Олимпиада «Ломоносов» по информатике
2025-2026 учебный год. Заключительный этап
Работа участника с id заявки 1618743, логином inf26f_180

Сводный итог по всем задачам в проверяющей системе

Run ID	Time	User name	Problem	Language	Result	Tests	Score
1234	3:53:30	inf26f_180	7	g++	OK	11	100
906	3:15:04	inf26f_180	1	g++	OK	23	100
230	1:02:49	inf26f_180	5	g++	Partial solution	80	77
155	0:50:59	inf26f_180	4	g++	OK	103	100
104	0:39:45	inf26f_180	3	g++	OK	23	100
34	0:24:27	inf26f_180	2	g++	OK	53	100

577 (пятьсот семьдесят семь) технических баллов
82 (восемьдесят два) итоговых балла

председатель Жюри кандидат физ.-мат. наук Малышко В. В.

зам. председателя Жюри кандидат физ.-мат. наук Корныхин Е. В.

Посылка по задаче 1

```
[1]  #include <iostream>
[2]  #include <vector>
[3]  #include <string>
[4]  #include <algorithm>
[5]  #include <unordered_map>
[6]  #include <cmath>
[7]  #include <iomanip>
[8]
[9]  using namespace std;
[10]
[11] long long inf = 1000000005;
[12]
[13] struct Point{
[14]     long long x,y,h;
[15]     Point(){}
[16]     Point(long long x, long long y, long long h) : x(x), y(y), h(h){}
[17] };
[18]
[19] struct Line{
[20]     long long a,b,c;
[21]     Line(Point A, Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[22]     Line(Line l, Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[23] };
[24]
[25] struct d_Line{};
[26]
[27] struct d_Point{
[28]     double x, y;
[29]     d_Point(Point A) : x(A.x), y(A.y){}
[30]     d_Point(double x, double y) : x(x), y(y){}
[31]     /*d_Point(d_Line a, d_Line b){
[32]         double d = a.a * b.b - a.b * b.a;
[33]         double dx = -(a.c * b.b - a.b * b.c);
[34]         double dy = -(a.a * b.c - a.c * b.a);
[35]         x = (double)dx / (double)d;
[36]         y = (double)dy / (double)d;
[37]     }*/
[38] };
[39]
[40] /*struct d_Line{
[41]     long long a,b,c;
[42]     d_Line(Line l) : a(l.a), b(l.b), c(l.c){}
[43]     d_Line(d_Point A, d_Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[44]     d_Line(d_Line l, d_Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[45] };*/
[46]
[47] struct SegTree{
[48]     struct Node{
[49]         long long sum, push = 0;
[50]     };
[51]     vector<Node> tree;
[52]     Node merge(Node a, Node b){
[53]         Node now;
[54]         now.sum = a.sum + b.sum;
[55]         return now;
[56]     }
[57]     void build(int v, int l, int r, vector<long long>& a){
[58]         if(v == 1) tree.resize(r << 2);
[59]         if(l + 1 == r){
[60]             tree[v].sum = a[l];
[61]             return;
[62]         }
[63]         int mid = (l + r) >> 1;
[64]         build(v << 1, l, mid, a);
[65]         build((v << 1) + 1, mid, r, a);
[66]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[67]     }
[68]     void push(int v, long long l, long long r){
[69]         if(!tree[v].push) return;
[70]         tree[v].sum += tree[v].push * (r - l);
[71]         if(l + 1 != r){
[72]             tree[v << 1].push += tree[v].push;
[73]             tree[(v << 1) + 1].push += tree[v].push;
[74]         }
[75]         tree[v].push = 0;
```

```
[76]      }
[77]      void add(int v, int l, int r, int ql, int qr, long long x){
[78]          push(v, l, r);
[79]          if(qr <= l || r <= ql) return;
[80]          if(ql <= l && r <= qr) {
[81]              tree[v].push += x;
[82]              push(v, l, r);
[83]              return;
[84]          }
[85]          int mid = (l + r) >> 1;
[86]          add(v << 1, l, mid, ql, qr, x);
[87]          add((v << 1) + 1, mid, r, ql, qr, x);
[88]          tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[89]      }
[90]      long long get(int v, int l, int r, int ql, int qr){
[91]          push(v, l, r);
[92]          if(qr <= l || r <= ql) return 0;
[93]          if(ql <= l && r <= qr) return tree[v].sum;
[94]          int mid = (l + r) >> 1;
[95]          return get(v << 1, l, mid, ql, qr) + get((v << 1) + 1, mid, r, ql, qr);
[96]      }
[97]  };
[98]
[99]  char can(Point from, Point to){
[100]     return from.x < to.x && from.y < to.y && from.h > to.h;
[101] }
[102]
[103] bool operator<(Point a, Point b){
[104]     return a.h < b.h;
[105] }
[106]
[107] long double PI = acos(-1);
[108] long double e = 1e-11;
[109]
[110] struct telo{
[111]     char init = 0;
[112]     long double a = 0, d = 0, h = 0; //alpha - восхождение, delta - склонение, h - расстояние
[113]     telo(){}
[114]     telo(string delta, string alpha, string dist) : init(1) {
[115]         {
[116]             char sign = delta[0];
[117]             int i = 0;
[118]             while(!(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9')){
[119]                 ++i;
[120]             }
[121]             string number = "";
[122]             while(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9'){
[123]                 number.push_back(delta[i]);
[124]                 ++i;
[125]             }
[126]             d += stold(number);
[127]             while(!(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9')){
[128]                 ++i;
[129]             }
[130]             number = "";
[131]             while(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9'){
[132]                 number.push_back(delta[i]);
[133]                 ++i;
[134]             }
[135]             d += stold(number) / 60.;
[136]             while(!(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9')){
[137]                 ++i;
[138]             }
[139]             number = "";
[140]             while(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9'){
[141]                 number.push_back(delta[i]);
[142]                 ++i;
[143]             }
[144]             d += stold(number) / 3600.;
[145]             d = d * PI / 180.;
[146]             if(sign == '-'){
[147]                 d = -d;
[148]             }
[149]         }
[150]         {
```

```
            int i = 0;
            while(!(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9')){
                ++i;
            }
            string number = "";
            while(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9'){
                number.push_back(alpha[i]);
                ++i;
            }
            a += stold(number) * PI / 12.;
            while(!(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9')){
                ++i;
            }
            number = "";
            while(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9'){
                number.push_back(alpha[i]);
                ++i;
            }
            a += stold(number) * PI / 720.;
            while(!(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9')){
                ++i;
            }
            number = "";
            while(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9'){
                number.push_back(alpha[i]);
                ++i;
            }
            a += stold(number) * PI / 43200.;
        }
        {
            h = stold(dist);
        }
    }
};

long double distance(telo a, telo b){
    long double cos_angle = sin(a.a) * sin(b.a) - cos(a.d) * cos(b.d) * cos(a.d - b.d);
    cout << sqrtl(a.h*a.h + b.h * b.h - 2. * a.h * b.h * cos_angle) << '\n';
    return sqrtl(a.h*a.h + b.h * b.h - 2. * a.h * b.h * cos_angle);
}

unordered_map<char, int> s;

int rim_to_int(string R){
    if(R == "N") return 0;
    vector<int> before(4); //0 - full, 1 - L, 2 - X, 3 - V;
    for(int i = 0; i < R.size(); ++i){
        if(R[i] == 'L'){
            before[0] += 50 - 2 * before[1];
            before[1] = 0;
            before[2] = 0;
            before[3] = 0;
        }else if(R[i] == 'X'){
            before[0] += 10 - 2 * before[2];
            before[1] += 10 - 2 * before[2];
            before[2] = 0;
            before[3] = 0;
        }else if(R[i] == 'V'){
            before[0] += 5 - 2 * before[3];
            before[1] += 5 - 2 * before[3];
            before[2] += 5 - 2 * before[3];
            before[3] = 0;
        }else{
            before[0]++;
            before[1]++;
            before[2]++;
            before[3]++;
        }
    }
    return before[0];
}

int main(){
    ios_base::sync_with_stdio(false);
    cin.tie(nullptr);
```

```
[226]     cout.tie(nullptr);
[227]     int k;
[228]     cin >> k;
[229]     string s;
[230]     cin >> s;
[231]     vector<int> ma;
[232]     vector<int> mi;
[233]     string null = "";
[234]     for(char c : s){
[235]         if(c == '.'){
[236]             ma.push_back(rim_to_int(null));
[237]             mi.push_back(rim_to_int(null));
[238]             null = "";
[239]         }else{
[240]             null.push_back(c);
[241]         }
[242]     }
[243]     ma.push_back(rim_to_int(null));
[244]     mi.push_back(rim_to_int(null));
[245]     null = "";
[246]     while(--k){
[247]         cin >> s;
[248]         vector<int> vr;
[249]         for(char c : s){
[250]             if(c == '.'){
[251]                 vr.push_back(rim_to_int(null));
[252]                 null = "";
[253]             }else{
[254]                 null.push_back(c);
[255]             }
[256]         }
[257]         vr.push_back(rim_to_int(null));
[258]         null = "";
[259]         if(vr.size() > ma.size()){
[260]             ma = vr;
[261]         }else if(vr.size() == ma.size()){
[262]             for(int i = 0; i < vr.size(); ++i){
[263]                 if(vr[i] > ma[i]){
[264]                     ma = vr;
[265]                     break;
[266]                 }
[267]                 if(vr[i] < ma[i]) break;
[268]             }
[269]         }
[270]         if(vr.size() < mi.size()){
[271]             mi = vr;
[272]         }else if(vr.size() == mi.size()){
[273]             for(int i = 0; i < vr.size(); ++i){
[274]                 if(vr[i] < mi[i]){
[275]                     mi = vr;
[276]                     break;
[277]                 }
[278]                 if(vr[i] > mi[i]) break;
[279]             }
[280]         }
[281]     }
[282]     for(int i = 0; i < mi.size(); ++i){
[283]         if(mi[i] < 26){
[284]             cout << char('A' + mi[i]);
[285]         }else{
[286]             cout << char('a' + mi[i] - 26);
[287]         }
[288]     }
[289]     cout << '\n';
[290]     for(int i = 0; i < ma.size(); ++i){
[291]         if(ma[i] < 26){
[292]             cout << char('A' + ma[i]);
[293]         }else{
[294]             cout << char('a' + ma[i] - 26);
[295]         }
[296]     }
[297]     cout << '\n';
[298] }
```

Посылка по задаче 2

```
[1]  #include <iostream>
[2]  #include <vector>
[3]
[4]  using namespace std;
[5]
[6]  struct Point{
[7]      long long x,y;
[8]      Point(){}
[9]      Point(long long x, long long y) : x(x), y(y){}
[10] };
[11]
[12] struct Line{
[13]     long long a,b,c;
[14]     Line(Point A, Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[15]     Line(Line l, Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[16] };
[17]
[18] struct d_Line{};
[19]
[20] struct d_Point{
[21]     double x, y;
[22]     d_Point(Point A) : x(A.x), y(A.y){}
[23]     d_Point(double x, double y) : x(x), y(y){}
[24]     /*d_Point(d_Line a, d_Line b){
[25]         double d = a.a * b.b - a.b * b.a;
[26]         double dx = -(a.c * b.b - a.b * b.c);
[27]         double dy = -(a.a * b.c - a.c * b.a);
[28]         x = (double)dx / (double)d;
[29]         y = (double)dy / (double)d;
[30]     }*/
[31] };
[32]
[33] /*struct d_Line{
[34]     long long a,b,c;
[35]     d_Line(Line l) : a(l.a), b(l.b), c(l.c){}
[36]     d_Line(d_Point A, d_Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[37]     d_Line(d_Line l, d_Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[38] };*/
[39]
[40] struct SegTree{
[41]     struct Node{
[42]         long long sum, push = 0;
[43]     };
[44]     vector<Node> tree;
[45]     Node merge(Node a, Node b){
[46]         Node now;
[47]         now.sum = a.sum + b.sum;
[48]         return now;
[49]     }
[50]     void build(int v, int l, int r, vector<long long>& a){
[51]         if(v == 1) tree.resize(r << 2);
[52]         if(l + 1 == r){
[53]             tree[v].sum = a[l];
[54]             return;
[55]         }
[56]         int mid = (l + r) >> 1;
[57]         build(v << 1, l, mid, a);
[58]         build((v << 1) + 1, mid, r, a);
[59]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[60]     }
[61]     void push(int v, long long l, long long r){
[62]         if(!tree[v].push) return;
[63]         tree[v].sum += tree[v].push * (r - l);
[64]         if(l + 1 != r){
[65]             tree[v << 1].push += tree[v].push;
[66]             tree[(v << 1) + 1].push += tree[v].push;
[67]         }
[68]         tree[v].push = 0;
[69]     }
[70]     void add(int v, int l, int r, int ql, int qr, long long x){
[71]         push(v, l, r);
[72]         if(qr <= l || r <= ql) return;
[73]         if(ql <= l && r <= qr) {
[74]             tree[v].push += x;
[75]             push(v, l, r);
```

```
[76]             return;
[77]         }
[78]         int mid = (l + r) >> 1;
[79]         add(v << 1, l, mid, ql, qr, x);
[80]         add((v << 1) + 1, mid, r, ql, qr, x);
[81]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[82]     }
[83]     long long get(int v, int l, int r, int ql, int qr){
[84]         push(v, l, r);
[85]         if(qr <= l || r <= ql) return 0;
[86]         if(ql <= l && r <= qr) return tree[v].sum;
[87]         int mid = (l + r) >> 1;
[88]         return get(v << 1, l, mid, ql, qr) + get((v << 1) + 1, mid, r, ql, qr);
[89]     }
[90] };
[91]
[92] int main(){
[93]     ios_base::sync_with_stdio(false);
[94]     cin.tie(nullptr);
[95]     cout.tie(nullptr);
[96]     vector<char> re(160001, 1);
[97]     re[0] = re[1] = 0;
[98]     for(int d = 2; d * d <= 160000; ++d){
[99]         if(re[d]){
[100]            for(int i = d * d; i <= 160000; i += d){
[101]                re[i] = 0;
[102]            }
[103]        }
[104]    }
[105]    int n;
[106]    cin >> n;
[107]    vector<int> cnt_br(160001, 0);
[108]    for(int i = 0; i < n; ++i){
[109]        int a;
[110]        cin >> a;
[111]        char br = 0;
[112]        for(int b = 2; b * b + b + 1 <= a; ++b){
[113]            int now_sum = b*b+b+1;
[114]            while(now_sum < a){
[115]                now_sum = now_sum * b + 1;
[116]            }
[117]            if(now_sum == a){
[118]                br = 1;
[119]                break;
[120]            }
[121]        }
[122]        if(re[a] && br) ++cnt_br[a];
[123]    }
[124]    int res = 0;
[125]    int cnt_res = 0;
[126]    for(int i = 1; i<= 160000; ++i){
[127]        if(!cnt_br[i]) continue;
[128]        //cout << i << ' ' << cnt_br[i] << '\n';
[129]        if(cnt_br[i] >= cnt_res){
[130]            res = i;
[131]            cnt_res = cnt_br[i];
[132]        }
[133]    }
[134]    cout << res;
[135] }
```

Посылка по задаче 3

```
[1]  #include <iostream>
[2]  #include <vector>
[3]  #include <cmath>
[4]  #include <iomanip>
[5]
[6]  using namespace std;
[7]
[8]  struct Point{
[9]      long long x,y;
[10]     Point(){}
[11]     Point(long long x, long long y) : x(x), y(y){}
[12] };
[13]
[14] struct Line{
[15]     long long a,b,c;
[16]     Line(Point A, Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[17]     Line(Line l, Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[18] };
[19]
[20] struct d_Line{};
[21]
[22] struct d_Point{
[23]     double x, y;
[24]     d_Point(Point A) : x(A.x), y(A.y){}
[25]     d_Point(double x, double y) : x(x), y(y){}
[26]     /*d_Point(d_Line a, d_Line b){
[27]         double d = a.a * b.b - a.b * b.a;
[28]         double dx = -(a.c * b.b - a.b * b.c);
[29]         double dy = -(a.a * b.c - a.c * b.a);
[30]         x = (double)dx / (double)d;
[31]         y = (double)dy / (double)d;
[32]     }*/
[33] };
[34]
[35] /*struct d_Line{
[36]     long long a,b,c;
[37]     d_Line(Line l) : a(l.a), b(l.b), c(l.c){}
[38]     d_Line(d_Point A, d_Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[39]     d_Line(d_Line l, d_Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[40] };*/
[41]
[42] struct SegTree{
[43]     struct Node{
[44]         long long sum, push = 0;
[45]     };
[46]     vector<Node> tree;
[47]     Node merge(Node a, Node b){
[48]         Node now;
[49]         now.sum = a.sum + b.sum;
[50]         return now;
[51]     }
[52]     void build(int v, int l, int r, vector<long long>& a){
[53]         if(v == 1) tree.resize(r << 2);
[54]         if(l + 1 == r){
[55]             tree[v].sum = a[l];
[56]             return;
[57]         }
[58]         int mid = (l + r) >> 1;
[59]         build(v << 1, l, mid, a);
[60]         build((v << 1) + 1, mid, r, a);
[61]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[62]     }
[63]     void push(int v, long long l, long long r){
[64]         if(!tree[v].push) return;
[65]         tree[v].sum += tree[v].push * (r - l);
[66]         if(l + 1 != r){
[67]             tree[v << 1].push += tree[v].push;
[68]             tree[(v << 1) + 1].push += tree[v].push;
[69]         }
[70]         tree[v].push = 0;
[71]     }
[72]     void add(int v, int l, int r, int ql, int qr, long long x){
[73]         push(v, l, r);
[74]         if(qr <= l || r <= ql) return;
[75]         if(ql <= l && r <= qr) {
```

```
[76]             tree[v].push += x;
[77]             push(v, l, r);
[78]             return;
[79]         }
[80]         int mid = (l + r) >> 1;
[81]         add(v << 1, l, mid, ql, qr, x);
[82]         add((v << 1) + 1, mid, r, ql, qr, x);
[83]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[84]     }
[85]     long long get(int v, int l, int r, int ql, int qr){
[86]         push(v, l, r);
[87]         if(qr <= l || r <= ql) return 0;
[88]         if(ql <= l && r <= qr) return tree[v].sum;
[89]         int mid = (l + r) >> 1;
[90]         return get(v << 1, l, mid, ql, qr) + get((v << 1) + 1, mid, r, ql, qr);
[91]     }
[92] };
[93]
[94] int main(){
[95]     ios_base::sync_with_stdio(false);
[96]     cin.tie(nullptr);
[97]     cout.tie(nullptr);
[98]     long long n;
[99]     cin >> n;
[100]    long long j, i;
[101]    cin >> j >> i;
[102]    long long l = 1, r = n;
[103]    long long ceil = 0;
[104]    while(l != j && l != i && r != j && r != i){
[105]        ceil += (n - 1) * 4ll;
[106]        n -= 2;
[107]        ++l;
[108]        --r;
[109]    }
[110]    if(n == 1){
[111]        cout << ceil;
[112]        return 0;
[113]    }
[114]    if(j == l && i != r){
[115]        ceil += i - l;
[116]    }else if(i == r && j != r){
[117]        ceil += n - 1;
[118]        ceil += j - l;
[119]    }else if(j == r && i != l){
[120]        ceil += 2 * (n - 1);
[121]        ceil += r - i;
[122]    }else{
[123]        ceil += 3 * (n - 1);
[124]        ceil += r - j;
[125]    }
[126]    cout << ceil;
[127] }
```

Посылка по задаче 4

```
[1]  #include <iostream>
[2]  #include <vector>
[3]  #include <algorithm>
[4]  #include <cmath>
[5]  #include <iomanip>
[6]
[7]  using namespace std;
[8]
[9]  long long inf = 1000000005;
[10]
[11] struct Point{
[12]     long long x,y;
[13]     Point(){}
[14]     Point(long long x, long long y) : x(x), y(y){}
[15] };
[16]
[17] struct Line{
[18]     long long a,b,c;
[19]     Line(Point A, Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[20]     Line(Line l, Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[21] };
[22]
[23] struct d_Line{};
[24]
[25] struct d_Point{
[26]     double x, y;
[27]     d_Point(Point A) : x(A.x), y(A.y){}
[28]     d_Point(double x, double y) : x(x), y(y){}
[29]     /*d_Point(d_Line a, d_Line b){
[30]         double d = a.a * b.b - a.b * b.a;
[31]         double dx = -(a.c * b.b - a.b * b.c);
[32]         double dy = -(a.a * b.c - a.c * b.a);
[33]         x = (double)dx / (double)d;
[34]         y = (double)dy / (double)d;
[35]     }*/
[36] };
[37]
[38] /*struct d_Line{
[39]     long long a,b,c;
[40]     d_Line(Line l) : a(l.a), b(l.b), c(l.c){}
[41]     d_Line(d_Point A, d_Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[42]     d_Line(d_Line l, d_Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[43] };*/
[44]
[45] struct SegTree{
[46]     struct Node{
[47]         long long sum, push = 0;
[48]     };
[49]     vector<Node> tree;
[50]     Node merge(Node a, Node b){
[51]         Node now;
[52]         now.sum = a.sum + b.sum;
[53]         return now;
[54]     }
[55]     void build(int v, int l, int r, vector<long long>& a){
[56]         if(v == 1) tree.resize(r << 2);
[57]         if(l + 1 == r){
[58]             tree[v].sum = a[l];
[59]             return;
[60]         }
[61]         int mid = (l + r) >> 1;
[62]         build(v << 1, l, mid, a);
[63]         build((v << 1) + 1, mid, r, a);
[64]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[65]     }
[66]     void push(int v, long long l, long long r){
[67]         if(!tree[v].push) return;
[68]         tree[v].sum += tree[v].push * (r - l);
[69]         if(l + 1 != r){
[70]             tree[v << 1].push += tree[v].push;
[71]             tree[(v << 1) + 1].push += tree[v].push;
[72]         }
[73]         tree[v].push = 0;
[74]     }
```

```
[75]      void add(int v, int l, int r, int ql, int qr, long long x){
[76]          push(v, l, r);
[77]          if(qr <= l || r <= ql) return;
[78]          if(ql <= l && r <= qr) {
[79]              tree[v].push += x;
[80]              push(v, l, r);
[81]              return;
[82]          }
[83]          int mid = (l + r) >> 1;
[84]          add(v << 1, l, mid, ql, qr, x);
[85]          add((v << 1) + 1, mid, r, ql, qr, x);
[86]          tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[87]      }
[88]      long long get(int v, int l, int r, int ql, int qr){
[89]          push(v, l, r);
[90]          if(qr <= l || r <= ql) return 0;
[91]          if(ql <= l && r <= qr) return tree[v].sum;
[92]          int mid = (l + r) >> 1;
[93]          return get(v << 1, l, mid, ql, qr) + get((v << 1) + 1, mid, r, ql, qr);
[94]      }
[95]  };
[96]
[97]  int main(){
[98]      ios_base::sync_with_stdio(false);
[99]      cin.tie(nullptr);
[100]     cout.tie(nullptr);
[101]     int n;
[102]     cin >> n;
[103]     vector<int> a(n);
[104]     vector<int> dp(n + 1, inf);
[105]     dp[0] = 0;
[106]     for(int i = 0; i < n; ++i) cin >> a[i];
[107]     reverse(a.begin(), a.end());
[108]     for(int i = 0; i < n; ++i){
[109]         auto it = upper_bound(dp.begin(),dp.end(), a[i]);
[110]         dp[it - dp.begin()] = a[i];
[111]     }
[112]     for(int i = n; i > 0; --i){
[113]         if(dp[i] != inf){
[114]             cout << n - i;
[115]             return 0;
[116]         }
[117]     }
[118] }
```

Посылка по задаче 5

```
[1]  #include <iostream>
[2]  #include <vector>
[3]  #include <algorithm>
[4]  #include <cmath>
[5]  #include <iomanip>
[6]
[7]  using namespace std;
[8]
[9]  long long inf = 1000000005;
[10]
[11] struct Point{
[12]     long long x,y,h;
[13]     Point(){}
[14]     Point(long long x, long long y, long long h) : x(x), y(y), h(h){}
[15] };
[16]
[17] struct Line{
[18]     long long a,b,c;
[19]     Line(Point A, Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[20]     Line(Line l, Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[21] };
[22]
[23] struct d_Line{};
[24]
[25] struct d_Point{
[26]     double x, y;
[27]     d_Point(Point A) : x(A.x), y(A.y){}
[28]     d_Point(double x, double y) : x(x), y(y){}
[29]     /*d_Point(d_Line a, d_Line b){
[30]         double d = a.a * b.b - a.b * b.a;
[31]         double dx = -(a.c * b.b - a.b * b.c);
[32]         double dy = -(a.a * b.c - a.c * b.a);
[33]         x = (double)dx / (double)d;
[34]         y = (double)dy / (double)d;
[35]     }*/
[36] };
[37]
[38] /*struct d_Line{
[39]     long long a,b,c;
[40]     d_Line(Line l) : a(l.a), b(l.b), c(l.c){}
[41]     d_Line(d_Point A, d_Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[42]     d_Line(d_Line l, d_Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[43] };*/
[44]
[45] struct SegTree{
[46]     struct Node{
[47]         long long sum, push = 0;
[48]     };
[49]     vector<Node> tree;
[50]     Node merge(Node a, Node b){
[51]         Node now;
[52]         now.sum = a.sum + b.sum;
[53]         return now;
[54]     }
[55]     void build(int v, int l, int r, vector<long long>& a){
[56]         if(v == 1) tree.resize(r << 2);
[57]         if(l + 1 == r){
[58]             tree[v].sum = a[l];
[59]             return;
[60]         }
[61]         int mid = (l + r) >> 1;
[62]         build(v << 1, l, mid, a);
[63]         build((v << 1) + 1, mid, r, a);
[64]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[65]     }
[66]     void push(int v, long long l, long long r){
[67]         if(!tree[v].push) return;
[68]         tree[v].sum += tree[v].push * (r - l);
[69]         if(l + 1 != r){
[70]             tree[v << 1].push += tree[v].push;
[71]             tree[(v << 1) + 1].push += tree[v].push;
[72]         }
[73]         tree[v].push = 0;
[74]     }
```

```
[75]      void add(int v, int l, int r, int ql, int qr, long long x){
[76]          push(v, l, r);
[77]          if(qr <= l || r <= ql) return;
[78]          if(ql <= l && r <= qr) {
[79]              tree[v].push += x;
[80]              push(v, l, r);
[81]              return;
[82]          }
[83]          int mid = (l + r) >> 1;
[84]          add(v << 1, l, mid, ql, qr, x);
[85]          add((v << 1) + 1, mid, r, ql, qr, x);
[86]          tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[87]      }
[88]      long long get(int v, int l, int r, int ql, int qr){
[89]          push(v, l, r);
[90]          if(qr <= l || r <= ql) return 0;
[91]          if(ql <= l && r <= qr) return tree[v].sum;
[92]          int mid = (l + r) >> 1;
[93]          return get(v << 1, l, mid, ql, qr) + get((v << 1) + 1, mid, r, ql, qr);
[94]      }
[95]  };
[96]
[97]  char can(Point from, Point to){
[98]      return from.x < to.x && from.y < to.y && from.h > to.h;
[99]  }
[100]
[101] bool operator<(Point a, Point b){
[102]     return a.h < b.h;
[103] }
[104]
[105] int main(){
[106]     ios_base::sync_with_stdio(false);
[107]     cin.tie(nullptr);
[108]     cout.tie(nullptr);
[109]     int n;
[110]     cin >> n;
[111]     vector<Point> peaks(n);
[112]     for(int i = 0; i < n; ++i){
[113]         long long x,y,h;
[114]         cin >> x >> y >> h;
[115]         peaks[i] = Point(x, y, h);
[116]     }
[117]     sort(peaks.rbegin(), peaks.rend());
[118]     vector<int> ans(n, 1);
[119]     for(int i = 0; i < n; ++i){
[120]         for(int j = i + 1; j < n; ++j){
[121]             if(can(peaks[i], peaks[j])) ans[j] = ans[i] + 1;
[122]         }
[123]     }
[124]     int result = 1;
[125]     for(int i = 0; i < n; ++i){
[126]         result = max(result, ans[i]);
[127]     }
[128]     cout << result;
[129] }
```

Посылка по задаче 6

Посылок по задаче 6 участником не было отправлено.

Посылка по задаче 7

```
[1]  #include <iostream>
[2]  #include <vector>
[3]  #include <string>
[4]  #include <algorithm>
[5]  #include <unordered_map>
[6]  #include <cmath>
[7]  #include <iomanip>
[8]
[9]  using namespace std;
[10]
[11] long long inf = 1000000005;
[12]
[13] struct Point{
[14]     long long x,y,h;
[15]     Point(){}
[16]     Point(long long x, long long y, long long h) : x(x), y(y), h(h){}
[17] };
[18]
[19] struct Line{
[20]     long long a,b,c;
[21]     Line(Point A, Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[22]     Line(Line l, Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[23] };
[24]
[25] struct d_Line{};
[26]
[27] struct d_Point{
[28]     double x, y;
[29]     d_Point(Point A) : x(A.x), y(A.y){}
[30]     d_Point(double x, double y) : x(x), y(y){}
[31]     /*d_Point(d_Line a, d_Line b){
[32]         double d = a.a * b.b - a.b * b.a;
[33]         double dx = -(a.c * b.b - a.b * b.c);
[34]         double dy = -(a.a * b.c - a.c * b.a);
[35]         x = (double)dx / (double)d;
[36]         y = (double)dy / (double)d;
[37]     }*/
[38] };
[39]
[40] /*struct d_Line{
[41]     long long a,b,c;
[42]     d_Line(Line l) : a(l.a), b(l.b), c(l.c){}
[43]     d_Line(d_Point A, d_Point B) : a(B.y - A.y), b(A.x - B.x), c(B.x * A.y - B.y * A.x){}
[44]     d_Line(d_Line l, d_Point A) : a(-l.b), b(l.a), c(l.b * A.x - l.a * A.y){}
[45] };*/
[46]
[47] struct SegTree{
[48]     struct Node{
[49]         long long sum, push = 0;
[50]     };
[51]     vector<Node> tree;
[52]     Node merge(Node a, Node b){
[53]         Node now;
[54]         now.sum = a.sum + b.sum;
[55]         return now;
[56]     }
[57]     void build(int v, int l, int r, vector<long long>& a){
[58]         if(v == 1) tree.resize(r << 2);
[59]         if(l + 1 == r){
[60]             tree[v].sum = a[l];
[61]             return;
[62]         }
[63]         int mid = (l + r) >> 1;
[64]         build(v << 1, l, mid, a);
[65]         build((v << 1) + 1, mid, r, a);
[66]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[67]     }
[68]     void push(int v, long long l, long long r){
[69]         if(!tree[v].push) return;
[70]         tree[v].sum += tree[v].push * (r - l);
[71]         if(l + 1 != r){
[72]             tree[v << 1].push += tree[v].push;
[73]             tree[(v << 1) + 1].push += tree[v].push;
[74]         }
[75]         tree[v].push = 0;
[76]     }
```

```
[77]     void add(int v, int l, int r, int ql, int qr, long long x){
[78]         push(v, l, r);
[79]         if(qr <= l || r <= ql) return;
[80]         if(ql <= l && r <= qr) {
[81]             tree[v].push += x;
[82]             push(v, l, r);
[83]             return;
[84]         }
[85]         int mid = (l + r) >> 1;
[86]         add(v << 1, l, mid, ql, qr, x);
[87]         add((v << 1) + 1, mid, r, ql, qr, x);
[88]         tree[v] = merge(tree[v << 1], tree[(v << 1) + 1]);
[89]     }
[90]     long long get(int v, int l, int r, int ql, int qr){
[91]         push(v, l, r);
[92]         if(qr <= l || r <= ql) return 0;
[93]         if(ql <= l && r <= qr) return tree[v].sum;
[94]         int mid = (l + r) >> 1;
[95]         return get(v << 1, l, mid, ql, qr) + get((v << 1) + 1, mid, r, ql, qr);
[96]     }
[97] };
[98]
[99] char can(Point from, Point to){
[100]     return from.x < to.x && from.y < to.y && from.h > to.h;
[101] }
[102]
[103] bool operator<(Point a, Point b){
[104]     return a.h < b.h;
[105] }
[106]
[107] long double PI = acos(-1);
[108] long double e = 1e-11;
[109]
[110] struct telo{
[111]     char init = 0;
[112]     long double a = 0, d = 0, h = 0; //alpha - восхождение, delta - склонение, h - расстояние
[113]     telo(){}
[114]     telo(string delta, string alpha, string dist) : init(1) {
[115]         {
[116]             char sign = delta[0];
[117]             int i = 0;
[118]             while(!(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9')){
[119]                 ++i;
[120]             }
[121]             string number = "";
[122]             while(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9'){
[123]                 number.push_back(delta[i]);
[124]                 ++i;
[125]             }
[126]             d += stold(number);
[127]             while(!(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9')){
[128]                 ++i;
[129]             }
[130]             number = "";
[131]             while(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9'){
[132]                 number.push_back(delta[i]);
[133]                 ++i;
[134]             }
[135]             d += stold(number) / 60.;
[136]             while(!(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9')){
[137]                 ++i;
[138]             }
[139]             number = "";
[140]             while(delta[i] == '.' || delta[i] >= '0' && delta[i] <= '9'){
[141]                 number.push_back(delta[i]);
[142]                 ++i;
[143]             }
[144]             d += stold(number) / 3600.;
[145]             d = d * PI / 180.;
[146]             if(sign == '-'){
[147]                 d = -d;
[148]             }
[149]         }
[150]         {
[151]             int i = 0;
[152]             while(!(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9')){
[153]                 ++i;
[154]             }
```

```
[155]            string number = "";
[156]            while(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9'){
[157]                number.push_back(alpha[i]);
[158]                ++i;
[159]            }
[160]            a += stold(number) * PI / 12.;
[161]            while(!(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9')){
[162]                ++i;
[163]            }
[164]            number = "";
[165]            while(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9'){
[166]                number.push_back(alpha[i]);
[167]                ++i;
[168]            }
[169]            a += stold(number) * PI / 720.;
[170]            while(!(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9')){
[171]                ++i;
[172]            }
[173]            number = "";
[174]            while(alpha[i] == '.' || alpha[i] >= '0' && alpha[i] <= '9'){
[175]                number.push_back(alpha[i]);
[176]                ++i;
[177]            }
[178]            a += stold(number) * PI / 43200.;
[179]        }
[180]        {
[181]            h = stold(dist);
[182]        }
[183]    }
[184] };
[185] /*meow +60°0′0″ 0h0m0s 1
[186] --
[187] meow +0°0′0″ 0h0m0s 1*/
[188] long double distance(telo a, telo b){
[189]    long double cos_angle = cos_angle = sin(a.d) * sin(b.d) + cos(a.d) * cos(b.d) * cos(a.a - b.a);
[190]    return sqrtl(a.h*a.h + b.h * b.h - 2. * a.h * b.h * cos_angle);
[191] }
[192]
[193] int main(){
[194]    cout << setprecision(10);
[195]    ios_base::sync_with_stdio(false);
[196]    cin.tie(nullptr); //meow +60°0′0″ 0h0m0s 1
[197]    cout.tie(nullptr);//meow +0°0′0″ 0h0m0s 1
[198]    unordered_map<string, telo> base;
[199]    string s, alpha, delta, dist;
[200]    cin >> s;
[201]    while(s != "--"){
[202]        cin >> delta >> alpha >> dist;
[203]        base[s] = telo(delta, alpha, dist);
[204]        cin >> s;
[205]    }
[206]    long double ans = -1;
[207]    char someone = 0;
[208]    while(cin >> s){
[209]        cin >> delta >> alpha >> dist;
[210]        if(base[s].init){
[211]            telo finish(delta, alpha, dist);
[212]            ans = max(ans, distance(base[s], finish));
[213]            someone = 1;
[214]        }
[215]    }
[216]    if(someone){
[217]        cout << setprecision(10) << ans;
[218]    }else{
[219]        cout << -1;
[220]    }
[221] }
```