**Contents:-**

1. **Modular Exponentiation**
2. **Euclid’s Algorithm for GCD**
3. **Chinese Remainder Theorem**
4. **Euler’s Remainder Theorem**
5. **Fermat’s Little Theorem**
6. **Quadratic Residue**
7. **Solving ax + by = d**
8. **Prime Number Generation**
9. **Convolution Codes**

**PART1: MODULAR EXPONENTIATION**

In Information Theory we often need to use modulo operation.

e.g 5 mod 3 (or 5 %3) = 2, similarly 3 mod 5 = 3

(In this document **^** sign represents exponent i.e. **x^y** is x to the power y)

Sometimes the numbers can be very big, e.g. if we are trying to do, e.g. (X^Y) mod z

e.g. **2^90** = 1237940039285380274899124224.

This will cause overflow in our calculator or computer program.

Calculating (2^90) mod 7 will need some other strategy.

We use the following properties:-

1. **(X \* Y) mod Z = (X mod Z \* Y mod Z) mod Z**
2. **(A^B) mod C = ((A mod C) ^ B) mod C**

Please see the following examples:-

**Example 1: (8^4) mod 5**

= ((8 mod 5) ^ 4) mod 5

We know that 8^5 = 3

This gives us (3^4) mod 5 = 81 mod 5 = 1

**Example 2:- (8 ^ 32) mod 9**

Notice how we do this step-by-step:-

(8^1) mod 9 = 8

(8^2) mod 9 = (8 \* 8) mod 9 = 64 mod 9 = 1

(8^4) mod 9 = ((8^2) \* (8^2)) mod 9 = ((8^2 mod 9) ^ 2) mod 9

= (1^2) mod 9 = 1

(8^8) mod 9 = ((8^4)\*(8^4))mod 9 = ((8^4 mod 9)^2) mod 9

= (1^2) mod 9 = 1

(8^16) mod 9 = ((8^8 mod 9)^2) mod 9 = (1^2) mod 9 = 1

(8^32) mod 9 = ((8^16 mod 9)^2) mod 9 = (1^2) mod 9 = 1

**Example 3:- ( 19 ^ 9) mod 3**

Note that here the exponent 9 is not a power of 2.

Now 9 in binary is 1001

(19^9) = 19 ^ (1001)

(19^9) = 19^8 \* 19^1

19^1 mod 3 = 19 mod 3 = 1

19^2 mod 3 = (1^2) mod 3 = 1

19^8 mod 3 = (1^3) mod 3 = 1

Therefore (19^9) mod 3 = 1 \* 1 mod 3 = 1

**Example 4:- (3200) mod 50**

3^1 mod 50 = 3

3^2 mod 50 = 9 mod 50 = 9

3^4 mod 50 = 81 mod 50 = 31

3^8 = ((3^4 mod 50) ^2) mod 50 = 31 ^ 2 mod 50 = 961 mod 50 = 11

3^16 = (11^2) mod 50 = 121 mod 50 = 21

3^32 = (21^2) mod 50 = 441 mod 50 = 41

3^64 = (41^2) mod 50 = 1681 mod 50 = 31

3^128 = (31^2) mod 50 = 11

Now 3^200 = 3^128 \* 3^64 \* 3^8 (because 128+64+8 = 200)

Therefore (3 ^ 200) mod 50 = (11 \* 31 \* 11) mod 50 = 3751 mod 50

**(3 ^ 200) mod 50 = 1 (Final answer)**

**PART2: EUCLID’S ALGORITHM for GCD**

* Euclid’s algorithm is based on the fact that the GCD of two numbers does not change if the larger number is replaced by the difference between the two numbers

**Example1: GCD of 27 and 45**

(27, 45)

(27, 18) --- Replace larger number 45 by the difference 45-27=18

(9, 18) --- Replace 27 by 27-18

(9, 9) --- Replace 18 by 18-9

(9,0) --- Replace 9 by 9-9

Therefore GCD is 9

**Example2: GCD of 17 and 41 (both are prime)**

(17, 41)

(17, 24) ---- replace 41 by difference 41-17

(17, 7) ---- replace 24 by 24-17

(10, 7) --- keep replacing greater no by difference

(3, 7)

(3, 4)

(3, 1)

(2, 1)

(1, 1)

(0, 1)

**GCD of 17 and 41 is 1**

**Example 3: GCD of 17 and 41 using Modulo arithmetic**

GCD (17, 41)

= GCD ( 17, 41%17) 41 = 2\*17 + 8

= GCD ( 17, 8)

= GCD (17%8, 8)

= GCD(1, 8)

=1

**Modulo method is useful** for large numbers or two numbers with a big difference between them

**Example 4: GCD of 65536 and 100**

GCD (65536, 100)

= GCD ( 65536 % 100, 100)

= GCD (36, 100)

= GCD ( 36, 100%36)

= GCD ( 36, 28)

= GCD ( 36%28, 28)

= GCD ( 8, 28)

= GCD (8, 28%8)

= GCD ( 8, 4)

= 4.

**PART3: CHINESE REMAINDER THEOREM**

Consider the following problem:-

* You are given N chocolates.
* If you divide N chocolates equally among 5 people you are left with 1 chocolate.
* If you divide N chocolates equally among 7 people you are left with 1 chocolate
* If you divide N chocolates equally among 11 people you are left with 3 chocolates
* Find the value of N

--- We solve this using Chinese remainder theorem.

--- First we have to understand the term **congruence**.(symbol of congruence is **≡** )

From the above data, we can write:-

N%5 = 1, N%7 = 1, N%11 = 3

---- OR ----

N ≡ 1mod 5 (we say that N is congruent to 1 mod 5)

N ≡ 1 mod 7

N ≡ 3 mod 11

In general, if you are given

X ≡ a1 mod m1

X ≡ a2 mod m2

X ≡ a3 mod m3

The formula for X is

**X = ( M1\*x1\*a1 + M2\*x2\*a2 + M3\*x3\*a3 ) mod M ……….(equation 1)**

Where

* M = m1\*m2\*m3
* M1 = M/m1 = m2\*m3, similarly M2 = M/m2 = m1\*m3 and M3 = M/m3 = m1\*m2
* a1, a2 and a3 are already given to us
* What are x1, x2 and x3?
* x1 is the multiplicative inverse of M1 i.e. M1\*x1 = 1 mod m1

(**don’t worry**, a simple numerical example will make this clear)

So let us go directly to the numerical example. We will take the same values given above.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **M = M1\*m2\*m3 = 5\*7\*11= 385** | | |  |  |
| a1 = 1 | m1 = 5 | M1=M/m1= 385/5=77 | | x1=? **3** | 1\*77\*3= 231 |
| a2 = 1 | m2 = 7 | M2=M/m2=385/7=55 | | x2=? **6** | 1\*55\*6=330 |
| a3 = 3 | m3 = 11 | M3=M/m3=385/11=35 | | x3=? **6** | 3\*35\*6= 630 |
|  |  |  |  |  | 231+330+630=**1191** |
|  |  |  |  |  |  |

**Example 1:**

N ≡ 1mod 5

N ≡ 1 mod 7

N ≡ 3 mod 11

**To calculate x1:**

M1\*x1 = 1mod 5

77 \* x1 = 1 mod 5

we can rewrite this as

(77 \* m1) mod 5 = 1

We can replace 77 by 77mod5

2 \* x1 = 1 mod 5

Now we take all values of x1 starting from 1,2,3…. To see which value satisfies the above equation

(2 \* 3) = 1 mod 5 because

(2 \* 3 ) mod 5 =1

therefore **x1 = 3**

**To calculate x2:**

55 \* x2 = 1 mod 7

(55 \* x2) mod 7 = 1

Replace 55 by 55 mod 7

(6 \* x2) mod 7 = 1

Try all values for x2 = 1,2,3……

**x2 = 6**

**To calculate x3:**

35 \* x3 = 1 mod 11

(35 \* x3) mod 11 = 1

(2 \* x3) mod 11 = 1

**x3 = 6**

Write these values in the table above.

Using equation 1, N = (1191) mod 385 = 36

Cross check your answer:-

36 mod 5 =1

36 mod 7 = 1

36 mod 11 = 3

**So our answer is correct. (N = 36)**

* For more practice examples on Chinese remainder theorem see this video:

<https://www.youtube.com/watch?v=Y5RcMWiUyyE>

* An interesting link on modular exponentiation:

<https://www.math.upenn.edu/~mlazar/math170/notes06-3.pdf>

**PART4: EULER’S REMAINDER THEOREM**

EULER’S TOTIENT FUNCTION: (Note the word is totient, not quotient).

Given a number n with prime factors a,b.c such that

n = (a^p) \* (b^q) \* (c^r) where

a,b,c are prime numbers and p,q,r are positive integers,

then the **Euler totient function** of n is defined as

φ(n) = n \* (1-1/a) \* (1-1/b) \* (1-1/c)

**Example1:** Find the totient function of 35

35=5×7

φ(35)=35 \* (1–1/5) \* (1–1/7) = 24

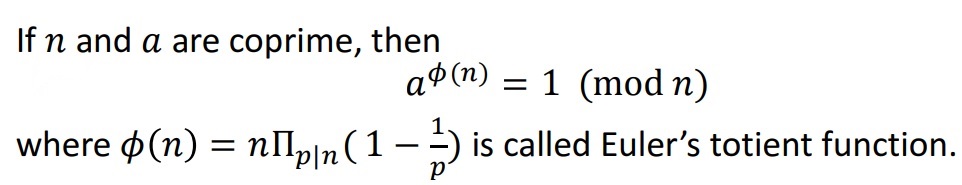
Thus the totient function of 35 is **24**.

**Example2**: **Find the remainder when**3**76 is divided by 35.**

**Solution:**

1. In the above example we found the totient function of 35, which is 24.
2. First we try to reduce the exponent i.e.
3. Remainder of (76 / φ(35)) = 76 / 24 = 4
4. Remaining power is 4
5. Hence, (34 mod 35 ) = 81 mod 35 = 11
6. Thus the final remainder comes out to be 11.

**The formal definition of Euler’s Remainder theorem:-**



* **Note** that n and a are prime **w.r.t each other** but may not be prime by **themselves.**

**PART5: FERMAT’S LITTLE THEOREM**

If **a** is an [integer](https://artofproblemsolving.com/wiki/index.php/Integer), **p** is a [prime number](https://artofproblemsolving.com/wiki/index.php/Prime_number) and **a**is not [divisible](https://artofproblemsolving.com/wiki/index.php/Divisibility) by  **p**

then $a^{p-1}\equiv 1 \pmod {p}$.

P = an integer Prime number

a = an integer which is not multiple of P

Let a = 2 and P = 17

According to Fermat's little theorem

2 17 - 1  ≡ 1 mod(17)

we got 65536 % 17 ≡ 1

that mean (65536-1) is an multiple of 17

Example2: Calculate 3**31** (mod 7)

By FLT (Fermat’s Little Theorem),

37-1 ≡ 1 mod 7

36 ≡ 1 mod 7

3^31 mod 7

= 3 \* (3^30) mod 7

= 3 \* (( 3 ^ 6) ^ 5) mod 7

= 3 \* (1^5) mod 7

= 3 mod 7

= 3

**A frequently used corollary of Fermat's Little Theorem is $a^p \equiv a \pmod {p}$.**

The above example just proved this.

**PART6: QUADRATIC RESIDUE**

**Definition:** Given integers a and m, **a** is called a quadratic residue of **m** if:-

1. **gcd (a, m) = 1** i.e. a and m are prime w.r.t each other
2. there exists an integer x such that **x2 = a mod m**

Example 1:- Is 5 a quadratic residue of 11?

If we choose x=4, then 4**2** = 16 mod 11 = 5.

Therefore we can say that 5 is a **quadratic residue** of 11.

How about 7?

No, there does not exist any x such that x2= 7 mod 11 (i.e. x2mod 11 =7)

Therefore 7 is a **quadratic non-residue** of 11

Given m=11, let us check all numbers from 1 to 10 (11-1)

|  |  |  |
| --- | --- | --- |
| n | n**2** | n**2** mod 11 |
| 1 | 1 | 1 |
| 2 | 4 | 4 |
| 3 | 9 | 9 |
| 4 | 16 | 5 |
| 5 | 25 | 3 |
| 6 | 36 | 3 |
| 7 | 49 | 5 |
| 8 | 64 | 9 |
| 9 | 81 | 4 |
| 10 | 100 | 1 |

The set of quadratic residues of 11 are 1,3,4,5,9

i.e. QR**11** = { 1,3,4,5,9}

NOTE:- For any odd prime p, number of quadratic residues are (p-1)/2 and number of quadratic non-residues are also (p-1)/2.

**Part7: Solving ax + by = d using Euclid’s gcd method.**

**( Extended Eucidean algorithm).**

**Definition:** Given integers a and b, there is always an integral solution to the equation ax+by=d.

**Example1:**

Find integers x and y such that 135x + 50y = 5 (1)

First we use the Euclid method to find gcd(135, 50):-

135 = 2 \* 50 + 35 (2)

50 = 1 \* 35 + 15 (3)

35 = 2 \* 15 + 5 (4)

15 = 3 \* 5 + 0 (stop, gcd=5). (5)

Going backwards,

5 = 35 - 2 \* 15 from (4)

= 35 – 2 \* (50-35)

= 35 – 2 \* 50 + 2 \* 35 expand above line

= 3 \* 35 – 2 \* 50

= 135 – 2 \* 50

= ( 2 \*50 + 35) - 2\*50 from (2)

= 3 \* 135 – 8 \* 50

5 = x \* 135 + y \* 50

Where x=3 and y = -8 (answer)

**PART8: Prime Number Generation**

Any integer n (n>1) which has no divisors other than 1 and n itself is called a prime number.

One of the most common methods to find all prime numbers upto some value X is the Sieve of Eratosthenes.

**Steps:-**

1. Create a list of consecutive integers from 2 to X ( 2, 3, 4, …. X).
2. Initially let p=2, the first prime number.
3. Starting from p, count up in increments of p and strike out all multiples of p (but not p itself); note that some of them may have already been striked out.
4. Find the next number greater than p which is not marked and repeat step 3; otherwise stop.
5. All numbers which are still unmarked are prime numbers.

Example: Let X = 20

01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20

P=2

01 02 03 ~~04~~ 05 ~~06~~ 07 ~~08~~ 09 ~~10~~ 11 ~~12~~ 13 ~~14~~ 15 ~~16~~ 17 ~~18~~ 19 ~~20~~

P=3

01 02 03 ~~04~~ 05 ~~06~~ 07 ~~08~~ ~~09~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~

P=4, 5, 6, 7 …20(no change)

01 02 03 ~~04~~ 05 ~~06~~ 07 ~~08~~ ~~09~~ ~~10~~ 11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~

The numbers which are **not** striked out are prime.

**PART9: Convolution Codes**

* Linear block codes process a block of bits at a time.
* Convolution codes on the other hand operate on the current block as well as previous blocks stored in memory.
* Each output bit is obtained by modulo-2 logic operations on a combination of incoming bits (or blocks) and previously stored bits (or blocks).
* Convolution codes are denoted by a three tuple **(n, k, m).**

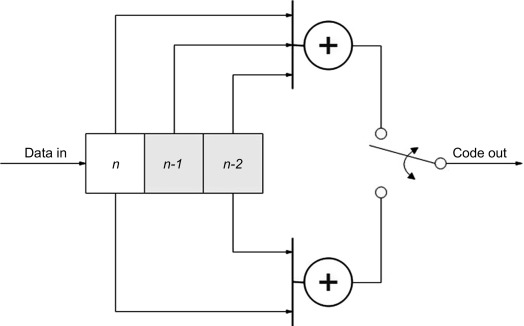
n = number of output bits

k = number of input bits (Usually k=1 for serial data streams).

m = memory size (no of shift registers or flip-flops used to store previous bits).

* Code Rate is defined as k/n
* The memory elements (shift register or flip-flops are initialized to 0 at the start of the operation.

The diagram below is a (2,1,3) convolutional encoder:

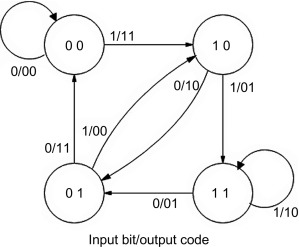


* Current incoming bit is n.
* n-1 and n-2 are previous bits stored in the shift register.
* Every new bit enters from the left and the rightmost bit is then discarded.
* In this diagram, **n-1 and n-2 bits define the current state** of the encoder.
* There are two outputs here:-
* The upper output is modulo-2 of n, n-1 and n-2 bits.
* The lower output is modulo-2 of n and n-2 bits.
* The encoder will have 4 possible states depending on values of bits n-1 and n-2: 00,01,10 and 11. A new bit results in a right shift and therefore a change in state.

See the table below:-

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n (incoming bit)** | **n-1 (previous bit)** | **n-2 (previous to previous bit)** | **Current State**  **(n-1, n-2)** | **Current Output1 (n⊕n-1⊕n-2)** | **Current Output2 (n⊕n-2)** | **Next state after shift ( n, n-1)** |
| 0 | 0 | 0 | 00 | 0 | 0 | 00 |
| 1 | 0 | 0 | 00 | 1 | 1 | 10 |
| 0 | 0 | 1 | 01 | 1 | 1 | 00 |
| 1 | 0 | 1 | 01 | 0 | 0 | 10 |
| 0 | 1 | 0 | 10 | 1 | 1 | 01 |
| 1 | 1 | 0 | 10 | 0 | 0 | 11 |
| 0 | 1 | 1 | 11 | 0 | 0 | 01 |
| 1 | 1 | 1 | 11 | 1 | 1 | 11 |

Same table expressed as a state diagram:



For this encoder

* k=1 (number of incoming bits) and K=3 (shift register stages)
* G1 = 1+x+x2 or (1, 1, 1)
* G2 = 1+x2 or (1, 0, 1)
* Here G1 and G2 are called generator polynomials for the two outputs.
* In these polynomials, x1 indicates a bit shifted by 1 position (also written as a delay of 1 unit D)
* x2 indicates a bit shifted by two places or a delay by two units etc.

Other than the table format and the state diagram, there are two more representations of the encoder:-

-- **Tree diagram**

-- **Trellis diagram.**

For these, see the video links below.

On the receiver side, two popular decoding techniques are

-Fano Decoding

- Viterbi decoding.

**Viterbi decoding** (also called Maximum Likelihood decoding) is more popular and is also explained in the video links.

Please go through the videos and practice couple of more examples on your own for

Table, state diagram, tree diagram and trellis diagram.

<https://www.youtube.com/watch?v=8a76JT0Ke7M>

<https://www.youtube.com/watch?v=cK1IchFQDfU>