Олимпиада СПбГУ по информатике 2020/21 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	55	0	455

Task A ()

```
#include <iostream>
#include <string>
#include <math.h>
#include <queue>
#include <assert.h>
#include <set>
#include <bitset>
#include <map>
#include <unordered_map>
#include <random>
#include <chrono>
#include <algorithm>
#include <iomanip>
#include <unordered_set>
#include <complex>
using namespace std;

#define ll long long
#define vii vector<int>
#define vll vector<ll>
#define pii pair<int, int>
#define pll pair<ll, ll>
#define pdd pair<double, double>
#define pbb pair<bool, bool>
#define pb push_back
#define f first
#define s second
#define vpii vector<pii >
#define ld long double
#define vbb vector<bool>
#define vdd vector<double>
#define vpll vector<pll >
#define mp make_pair

#define M_PI (ld)(3.14159265358979323846)


//#define _FORTIFY_SOURCE 0
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,tune=native")
//#pragma GCC target("avx")
//#pragma GCC optimize(3)
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("inline")
//#pragma GCC optimize("-fgcse")
//#pragma GCC optimize("-fgcse-lm")
//#pragma GCC optimize("-fipa-sra")
//#pragma GCC optimize("-ftree-pre")
//#pragma GCC optimize("-ftree-vrp")
//#pragma GCC optimize("-fpeephole2")
//#pragma GCC optimize("-ffast-math")
//#pragma GCC optimize("-fsched-spec")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC optimize("-falign-jumps")
```

```
//#pragma GCC optimize("-falign-loops")
//#pragma GCC optimize("-falign-labels")
//#pragma GCC optimize("-fdevirtualize")
//#pragma GCC optimize("-fcaller-saves")
//#pragma GCC optimize("-fcrossjumping")
//#pragma GCC optimize("-fthread-jumps")
//#pragma GCC optimize("-funroll-loops")
//#pragma GCC optimize("-fwhole-program")
//#pragma GCC optimize("-freorder-blocks")
//#pragma GCC optimize("-fschedule-insns")
//#pragma GCC optimize("inline-functions")
//#pragma GCC optimize("-ftree-tail-merge")
//#pragma GCC optimize("-fschedule-insns2")
//#pragma GCC optimize("-fstrict-aliasing")
//#pragma GCC optimize("-fstrict-overflow")
//#pragma GCC optimize("-falign-functions")
//#pragma GCC optimize("-fcse-skip-blocks")
//#pragma GCC optimize("-fcse-follow-jumps")
//#pragma GCC optimize("-fsched-interblock")
//#pragma GCC optimize("-fpartial-inlining")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("-freorder-functions")
//#pragma GCC optimize("-findirect-inlining")
//#pragma GCC optimize("-fhoist-adjacent-loads")
//#pragma GCC optimize("-frerun-cse-after-loop")
//#pragma GCC optimize("inline-small-functions")
//#pragma GCC optimize("-finline-small-functions")
//#pragma GCC optimize("-ftree-switch-conversion")
//#pragma GCC optimize("-foptimize-sibling-calls")
//#pragma GCC optimize("-fexpensive-optimizations")
//#pragma GCC optimize("-funsafe-loop-optimizations")
//#pragma GCC optimize("inline-functions-called-once")
//#pragma GCC optimize("-fdelete-null-pointer-checks")

//#define cerr cerr

#if defined(cerr) //|| !defined(LOCAL)
#define cerr cerr1
template<typename T>
struct vector1 : public vector<T>{
        using vector<T>::vector;
        const T& operator[](size_t x) const{
                return (*this).at(x);
        }
        T& operator[](size_t x){
                return (*this).at(x);
        }
};
#define vector vector1

#define endl '\n'
struct shitClass
{
};

template<class T>
shitClass& operator<<(shitClass& c1, const T& )
{
        return c1;
};

shitClass cerr;
#endif

#define endl '\n'

#define MOD (ll)(1e9 + 7)//21)
#define MOD2 (ll)(1e9 + 33)

#define inf ((ll)(1e9) + 7)
#define eps (long double)(1e-6)

ostream &operator<<(ostream &os, const vll &a)
{
```

```
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const vii &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const pll &a)
{
        os << a.first << " " << a.s;
        return os;
}

ostream &operator<<(ostream &os, const vpll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i].first << " " << a[i].s << endl;
        return os;
}

ostream &operator<<(ostream &os, const vector<complex<double> > &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

istream &operator>>(istream &is, pll &a)
{
        is >> a.first >> a.s;
        return is;
}

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

ll myRnd(ll l, ll r)
{
        std::uniform_int_distribution<ll> range(l, r);
        return range(rng);
}

int get(char c1, char c2)
{
        if((c1 == '(' || c1 == ')') && (c2 == '(' || c2 == ')'))
                return 0;
        else if((c1 == '[' || c1 == ']') && (c2 == '[' || c2 == ']'))
                return 0;
        else
                return 1;
}

int main()
{
        ios_base::sync_with_stdio(0); cin.tie(0); cout.tie(0);
        cout.precision(30);

#ifdef LOCAL
        freopen("input.txt", "r", stdin);
        freopen("output.txt", "w", stdout);
#endif

        int k;
        cin >> k;

        ll fromPrev = 0;
        for(int i = 1; i < k; i++)
        {
                fromPrev += i;
                fromPrev /= 10;
```

```
        }
        cout << (k + fromPrev) % 10 << endl;
        return 0;
}
```

Task B ()

```
#include <iostream>
#include <string>
#include <math.h>
#include <queue>
#include <assert.h>
#include <set>
#include <bitset>
#include <map>
#include <unordered_map>
#include <random>
#include <chrono>
#include <algorithm>
#include <iomanip>
#include <unordered_set>
#include <complex>
using namespace std;

#define ll long long
#define vii vector<int>
#define vll vector<ll>
#define pii pair<int, int>
#define pll pair<ll, ll>
#define pdd pair<double, double>
#define pbb pair<bool, bool>
#define pb push_back
#define f first
#define s second
#define vpii vector<pii >
#define ld long double
#define vbb vector<bool>
#define vdd vector<double>
#define vpll vector<pll >
#define mp make_pair

#define M_PI (ld)(3.14159265358979323846)


//#define _FORTIFY_SOURCE 0
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,tune=native")
//#pragma GCC target("avx")
//#pragma GCC optimize(3)
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("inline")
//#pragma GCC optimize("-fgcse")
//#pragma GCC optimize("-fgcse-lm")
//#pragma GCC optimize("-fipa-sra")
//#pragma GCC optimize("-ftree-pre")
//#pragma GCC optimize("-ftree-vrp")
//#pragma GCC optimize("-fpeephole2")
//#pragma GCC optimize("-ffast-math")
//#pragma GCC optimize("-fsched-spec")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC optimize("-falign-jumps")
//#pragma GCC optimize("-falign-loops")
//#pragma GCC optimize("-falign-labels")
//#pragma GCC optimize("-fdevirtualize")
//#pragma GCC optimize("-fcaller-saves")
//#pragma GCC optimize("-fcrossjumping")
//#pragma GCC optimize("-fthread-jumps")
//#pragma GCC optimize("-funroll-loops")
//#pragma GCC optimize("-fwhole-program")
//#pragma GCC optimize("-freorder-blocks")
//#pragma GCC optimize("-fschedule-insns")
//#pragma GCC optimize("inline-functions")
//#pragma GCC optimize("-ftree-tail-merge")
//#pragma GCC optimize("-fschedule-insns2")
//#pragma GCC optimize("-fstrict-aliasing")
//#pragma GCC optimize("-fstrict-overflow")
//#pragma GCC optimize("-falign-functions")
```

```
//#pragma GCC optimize("-fcse-skip-blocks")
//#pragma GCC optimize("-fcse-follow-jumps")
//#pragma GCC optimize("-fsched-interblock")
//#pragma GCC optimize("-fpartial-inlining")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("-freorder-functions")
//#pragma GCC optimize("-findirect-inlining")
//#pragma GCC optimize("-fhoist-adjacent-loads")
//#pragma GCC optimize("-frerun-cse-after-loop")
//#pragma GCC optimize("inline-small-functions")
//#pragma GCC optimize("-finline-small-functions")
//#pragma GCC optimize("-ftree-switch-conversion")
//#pragma GCC optimize("-foptimize-sibling-calls")
//#pragma GCC optimize("-fexpensive-optimizations")
//#pragma GCC optimize("-funsafe-loop-optimizations")
//#pragma GCC optimize("inline-functions-called-once")
//#pragma GCC optimize("-fdelete-null-pointer-checks")

//#define cerr cerr

#if defined(cerr) //|| !defined(LOCAL)
#define cerr cerr1
template<typename T>
struct vector1 : public vector<T>{
        using vector<T>::vector;
        const T& operator[](size_t x) const{
                return (*this).at(x);
        }
        T& operator[](size_t x){
                return (*this).at(x);
        }
};
#define vector vector1

#define endl '\n'
struct shitClass
{
};

template<class T>
shitClass& operator<<(shitClass& c1, const T& )
{
        return c1;
};

shitClass cerr;
#endif

#define endl '\n'

#define MOD (ll)(1e9 + 7)//21)
#define MOD2 (ll)(1e9 + 33)

#define inf ((ll)(1e18) + 7)
#define eps (long double)(1e-6)

ostream &operator<<(ostream &os, const vll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const vii &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const pll &a)
{
        os << a.first << " " << a.s;
        return os;
```

```
}

ostream &operator<<(ostream &os, const vpll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i].first << " " << a[i].s << endl;
        return os;
}

ostream &operator<<(ostream &os, const vector<complex<double> > &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

istream &operator>>(istream &is, pll &a)
{
        is >> a.first >> a.s;
        return is;
}

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

ll myRnd(ll l, ll r)
{
        std::uniform_int_distribution<ll> range(l, r);
        return range(rng);
}

int main()
{
        ios_base::sync_with_stdio(0); cin.tie(0); cout.tie(0);
        cout.precision(30);

#ifdef LOCAL
        freopen("input.txt", "r", stdin);
        freopen("output.txt", "w", stdout);
#endif

        ll n, k;
        cin >> n >> k;

        string st;
        cin >> st;

        ll answ = 1;
        vii let(26);
        let[st[0] - 'a'] = 1;
        int cnt = 1;
        ll prev = 0;

        for(int i = 1; i < n; i++)
        {
                //cerr << "i " << i << " cnt " << cnt << endl;
                if((cnt == 3 && let[st[i] - 'a'] == 0) || i - prev == k)
                {
                        answ++;
                        cnt = 0;
                        let.resize(0);
                        let.resize(26, 0);
                        prev = i;
                }

                if(let[st[i] - 'a'] == 0)
                        cnt++;

                let[st[i] - 'a']++;
        }
        cout << answ << endl;
        return 0;
}
```

Task C ()

```
#include <iostream>
#include <string>
#include <math.h>
#include <queue>
#include <assert.h>
#include <set>
#include <bitset>
#include <map>
#include <unordered_map>
#include <random>
#include <chrono>
#include <algorithm>
#include <iomanip>
#include <unordered_set>
#include <complex>
using namespace std;

#define ll long long
#define vii vector<int>
#define vll vector<ll>
#define pii pair<int, int>
#define pll pair<ll, ll>
#define pdd pair<double, double>
#define pbb pair<bool, bool>
#define pb push_back
#define f first
#define s second
#define vpii vector<pii >
#define ld long double
#define vbb vector<bool>
#define vdd vector<double>
#define vpll vector<pll >
#define mp make_pair

#define M_PI (ld)(3.14159265358979323846)


//#define _FORTIFY_SOURCE 0
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,tune=native")
//#pragma GCC target("avx")
//#pragma GCC optimize(3)
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("inline")
//#pragma GCC optimize("-fgcse")
//#pragma GCC optimize("-fgcse-lm")
//#pragma GCC optimize("-fipa-sra")
//#pragma GCC optimize("-ftree-pre")
//#pragma GCC optimize("-ftree-vrp")
//#pragma GCC optimize("-fpeephole2")
//#pragma GCC optimize("-ffast-math")
//#pragma GCC optimize("-fsched-spec")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC optimize("-falign-jumps")
//#pragma GCC optimize("-falign-loops")
//#pragma GCC optimize("-falign-labels")
//#pragma GCC optimize("-fdevirtualize")
//#pragma GCC optimize("-fcaller-saves")
//#pragma GCC optimize("-fcrossjumping")
//#pragma GCC optimize("-fthread-jumps")
//#pragma GCC optimize("-funroll-loops")
//#pragma GCC optimize("-fwhole-program")
//#pragma GCC optimize("-freorder-blocks")
//#pragma GCC optimize("-fschedule-insns")
//#pragma GCC optimize("inline-functions")
//#pragma GCC optimize("-ftree-tail-merge")
//#pragma GCC optimize("-fschedule-insns2")
//#pragma GCC optimize("-fstrict-aliasing")
//#pragma GCC optimize("-fstrict-overflow")
//#pragma GCC optimize("-falign-functions")
```

```
//#pragma GCC optimize("-fcse-skip-blocks")
//#pragma GCC optimize("-fcse-follow-jumps")
//#pragma GCC optimize("-fsched-interblock")
//#pragma GCC optimize("-fpartial-inlining")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("-freorder-functions")
//#pragma GCC optimize("-findirect-inlining")
//#pragma GCC optimize("-fhoist-adjacent-loads")
//#pragma GCC optimize("-frerun-cse-after-loop")
//#pragma GCC optimize("inline-small-functions")
//#pragma GCC optimize("-finline-small-functions")
//#pragma GCC optimize("-ftree-switch-conversion")
//#pragma GCC optimize("-foptimize-sibling-calls")
//#pragma GCC optimize("-fexpensive-optimizations")
//#pragma GCC optimize("-funsafe-loop-optimizations")
//#pragma GCC optimize("inline-functions-called-once")
//#pragma GCC optimize("-fdelete-null-pointer-checks")

//#define cerr cerr

#if defined(cerr) //|| !defined(LOCAL)
#define cerr cerr1
template<typename T>
struct vector1 : public vector<T>{
        using vector<T>::vector;
        const T& operator[](size_t x) const{
                return (*this).at(x);
        }
        T& operator[](size_t x){
                return (*this).at(x);
        }
};
#define vector vector1

#define endl '\n'
struct shitClass
{
};

template<class T>
shitClass& operator<<(shitClass& c1, const T& )
{
        return c1;
};

shitClass cerr;
#endif

#define endl '\n'

#define MOD (ll)(1e9 + 7)//21)
#define MOD2 (ll)(1e9 + 33)

#define inf ((ll)(1e18) + 7)
#define eps (long double)(1e-6)

ostream &operator<<(ostream &os, const vll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const vii &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const pll &a)
{
        os << a.first << " " << a.s;
        return os;
```

```
}

ostream &operator<<(ostream &os, const vpll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i].first << " " << a[i].s << endl;
        return os;
}

ostream &operator<<(ostream &os, const vector<complex<double> > &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

istream &operator>>(istream &is, pll &a)
{
        is >> a.first >> a.s;
        return is;
}

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

ll myRnd(ll l, ll r)
{
        std::uniform_int_distribution<ll> range(l, r);
        return range(rng);
}

int pr[501][250001];

int main()
{
        ios_base::sync_with_stdio(0); cin.tie(0); cout.tie(0);
        cout.precision(30);

#ifdef LOCAL
        freopen("input.txt", "r", stdin);
        freopen("output.txt", "w", stdout);
#endif

        for(int i = 0; i < 501; i++)
        {
                for(int j = 0; j < 250001; j++)
                        pr[i][j] = -1;
        }
        int n, x, y;
        cin >> n >> x >> y;

        vpii a(n);
        for(int i = 0; i < n; i++)
                cin >> a[i].f;

        for(int i = 0; i < n; i++)
                cin >> a[i].s;

        vll dpPrev(x + 1, -1);
        dpPrev[0] = 0;

        vll dpCur(x + 1, -1);
//      vpll pr(x + 1, {-1, -1});

        for(int i = 0; i < n; i++)
        {
                for(int j = x; j >= 0; j--)
                {
                        dpCur[j] = max(dpCur[j], dpPrev[j]);
                        if(a[i].f > j || dpPrev[j - a[i].f] == -1)
                                continue;

                        if(dpPrev[j - a[i].f] + a[i].s > dpCur[j])
                        {
                                dpCur[j] = dpPrev[j - a[i].f] + a[i].s;
```

```
                                pr[i][j] = j - a[i].f;
                        }

                }
                dpPrev = dpCur;
                dpCur.resize(0);
                dpCur.resize(x + 1, -1);
        }

        ll mx = -1, mxId = -1;
        for(int i = 0; i <= x; i++)
        {
                if(dpPrev[i] > mx)
                {
                        mx = dpPrev[i];
                        mxId = i;
                }
        }

        ll sumY = 0;
        for(int i = 0; i < n; i++)
                sumY += a[i].s;

        if(sumY - mx > y)
        {
                cout << -1 << endl;
                return 0;
        }

        vbb isFir(n);
        ll curPrev = mxId;
        ll curStep = n;

        while(curPrev != 0)
        {
                while(pr[curStep][curPrev] == -1)
                        curStep--;

                isFir[curStep] = 1;
                curPrev = pr[curStep][curPrev];
                curStep--;
        }

        for(int i = 0; i < n; i++)
        {
                if(isFir[i])
                        cout << "x";
                else
                        cout << "y";
        }
        cout << endl;
        return 0;
}
```

Task D ()

```
#include <iostream>
#include <string>
#include <math.h>
#include <queue>
#include <assert.h>
#include <set>
#include <bitset>
#include <map>
#include <unordered_map>
#include <random>
#include <chrono>
#include <algorithm>
#include <iomanip>
#include <unordered_set>
#include <complex>
using namespace std;

#define ll long long
#define vii vector<int>
#define vll vector<ll>
#define pii pair<int, int>
#define pll pair<ll, ll>
#define pdd pair<double, double>
#define pbb pair<bool, bool>
#define pb push_back
#define f first
#define s second
#define vpii vector<pii >
#define ld long double
#define vbb vector<bool>
#define vdd vector<double>
#define vpll vector<pll >
#define mp make_pair

#define M_PI (ld)(3.14159265358979323846)


//#define _FORTIFY_SOURCE 0
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,tune=native")
//#pragma GCC target("avx")
//#pragma GCC optimize(3)
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("inline")
//#pragma GCC optimize("-fgcse")
//#pragma GCC optimize("-fgcse-lm")
//#pragma GCC optimize("-fipa-sra")
//#pragma GCC optimize("-ftree-pre")
//#pragma GCC optimize("-ftree-vrp")
//#pragma GCC optimize("-fpeephole2")
//#pragma GCC optimize("-ffast-math")
//#pragma GCC optimize("-fsched-spec")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC optimize("-falign-jumps")
//#pragma GCC optimize("-falign-loops")
//#pragma GCC optimize("-falign-labels")
//#pragma GCC optimize("-fdevirtualize")
//#pragma GCC optimize("-fcaller-saves")
//#pragma GCC optimize("-fcrossjumping")
//#pragma GCC optimize("-fthread-jumps")
//#pragma GCC optimize("-funroll-loops")
//#pragma GCC optimize("-fwhole-program")
//#pragma GCC optimize("-freorder-blocks")
//#pragma GCC optimize("-fschedule-insns")
//#pragma GCC optimize("inline-functions")
//#pragma GCC optimize("-ftree-tail-merge")
//#pragma GCC optimize("-fschedule-insns2")
//#pragma GCC optimize("-fstrict-aliasing")
//#pragma GCC optimize("-fstrict-overflow")
//#pragma GCC optimize("-falign-functions")
```

```
//#pragma GCC optimize("-fcse-skip-blocks")
//#pragma GCC optimize("-fcse-follow-jumps")
//#pragma GCC optimize("-fsched-interblock")
//#pragma GCC optimize("-fpartial-inlining")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("-freorder-functions")
//#pragma GCC optimize("-findirect-inlining")
//#pragma GCC optimize("-fhoist-adjacent-loads")
//#pragma GCC optimize("-frerun-cse-after-loop")
//#pragma GCC optimize("inline-small-functions")
//#pragma GCC optimize("-finline-small-functions")
//#pragma GCC optimize("-ftree-switch-conversion")
//#pragma GCC optimize("-foptimize-sibling-calls")
//#pragma GCC optimize("-fexpensive-optimizations")
//#pragma GCC optimize("-funsafe-loop-optimizations")
//#pragma GCC optimize("inline-functions-called-once")
//#pragma GCC optimize("-fdelete-null-pointer-checks")

//#define cerr cerr

#if defined(cerr) //|| !defined(LOCAL)
#define cerr cerr1
template<typename T>
struct vector1 : public vector<T>{
        using vector<T>::vector;
        const T& operator[](size_t x) const{
                return (*this).at(x);
        }
        T& operator[](size_t x){
                return (*this).at(x);
        }
};
#define vector vector1

#define endl '\n'
struct shitClass
{
};

template<class T>
shitClass& operator<<(shitClass& c1, const T& )
{
        return c1;
};

shitClass cerr;
#endif

#define endl '\n'

#define MOD (ll)(1e9 + 7)//21)
#define MOD2 (ll)(1e9 + 33)

#define inf ((ll)(1e9) + 7)
#define eps (long double)(1e-6)

ostream &operator<<(ostream &os, const vll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const vii &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const pll &a)
{
        os << a.first << " " << a.s;
        return os;
```

```
}

ostream &operator<<(ostream &os, const vpll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i].first << " " << a[i].s << endl;
        return os;
}

ostream &operator<<(ostream &os, const vector<complex<double> > &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

istream &operator>>(istream &is, pll &a)
{
        is >> a.first >> a.s;
        return is;
}

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

ll myRnd(ll l, ll r)
{
        std::uniform_int_distribution<ll> range(l, r);
        return range(rng);
}

int get(char c1, char c2)
{
        if((c1 == '(' || c1 == ')') && (c2 == '(' || c2 == ')'))
                return 0;
        else if((c1 == '[' || c1 == ']') && (c2 == '[' || c2 == ']'))
                return 0;
        else
                return 1;
}

int main()
{
        ios_base::sync_with_stdio(0); cin.tie(0); cout.tie(0);
        cout.precision(30);

#ifdef LOCAL
        freopen("input.txt", "r", stdin);
        freopen("output.txt", "w", stdout);
#endif

        ll n;
        cin >> n;
        n *= 2;

        string st;
        cin >> st;

        for(int i = 0; i < st.size(); i++)
        {
                if(st[i] == ')')
                        st[i] = '(';
                if(st[i] == ']')
                        st[i] = '[';
        }

        deque<char> dq;
        for(int i = 0; i < n; i++)
        {
                //if(!dq.empty())
                //      cerr << "dq.back() " << dq.back() << endl;

                if(!dq.empty() && dq.back() == st[i])
                {
                        dq.pop_back();
```

```
                continue;
            }
            dq.push_back(st[i]);
        }

        st = "";
        while(!dq.empty())
        {
            st += dq.front();
            dq.pop_front();
        }
        //cerr << "st " << st <<endl;
        n = st.size();

        cout << n / 2 << endl;
        return 0;

        if(n == 0)
        {
            cout << 0 << endl;
            return 0;
        }

        vector<vii > dp(n, vii(n, inf));

        for(int i = 0; i < n; i++)
        {
            if(i + 1 == n)
                continue;

            dp[i][i + 1] = get(st[i], st[i + 1]);
        }

        for(int len = 4; len <= n; len += 2)
        {
            for(int i = 0; i < n; i++)
            {
                int l = i;
                int r = i + len - 1;

                if(r >= n)
                    continue;

                for(int j = i + 2; j < len; j += 2)
                {
                    if(dp[l][j - 1] == inf || dp[j][r] == inf)
                        continue;

                    dp[l][r] = min(dp[l][r], dp[l][j - 1] + dp[j][r]);
                }
                dp[l][r] = min(dp[l][r], dp[l + 1][r - 1] + get(st[l], st[r]));
            }
        }
        cout << dp[0][n - 1] << endl;
        return 0;
}
```

Task E ()

```
#include <iostream>
#include <string>
#include <math.h>
#include <queue>
#include <assert.h>
#include <set>
#include <bitset>
#include <map>
#include <unordered_map>
#include <random>
#include <chrono>
#include <algorithm>
#include <iomanip>
#include <unordered_set>
#include <complex>
using namespace std;

#define ll long long
#define vii vector<int>
#define vll vector<ll>
#define pii pair<int, int>
#define pll pair<ll, ll>
#define pdd pair<double, double>
#define pbb pair<bool, bool>
#define pb push_back
#define f first
#define s second
#define vpii vector<pii >
#define ld long double
#define vbb vector<bool>
#define vdd vector<double>
#define vpll vector<pll >
#define mp make_pair

#define M_PI (ld)(3.14159265358979323846)


//#define _FORTIFY_SOURCE 0
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC target("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,tune=native")
//#pragma GCC target("avx")
//#pragma GCC optimize(3)
//#pragma GCC optimize("Ofast")
//#pragma GCC optimize("inline")
//#pragma GCC optimize("-fgcse")
//#pragma GCC optimize("-fgcse-lm")
//#pragma GCC optimize("-fipa-sra")
//#pragma GCC optimize("-ftree-pre")
//#pragma GCC optimize("-ftree-vrp")
//#pragma GCC optimize("-fpeephole2")
//#pragma GCC optimize("-ffast-math")
//#pragma GCC optimize("-fsched-spec")
//#pragma GCC optimize("unroll-loops")
//#pragma GCC optimize("-falign-jumps")
//#pragma GCC optimize("-falign-loops")
//#pragma GCC optimize("-falign-labels")
//#pragma GCC optimize("-fdevirtualize")
//#pragma GCC optimize("-fcaller-saves")
//#pragma GCC optimize("-fcrossjumping")
//#pragma GCC optimize("-fthread-jumps")
//#pragma GCC optimize("-funroll-loops")
//#pragma GCC optimize("-fwhole-program")
//#pragma GCC optimize("-freorder-blocks")
//#pragma GCC optimize("-fschedule-insns")
//#pragma GCC optimize("inline-functions")
//#pragma GCC optimize("-ftree-tail-merge")
//#pragma GCC optimize("-fschedule-insns2")
//#pragma GCC optimize("-fstrict-aliasing")
//#pragma GCC optimize("-fstrict-overflow")
//#pragma GCC optimize("-falign-functions")
```

```
//#pragma GCC optimize("-fcse-skip-blocks")
//#pragma GCC optimize("-fcse-follow-jumps")
//#pragma GCC optimize("-fsched-interblock")
//#pragma GCC optimize("-fpartial-inlining")
//#pragma GCC optimize("no-stack-protector")
//#pragma GCC optimize("-freorder-functions")
//#pragma GCC optimize("-findirect-inlining")
//#pragma GCC optimize("-fhoist-adjacent-loads")
//#pragma GCC optimize("-frerun-cse-after-loop")
//#pragma GCC optimize("inline-small-functions")
//#pragma GCC optimize("-finline-small-functions")
//#pragma GCC optimize("-ftree-switch-conversion")
//#pragma GCC optimize("-foptimize-sibling-calls")
//#pragma GCC optimize("-fexpensive-optimizations")
//#pragma GCC optimize("-funsafe-loop-optimizations")
//#pragma GCC optimize("inline-functions-called-once")
//#pragma GCC optimize("-fdelete-null-pointer-checks")

//#define cerr cerr

#if defined(cerr) //|| !defined(LOCAL)
#define cerr cerr1
template<typename T>
struct vector1 : public vector<T>{
        using vector<T>::vector;
        const T& operator[](size_t x) const{
                return (*this).at(x);
        }
        T& operator[](size_t x){
                return (*this).at(x);
        }
};
#define vector vector1

#define endl '\n'
struct shitClass
{
};

template<class T>
shitClass& operator<<(shitClass& c1, const T& )
{
        return c1;
};

shitClass cerr;
#endif

#define endl '\n'

#define MOD (ll)(1e9 + 7)//21)
#define MOD2 (ll)(1e9 + 33)

#define inf ((ll)(1e9) + 7)
#define eps (long double)(1e-6)

ostream &operator<<(ostream &os, const vll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const vii &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

ostream &operator<<(ostream &os, const pll &a)
{
        os << a.first << " " << a.s;
        return os;
```

```
}

ostream &operator<<(ostream &os, const vpll &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i].first << " " << a[i].s << endl;
        return os;
}

ostream &operator<<(ostream &os, const vector<complex<double> > &a)
{
        for (int i = 0; i < a.size(); i++)
                os << a[i] << " ";
        return os;
}

istream &operator>>(istream &is, pll &a)
{
        is >> a.first >> a.s;
        return is;
}

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

ll myRnd(ll l, ll r)
{
        std::uniform_int_distribution<ll> range(l, r);
        return range(rng);
}

vii groupTwo;

ll getMy(vll a)
{
        for(int i = 0; i < a.size(); i++)
        {
                for(int j = i + 1; j < a.size(); j++)
                {
                        for(int t = j + 1; t < a.size(); t++)
                        {
                                ll leftInd;
                                for(int k = 0; k < a.size(); k++)
                                {
                                        if(k != i && k != j && k != t)
                                        {
                                                leftInd = k;
                                                break;
                                        }
                                }

                                ll curMask = ((1 << a[i]) | (1 << a[j]) | (1 << a[t]));
                                if(groupTwo[curMask] == a[leftInd])
                                        return a[leftInd];
                        }
                }
        }
        return -1;
}

int main()
{
        ios_base::sync_with_stdio(0); cin.tie(0); cout.tie(0);
        cout.precision(30);

#ifdef LOCAL
        freopen("input.txt", "r", stdin);
        freopen("output.txt", "w", stdout);
#endif

        groupTwo.resize((1 << 10) + 1, -1);

        for(int i = 0; i < 10; i++)
        {
                for(int j = i + 1; j < 10; j++)
```

```
                {
                        for(int t = j + 1; t < 10; t++)
                        {
                                for(int k = 0; k < 10; k++)
                                {
                                        if(k == i || k == j || k == t)
                                                continue;

                                        ll my = getMy({i, j, t, k});
                                        if(my != -1)
                                                continue;
                                        ll curMask = ((1 << i) | (1 << j) | (1 << t));
                                        //cerr << "k " << k << " for " << i << " " << j << " " <<
                                            t << endl;
                                        groupTwo[curMask] = k;
                                        break;
                                }
                                ll curMask = ((1 << i) | (1 << j) | (1 << t));
                                //      if(groupTwo[curMask] == -1)
                                //      cerr << "WASTED " << i << " " << j << " " << t << endl;
                        }
                }
        }
        vll myWay;
        for(int i = 0; i < (1e5); i++)
                myWay.pb(i);

        for(ll i = 0; i < (1e5); i++)
                swap(myWay[i], myWay[(((i + (i + 252) * (i + 3423)) % 234 + i * 324 + (342445353 -
                     i) * 10 + 92847) % 93235 + (1ll << (i % 25)) + (i * 32423411) + 34223451) % (
                    ll)(1e5)]);
        //cerr << myWay[0] << endl;
//      for(int i = 0; i < myWay.size(); i++)
//              cout << myWay[i] << ", ";
//      return 0;
        string type;
        cin >> type;

        if(type == "add")
        {
                ll t;
                cin >> t;
                for(int counter = 0; counter < t; counter++)
                {
                        ll n, k;
                        cin >> n >> k;

                        vll a(k);
                        vbb used(1e6 + 1);
                        for(int i = 0; i < k; i++)
                        {
                                cin >> a[i];
                                a[i]--;
                                used[a[i]] = 1;
                        }

                        if(n == 1e6)
                        {
                                cout << 151954 << endl;
                        }
                        if(n == 10)
                        {
                                ll curMask = ((1 << a[0]) | (1 << a[1]) | (1 << a[2]));
                                cout << groupTwo[curMask] + 1 << endl;
                        }
                        if(n == 1e5)
                        {
                                int cur = 0;
                                while(used[myWay[cur]])
                                        cur++;

                                cout << myWay[cur] + 1 << endl;
                        }
                }
```

```
        }
        else
        {
                ll t;
                cin >> t;
                for(int counter = 0; counter < t; counter++)
                {
                        ll n, k;
                        cin >> n >> k;

                        vll a(k + 1);
                        vll copyA(k + 1);
                        vbb used(1e6 + 1);

                        for(int i = 0; i < k + 1; i++)
                        {
                                cin >> a[i];
                                copyA[i] = a[i] - 1;
                                used[copyA[i]] = 1;
                        }

                        ll cur;
                        if(n == 10)
                        {
                                cur = getMy(copyA) + 1;
                        }
                        if(n == 1e6)
                        {
                                cur = 151954;
                        }
                        if(n == 1e5)
                        {
                                ll cnt = 0;
                                while(cnt + 1 < myWay.size() && used[cnt + 1])
                                        cnt++;

                                cur = myWay[cnt] + 1;
                        }
                        for(int i = 0; i < a.size(); i++)
                        {
                                if(a[i] != cur)
                                        cout << a[i] << "␣";
                        }
                        cout << endl;
                }
        }
        return 0;
}
```

Task F ()