解方程

MicDZ

长沙市长郡中学

1 题目大意

求多项式方程：

$$a_0 + a_1 x + a_2 x^2 + ... + a_n x^x = 0$$

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

$$f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^x$$

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

```cpp
#include<iostream>
#include<cstdio>
#include<cmath>
#include<cstring>
#include<algorithm>

using namespace std;
#define int ll
#define REP(i,e,s) for(register int i=e; i<=s; i++)
#define DREP(i,e,s) for(register int i=e; i>=s; i--)
#define ll long long
#define DE(...) fprintf(stderr,__VA_ARGS__)
#define DEBUG(a) DE("DEBUG: %d\n",a)
#define file(a) freopen(a".in","r",stdin);freopen(a".out","w",stdout)

const int MAXN=100000+10,MOD=19260817;

int read() {
	int x=0,f=1,ch=getchar();
	while(ch>'9'||ch<'0'){if(ch=='-')f=-1;ch=getchar();}
	while(ch>='0'&&ch<='9'){x=(x*10+ch-'0')%MOD;ch=getchar();}
	return x*f;
}

int a[MAXN],n,m;

int f(int x,int mod) {
	int ans=0,prod=1;
	REP(i,0,n) {
		ans=(ans+prod*a[i])%MOD;
		prod=(prod*x)%MOD;
	}
	return ans;
}

int book[MAXN];

```

```
int ans[MAXN],cnt;

signed main() {
        n=read(),m=read();

        REP(i,0,n) a[i]=read();

        REP(i,1,m) if(f(i,MOD)==0) ans[++cnt]=i;
        printf("%lld\n",cnt);
        REP(i,1,cnt) printf("%lld\n",ans[i]);
        return 0;
}
```

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

4 秦九韶算法

这是一个高性能的多项式求值的方法。也称作霍纳法则。

$$f(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0=((\cdots(((a_nx+a_{n-1})x+a_{n-2})x+a_{n-3})\cdots)x+a_1)x+a_0$$

该算法在实际的实现上也是 $O(n)$ 级别的，但是其常数会小不少。

考虑下面的这个例子：

$$g(x)=ax^2+bx+c$$

用朴素算法不难得出这样的算法流程：

```
prod=1   ans+=prod*c
prod*=x  ans+=prod*b
prod*=x  ans+=prod*a
```

使用秦九韶算法的算法流程：

```
ans=0    ans+=a
ans*=x   ans+=b
ans*=x   ans+=c
```

这样总的运算次数得到降低，算法的常数得到优化。