1. **Apply Extended Euclid algorithm to compute GCD (99,78). Show all the computations.**

Ans. 1)

GCD (99 , 78) using Extended Euclid's Algorithm

Q	A	B	R
1	99	78	21
3	78	21	15
1	21	15	6
2	15	6	3
1	6	3	0
	3	0	

GCD (99 , 78) = 3

Computations

- a = b * q + r
- 99 = 78 * 1 +21
- 78 = 21 * 3 +15
- 21 = 15 * 1 +6
- 15 = 6 * 2 +3
- 6 = 3 * 2 +0

2. a) What is Vigenère Cipher? Explain it's working using suitable example.

Ans. 2)

- Vigenere Cipher is a method of encrypting alphabetic text.
- It uses a simple form of polyalphabetic substitution.
- A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets.
- The encryption of the original text is done using the *Vigenère square or Vigenère table*.
    - The table consists of the alphabets written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar Ciphers.
    - At different points in the encryption process, the cipher uses a different alphabet from one of the rows.
    - The alphabet used at each point depends on a repeating keyword.

(2) Vigener's cipher :-

(1) It is a poly alphabetic Substitution cipher.

→ Plain text : GIVEMONEY
key : LOCK

→

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	

M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
12	13	14	15	16	17	18	19	20	21	22	23	24	25

(ii) Encryption :-

Plaintext :	G	I	V	E	M	O	N	E	Y
	↓	↓	↓	↓	↓	↓	↓	↓	↓
	6	8	21	4	12	14	13	4	24
key :	L	O	C	K	L	O	C	K	L
	↓	↓	↓	↓	↓	↓	↓	↓	↓
	11	14	2	10	11	14	2	10	11
	6+11 = 17	8+14 = 22	21+2 = 23	4+10 = 14	12+11 = 23	14+14 = 28	13+2 = 15	4+10 = 14	24+11 = 35
	17 mod 26	22 mod 26	23 mod 26	14 mod 26	23 mod 26	28 mod 26	15 mod 26	14 mod 26	35 mod 26
	17	22	23	14	23	2	15	14	9
	R	W	X	O	X	C	P	O	J

Cipher text :- RWXOX CPOJ

decrypt :-

Plaintext :-

	G	I	V	E	M	O	N	E	Y
	↓	↓	↓	↓	↓	↓	↓	↓	↓
	6	8	21	4	12	14	13	4	24
Key :	L	O	C	K	L	O	C	K	L
	↓	↓	↓	↓	↓	↓	↓	↓	↓
	11	14	2	10	11	14	2	10	11
Ciphertext :	R	W	X	O	X	C	P	O	J
	(17)	(22)	(23)	(14)	(23)	(2)	(15)	(14)	(9)

$D_i = (E_i - k_i) \bmod 26$
$= (17 - 11) \bmod 26$
$= 6 \bmod 26$
$\boxed{D_i = 6}$ → G

$D_i = (E_i - k_i) \bmod 26$
$= (X - C) \bmod 26$
$= (23 - 2) \bmod 26$
$= 21 \bmod 26$
$\boxed{D_i = 21}$ → V

$D_i = (E_i - k_i) \bmod 26$
$= (W - O) \bmod 26$
$= (22 - 14) \bmod 26$
$= 8 \bmod 26$
$\boxed{D_i = 8}$ → I

$D_i = (E_i - k_i) \bmod 26$
$= (O - K) \bmod 26$
$= (14 - 10) \bmod 26$
$= 4 \bmod 26$
$\boxed{D_i = 4}$ → E

$D_i = (E_i - k_i) \bmod 26$
$= (X - L) \bmod 26$
$= (23 - 11) \bmod 26$
$= (12 \bmod 26)$
$\boxed{D_i = 12}$
↓
M

$D_i = (E_i - k_i) \bmod 26$
$= (J - L) \bmod 26$
$= (9 - 11) \bmod 26$
$= -2 \bmod 26$
$26 - 2 = 24$
$\boxed{D_i = 24}$
↓
Y

$D_i = (E_i - k_i) \bmod 26$
$= (C - O) \bmod 26$
$= (2 - 14) \bmod 26$
$= -12 \bmod 26$
$26 - 12 = 14$
$\boxed{D_i = 14}$
↓
O

$D_i = (E_i - k_i) \bmod 26$
$= (P - C) \bmod 26$
$= (15 - 2) \bmod 26$
$= 13 \bmod 26$
$\boxed{D_i = 13}$
↓
N

$D_i = (E_i - k_i) \bmod 26$
$= (O - K) \bmod 26$
$= (14 - 10) \bmod 26$
$= 4 \bmod 26$
$\boxed{D_i = 4}$
↓
E

plaintext :- GIVE MONEY

2. b) Demonstrate the working of encryption and decryption procedure in Hill Cipher with respect to following parameters:

Plain Text : ACOLLEGE **Key :**

7	**8**
19	**3**

3. Explain Key Calculation Procedure in Simplified DES algorithm.

Ans. 3)

The Simplified Data Encryption Standard (S-DES) is a simplified version of the original Data Encryption Standard (DES) algorithm. It uses a shorter key length and fewer rounds for simplicity.

The key calculation procedure in S-DES involves generating two subkeys from an initial 10-bit key. Here's an explanation of the key calculation procedure in S-DES:

Initial 10-Bit Key (K): The S-DES algorithm starts with an initial 10-bit key, represented as K. This key is provided as input to the encryption and decryption processes.

Permutation P10: The first step in key calculation is to permute the 10-bit key using a fixed permutation called P10. P10 rearranges the bits in the following way:

3 5 2 7 4 10 1 9 8 6

The bits of the initial key are rearranged according to this permutation. The resulting 10-bit value is divided into two 5-bit halves, often denoted as left and right halves (L0 and R0).

Key Generation Rounds:

Round 1:

Left Circular Shift (LS-1):Both L0 and R0 are separately subjected to a left circular shift by one bit. This means that the leftmost bit is moved to the rightmost position, and the other bits are shifted one position to the left.

L0: 3 5 2 7 4 10 1 9 8 6

R0: 5 2 7 4 10 1 9 8 6 3

Permutation P8: After the left circular shift, both L0 and R0 are subjected to another fixed permutation called P8. P8 selects and permutes specific bits from the 10-bit halves to generate the first subkey, often denoted as K1.

P8: 6 3 7 4 8 5 10 9

The bits selected by P8 from both L0 and R0 are combined to form K1, which is a 8-bit subkey.

Round 2:

Left Circular Shift (LS-2): Both L0 and R0 (after the first round) are separately subjected to a left circular shift by two bits this time.

L1: 2 7 4 10 1 9 8 6 3 5

R1: 7 4 10 1 9 8 6 3 5 2

Permutation P8: After the left circular shift, both L1 and R1 are subjected to the P8 permutation again to generate the second subkey, often denoted as K2.

P8: 6 3 7 4 8 5 10 9

The bits selected by P8 from both L1 and R1 are combined to form K2, which is another 8-bit subkey.

At the end of the key calculation procedure, you have generated two 8-bit subkeys, K1 and K2, from the original 10-bit key K. These subkeys are used in the S-DES encryption and decryption processes.

In S-DES, these subkeys are used in a Feistel network structure to perform the initial and final permutations, as well as the rounds of substitution and permutation. This process helps encrypt and decrypt the plaintext.

4. Explain in detail about encryption procedure in IDEA algorithm.

Ans. 4)

The International Data Encryption Algorithm (IDEA) is a symmetric-key block cipher that operates on 64-bit blocks of data and uses a 128-bit key for encryption and decryption. IDEA is a well-regarded encryption algorithm known for its security and efficiency. The encryption procedure in IDEA in detail:

IDEA Encryption Procedure:

IDEA operates on 64-bit blocks of plaintext and uses a 128-bit key. Here's a step-by-step explanation of the encryption process:

1. Key Expansion: The 128-bit encryption key is expanded into 52 round subkeys. These round subkeys will be used in each of the 8.5 rounds of the encryption process. Each subkey is 16 bits in length.

2. Initial Permutation: The 64-bit block of plaintext is subjected to an initial permutation. This permutation rearranges the bits in the block according to a fixed pattern.

3. Rounds: IDEA consists of 8.5 rounds (16 rounds divided by 2, where 0.5 rounds are applied to the middle of the data block). Each round consists of the following steps:

a. **Subkey Mixing:** The 64-bit data block is divided into four 16-bit blocks (X1, X2, X3, X4). Each of these blocks is then mixed with a 16-bit round subkey.

b. **Substitution (S-Box):** Each of the four 16-bit blocks is passed through a substitution (S-box) step. IDEA uses eight 16x16 S-boxes in this step.

c. **Permutation (P-Box):** After the substitution, each of the four 16-bit blocks goes through a permutation (P-box) step, which shuffles the bits within the blocks.

d. **Linear Transformation:** The outputs of the S-boxes and P-boxes are combined in a linear transformation step, which involves bitwise XOR and modulo addition.

4. Final Permutation: After all rounds are completed, the resulting data block is subjected to a final permutation, which is the inverse of the initial permutation.

5. Output: The final 64-bit block, after the final permutation, is the ciphertext.

It's important to note that IDEA is a symmetric-key encryption algorithm, which means the same key is used for both encryption and decryption. To decrypt data encrypted with IDEA, you would perform the inverse of the encryption process using the same key and round subkeys.

5. Demonstrate the working of RSA decryption algorithm with following parameters:

Cipher Text C = 10 Public Key (e,n) = (5,35)

Ans. 5)

In the RSA algorithm, the encryption and decryption processes are based on the modular exponentiation operation. The encryption operation is defined as $C \equiv M^e \bmod n$, where C is the ciphertext, M is the plaintext, e is the public exponent, and n is the modulus.

Given C=10, e=5, and n=35, the goal is to find the plaintext M.

The decryption operation is defined as $M \equiv C^d \bmod n$, where d is the private exponent. In RSA, d is calculated as the modular multiplicative inverse of e modulo (n), where $\phi(n)$ is Euler's totient function.

To find d, we need to calculate $\phi(n)$. For n=35, $\phi(35)$ is calculated as follows:

$\phi(35) = \phi(5\times7) = (5-1) \times (7-1) = 4\times6 = 24$

Now, find d such that $5 \times d \equiv 1 \bmod 24$.

In this case, d=5 because $5 \times 5 \equiv 1 \bmod 24$.

Now, use d to decrypt the ciphertext C=10 and find the plaintext M:

$M \equiv C^d \bmod n$

$M \equiv 10^5 \bmod 35$

Calculate M:

$M \equiv 100000 \bmod 35$

$M \equiv 5$

So, the plaintext M is 5.

6. Apply the Chinese Remainder Theorem to solve following congruent equations.

X ≡ (mod 3) X ≡ (mod 5) X ≡ (mod 7)

Ans. 6)

To solve the system of congruent equations using the Chinese Remainder Theorem (CRT), follow these steps:

1. Write down the given congruences:

--> $X \equiv 1 \pmod{3}$

--> $X \equiv 2 \pmod{5}$

--> $X \equiv 3 \pmod{7}$

2. Calculate the products of the moduli:

Find N, which is the product of all the moduli (3, 5, and 7):

$N = 3 * 5 * 7 = 105$

3. Find the individual remainders when N is divided by each modulus:

--> $N_1 = N/3 = 35$

--> $N_2 = N/5 = 21$

--> $N_3 = N/7 = 15$

4. **Calculate the modular inverses:**

For each modulus, find the modular inverse of N_i modulo the corresponding modulus. In this case, the modular inverses are as follows:

--> $y_1 \equiv 35^{-1} \pmod{3}$ (Find the modular inverse of 35 modulo 3)

--> $y_2 \equiv 21^{-1} \pmod{5}$ (Find the modular inverse of 21 modulo 5)

--> $y_3 \equiv 15^{-1} \pmod{7}$ (Find the modular inverse of 15 modulo 7)

The modular inverses are:

--> $y_1 \equiv 2 \pmod{3}$

--> $y_2 \equiv 1 \pmod{5}$

--> $y_3 \equiv 1 \pmod{7}$

5. Compute the final solution:

Use the Chinese Remainder Theorem formula to find X:

$X = (a_1 * N_1 * y_1) + (a_2 * N_2 * y_2) + (a_3 * N_3 * y_3) \pmod{N}$

where a_1, a_2, a_3 are the remainders when dividing X by the respective moduli:

--> $a_1 = 1$ (from the first congruence)

--> $a_2 = 2$ (from the second congruence)

--> $a_3 = 3$ (from the third congruence)

Now, plug these values into the formula:

$X = (1 * 35 * 2) + (2 * 21 * 1) + (3 * 15 * 1) \pmod{105}$

$X = 70 + 42 + 45 \pmod{105}$

$X = 157 \pmod{105}$

Finally, reduce X modulo 105 to get the smallest non-negative solution:

$X \equiv 52 \pmod{105}$

So, the solution to the system of congruent equations is $X \equiv 52 \pmod{105}$.