

Олимпиада СПбГУ по информатике 2021/22 учебного года

A	B	C	D	E	F	Sum
100	100	100	80	100	31	511

Task A ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#pragma GCC optimize "Ofast"
#pragma GCC optimize "no-exceptions"
using namespace std;
#define int long long
#define pint pair<int, int>
constexpr int N = 2000000;
signed main() {
    cin.tie(0);
    cout.tie(0);
    ios_base::sync_with_stdio(0);

    int n;
    cin >> n;
    vector<int> a(N);
    vector<pint> q(n);
    for (auto &x : q) cin >> x.first >> x.second;
    sort(q.begin(), q.end(), [&](pint &x, pint &y) {
        if (x.second != y.second) return x.second < y.second;
        return x.first < y.first;
    });
    int base = q[0].second;
    for (auto [x, y] : q) {
        if (y < base + N) a[y - base] += x;
        else break;
    }
    for (int i = 0; i < N - 1; ++i) {
        a[i + 1] += a[i] / 10;
        a[i] %= 10;
        if (a[i] != 0) {
            cout << i + base;
            return 0;
        }
    }
    return 0;
}
```

Task B ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>
#include <map>
#include <bitset>
#include <cstring>
#pragma GCC optimize "Ofast"
#pragma GCC optimize "no-exceptions"
#pragma GCC optimize "fast-math"

using namespace std;
#define uint unsigned int
#define ull uint64_t
#define uint128 __uint128_t

constexpr ull MXLEN = 10000000000000000001;

constexpr uint MOD = 998244353;

uint qmod(const uint &x) {return (x < MOD ? x : x - MOD);}

void qadd(uint &x, const uint &y) {x = (x + y < MOD ? x + y : x + y - MOD);}

uint qadd2(uint &x, const uint &y) {return x = (x + y < MOD ? x + y : x + y - MOD);}

uint qdif(const uint &x, const uint &y) {return (x < y ? x + MOD - y : x - y);}

ull b[3];
signed main() {
    auto flip = [&]() {cout << "Flip_and_wait" << endl;};
    auto wait = [&]() {cout << "Wait" << endl;};
    auto gcnt = [&]() {
        string s;
        cin >> s;
        return (s.size() / 2) - 1;
    };
    int n;
    cin >> n;
    vector<int> t(n);
    for (auto &x : t) cin >> x;
    int mxb = -1;
    for (int i = 0; i < n; ++i) if (t[i]) mxb = i;
    if (mxb == -1) {
        cout << "Stop" << endl;
        return 0;
    }
    flip();
    while (1) {
        int lastbeep = gcnt() - 1;
        while (lastbeep < mxb) {
            wait();
            lastbeep += gcnt();
        }
        t[mxb] -= 1;
        mxb = -1;
        for (int i = 0; i < n; ++i) if (t[i]) mxb = i;
        if (mxb == -1) {
            cout << "Stop" << endl;
            return 0;
        }
        flip();
    }
}

// a[0][0] = 1;
// a[0][1] = 0;
// b[0] = 1;
// uint isq = 1;
```

```

//      for (uint i = 1; i < MOD + 1; ++i) {
//          uint ai0 = a[0];
//          qadd(a[0], a[1]);
//          a[1] = (ai0 * isq) % MOD;
//          isq += 2 * i + 1;
//          while (isq >= MOD) isq -= MOD;
//          if ((a[0] + a[1]) % MOD == 0) cout << i << ' ';
//      }

//a[i] = a[i-1] + a[i-2] * isq
//      for (uint i = 1; i < MOD + 1; ++i) {
//          a[1][0] = a[0][0] + a[0][1];
//          a[1][1] = a[0][0] * isq;
//          isq += 2 * i + 1;
////          isq %= MOD;
//          if ((a[1][0] + a[1][1]) == MOD) cout << i << ' ';
//      }
//      uint dt = 3;
//      for (uint i = 1; i < MOD + 1; ++i) {
//          b[2] = b[1] + b[0] * isq;
//          qadd(isq, dt);
//          qadd(dt, 2);
//          if ((b[0] + b[1]) % MOD == 0) cout << i << ' ';
//      }
//      return 0;
}

```

Task C ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>
#include <map>
#pragma GCC optimize "Ofast"
#pragma GCC optimize "no-exceptions"
using namespace std;

signed main() {

    int t;
    cin >> t;
    map<string, string> dmap;
    dmap["01"] = "0?";
    dmap["00"] = "?0";
    dmap["10"] = "1?";
    dmap["11"] = "?1";
    map<string, string> rmap;
    for (auto [x, y] : dmap) rmap[y] = x;
    while (t--) {
        string a;
        cin >> a;
        bool qm = false;
        for (char c : a) {
            if (c == '?') {
                qm = true; break;
            }
        }
        int n = (int)a.size();
        if (!qm) {
            for (int i = 0; i + 1 < n; i += 2) {
                string b;
                b += a[i];
                b += a[i + 1];
                if (b == "01") cout << "0?";
                else if (b == "10") cout << "1?";
                else if (b == "11") cout << "?1";
                else cout << "?0";
            }
            if (n % 2) cout << a[n - 1];
            cout << '\n';
        } else {
            for (int i = 0; i + 1 < n; i += 2) {
                string b;
                b += a[i];
                b += a[i + 1];
                if (b == "0?") cout << "01";
                else if (b == "1?") cout << "10";
                else if (b == "?1") cout << "11";
                else cout << "00";
            }
            if (n % 2) cout << a[n - 1];
            cout << '\n';
        }
    }
    return 0;
}
```

Task D ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>
#include <map>
#include <bitset>
#include <cstring>
#pragma GCC optimize "Ofast"
#pragma GCC optimize "no-exceptions"
#pragma GCC optimize "fast-math"

using namespace std;
#define uint unsigned int
#define ull uint64_t
#define uint128 __uint128_t

constexpr ull MXLEN = 10000000000000000001;

constexpr uint MOD = 998244353;

uint qmod(const uint &x) {return (x < MOD ? x : x - MOD);}

void qadd(uint &x, const uint &y) {x = (x + y < MOD ? x + y : x + y - MOD);}

uint qdif(const uint &x, const uint &y) {return (x < y ? x + MOD - y : x - y);}

ull b[3];
signed main() {
    cin.tie(0);
    cout.tie(0);
    ios_base::sync_with_stdio(0);
    ull n;
    cin >> n;
    b[0] = 1;
    b[1] = 2;
    b[2] = 6;
    if (n == 1) cout << "0_2";
    else if (n == 2) cout << 0 << ' ' << 6;
    else if (n == 3) cout << 0 << ' ' << 24;
    else {
        uint cc = n / 3;
        uint i = 3;
        uint isq = 9;
        for (uint itr = 0; itr < cc; ++itr) {
            b[0] = qmod(b[2] + b[1] * isq % MOD);
            qadd(isq, qmod(2 * (i++) + 1));
            b[1] = qmod(b[0] + b[2] * isq % MOD);
            qadd(isq, qmod(2 * (i++) + 1));
            b[2] = qmod(b[1] + b[0] * isq % MOD);
            qadd(isq, qmod(2 * (i++) + 1));
        }
        cout << 0 << ' ' << b[n % 3];
    }
}

// a[0][0] = 1;
// a[0][1] = 0;
// b[0] = 1;
// uint isq = 1;

// for (uint i = 1; i < MOD + 1; ++i) {
//     uint ai0 = a[0];
//     qadd(a[0], a[1]);
//     a[1] = (ai0 * isq) % MOD;
//     isq += 2 * i + 1;
//     while (isq >= MOD) isq -= MOD;
//     if ((a[0] + a[1]) % MOD == 0) cout << i << ' ';
// }

// a[i] = a[i-1] + a[i-2] * isq
```

```

//      for (uint i = 1; i < MOD + 1; ++i) {
//          a[1][0] = a[0][0] + a[0][1];
//          a[1][1] = a[0][0] * isq;
//          isq += 2 * i + 1;
////          isq %= MOD;
//          if ((a[1][0] + a[1][1]) == MOD) cout << i << ' ';
//      }
//      uint dt = 3;
//      for (uint i = 1; i < MOD + 1; ++i) {
//          b[2] = b[1] + b[0] * isq;
//          qadd(isq, dt);
//          qadd(dt, 2);
//          if ((b[0] + b[1]) % MOD == 0) cout << i << ' ';
//      }
//      return 0;
}

```

Task E ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>
#include <map>
#include <bitset>
#include <cstring>
#pragma GCC optimize "Ofast"
#pragma GCC optimize "no-exceptions"
using namespace std;
#define uint unsigned int
#define ull uint64_t
#define uint128 __uint128_t

constexpr uint MAXN = 300000;
constexpr uint BS = 2400;
constexpr uint CNT = MAXN / BS;
constexpr uint MAXD = 1000001;

uint a[MAXN];
uint av[CNT][MAXD];
uint stupid[MAXN];
uint vc[300001];

signed main() {
    cin.tie(0);
    cout.tie(0);
    ios_base::sync_with_stdio(0);
    for (uint i = 0; i < 300001; ++i) {
        vc[i] = 0;
    }
    for (uint i = 0; i < MAXN; ++i) {
        a[i] = 1;
    }
    for (uint i = 0; i < MAXN; ++i) {
        stupid[i] = 0;
    }
    for (uint i = 0; i < CNT; ++i) {
        for (uint j = 0; j < MAXD; ++j) {
            av[i][j] = 0;
        }
    }
    uint n;
    cin >> n;
    for (uint i = 0; i < n; ++i) cin >> a[i]; //a[i] = (i * 13) % 300000 + 1;
    stupid[0] = (a[0] == 1);
    for (uint i = 1; i < n; ++i) stupid[i] = stupid[i - 1] + (a[i] == 1);

    uint p = 0;
    for (uint i = 0; i < CNT; ++i, p += BS) {
        uint *cur = av[i];

        memset(vc, 0, sizeof(vc));
        for (uint j = p; j < p + BS; ++j) {
            if (a[j] != 1) ++vc[a[j]];
        }
        for (uint j = 2; j < 300001; ++j) {
            if (vc[j]) {
                for (uint k = 1; k < MAXD; k += j) {
                    cur[k] += vc[j];
                }
            }
        }
        for (uint j = 1; j < MAXD; ++j) cur[j] += cur[j - 1];
    }
    // cout << av[30][15] << ' ';
    int q;
    cin >> q;
    while (q--) {
        uint bg, nd, d;
        cin >> bg >> nd >> d;
        if (nd == bg) {cout << 0 << '\n'; continue;}
    }
}
```

```

uint ndiv = d - 1;
--nd;
uint bgb = bg / BS;
uint ndb = nd / BS;
ull ans = 0;
if (bgb == ndb) {
    for (uint i = bg; i < nd + 1; ++i) {
        ans += ndiv / a[i] + 1;
    }
} else {
    uint fbgn = (bgb + 1) * BS;
    uint lend = ndb * BS;
    ans = (ull)(stupid[lend - 1] - stupid[fbgn - 1]) * (ull)d;
    for (uint i = bg; i < fbgn; ++i) ans += ndiv / a[i] + 1;
    for (uint i = lend; i <= nd; ++i) ans += ndiv / a[i] + 1;
    //      cout << ans << ' ';
    for (uint i = bgb + 1; i < ndb; ++i) ans += av[i][d];
}
cout << ans << '\n';
}
cout << endl;
return 0;
}
/**

100
1
1 100 899850
346

*/

```


Task F ()

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>
#include <map>
#include <bitset>
#include <cstring>
#include <array>
#pragma GCC optimize "Ofast"
#pragma GCC optimize "no-exceptions"
#pragma GCC optimize "fast-math"

using namespace std;
#define uint unsigned int
#define ull uint64_t
#define uint128 __uint128_t
#define ll long long

const array<ll, 2> NOPE = array<ll, 2>{2347234818213, -1347891389129};
const array<ll, 2> ZERO = array<ll, 2>{0, 0};

void mxu(int &x, int y) {if (x < y) x = y;}

signed main() {
    int n;
    cin >> n;
    vector<array<int, 2>> a(n);
    for (auto &x : a) cin >> x[0] >> x[1];
    vector<array<ll, 2>> carry(n);
    for (int i = 0; i < n; ++i) carry[i][0] = a[i][0], carry[i][1] = a[i][1];
    for (int i = 1; i < n; ++i) {
        carry[i][0] += carry[i - 1][0];
        carry[i][1] += carry[i - 1][1];
    }
    vector<int> kek(n);
    // cout << '\n';
    // for (auto &x : carry) cout << x[0] << ' ' << x[1] << '\n';
    // cout << '\n';
    for (int i = 0; i < n; ++i) {
        if (carry[i][0] <= 0 && 0 <= carry[i][1]) kek[i] = 1;
        for (int j = 0; j < i; ++j) {
            if (carry[i][0] - carry[j][0] <= 0 && 0 <= carry[i][1] - carry[j][1]) {
                mxu(kek[i], kek[j] + 1);
            }
        }
    }
    // for (auto &x : kek) cout << x << ' ';
    // cout << '\n';
    int ans = 0;
    for (auto x : kek) mxu(ans, x);
    cout << ans + 1;
    return 0;
}
```