Fermat’s Little Theorem

Fermat’s little theorem states that given a prime number **p** and any integer **a,**

$$(a^p) \cong a \pmod{p}$$

In other words, the number **(a^p - a)** is an integer multiple of **p**.

If a is not divisible by p, then

$$(a^{p-1}) \cong 1 \pmod{p}$$

In other words, **(a^{p-1} - 1)** is an integer multiple of **p**.

Proof:

We have,

$$(a^p) \bmod p = a \bmod p$$

$$a^p \cong a \pmod{p}$$

Dividing by a on both sides, where a is not divisible by p

$$\mathbf{(a^{p-1}) \cong 1 \pmod{p}}$$

For example, if a = 2 and p = 7, then 2^6 = 64, and 64 − 1 = 63 = 7 × 9 is thus a multiple of 7.

Use of Fermat’s little theorem

Fermat’s little theorem can be used to find a multiplicative modulo inverse, given that m is **prime**.

$$B = A^{-1} \bmod m,$$

where B is multiplicative modulo inverse of A

According to Fermat’s little theorem, for any integer a and prime p,

$$a^p \cong a \pmod{p}$$

If a and p are relatively prime,

$$(a^{p-1}) \cong 1 \pmod{p}$$

Rewriting the above equation as,

$$a^{p-2}.a \cong 1 \pmod{p}$$

Multiplying by a^{-1} on both sides, we get,

$$a^{p-2}.a.a^{-1} \cong a^{-1} \pmod{p}$$

$$a^{p-2} \cong a^{-1} \pmod{p}$$

Hence if we want to find A^{-1} mod m, then

$$A^{m-2} \cong A^{-1} \pmod{m}$$

or,

$$A^{-1} \bmod m = (A^{m-2}) \bmod m$$

R.H.S. of the above equation can easily be calculated using modular exponentiation.

Pseudocode

```
function GCD(a,b)

    //   Check if b equals 0 then return 'a' as the GCD
    if b equals 0
        return a
    //   Otherwise recursively call GCD(b,a mod b)
    else
        return GCD(b,a mod b)

/* Input a and b are non-negative integers and m is the modulo,  returns a^b mod m*/
function modularExpo(a,b,m)

    //   Base Case
    if a equals 0
        return 0
    if b equals 0
        return 1

    //   If b is even
    if b mod 2 equals 0
        res = modularExpo(a,b/2) mod m
        return (res * res) mod m
    //   Else if b is odd
    else
        res = modularExpo(a,(b - 1)/2) mod m
        return ((a mod m) * (res * res) mod m) mod m


function modInverse(a, m)
    //   if gcd(a, m) is not equal to 1 then inverse doesn't exist
    if gcd(a, m) not equals 1
        print("Inverse doesn't exist")

    inverse = modularExpo(a, m-2, m);
    print(inverse)
```

Time complexity: $O(\log_2 m)$