

# Московская олимпиада школьников по информатике 2025–2026 9 класс (второй отбор)

4 янв 2026 г., 12:00 — 15 янв 2026 г., 23:59

100 баллов

## Будни авиадиспетчера

Не начинайте решение задач, пока вы не включили запись экрана с видео участника, см. регламент олимпиады, без корректной записи работа не будет засчитана

Максим работает авиадиспетчером в аэропорту с единственной взлетно-посадочной полосой (ВПП). Сегодня ему предстоит посадить  $n$  самолетов, и ваша задача — рассчитать время завершения этого процесса.

Правила посадки:

1. Продолжительность: Процесс посадки одного самолета занимает ровно  $b$  минут. В это время ВПП считается занятой.
2. Очередь и круги: Если самолет прибывает к аэропорту, а ВПП занята, он отправляется на дополнительный круг. Один круг занимает  $f$  минут. После завершения круга самолет снова оказывается у ВПП и проверяет её готовность.
3. Свободная полоса: Если в момент прибытия (или возвращения с круга) ВПП свободна, самолет немедленно приступает к посадке.
4. Конфликты: Если несколько самолетов оказываются у свободной ВПП одновременно, один из них (любой) начинает посадку, а остальные немедленно отправляются на дополнительный круг длительностью  $f$  минут.

Задание: Определите момент времени, когда последний самолет завершит свою посадку.

### Входные данные

Первая строка входных данных содержит три целых числа  $n$ ,  $b$  и  $f$  ( $1 \leq n \leq 1000$ ,  $1 \leq b, f \leq 10^9$ ).

Вторая строка входных данных содержит  $n$  целых чисел  $t_i$  ( $0 \leq t_i \leq 10^9$ ) — времена первоначального прибытия самолетов (в произвольном порядке).

### Выходные данные

Выведите одно число — время, в которое последний самолет закончит посадку.

#### Примеры

```
10 10 12
13 0 1 10 20 20 2 1 10 20
```

```
120
```

#### Ограничения

Время выполнения: 1 секунда

Память: 256 МВ

Код

C++

```
1 #include <iostream>
2 #include <map>
3 #include <set>
4 using namespace std;
5 #define ll long long
6 int main() {
7     ll n, b, f; cin >> n >> b >> f;
8     multiset<ll> s;
9     for (int i = 0; i < n; i++) {
10         ll x; cin >> x;
11         s.insert(x);
12     }
13     ll cur = 0;
14     while (s.size() != 0) {
15         ll x = *(s.begin());
16         s.erase(s.begin());
17         if (x >= cur) cur = x + b;
18         else {
19             ll t = (cur - x) / f;
20             if ((cur - x) % f != 0) t++;
21             s.insert(t * f + x);
22         }
23     }
24     cout << cur;
25 }
```

100 баллов

## Готовим к ЕГЭ по информатике старших братьев

*Не начинайте решение задач, пока вы не включили запись экрана с видео участника, см. регламент олимпиады, без корректной записи работа не будет засчитана*

В ЕГЭ по информатике может встретиться следующая задача. Дана последовательность прописных латинских букв и цифр. Определить максимальную длину подряд идущих цифр, сумма которых делится на  $k$ .

### Входные данные

Первая строка входных данных содержит одно целое положительное число  $k$ , на которое должна делиться сумма последовательности из цифр. ( $1 \leq k \leq 10^3$ ).

Вторая строка содержит непустую строку  $s$ . Строка состоит из прописных латинских букв и цифр. Длина строки  $s$  не превышает  $10^6$ .

### Выходные данные

Выведите одно целое число — длину искомой последовательности.

### Система оценивания

В этой задаче три группы тестов. В первой группе длина строки не превосходят 100, во второй группе — 1000, в третьей группе нет дополнительных ограничений. Баллы за группу можно получить только при прохождении всех тестов группы и всех тестов предыдущих групп.

#### Примеры

```
3
ab99c112111d
```

```
5
```

```
3
ab7c11d
```

```
0
```

```
1
x999ууу0123456789
```

```
10
```

#### Ограничения

Время выполнения: 3 секунды

Память: 256 МВ

Код

C++

```

1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  #include <cmath>
5  #include <map>
6  #include <set>
7  #include <unordered_map>
8  #include <queue>
9
10 using namespace std;
11
12 #define int long long
13
14 bool isdigit(char c){
15     .... return c >= '0' && c <= '9';
16 }
17
18 signed main()
19 {
20     .... ios::sync_with_stdio(0);
21     .... cout.tie(0);
22     .... cin.tie(0);
23     ....
24     .... int k;
25     .... cin >> k;
26     ....
27     .... string s;cin >> s;
28     .... int n = s.size();
29     ....
30     .... int ans = 0;
31     .... for(int i = 0; i < n; i++){
32         .... if(!isdigit(s[i]))continue;
33         ....
34         .... int j = i;
35         .... int sum = s[i] - '0';
36         .... unordered_map<int, int> mp;
37         .... mp[0] = i - 1;
38         ....
39         .... if(sum % k == 0){
40             .... ans = max(ans, 1ll);
41         .... }
42         .... else{
43             .... mp[sum%k] = i;
44         .... }
45         ....
46         .... while(j + 1 < n && isdigit(s[j+1])){
47             .... j++;
48             .... sum += s[j] - '0';
49             .... if(mp.count(sum % k))ans = max(ans, j - mp[sum%k]);
50             .... else mp[sum%k] = j;
51         .... }
52         ....
53         .... i = j;
54     .... }
55     ....
56     .... cout << ans;
57     .... return 0;
58 }

```

100 баллов

## Восстановление исходного массива

Не начинайте решение задач, пока вы не включили запись экрана с видео участника, см. регламент олимпиады, без корректной записи работа не будет засчитана

Представьте, что у вас был массив из  $N$  натуральных чисел  $a_1, a_2, \dots, a_N$ . На его основе вы построили таблицу  $N \times N$  по следующим правилам:

1. В ячейках на главной диагонали (где номер строки  $i$  совпадает с номером столбца  $j$ ) всегда записывается 0.
2. В остальных ячейках (где  $i \neq j$ ) записывается максимальное из двух чисел:  $a_i$  и  $a_j$ .

Вам дана уже готовая таблица. Требуется «расшифровать» её и найти исходный массив  $a_1, \dots, a_N$ . Если такая таблица не могла быть получена по этим правилам, выведите  $-1$ .

### Входные данные

В первой строке находится число  $N$  ( $1 \leq N \leq 500$ ) — размер таблицы. В последующих  $N$  строках находятся по  $N$  целых чисел (от 0 до 1000) — сама таблица.

### Входные данные

Если решение существует, выведите  $N$  чисел через пробел. Если решений несколько, выведите любое. Если решения нет, выведите  $-1$ .

#### Примеры

```
3
0 4 6
4 0 6
6 6 0
```

```
4 4 6
```

#### Ограничения

Время выполнения: 1 секунда

Память: 256 MB

Код

C++

```

1  #include <iostream>
2  #include <algorithm>
3  #include <cmath>
4  #include <string>
5  #include <vector>
6  using namespace std;
7  typedef long long ll;
8  const int MAX_N = 2e3+6;
9  const double INF = 1e9+1;
10 int main() {
11     int n; cin >> n;
12     vector<vector<int>> matrix(n, vector<int>(n));
13     for (int i = 0; i < n; i++) {
14         for (int j = 0; j < n; j++) {
15             cin >> matrix[i][j];
16         }
17     }
18     vector<int> a(n);
19     for (int i = 0; i < n; i++) {
20         int mn = 100001;
21         for (int j = 0; j < n; j++) {
22             if (j == i) continue;
23             mn = min(mn, matrix[i][j]);
24             if (matrix[i][j] != matrix[j][i]) {
25                 cout << -1 << endl;
26                 return 0;
27             }
28         }
29         if (mn == 100001 || mn == 0) mn = 1;
30         a[i] = mn;
31         if (matrix[i][i] != 0) {
32             cout << -1 << endl;
33             return 0;
34         }
35     }
36     for (int i = 0; i < n; i++) {
37         for (int j = 0; j < n; j++) {
38             if (i == j) continue;
39             if (matrix[i][j] != max(a[i], a[j])) {
40                 cout << -1 << endl;
41                 return 0;
42             }
43         }
44     }
45     for (int x : a) {
46         cout << x << ' ';
47     }
48 }

```

100 баллов

## Реверс и неподвижные точки

*Не начинайте решение задач, пока вы не включили запись экрана с видео участника, см. регламент олимпиады, без корректной записи работа не будет засчитана*

В руке у Мате находится  $N$  карт, на которых написаны числа — перестановка чисел от 1 до  $N$ . Карты лежат в фиксированном порядке. Неподвижной точкой называется ситуация, когда число на карте совпадает с её порядковым номером в руке (считая слева направо с 1).

Домагой может один раз выбрать любой непрерывный отрезок карт и развернуть его (первая карта отрезка меняется местами с последней, вторая – с предпоследней и так далее).

Задание: Помогите Домагой выбрать такой отрезок для разворота, чтобы общее количество неподвижных точек во всей руке стало максимально возможным.

Решения, верно работающие при  $N \leq 500$  наберут не менее 30 баллов.

Решения, верно работающие при  $N \leq 5000$  наберут не менее 60 баллов.

## Входные данные

Число  $N$  ( $1 \leq N \leq 5 \cdot 10^5$ ). Перестановка из  $N$  различных целых чисел от 1 до  $N$ .

## Выходные данные

Выведите через пробел два числа: значение первой и значение последней карты в выбранном оптимальном отрезке.

Если решений несколько, выведите любое.

Если разворот не может увеличить число точек, можно вывести значения, соответствующие отрезку из одной любой карты.

### Примеры

```
4
3 2 1 4
```

```
3 1
```

```
2
1 2
```

```
1 1
```

```
7
3 6 5 7 4 1 2
```

```
6 1
```

### Ограничения

Время выполнения: 1 секунда

Память: 256 МВ

Код

C++

```

1  #include <bits/stdc++.h>
2
3  #define FOR(i, a, b) for (int (i) = (a); (i) < (b); i++)
4  #define REP(i, n) FOR(i, 0, n)
5  #define TRACE(x) cerr << #x << " " << x << endl
6  #define eb emplace_back
7  using namespace std;
8
9  const int MaxN = 1000100;
10
11 //everything is 1-indexed
12
13 int N;
14 int perm[MaxN];
15 vector <int> radii[2 * MaxN]; // center and radius are doubled
16 int prefFixedPoints[MaxN];
17
18
19 void findSegmentBounds(const int center, const int radius, int&
left, int& right){
20     left = (center - radius) / 2;
21     right = (center + radius) / 2;
22 }
23
24 int fixedPointsInSegment(int a, int b){ // [a, b]
25     return prefFixedPoints[b] - prefFixedPoints[a - 1];
26 }
27
28 void precomputePrefFixedPoints(){
29     for (int i = 1; i <= N; i++)
30         prefFixedPoints[i] = prefFixedPoints[i - 1] + (
perm[i] == i );
31 }
32
33 void findCenters(){
34     for (int i = 1; i <= N; i++)
35         radii[perm[i] + i].emplace_back(abs(perm[i]
- i));
36
37     for (int center = 2; center <= N + N; center++){ // all
possible centers
38         sort(radii[center].begin(), radii[center].end());
// the log in the complexity may be evaded by smart insertion
39     }
40 }
41
42 void findBestFlip(){
43     int targetMax = -MaxN, targetLeft = -1, targetRight = -1;
44     for (int center = 2; center <= N + N; center++){
45         int numOfCreatedFixedPoints = 0;
46         for (auto radius: radii[center]){
47             numOfCreatedFixedPoints++;
48
49             int left, right;
50             findSegmentBounds(center, radius, left,
right);
51
52             int numOfLostFixedPoints =
fixedPointsInSegment(left, right); // the center as the fixed
point is counted both as a lost and created
53             int res = numOfCreatedFixedPoints -
numOfLostFixedPoints; // so the net gain from it is zero

```

```
54
55         if (targetMax < res){
56             targetMax = res;
57             targetLeft = left;
58             targetRight = right;
59         }
60     }
61 }
62
63     printf("%d %d\n", perm[targetLeft], perm[targetRight]);
64 }
65
66 void load(){
67     scanf("%d", &N);
68     for (int i = 1; i <= N; i++)
69         scanf("%d", perm + i);
70 }
71
72 int main(){
73     load();
74     precomputePrefFixedPoints();
75     findCenters();
76     findBestFlip();
77     return 0;
78 }
```

100 баллов

## Пересчитай лестницы

Не начинайте решение задач, пока вы не включили запись экрана с видео участника, см. регламент олимпиады, без корректной записи работа не будет засчитана

Мемориальную лестницу было решено изготовить в виде возрастающей по высоте последовательности каменных ступенек. То есть каждая следующая ступенька строго выше предыдущей. Пусть у нас есть  $n$  каменоломен, в каждой из которых изготовлено ровно  $k$  каменных ступенек. Из каждой каменоломни было решено взять ровно одну ступеньку, причём ступенька, выбранная из  $i$ -й каменоломни, должна быть ниже, чем ступенька, выбранная из  $(i + 1)$ -й. Определите количество способов выбрать ступеньки в каменоломнях таким образом.

Ответ может быть очень большим, посчитайте его по модулю 998244353.

### Входные данные

В первой строке входных данных расположены два целых числа  $n$  и  $k$  — число каменоломен и количество ступенек в каждой из них,  $1 \leq n \leq 100$  и  $1 \leq k \leq 10\,000$ .

В следующих  $n$  строках находится описание ступенек в каменоломнях. Каждая каменоломня описывается  $k$  целыми числами в одной строке  $h_1, \dots, h_k$  — высоты ступенек в ней,  $1 \leq h_1 < h_2 < \dots < h_k \leq 10^9$ .

### Выходные данные

Выведите единственное целое число — количество возрастающих лестниц по модулю 998244353.

### Система оценивания

В этой задаче три группы тестов. В первой группе  $n$  и  $k$  не превосходят 10, во второй группе — 100, в третьей группе нет дополнительных ограничений. Баллы за группу можно получить только при прохождении всех тестов группы и всех тестов предыдущих групп.

#### Примеры

```
2 2
2 3
1 5
```

```
2
```

```
2 4
5 6 7 8
1 2 3 4
```

```
0
```

#### Ограничения

Время выполнения: 2 секунды

Память: 256 МВ

Код

C++

```

1  #define _CRT_SECURE_NO_WARNINGS
2
3  #pragma comment(linker, "/stack:200000000")
4  #pragma GCC optimize("Ofast")
5  #pragma GCC
   target_tbl("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,avx,tune=native
   ")
6
7
8  // #include "stdafx.h"
9  #include "iostream"
10 #include "fstream"
11 #include "algorithm"
12 #include "iomanip"
13 #include "stack"
14 #include "queue"
15 #include "string"
16 #include "vector"
17 #include "map"
18 #include "set"
19 #include "list"
20 #include "deque"
21 #include "complex"
22 #include "bitset"
23 #include "cmath"
24 #include "unordered_set"
25 #include "unordered_map"
26 #include "iterator"
27 #include <ctime>
28 #include <cassert>
29 #include "numeric"
30 #include <cstdio>
31 #include "random"
32 #include "chrono"
33 #include "cstring"
34
35
36 using namespace std;
37
38 #define maxi(a,b) a = max(a, b);
39 #define mini(a,b) a = min(a, b);
40
41 ////////////////
42 #define endl '\n'
43 ////////////////
44
45 #define all(a) a.begin(), a.end()
46 #define rall(a) a.rbegin(), a.rend()
47 #define sqr(x) ((x) * (x))
48 #define SZ(a) ((int)(a.size()))
49 #define watch(x) cout << (#x) << " = " << x << endl;
50 //typedef long long ll;
51 typedef long double ld;
52
53 // #define int unsigned int
54 #define int long long
55 #define double ld
56 typedef pair<int, int> pii;
57 typedef pair<double, double> pdd;
58 typedef vector<int> vi;
59 typedef vector<double> vd;
60 typedef vector<pii> vpii;

```

```

61 typedef vector<vi> vvi;
62
63 template<class T>
64 void show(const vector<T> &a)
65 {
66     for (T x : a)
67         cout << x << " ";
68     cout << endl;
69 }
70 mt19937
   rng(chrono::steady_clock::now().time_since_epoch().count());
71 mt19937_64
   rng_64(chrono::steady_clock::now().time_since_epoch().count());
72
73 int xx[8] = { 1, -1, 0, 0, 1, -1, 1, -1 };
74 int yy[8] = { 0, 0, 1, -1, -1, -1, 1, 1 };
75 string dir = "RDLU";
76 string travel = "ENWS";
77
78 const int N = 4e5 + 50, oo = 1e18 + 500;
79 const int mod = 998244353;
80 //const int mod = 1e9 + 7;
81 //const int mod = 1e9 + 9;
82 const int M2 = 1000000093, x2 = 27162;
83 const int M1 = 1000000087, x1 = 241;
84 const double eps = 1e-18, PI = 2 * acos(0.0);
85
86
87 signed main()
88 {
89     ios::sync_with_stdio(0);
90     cout.tie(0); cin.tie(0);
91     //freopen("input.txt", "r", stdin);
92     //freopen("output.txt", "w", stdout);
93     //freopen("paintbarn.in", "r", stdin);
94     //freopen("paintbarn.out", "w", stdout);
95
96     int n, k;
97     cin >> n >> k;
98
99     vvi a(n, vi(k));
100    for (int i = 0; i < n; i++)
101    {
102        for (int j = 0; j < k; j++)
103        {
104            cin >> a[i][j];
105        }
106    }
107
108    vvi dp(n, vi(k, 0));
109    for (int j = 0; j < k; j++)
110    {
111        dp[0][j] = j + 1;
112    }
113
114    for (int i = 1; i < n; i++)
115    {
116        for (int j = 0; j < k; j++)
117        {
118            int jj = lower_bound(all(a[i - 1]), a[i]
119 [j]) - a[i - 1].begin();
120            jj--;

```

```
121         if (jj >= 0)
122         {
123             dp[i][j] += dp[i - 1][jj];
124             if (dp[i][j] >= mod)
125                 dp[i][j] -= mod;
126         }
127     }
128
129     for (int j = 1; j < k; j++)
130     {
131         dp[i][j] += dp[i][j-1];
132         if (dp[i][j] >= mod)
133             dp[i][j] -= mod;
134     }
135 }
136
137 /*int ans = 0;
138 for (int j = 0; j < k; j++)
139 {
140     ans = (ans + dp[n - 1][j]) % mod;
141 }
142 cout << ans;*/
143
144 cout << dp[n - 1][k - 1];
145 }
```

100 баллов

## Минимальное не встречающееся число

Не начинайте решение задач, пока вы не включили запись экрана с видео участника, см. регламент олимпиады, без корректной записи работа не будет засчитана

Есть  $n$  целых неотрицательных чисел  $a_1, \dots, a_n$ . Нужно узнать, есть ли подмассив (часть массива, состоящая из нескольких подряд идущих элементов массива), минимальное не встречающееся в нем число (MEX) равно  $i$ .

MEX-ом от набора чисел называется наименьшее неотрицательное число, которое не встречается в наборе. Например  $MEX(1, 2, 2, 3, 5) = 0$ , а вот  $MEX(1, 2, 2, 0, 3, 5) = 4$ .

Интересуют ответы на такие вопросы для всех  $i$  от 0 до  $n$ .

### Входные данные

Первая строка содержит единственное целое число  $n$  — количество чисел. ( $1 \leq n \leq 10^5$ ). Вторая строка содержит  $n$  целых чисел —  $a_1, \dots, a_n$ . ( $0 \leq a_i \leq n - 1$ ).

### Выходные данные

В единственной строке выведите  $n + 1$  целых чисел — ответы на вопросы. Если ответ на  $i$ -ый вопрос <<Да>>, то соответствующее число должно быть 1. Если ответ на  $i$ -ый вопрос <<Нет>>, то соответствующее число должно быть 0.

### Система оценивания

В этой задаче четыре группы тестов. В первой группе  $n$  не превосходит 50, во второй — 500, в третьей — 1000. Баллы за группу можно получить только при прохождении всех тестов группы и всех тестов предыдущих групп.

#### Примеры

```
3
0 1 2
```

```
1 1 1 1
```

#### Ограничения

Время выполнения: 1 секунда

Память: 256 МВ

Код

C++

```

1  #define _CRT_SECURE_NO_WARNINGS
2
3  #include <random>
4  #include <iterator>
5  #include <chrono>
6  #include <iostream>
7  #include <fstream>
8  #include <vector>
9  #include <map>
10 #include <set>
11 #include <algorithm>
12 #include <math.h>
13 #include <iomanip>
14 #include <unordered_map>
15 #include <cassert>
16 #include <string>
17
18 using namespace std;
19
20
21 typedef unsigned long long ull;
22 typedef long long ll;
23 typedef double ld;
24
25
26 // #define int ll
27 #define rand shitTTTTTY_SHIT
28 mt19937
   rng(chrono::steady_clock::now().time_since_epoch().count());
29 mt19937_64
   rng_64(chrono::steady_clock::now().time_since_epoch().count());
30
31
32 typedef pair<int, int> pii;
33 typedef pair<string, string> pst;
34 typedef pair<pii, pii> piii;
35 typedef vector<piii> vpii;
36 typedef pair<ll, ll> pll;
37 typedef vector<int> vi;
38 typedef vector< vi > vvi;
39 typedef vector< vvi > vvvi;
40 typedef vector<pii> vpi;
41 typedef vector<short> vs;
42 typedef vector<vs> vvs;
43 typedef vector<vvs> vvvs;
44 typedef vector<ll> vl;
45 typedef vector<vl> vvl;
46 typedef vector<vvl> vvvl;
47 typedef vector<ld> vld;
48 typedef vector<vld> vvld;
49 typedef vector<vvld> vvvld;
50 typedef vector<string> vst;
51 typedef vector<vst> vvst;
52 typedef pair<ld, ld> pld;
53 typedef pair<pld, pld> pldd;
54 typedef vector<pld> vpd;
55 //typedef complex<double> base;
56
57
58 #define inmin(a, b) a = min(a, (b))
59 #define inmax(a, b) a = max(a, (b))
60 #define my_abs(a) (((a) >= 0) ? (a) : -(a))

```

```

61 #define mp(a, b) make_pair(a, (b))
62 #define ALL(a) a.begin(),a.end()
63 #define RALL(a) a.rbegin(),a.rend()
64 #define sqr(x) ((x) * (x))
65 #define fori(i, n) for(int i = 0; i < int(n); ++i)
66 #define SZ(a) ((int)((a).size()))
67 #define MODIK(a) (a >= M ? a - M : a)
68 #define triple(T) tuple<T, T, T>
69 #define quad(T) tuple<T, T, T, T>
70 #define watch(x) cout << (#x) << " = " << x << endl;
71 #ifndef ART_HOME
72 #define cerr cout
73 #else
74 #define cerr if (false) cerr
75 #endif
76
77 // #define cerr if (false) cerr
78
79 const double PI = 2 * acos(0.0);
80
81 const string DIGITS = "0123456789";
82 const string ALPH = "abcdefghijklmnopqrstuvwxyz";
83
84 istream& operator >> (istream& in, pii& a)
85 {
86     in >> a.first >> a.second;
87     return in;
88 }
89
90
91 ostream& operator << (ostream& out, pii& a)
92 {
93     out << a.first << ' ' << a.second;
94     return out;
95 }
96
97
98 istream& operator >> (istream& in, pld& a)
99 {
100     in >> a.first >> a.second;
101     return in;
102 }
103
104
105 ostream& operator << (ostream& out, pld& a)
106 {
107     out << a.first << ' ' << a.second;
108     return out;
109 }
110
111
112 template <class T0, class T1>
113 inline ostream& operator << (ostream& out, pair<T0, T1>& a) {
114     return out << "{" << a.first << ", " << a.second << "}";
115 }
116
117 template <class T0, class T1, class T2>
118 inline ostream& operator << (ostream& out, tuple<T0, T1, T2>& a) {
119     return out << "{" << get<0>(a) << ", " << get<1>(a) << ", " <<
get<2>(a) << "}";
120 }
121
122 template <class T0, class T1, class T2, class T3>

```

```

123 inline ostream& operator << (ostream& out, tuple<T0, T1, T2, T3>&
    a) {
124     return out << "{" << get<0>(a) << ", " << get<1>(a) <<
125     ", " << get<2>(a) << ", " << get<3>(a) << "}";
126 }
127
128 template<class T>
129 inline ostream& operator << (ostream& out, vector<T>& a) {
130     out << "[";
131     fori(i, a.size())
132         out << a[i] << vector<string>{"", ",", ""}[i + 1 ==
    a.size()] ;
133     return out;
134 }
135
136
137 void print(int a, int b, int n, vi h)
138 {
139     ofstream cout("output.txt");
140     cout << SZ(h) << " " << a << " " << b << "\n";
141     for (int i = 0; i < SZ(h); ++i) {
142         cout << h[i] << " ";
143     }
144     cout.close();
145     exit(0);
146 }
147
148
149 inline bool break_time(ld time)
150 {
151     return clock() > CLOCKS_PER_SEC * time;
152 }
153
154
155 void bonds()
156 {
157     ld price, nkd, kd, year;
158     int cnt;
159     cout << "price nkd kd cnt year\n";
160     cout << endl;
161     cin >> price >> nkd >> kd >> cnt >> year;
162     cout << 100 * ((500 + kd * cnt - price * 5 - nkd) / (price * 5)
    / year);
163 }
164
165
166 ld get_rand_ld(ld l, ld r)
167 {
168     uniform_real_distribution<double> gen(l, r);
169     return gen(rng);
170 }
171
172
173 bool may(int E, ld T)
174 {
175     if (E < 0)
176         return 1;
177
178     double eps = get_rand_ld(0, 1);
179     return eps < exp(-E / T);
180 }
181
182

```

```

183
184 void smain();
185
186 signed main() {
187     ios::sync_with_stdio(0);
188     cin.tie(0); cout.tie(0);
189
190 #ifdef ART_HOME
191     freopen("input.txt", "r", stdin);
192     clock_t start = clock();
193 #else
194     // freopen("greedy.in", "r", stdin);
195     // freopen("greedy.out", "w", stdout);
196 #endif
197     cout << setprecision(12) << fixed;
198     smain();
199 #ifdef ART_HOME
200     cout << "\nTOTAL EXECUTION TIME: " << float(clock() - start) /
CLOCKS_PER_SEC << endl;
201 #endif
202 }
203
204
205 const int N = 1 << 17, K = 1e6 + 100, M = 1e9 + 7, LOG = 18, A =
26, oo = 1e9, B = 30;
206 const int M1 = 1000000093, x1 = 27162;
207 const int M2 = 1000000087, x2 = 241;
208 const ld EPS = 1e-9;
209
210
211 int t[N << 1];
212 vi pos[N];
213
214
215 int get(int l, int r, int lb, int rb, int ind) {
216     if (l == lb && r == rb)
217         return t[ind];
218
219     int m = lb + rb >> 1;
220     int first = 0, second = 0;
221
222     if (l <= m)
223         first = get(l, min(m, r), lb, m, ind << 1);
224
225     if (r > m)
226         second = get(max(l, m + 1), r, m + 1, rb, ind << 1 | 1);
227
228     return max(first, second);
229 }
230
231
232 void update(int ind, int v) {
233     ind += N - 1;
234     t[ind] = v;
235     ind >>= 1;
236
237     while (ind) {
238         t[ind] = max(t[ind << 1], t[ind << 1 | 1]);
239         ind >>= 1;
240     }
241 }
242
243

```

```

244 vi solve(vi kek) {
245     for (int i = 0; i < N; ++i)
246         pos[i].clear(), pos[i].push_back(N - 1);
247
248     int n = SZ(kek);
249
250     for (int i = n - 1; i >= 0; --i)
251         pos[kek[i]].push_back(i);
252
253     for (int i = 0; i < N; ++i)
254         t[i + N] = pos[i].back();
255
256     for (int i = N - 1; i; --i)
257         t[i] = max(t[i << 1], t[i << 1 | 1]);
258
259     set<int> flex;
260     for (int i = 0; i <= n; ++i)
261         flex.insert(i);
262
263     vi mex(n), ans(n + 1);
264     for (int i = 0; i < n; ++i) {
265         flex.erase(kek[i]);
266         mex[i] = *flex.begin();
267     }
268
269     for (int v : mex)
270         ans[v] = 1;
271
272     for (int v : kek)
273         if (v)
274             ans[0] = 1;
275
276     for (int v : kek) {
277         pos[v].pop_back();
278
279         if (v) {
280             int ps = get(1, v, 1, N, 1);
281             if (ps < pos[v].back())
282                 ans[v] = 1;
283         }
284
285         update(v + 1, pos[v].back());
286     }
287
288     return ans;
289 }
290
291
292
293 vi stupid(vi kek) {
294     int n = SZ(kek);
295     vi ans(n + 1);
296
297     for (int i = 0; i < n; ++i) {
298         set<int> flex;
299         for (int x = 0; x <= n; ++x)
300             flex.insert(x);
301
302         for (int j = i; j < n; ++j) {
303             flex.erase(kek[j]);
304             ans[*flex.begin()] = 1;
305         }
306     }

```

```

307
308     return ans;
309 }
310
311
312 void smain() {
313     //int t = 0;
314     //while (1) {
315         //t++;
316         //if (t % 1000 == 0)
317             cout << t << endl;
318
319         //int n = 20;
320         //vi kek(n);
321         //for (int i = 0; i < n; ++i)
322             kek[i] = rng() % n;
323
324         //if (solve(kek) != stupid(kek)) {
325             //for (int v : kek)
326                 cout << v << ' ';
327
328             //cout << "\nSolve : ";
329             //for (int v : solve(kek))
330                 cout << v << ' ';
331             //cout << "\nStupid : ";
332             //for (int v : stupid(kek))
333                 cout << v << ' ';
334
335             //cout << '\n';
336             //assert(solve(kek) == stupid(kek));
337         //}
338     //}
339
340     int n;
341     cin >> n;
342
343     vi kek(n);
344     for (int i = 0; i < n; ++i)
345         cin >> kek[i];
346
347     vi ans = solve(kek);
348     for (int v : ans)
349         cout << v << ' ';
350 }
351

```