPC Regionals

---

用 *if* 判断一下即可,  $\lceil \frac{a}{b} \rceil = (a + b - 1) / b$  (整数意义下)

```
1 void solve() {
2     int n, k;
3     cin >> n >> k;
4     if (k <= (n + 9) / 10) cout << "Gold Medal" << "\n";
5     else if (k <= 3 * (n + 9) / 10) cout << "Silver Medal" << "\n";
6     else if (k <= 6 * (n + 9) / 10) cout << "Bronze Medal" << "\n";
7     else cout << "Da Tie" << "\n";
8 }
```

Fence 1

## B 小笨的卡牌游戏

---

要么取点数最大的那张，要么把所有牌丢掉取  $k$ 。

```
1  void solve() {
2      int n, mx = 0;
3      cin >> n;
4      for (int i = 1; i <= n; ++i) {
5          int x;
6          cin >> x;
7          mx = max(mx, x);
8      }
9      cout << max(mx, n) << "\n";
10 }
```

Fence 2

# C Palindrome Coloring

---

分别取出所有 0 和所有 1 即可处理所有情况，特判一下回文即可。

```
1 void solve() {
2     string s;
3     cin >> s;
4     char tar = s[0];
5     bool f = true;
6     for (int i = 0; i <= s.size() / 2; ++i) {
7         if (s[i] != s[s.size() - i - 1]) {
8             f = false;
9             break;
10        }
11    }
12    if (f) cout << 1 << "\n";
13    else cout << 2 << "\n";
14 }
```

Fence 3

## D 小笨的数字配对

从最大到最小逐一贪心，如果不影响先前位的情况下，这一位为 1 的数量大于等于  $n$ ，这一位就可以置 1。

```
1 void solve() {
2     int n;
3     cin >> n;
4     vector<int> a(2 * n + 1);
5     for (int i = 1; i <= 2 * n; ++i) cin >> a[i];
6     int ans = 0;
7     for (int mask = 31; mask >= 0; --mask) {
8         int cnt = 0, cand = ans | (1 << mask);
9         for (int i = 1; i <= 2 * n; ++i) {
10             if ((cand & a[i]) == cand) cnt++;
11         }
12         if (cnt >= n) ans = cand;
13     }
14     cout << ans << "\n";
15 }
```

Fence 4

一个更加便于理解的版本

```
1 void solve() {
2     int n;
3     cin >> n;
4     vector<int> a(2 * n + 1);
5     for (int i = 1; i <= 2 * n; ++i) cin >> a[i];
6
7     int ans = 0;
8     for (int mask = 31; mask >= 0; --mask) {
9         vector<int> b = a;
10        a.clear();
11        ll cand = ans | (1 << mask);
12        for (auto v : b) {
13            if ((cand & v) == cand) a.pb(v);
14        }
15        if (a.size() >= n) ans = cand;
16        else a = b;
17    }
18    cout << ans << "\n";
19 }
```

Fence 5

# E 小笨的美丽序列

一道正难则反的题目。

显然答案为 非空序列数 -  $gcd$  在内的序列数，我们只需要枚举  $gcd$ ，它的所有倍数即可构成所有非美丽序列。

```
1 void solve() {
2     int n;
3     cin >> n;
4     vector<int> cnt(maxn + 10);
5     for (int i = 1; i <= n; ++i) {
6         int x;
7         cin >> x;
8         cnt[x]++;
9     }
10
11     long long ans = power(2, n) - 1;
12     for (int i = 1; i <= maxn; ++i) {
13         int len = 0, st;
14         if (cnt[i]) {
15             len += cnt[i];
16             st = cnt[i];
17             for (int j = i + i; j <= maxn; j += i) {
18                 len += cnt[j];
19             }
20             for (int j = 1; j <= st; ++j) {
21                 ans -= power(2, len - j);
22             }
23         }
24     }
25     cout << ans << "\n";
26 }
```

Fence 6

当然也可以反做

对每个  $d$  计数：记  $F[d]$  为“由数组中所有能被  $d$  整除的元素构成的、且  $gcd$  恰好为  $d$  的非空子序列数”。

我们可以用常见的“枚举倍数，从大到小消去”的方法得到所有  $F[d]$ ：

$$G[d] = 2^{m[d]} - 1, F[d] = G[d] - \sum_{k \geq 2} F[d \cdot k]$$

其中  $m[d]$  是数组中能被  $d$  整除的元素个数， $G[d]$  是这些元素的所有非空子序列数。

但我们真正需要的是 不包含值为  $d$  的那些子序列（即子序列中不出现  $d$ ），设为  $F'[d]$ 。

注意：在所有只选取“能被  $d$  整除”的子序列中，恰包含至少一个等于  $d$  的子序列数为

$$S[d] = (2^{cnt[d]} - 1) \cdot 2^{m[d] - cnt[d]} = 2^{m[d]} - 2^{m[d] - cnt[d]}$$

其中  $cnt[d]$  是数组中等于  $d$  的个数；因为选至少一个等于  $d$  的方式有  $2^{cnt[d]} - 1$  种，其他位置任意选或不选，因此：

$$F'[d] = F[d] - S[d]$$

最终答案就是对所有  $d$  ( $1 \leq d \leq n$ ) 将  $F[d]$  累加并对 998244353 取模。

```
1 void solve() {
2     int n;
3     cin >> n;
4     vector<int> cnt(maxn + 10);
5     for (int i = 1; i <= n; ++i) {
6         int x;
7         cin >> x;
8         cnt[x]++;
9     }
10
11     vector<int> m(n + 1, 0);
12     for (int d = 1; d <= n; ++d) {
13         for (int j = d; j <= n; j += d) m[d] += cnt[j];
14     }
15
16     vector<Z> p2(n + 1, 1);
17     for (int i = 1; i <= n; ++i) p2[i] = p2[i - 1] + p2[i - 1];
18
19     vector<Z> G(n + 1, 0);
20     for (int d = 1; d <= n; ++d) {
21         if (m[d] == 0) G[d] = 0;
22         else G[d] = p2[m[d]] - 1;
23     }
24
25     vector<Z> f(n + 1, 0);
26     for (int d = n; d >= 1; --d) {
27         Z val = G[d];
28         for (int dk = d * 2; dk <= n; dk += d) {
29             val -= f[dk];
30         }
31         f[d] = val;
32     }
33
34     vector<Z> S(n + 1, 0);
35     for (int d = 1; d <= n; ++d) {
36         if (m[d] == 0) {
37             S[d] = 0;
38             continue;
39         }
40         S[d] = p2[m[d]] - p2[m[d] - cnt[d]];
41     }
42
43     Z ans = 0;
44     for (int d = 1; d <= n; ++d) {
45         Z fp = f[d] - S[d];
46         ans += fp;
47     }
48     cout << ans << "\n";
49 }
```

## F 小笨的括号计数

如果记 '(' 为  $+1$ , ')' 为  $-1$ , 如果一个括号序列可以连接上去, 那么对于每一位, 都有前缀和  $pre[i] \geq 0$ 。

记这个括号序列等价表示的最小值为  $mn[i]$ , 那么如果

$$pre[\text{当前的选择情况}] + mn[\text{当前括号序列}] \geq 0$$

那么

$$dp[\text{选了这个括号序列的选择情况}] += dp[\text{当前的选择情况}]$$

最终答案即为  $dp[\text{选择所有序列}]$ , 需要特判不合法的情况。

```
1 void solve() {
2     int n;
3     cin >> n;
4     vector<int> a(n), b(n);
5     for (int i = 0; i < n; ++i) {
6         string s;
7         cin >> s;
8         int mn = 0;
9         for (auto v : s) {
10             if (v == '(') a[i]++;
11             else a[i]--;
12             mn = min(mn, a[i]);
13         }
14         b[i] = mn;
15     }
16     int m = 1 << n;
17     vector<int> pre(m);
18     for (int i = 1; i < m; ++i) {
19         int prev = __builtin_ctz(i);
20         pre[i] = pre[i ^ (1 << prev)] + a[prev];
21     }
22
23
24     if (pre[m - 1] != 0) {
25         cout << 0 << "\n";
26         return;
27     }
28
29     vector<Z> dp(m);
30     dp[0] = 1;
31     for (int i = 0; i < m; ++i) {
32         if (dp[i] == 0) continue;
33         int s = pre[i];
34         for (int j = 0; j < n; ++j) {
35             if ((i >> j) & 1) continue;
36             if (s + b[j] >= 0) {
37                 int nxt = i | (1 << j);
38                 dp[nxt] += dp[i];
39             }
40         }
41     }
```

```
41     }  
42     cout << dp[m - 1] << "\n";  
43 }
```

Fence 8



# 头文件

```
1 //Another
2 #include<bits/stdc++.h>
3 #include<bits/extc++.h>
4 #define pb push_back
5 #define eb emplace_back
6 #define fi first
7 #define se second
8 #define all(a) a.begin(), a.end()
9 #define rall(a) a.rbegin(), a.rend()
10 using namespace std;
11
12 typedef long long ll;
13 typedef long double ld;
14 typedef unsigned long long ull;
15 typedef __int128 i128;
16 typedef pair<int, int> PII;
17 typedef pair<ll, ll> PLL;
18 typedef tuple<ll, ll, ll> TLLL;
19 typedef __gnu_pbds::tree<PLL, __gnu_pbds::null_type, less<PLL>,
    __gnu_pbds::rb_tree_tag, __gnu_pbds::tree_order_statistics_node_update>
    Tree;
20 // typedef __gnu_pbds::tree<ll, __gnu_pbds::null_type, less<ll>,
    __gnu_pbds::rb_tree_tag, __gnu_pbds::tree_order_statistics_node_update>
    Tree;
21
22 constexpr int inf = (ll)1e9 + 7;
23 constexpr ll INF = (ll)2e18 + 9;
24 // constexpr ll INF = (ll)4e18;
25 // constexpr ll MOD = 1e9 + 7;
26 constexpr ll MOD = 998244353;
27 constexpr ld PI = acos(-1.0);
28 constexpr ld eps = 1e-10;
29
30 mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
31 ull randint(ull l, ull r) {uniform_int_distribution<unsigned long long>
    dist(l, r); return dist(rng);}
32
33 void init() {
34
35 }
36
37 void solve() {
38
39 }
40
41 int main() {
42     ios::sync_with_stdio(0);
43     cin.tie(0); cout.tie(0);
44
45     init();
46
47     int t = 1;
```

```
48     cin >> t;
49     for (int _ = 1; _ <= t; ++_) {
50         solve();
51     }
52     return 0;
53 }
```

Fence 9

# 自动取模

```
1  template<class T>
2  constexpr T power(T a, ll b) {
3      T res = 1;
4      while (b) {
5          if (b & 1) res *= a;
6          b >>= 1;
7          a *= a;
8      }
9      return res;
10 }
11
12 template<int P>
13 struct MInt {
14     int x;
15     constexpr MInt(): x() {}
16     constexpr MInt(ll x) : x{norm(x % getMod())} {}
17     static int Mod;
18     constexpr static int getMod() {
19         if (P > 0) {
20             return P;
21         } else {
22             return Mod;
23         }
24     }
25     constexpr static void setMod(int Mod_) {
26         Mod = Mod_;
27     }
28     constexpr int norm(int x) const {
29         if (x < 0) {
30             x += getMod();
31         }
32         if (x >= getMod()) {
33             x -= getMod();
34         }
35         return x;
36     }
37     constexpr int val() const {
38         return x;
39     }
40     explicit constexpr operator int() const {
41         return x;
42     }
43     constexpr MInt operator-() const {
44         MInt res;
45         res.x = norm(getMod() - x);
46         return res;
47     }
48     constexpr MInt inv() const {
49         assert(x != 0);
50         return power(*this, getMod() - 2);
51     }
52     constexpr MInt &operator*=(MInt rhs) & {
```

```

53         x = 1LL * x * rhs.x % getMod();
54         return *this;
55     }
56     constexpr MInt &operator+=(MInt rhs) & {
57         x = norm(x + rhs.x);
58         return *this;
59     }
60     constexpr MInt &operator-=(MInt rhs) & {
61         x = norm(x - rhs.x);
62         return *this;
63     }
64     constexpr MInt &operator/=(MInt rhs) & {
65         return *this *= rhs.inv();
66     }
67     friend constexpr MInt operator*(MInt lhs, MInt rhs) {
68         MInt res = lhs;
69         res *= rhs;
70         return res;
71     }
72     friend constexpr MInt operator+(MInt lhs, MInt rhs) {
73         MInt res = lhs;
74         res += rhs;
75         return res;
76     }
77     friend constexpr MInt operator-(MInt lhs, MInt rhs) {
78         MInt res = lhs;
79         res -= rhs;
80         return res;
81     }
82     friend constexpr MInt operator/(MInt lhs, MInt rhs) {
83         MInt res = lhs;
84         res /= rhs;
85         return res;
86     }
87     friend constexpr istream &operator>>(istream &is, MInt &a) {
88         ll v;
89         is >> v;
90         a = MInt(v);
91         return is;
92     }
93     friend constexpr ostream &operator<<(ostream &os, const MInt &a) {
94         return os << a.val();
95     }
96     friend constexpr bool operator==(MInt lhs, MInt rhs) {
97         return lhs.val() == rhs.val();
98     }
99     friend constexpr bool operator!=(MInt lhs, MInt rhs) {
100         return lhs.val() != rhs.val();
101     }
102 };
103
104 template<>
105 int MInt<0>::Mod = 998244353;
106
107 using Z = MInt<MOD>;

```

