

CF577B Modulo Sum

Description

给出 1 个长度为 n 的序列, 以及 1 个正整数 m 。问这个原序列中是否存在非空子序列, 使其元素之和能被 m 整除。

$n \leq 10^6, m \leq 10^3$ 。

CF577B Modulo Sum

Solution

首先发现这是一个非常sb的01背包模板.

时间 $O(nm)$ 空间 $O(nm)$ 就不用想了.

CF577B Modulo Sum

Solution

如果存在

$$(a_1 + \dots + a_r) \bmod m = 0,$$

$$\text{then } (sum_r - sum_{1-1}) \bmod m = 0$$

$$\rightarrow (sum_r \bmod m - sum_{1-1} \bmod m + m) \bmod m = 0$$

$$\rightarrow sum_r \bmod m = sum_{1-1} \bmod m$$

CF577B Modulo Sum

Solution

如果 $n > m$

由抽屉原理得 $\text{sum}_i \bmod m$ 必有重复。

那么一定存在 l 和 r , 使得 $\text{sum}_r \bmod m = \text{sum}_{l-1} \bmod m$

所以说, $n > m$ 时答案一定是 YES (其实 $n = m$ 也可, 为什么?)

$n \geq m$



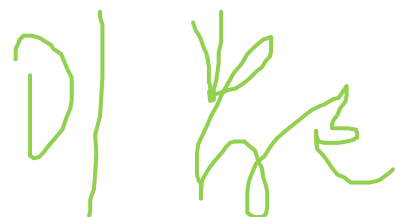
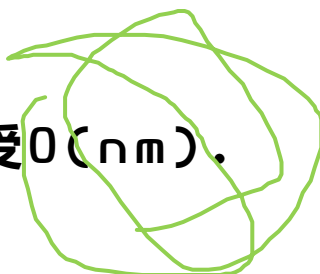
CF577B Modulo Sum

Solution

如果 $n < m$

则 $n < m \leq 1000$,

时间和空间都可以承受 $O(nm)$.



CF577B Modulo Sum

Code:

```
rep(i, 1, n + 1) {  
    f[i][a[i] % m] = 1;  
    rep(j, 0, m) {  
        f[i][j] |= f[i - 1][j];  
        f[i][(j + a[i]) % m] |= f[i - 1][j];  
    }  
    ok |= f[i][0];  
}
```

CF1359D Yet Another Yet Another Task

Description

给一个序列，求一个子串，使得子串和减掉子串中最大值的差最大，输出最大值。

$n \leq 100000, -30 \leq a[i] \leq 30$

CF1359D Yet Another Yet Another Task

Solution

这 $a[i]$ 如此小, 肯定有诈!

考虑枚举子串内最大值. 对于每个值, 我们跑一下区间内的最大子串和, 这个题就写完了.

$O(n \times \max a[i])$

— m!

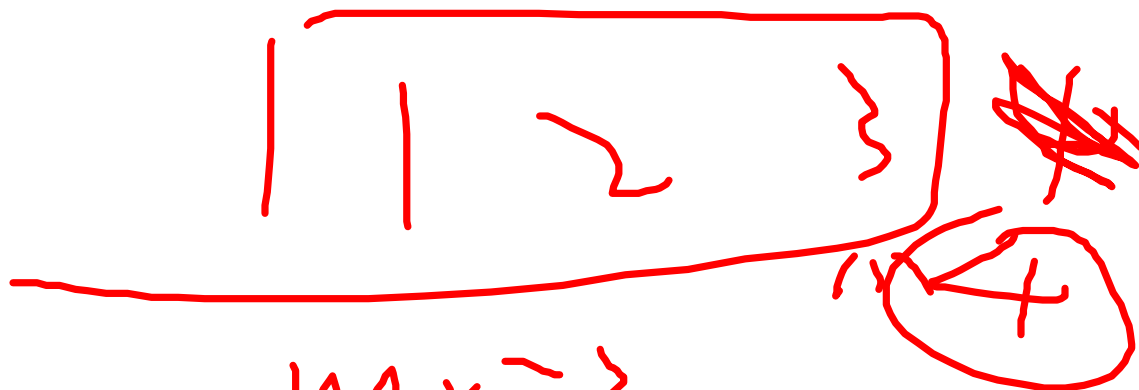
CF1359D Yet Another Yet Another Task

Solution

但是我们枚举 mx 的时候要考虑:

1. 当前子串中~~最大值~~ $\leq mx$
2. 当前子串中~~最大值~~ $> mx$

$$mx = 4$$



$$mx = 3$$

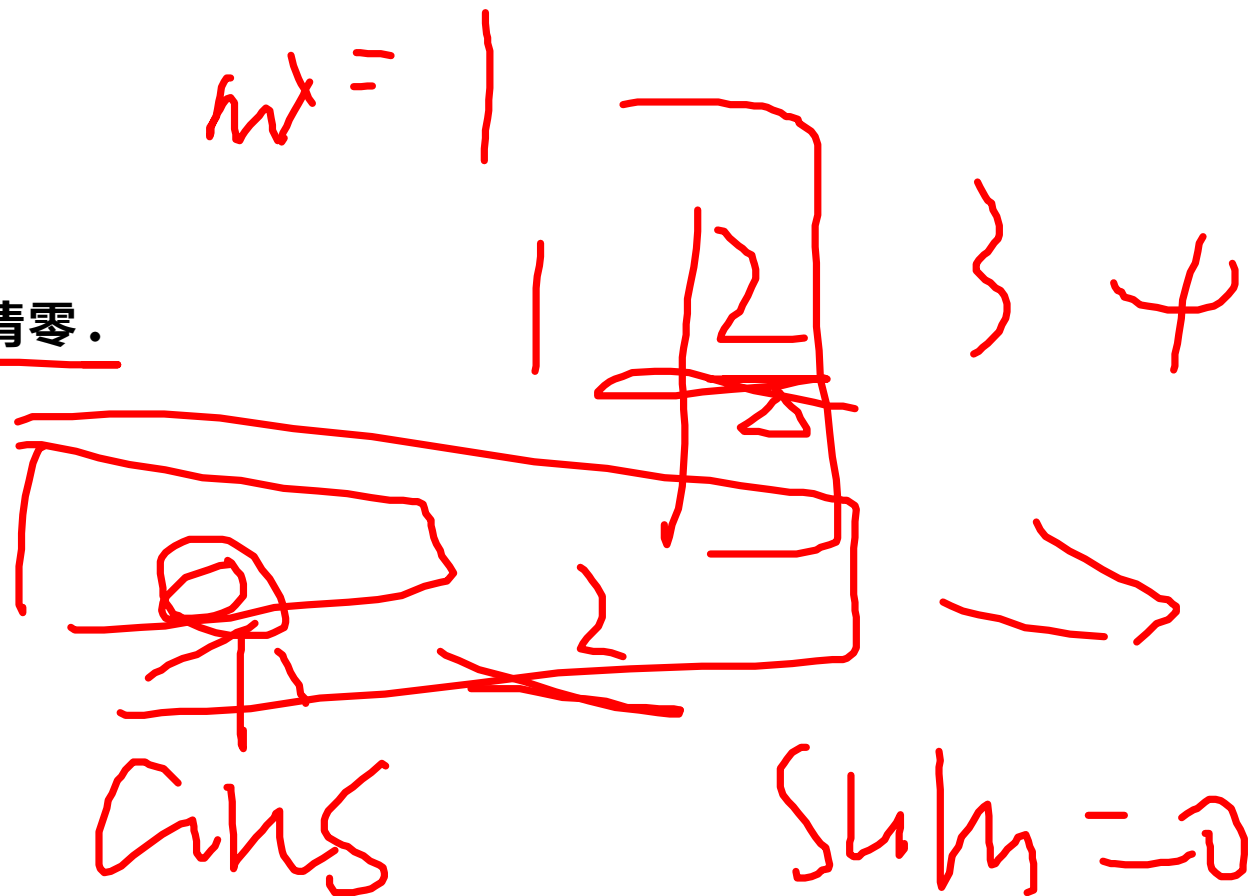
$$\begin{aligned} 6 - 3 &= 3 \\ mx &= 4 \\ \text{ans} &= 6 - 4 = 2 \end{aligned}$$

CF1359D Yet Another Yet Another Task

Solution

1. 不管.
2. 当前sum强制清零.

没了.



CF1359D Yet Another Yet Another Task

Code

```
rep(mx, 0, 31) {  
    int res = 0;  
    rep(i, 0, n) {  
        res += a[i];  
        if(res < 0 || a[i] > mx) res = 0;  
        ans = max(ans, res - mx);  
    }  
}
```

ABC171F Strivore

Statement

How many strings can be obtained by applying the following operation on a string S exactly K times: "choose one lowercase English letter and insert it somewhere"?

通过对字符串 S 进行以下操作恰好 K 次，可以得到多少个不同的字符串：“选择一个小写英文字母并将其插入某处”？

ABC171F Strivore

Solution

设 $n = s.size()$.

$n = 4$

a a b c d

1

A

B

C

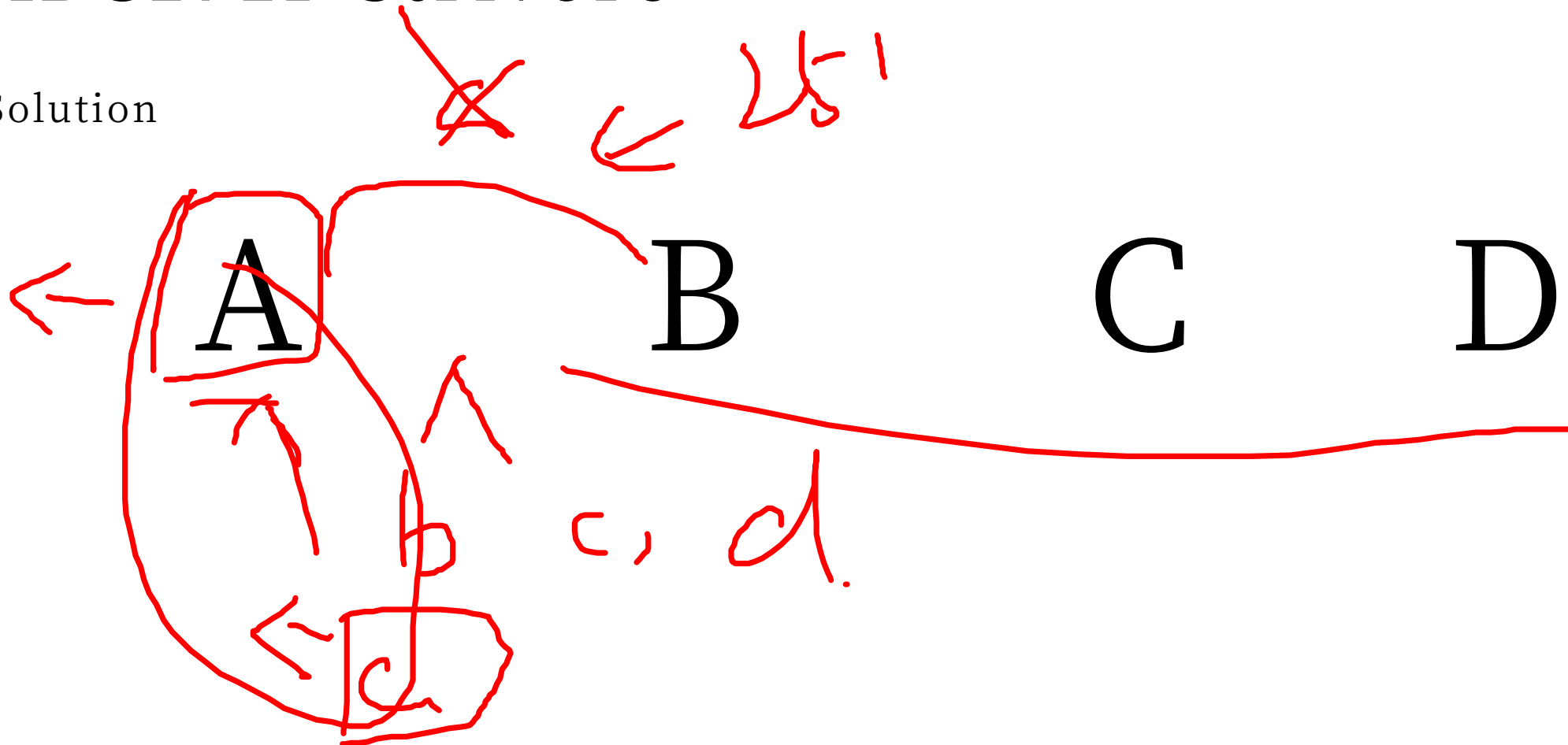
D

^

a b
26ⁱ

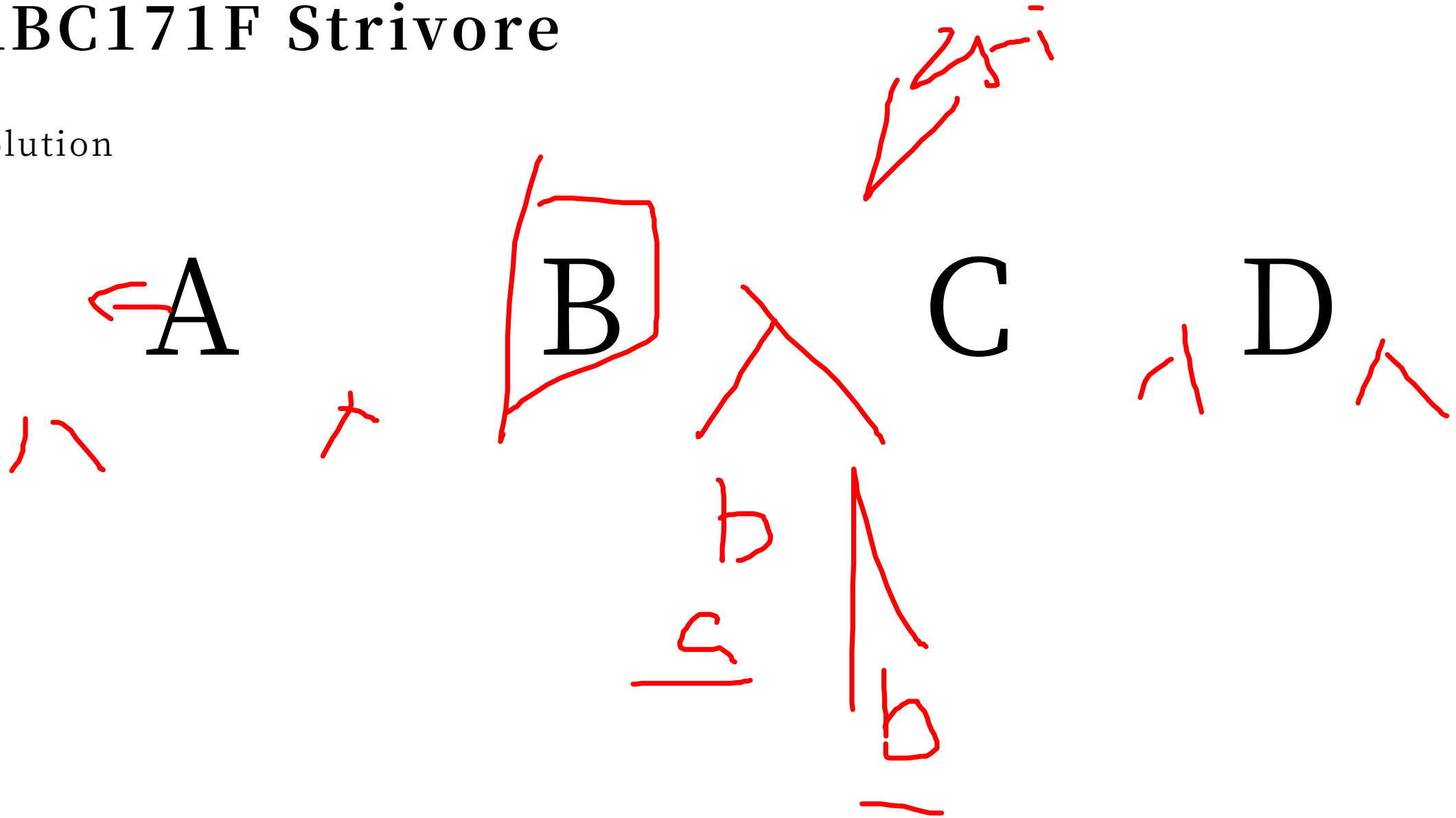
ABC171F Strivore

Solution



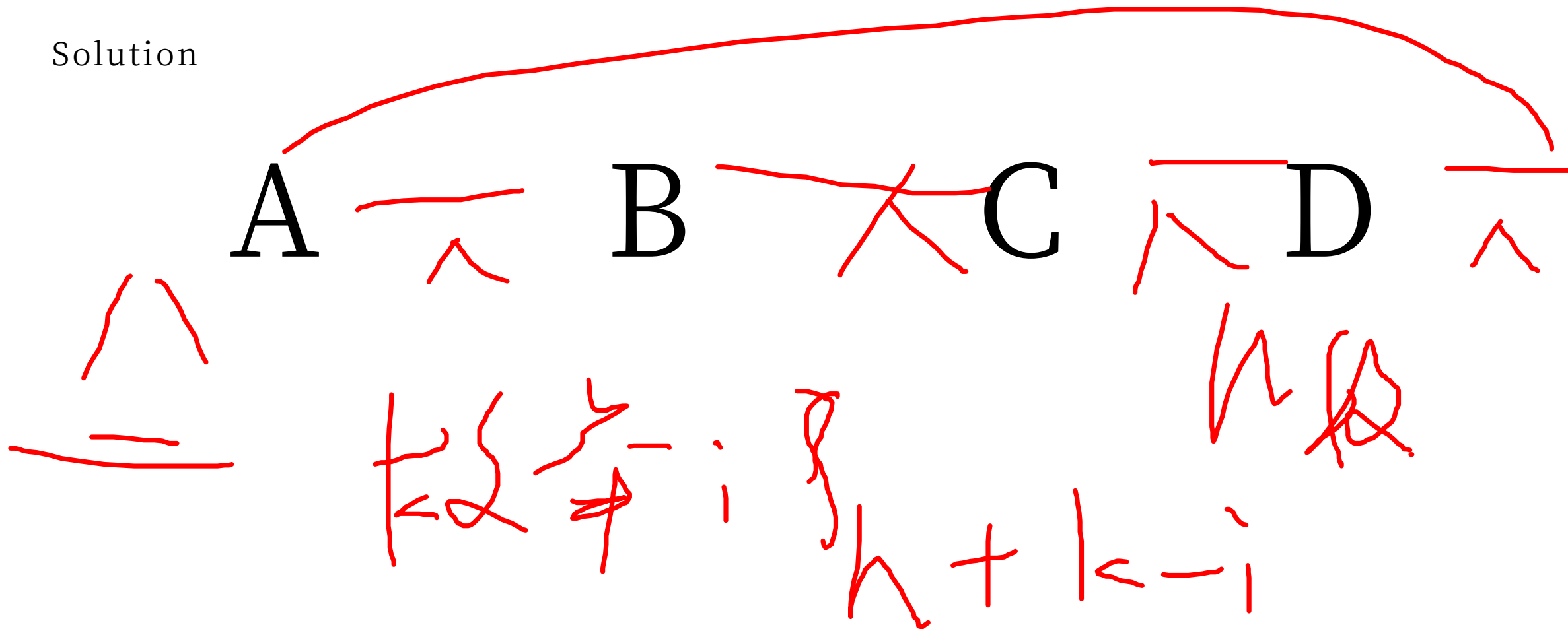
ABC171F Strivore

Solution



ABC171F Strivore

Solution



ABC171F Strivore

Solution

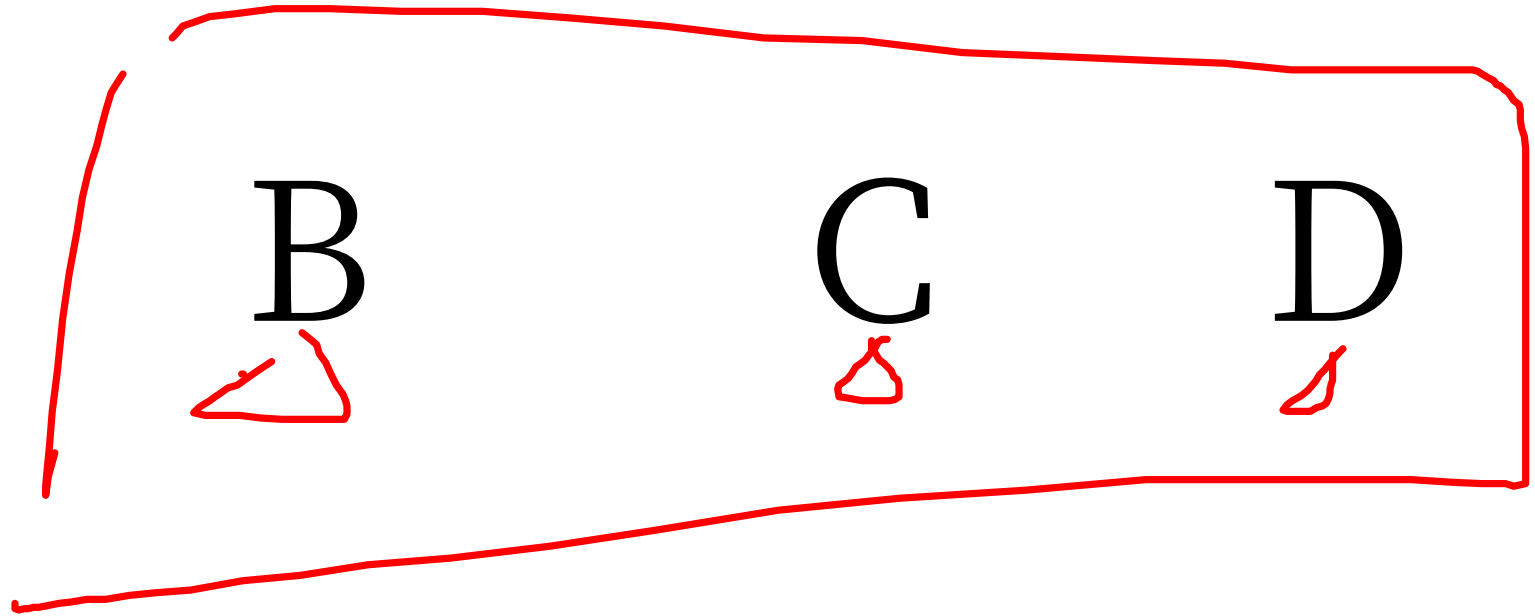
261

A

B

C

D



ABC171F Strivore

Solution

0 ~ k

所以答案也就出来了.

↓

我设第一个字母左边添加*i*个字符,后面则添加*k-i*个,那么答案就是

$$26^i * 25^{k-i} * C(n-1, k+n-i-1)$$

$$\sum_{i=0}^k 26^i * 25^{k-i} * C(n-1, k+n-i-1)$$

ABC171F Strivore

Code

```
int n, k;
std::string s;
std::cin >> k >> s;
n = s.size();
int ans = 0;
prework(2000000);
rep(i, 0, k + 1) ans = (ans + qpow(26, i) * qpow(25, k - i) % MOD *
C(n - 1, k + n - i - 1) % MOD) % MOD;
print(ans, '\n');
```

P7076 [CSP-S2020] 动物园

Statement

存在 2^k 种不同的动物，它们被编号为 $0 \sim 2^k - 1$ 。动物园里饲养了其中的 n 种，其中第 i 种动物的编号为 a_i 。有 m 条要求，第 j 条要求形如“如果动物园中饲养着某种动物，满足其编号的二进制表示的第 p_j 位为 1，则必须购买第 q_j 种饲料”。其中饲料共有 c 种，它们从 $1 \sim c$ 编号。实际上根据购买到的饲料，动物园可能可以饲养更多的动物。计算动物园目前还能饲养多少种动物。
 $0 \leq n, m \leq 10^6, 0 \leq k \leq 64$ 。

P7076 [CSP-S2020] 动物园

Solution

显然如果第 i 个饲料没买，那么所有第 p_i 位是1的动物都不成立。

那我们看一下哪些饲料必须买，那么剩下的那些就都不行。

判断一个饲料买不买，只需要看所有动物的 and 和这个饲料与一下，如果等于零，那么不买。



P7076 [CSP-S2020] 动物园

Solution

最后看一下有多少位不能为1，那么剩下的位都可以填1。那答案就是 $2^{k-\text{cannot}-n}$ 。

注意 $k-\text{cannot}=64$ ， $n=0$ 时的特判。

2⁶⁴

1
2
D
2
X

P7076 [CSP-S2020] 动物园

Code

```
rep(i, 0, n) sum |= a[i];
```

```
ans = (1ll << k) - n;
```

```
rep(i, 0, m) {
```

```
    int p, q;
```

```
    read(p), read(q);
```

```
    if(!((1ll << p) & sum)) sum2 += (cannot & (1ll << p)) == 0, cannot |= (1ll << p);
```

```
    }
```

```
ans = (1ll << k - sum2) - n;
```

```
print(ans);
```

枚举 p 和 q

ABC137_d Summer Vacation

ABC137_d Summer Vacation

Description

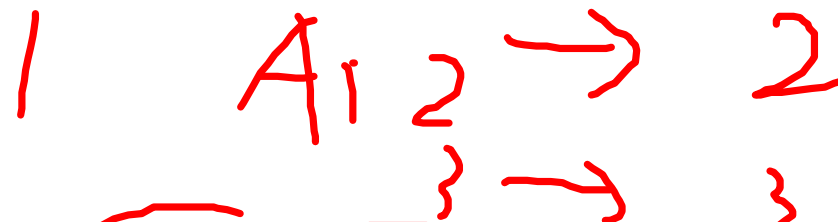
这里有 n 个工作. 如果你做了第 i 个工作并且完成它, 从你做它的那天数起的第 A_i 天, 你会获得 B_i 的金钱;

你一天最多只能做并完成一个工作, 你也可以选择不做;

你也不能做一个你已经做过的工作;

求出在 M 天内你可以获得的最大价值;

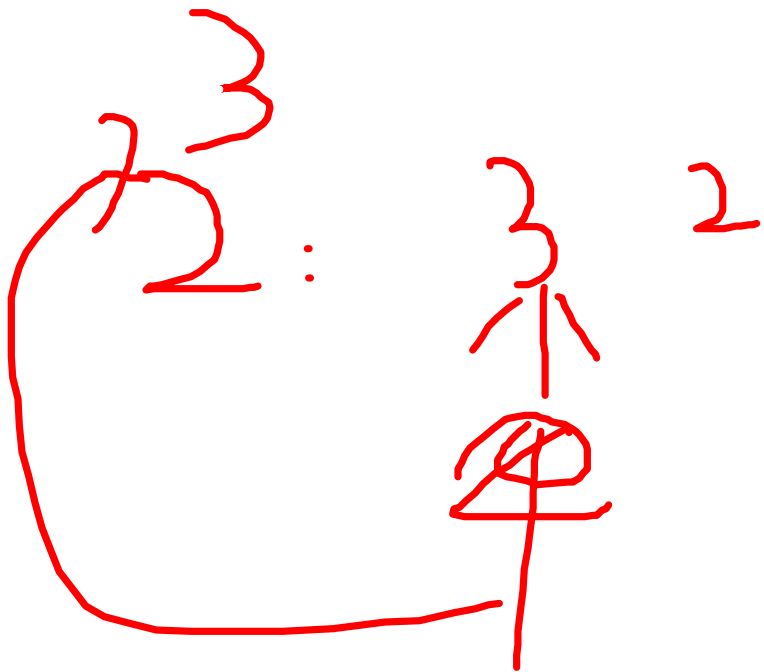
$N, M \leq 100000$



ABC137_d Summer Vacation

正常想法:

这是个01背包!



ABC137_d Summer Vacation

01 背包?

我们发现,这里没有“体积”的概念,更进一步的,我们也没法保证选掉了的物品的价值一定能加在最终结果中。

ABC137_d Summer Vacation

01 背包?

这条路行不通!

ABC137_d Summer Vacation

第二想法:

这是个贪心!

ABC137_d Summer Vacation

贪心？

先给他排个序，然后去最大的，然后？

日期加一？

ABC137_d Summer Vacation

贪心?

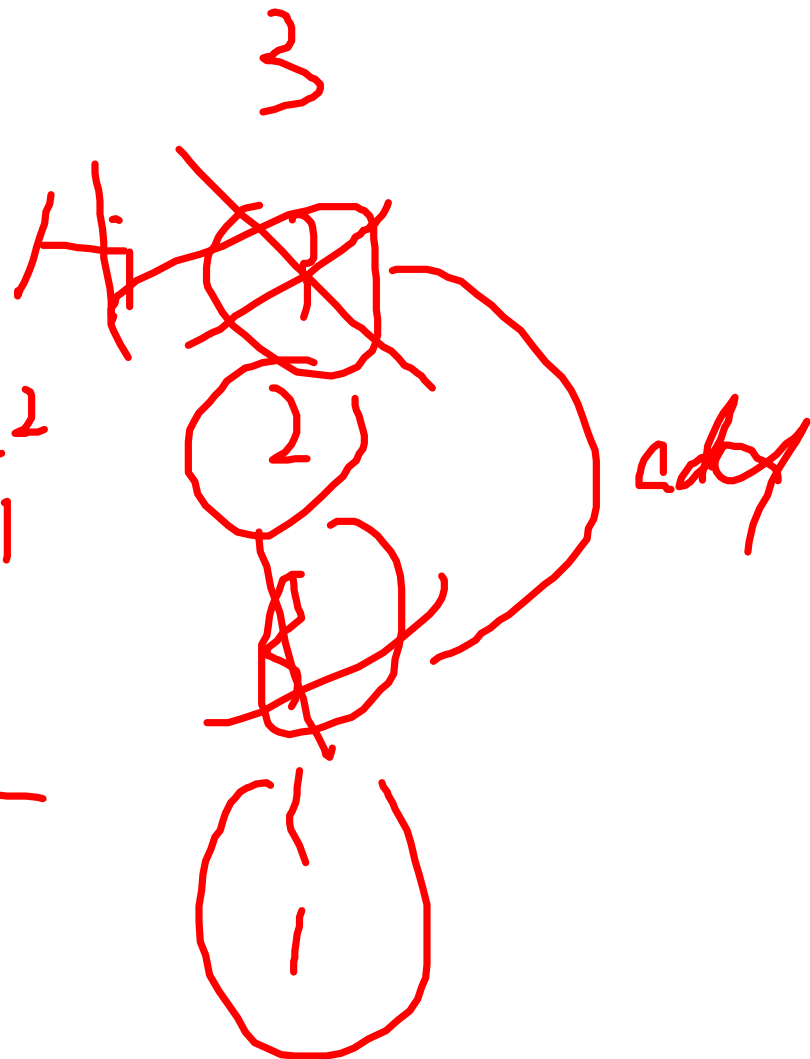
你惊讶的发现你要删掉一堆元素。

这不是重点，关键是你把正解也删掉了

考虑下列数据：

增加

$d \begin{matrix} 1 \rightarrow 2 \\ 2 \rightarrow 1 \end{matrix}$



ABC137_d Summer Vacation

贪心？

考虑下列数据：

4 5

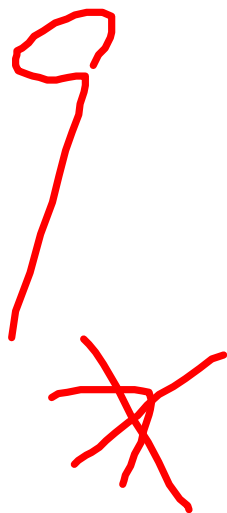
4 3

3 4

1 1

1 1

1 1



ABC137_d Summer Vacation

贪心？

你第一步选了 3 4。

然后你把 ~~4 3~~ 删了。

你就选不了 4 3 了。

你的答案就错掉了。

ABC137_d Summer Vacation

所以！

正解已经出来了！

ABC137_d Summer Vacation

逆向贪心!

ABC137_d Summer Vacation

逆向贪心!

日期从m到1，每一天加入当天工作可以获得报酬的工作，在这堆东西里面求最大值即可。



ABC137_d Summer Vacation

逆向贪心！

于是我们就想到了优先队列。

然后你发现这个题做完了。

Atcoder DDCC2020Qual D Digit Sum Replace

Atcoder DDCC2020Qual D Digit Sum Replace

Description

给你一个数 n ,他的前 d_1 位是 c_1 ,接下来 d_2 位是 c_2 ,直到最后 d_m 位是

c_m

你每次可以进行一次操作:选中 n 的相邻两个数位,将他们替换为它们的和.如 $2378 \rightarrow 578$ 或 2108 或 2315

求让 n 只剩一位数的最多操作次数.

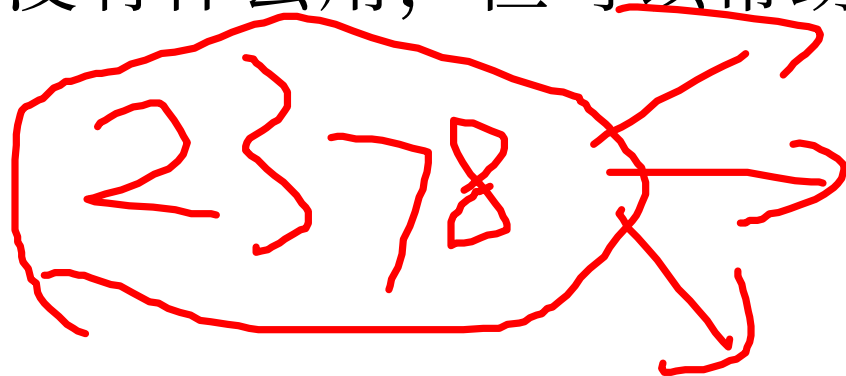
Atcoder DDCC2020Qual D Digit Sum Replace

算法1:暴力

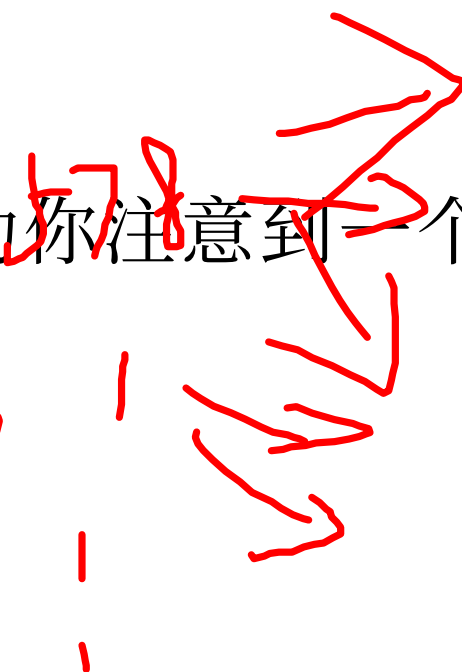
每次选两个数，给他加起来。

虽然没有什么用，但可以帮助你注意到一个性质：

2378



578



Atcoder DDCC2020Qual D Digit Sum Replace

无论怎么操作，总操作数不变！

$$278 \rightarrow \underline{\underline{5}}$$

Atcoder DDCC2020Qual D Digit Sum Replace

于是我们可以想到第一种AC方法：

预处理每一块同一个数的操作个数与最后生成的值，对于每一块
 $O(1)$ 计算，最后暴力合并。

$$\begin{array}{c|c} C_1 & C_2 \\ \hline \hline \end{array}$$

2×10

$$0 \sim 9$$

Atcoder DDCC2020Qual D Digit Sum Replace

举个例子。 *CNS*

3

33

333

3333

33333

333333



$$\frac{5}{3} + 5 - 1$$

Atcoder DDCC2020 Qual D Digit Sum Replace

但是我们可以进一步优化。

考虑每次操作他干了什么。

Atcoder DDCC2020Qual D Digit Sum Replace

如果有进位：他的数字和减少9.

如果没有进位：他的数字个数减少1

再审视一下目标：数字和小于等于9且数字个数为1.

2378 20
 19
 11
 108

2378 → 578
 4 3

Atcoder DDCC2020Qual D Digit Sum Replace

做完了。

$$\text{ans} = \frac{\text{sum} + 1}{9} + \text{num} - 1$$

ABC065D Build?

Description:

给你 n 个点，每个点的坐标是 (x_i, y_i) ，定义 (a, b) 与 (c, d) 两点间的距离为 $\min\{|a-c|, |b-d|\}$ ，求这个图的MST。

$n \leq 200000$

ABC065D Build?

最naive的做法:

建 n^2 条边,在这个图上跑MST

显然过不去

$O(n^2)$

ABC065D Build?

于是我们考虑能不能少建一点边.以横坐标为例.

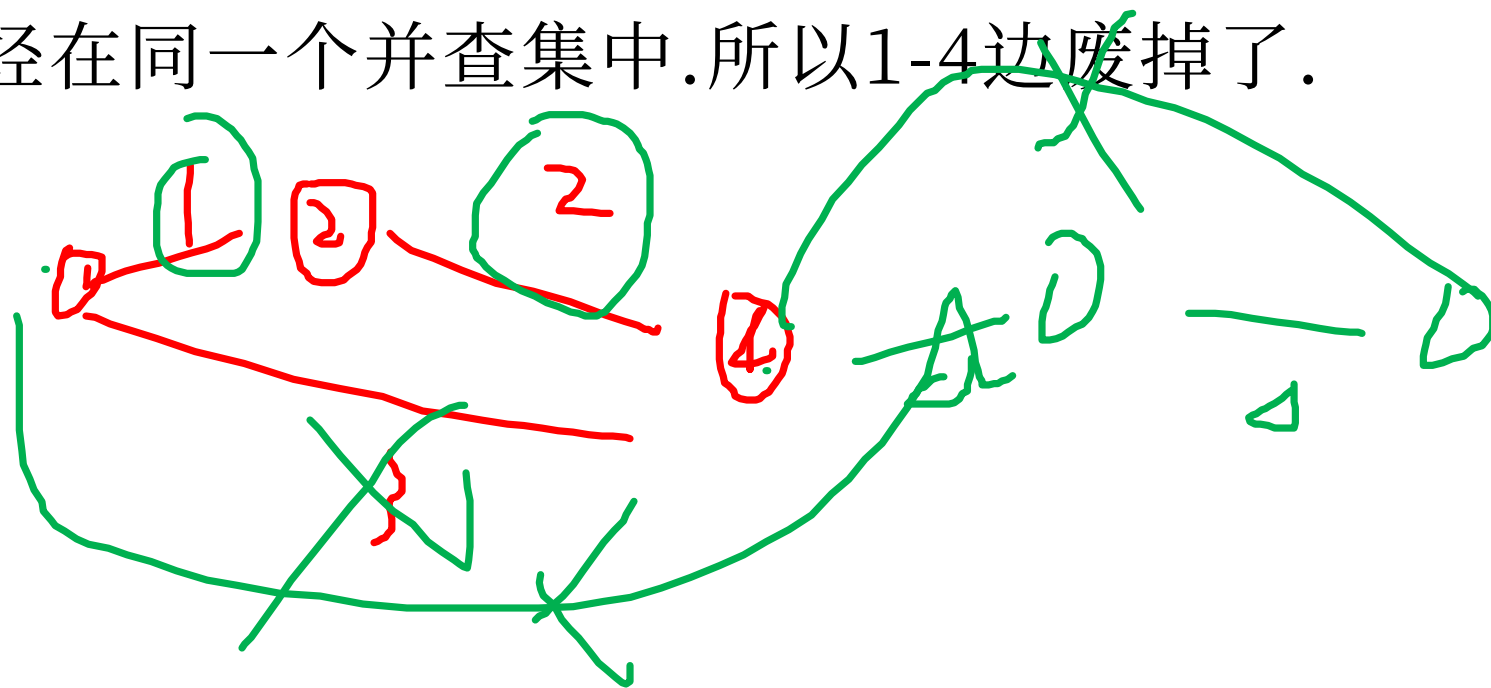
我们有三个点横坐标为1,2,4

那么我们会建1-2,2-4,1-4三条边.

ABC065D Build?

但是用Kruscal跑一下呢?

会发现1-2,2-4两条边肯定在1-4被循环到的时候已经建掉了.这时这三个点已经在同一个并查集中.所以1-4边废掉了.



ABC065D Build?

于是我们就得到了一个做法.

将所有点按横坐标排序,在相邻的两个点之间建边.

接下来将所有点按纵坐标排序,在相邻的两个点之间建边.

ABC065D Build?

这时我们发现边数只剩下 $2(n-1)$ 条,可以通过本题.

CF1486D Max Median

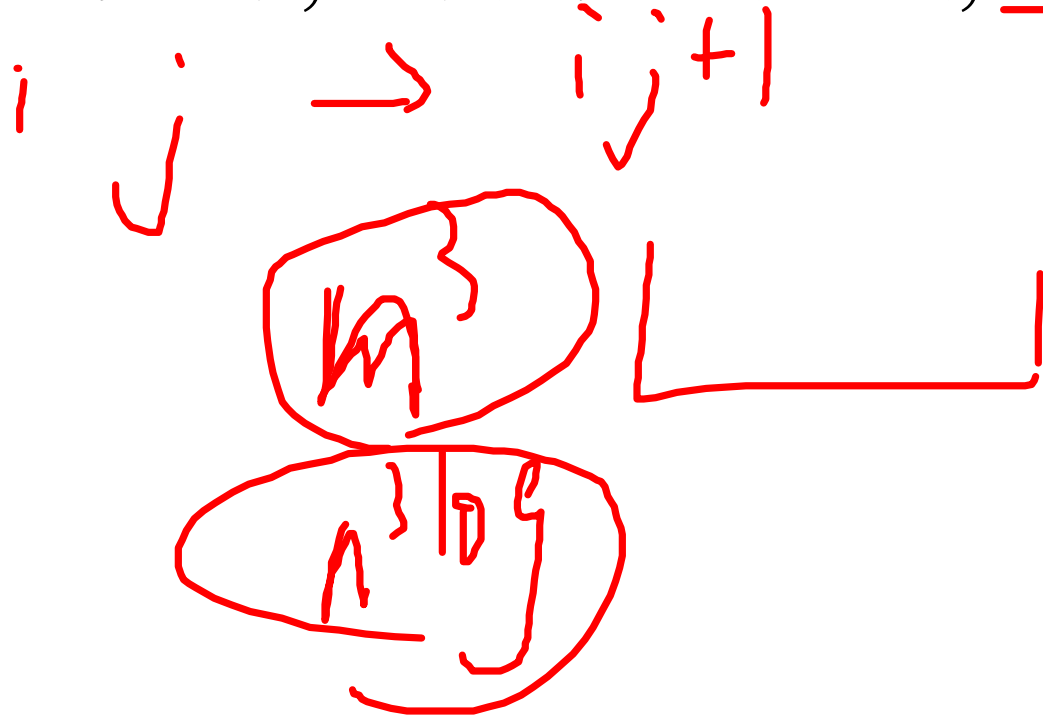
Description

给你一个长度为 n 的数列 a ，求 a 中所有长度 $\leq k$ 的子串的中位数的最大值。 $n \leq 200000$

CF1486D Max Median

暴力?

枚举每一个子串,用平衡树做中位数, $O(n^2 \log n)$



CF1486D Max Median

显然不能暴力做.

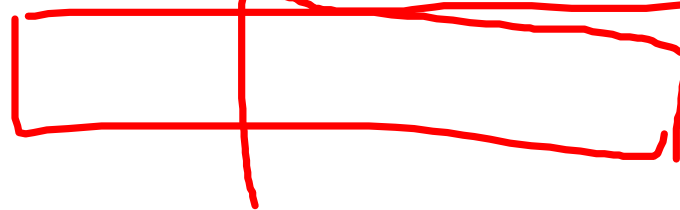
因为求的是最大值,考虑二分.

CF1486D Max Median

为了保证满足二分性,二分返回的是“中位数是否有可能大于或等于 x ”。

因为没说中位数必须是 x ,所以我只需要考虑当前数和 x 的大小关系。

$$\begin{array}{r} x = 1 \\ \hline x = 0 \end{array}$$

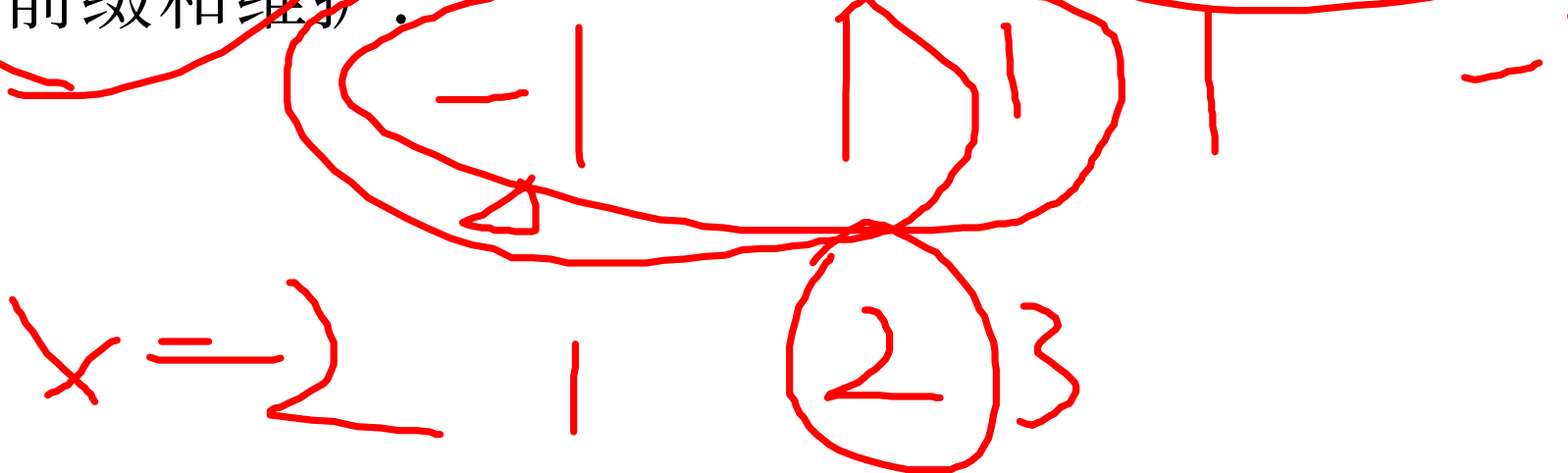


CF1486D Max Median

定义 $b_i = \begin{cases} 1, & a_i \geq x \\ -1, & a_i < x \end{cases}$

则 $[i, j]$ 区间中位数大于等于 x 的充要条件是 $\sum_{k=i}^j b_k > 0$

于是想到前缀和维护.



CF1486D Max Median

那么 $\text{sumj} - \text{sumi} > 0$

但是,子串的长度是不一定的.所以我要最大化 $\text{sumj} - \text{sumi}$.

我去循环j,则sumj是定值.sumi可以取 $1 \sim j - k$ 之间最小的sumi.

做最小值可以 另开一个数组维护.

CF1486D Max Median

```
bool check(int x) {
```

```
    rep(i, 1, n) b[i] = a[i] >= x ? 1 : -1;
```

```
    rep(i, 1, n) sum[i] = sum[i - 1] + b[i];
```

```
    rep(i, 1, n) m[i] = min(m[i - 1], sum[i]);
```

```
    rep(i, k, n) if(sum[i] - m[i - k] > 0) return 1;
```

```
    return 0;
```

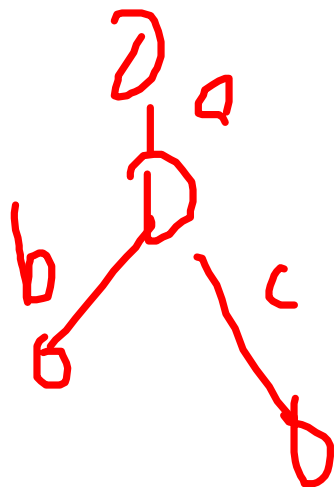
```
}
```



ni sum b

CF1624G MinOr Tree

给你一张图，求他在或运算意义下的 MST。

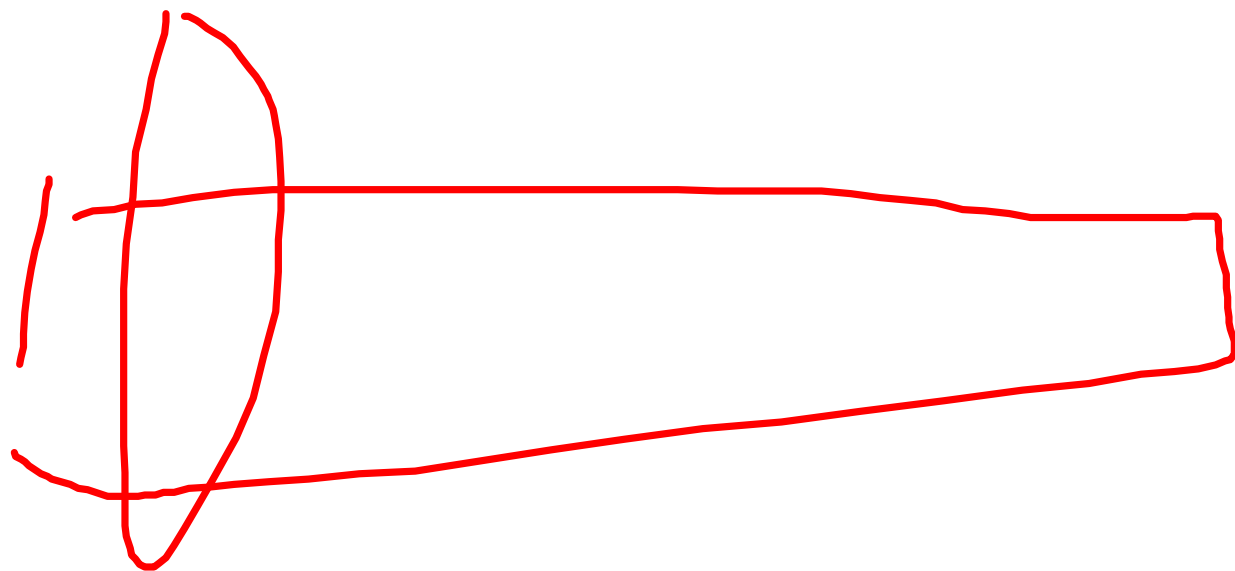


a | b | c
min

CF1624G MinOr Tree

考虑按位去做这个题。

从高往低枚举每一位，如果这一位能不选，我就不选他。



CF1624G MinOr Tree

怎样判断他是否要选呢？

我会用并查集！

我把不含这一位的所有边连接的两个点 merge 起来，最后如果只剩一个并查集，说明不用选也可以。



CF1624G MinOr Tree

如果这一位不要，那我就把所有含有这一位的边全部删光。

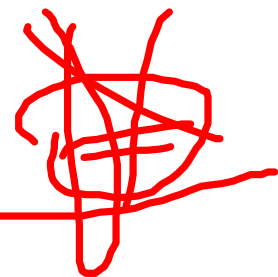
将他的权值设为 -1 然后排个序,则这条边就送到数组末端，最后直接减少边的条数就能完成删边操作。

```
bool cmp(edge i, edge j) {return e[i].val>e[j].val}
```

```
rep(i,1,cnt) if(e[i].val&(1<<v)) e[i].val=-1;
```

```
std::sort(e+1,e+cnt+1,cmp);
```

```
rep(i,1,cnt) if(e[i].val==-1) {cnt=i-1;break;}
```



CF1624G MinOr Tree

如果这一位要，累加 ans 。做完了。

$ans + = 1 \ll v$

