

# 解方程

MicDZ

长沙市长郡中学

## 1 题目大意

求多项式方程：

$$a_0 + a_1x + a_2x^2 + \dots + a_nx^n = 0$$

在  $[1, m]$  内的整数解,  $n \leq 100, |a_i| \leq 10^{10000}, m < 10^6$

## 2 题解

### 2.1 $O(nm)$ 做法

设原方程的对应函数为

$$f(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

当  $f(x) = 0$  时,  $f(x) \bmod p = 0$ , 只需要随便选取一个  $p$ , 将  $x \in [1, m]$  中的每一个  $x$  带入原方程, 就有一定概率获得正确答案。

### 2.2 $O(n \log^2 n)$ 做法

Karry5307 提供一种用多项式多点求值解决此题的方法。想听的话可以让他来讲一讲。

### 2.3 $O(pn + m)$ 做法

这个做法还是比较靠谱的做法 (比 2.2 的算法靠谱)。设  $x \equiv b \pmod{p}$ , 那么显然的是  $f(x) \equiv f(b) \pmod{p}$ , 即  $f(x) \equiv f(x \bmod p) \pmod{p}$ 。那么可以直接预处理出  $f(x)$  在  $[1, p-1]$  之间的所有值, 然后再在  $[1, m]$  中间查表。

选择一个好的模数很重要, 如果不放心可以多选几个。

### 2.4 更快的做法

我不会, 有写过的同学可以分享一下。

## 3 代码

### 3.1 $O(nm)$ 做法

---

```
1 #include<iostream>
2 #include<cstdio>
3 #include<cmath>
4 #include<cstring>
5 #include<algorithm>
6
7 using namespace std;
8 #define int ll
9 #define REP(i,e,s) for(register int i=e; i<=s; i++)
10 #define DREP(i,e,s) for(register int i=e; i>=s; i--)
11 #define ll long long
12 #define DE(...) fprintf(stderr, __VA_ARGS__)
13 #define DEBUG(a) DE("DEBUG: %d\n", a)
14 #define file(a) freopen(a".in", "r", stdin); freopen(a".out", "w",
    stdout)
15
16 const int MAXN=100000+10, MOD=19260817;
17
18 int read() {
19     int x=0, f=1, ch=getchar();
20     while(ch>'9' || ch<'0') { if(ch=='-') f=-1; ch=getchar(); }
21     while(ch>='0' && ch<='9') { x=(x*10+ch-'0')%MOD; ch=getchar(); }
22     return x*f;
23 }
24
25 int a[MAXN], n, m;
26
27 int f(int x, int mod) {
28     int ans=0, prod=1;
29     REP(i, 0, n) {
30         ans=(ans+prod*a[i])%MOD;
31         prod=(prod*x)%MOD;
32     }
33     return ans;
34 }
35
36 int book[MAXN];
37
```

```

38 int ans[MAXN], cnt;
39
40 signed main() {
41     n=read(), m=read();
42
43     REP(i, 0, n) a[i]=read();
44
45     REP(i, 1, m) if(f(i, MOD)==0) ans[++cnt]=i;
46     printf("%lld\n", cnt);
47     REP(i, 1, cnt) printf("%lld\n", ans[i]);
48     return 0;
49 }

```

---

### 3.2 $O(n \log^2 n)$ 做法

写不出来，也没人写过

### 3.3 $O(pn + m)$ 做法

<https://www.luogu.org/blog/Thinking/solution-p2312>