# 6.6逆元

**6.6.1：概述**

如果一个线性同余方程ax≡1（mod b）,则称x是a mod b的逆元，记作a-1。

**6.6.2：扩欧求逆元**

1. **void** exgcd(**int** a, **int** b, **int**& x, **int**& y) {
2. **if** (b == 0) {
3. x = 1, y = 0;
4. **return**;
5. }
6. exgcd(b, a % b, y, x);
7. y -= a / b \* x;
8. }

**6.6.3：费马小定理求逆元**

**费马小定理：**若存在整数a,p且gcd(a,p)=1,即二者互为质数，则有a^(p-1)≡1(mod p)。

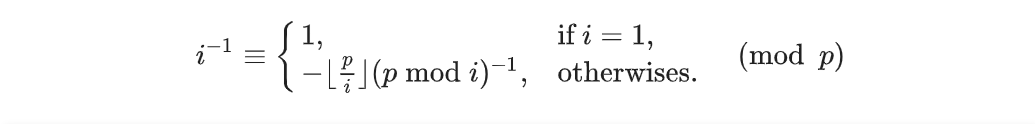
此时 a^(p-2)是a mod p的逆元

由于p一般很大，用快速幂来求

1. ll qpow(ll a,ll n)
2. {
3. ll ans=1;
4. **while**(n)
5. {
6. **if**(n%2) ans=ans\*a%mod;
7. a=a\*a%mod;
8. n/=2;
9. }
10. **return** ans;
11. }

**6.6.4：线性求逆元**

**6.6.4.1 求1-n的逆元**



1. inv[1] = 1;
2. **for** (**int** i = 2; i <= n; ++i) {
3. inv[i] = (**long** **long**)(p - p / i) \* inv[p % i] % p;
4. }

**6.6.4.2 求给定n个数的逆元**

首先计算n个数的前缀积，记为prei，然后使用快速幂或扩展欧几里得法计算pren 的逆元，记为Invn 。

因为Invn是n个数的积的逆元，所以当我们把它乘上an时，就会和an的逆元抵消，于是就得到了a1 到an-1的积逆元，记为Invn-1。

同理我们可以依次计算出所有的Invi，于是ai-1就可以用prei-1\*Invi求得。

所以我们就在O(n+logp)的时间内计算出了n个数的逆元。

1. **void** init()
2. {
3. pre[0] = 1;
4. **for** (**int** i = 1; i <= n; ++i) pre[i] = pre[i - 1] \* a[i] % p;
5. inv[n] = qpow(pre[n], p - 2);
6. // 当然这里也可以用 exgcd 来求逆元,视个人喜好而定.
7. **for** (**int** i = n; i >= 1; --i) inv[i - 1] = inv[i] \* a[i] % p;
8. **for** (**int** i = 1; i <= n; ++i) inv[i] = inv[i] \* pre[i - 1] % p;
9. }

**6.6.5 经典例题**

**6.6.5.1 LiberOJ #161 乘法逆元2**

**题目描述**

这可能是一道模板题。

给定 个正整数 ，求每个数在模 意义下的乘法逆元。

提示：请使用高效的读入方式。

**输入格式**

第一行一个整数n。

第二行n个整数ai。

**输出格式**

一行一个数，表示Σi=1nai\*998244353n-i（mod p）

**输入样例**

5

4 7 8 12 123456

**输出样例**

650798912

**题解：**

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. **typedef** **long** **long** ll;
4. **const** ll mod=1e9+7;
5. **const** ll N=998244353;
6. ll a[5000005];
7. ll pre[5000005];
8. ll qpow(ll a,ll n)
9. {
10. ll ans=1;
11. **while**(n)
12. {
13. **if**(n&1) ans=(ans%mod)\*(a%mod)%mod;
14. a=a\*a%mod;
15. n/=2;
16. }
17. **return** ans;
18. }
20. **int** read() {
21. **int** x=0,f=1;
22. **char** c=getchar();
23. **while**(c<'0'||c>'9'){**if**(c=='-') f=-1;c=getchar();}
24. **while**(c>='0'&&c<='9') x=x\*10+c-'0',c=getchar();
25. **return** x\*f;
26. }
27. ll inv[5000005];
28. **int** main()
29. {
30. **int** n;
31. scanf("%d",&n);
32. pre[0]=1;
33. ll ans=0;
34. **for**(**int** i=1;i<=n;i++)
35. {
36. a[i]=read();
37. pre[i]=pre[i-1]\*a[i]%mod;
38. }
39. inv[n]=qpow(pre[n],mod-2);
40. **for**(**int** i=n;i>=1;i--) inv[i-1]=inv[i]%mod\*a[i]%mod;
41. **for**(**int** i=1;i<=n;i++)
42. {
43. ans=ans\*N%mod+inv[i]\*pre[i-1]%mod;
44. }
45. printf("%lld\n",ans%mod);
46. **return** 0;
47. }