递归写法

```
// 递归写法
typedef long long ll;
// 这里的x为底数，n为指数，m为取模的基数（有的题目不需要取模，那就将这个去掉即可）
ll binartpow(ll x, ll n, ll m) {  // 注意可以提前判断一下m是否为１，因为任何数对１取模都是０
    if(n == 0) return 1;
    else if(n & 1) {
        return x*binarypow(x, n - 1, m) % m;
    }
    else {
        ll temp = binarypow(x, n>>1, m) % m;  // 注意原则是“步步取模”
        return temp * temp % m;
    }
}
```

迭代写法

```
// 迭代写法
typedef long long ll;
ll bianarypow(ll x, ll n, ll m) {
    ll ans = 1;
    x = x % m;  // 注意要加上这一句，即先对x取模，因为取模后相乘最后取模是不变的（根据下图运算法则）
    while(n) {
        if(n&1) ans = ans * x % m;
        n >>= 1;
        x = x * x % m;
    }
    return ans;
}
```

模运算规则

模运算与基本四则运算有些相似，但是除法例外。其规则如下：

1. (a + b) % p = (a % p + b % p) % p （1）
2. (a - b) % p = (a % p - b % p) % p （2）
3. (a * b) % p = (a % p * b % p) % p （3）
4. a ^ b % p = ((a % p)^b) % p （4）

各种运算符所要消耗的cpu时钟

1. +、-：需要２个cpu时钟
2. 位运算只需要１个cpu时钟
3. 乘法需要4个cpu时钟
4. 除法需要４0个cpu时钟

参考资料

算法学习笔记(4)：快速幂
快速幂