Олимпиада СПбГУ по информатике 2021/22 учебного года

A	B	C	D	E	F	Sum
100	100	100	100	58	0	458

Task A ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

using v = vector<long long>;
using vv  = vector<vector<long long>>;
using ll = long long;

int main(){
    int n;
    cin >> n;
    vv N(n, v(2));
    int k, delta = 0;
    for (int i = 0; i < n; i++){
        cin >> k;
        delta = 0;
        while (k % 10 == 0){
            k /= 10;
            delta++;
        }
        N[i][1] = k;
        cin >> N[i][0];
        N[i][0] += delta;
    }

    stable_sort(N.begin(), N.end());

    int i = 0;
    ll best = N[0][0];
    ll summ = N[0][1];
    for (i = 1; i < n; i++){
        if (N[i][0] > best){
            while (summ % 10 == 0 && best < N[i][0]){
                summ /= 10;
                best++;
            }
            if (best < N[i][0]){
                break;
            }
            else{
                summ += N[i][1];
            }
        }
        else if (N[i][0] == best){
            summ += N[i][1];
        }
    }

    if (i == n){
        while (summ % 10 == 0){
            summ /= 10;
            best++;
        }
    }
```

```
    cout << best;
    return 0;
}
```

Task B ()

```
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

using v = vector<long long>;
using vv  = vector<vector<long long>>;
using ll = long long;

int main(){
    int n;
    cin >> n;
    v K(n);
    for (int i = 0; i < n; i++){
        cin >> K[i];
    }

    string command = "Start";
    int k_counter = 0;

    if (n == 1 && K[0] == 0){
        cout << "Stop" << endl;
        return 0;
    }
    else if (n == 1){
        cout << "Flip and wait" << endl;
        k_counter++;
        while (k_counter < K[0]){
            cin >> command;
            cout << "Flip and wait" << endl;
            k_counter++;
        }
        cin >> command;
        cout << "Stop" << endl;
        return 0;
    }

    int beep_wait_count = 0;
    int now_index = n - 1;
    while (K[now_index] == 0){
        now_index--;
    }
    int ind = 0;
    int delta = 0;
    if (now_index >= 0){
        cout << "Flip and wait" << endl;
    }
    else{
        cout << "Stop" << endl;
    }

    while (now_index >= 0){
        cin >> command;
        if (command == "Burn" || command == "Tired" || command == "Fail"){
            return 0;
        }
        delta += (command.length() - 3) / 2;
        if (delta < now_index + 1){
            cout << "Wait" << endl;
        }
        else{
            delta = 0;
            k_counter++;
            if (k_counter == K[now_index]){
                now_index--;
                while (K[now_index] == 0){
                    now_index--;
                }
                k_counter = 0;
            }
            if (now_index == -1){
```

```
                cout << "Stop";
                return 0;
            }
            cout << "Flip and wait" << endl;
        }
    }

    return 0;
}

        // if (k_counter <= 2 * (K[now_index] / 2)){
        //     if (ind == 0){
        //         cout << "Flip and wait" << endl;
        //         k_counter++;
        //     }
        //     else{
        //         cout << "Flip and wait" << endl;
        //         cin >> command;
        //         cout << "Flip and wait" << endl;
        //         cin >> command;
        //         cout << "Wait" << endl;
        //         k_counter++;
        //     }
        // }
        // else if (K[now_index] % 2 == 0){
        //     ind += K[now_index] % 2;
        //     k_counter = 0;
        //     now_index++;
        // }
        // else {
        //     ind += K[now_index] % 2;
        // }

        // cin >> command;
```

Task C ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <string>

using namespace std;

using v = vector<long long>;
using vv  = vector<vector<long long>>;
using ll = long long;

string new_string(char a, char b){
    if (a == '0' && b == '0'){
        return "0?";
    }
    else if (a == '0' && b == '1'){
        return "?1";
    }
    else if (a == '1' && b == '0'){
        return "?0";
    }
    else if (a == '1' && b == '1'){
        return "1?";
    }
    else if (a == '0' && b == '?'){
        return "00";
    }
    else if (a == '1' && b == '?'){
        return "11";
    }
    else if (a == '?' && b == '0'){
        return "10";
    }
    else if (a == '?' && b == '1'){
        return "01";
    }
    else{
        return "00";
    }
}

void solve(){
    string s, new_s = "";
    cin >> s;

    char first, second;
    for (int i = 0; i < s.length() / 2; i++){
        first = s[2 * i];
        second = s[2 * i + 1];
        new_s += new_string(first, second);
    }
    if (s.length() % 2 == 1){
        new_s += s[s.length() - 1];
    }

    cout << new_s << '\n';
    return;
}

int main(){
    int t;
    cin >> t;
    for (int i = 0; i < t; i++){
        solve();
    }

    return 0;
}
```

Task D ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <cmath>

using namespace std;

using v = vector<long long>;
using vv  = vector<vector<long long>>;
using ll = long long;

int main(){
    ll p = 998244353;
    ll key = 911660635;
    ll n;
    cin >> n;

    ll p_power = (n + 1) / p + (n + 1) / (p * p);
    ll fact = 1;
    if (((n + 1) / p) % 2 == 1){
        fact = p - 1;
    }

    int mod = (n + 1) % p;
    if (mod <= (p - 1) / 2){
        for (int i = 1; i <= mod; i++){
            fact *= i;
            fact %= p;
        }
    }
    else{
        int d = (mod - (p - 1) / 2) % 2;
        if (d == 0){
            fact *= key;
            fact %= p;
        }
        else{
            fact *= (p - key);
            fact %= p;
        }
        for (int i = (p - 1) / 2; i >= p - mod; i--){
            fact *= i;
            fact %= p;
        }
    }



    cout << p_power << ' ' << fact;

    return 0;
}
```

Task E ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <cmath>

using namespace std;

using v = vector<long long>;
using vv  = vector<vector<long long>>;
using ll = long long;
using dou = double;

int main(){
    int n;
    cin >> n;
    v N(n);
    for (int i = 0; i < n; i++){
        cin >> N[i];
    }

    int q;
    cin >> q;
    int a, b, d;
    ll result = 0;
    for (int w = 0; w < q; w++){
        cin >> a >> b >> d;

        result = 0;
        for (int i = a; i < b; i++){
            result += d / N[i];
            if (d % N[i] != 0){
                result++;
            }
        }
        cout << result << '\n';
    }
}
```

Task F ()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;

using v = vector<long long>;
using vv  = vector<vector<long long>>;
using ll = long long;

int main(){
    int n;
    cin >> n;
    v L(n);
    v R(n);
    for (int i = 0; i < n; i++){
        cin >> L[i] >> R[i];
    }

    v dL(n + 1);
    v dR(n + 1);
    for (int i = 0; i < n; i++){
        dL[i + 1] = dL[i] + L[i];
        dR[i + 1] = dR[i] + R[i];
    }

    v good(n + 1);
    int best = n + 1;
    for (int i = 0; i < n + 1; i++){
        best = n + 1;
        for (int j = i + 1; j < n + 1; j++){
            if (dL[j] - dL[i] <= 0 && dR[j] - dR[i] >= 0){
                best = j;
                break;
            }
        }
        good[i] = best;
    }

    v used(n + 1);
    int max_count = 1, count = 1;
    int next = -1;
    for (int i = 0; i < n; i++){
        if (!used[i]){
            next = i;
            used[next] = 1;
            count = 1;
            while (good[next] != n + 1){
                next = good[next];
                used[next] = 1;
                count++;
            }

            if (count > max_count){
                max_count = count;
            }
        }
    }

    cout << max_count;
    return 0;
}
```