

# 牛客练习赛87

[比赛链接](#)

## C - 牛老板

### 题意

用最少的  $6^k, 9^k$  表示  $n$  ( $1 \leq n \leq 10^{12}$ )

### 题解

预处理  $n \leq 10^6$  的答案，然后启发式搜索。

```
const LL MAXV=1e12+5,MAXN=1e6;
vector<LL> vec;
LL dp[MAXN];
void Init(){
    vec.push_back(1);
    LL base=6;
    while(base<MAXV){
        vec.push_back(base);
        base*=6;
    }
    base=9;
    while(base<MAXV){
        vec.push_back(base);
        base*=9;
    }
    sort(vec.begin(),vec.end());
    _for(i,1,MAXN){
        dp[i]=i;
        for(int v:vec){
            if(v>=i)
                dp[i]=min(dp[i],dp[i-v]+1);
            else
                break;
        }
    }
}
LL ans;
void dfs(LL x,int pos,int step){
    if(x<MAXN){
        ans=min(ans,dp[x]+step);
        return;
    }
```

```
}
if(pos==0){
    ans=min(ans,x+step);
    return;
}
LL m=x/vec[pos];
for(int i=m;i>=0;i--){
    if(m+step>=ans)return;
    dfs(x-i*vec[pos],pos-1,step+i);
}
}
int main()
{
    Init();
    int T=read_int();
    while(T--){
        LL x=read_LL();
        ans=1e9;
        dfs(x,vec.size()-1,0);
        enter(ans);
    }
    return 0;
}
```

## D - 小G的排列-加强版

### 题意

求不存在某个长度超过  $m$  的连续子串恰好是  $i, i+1, i+2 \cdots i+k-1$  或  $i+k-1, i+k-2 \cdots i$  的  $1 \sim n$  的排列的个数。

数据保证  $m \geq \lfloor \frac{n-1}{2} \rfloor$

### 题解

称  $i, i+1, i+2 \cdots i+k-1$  或  $i+k-1, i+k-2 \cdots i$  为长度等于  $k$  的非法子串。

显然，当  $m \geq \lfloor \frac{n-1}{2} \rfloor$  时任意一个排列不存在两个不相交的长度超过  $m$  的非法子串。

设  $f(n, m)$  表示所有  $1 \sim n$  的排列包含的长度为  $m$  的非法子串个数。

考虑非法子串的元素选中，共  $n-m+1$  种，然后将非法子串当成一个元素和其他正常元素可以任意排列，因此有  $(n-m+1)$  排列方式。

最后  $m$  超过  $1$  时还要考虑  $i, i+1, i+2 \cdots i+k-1$  或  $i+k-1, i+k-2 \cdots i$  不同的贡献，于是有

$$f(n, m) = (n-m+1)(n-m+1)!(1+(m \neq 1))$$

最后，考虑每个含有长度超过  $m$  的非法子串的排列，假设他含有的最长非法子串长度为  $k$  则他对  $f(n, m+1)$  的计数贡献为  $k-m$

同时他对  $f(n, m+2)$  的计数贡献为  $k-m-1$  于是每个含有长度超过  $m$  的非法子串的排列对  $f(n, m+1)-f(n, m+2)$  的贡献恰好为  $1$ 。

于是  $f(n, m+1)-f(n, m+2)$  就等于所有含有长度超过  $m$  的非法子串的排列个数。根据容斥，答案为  $n!-f(n, m+1)+f(n, m+2)$

```
const int MAXN=1e7+5,mod=1e9+7;
int frac[MAXN];
void Init(){
    frac[0]=1;
    _for(i,1,MAXN)
        frac[i]=1LL*frac[i-1]*i%mod;
}
int cal(int n,int m){
    if(m>n)
        return 0;
    else
        return (1LL+(m!=1))*(n-m+1)%mod*frac[n-m+1]%mod;
}
int main()
{
    Init();
    int T=read_int();
    while(T--){
        int n=read_int(),m=read_int();
        int ans=(frac[n]-(cal(n,m+1)-cal(n,m+2)))%mod;
        enter((ans+mod)%mod);
    }
    return 0;
}
```

## F - 简洁的题面

### 题意

$$\sum_{i_1=0}^n \sum_{i_2=0}^n \cdots \sum_{i_k=0}^n \sum_{d=1}^g g[d] \text{ast} (i_1+i_2+\cdots i_k) \leq n \prod_{j=1}^k \{n-d \text{ast} \sum_{h=1}^{j-1} i_h \text{choose} d \text{ast} i_j\}$$

$$1 \leq g, k \leq 3, n \leq 10^9$$

### 题解

化简，得上式等于

$$\sum_{d=1}^g \sum_{i_1+i_2+\dots+i_k \leq n} \frac{n!}{(i_1)!(i_2)!\dots(i_k)!(n-i_1-i_2-\dots-i_k)!} d^{i_1} d^{i_2} \dots d^{i_k}$$

由于  $g$  很小，可以暴力枚举  $d$  对固定的  $d$  相当于查询长度为  $n$  的  $k+1$  重集排列的个数，满足前  $k$  种元素出现次数都是  $d$  的倍数。

设  $\text{dp}(n, i_1, i_2, \dots, i_k)$  表示长度为  $n$  的  $k+1$  重集排列的个数，满足前  $k$  种元素出现次数模  $d$  分别为  $i_1, i_2, \dots, i_k$

用  $k$  位  $d$  进制数表示状态  $(i_1, i_2, \dots, i_k)$  矩阵快速幂加速  $\text{dp}$  时间复杂度  $O(g^{3k} \log n)$

```
const int MAXS=27;
int mod;
struct Matrix{
    int a[MAXS][MAXS];
    Matrix(int type=0){
        mem(a,0);
        if(type){
            _for(i,0,MAXS)
                a[i][i]=1;
        }
    }
    Matrix operator * (const Matrix &b) const{
        Matrix c;
        _for(i,0,MAXS)_for(j,0,MAXS)_for(k,0,MAXS)
            c.a[i][j]=(c.a[i][j]+1LL*a[i][k]*b.a[k][j])%mod;
        return c;
    }
};
Matrix quick_pow(Matrix n,int k){
    Matrix ans=Matrix(1);
    while(k){
        if(k&1)ans=ans*n;
        n=n*n;
        k>>=1;
    }
    return ans;
}
int c[5];
void decode(int s,int d,int k){
    _for(i,0,k){
        c[i]=s%d;
        s/=d;
    }
}
int encode(int d,int k){
    int s=0;
    for(int i=k-1;i>=0;i--)
        s=s*d+c[i];
}
```

```
        return s;
    }
    int main()
    {
        int T=read_int();
        while(T--){
            int n,k,g,ans=0;
            n=read_int(),k=read_int(),mod=read_int(),g=read_int();
            _rep(d,1,g){
                Matrix m;
                int s=1;
                _for(i,0,k)s*=d;
                _for(i,0,s){
                    decode(i,d,k);
                    m.a[i][i]=1;
                    _for(j,0,k){
                        c[j]=(c[j]+1)%d;
                        m.a[i][encode(d,k)]++;
                        c[j]=(c[j]+d-1)%d;
                    }
                }
                m=quick_pow(m,n);
                ans=(ans+m.a[0][0])%mod;
            }
            enter(ans);
        }
        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%9B87](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%9B87)

Last update: 2021/08/21 22:40