

# Week 12

## 比赛简记

### Max.D.

#### 专题

暂无

#### 比赛

一场cf edu, 好惨好惨

#### 题目

暂无

### Hardict

#### 专题

无

#### 比赛

cf div2

#### 题目

暂无

### MountVoom

#### 专题

无

## 比赛

cf edu, 算错了复杂度浪费了半个小时。

## 题目

无

## 个人总结

### 陈铭煊 Max.D.

补题+学习

### 龙鹏宇 Hardict

补题+整理板子

### 肖思炀 MountVoom

补题, 学点新东西

## 本周推荐

### 陈铭煊 Max.D.

来源 :

[Codeforces Round 662\(Div. 2\) E2 Twilight and Ancient Scroll \(harder version\)](#)

标签 :

字符串处理, 动态规划, 哈希, 双指针

**题意：**

给出  $n$  个非空字符串，每个长  $L_i$ 。现在让你在每个字符串中选择删除一个字符，或者是不删除，使得得到的字符串组按顺序字典序不减，求符合要求的操作的个数，答案对  $1000000007$  取模。

$$1 \leq n \leq 10^5, \sum L_i \leq 10^6$$

**题解：**

我们容易想到设置  $dp$  状态为  $dp[i][j]$  表示到了第  $i$  个字符串删除第  $j$  ( $0 \leq j \leq L_i$ ) 个字符，满足前  $i$  个不减的种类数（其中  $j=L_i$  代表不删除）。这个时候我们会发现即使用一些预处理来快速比较相邻两个删除后的字符串，由于更新时要枚举位置，所以效率不会比  $O(n \cdot \sum L_i)$  更优。

这时候有个很精彩的技巧，我们如果能够预处理出每个字符串删除之后得到的字符串的一个字典序数组，那么  $dp$  更新时等价于做一个区间加操作——对于前状态的字符串，用双指针找到第一个比其大的后状态，接着后面的所有状态都可以加上这个前状态的值。

怎么预处理呢？通过双指针求出一个  $nxt$  数组  $nxt[i]$  代表第  $i$  个字符之后第一个出现不同的位置。那么对于两个删除点  $p, q$  我们就可以通过对  $nxt$  和其大小的讨论，对状态字符串进行排序。

```
for (int i = 0; i < res.size(); i++)
    res[i] = i;
sort(res.begin(), res.end(), [&](int x, int y) {
    bool f = true;
    if (x > y)
        swap(x, y), f = false;
    if (nxt[x] <= y) {
        f ^= s[nxt[x]] > s[x];
    } else
        f = !f;
    return f;
});
```

剩下还有一个问题，怎么快速比较前后两个状态字符串大小呢？这里用哈希求出公共前缀更方便，当然用后缀数组也可以。

因此总的效率为  $O((n + \log \sum L_i) \cdot \log \sum L_i)$

**评论：**

离散动态规划我们可以将状态排序使之紧凑，很好的思路。

## 龙鹏宇 Hardict

**来源：**

[HDU 6061](#)

标签：

多项式卷积,NTT

题意：

先给定一个  $f(x) = \sum_{i=0}^n c_i x^i$ , 然后求解  $g(x) = f(x - \sum_{i=1}^n a_i) = \sum_{i=0}^n b_i x^i$

输出  $b_i$

题解：

$A = -\sum a_i$  并取模变成求解  $f(x+A)$  少去正负计算

然后按  $f(x+A)$  每一项进行一个展开(会成一个三角表)

目标多项式系数  $b_j = \sum_{i \geq j} c_i A^{i-j} C_i^j$

化解后有  $j! A^j b_j = \sum_{i=j}^n \frac{c_i i! A^i}{(i-j)!}$

整体上  $g(x) = \sum_{i=0}^n \frac{x^i}{i!} A^i \sum_{j=0}^{n-i} \frac{c_{i+j} (i+j)! A^{i+j}}{j!}$

这里令  $k = n - (i+j)$ ,  $g(x) = \sum_{i=0}^n \frac{x^i}{i!} A^i \sum_{j+k=n-i} \frac{c_{n-k} (n-k)! A^{n-k}}{j!}$

$= \sum_{k=0}^n \frac{x^k}{k!} A^k \sum_{n+k=i+j} \frac{c_i (i)! A^i}{(n-j)!}$

将  $\frac{1}{j!}$  当作一个翻转(两者都可), 求  $\sum_{i=0}^n c_i i! A^i x^i$  与  $\sum_{i=0}^n \frac{1}{(n-i)!} x^i$  的卷积, 然后取结果  $n+i$  项系数即可

## 肖思炀 MountVoom

来源：

[D - Moving Piece](#)

标签：

枚举, 贪心

题意：

给定步数和诸多环，求在某个环上走的最大收益。

**题解：**

枚举每个点作为起点然后分类讨论，如果步数不够一圈就模拟一下找一个最大值。

如果够一圈则在模拟走一圈内的最大值和走多圈和余下步数的最大值取个最大值即可。

**评论：**

分类没有处理好WA了很多发，这种简单题也要注意细节。

From:

<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:

[https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:alchemist:weekly\\_digest\\_12&rev=1598596589](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:alchemist:weekly_digest_12&rev=1598596589)



Last update: **2020/08/28 14:36**