Олимпиада СПбГУ по информатике 2020/21 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	55	20	475

Task A ()

```
#include <bits/stdc++.h>

using namespace std;

//#define int long long


const long double PI = 3.1415926535897;

struct Point
{
    long long x, y;
};

Point operator -(Point a)
{
    return {-a.x, -a.y};
}

Point operator +(Point a, Point b)
{
    return {a.x + b.x, a.y + b.y};
}

long long sp(Point a, Point b)
{
    return a.x * b.x + a.y * b.y;
}

long long vp(Point a, Point b)
{
    return a.x * b.y - a.y * b.x;
}

Point vect(Point a, Point b)
{
    return {b.x - a.x, b.y - a.y};
}

long double len(Point a)
{
    return sqrt(a.x * a.x + a.y * a.y);
}

long long angle(Point a, Point b)
{
    return atan2(vp(a, b), sp(a, b));
}


bool cmp(Point a, Point b)
{
        return vp(a, b) > 0 || (vp(a, b) == 0 && len(a) < len(b));
}

long double sq(vector<Point> &a)
```

```
{
        long double res = 0;
        for (int i = 0; i < a.size(); i++)
        {
                Point p1 = a[(i - 1 + a.size()) % a.size()];
        Point p2 = a[i];
                res += (p1.x - p2.x) * (p1.y + p2.y);
        }
        return abs(res) / 2;
}

long long per(vector<Point> &a)
{
    long long res = 0;
        for (int i = 0; i < a.size() - 1; i++)
        {
                Point p1 = vect(a[i], a[i + 1]);
                res += len(p1);
        }
        Point p1 = vect(a[0], a[a.size() - 1]);
    res += len(p1);
    return res;
}

vector<Point> convex_hull(vector<Point> z)
{
    int n = z.size();
    Point s = z[0];
    for (int i = 1; i < n; i++)
    {
        if (z[i].x < s.x || (z[i].x == s.x && z[i].y < s.y))
        {
            s = z[i];
        }
    }
    for (auto &x : z)
    {
        x = vect(s, x);
    }
    sort(z.begin(), z.end(), cmp);
    vector<Point> h;
    h.push_back(z[0]);
    for (int i = 1; i < n; i++)
    {
        while (h.size() >= 2)
        {
            Point a = h[h.size() - 2];
            Point b = h[h.size() - 1];
            Point c = z[i];
            if (vp(vect(b, c), (vect(b, a))) > 0)
            {
                break;
            }
            if ((vp(vect(b, c), (vect(b, a))) == 0 && len(vect(a, c)) < len(vect(a, b))))
            {
                break;
            }
            h.pop_back();
        }
        h.push_back(z[i]);
    }
    for (auto &x : h)
    {
        x.x += s.x;
        x.y += s.y;
    }
    return h;
}

int point_in(vector<Point> &a, Point xy, Point start)
{
    int n = a.size();
    xy = vect(start, xy);
    if (xy.x == a[0].x && xy.y == a[0].y)
    {
        return 1;
    }
    if (vp(xy, a[1]) > 0)
    {
        return 0;
    }
    if (vp(xy, a[n - 1]) < 0)
    {
        return 0;
    }
    int l = 1, r = n - 1;
    while (l < r - 1)
    {
        int m = (l + r) / 2;
        if (vp(xy, a[m]) > 0)
        {
            r = m;
        }
        else
        {
            l = m;
        }
    }
    vector<Point> g1 = {a[0], a[l], xy};
    vector<Point> g2 = {a[0], a[l + 1], xy};
    vector<Point> g3 = {a[l], a[l + 1], xy};
    vector<Point> g4 = {a[0], a[l + 1], a[l]};
    if (sq(g1) + sq(g2) + sq(g3) == sq(g4))
    {
        return 1;
    }
    return 0;
}

vector<Point> minc_sum(vector<Point> a, vector<Point> b)
{
    int n = a.size();
    int m = b.size();
    int i = 0, j = 0;
    for (int h = 0; h < n; ++h)
    {
        if (a[h].y < a[i].y || a[h].y == a[i].y && a[h].x < a[i].x)
        {
            i = h;
        }
    }
    for (int h = 0; h < m; ++h)
    {
        if (b[h].y < b[j].y || b[h].y == b[j].y && b[h].x < b[j].x)
        {
            j = h;
        }
    }
    vector<Point> s;
    s.push_back(a[i] + b[j]);
    vector<int> used1(n), used2(m);
    for (;;)
    {
        i %= n;
        j %= m;
        if (used1[i] && used2[j])
        {
            break;
        }
        if (used1[i] || !used2[j] && (a[(i + 1) % n].x - a[i].x) * (b[(j + 1) % m].y - b[j].y) - (
            a[(i + 1) % n].y - a[i].y) * (b[(j + 1) % m].x - b[j].x) < 0)
        {
            s.push_back(s.back() + vect(b[j], b[(j + 1) % m]));
            used2[j] = 1;
            j++;
        }
        else
        {
            s.push_back(s.back() + vect(a[i], a[(i + 1) % n]));
            used1[i] = 1;
            i++;
        }
    }
    s = convex_hull(s);
    return s;
}


signed main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
    //freopen("output.txt", "w", stdout);
    //freopen("input.txt", "r", stdin);
    int k;
    cin >> k;
    if (k <= 10)
    {
        cout << k % 10 << endl;
    }
    else
    {
        if ((k - 10) % 9 != 0)
        {
            cout << (k - 10) % 9 + 1 << endl;
        }
        else
        {
            cout << 0;
        }
    }
}

//tyanocku hocheca
```

Task B ()

```cpp
#include <bits/stdc++.h>

using namespace std;

//#define int long long

vector<vector<long long> > t;


int n;
int sum2[1501][1501][21];


long long sum(int x, int y)
{
    long long result = 0;
    for (int i = x; i >= 0; i = (i & (i+1)) - 1)
    {
        for (int j = y; j >= 0; j = (j & (j+1)) - 1)
        {
            result += t[i][j];
        }
    }
    return result;
}
void inc(int x, int y, int delta)
{
    for (int i = x; i < n + 1; i = (i | (i+1)))
    {
        for (int j = y; j < n + 1; j = (j | (j+1)))
        {
            t[i][j] += delta;
        }
    }

}

const long double PI = 3.1415926535897;

struct Point
{
    long long x, y;
};

Point operator -(Point a)
{
    return {-a.x, -a.y};
}

Point operator +(Point a, Point b)
{
    return {a.x + b.x, a.y + b.y};
}

long long sp(Point a, Point b)
{
    return a.x * b.x + a.y * b.y;
}

long long vp(Point a, Point b)
{
    return a.x * b.y - a.y * b.x;
}

Point vect(Point a, Point b)
{
    return {b.x - a.x, b.y - a.y};
}

long double len(Point a)
{
    return sqrt(a.x * a.x + a.y * a.y);
```

```
}

long long angle(Point a, Point b)
{
    return atan2(vp(a, b), sp(a, b));
}


bool cmp(Point a, Point b)
{
        return vp(a, b) > 0 || (vp(a, b) == 0 && len(a) < len(b));
}

long double sq(vector<Point> &a)
{
        long double res = 0;
        for (int i = 0; i < a.size(); i++)
        {
                Point p1 = a[(i - 1 + a.size()) % a.size()];
        Point p2 = a[i];
                res += (p1.x - p2.x) * (p1.y + p2.y);
        }
        return abs(res) / 2;
}

long long per(vector<Point> &a)
{
    long long res = 0;
        for (int i = 0; i < a.size() - 1; i++)
        {
                Point p1 = vect(a[i], a[i + 1]);
                res += len(p1);
        }
        Point p1 = vect(a[0], a[a.size() - 1]);
    res += len(p1);
    return res;
}

vector<Point> convex_hull(vector<Point> z)
{
    int n = z.size();
    Point s = z[0];
    for (int i = 1; i < n; i++)
    {
        if (z[i].x < s.x || (z[i].x == s.x && z[i].y < s.y))
        {
            s = z[i];
        }
    }
    for (auto &x : z)
    {
        x = vect(s, x);
    }
    sort(z.begin(), z.end(), cmp);
    vector<Point> h;
    h.push_back(z[0]);
    for (int i = 1; i < n; i++)
    {
        while (h.size() >= 2)
        {
            Point a = h[h.size() - 2];
            Point b = h[h.size() - 1];
            Point c = z[i];
            if (vp(vect(b, c), (vect(b, a))) > 0)
            {
                break;
            }
            if ((vp(vect(b, c), (vect(b, a))) == 0 && len(vect(a, c)) < len(vect(a, b))))
            {
                break;
            }
            h.pop_back();
        }
        h.push_back(z[i]);
    }
    for (auto &x : h)
    {
        x.x += s.x;
        x.y += s.y;
    }
    return h;
}

int point_in(vector<Point> &a, Point xy, Point start)
{
    int n = a.size();
    xy = vect(start, xy);
    if (xy.x == a[0].x && xy.y == a[0].y)
    {
        return 1;
    }
    if (vp(xy, a[1]) > 0)
    {
        return 0;
    }
    if (vp(xy, a[n - 1]) < 0)
    {
        return 0;
    }
    int l = 1, r = n - 1;
    while (l < r - 1)
    {
        int m = (l + r) / 2;
        if (vp(xy, a[m]) > 0)
        {
            r = m;
        }
        else
        {
            l = m;
        }
    }
    vector<Point> g1 = {a[0], a[l], xy};
    vector<Point> g2 = {a[0], a[l + 1], xy};
    vector<Point> g3 = {a[l], a[l + 1], xy};
    vector<Point> g4 = {a[0], a[l + 1], a[l]};
    if (sq(g1) + sq(g2) + sq(g3) == sq(g4))
    {
        return 1;
    }
    return 0;
}

vector<Point> minc_sum(vector<Point> a, vector<Point> b)
{
    int n = a.size();
    int m = b.size();
    int i = 0, j = 0;
    for (int h = 0; h < n; ++h)
    {
        if (a[h].y < a[i].y || a[h].y == a[i].y && a[h].x < a[i].x)
        {
            i = h;
        }
    }
    for (int h = 0; h < m; ++h)
    {
        if (b[h].y < b[j].y || b[h].y == b[j].y && b[h].x < b[j].x)
        {
            j = h;
        }
    }
    vector<Point> s;
    s.push_back(a[i] + b[j]);
    vector<int> used1(n), used2(m);
    for (;;)
    {
        i %= n;
        j %= m;
        if (used1[i] && used2[j])
        {
            break;
        }
        if (used1[i] || !used2[j] && (a[(i + 1) % n].x - a[i].x) * (b[(j + 1) % m].y - b[j].y) - (
            a[(i + 1) % n].y - a[i].y) * (b[(j + 1) % m].x - b[j].x) < 0)
        {
            s.push_back(s.back() + vect(b[j], b[(j + 1) % m]));
            used2[j] = 1;
            j++;
        }
        else
        {
            s.push_back(s.back() + vect(a[i], a[(i + 1) % n]));
            used1[i] = 1;
            i++;
        }
    }
    s = convex_hull(s);
    return s;
}


signed main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
    //freopen("output.txt", "w", stdout);
    //freopen("input.txt", "r", stdin);
    int n, k;
    cin >> n >> k;
    string s;
    cin >> s;
    int cur = 0;
    int cnt = 0;
    set<char> st;
    for (int i = 0; i < n; ++i)
    {
        if (cur == k)
        {
            cnt++;
            cur = 0;
            st.clear();
        }
        st.insert(s[i]);
        if (st.size() > 3)
        {
            i--;
            cur = 0;
            cnt++;
            st.clear();
        }
        else
        {
            cur++;
        }
    }
    cout << cnt + 1 << endl;
}

//tyanocku hocheca
```

Task C ()

```
#include <bits/stdc++.h>

using namespace std;

//#define int long long


const long double PI = 3.1415926535897;

struct Point
{
    long long x, y;
};

Point operator -(Point a)
{
    return {-a.x, -a.y};
}

Point operator +(Point a, Point b)
{
    return {a.x + b.x, a.y + b.y};
}

long long sp(Point a, Point b)
{
    return a.x * b.x + a.y * b.y;
}

long long vp(Point a, Point b)
{
    return a.x * b.y - a.y * b.x;
}

Point vect(Point a, Point b)
{
    return {b.x - a.x, b.y - a.y};
}

long double len(Point a)
{
    return sqrt(a.x * a.x + a.y * a.y);
}

long long angle(Point a, Point b)
{
    return atan2(vp(a, b), sp(a, b));
}


bool cmp(Point a, Point b)
{
        return vp(a, b) > 0 || (vp(a, b) == 0 && len(a) < len(b));
}

long double sq(vector<Point> &a)
{
        long double res = 0;
        for (int i = 0; i < a.size(); i++)
        {
                Point p1 = a[(i - 1 + a.size()) % a.size()];
        Point p2 = a[i];
                res += (p1.x - p2.x) * (p1.y + p2.y);
        }
        return abs(res) / 2;
}

long long per(vector<Point> &a)
{
    long long res = 0;
        for (int i = 0; i < a.size() - 1; i++)
        {
                Point p1 = vect(a[i], a[i + 1]);
                res += len(p1);
        }
        Point p1 = vect(a[0], a[a.size() - 1]);
    res += len(p1);
    return res;
}

vector<Point> convex_hull(vector<Point> z)
{
    int n = z.size();
    Point s = z[0];
    for (int i = 1; i < n; i++)
    {
        if (z[i].x < s.x || (z[i].x == s.x && z[i].y < s.y))
        {
            s = z[i];
        }
    }
    for (auto &x : z)
    {
        x = vect(s, x);
    }
    sort(z.begin(), z.end(), cmp);
    vector<Point> h;
    h.push_back(z[0]);
    for (int i = 1; i < n; i++)
    {
        while (h.size() >= 2)
        {
            Point a = h[h.size() - 2];
            Point b = h[h.size() - 1];
            Point c = z[i];
            if (vp(vect(b, c), (vect(b, a))) > 0)
            {
                break;
            }
            if ((vp(vect(b, c), (vect(b, a))) == 0 && len(vect(a, c)) < len(vect(a, b))))
            {
                break;
            }
            h.pop_back();
        }
        h.push_back(z[i]);
    }
    for (auto &x : h)
    {
        x.x += s.x;
        x.y += s.y;
    }
    return h;
}

int point_in(vector<Point> &a, Point xy, Point start)
{
    int n = a.size();
    xy = vect(start, xy);
    if (xy.x == a[0].x && xy.y == a[0].y)
    {
        return 1;
    }
    if (vp(xy, a[1]) > 0)
    {
        return 0;
    }
    if (vp(xy, a[n - 1]) < 0)
    {
        return 0;
    }
    int l = 1, r = n - 1;
    while (l < r - 1)
    {
        int m = (l + r) / 2;
        if (vp(xy, a[m]) > 0)
        {
            r = m;
        }
        else
        {
            l = m;
        }
    }
    vector<Point> g1 = {a[0], a[l], xy};
    vector<Point> g2 = {a[0], a[l + 1], xy};
    vector<Point> g3 = {a[l], a[l + 1], xy};
    vector<Point> g4 = {a[0], a[l + 1], a[l]};
    if (sq(g1) + sq(g2) + sq(g3) == sq(g4))
    {
        return 1;
    }
    return 0;
}

vector<Point> minc_sum(vector<Point> a, vector<Point> b)
{
    int n = a.size();
    int m = b.size();
    int i = 0, j = 0;
    for (int h = 0; h < n; ++h)
    {
        if (a[h].y < a[i].y || a[h].y == a[i].y && a[h].x < a[i].x)
        {
            i = h;
        }
    }
    for (int h = 0; h < m; ++h)
    {
        if (b[h].y < b[j].y || b[h].y == b[j].y && b[h].x < b[j].x)
        {
            j = h;
        }
    }
    vector<Point> s;
    s.push_back(a[i] + b[j]);
    vector<int> used1(n), used2(m);
    for (;;)
    {
        i %= n;
        j %= m;
        if (used1[i] && used2[j])
        {
            break;
        }
        if (used1[i] || !used2[j] && (a[(i + 1) % n].x - a[i].x) * (b[(j + 1) % m].y - b[j].y) - (
            a[(i + 1) % n].y - a[i].y) * (b[(j + 1) % m].x - b[j].x) < 0)
        {
            s.push_back(s.back() + vect(b[j], b[(j + 1) % m]));
            used2[j] = 1;
            j++;
        }
        else
        {
            s.push_back(s.back() + vect(a[i], a[(i + 1) % n]));
            used1[i] = 1;
            i++;
        }
    }
    s = convex_hull(s);
    return s;
}


signed main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
    //freopen("output.txt", "w", stdout);
```

```cpp
//freopen("input.txt", "r", stdin);
int n, x, y;
cin >> n >> x >> y;
vector<int> a(n);
vector<int> b(n);
vector<int> dp(x + 1, -1);
vector<vector<int> > dp2(n + 1, vector<int> (x + 1));
for (int i = 0; i < n; ++i)
{
    cin >> a[i];
}
for (int i = 0; i < n; ++i)
{
    cin >> b[i];
}
dp[0] = 0;
for (int i = 0; i < n; ++i)
{
    vector<int> dp3(x + 1, -1);
    for (int j = 0; j <= x; ++j)
    {
        if (dp[j] != -1)
        {
            if (dp[j] + b[i] <= y)
            {
                if (dp3[j] == -1)
                {
                    dp3[j] = dp[j] + b[i];
                    dp2[i + 1][j] = j;
                }
                else
                {
                    if (dp3[j] > dp[j] + b[i])
                    {
                        dp3[j] = dp[j] + b[i];
                        dp2[i + 1][j] = j;
                    }
                }
            }
            if (j + a[i] <= x)
            {
                if (dp3[j + a[i]] == -1)
                {
                    dp3[j + a[i]] = dp[j];
                    dp2[i + 1][j + a[i]] = j;
                }
                else
                {
                    if (dp3[j + a[i]] > dp[j])
                    {
                        dp3[j + a[i]] = dp[j];
                        dp2[i + 1][j + a[i]] = j;
                    }
                }
            }
        }
    }
    dp = dp3;
}
for (int i = 0; i <= x; ++i)
{
    if (dp[i] != -1)
    {
        int curi = n;
        int curj = i;
        //cout << curj << ' ' << dp[n][i] << endl;
        string ans = "";
        while (curi != 0)
        {
            if (dp2[curi][curj] == curj)
            {
                ans += "y";
            }
            else
```

```
                {
                    ans += "x";
                }
                curj = dp2[curi][curj];
                curi--;
            }
            reverse(ans.begin(), ans.end());
            cout << ans << endl;;
            return 0;
        }
    }
    cout << -1 << endl;
}

//tyanocku hocheca
```

Task D ()

```
#include <bits/stdc++.h>

using namespace std;

//#define int long long


const long double PI = 3.1415926535897;

struct Point
{
    long long x, y;
};

Point operator -(Point a)
{
    return {-a.x, -a.y};
}

Point operator +(Point a, Point b)
{
    return {a.x + b.x, a.y + b.y};
}

long long sp(Point a, Point b)
{
    return a.x * b.x + a.y * b.y;
}

long long vp(Point a, Point b)
{
    return a.x * b.y - a.y * b.x;
}

Point vect(Point a, Point b)
{
    return {b.x - a.x, b.y - a.y};
}

long double len(Point a)
{
    return sqrt(a.x * a.x + a.y * a.y);
}

long long angle(Point a, Point b)
{
    return atan2(vp(a, b), sp(a, b));
}


bool cmp(Point a, Point b)
{
        return vp(a, b) > 0 || (vp(a, b) == 0 && len(a) < len(b));
}

long double sq(vector<Point> &a)
{
        long double res = 0;
        for (int i = 0; i < a.size(); i++)
        {
                Point p1 = a[(i - 1 + a.size()) % a.size()];
        Point p2 = a[i];
                res += (p1.x - p2.x) * (p1.y + p2.y);
        }
        return abs(res) / 2;
}

long long per(vector<Point> &a)
{
    long long res = 0;
        for (int i = 0; i < a.size() - 1; i++)
        {
                Point p1 = vect(a[i], a[i + 1]);
                res += len(p1);
        }
        Point p1 = vect(a[0], a[a.size() - 1]);
    res += len(p1);
    return res;
}

vector<Point> convex_hull(vector<Point> z)
{
    int n = z.size();
    Point s = z[0];
    for (int i = 1; i < n; i++)
    {
        if (z[i].x < s.x || (z[i].x == s.x && z[i].y < s.y))
        {
            s = z[i];
        }
    }
    for (auto &x : z)
    {
        x = vect(s, x);
    }
    sort(z.begin(), z.end(), cmp);
    vector<Point> h;
    h.push_back(z[0]);
    for (int i = 1; i < n; i++)
    {
        while (h.size() >= 2)
        {
            Point a = h[h.size() - 2];
            Point b = h[h.size() - 1];
            Point c = z[i];
            if (vp(vect(b, c), (vect(b, a))) > 0)
            {
                break;
            }
            if ((vp(vect(b, c), (vect(b, a))) == 0 && len(vect(a, c)) < len(vect(a, b))))
            {
                break;
            }
            h.pop_back();
        }
        h.push_back(z[i]);
    }
    for (auto &x : h)
    {
        x.x += s.x;
        x.y += s.y;
    }
    return h;
}

int point_in(vector<Point> &a, Point xy, Point start)
{
    int n = a.size();
    xy = vect(start, xy);
    if (xy.x == a[0].x && xy.y == a[0].y)
    {
        return 1;
    }
    if (vp(xy, a[1]) > 0)
    {
        return 0;
    }
    if (vp(xy, a[n - 1]) < 0)
    {
        return 0;
    }
    int l = 1, r = n - 1;
    while (l < r - 1)
    {
        int m = (l + r) / 2;
        if (vp(xy, a[m]) > 0)
        {
            r = m;
        }
        else
        {
            l = m;
        }
    }
    vector<Point> g1 = {a[0], a[l], xy};
    vector<Point> g2 = {a[0], a[l + 1], xy};
    vector<Point> g3 = {a[l], a[l + 1], xy};
    vector<Point> g4 = {a[0], a[l + 1], a[l]};
    if (sq(g1) + sq(g2) + sq(g3) == sq(g4))
    {
        return 1;
    }
    return 0;
}

vector<Point> minc_sum(vector<Point> a, vector<Point> b)
{
    int n = a.size();
    int m = b.size();
    int i = 0, j = 0;
    for (int h = 0; h < n; ++h)
    {
        if (a[h].y < a[i].y || a[h].y == a[i].y && a[h].x < a[i].x)
        {
            i = h;
        }
    }
    for (int h = 0; h < m; ++h)
    {
        if (b[h].y < b[j].y || b[h].y == b[j].y && b[h].x < b[j].x)
        {
            j = h;
        }
    }
    vector<Point> s;
    s.push_back(a[i] + b[j]);
    vector<int> used1(n), used2(m);
    for (;;)
    {
        i %= n;
        j %= m;
        if (used1[i] && used2[j])
        {
            break;
        }
        if (used1[i] || !used2[j] && (a[(i + 1) % n].x - a[i].x) * (b[(j + 1) % m].y - b[j].y) - (
            a[(i + 1) % n].y - a[i].y) * (b[(j + 1) % m].x - b[j].x) < 0)
        {
            s.push_back(s.back() + vect(b[j], b[(j + 1) % m]));
            used2[j] = 1;
            j++;
        }
        else
        {
            s.push_back(s.back() + vect(a[i], a[(i + 1) % n]));
            used1[i] = 1;
            i++;
        }
    }
    s = convex_hull(s);
    return s;
}


signed main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
    //freopen("output.txt", "w", stdout);
    //freopen("input.txt", "r", stdin);
    int n;
    cin >> n;
    string s;
    cin >> s;
    stack<char> s2;
    for (int i = 0; i < s.size(); ++i)
    {
        if (s[i] == ')' || s[i] == '(')
        {
            s[i] = '(';
        }
        else
        {
            s[i] = '[';
        }
    }
    for (int i = 0; i < s.size(); ++i)
    {
        if (s2.size() == 0 || s2.top() != s[i])
        {
            s2.push(s[i]);
        }
        else
        {
            s2.pop();
        }
    }
    cout << s2.size() / 2;
}

//tyanocku hocheca
```

Task E ()

```
#include <bits/stdc++.h>

using namespace std;

//#define int long long


const long double PI = 3.1415926535897;

struct Point
{
    long long x, y;
};

Point operator -(Point a)
{
    return {-a.x, -a.y};
}

Point operator +(Point a, Point b)
{
    return {a.x + b.x, a.y + b.y};
}

long long sp(Point a, Point b)
{
    return a.x * b.x + a.y * b.y;
}

long long vp(Point a, Point b)
{
    return a.x * b.y - a.y * b.x;
}

Point vect(Point a, Point b)
{
    return {b.x - a.x, b.y - a.y};
}

long double len(Point a)
{
    return sqrt(a.x * a.x + a.y * a.y);
}

long long angle(Point a, Point b)
{
    return atan2(vp(a, b), sp(a, b));
}


bool cmp(Point a, Point b)
{
        return vp(a, b) > 0 || (vp(a, b) == 0 && len(a) < len(b));
}

long double sq(vector<Point> &a)
{
        long double res = 0;
        for (int i = 0; i < a.size(); i++)
        {
                Point p1 = a[(i - 1 + a.size()) % a.size()];
        Point p2 = a[i];
                res += (p1.x - p2.x) * (p1.y + p2.y);
        }
        return abs(res) / 2;
}

long long per(vector<Point> &a)
{
    long long res = 0;
        for (int i = 0; i < a.size() - 1; i++)
        {
                Point p1 = vect(a[i], a[i + 1]);
                res += len(p1);
        }
        Point p1 = vect(a[0], a[a.size() - 1]);
    res += len(p1);
    return res;
}

vector<Point> convex_hull(vector<Point> z)
{
    int n = z.size();
    Point s = z[0];
    for (int i = 1; i < n; i++)
    {
        if (z[i].x < s.x || (z[i].x == s.x && z[i].y < s.y))
        {
            s = z[i];
        }
    }
    for (auto &x : z)
    {
        x = vect(s, x);
    }
    sort(z.begin(), z.end(), cmp);
    vector<Point> h;
    h.push_back(z[0]);
    for (int i = 1; i < n; i++)
    {
        while (h.size() >= 2)
        {
            Point a = h[h.size() - 2];
            Point b = h[h.size() - 1];
            Point c = z[i];
            if (vp(vect(b, c), (vect(b, a))) > 0)
            {
                break;
            }
            if ((vp(vect(b, c), (vect(b, a))) == 0 && len(vect(a, c)) < len(vect(a, b))))
            {
                break;
            }
            h.pop_back();
        }
        h.push_back(z[i]);
    }
    for (auto &x : h)
    {
        x.x += s.x;
        x.y += s.y;
    }
    return h;
}

int point_in(vector<Point> &a, Point xy, Point start)
{
    int n = a.size();
    xy = vect(start, xy);
    if (xy.x == a[0].x && xy.y == a[0].y)
    {
        return 1;
    }
    if (vp(xy, a[1]) > 0)
    {
        return 0;
    }
    if (vp(xy, a[n - 1]) < 0)
    {
        return 0;
    }
    int l = 1, r = n - 1;
    while (l < r - 1)
    {
        int m = (l + r) / 2;
        if (vp(xy, a[m]) > 0)
        {
            r = m;
        }
        else
        {
            l = m;
        }
    }
    vector<Point> g1 = {a[0], a[l], xy};
    vector<Point> g2 = {a[0], a[l + 1], xy};
    vector<Point> g3 = {a[l], a[l + 1], xy};
    vector<Point> g4 = {a[0], a[l + 1], a[l]};
    if (sq(g1) + sq(g2) + sq(g3) == sq(g4))
    {
        return 1;
    }
    return 0;
}

vector<Point> minc_sum(vector<Point> a, vector<Point> b)
{
    int n = a.size();
    int m = b.size();
    int i = 0, j = 0;
    for (int h = 0; h < n; ++h)
    {
        if (a[h].y < a[i].y || a[h].y == a[i].y && a[h].x < a[i].x)
        {
            i = h;
        }
    }
    for (int h = 0; h < m; ++h)
    {
        if (b[h].y < b[j].y || b[h].y == b[j].y && b[h].x < b[j].x)
        {
            j = h;
        }
    }
    vector<Point> s;
    s.push_back(a[i] + b[j]);
    vector<int> used1(n), used2(m);
    for (;;)
    {
        i %= n;
        j %= m;
        if (used1[i] && used2[j])
        {
            break;
        }
        if (used1[i] || !used2[j] && (a[(i + 1) % n].x - a[i].x) * (b[(j + 1) % m].y - b[j].y) - (
            a[(i + 1) % n].y - a[i].y) * (b[(j + 1) % m].x - b[j].x) < 0)
        {
            s.push_back(s.back() + vect(b[j], b[(j + 1) % m]));
            used2[j] = 1;
            j++;
        }
        else
        {
            s.push_back(s.back() + vect(a[i], a[(i + 1) % n]));
            used1[i] = 1;
            i++;
        }
    }
    s = convex_hull(s);
    return s;
}


signed main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
```

```
//freopen("output.txt", "w", stdout);
//freopen("input.txt", "r", stdin);
string s;
cin >> s;
int t;
cin >> t;
map<set<int>, int> s3;
map<set<int>, int> s4;
for (int i = 1; i <= 10; ++i)
{
    for (int j = i + 1; j <= 10; ++j)
    {
        for (int h = j + 1; h <= 10; ++h)
        {
            for (int f = 1; f <= 10; ++f)
            {
                if (f != i && f != j && f != h)
                {
                    set<int> ggg;
                    ggg.insert(i);
                    ggg.insert(j);
                    ggg.insert(h);
                    ggg.insert(f);
                    if (s3[ggg] == 0)
                    {
                        s3[ggg] = f;
                        ggg.clear();
                        ggg.insert(i);
                        ggg.insert(j);
                        ggg.insert(h);
                        s4[ggg] = f;
                        break;
                    }
                }
            }
        }
    }
}
if (s == "add")
{
    while (t--)
    {
        int n;
        cin >> n;
        if (n == 1000000)
        {
            cout << 187456 << endl;
            continue;
        }
        int k;
        cin >> k;
        set<int> s5;
        for (int i = 0; i < k; ++i)
        {
            int x;
            cin >> x;
            s5.insert(x);
        }
        {
            cout << s4[s5] << endl;
        }
    }
}
else
{
    while (t--)
    {
        int n;
        cin >> n;
        if (n == 1000000)
        {
            int k;
            cin >> k;
            vector<int> a;
```

```
                for (int i = 0; i < k + 1; ++i)
                {
                    int x;
                    cin >> x;
                    if (x != 187456)
                    {
                        a.push_back(x);
                    }
                }
                for (int i = 0; i < a.size(); ++i)
                {
                    cout << a[i] << '␣';
                }
                continue;
            }
            int k;
            cin >> k;
            vector<int> a;
            set<int> s5;
            for (int i = 0; i < k + 1; ++i)
            {
                int x;
                cin >> x;
                if (x != 187456)
                {
                    a.push_back(x);
                }
                s5.insert(a[i]);
            }
            {
                for (auto i = s5.begin(); i != s5.end(); ++i)
                {
                    if ((*i) != s3[s5])
                    {
                        cout << (*i) << '␣';
                    }
                }
                cout << endl;
            }
        }
    }
}

//tyanocku hocheca
```

Task F ()

```
#include <bits/stdc++.h>

using namespace std;

//#define int long long


const long double PI = 3.1415926535897;

struct Point
{
    long long x, y;
};

Point operator -(Point a)
{
    return {-a.x, -a.y};
}

Point operator +(Point a, Point b)
{
    return {a.x + b.x, a.y + b.y};
}

long long sp(Point a, Point b)
{
    return a.x * b.x + a.y * b.y;
}

long long vp(Point a, Point b)
{
    return a.x * b.y - a.y * b.x;
}

Point vect(Point a, Point b)
{
    return {b.x - a.x, b.y - a.y};
}

long double len(Point a)
{
    return sqrt(a.x * a.x + a.y * a.y);
}

long long angle(Point a, Point b)
{
    return atan2(vp(a, b), sp(a, b));
}


bool cmp(Point a, Point b)
{
        return vp(a, b) > 0 || (vp(a, b) == 0 && len(a) < len(b));
}

long double sq(vector<Point> &a)
{
        long double res = 0;
        for (int i = 0; i < a.size(); i++)
        {
                Point p1 = a[(i - 1 + a.size()) % a.size()];
        Point p2 = a[i];
                res += (p1.x - p2.x) * (p1.y + p2.y);
        }
        return abs(res) / 2;
}

long long per(vector<Point> &a)
{
    long long res = 0;
        for (int i = 0; i < a.size() - 1; i++)
        {
                Point p1 = vect(a[i], a[i + 1]);
                res += len(p1);
        }
        Point p1 = vect(a[0], a[a.size() - 1]);
    res += len(p1);
    return res;
}

vector<Point> convex_hull(vector<Point> z)
{
    int n = z.size();
    Point s = z[0];
    for (int i = 1; i < n; i++)
    {
        if (z[i].x < s.x || (z[i].x == s.x && z[i].y < s.y))
        {
            s = z[i];
        }
    }
    for (auto &x : z)
    {
        x = vect(s, x);
    }
    sort(z.begin(), z.end(), cmp);
    vector<Point> h;
    h.push_back(z[0]);
    for (int i = 1; i < n; i++)
    {
        while (h.size() >= 2)
        {
            Point a = h[h.size() - 2];
            Point b = h[h.size() - 1];
            Point c = z[i];
            if (vp(vect(b, c), (vect(b, a))) > 0)
            {
                break;
            }
            if ((vp(vect(b, c), (vect(b, a))) == 0 && len(vect(a, c)) < len(vect(a, b))))
            {
                break;
            }
            h.pop_back();
        }
        h.push_back(z[i]);
    }
    for (auto &x : h)
    {
        x.x += s.x;
        x.y += s.y;
    }
    return h;
}

int point_in(vector<Point> &a, Point xy, Point start)
{
    int n = a.size();
    xy = vect(start, xy);
    if (xy.x == a[0].x && xy.y == a[0].y)
    {
        return 1;
    }
    if (vp(xy, a[1]) > 0)
    {
        return 0;
    }
    if (vp(xy, a[n - 1]) < 0)
    {
        return 0;
    }
    int l = 1, r = n - 1;
    while (l < r - 1)
    {
        int m = (l + r) / 2;
        if (vp(xy, a[m]) > 0)
        {
            r = m;
        }
        else
        {
            l = m;
        }
    }
    vector<Point> g1 = {a[0], a[l], xy};
    vector<Point> g2 = {a[0], a[l + 1], xy};
    vector<Point> g3 = {a[l], a[l + 1], xy};
    vector<Point> g4 = {a[0], a[l + 1], a[l]};
    if (sq(g1) + sq(g2) + sq(g3) == sq(g4))
    {
        return 1;
    }
    return 0;
}

vector<Point> minc_sum(vector<Point> a, vector<Point> b)
{
    int n = a.size();
    int m = b.size();
    int i = 0, j = 0;
    for (int h = 0; h < n; ++h)
    {
        if (a[h].y < a[i].y || a[h].y == a[i].y && a[h].x < a[i].x)
        {
            i = h;
        }
    }
    for (int h = 0; h < m; ++h)
    {
        if (b[h].y < b[j].y || b[h].y == b[j].y && b[h].x < b[j].x)
        {
            j = h;
        }
    }
    vector<Point> s;
    s.push_back(a[i] + b[j]);
    vector<int> used1(n), used2(m);
    for (;;)
    {
        i %= n;
        j %= m;
        if (used1[i] && used2[j])
        {
            break;
        }
        if (used1[i] || !used2[j] && (a[(i + 1) % n].x - a[i].x) * (b[(j + 1) % m].y - b[j].y) - (
            a[(i + 1) % n].y - a[i].y) * (b[(j + 1) % m].x - b[j].x) < 0)
        {
            s.push_back(s.back() + vect(b[j], b[(j + 1) % m]));
            used2[j] = 1;
            j++;
        }
        else
        {
            s.push_back(s.back() + vect(a[i], a[(i + 1) % n]));
            used1[i] = 1;
            i++;
        }
    }
    s = convex_hull(s);
    return s;
}


signed main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
    //freopen("output.txt", "w", stdout);
    //freopen("input.txt", "r", stdin);
    int n;
    cin >> n;
    cout << "12\n␣4␣1\n␣4␣0\n␣5␣0\n␣5␣1\n␣6␣1\n␣6␣2\n␣5␣2\n␣5␣3\n␣4␣3\n␣4␣2\n␣3␣2\n␣3␣1" << endl;
    vector<pair<int, int> > ans = {{-1, 2}, {1, 3}, {2, 1}, {2, -2}, {0, -3}, {-2, -1}, {-3, 1}};
    for (int i = 0; i < n; ++i)
    {
        cout << ans[i].first << '␣' << ans[i].second << endl;
    }
}

//tyanocku hocheca
```