

# 牛客练习赛83

[比赛链接](#)

## D-数列递推

### 题意

给定  $n, f_0$  求  $f_{\sim n}$  其中

$$f_i = \sum_{j=1}^i \text{if}_{i \bmod j}$$

### 题解

设  $i = kb + r$  考虑整数分块枚举  $k$  设  $t = i \bmod k$  此时有  $r = t, t+k, t+2k, \dots, i-k$  的前缀和

当  $k < \sqrt{n}$  时，如果能提前维护  $f_t, f_{t+2k}, f_{t+3k}, \dots$  的前缀和，就可以  $O(1)$  计算贡献。

当  $k \geq \sqrt{n}$  时，显然  $b, r$  唯一，直接计算贡献即可。于是整数分块部分的时间复杂度为  $O(\sqrt{n})$

对每个  $i$  枚举  $k < \sqrt{n}$  更新  $f_t, f_{t+2k}, f_{t+3k}, \dots$  的前缀和的时间复杂度为  $O(\sqrt{n})$  于是总时间复杂度  $O(n\sqrt{n})$

```
const int MAXN=1e5+5,MAXM=405,Mod=998244353;
int f[MAXN];
int s[MAXM][MAXN];
int main()
{
    int n=read_int(),v0=read_int(),m=sqrt(n)+1;
    f[0]=1;
    for(i,0,m)
        s[i][0]=1;
    rep(i,1,n){
        int lef=1,rig=0;
        while(left<=i){
            rig=i/(i/left);
            int k=i/left;
            if(k<m)
                f[i]=(f[i]+s[k][i-left*k])%Mod;
            else
                f[i]=(f[i]+f[i-left*k])%Mod;
            left=rig+1;
        }
        for(k,1,m){

```

```
        if(i>=k)
            s[k][i]=(s[k][i-k]+f[i])%Mod;
        else
            s[k][i]=f[i];
    }
}
__rep(i,1,n){
    f[i]=(1LL*f[i]*v0%Mod+Mod)%Mod;
    space(f[i]);
}
return 0;
}
```

## E-小L的疑惑

### 题意

给定互素的  $a, b$  求  $ax + by (x, y \geq 0)$  不能表示的数中第  $k$  大的数。

### 题解

首先给定结论当  $a, b$  互素时  $ax + by (x, y \geq 0)$  不能表示的正数等价于所有形如  $ab - na - mb (n, m \geq 1)$  的正数，具体见 [证明](#)

接下来问题转化为求所有形如  $ab - na - mb (n, m \geq 1)$  的正数中第  $k$  大的元素。

考虑维护两个队列，队列一维护所有  $ab - a - mb (m = 1)$  且保证递减。

队列二维护所有  $ab - na - mb (n \geq 1, m \geq 2)$  且保证递减。每次选择当前两个队列的队首的较大者进行 `\text{pop}` 连续  $k$  次即可得到第  $k$  大。

队列一的维护仅需要每次 `\text{pop}` 后  $m$  加一即可。队列二初始时为空，每次 `\text{pop}` 元素  $t$  后将  $t - b$  加入队列。

现证明这样得到的队列二一定满足递减性质。假设先前 `\text{pop}` 的所有元素确实为前  $k$  大，记为  $a_1, a_2 \dots a_k$

记本次 `\text{pop}` 的元素为  $a_{k+1}$  则队列二的当前所有元素一定是由  $a_t - b$  得到的，由于  $a_t > a_{k+1}$  所以  $a_t - b > a_{k+1} - b$  满足单调性。

于是总时间复杂度  $O(k)$

```
int main()
{
    LL a=read_int(), b=read_int(), k=read_int(), pos=1;
    queue<LL> q;
    while(true){
```

```
if(q.empty() || q.front()<a*b-pos*a-b){
    k--;
    if(k==0){
        enter(a*b-pos*a-b);
        break;
    }
    q.push(a*b-pos*a-2*b);
    pos++;
}
else{
    k--;
    if(k==0){
        enter(q.front());
        break;
    }
    q.push(q.front()-b);
    q.pop();
}
}
return 0;
}
```

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



Permanent link:  
[https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal\\_string:jxm2001:contest:%E7%89%9B%E5%AE%A2%E7%BB%83%E4%B9%A0%E8%B5%9B&rev=1621734727](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:legal_string:jxm2001:contest:%E7%89%9B%E5%AE%A2%E7%BB%83%E4%B9%A0%E8%B5%9B&rev=1621734727)

Last update: 2021/05/23 09:52