康托展开

Cantor Expansion

康托展开给出了全排列到自然数的一个双射，具体的，第 i 个全排列与自然数 i 相对应。

注意：$12\cdots n$ 算第 0 个全排列。

康托展开

设全排列 $a_1a_2\cdots a_n$，欲得到它的排名（从 0 开始计数），只需统计小于它的全排列有多少个。逐位考虑，小于它的全排列有两种情形：

- 当前位更小，后面的位任意在未出现的数字中进行排列，共 $b_i(n-i)!$ 种，其中 b_i 表示 $<a_i$ 的且在 $a_1\ldots a_{i-1}$ 中没有出现的数的个数。
- 当前位相等，问题转化成子问题。

综上，有康托展开：

$$X=b_1(n-1)!+b_2(n-2)!+\cdots+b_n0!=\sum_{i=1}^n b_i(n-i)!$$

Implement：$<a_i$ 的且在 $a_1\cdots a_{i-1}$ 中没有出现的数的个数，这个树状数组即可。

Complexity：$O(n\lg n)$

Code：

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

using namespace std;

const int MOD = 998244353;
const int N = 1000005;

int n, a[N], fact[N], ans;

int c[N];
inline int lowbit(int x){ return x & -x; }
inline void add(int x, int val){
    while(x <= n){
        c[x] += val;
        x += lowbit(x);
    }
}
inline int sum(int x){
    int res = 0;
    while(x){
        res += c[x];
        x -= lowbit(x);
    }
    return res;
}

int main(){
    fact[0] = 1; for(int i = 1; i <= 1000000; i++) fact[i] = 1ll * fact[i-1] * i % MOD;
    scanf("%d", &n);
    for(int i = 1; i <= n; i++){
        scanf("%d", &a[i]);
        ans = (ans + 1ll * (a[i] - sum(a[i]) - 1) * fact[n-i] % MOD) % MOD;
        add(a[i], 1);
```

```
    }
    printf("%d\n", ans + 1);
    return 0;
}
```

逆康托展开

给定自然数 X，找到排名 X 的全排列（注意这里所谓排名是从 0 开始计数的）。

排名为 X，说明有 X 个小于它的全排列。逐位考虑，第 i 位小于它的全排列最多 $b_i(n-i)!$ 个，所以 $\left\lfloor \frac{X}{(n-i)!} \right\rfloor$ 就是 b_i；随后 X 减去 $b_i(n-1)!$，变成相同的子问题，继续进行即可。

现在得到了 b_i 如何得到 a_i？也就是说，我要找到当前**未出现的数中**第 b_i+1 小的数是多少。线段树或者树状数组倍增可以做到 $O(n\lg n)$，二分+树状数组可以 $O(n\lg^2 n)$.