CF1045B Space Isaac

[原题链接]()

题目大意

$0 \sim m-1$的数被分成两个集合，你可以分别从两个集合中取一个数相加并对m取模，求一不能构造出的数。

题解

感觉如果sxd666来做这题肯定能一眼秒，然而他正忙着切其他题。

首先我们发现如果要让$a+b\equiv x \pmod m$，如果已知a,x，那b一定是唯一的。也就是说，假设给定集合是A，与之对应的集合为B，如果有$a\in A$但找不到$b\in A$使得$a+b\equiv x \pmod m$。那么$x\in A+B$（定义$A+B=\{a+b:a\in A,b\in B\}$）。反过来讲，如果$x\notin A+B$，那么一定能把$A$中所有元素配对（可能两个数相同），也即$x\notin A+B \iff A=x-A$（定义$x-A=\{x-a:a\in A\}$）。

然后我们如果把小于m的整数看成一个环，如果有两个数a,b使$a+b\equiv x \pmod m$，a顺时针时针移动，b肯定逆时针移动（即运动方向相反，且移动的长度应该是相等的（$(a+k) \mod m+(b-k) \mod m\equiv a+b \pmod m$嘛）。

于是我们画两个圆，都表示集合$\{a_i\}$（假设a_i已经排好序），我们要把第一个圆的点与第二个圆的点匹配。

假设a_i与a_j匹配。我们把i移动至$i+1$，那么根据上面推出的单调性，j必须移至$j-1$（因为$a_i\sim a_{i+1}$之间没有数了，所以j也只能移动一格），又因为移动距离必须相等，即$a_{i+1}-a_i=a_j-a_{j-1}$。

所以我们令$b_i=a_i-a_{i-1}$（$b_1=(a_1-a_n) \mod m$），设串$s_1=b_nb_{n-1}b_{n-2}\cdots b_1,s_2=b_1b_2b_3\cdots b_n$，我们要找的是$s_1$与$s_2$成环后相等，并找到一对匹配的数，他们加起来模$m$即为一组解。我们令$s_3=s_2+s_2$，找到$s_3$中所有等于$s_1$的子串，就得到了所有解，这个问题用[KMP]()或是[Z]()都能解决。

还是贴一下代码吧：

```
#include <cstdio>
#include <set>
#include <vector>
#include <iostream>
#include <algorithm>
```

```
using namespace std;

typedef long long LL;

const int maxn = 200005;

LL aa[maxn]; // 读入的a
LL bb[maxn]; // 即上面说的b
vector<LL> gou;
int in[maxn << 2];
LL Z[maxn << 2];
set<LL> ans;

int main()
{
    int n;
    LL m;
    scanf("%d%lld", &n, &m);
    for(int i = 1; i <= n; ++i)
        scanf("%lld", &aa[i]);
    bb[1] = ((aa[1] - aa[n]) + m) % m;
    for(int i = 2; i <= n; ++i)
        bb[i] = ((aa[i] - aa[i-1]) % m + m) % m;
    for(int i = n; i; --i) // 这里用的是Z算法，所以合并成了一个串
    {
        gou.push_back(bb[i]);
        in[gou.size() - 1] = i;
    }
    gou.push_back(-1LL);
    for(int i = 1; i <= n; ++i)
    {
        gou.push_back(bb[i]);
        in[gou.size() - 1] = i;
    }
    for(int i = 1; i <= n; ++i)
    {
        gou.push_back(bb[i]);
        in[gou.size() - 1] = i;
    }
    Z[0] = gou.size();
    for(int i = 1, j = 1, k; i < (int) gou.size(); i = k) // Z算法
    {
        j = max(j, i);
        while(gou[j] == gou[j - i])
            ++j;
        Z[i] = j - i;
        k = i + 1;
        while(k + Z[k - i] < j)
        {
            Z[k] = Z[k - i];
            ++k;
```

```cpp
        }
    }
    for(int i = 1; i < (int) gou.size(); ++i)
        if(Z[i] >= n) // 大力记录答案
            ans.insert((aa[in[i] - 1 ? in[i] - 1 : n] + aa[n]) % m);
    printf("%d\n", (int) ans.size());
    for(auto it = ans.begin(); it != ans.end(); ++it)
        printf("%lld ", *it);
    return 0;
}
```