Modular Exponentiation

recursive implemetation

```
#include <bits/stdc++.h>
using namespace std;

//exponetiation O(log(n))
long long int exp(int x,int n)
{
	long long int y;
	if(n == 0)
		return 1;
	if(n%2 == 0)
	{
		y = exp(x,n/2);
		return y*y;
	}
	else
		return x*exp(x,n-1);
}

//modular exponentaition O(log(n))
long long int modexp(int a,int n,int m)
{
	long long int y=0;
	if(n == 0)
		return 1;
	if( n%2 == 0)
	{
		y = modexp(a,n/2,m);
		return ((y%m)*(y%m))%m;
	}
	else
		return (a*modexp(a,n-1,m))%m;
}
int main()
{
	int a,b,c;
	cin>>a>>b>>c;
	cout<<exp(a,b);
	cout<<endl;
	cout<<modexp(a,b,c);
}
```