Решения задач очного этапа Открытой олимпиады Университета Иннополис, 2014-2015 учебный год.

Задача A. Сисадмин.

Решение задачи на языке C++.

```
#include <cstdio>
#include <cstdlib>
#include <map>

using namespace std;
int n;
int imp;

map<int, char> states; // states[x] = mask, где mask = (соединены левый и правый)(соединен правый с верхним)(соединен левый с верхним)

void update(int x, int bit)
{
	int t = states[x];
	if (__builtin_popcount(t) < 2)
	{
		t |= (1 << bit);
		if (__builtin_popcount(t) >= 2)
			imp--;
		states[x] = t;
	}
}

void init()
{
	imp = n / 2;
	update(1, 1);
	if (n % 2 == 0)
		update(n / 2, 2);
}

int h(int x)
{
	return 32 - (int)__builtin_clz(x);
}

void join(int a, int b)
{
	if (h(a) > h(b))
		swap(a, b);
	if ((b >> (h(b) - h(a))) == a)
	{
		int last = b & 1;
		b >>= 1;
		while (b != a)
		{
			update(b, last);
			last = b & 1;
			b >>= 1;
```

```
            }
      } else {
            int lasta = a & 1, lastb = b & 1;
            a >>= 1;
            b >>= 1;
            while (h(b) != h(a))
            {
                  update(b, lastb);
                  lastb = b & 1;
                  b >>= 1;
            }
            while (a != b)
            {
                  update(a, lasta);
                  update(b, lastb);
                  lasta = a & 1,
                  lastb = b & 1,
                  a >>= 1,
                  b >>= 1;
            }
            update(a, 2);
      }
}

int main()
{
      int m;
      freopen("sysadmin.in", "r", stdin);
      freopen("sysadmin.out", "w", stdout);
      scanf("%d%d", &n, &m);

      init();
      printf("%d\n", imp);
      for(int i = 0; i < m; i++)
      {
            int x, y;
            scanf("%d%d", &x, &y);
            join(x, y);
            printf("%d\n", imp);
      }
}
```

Задача B. Гонка.

Решение задачи на языке C++.

```
#include <bits/stdc++.h>
using namespace std;

string fName = "race";

vector <int> fl;
vector <int> fr;

void build(vector <int> a, int k){
    fill(fl.begin(), fl.end(), -1);
    fill(fr.begin(), fr.end(), -1);
```

```
    k--;
    for (int i = 0; i < a.size(); i++){
        if (i % k){
            fl[i] = max(fl[i - 1], a[i]);
        } else {
            fl[i] = a[i];
        }
    }
    fr.back() = a.back();
    for (int i = a.size() - 2; i >= 0; i--){
        if ((i + 1) % k){
            fr[i] = max(fr[i + 1], a[i]);
        } else {
            fr[i] = a[i];
        }
    }
}

int read() {
int c = getchar();
while (c <= 32) c = getchar();
int x = 0;
while (c > 32) { x = x * 10 + c - '0'; c = getchar(); }
return x;
}

int main(){
    ifstream cin((fName + ".in").c_str());
    ofstream cout((fName + ".out").c_str());
    int n, X1, Y1, M1, D1, X2, Y2, M2, D2;
    cin >> n >> X1 >> Y1 >> M1 >> D1 >> X2 >> Y2 >> M2 >> D2;
    vector <int> a(n);
    vector <int> b(n);
    vector <int> mp(2*n);
        a[0] = D1;
        b[0] = D2;
    for (int i = 0; i < n; i++){
        if (i > 0) a[i] = ((long long) a[i - 1] * X1 + Y1) % M1;
        mp[i] = a[i];
    }
    for (int i = 0; i < n; i++){
        if (i > 0) b[i] = ((long long) b[i - 1] * X2 + Y2) % M2;
        mp[n + i] = b[i];
    }
    sort(mp.begin(), mp.end());
    mp.resize(distance(mp.begin(), unique(mp.begin(), mp.end())));
    for (int i = 0; i < n; i++){
        a[i] = distance(mp.begin(), lower_bound(mp.begin(), mp.end(),
a[i]));
        b[i] = distance(mp.begin(), lower_bound(mp.begin(), mp.end(),
b[i]));
    }
    int l = 0, r = n + 1;
    vector <int> used(mp.size());
    fl.resize(n);
    fr.resize(n);

    while (r - l > 1){
```

```
        int m = (l + r) / 2;
        bool can = false;
        fill(used.begin(), used.end(), 0);
        if (m > 1) {
            build(a, m);
        }
        for (int i = 0; i < n - m + 1; i++){
            if (m > 1) {
                used[max(fr[i], fl[i + m - 1])] = 1;
            } else {
                used[a[i]] = 1;
            }
        }
        if (m > 1) {
            build(b, m);
        }
        for (int i = 0; i < n - m + 1; i++){
            int mx = m > 1? max(fr[i], fl[i + m - 1]) : b[i];
            if (used[mx]){
                can = true;
                break;
            }
        }
        if (can){
            l = m;
        } else {
            r = m;
        }

    }
    cout << l;
    return 0;
}
```

Задача C. Аттракцион.

Решение задачи на языке C++.

```
#include <cstdio>
#include <iostream>
#include <string>
#include <vector>
using namespace std;

int solve(string s, string t)
{
     int ans = 0;
     for(char c = '0'; c <= '1'; c++)
     {
          string ch(1, c);
          string kmp = s + "2" + t + ch + t; // проверяем, что первая
строчка является \

циклическим сдвигом второй с символом c.
          vector<int> p(kmp.length(), 0);
          for (int i = 1; i < kmp.length(); i++) {
               int t = p[i - 1];
               while (t > 0 && kmp[i] != kmp[t])
                    t = p[t - 1];
```

```
                if (kmp[i] == kmp[t])
                      t++;
                p[i] = t;
                if (i > s.length() && p[i] == s.length())
                {
                      if (c == '0')
                            ans--;
                      else
                            ans++;
                      break;
                }
          }
      }
      return ans;

}

int main()
{
      freopen("roulette.in", "r", stdin);
      freopen("roulette.out", "w", stdout);
      int n;
      cin >> n;
      string s, t;
      getline(cin, s);
      getline(cin, s);
      getline(cin, t);
      int ans = solve(s, t);
      if (ans == -1)
            cout << "No" << endl;
      else if (ans == 0)
            cout << "Random" << endl;
      else
            cout << "Yes" << endl;
}
```

Решение задачи на языке Java.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.StringTokenizer;
import java.io.FileReader;
public class roulette_pm {

    private void solve() throws IOException {
        int n = nextInt();
        String a = next();
        String b = next();
        int res = 0;
        for (int i = 0; i < 2; i++) {
            for (int j = 0; j < 2; j++) {
                if (isShift(a + i, b + j)) {
                    res |= (1 << j);
                }
            }
        }
        out.println(new String[]{"", "No", "Yes", "Random"}[res]);
    }
```

```
    private boolean isShift(String a, String b) {
        String s = a + "$" + b + b;
        int[] p = new int[s.length() + 1];
        for (int i = 2; i <= s.length(); i++) {
            int k = p[i - 1];
            while (k > 0 && s.charAt(k) != s.charAt(i - 1)) {
                k = p[k];
            }
            if (s.charAt(k) == s.charAt(i - 1)) k++;
            p[i] = k;
            if (i > a.length() && k == a.length()) {
                return true;
            }
        }
        return false;
    }

    BufferedReader br;
    StringTokenizer st;
    PrintWriter out;

    String next() throws IOException {
        while (st == null || !st.hasMoreTokens()) {
            st = new StringTokenizer(br.readLine());
        }
        return st.nextToken();
    }

    int nextInt() throws IOException {
        return Integer.parseInt(next());
    }

    public static void main(String[] args) throws IOException {
      new roulette_pm().run();
    }

    private void run() throws IOException {
      br = new BufferedReader(new FileReader("roulette.in"));
      out = new PrintWriter("roulette.out");
        solve();
        out.close();
    }
}
```

Задача D. Телепортация.

Решение задачи на языке C++.

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

using namespace std;

struct tele {
    int x, l, r, c, id;
};

int read() {
    int c = getchar();
```

```
    while (c <= 32) c = getchar();
    int x = 0;
    while (c > 32) {
        x = x * 10 + c - '0';
        c = getchar();
    }
    return x;
}


int const N = 1234567;
long long const INF = 1LL << 60;
tele a[N];
long long d[N];
int ways[N];
int p[N];
int tn;
int rev[N];

struct comp {
    bool operator () (int i, int j) const {
        if (d[i] != d[j]) return d[i] > d[j];
        return i < j;
    }
};

int get(int x) {
    return x == p[x] ? x : (p[x] = get(p[x]));
}

int main() {
    freopen("teleports.in", "r", stdin);
    freopen("teleports.out", "w", stdout);
    int n, s;
    scanf("%d%d", &n, &s);
    --s;
    tn = 1;
    while (tn < n) tn *= 2;
    for (int i = 0; i < n; i++) {
        int x, l, r, c;
        scanf("%d%d%d%d", &x, &l, &r, &c);
        a[i] = {x, l, r, c, i};
    }
    auto cmpx = [](tele const & a, tele const & b) {
        return a.x < b.x;
    };
    std::sort(a, a + n, cmpx);
    for (int i = 0; i < n; i++) if (s == a[i].id) {
        s = i;
        break;
    }
    for (int i = 0; i <= n; i++) p[i] = i;
    for (int i = 0; i < n; i++) d[i] = INF;
    priority_queue<int, vector<int>, comp> q;
    d[s] = a[s].c;
    ways[s] = 1;
    q.push(s);
    while (!q.empty()) {
```

```
        int v = q.top();
        q.pop();
        for (int dir = -1; dir <= 1; dir += 2) {
            tele f;
            int left = a[v].x + a[v].r * dir;
            int right = a[v].x + a[v].l * dir;
            if (left > right) std::swap(left, right);
            f.x = left;
            int start = std::lower_bound(a, a + n, f, cmpx) - a;
            while (true) {
                start = get(start);
                if (start >= n || right < a[start].x) break;
                if (d[start] >= d[v] + a[start].c) {
                    if (d[start] == INF) {
                        d[start] = d[v] + a[start].c;
                        q.push(start);
                    }
                    assert(d[v] + a[start].c >= d[start]);
                    ways[start] += ways[v];
                    if (ways[start] > 2) ways[start] = 2;
                } else {
                    p[get(start)] = get(start + 1);
                    start++;
                    continue;
                }
                if (ways[start] >= 2) p[get(start)] = get(start + 1);
                start++;
            }
        }
    }
    for (int i = 0; i < n; i++) rev[a[i].id] = i;
    for (int it = 0; it < n; it++) {
        int i = rev[it];
        if (d[i] == INF) puts("-1"); else {
            printf("%lld %s\n", d[i] - a[i].c, ways[i] >= 2 ? "YES" :
"NO");
        }
    }
}
```

Задача E. Клевер.

Решение задачи на языке Pascal.

```
Uses      Math;
Const     eps = 1e-8;

type
    TPoint = record x, y: extended;
    End;
    TLine = record a, b, c: extended;
    End;
var       V :   array[1..10] of TPoint;
     L1, L2 :   TLine;
     S, Min :   extended;
  i, k, ans :   integer;
     j, n   :   integer;
     flag   :   boolean;
{*************************************************************}
```

```
Function Eq(q, w: extended): boolean;
Begin
    Result := abs(q - w) < eps;
End;
{*****************************************************************}
Function More(q, w: extended): boolean;
Begin
    Result := q - eps > w;
End;
{*****************************************************************}
Function Dist(q, w: TPoint): extended;
Begin
    Result := sqrt(sqr(q.x - w.x) + sqr(q.y - w.y));
End;
{*****************************************************************}
Function Line(q, w: TPoint): TLine;
Begin
    Result.a := q.y - w.y;
    Result.b := w.x - q.x;
    Result.c := q.x * w.y - w.x * q.y;
End;
{*****************************************************************}
Function Cross(q, w: TLine): TPoint;
Var d : extended;
Begin
    d := q.a * w.b - w.a * q.b;
    Result.x := -(q.c * w.b - w.c * q.b)/d;
    Result.y := -(q.a * w.c - w.a * q.c)/d;
End;
{*****************************************************************}
Function NonCross(q, w: TLine): boolean;
Begin
    Result := Eq(q.a * w.b - w.a * q.b, 0);
End;
{*****************************************************************}
Function SquareTriangle(var P1, P2, P3: TPoint): extended;
Begin
    Result := (P1.x*(P2.y-P3.y)-P1.y*(P2.x-P3.x)+(P2.x*P3.y-
P3.x*P2.y))/2;
End;
{*****************************************************************}
Function SummOfDistance(q: TPoint): extended;
Var k : integer;
    S : extended;
Begin
    S:= 0;
    for k:= 1 to 4 do
        S:= S + Dist(q, V[k]);
    Result:= S;
End;
{*****************************************************************}
Procedure Collinear(i: integer);
Var k, m : integer;
Begin
    Min:= Power(1000000, 3);
    for k:= i to i+2 do begin
        if k > 4 then m:= k - 4 else m:= k;
        S:= SummOfDistance(V[m]);
```

```
        if More(Min, S) then begin
           Min:= S;  Ans:= m;  end;
    end;
    writeln(V[Ans].x:0:15,' ', V[Ans].y:0:15);
End;

{*****************************************************************}
BEGIN
    assign(input, 'clover.in');  reset(input);
    assign(output,'clover.out'); rewrite(output);
    readln(n);
    for j:= 1 to n do begin
        for i:= 1 to 4 do
            read(V[i].x, V[i].y);
        flag:= false;
        for i:= 1 to 4 do begin
            S:= SquareTriangle(V[i], V[i mod 4 + 1], V[(i+1) mod 4 +
1]);
            if Eq(S, 0) then begin Collinear(i); flag:= true; break;
end;
        end;
        if flag then continue;
        i:= 4;
        for k:= 2 to 4 do begin
            L1 := Line(V[1], V[k]);
            if k = 2 then L2 := Line(V[3], V[4]);
            if k = 3 then L2 := Line(V[4], V[2]);
            if k = 4 then L2 := Line(V[2], V[3]);
            if NonCross(L1, L2) then continue;
            Inc(i);     V[i] := Cross(L1, L2);
        end;
        Min:= Power(1000000, 3);
        for k:= 1 to i do begin
            S:= SummOfDistance(V[k]);
            if More(Min, S) then begin
               Min:= S;  Ans:= k; end;
        end;
        writeln(V[Ans].x:0:15,' ', V[Ans].y:0:15);
    end;
END.
```