①

IT-21001

②

Q1) Prove Fermat's Little Theorem and use it to compute $a^{p-1} \bmod p$ for given values of $a = 7$, $p = 13$. Then discuss how this theorem is useful cryptographic algorithms like RSA.

Soln:

Fermat's Little Theorem:

If p is prime and $\gcd(a,p) = 1$, then:

$$a^{p-1} \equiv 1 \pmod p$$

proof:

multiply set $1, 2, \ldots, p-1$ by a : $a, 2a, \ldots, (p-1)a \bmod p$ is a permutation of Z_p^*.

So: $a^{p-1} \cdot (p-1)! \equiv (p-1)! \bmod p$

$\Rightarrow a^{p-1} \equiv 1 \bmod p$ (proved)

computing $a^{p-1} \equiv 1 \bmod p$ where $p = 13$ and $a = 7$.

$$7^{13-1} = 7^{12} \equiv 1 \pmod{13}$$

Ans: 1

Use in RSA:

- RSA uses: $c = m^e \bmod n$,
  $m = c^d \bmod n$
- correctness relies on:
  $m^{\phi(n)} \equiv 1 \bmod n$

Fermat's Little theorem underpins RSA encryption/decryption by ensuring modular exponentiation behaves predictably.

Q2 Euler Totient Function: Compute $\phi(n)$ for $n = 35, 45, 100$. Prove that if a and n are coprime then $a^{\phi(n)} \equiv 1 \bmod n$.

Soln: i) $n = 35 = 5 * 7$:

$\therefore \phi(35) = (5-1)(7-1) = 4 * 6 = \boxed{24}$

ii) $n = 45 = 3^2 * 5$:

$\phi(45) = 45\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right) = 45 * \frac{2}{3} * \frac{4}{5} = \boxed{24}$

iii) $n = 100 = 2^2 \times 5^2$:

$\therefore \phi(100) = 100\left(1-\frac{1}{2}\right)\left(1-\frac{1}{5}\right) = 100 \times \frac{1}{2} \times \frac{4}{5} = \boxed{40}$

Euler's Theorem - statement and Proof:

If $\gcd(a,n) = 1$, then:

$$a^{\phi(n)} \equiv 1 \mod n$$

Proof idea!

+ let Z_n^* = set of integers < n and coprime to n
- Multiply each by a ⇒ permutation
- Product of elements stays the same:

$$a^{\phi(n)} \cdot r_1 r_2 \cdots r_{\phi(n)} \equiv r_1 r_2 - r_{\phi(n)} \mod n \Rightarrow a^{\phi(n)} \equiv 1 \mod n$$

[Proved]

Q3 Solve the system of congruences using the CRT and prove that x congruent to 11 on mod $N = 3 \times 4 \times 5 = 60$

$$x \equiv 2 \mod 3, x \equiv 3 \mod 4, x \equiv 1 \mod 5$$

Soln! Given,

$x \equiv 2 \mod 3$ $\quad N = 3 \times 4 \times 5 = 60$

$x \equiv 3 \mod 4$

$x \equiv 1 \mod 5$

step-1: compute components

Let: $N_1 = 60/3 = 20$

$N_2 = 60/4 = 15$

$N_3 = 60/5 = 12$

Find inverses y_i such that:

→ $20\, y_1 \equiv 1 \bmod 3 \Rightarrow y_1 = 2$ (since $20 * 2 = 40 \equiv 1 \bmod 3$)

→ $15\, y_2 \equiv 1 \bmod 4 \Rightarrow y_2 = 3$

→ $12\, y_3 \equiv 1 \bmod 5 \Rightarrow y_3 = 3$

Step 2: Now applying CRT formula

$x \equiv a_1 N_1 y_1 + a_2 N_2 y_2 + a_3 N_3 y_3 \bmod 60$

$x \equiv 2 \cdot 20 \cdot 2 + 3 \cdot 15 \cdot 3 + 1 \cdot 12 \cdot 3$

$x \equiv 80 + 135 + 36 = 251 \bmod 60$

$\therefore x \equiv 11 \bmod 60$

Q4 Soln:

Prime factorization of $561 = 3 \times 11 \times 17$

Korselt's criterion:

check if for each prime p, $p-1 \mid 561-1$

$\rightarrow 3-1 = 2 \mid 560$

$\rightarrow 11-1 = 10 \mid 560$

$\rightarrow 17-1 = 16 \mid 560$

all conditions satisfied

Fermat's Test

check $a^{560} \equiv 1 \mod 561$ for some $\gcd(a, 561) = 1$

Try $a = 2$:

$2^{560} \equiv 1 \mod 561$ which is true.

this holds for many bases a, so 561 passes Fermat's test.

Thus, 561 is a Carmichael number.

⑥

IT-21001

Q5 $\phi(17) = 16 \rightarrow$ we want g such that:

$g^k \not\equiv 1 \bmod 17$ ~~fro~~ for any $k < 16$

let's try $g = 3$, test orders via prime divisors of 16: 2, 4, 8

$\rightarrow 3^2 = 9 \not\equiv 1 \bmod 17$

$\rightarrow 3^4 = 81 \equiv 13 \not\equiv 1 \bmod 17$

$\rightarrow 3^8 = 13^2 = 169 \equiv -1 \bmod 17$

$\rightarrow 3^{16} \equiv 1 \bmod 17$

$\therefore$ it passes all. So, 3 is a primitive root of modulo 17.

Ans: 3 is a generator of $\mathbb{Z}^*_{17}$

Q6 soln' let's try sucessive powers of 3 modulo 17

if $x = 1$, $3^1 \bmod 17 = 3 \bmod 17 = 3$

if $x = 2$, $3^2 \bmod 17 = 9 \bmod 17 = 9$

if $x = 3$, $3^3 \bmod 17 = 27 \bmod 17 = 10$

if $x = 4$, $3^4 \bmod 17 = 81 \bmod 17 = 13$

$\therefore 3^4 \equiv 13 \mod 17$

Ans: $x = 4$

Q7) Diffie-Hellman allows two parties to securely compute a shared secret over a public channel using modular exponentiation.

Mathematical Basis:

→ Choose a large prime p and primitive root g

→ Each party picks a private key:

- Alice: a, computes $A = g^a \mod p$
- Bob: b, compute $B = g^b \mod p$

→ Shared key:

$$K = B^a \equiv A^b \equiv g^{ab} \mod p$$

Security relies on:

The Discrete Logarithm Problem (DLP):
Given g, p, and $g^a \mod p$, it's computationally hard to find a. The discrete-logarithm ensure Diffie-Hellman is secure by making it hard to reverse $g^a \mod p$ and recover the private key.

IT-21001

Q8) 1. Substitution cipher

- Mechanism: Replaces each letter with another letter (e.g., Caesar cipher)
- Key space: 26! (for mono alphabetic), small for Caesar (25 keys)
- Frequency vulnerability: High (same letter → same cipher).

Example: Plaintext: HELLO
Caesar (+3): KHOOR

2. Transposition cipher:

- Mechanism: Rearranges letter positions, letters stay unchanged.
- Key Space: Depends on message length (factorial permutations).
- Frequency vulnerability: Less vulnerable (letter frequency preserved)

Example: HELLO
Transposed (swap 1-2, 3-4): EHLLO

3. Playfair Cipher:

- Mechanism: Encrypts digraphs using a 5x5 matrix.
- Key space: Large (based on 25-letter keyword permutations).
- Frequency vulnerability: Lower (breaks mono-letter frequency).

Example: Plaintext: HELLO → HE LL O → XM ck gz (using digraph rules)

99) Encryption:

→ Plaintext: DEPTOFICTMBSTU

→ Numeric: 3 4 15 19 14 5 8 2 19 12 1 18 19 20

→ Encrypt $E(x) = (5x+8) \bmod 26$

Plain x	3	4	15	19	14	5	8	2	19	12	1	18	19	20
5x+8	23	28→2	83→5	103→25	78→0	33→7	48-22	18	103→25	68→16	13	98→20	103→25	108→4
Cipher	X	C	F	Z	A	H	W	S	Z	Q	N	U	Z	E

∴ ciphertext: XCFZAHWSZQNUZE

Decryption:

→ Compute $a^{-1} \bmod 26$. since

$5 \cdot 21 = 105 \equiv 1, \; a^{-1} = 21$

IT-21001

→ Decrypt $D(y) = 21,(y-8) \bmod 26$. Applying to each cipher letter recovers:

Cipher y	23	2	5	25	0	7	22	18	25	16	13	20	25	4
y−8	15	−6→20	−3→23	17	−8→18	−1→25	14	10	17	8	5	12	17	−4→22
×26 mod 26	3	4	15	19	14	5	8	2	19	12	1	18	19	20
Plain	D	E	P	T	O	F	I	C	T	M	B	S	T	U

Recovered: DEPTOFICTMBSTU

(i.e., "Dept of ICT, MBSTU")

Q10) Novel Cipher: SubPerm Cipher

A hybrid cipher combining Substitution + Permutation with lightweight PRNG-based key scheduling.

Encryption process:

1. Key: A 3-digit numeric seed (e.g, 493)
   - used to generate a pseudo-random substitution table and permutation pattern.

2. Substitution (Monoalphabetic)
   - Use the key as a seed for PRNG to shuffle the alphabet (e.g, using Fisher-Yates shuffle)
   - E.g., A → Q, B → L, ... , Z → M

3. Permutation (Block Transposition)
   - Divide ciphertext into 5-letter blocks.

(4)

- Generate block permutation from key on each block)
- Permute characters within each block using the pattern.

Decryption Process:

1. Reverse permutation using inverse of the key pattern.
2. Reverse substitution using the inverse lookup table.

Cryptanalysis & Vulerabilites:

Type	Risk
substitution-only frequency	Mitigated by permutation disrupting patterns.
Small key space (3-digit seed)	vulnerable to brute-force (only 1000 keys)
known-plaintext attack	If both perm & sub tables are discovered.
Block-level confusion	Increase diffusion (permutation step helps)