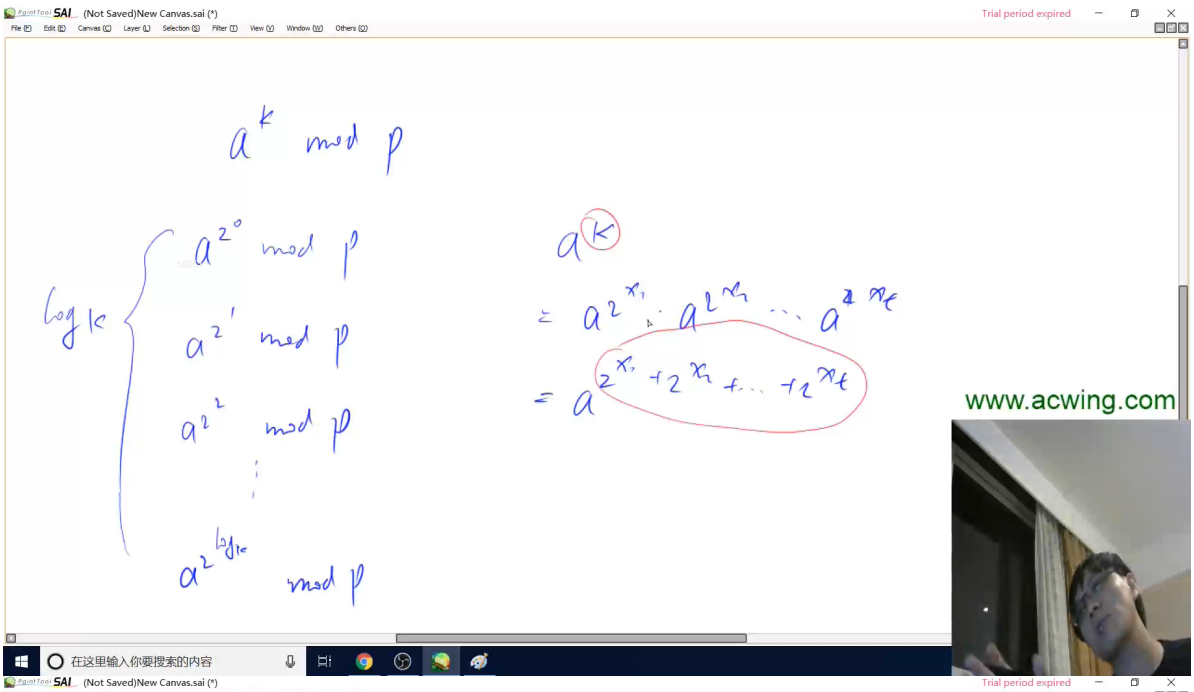
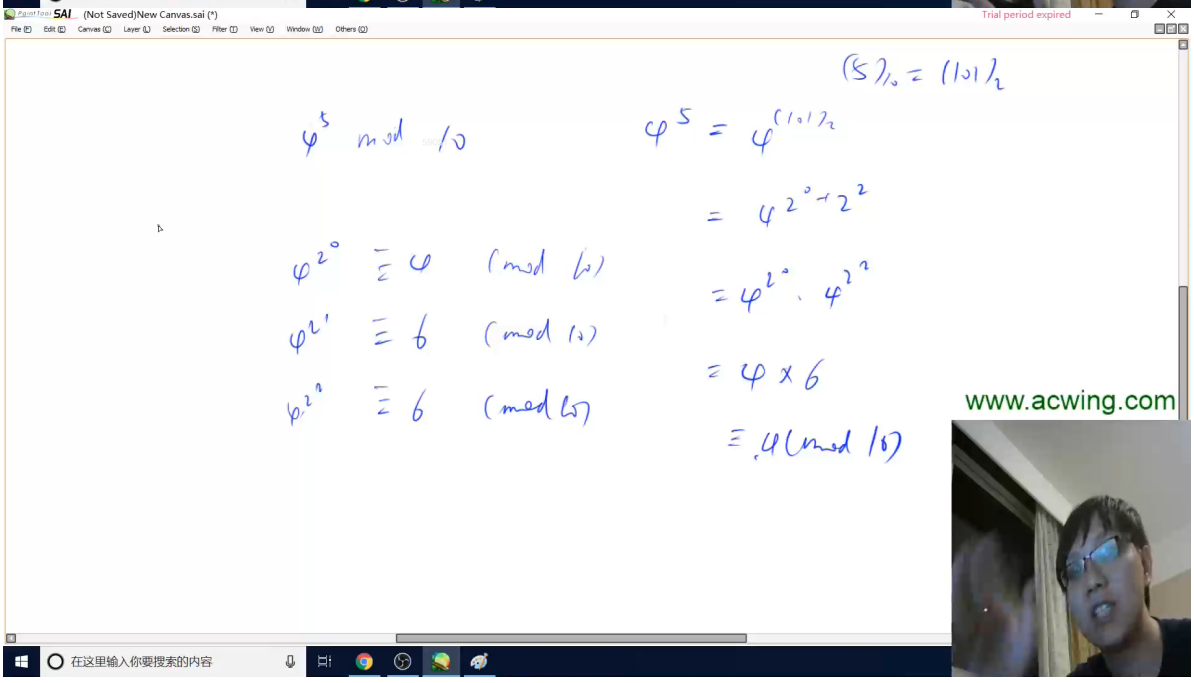
快速幂

取模运算：a mod p 表示a除以p的余数





以下为y总代码：

```
#include
#include

using namespace std;

typedef long long LL;
```

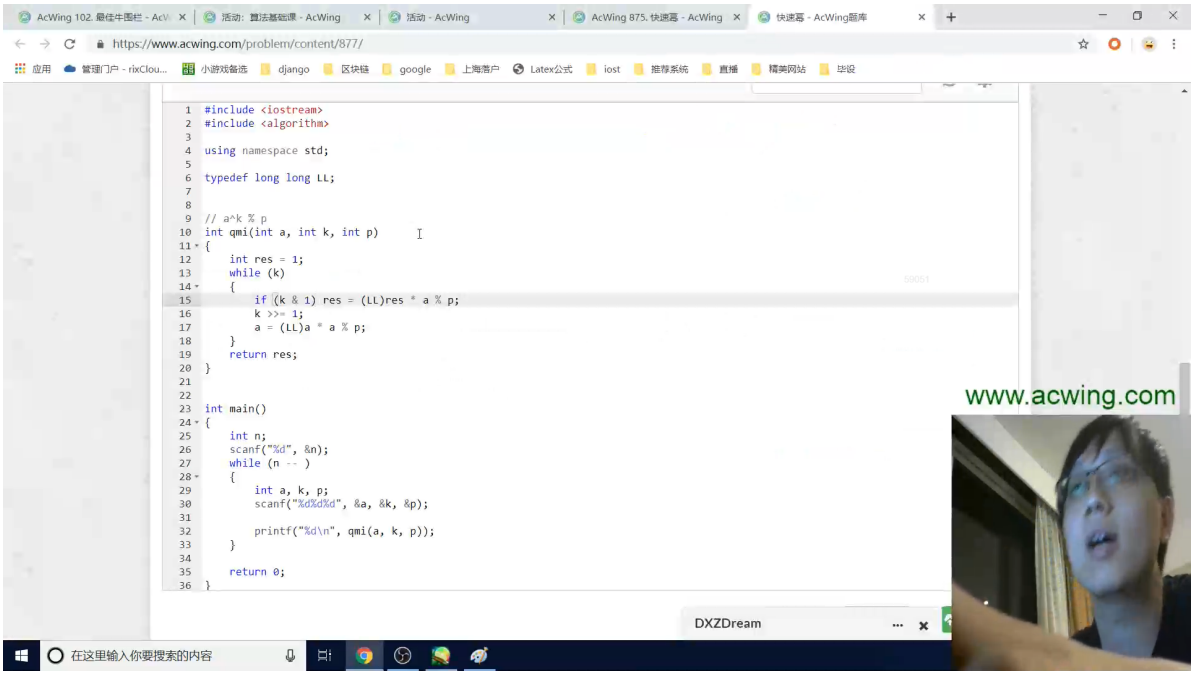
```
LL qmi(int a, int b, int p)
{
    LL res = 1 % p;
    while (b)
    {
        if (b & 1) res = res * a % p;
        a = a * (LL)a % p;
        b >>= 1;
    }
    return res;
}

int main()
{
    int n;
    scanf("%d", &n);
    while (n -- )
    {
        int a, b, p;
        scanf("%d%d%d", &a, &b, &p);
        printf("%lld\n", qmi(a, b, p));
    }
```

```
    return 0;
```

```
}
```

作者：yxc
链接：https://www.acwing.com/activity/content/code/content/50237/
来源：AcWing
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。



对于 a ^ b来说，若果把 b 写成2 进制，那么b 就可以写成若干二次幂之和，如13 的二进制 1101，于是 3 号位 、2号位、0号位就都是1，那么就可以得到13 = 2^3 + 2^2 + 2^1 = 8 + 4 + 1。所以a ^13 = a^8 * a^4 * a^1。

通过同样的推导，我们可以把任意的a^b 表示成 a^(2^k)……、a^8、a^4、a^2、a^1中若干的乘积。若果二进制的i号位为1.那么想中的a^(2^i)就被选中。于是可以得到计算a^b的大致思路：令i 从0到k枚举b的二进制的每一位，如果为1 那就累计a^(2^i)。注意

a^(2^k)……、a^8、a^4、a^2、a^1前一项总是等于后一项的平方。具体步骤。

（1）初始令ans = 1,用来存放累积的结果。

（2）判断b的二进制末尾是否为1 ，（及判断 b&1 是否为 1），也可以理解为判断b 是否为奇数。如果是的话，令ans乘上a的值。

（3）令a平方，并使b右移一位，（也可以理解为，b/2）

（4）只要b 大于0，就返回（2）。

```
typedef long long ll
ll binaryPow(ll a, ll b, ll m){
    ll ans = 1;
    while(b > 0){
        if(b & 1){
            ans = ans * a % m;
        }
        a = a * a % m;
        b >>= 1;
    }
    return ans;
}
```

例：a^13

b	b&1	ans	a
		1	a
1101	1	1*a=a	a^2
110	0	a	a^4
11	1	a*a^4 = a^5	a^8
1	1	a^5 * a^8 = a ^ 13	