**Олимпиада «Ломоносов» по информатике**
**2023-2024 учебный год. Заключительный тур**
**Работа участника с id заявки 1261660, логином inf24f_236**

**Сводный итог по всем задачам в проверяющей системе**

| Run ID | Time | User name | Problem | Language | Result | Tests | Score |
|---|---|---|---|---|---|---|---|
| 57 | 0:28:34 | inf24f_236 | 1 | g++ | OK | 28 | 100 |
| 185 | 1:11:02 | inf24f_236 | 2 | g++ | OK | 28 | 100 |
| 581 | 3:42:10 | inf24f_236 | 3 | python3 | OK | 28 | 100 |
| 358 | 2:23:00 | inf24f_236 | 4 | g++ | OK | 21 | 100 |
| 421 | 2:48:34 | inf24f_236 | 5 | g++ | OK | 22 | 100 |
| 692 | 3:59:22 | inf24f_236 | 6 | g++ | Partial solution | 0 | 0 |

500 технических баллов
83 итоговых балла

## Посылка по задаче 1

```
[1]  #include <bits/stdc++.h>
[2]  #include <ext/rope>
[3]  #include <ext/pb_ds/assoc_container.hpp>
[4]  #include <ext/pb_ds/tree_policy.hpp>
[5]
[6]
[7]  using ll = long long;
[8]  using iii = __int128_t;
[9]  using namespace std;
[10]
[11]
[12] using namespace __gnu_cxx;
[13] using namespace __gnu_pbds;
[14] template <typename K, typename Cmp = less<K>, typename V = null_type>
[15] using ordered_map = tree<K, V, Cmp, rb_tree_tag, tree_order_statistics_node_update>;
[16]
[17]
[18] #define citers(c) (c).begin(), (c).end()
[19]
[20]
[21] // #define __INTERACTIVE_TASK__
[22] #if defined(__LOCAL__)
[23] #       define local(arg) arg
[24] #       define prod(arg)
[25] #else
[26] #       define local(arg)
[27] #       define prod(arg) arg
[28] #endif
[29]
[30] #if !(defined(__LOCAL__) || defined(__INTERACTIVE_TASK__))
[31] #       define endl '\n'
[32] #endif
[33]
[34]
[35] static inline void __io_speedup [[maybe_unused]] ()
[36] {
[37]         ios_base::sync_with_stdio(false);
[38]         cin.tie(nullptr);
[39]         cout.tie(nullptr);
[40] }
[41]
[42]
[43] const ll TSZ = 42;
[44] ll T[TSZ] = {0};
[45] // vector<ll> nums;
[46]
[47]
[48] // void p(int idx, int num, int cnt1 = 0)
[49] // {
[50] //      if (idx == 3)
[51] //      {
[52] //              nums.push_back(num);
[53] //              return;
[54] //      }
[55] //      p(idx - 1, num, 0);
[56] //      if (cnt1 < 3) p(idx - 1, num + T[idx], cnt1 + 1);
[57] // }
[58]
[59]
```

```
[60] signed main()
[61] {
[62]         __io_speedup();
[63]
[64]         T[3] = 1;
[65]         for (ll i = 4; i < TSZ; ++i) T[i] = T[i - 1] + T[i - 2] + T[i - 3] + T[i - 4];
[66]
[67]         // for (ll i = 0; i < TSZ; ++i) cout << T[i] << ' ';
[68]         // cout << endl;
[69]         // p(28, 0);
[70]         // sort(citers(nums));
[71]         // auto __it = unique(citers(nums));
[72]         // nums.erase(__it, nums.end());
[73]         // ll mex = 0;
[74]         // for (ll i : nums)
[75]         // {
[76]         //      if (i == mex)
[77]         //              ++mex;
[78]         //      else
[79]         //              break;
[80]         // }
[81]         // // for (ll i : nums) cout << i << ' ';
[82]         // cout << nums.back() << endl;
[83]         // cout << nums.size() << endl;
[84]         // cout << mex << endl;
[85]         // cout << T[41] << endl;
[86]         ll ans = 0;
[87]         ll n;
[88]         cin >> n;
[89]         while (n--)
[90]         {
[91]                 ll num;
[92]                 cin >> num;
[93]                 ll one_cnt = 0;
[94]                 for (ll i = TSZ - 1; i > 3; --i)
[95]                 {
[96]                         if (num >= T[i])
[97]                         {
[98]                                 num -= T[i];
[99]                                 ++one_cnt;
[100]                        }
[101]                }
[102]                ans += ((one_cnt % 2) == 0);
[103]        }
[104]        cout << ans << endl;
[105]        return 0;
[106] }
```

## Посылка по задаче 2

```
[1]  #include <bits/stdc++.h>
[2]  #include <ext/rope>
[3]  #include <ext/pb_ds/assoc_container.hpp>
[4]  #include <ext/pb_ds/tree_policy.hpp>
[5]
[6]
[7]  using ll = long long;
[8]  using iii = __int128_t;
[9]  using namespace std;
[10]
[11]
[12] using namespace __gnu_cxx;
[13] using namespace __gnu_pbds;
[14] template <typename K, typename Cmp = less<K>, typename V = null_type>
[15] using ordered_map = tree<K, V, Cmp, rb_tree_tag, tree_order_statistics_node_update>;
[16]
[17]
[18] #define citers(c) (c).begin(), (c).end()
[19]
[20]
[21] // #define __INTERACTIVE_TASK__
[22] #if defined(__LOCAL__)
[23] #       define local(arg) arg
[24] #       define prod(arg)
[25] #else
[26] #       define local(arg)
[27] #       define prod(arg) arg
[28] #endif
[29]
[30] #if !(defined(__LOCAL__) || defined(__INTERACTIVE_TASK__))
[31] #       define endl '\n'
[32] #endif
[33]
[34]
[35] static inline void __io_speedup [[maybe_unused]] ()
[36] {
[37]         ios_base::sync_with_stdio(false);
[38]         cin.tie(nullptr);
[39]         cout.tie(nullptr);
[40] }
[41]
[42]
[43] int color2idx[256];
[44] const int END_DEPTH = 5000;
[45] const int COLORS_CNT = 9;
[46] char idx2color[COLORS_CNT];
[47] string num2bin[] = {
[48]         "000",
[49]         "001",
[50]         "010",
[51]         "011",
[52]         "100",
[53]         "101",
[54]         "110",
[55]         "111"
[56] };
[57] int cnt[COLORS_CNT][END_DEPTH] = {{0}};
[58] int ans[END_DEPTH] = {0};
[59]
[60]
[61] bool is_r_gt(int * l, int * r)
[62] {
[63]         for (int i = 0; i < END_DEPTH; ++i)
[64]         {
[65]                 if (r[i] > l[i]) return true;
[66]                 if (r[i] < l[i]) return false;
[67]         }
[68]         return false;
[69] }
[70]
[71]
[72] void norm(int * a)
[73] {
[74]         for (int i = END_DEPTH - 1; i > 0; --i)
[75]         {
[76]                 if (a[i] >= 8)
[77]                 {
[78]                         a[i - 1] += a[i] / 8;
[79]                         a[i] %= 8;
[80]                 }
[81]         }
[82] }
```

```
[83]
[84]
[85] int process(const string & s, int i, int depth)
[86] {
[87]         for (int processed_cnt = 0; processed_cnt < 8; ++processed_cnt)
[88]         {
[89]                 if (s[i] == 'Q')
[90]                 {
[91]                         i = process(s, i + 1, depth + 1);
[92]                 }
[93]                 else
[94]                 {
[95]                         cnt[color2idx[(int)s[i]]][depth] += 1;
[96]                         ++i;
[97]                 }
[98]         }
[99]         return i;
[100] }
[101]
[102]
[103] signed main()
[104] {
[105]         __io_speedup();
[106]
[107]         color2idx['B'] = 0;
[108]         color2idx['C'] = 1;
[109]         color2idx['D'] = 2;
[110]         color2idx['G'] = 3;
[111]         color2idx['O'] = 4;
[112]         color2idx['R'] = 5;
[113]         color2idx['V'] = 6;
[114]         color2idx['W'] = 7;
[115]         color2idx['Y'] = 8;
[116]
[117]         idx2color[0] = 'B';
[118]         idx2color[1] = 'C';
[119]         idx2color[2] = 'D';
[120]         idx2color[3] = 'G';
[121]         idx2color[4] = 'O';
[122]         idx2color[5] = 'R';
[123]         idx2color[6] = 'V';
[124]         idx2color[7] = 'W';
[125]         idx2color[8] = 'Y';
[126]
[127]         string s;
[128]         cin >> s;
[129]         if (s.size() == 1)
[130]         {
[131]                 cout << s << endl;
[132]                 cout << "1.0" << endl;
[133]                 return 0;
[134]         }
[135]
[136]         process(s, 1, 1);
[137]         int ans_color = 0;
[138]         for (int i = 0; i < COLORS_CNT; ++i)
[139]         {
[140]                 norm(cnt[i]);
[141]                 if (is_r_gt(ans, cnt[i]))
[142]                 {
[143]                         memcpy(ans, cnt[i], sizeof(int) * END_DEPTH);
[144]                         ans_color = i;
[145]                 }
[146]         }
[147]
[148]         cout << idx2color[ans_color] << endl;
[149]         string ans_str = "0.";
[150]         ans_str.reserve(END_DEPTH * 3 + 10);
[151]         ans_str[0] = ans[0] + '0';
[152]         for (int i = 1; i < END_DEPTH; ++i) ans_str.append(num2bin[ans[i]]);
[153]         int last1idx = -1;
[154]         for (int i = (int)ans_str.size() - 1; i > 0; --i)
[155]         {
[156]                 if (ans_str[i] == '1')
[157]                 {
[158]                         last1idx = i;
[159]                         break;
[160]                 }
[161]         }
[162]         for (int i = 0; i <= last1idx; ++i) cout << ans_str[i];
[163]         cout << endl;
[164]         return 0;
[165] }
```

## Посылка по задаче 3

```
dig2elf = [
    "i",
        "i(",
        "i((",
        "I",
        "I(",
        "I((",
        "J",
        "J)",
        "J))",
        "j",
        "j)",
        "j))"
]

elfi2bnum = {
    'i': 1,
    'I': 4,
    'J': 7,
    'j': 10
}


def read_elf_digit(s, i):
    n = len(s)
    if i + 1 < n and (s[i + 1] == '(' or s[i + 1] == ')'):
        if i + 2 < n and (s[i + 2] == '(' or s[i + 2] == ')'):
            return (elfi2bnum[s[i]] + 2, i + 3)
        return (elfi2bnum[s[i]] + 1, i + 2)
    else:
        return (elfi2bnum[s[i]], i + 1)


def elf2dec(s):
    p = 1
    res = 0
    i = 0
    n = len(s)
    if n == 2 and s == "[]":
        return 0

    while i < n:
        digit, i = read_elf_digit(s, i)
        res += digit * p
        p *= 12
    return res


def dec2elf(n):
    res = []
    while n != 0:
        rem = n % 12
        if rem == 0:
            res.append(dig2elf[11])
            n -= 12
        else:
            res.append(dig2elf[rem - 1])
            n -= rem
        n //= 12
    return res


N = int(input())
chests = [elf2dec(input()) for _ in range(N)]

k, l = -1, -1
diff = -1
for i in range(N):
    for j in range(i + 1, N):
        cur_diff = abs(chests[i] - chests[j])
        if cur_diff > diff:
            diff = cur_diff
            k, l = i, j
        elif cur_diff == diff and (i + j) > (k + l):
            k, l = i, j

print("".join(dec2elf(k + 1)))
print("".join(dec2elf(l + 1)))
```

## Посылка по задаче 4

```
[1]  #include <bits/stdc++.h>
[2]  #include <ext/rope>
[3]  #include <ext/pb_ds/assoc_container.hpp>
[4]  #include <ext/pb_ds/tree_policy.hpp>
[5]
[6]
[7]  using ll = long long;
[8]  using iii = __int128_t;
[9]  using namespace std;
[10]
[11]
[12] using namespace __gnu_cxx;
[13] using namespace __gnu_pbds;
[14] template <typename K, typename Cmp = less<K>, typename V = null_type>
[15] using ordered_map = tree<K, V, Cmp, rb_tree_tag, tree_order_statistics_node_update>;
[16]
[17]
[18] #define citers(c) (c).begin(), (c).end()
[19]
[20]
[21] // #define __INTERACTIVE_TASK__
[22] #if defined(__LOCAL__)
[23] #       define local(arg) arg
[24] #       define prod(arg)
[25] #else
[26] #       define local(arg)
[27] #       define prod(arg) arg
[28] #endif
[29]
[30] #if !(defined(__LOCAL__) || defined(__INTERACTIVE_TASK__))
[31] #       define endl '\n'
[32] #endif
[33]
[34]
[35] static inline void __io_speedup [[maybe_unused]] ()
[36] {
[37]         ios_base::sync_with_stdio(false);
[38]         cin.tie(nullptr);
[39]         cout.tie(nullptr);
[40] }
[41]
[42]
[43] const int MAX_H = 20;
[44] const int MAX_W = 200;
[45] int f[MAX_W][MAX_H];
[46] int dp_ga_ro_ge[MAX_W][MAX_H][MAX_H][MAX_H];
[47]
[48]
[49] signed main()
[50] {
[51]         __io_speedup();
[52]
[53]         int h, w;
[54]         cin >> h >> w;
[55]         int init_ga, init_ro, init_ge;
[56]         cin >> init_ga >> init_ro >> init_ge;
[57]         for (int j = 0; j < h; ++j)
[58]         {
[59]                 for (int i = 0; i < w; ++i) cin >> f[i][j];
[60]         }
```

```
[61]         auto get_val = [&](int i, int ga, int ro, int ge) -> int
[62]         {
[63]                 if (ga == ro && ro == ge) return f[i][ga];
[64]                 if (ga == ro) return f[i][ga] + f[i][ge];
[65]                 if (ga == ge) return f[i][ga] + f[i][ro];
[66]                 if (ro == ge) return f[i][ga] + f[i][ro];
[67]                 return f[i][ga] + f[i][ro] + f[i][ge];
[68]         };
[69]         for (int ga = 0; ga < h; ++ga)
[70]         {
[71]                 for (int ro = 0; ro < h; ++ro)
[72]                 {
[73]                         for (int ge = 0; ge < h; ++ge) dp_ga_ro_ge[0][ga][ro][ge] = -1;
[74]                 }
[75]         }
[76]         dp_ga_ro_ge[0][init_ga][init_ro][init_ge] = get_val(0, init_ga, init_ro, init_ge);
[77]         for (int i = 1; i < w; ++i)
[78]         {
[79]                 for (int ga = 0; ga < h; ++ga)
[80]                 {
[81]                         for (int ro = 0; ro < h; ++ro)
[82]                         {
[83]                                 for (int ge = 0; ge < h; ++ge)
[84]                                 {
[85]                                         dp_ga_ro_ge[i][ga][ro][ge] = -1;
[86]                                         for (int prev_ga = max(ga -
[87]  1, 0); prev_ga < min(ga + 2, h); ++prev_ga)
[88]                                         {
[89]                                                 for (int prev_ro = max(ro -
[90]  1, 0); prev_ro < min(ro + 2, h); ++prev_ro)
[91]                                                 {
[92]                                                         for (int prev_ge = max(ge -
[93]  1, 0); prev_ge < min(ge + 2, h); ++prev_ge)
[94]                                                         {
[95]                                                                 if (dp_ga_ro_ge[i - 1][prev_ga]
[96] [prev_ro][prev_ge] == -1) continue;
[97]                                                                 dp_ga_ro_ge[i][ga][ro][ge] = max(
[98]                                                                         dp_ga_ro_ge[i][ga][ro][ge],
[99]                                                                         dp_ga_ro_ge[i - 1][prev_ga]
[100][prev_ro][prev_ge] + get_val(i, ga, ro, ge)
[101]                                                                 );
[102]                                                         }
[103]                                                 }
[104]                                         }
[105]                                 }
[106]                         }
[107]                 }
[108]         }
[109]         int ans = 0;
[110]         for (int ga = 0; ga < h; ++ga)
[111]         {
[112]                 for (int ro = 0; ro < h; ++ro)
[113]                 {
[114]                         for (int ge = 0; ge < h; ++ge) ans = max(ans, dp_ga_ro_ge[w - 1][ga][ro]
     [ge]);
                 }
         }
         cout << ans << endl;
         return 0;
     }
```

## Посылка по задаче 5

```
[1]  #include <bits/stdc++.h>
[2]  #include <ext/rope>
[3]  #include <ext/pb_ds/assoc_container.hpp>
[4]  #include <ext/pb_ds/tree_policy.hpp>
[5]
[6]
[7]  using ll = long long;
[8]  using iii = __int128_t;
[9]  using namespace std;
[10]
[11]
[12] using namespace __gnu_cxx;
[13] using namespace __gnu_pbds;
[14] template <typename K, typename Cmp = less<K>, typename V = null_type>
[15] using ordered_map = tree<K, V, Cmp, rb_tree_tag, tree_order_statistics_node_update>;
[16]
[17]
[18] #define citers(c) (c).begin(), (c).end()
[19]
[20]
[21] // #define __INTERACTIVE_TASK__
[22] #if defined(__LOCAL__)
[23] #       define local(arg) arg
[24] #       define prod(arg)
[25] #else
[26] #       define local(arg)
[27] #       define prod(arg) arg
[28] #endif
[29]
[30] #if !(defined(__LOCAL__) || defined(__INTERACTIVE_TASK__))
[31] #       define endl '\n'
[32] #endif
[33]
[34]
[35] static inline void __io_speedup [[maybe_unused]] ()
[36] {
[37]         ios_base::sync_with_stdio(false);
[38]         cin.tie(nullptr);
[39]         cout.tie(nullptr);
[40] }
[41]
[42]
[43] // template <ll M>
[44] // struct ModLL
[45] // {
[46] //      ll val;
[47]
[48] //      constexpr ModLL() {}
[49] //      constexpr ModLL(int val) : val(((val % M) + M) % M) {}
[50]
[51] //      constexpr ModLL<M> operator+(ModLL<M> o) { return val + o.val; }
[52] //      constexpr ModLL<M> operator-(ModLL<M> o) { return val - o.val; }
[53] //      constexpr ModLL<M> operator*(ModLL<M> o) { return val * o.val; }
[54] // };
[55]
[56]
[57] // template <typename T> inline T ord(char c)
[58] // {
[59] //      return c - 33;
[60] // }
[61]
[62] // template <typename T>
[63] // struct RevHash
[64] // {
[65] //      vector<T> kpow;
[66] //      vector<T> rhash;
[67] //      T K;
[68]
[69] //      RevHash(const string & s, ll n, T K) : kpow(n), rhash(n), K(K)
[70] //      {
[71] //              kpow[0] = 1;
[72] //              for (ll i = 1; i < n; ++i) kpow[i] = kpow[i - 1] * K;
[73]
[74] //              rhash[0] = ord<T>[s[0]];
[75] //              for (ll i = 1; i < n; ++i) rhash[i] = rhash[i - 1] * K + ord<T>(s[i]);
[76] //      }
[77]
[78] //      inline T get_full_hash() { return rhash.back(); }
[79]
[80] //      inline T substr(ll l, ll r)
[81] //      {
[82] //              if (l == 0) return rhash[r];
[83] //              return rhash[r] - rhash[l - 1];
[84] //      }
[85]
[86] //      inline T substrl(ll i, ll len) { return substr(i, i + len - 1); }
[87] // };
[88]
```

```
[89]
[90] // using MLL1 = ModLL<((ll)1e9) + 7>;
[91] // using MLL2 = ModLL<999999937>;
[92] // const ll first_k = 131;
[93] // const ll second_k = 151;
[94]
[95]
[96] // bool is_prime(ll n)
[97] // {
[98] //      ll __end = sqrt(n) + 10;
[99] //      for (ll i = 2; i < __end; ++i)
[100]//      {
[101]//              if (n % i == 0) return false;
[102]//      }
[103]//      return true;
[104]// }
[105]
[106]
[107]const int MIN_CODE [[maybe_unused]] = 33;
[108]const int MAX_CODE [[maybe_unused]] = 126;
[109]const int SZ = 128;
[110]int needed_cnt[SZ];
[111]int window_cnt[SZ];
[112]string orig;
[113]string needed;
[114]
[115]
[116]int can(int len)
[117]{
[118]        memset(window_cnt, 0, sizeof(int) * SZ);
[119]        for (int i = 0; i < len; ++i) ++(window_cnt[(int)orig[i]]);
[120]        int satisfy_cnt = 0;
[121]        for (int i = 0; i < SZ; ++i)
[122]        {
[123]                if (window_cnt[i] >= needed_cnt[i]) ++satisfy_cnt;
[124]        }
[125]        if (satisfy_cnt == SZ) return 0;
[126]        for (int i = len; i < (int)orig.size(); ++i)
[127]        {
[128]                int del = orig[i - len];
[129]                int add = orig[i];
[130]
[131]                bool old_satis = window_cnt[del] >= needed_cnt[del];
[132]                --(window_cnt[del]);
[133]                bool cur_satis = window_cnt[del] >= needed_cnt[del];
[134]                if (old_satis && (!cur_satis)) --satisfy_cnt;
[135]
[136]                old_satis = window_cnt[add] >= needed_cnt[add];
[137]                ++(window_cnt[add]);
[138]                cur_satis = window_cnt[add] >= needed_cnt[add];
[139]                if ((!old_satis) && cur_satis) ++satisfy_cnt;
[140]
[141]                if (satisfy_cnt == SZ) return i - len + 1;
[142]        }
[143]        return -1;
[144]}
[145]
[146]
[147]signed main()
[148]{
[149]        __io_speedup();
[150]
[151]        cin >> orig;
[152]        cin >> needed;
[153]        for (char c : needed) ++(needed_cnt[(int)c]);
[154]
[155]        int l = 0, r = orig.size() + 1;
[156]        int ans_begin = -1;
[157]        while (l + 1 < r)
[158]        {
[159]                int mid = (l + r) / 2;
[160]                int cur_ans;
[161]                if ((cur_ans = can(mid)) != -1)
[162]                {
[163]                        r = mid;
[164]                        ans_begin = cur_ans;
[165]                }
[166]                else { l = mid; }
[167]        }
[168]        if (ans_begin != -1)
[169]        {
[170]                // cout << orig.substr(ans_begin, r);
[171]                for (int i = ans_begin; i < ans_begin + r; ++i) cout << orig[i];
[172]        }
[173]        cout << endl;
[174]        return 0;
[175]}
```

### Посылка по задаче 6

```
[1]  #include <bits/stdc++.h>
[2]  #include <ext/rope>
[3]  #include <ext/pb_ds/assoc_container.hpp>
[4]  #include <ext/pb_ds/tree_policy.hpp>
[5]
[6]
[7]  using ll = long long;
[8]  using iii = __int128_t;
[9]  using namespace std;
[10]
[11]
[12] using namespace __gnu_cxx;
[13] using namespace __gnu_pbds;
[14] template <typename K, typename Cmp = less<K>, typename V = null_type>
[15] using ordered_map = tree<K, V, Cmp, rb_tree_tag, tree_order_statistics_node_update>;
[16]
[17]
[18] #define citers(c) (c).begin(), (c).end()
[19]
[20]
[21] // #define __INTERACTIVE_TASK__
[22] #if defined(__LOCAL__)
[23] #       define local(arg) arg
[24] #       define prod(arg)
[25] #else
[26] #       define local(arg)
[27] #       define prod(arg) arg
[28] #endif
[29]
[30] #if !(defined(__LOCAL__) || defined(__INTERACTIVE_TASK__))
[31] #       define endl '\n'
[32] #endif
[33]
[34]
[35] static inline void __io_speedup [[maybe_unused]] ()
[36] {
[37]         ios_base::sync_with_stdio(false);
[38]         cin.tie(nullptr);
[39]         cout.tie(nullptr);
[40] }
[41]
[42]
[43] pair<string, bool> get_field()
[44] {
[45]         string res;
[46]         int c;
[47]         while ((c = cin.get(), c != ';' && c != '\n' && c != EOF))
[48]         {
[49]                 if (c == '\'' || c == '"') continue;
[50]                 res.push_back(c);
[51]         }
[52]         return make_pair(res, c == '\n' || c == EOF);
[53] }
[54]
[55]
```

```
[56] signed main()
[57] {
[58]         __io_speedup();
[59]
[60]         map<string, vector<vector<bool>>> task2test2bitset;
[61]         map<string, int> task2submissions_cnt;
[62]         while (true)
[63]         {
[64]                 string name = get_field().first;
[65]                 if (name.size() == 0) break;
[66]
[67]                 auto it = task2test2bitset.find(name);
[68]                 if (it != task2test2bitset.end())
[69]                 {
[70]                         string token;
[71]                         bool is_ended = false;
[72]                         int sub_idx = task2submissions_cnt[name];
[73]                         for (int i = 0; !is_ended; ++i)
[74]                         {
[75]                                 tie(token, is_ended) = get_field();
[76]                                 it->second[i][sub_idx] = token == "OK";
[77]                         }
[78]                 }
[79]                 else
[80]                 {
[81]                         auto ref = task2test2bitset[name];
[82]                         string token;
[83]                         bool is_ended = false;
[84]                         for (int i = 0; !is_ended; ++i)
[85]                         {
[86]                                 tie(token, is_ended) = get_field();
[87]                                 it->second.push_back(vector<bool>(10000));
[88]                                 it->second.back()[0] = (token == "OK");
[89]                         }
[90]                 }
[91]                 ++(task2submissions_cnt[name]);
[92]         }
[93]         map<string, map<vector<bool>, int>> task2bitset2cnt;
[94]         for (auto [n, v] : task2test2bitset)
[95]         {
[96]                 for (auto b : v)
[97]                 {
[98]                         ++(task2bitset2cnt[n][b]);
[99]                 }
[100]        }
[101]
[102]        auto it = task2test2bitset.begin();
[103]        auto it2 = task2bitset2cnt.begin();
[104]        while (it != task2test2bitset.end())
[105]        {
[106]                cout << it->second.size() << ' ' << it2->second.size() << endl;
[107]                ++it;
[108]                ++it2;
[109]        }
[110]        return 0;
[111] }
```