Первый отборочный тур

Задача 1. Упаковка числа (Варианты 1,3,5,7) Рассмотрим способ упакованной записи чисел в двоичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 1111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в двоичной системе и закрывающей скобки –]. Так число 111111100 в упакованной записи будет записано следующим образом: 1[1[1[10]]]0[10] Найдите наименьшую по длине упакованную запись двоичного числа :

1. 11100000000100000111111111110
3. 11111111111000111000000000100000
5. 11000001111111111110000111000000001
7. 11000000001110000001111111111110001

Задача будет оценена в зависимости от верности упаковки и длины получившейся последовательности.

Задача 1. Упаковка числа (Варианты 2,4,6,8) Рассмотрим способ упакованной записи чисел в троичной системе. Идея способа состоит в том, чтобы одинаковые подряд идущие цифры, например 111111111, заменять на конструкцию, состоящую из повторяемой цифры (в примере – 1), открывающей скобки перед числом повторений – [, упакованной записи числа повторений в троичной системе и закрывающей скобки –]. Так число 11111111000 в упакованной записи будет записано следующим образом: 1[2[2]]0[10] Найдите наименьшую по длине упакованную запись троичного числа:

2. 11122222222100001111111111110
4. 11111111111110011122222222210000
6. 22222222100011111111111111001110
8. 11102222222210001111111111111100

Задача будет оценена в зависимости от верности упаковки и длины получившейся последовательности.

Ответы:

1	1110[1000]1000001[1011]0	24
2	1112[22]100001[111]0	20
3	1[1011]0001110[1000]100000	26
4	1[111]001112[22]10000	21
5	11000001[1011]0001110[1000]1	28
6	2[22]10001[111]001110	21
7	110[1000]111000001[1011]0001	28
8	11102[22]10001[111]00	21

Задача 2. Последовательность

1. Последовательность чисел 38, 37, 36, 35, 39, 34, 33, 32, 31, 30, ... , 100 была получена выписыванием 500 первых натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите число, стоящее в последовательности на 114-м месте.
2. Последовательность чисел 38, 37, 36, 35, 39, 34, 33, 32, 31, 30, ... , 100 была получена выписыванием 500 первых натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите номер места, на котором стоит в последовательности число 114.
3. Последовательность чисел 100, 200, 300, 301, 302, 303, 304, 309, 350, 351 ... 38 была получена выписыванием 500 первых натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в алфавитном порядке. Определите число, стоящее в последовательности на 114-м месте.
4. Последовательность чисел 100, 200, 300, 301, 302, 303, 304, 309, 350, 351 ... 38 была получена выписыванием 500 первых натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в алфавитном порядке. Определите номер места, на котором стоит в последовательности число 114.
5. Последовательность чисел 38, 36, 34, 32, 30, 28, 26, 24, 22, 20, ... , 100 была получена выписыванием 500 первых чётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите число, стоящее в последовательности на 114-м месте.
6. Последовательность чисел 38, 36, 34, 32, 30, 28, 26, 24, 22, 20, ... , 100 была получена выписыванием 500 первых чётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите номер места, на котором стоит в последовательности число 114.
7. Последовательность чисел 100, 200, 300, 302, 304, 350, 352, 354, 356, 358, ... , 38 была получена выписыванием 500 первых чётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в алфавитном порядке. Определите номер места, на котором стоит в последовательности число 114.
8. Последовательность чисел 38, 37, 36, 35, 39, 34, 33, 32, 31, 30, ... , 100 была получена выписыванием 500 первых натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите число, стоящее в последовательности на 114-м месте.
9. Последовательность чисел 37, 35, 39, 33, 31, 27, 25, 29, 23, 21, ... , 301 была получена выписыванием 500 первых нечётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите число, стоящее в последовательности на 411-м месте.
10. Последовательность чисел 37, 35, 39, 33, 31, 27, 25, 29, 23, 21, ... , 301 была получена выписыванием 500 первых нечётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в обратном алфавитном порядке. Определите номер места, на котором стоит в последовательности число 411.

11. Последовательность чисел 301, 303, 309, 351, 353, 359, 355, 357, 361, 363, ... , 37 была получена выписыванием 500 первых нечётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в алфавитном порядке. Определите число, стоящее в последовательности на 411-м месте.
12. Последовательность чисел 301, 303, 309, 351, 353, 359, 355, 357, 361, 363, ... , 37 была получена выписыванием 500 первых нечётных натуральных чисел так, чтобы их записи римскими цифрами были отсортированы в алфавитном порядке. Определите номер места, на котором стоит в последовательности число 411.

Ответы:

1	2	3	4	5	6	7	8	9	10	11	12
125	**136**	**255**	**365**	**644**	**268**	**233**	**125**	**217**	**370**	**569**	**131**

Задача 3.

Возможны другие правильные ответы.

Вариант	Возможный ответ
1	INV(MASK(TRANS(MASK(MAX(X, INV(X))))))
2	INV(TRANS(MASK(INV(DIF(X, X)))))
3	DIF(TRANS(MASK(INV(DIF(X, X)))), MASK(INV(DIF(X, X))))
4	DIF(TRANS(MASK(DIF(X, X))), MASK(DIF(X, X)))
5	INV(MIN(MASK(TRANS(INV(MASK(MIN(INV(X),X))))), TRANS(MASK(TRANS(INV(MASK(MIN(INV(X),X))))))))
6	MAX(TRANS(MASK(MIN(X, INV(X)))) MASK(MIN(X, INV(X))))
7	MASK(TRANS(MASK(MIN(A, INV(A))))

Задача 4.

В задаче требуется написать программу. При запаковке следовало учитывать, что короткие последовательности не требуется изменять.

Задача 5.

Вариант	Ответ

1	punycode-был-разработан-для-однозначного-преобразования-доменных-имён-в-последовательность-ascii-символов
2	punycode--стандартизированный-метод-преобразования-последовательностей-unicode-символов
3	punycode-позволяет-конвертировать-набор-символов-в-кодировке-unicode-в-набор-символов-для-существующей-dns
4	punycode--это-способ-приведения-интернационализированных-доменных-имен-содержащих-в-себе-unicode-символы

Задача 6.

Вариант	Ответ
1	221111
2	12122112212122
3	22121121121211
4	11212212212122

Задача 7.

Вариант	Ответ
1	333359 5/6 333353 5/6 333286 1/3
2	333692 5/6 333298 5/6 333008 1/3
3	333109 1/3 333335 5/6 333554 5/6
4	333386 5/6 333388 1/3 333224 5/6
5	333382

	333168 333450
6	333233 1/3 333619 1/3 333147 1/3
7	333555 5/6 332897 1/3 333546 5/6
8	333861 1/2 333015 333123 1/2

Задача 8.

```
0]0][?(]?)[}[#]{+}]4~/]4*]{}=(0]{=([}[#]{+}]4~}/]4*]{=(Y.E.S.")N.O.")
Y.E.S.")N.O."
```

Задача 9.

```
0]0]?((+#]?]!([@]1!)[]]@=([!])[)[_}[]]{}>([{]}1!){0]?)[[."
```

Задача 10.

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <string>
#include <cstring>
#include <cctype>

using namespace std;

const char *parse_string = "([920, 438]7(27)8([0,
0]14(39)16(30)-1)11(49)12(48)13([56, 536]13([86, 856]11([176,
526]13(2)14([186, 96]14([196, 106]13(35)14([216, 156]17(31)18([216,
396]11(4)12([246, 906]8([266, 446]13([316, 976]15(41)16([326,
646]14([336, 216]11([336, 686]10(38)11([396, 876]10(5)11([426,
0]12(44)13([426, 966]17(9)18([436, 956]10(36)11([446,
806]12(18)13([466, 796]13([476, 396]16([476, 556]13(10)14([546,
226]12([626, 456]14(40)15([636, 86]12([686, 856]7(19)8([716,
706]11(23)12([726, 216]15(3)16([736, 196]10([746, 766]13([766,
726]12([796, 886]13([816, 356]16(37)17([856, 926]13([886,
626]11(20)12([926, 96]11([946, 636]8(11)9([956, 616]12([956,
```

```
816]14(15)15(34)-1)13(32)-1)-1)12(33)-1)-1)14(22)-1)-1)14(43)-1)13(16
)-1)14(46)-1)11(14)-1)-1)-1)-1)13(21)-1)-1)13(26)-1)-1)17(17)-1)14(28
)-1)-1)-1)-1)-1)-1)-1)12(50)-1)15(7)-1)-1)14(45)-1)9(6)-1)-1)-1)-1)15
(13)-1)-1)12(29)-1)14(25)-1)14(47)15(8)16(1)17(24)18(42)19(12)-1)";

struct Tree
{
    int w = -1;
    int h = -1;
    vector< pair< int, Tree * > > sons;
    friend std::ostream&operator<<(std::ostream &out, const Tree
&tree)
    {
        out << "(";
        if (tree.w != -1) {
            out << "[" << tree.w << ", " << tree.h << "]";
            for (auto nxt : tree.sons) {
                out << nxt.first << *(nxt.second);
            }
            out << "-1)";
        } else {
            out << tree.h << ")";
        }
        return out;
    }
    int height() const
    {
        int res = 0;
        for (auto nxt : sons) {
            int val = nxt.second->height() + 1;
            if (res < val) {
                res = val;
            }
        }
        return res;
    }
    ~Tree()
    {
        for (auto nxt : sons) {
            delete nxt.second;
        }
    }
};
```

```
int
getValue(const char **s)
{
    while (**s != '-' && !isdigit(**s)) {
        ++(*s);
    }
    int res = 0;
    bool sign = false;
    if (**s == '-') {
        sign = true;
        ++(*s);
    }
    while (isdigit(**s)) {
        res = res * 10 + **s - '0';
        ++(*s);
    }
    ++(*s);
    return sign ? -res : res;
}

Tree *
construct(const char **s)
{
    Tree *res = new Tree();
    const char *ptr = *s;
    if (**s == '[') {
        res->w = getValue(&ptr);
        res->h = getValue(&ptr);
        if (res->w == -1 || res->h == -1) {
            cerr << "fail!" << endl;
            cerr << *s << endl;
        }
        int tmp = getValue(&ptr);
        while (tmp != -1) {
            Tree *new_son = construct(&ptr);
            res->sons.push_back(make_pair(tmp, new_son));
            tmp = getValue(&ptr);
        }
    } else { // List
        res->h = getValue(&ptr);
    }
    *s = ptr;
```

```
    return res;
}


int
main()
{
    const char *s1 = parse_string + 1;
    Tree *decision_tree = construct(&s1);
    while (decision_tree->w != -1) {
        int curX = decision_tree->w, curY = decision_tree->h;
        cout << "? " << curX << " " << curY << endl;
        int sum;
        cin >> sum;
        bool findVal = false;
        for (auto x : decision_tree->sons) {
            if (x.first == sum) {
                findVal = true;
                decision_tree = x.second;
                break;
            }
        }
        if (!findVal) {
            cerr << "Oops!" << endl;
            return 1;
        }
    }
    cout << "! " << decision_tree->h << endl;
    return 0;
}
```

Второй отборочный тур

Задача 1. Восстановите связное сообщение по тексту, который был получен после некоторого преобразования исходного текста.

Вариант	Ответ
1	A numeric character reference in HTML refers to a character by its Universal Character Set/Unicode code point.
2	Web pages authored using HTML may contain multilingual text represented with the Unicode universal character set.
3	The relationship between Unicode and HTML tends to be a difficult topic for many computer professionals.
4	It is possible to represent characters from the whole of Unicode inside an HTML document by using a numeric character reference.

Задача 2. Последовательность.

Вариант:	1	2	3	4
Ответ:	**297**	**2016**	**134**	**555**

Задача 3.
Возможное решение.

```
import sys

def main():
    number_of_view = 0
    number_of_sharing = 0

    for line in sys.stdin:
        date, time, action = line.split()
        if date < '2014-12-31' or
           date == '2014-12-31' and time < '07:00:00':
            if action == 'W':
                number_of_view += 1
            elif action == 'S':
                number_of_sharing += 1
            else:
```

```
                pass

    christmas_tree_size = number_of_view + 5 * number_of_sharing

    fibonacci = [1, 1]
    sum_fibonacci = [0, 1]

    while sum_fibonacci[-1] < christmas_tree_size:
        fibonacci.append(fibonacci[-1] + fibonacci[-2])
        sum_fibonacci.append(fibonacci[-1] + sum_fibonacci[-1])

    length = 100 + (len(sum_fibonacci) - 2) * 20
    print(length)

main()
```

Задача 4.

```
#include <cstdio>
#include <cstdlib>
#include <cmath>
#include <cstring>
#include <ctime>
#include <cassert>
#include <iostream>
#include <fstream>
#include <iomanip>
#include <vector>
#include <set>
#include <map>
#include <algorithm>
#include <string>
#include <sstream>

using namespace std;

int getnum(const string &s, int &i) {
     int ans = 0;
     int lastd = -1;

     for (; i < s.length(); i++) {
          if (s[i] != '[' && lastd != -1) {
               ans *= 2;
```

```
                ans += lastd;
            }

            if (s[i] == '[') {
                i++;
                int num = getnum(s, i);
                for (int j = 0; j < num; j++) {
                    ans *= 2;
                    ans += lastd;
                }
                lastd = -1;
                continue;
            }

            if (s[i] == ']') {
                return ans;
            }
            lastd = s[i] - '0';
        }
    }

    string convert(const string& s) {
        string ans = "";
        for (int i = 0; i < s.length(); i++) {
            if (i + 1 == s.length()) {
                ans += s[i];
                continue;
            }
            if (s[i + 1] == '[') {
                char c = s[i];
                i += 2;
                int num = getnum(s, i);
                for (int j = 0; j < num; j++) {
                    ans += c;
                }
            } else {
                ans += s[i];
            }
        }
        return ans;
    }

    bool lessnumber(const string & a, const string & b) {
```

```
	if (a.length() != b.length()) return a.length() < b.length();
	return a < b;
}

int main()
{
	string s, t;
	cin >> s >> t;

	string ss = convert(s), tt = convert(t);

	if (lessnumber(ss, tt)) {
		cout << "<\n";
	} else if (ss == tt) {
		cout << "=\n";
	} else {
		cout << ">\n";
	}

	return 0;
}
```

Задача 5.
В задаче может быть несколько правильных ответов.

Задача 6.
111111222212212122

Задача 7.
Вариант 1.
```
Balan 34
Cynric 34
Arthur 32
Geraint 32
Gingalain 32
Josephus 32
Dinadan 30
Mordred 30
Bruin 28
Calogrenant 28
Lanval 28
Owain 28
```

Rience 28
Claudin 26
Culhwch 26
Elyan 26
Epinogres 26
Maleagant 26
Mark 26
Meliodas 26
Palamedes 26
Ysbaddaden 26
Aglovale 24
Balin 24
Ban 24
Gaheris 24
Galahad 24
Gareth 24
Gorlois 24
Griflet 24
Laudine 24
Lionel 24
Daniel 22
Edern 22
Galeschin 22
Lamorak 22
Lot 22
Morien 22
Pelleas 22
Pelles 22
Tom 22
Vortimer 22
Bedivere 20
Brutus 20
Ector 20
Esclados 20
Galehault 20
Gwyn 20
Hengest 20
Horsa 20
Lunete 20
Morvydd 20
Pellinore 20
Sagramore 20
Dagonet 18

Eliwlod 18
Erec 18
Faerie 18
Gawain 18
Kahedin 18
Leodegrance 18
Lucan 18
Meirchion 18
Oberon 18
Safir 18
Breunor 16
Cador 16
Cerdic 16
Claudas 16
Constantine 16
Feirefiz 16
Fisher 16
Hueil 16
Loholt 16
Merlin 16
Segwarides 16
Tor 16
Ywain 16
Amr 14
Bagdemagus 14
Caradoc 14
Constans 14
Esclabor 14
Gornemant 14
Joseph 14
Kay 14
Lucius 14
Mabon 14
Melehan 14
Morgan 14
Taliesin 14
Brangaine 12
Durnure 12
Modron 12
Manawydan 10
Percival 10
Vortigern 10
Catigern 8

Menw 8

Вариант 2.

Galehault 38
Gwyn 38
Hueil 36
Arthur 30
Balin 30
Gornemant 30
Josephus 30
Manawydan 30
Sagramore 30
Segwarides 30
Cerdic 28
Constantine 28
Feirefiz 28
Geraint 28
Hector 28
Lucan 28
Merlin 28
Vortigern 28
Ywain 28
Amr 26
Brangaine 26
Brutus 26
Esclados 26
Gareth 26
Horsa 26
Lunete 26
Rience 26
Aurelius 24
Blanchefleur 24
Esclabor 24
Lionel 24
Morien 24
Percival 24
Agravain 22
Bagdemagus 22
Bedivere 22
Catigern 22
Lamorak 22
Lucius 22
Maleagant 22

Modron 22
Mordred 22
Morgan 22
Pelleas 22
Balan 20
Claudin 20
Edern 20
Eliwlod 20
Guiron 20
Hoel 20
Kahedin 20
Lancelot 20
Ysbaddaden 20
Daniel 18
Dinadan 18
Dindrane 18
Gaheris 18
Galahad 18
Gorlois 18
Joseph 18
Madoc 18
Meliodas 18
Pellam 18
Tom 18
Breunor 16
Cador 16
Meirchion 16
Pellinore 16
Aglovale 14
Culhwch 14
Dagonet 14
Epinogres 14
Erec 14
Faerie 14
Gawain 14
Laudine 14
Leodegrance 14
Lot 14
Morholt 14
Oberon 14
Owain 14
Pelles 14
Taliesin 14

Urien 14
Vortimer 14
Bruin 12
Claudas 12
Constans 12
Fisher 12
Loholt 12
Mabon 12
Palamedes 12
Tor 12
Ector 10
Elyan 10
Hengest 10
Tristan 10
Durnure 8
Melehan 8

Вариант 3.

Gingalain 42
Brutus 34
Kahedin 34
Lamorak 30
Leodegrance 30
Pelleas 30
Catigern 28
Edern 28
Gorlois 28
Joseph 28
Lancelot 28
Mordred 28
Percival 28
Arthur 26
Ban 26
Caradoc 26
Cerdic 26
Dindrane 26
Elyan 26
Lanval 26
Merlin 26
Pellam 26
Rience 26
Safir 26
Tristan 26

Eliwlod 24
Epinogres 24
Esclados 24
Guiron 24
Hueil 24
Kay 24
Loholt 24
Madoc 24
Melehan 24
Morgan 24
Palamedes 24
Blanchefleur 22
Calogrenant 22
Claudin 22
Durnure 22
Esclabor 22
Hector 22
Maleagant 22
Aurelius 20
Bagdemagus 20
Galahad 20
Gornemant 20
Lucius 20
Meirchion 20
Owain 20
Pelles 20
Tom 20
Tor 20
Urien 20
Balan 18
Bruin 18
Cador 18
Constantine 18
Culhwch 18
Dagonet 18
Gaheris 18
Gawain 18
Josephus 18
Laudine 18
Lionel 18
Manawydan 18
Taliesin 18
Ysbaddaden 18

Brangaine 16
Claudas 16
Constans 16
Daniel 16
Galehault 16
Horsa 16
Mabon 16
Menw 16
Morholt 16
Dinadan 14
Faerie 14
Hoel 14
Morien 14
Oberon 14
Segwarides 14
Aglovale 12
Breunor 12
Galeschin 12
Gareth 12
Geraint 12
Lot 12
Lucan 12
Lunete 12
Mark 12
Meliodas 12
Modron 12
Morvydd 10
Amr 8
Balin 8
Ector 8
Hengest 8

Вариант 4.

Edern 38
Pellam 34
Ban 32
Geraint 30
Lionel 30
Brangaine 28
Elyan 28
Erec 28
Fisher 28
Guiron 28

Breunor 26
Bruin 26
Culhwch 26
Durnure 26
Galahad 26
Gorlois 26
Laudine 26
Segwarides 26
Arthur 24
Aurelius 24
Bagdemagus 24
Dagonet 24
Dinadan 24
Esclados 24
Gaheris 24
Gareth 24
Griflet 24
Lanval 24
Madoc 24
Mark 24
Modron 24
Palamedes 24
Tor 24
Vortigern 24
Agravain 22
Calogrenant 22
Catigern 22
Constantine 22
Faerie 22
Hoel 22
Hueil 22
Loholt 22
Lunete 22
Morien 22
Ysbaddaden 22
Bedivere 20
Brutus 20
Gingalain 20
Gornemant 20
Mabon 20
Menw 20
Morvydd 20
Pelleas 20

Taliesin 20
Tom 20
Ywain 20
Aglovale 18
Claudas 18
Ector 18
Feirefiz 18
Gwyn 18
Kahedin 18
Lancelot 18
Lucan 18
Lucius 18
Manawydan 18
Merlin 18
Mordred 18
Pelles 18
Sagramore 18
Caradoc 16
Claudin 16
Constans 16
Daniel 16
Dindrane 16
Eliwlod 16
Hengest 16
Joseph 16
Kay 16
Lot 16
Meliodas 16
Rience 16
Vortimer 16
Cador 14
Galehault 14
Gawain 14
Horsa 14
Melehan 14
Morgan 14
Morholt 14
Epinogres 12
Hector 12
Owain 12
Galeschin 10
Cynric 8
Esclabor 8

```
Josephus 8
Urien 8
Pellinore 6
```

Задача 8.
Возможное решение.

```
?](?])[[.[(.[)"
```

Задача 9.
Возможное решение.

```
?}(1](1+]]}{}%]0=!]{]]{}=!}[}{}{&]{[){=(Y.1]!)[!(N.!)?})"
```

Задача 10.

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <string>
#include <cstring>
#include <cctype>

using namespace std;

const char *parse_string = "([2, 124]0([0, 2]0([0, 0]0([0, 5]0([0,
8]0(42)1(1)-1)1(24)-1)1([0, 1]0(47)1(39)-1)-1)1([0, 3]0([0,
7]0(12)1(8)-1)1([0, 4]0(27)1([0, 5]0(49)1(30)-1)-1)-1)-1)1([56,
539]0([86, 858]0([176, 528]0(2)1([186, 97]0([196, 108]0(35)1([216,
162]0(31)1([216, 401]0([247, 908]0(6)1([268, 450]0(45)1([318,
979]0(41)1([326, 647]0([336, 688]0(38)1([337, 217]0([396,
877]0(5)1([426, 0]0(44)1([426, 974]0([436, 957]0(36)1([446,
806]0(18)1([466, 802]0(28)1([476, 398]0([476, 561]0([546, 226]0([626,
457]0(40)1([636, 86]0([686, 860]0(19)1([716, 708]0(23)1([726,
221]0([736, 197]0([746, 770]0([766, 726]0([796, 887]0([816,
361]0([856, 927]0([886, 628]0(20)1([916, 436]0(48)1([926, 99]0([946,
638]0(11)1([956, 616]0([957,
821]0(34)1(15)-1)1(32)-1)-1)1(33)-1)-1)-1)1(22)-1)1(37)-1)1(43)-1)1(1
6)-1)1(46)-1)1(14)-1)1(3)-1)-1)-1)1(21)-1)-1)1(26)-1)1(10)-1)1(17)-1)
-1)-1)-1)1(9)-1)-1)-1)1(50)-1)-1)1(7)-1)-1)-1)-1)1(4)-1)-1)-1)1(13)-1
)-1)1(29)-1)1(25)-1)-1)";
```

```
struct Tree
{
    int w = -1;
    int h = -1;
    vector< pair< int, Tree * > > sons;
    friend std::ostream&operator<<(std::ostream &out, const Tree
&tree)
    {
        out << "(";
        if (tree.w != -1) {
            out << "[" << tree.w << ", " << tree.h << "]";
            for (auto nxt : tree.sons) {
                out << nxt.first << *(nxt.second);
            }
            out << "-1)";
        } else {
            out << tree.h << ")";
        }
        return out;
    }
    int height() const
    {
        int res = 0;
        for (auto nxt : sons) {
            int val = nxt.second->height() + 1;
            if (res < val) {
                res = val;
            }
        }
        return res;
    }
    ~Tree()
    {
        for (auto nxt : sons) {
            delete nxt.second;
        }
    }
};

int
getValue(const char **s)
{
    while (**s != '-' && !isdigit(**s)) {
```

```
        ++(*s);
    }
    int res = 0;
    bool sign = false;
    if (**s == '-') {
        sign = true;
        ++(*s);
    }
    while (isdigit(**s)) {
        res = res * 10 + **s - '0';
        ++(*s);
    }
    ++(*s);
    return sign ? -res : res;
}

Tree *
construct(const char **s)
{
    Tree *res = new Tree();
    const char *ptr = *s;
    if (**s == '[') {
        res->w = getValue(&ptr);
        res->h = getValue(&ptr);
        if (res->w == -1 || res->h == -1) {
            cerr << "fail!" << endl;
            cerr << *s << endl;
        }
        int tmp = getValue(&ptr);
        while (tmp != -1) {
            Tree *new_son = construct(&ptr);
            res->sons.push_back(make_pair(tmp, new_son));
            tmp = getValue(&ptr);
        }
    } else { // List
        res->h = getValue(&ptr);
    }
    *s = ptr;
    return res;
}


int
```

```
main()
{
    const char *s1 = parse_string + 1;
    Tree *decision_tree = construct(&s1);
    while (decision_tree->w != -1) {
        int curX = decision_tree->w, curY = decision_tree->h;
        cout << "? " << curX << " " << curY << endl;
        int sum;
        cin >> sum;
        bool findVal = false;
        for (auto x : decision_tree->sons) {
            if (x.first == sum) {
                findVal = true;
                decision_tree = x.second;
                break;
            }
        }
        if (!findVal) {
            cerr << "Oops!" << endl;
            return 1;
        }
    }
    cout << "! " << decision_tree->h << endl;
    return 0;
}
```