# 6.5 线性同余方程

**6.5.1 概述**

a,b是整数，形如ax≡b(mod m)，且x是未知整数的同余式称为一元线性同余方程。

容易发现，可以化成ax+my=b的形式，于是就可以用扩欧或欧拉定理求解，当(a,m)|b时有解。

* + 1. **经典例题**

**6.5.2.1 P1082 [NOIP2012 提高组] 同余方程**

**题目描述**

求关于x的同余方程 ax≡1(mod b) 的最小正整数解。

**输入格式**

一行，包含两个正整数 a,b用一个空格隔开。

**输出格式**

一个正整数 x0，即最小正整数解。输入数据保证一定有解。

**输入输出样例**

**输入#1**

3 10

**输出#1**

7

**题解1.欧拉定理**

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define int long long
4. **int** qpow(**int** a,**int** n,**int** b)
5. {
6. **int** ans=1;
7. **while**(n)
8. {
9. **if**(n&1) ans=ans%b\*a%b;
10. a=a%b\*a%b;
11. n>>=1;
12. }
13. **return** ans;
14. }
15. **int** euler(**int** n)
16. {
17. **int** ans=n;
18. **for**(**int** i=2;i\*i<=n;i++)
19. {
20. **if**(n%i==0) ans-=ans/i;
21. **while**(n%i==0) n/=i;
22. }
23. **if**(n>1) ans-=ans/n;
24. **return** ans;
25. }
26. **int** exgcd(**int** a,**int** b,**int** &x,**int** &y)
27. {
28. **if**(b==0)
29. {
30. x=1;
31. y=0;
32. **return** a;
33. }
34. **int** ans=exgcd(b,a%b,x,y);
35. **int** tmp=x;
36. x=y;
37. y=tmp-(a/b)\*y;
38. **return** ans;
39. }
40. **signed** main()
41. {
42. **int** a,b;
43. cin>>a>>b;
44. cout<<qpow(a,euler(b)-1,b)<<endl;
45. **return** 0;
46. }

**题解2.扩欧**

#include<bits/stdc++.h>

**using** **namespace** std;

#define int long long

**int** exgcd(**int** a,**int** b,**int** &x,**int** &y)

{

**if**(b==0)

    {

        x=1;

        y=0;

**return** a;

    }

**int** ans=exgcd(b,a%b,x,y);

**int** tmp=x;

    x=y;

    y=tmp-(a/b)\*y;

**return** ans;

}

**signed** main()

{

**int** a,b;

**int** x,y;

    cin>>a>>b;

    exgcd(a,b,x,y);

    x=(x%b+b)%b;

    cout<<x<<endl;

**return** 0;

}

* + - 1. **洛谷P1516 青蛙的约会**

**题目描述**

两只青蛙在网上相识了，它们聊得很开心，于是觉得很有必要见一面。它们很高兴地发现它们住在同一条纬度线上，于是它们约定各自朝西跳，直到碰面为止。可是它们出发之前忘记了一件很重要的事情，既没有问清楚对方的特征，也没有约定见面的具体位置。不过青蛙们都是很乐观的，它们觉得只要一直朝着某个方向跳下去，总能碰到对方的。但是除非这两只青蛙在同一时间跳到同一点上，不然是永远都不可能碰面的。为了帮助这两只乐观的青蛙，你被要求写一个程序来判断这两只青蛙是否能够碰面，会在什么时候碰面。

我们把这两只青蛙分别叫做青蛙A和青蛙B，并且规定纬度线上东经0度处为原点，由东往西为正方向，单位长度1米，这样我们就得到了一条首尾相接的数轴。设青蛙A的出发点坐标是x，青蛙B的出发点坐标是y。青蛙A一次能跳m米，青蛙B一次能跳n米，两只青蛙跳一次所花费的时间相同。纬度线总长L米。现在要你求出它们跳了几次以后才会碰面。

**输入格式**

输入只包括一行5个整数x，y，m，n，L

其中0<x≠y < =2000000000，0 < m、n < =2000000000，0 < L < =2100000000。

**输出格式**

输出碰面所需要的天数，如果永远不可能碰面则输出一行"Impossible"。

**输入输出样例**

**输入#1**

1 2 3 4 5

**输出#1**

4

**题解**

假设在时间t后相遇，则A在x+t\*m-1(mod L)+1,B在y+t\*n-1(mod L)+1,于是就有方程

(m-n)\*t≡y-x(mod L),扩欧即可

**代码**

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. **typedef** **long** **long** ll;
4. ll exgcd(ll a,ll b,ll &x,ll &y)
5. {
6. **if**(b==0)
7. {
8. x=1;
9. y=0;
10. **return** a;
11. }
12. ll ans=exgcd(b,a%b,x,y);
13. ll tmp=x;
14. x=y;
15. y=tmp-(a/b)\*y;
16. **return** ans;
17. }
18. **int** main()
19. {
20. ll x,y,m,n,l;
21. cin>>x>>y>>m>>n>>l;
22. ll a,b;
23. ll gcd=exgcd(m-n,l,a,b);
24. **if**((y-x)%gcd)
25. {
26. cout<<"Impossible"<<endl;
27. **return** 0;
28. }
29. **else**
30. {
31. ll times=(y-x)/gcd;
32. ll d=abs(l/gcd);
33. cout<<((a\*times)%d+d)%d<<endl;
34. }
35. **return** 0;
36. }