3. Write a program to implement encryption and decryption using Brute force attack cipher.

To help clarify the mathematical steps for encryption and decryption, I'll provide a complete breakdown for both processes using your input/output.

Encryption Process (Caesar Cipher)

Given Input:

- Plaintext: "**KHOOR**"
- Shift: **1**

Mathematical Encryption Formula:

For each character in the plaintext, we apply the following formula to get the encrypted character:

$$E(c) = (P(c) + \text{shift}) \mod 26$$

Where:

- $E(c)$ is the encrypted character.
- $P(c)$ is the position of the character in the alphabet (0 for 'A', 1 for 'B', ..., 25 for 'Z').
- The shift value is the number of positions to move forward in the alphabet.

Step-by-step encryption:

1. **K -> L:**
   - Position of 'K' = 10 (A=0, B=1, ..., K=10)
   - Applying shift:

$$E(K) = (10 + 1) \mod 26 = 11 \quad (\text{L})$$

2. **H -> I:**
   - Position of 'H' = 7
   - Applying shift:

$$E(H) = (7 + 1) \mod 26 = 8 \quad (\text{I})$$

3. **O -> P:**
   - Position of 'O' = 14
   - Applying shift:

$$E(O) = (14 + 1) \mod 26 = 15 \quad (\text{P})$$

4. **O -> P:**
    - Again, for the second 'O' (same calculation as above):

$$E(O) = (14 + 1) \mod 26 = 15 \quad (\text{P})$$

5. **R -> S:**
    - Position of 'R' = 17
    - Applying shift:

$$E(R) = (17 + 1) \mod 26 = 18 \quad (\text{S})$$

Result:

Encrypted text after applying shift 1 to "KHOOR" is "**LIPPS**".

Decryption Process (Caesar Cipher)

Given Input:

- Ciphertext: "**LIPPS**"
- Shift: **1**

Mathematical Decryption Formula:

For each character in the ciphertext, we apply the following formula to get the decrypted character:

$$D(c) = (C(c) - \text{shift}) \mod 26$$

Where:

- $D(c)$ is the decrypted character.
- $C(c)$ is the position of the character in the ciphertext (0 for 'A', 1 for 'B', ..., 25 for 'Z').
- The shift value is the number of positions to move backward in the alphabet.

Step-by-step decryption:

1. **L -> K:**
   - Position of 'L' = 11
   - Applying reverse shift:

$$D(L) = (11 - 1) \mod 26 = 10 \quad (\text{K})$$

2. **I -> H:**
   - Position of 'I' = 8
   - Applying reverse shift:

$$D(I) = (8 - 1) \mod 26 = 7 \quad (\text{H})$$

3. **P -> O:**
   - Position of 'P' = 15
   - Applying reverse shift:

$$D(P) = (15 - 1) \mod 26 = 14 \quad (\text{O})$$

   - Again, for the second 'P' (same calculation as above):

$$D(P) = (15 - 1) \mod 26 = 14 \quad (\text{O})$$

5. **S -> R:**
   - Position of 'S' = 18
   - Applying reverse shift:

$$D(S) = (18 - 1) \mod 26 = 17 \quad (\text{R})$$

Result:

Decrypted text after applying shift 1 to "LIPPS" is "**KHOOR**".

Brute Force Decryption (Checking All Shifts)

In the brute-force decryption, all 26 shifts are tested on the ciphertext "**LIPPS**". Here's the calculation for each shift:

1. **Shift 0**:

$$D(LIPPS, 0) = LIPPS \quad (\text{no change})$$

   Result: "**LIPPS**"

2. **Shift 1**:

$$D(LIPPS, 1) = KHOOR \quad (\text{correct plaintext})$$

   Result: "**KHOOR**"

3. **Shift 2**:

$$D(LIPPS, 2) = JGNNQ$$

   Result: "**JGNNQ**"

4. **Shift 3**:

$$D(LIPPS, 3) = IFMMP$$

   Result: "**IFMMP**"

5. **Shift 4**:

$$D(LIPPS, 4) = HELLO$$

   Result: "**HELLO**"

6. **Shift 5**:

$$D(LIPPS, 5) = GDKKN$$

   Result: "**GDKKN**"

... and so on, testing each shift from 0 to 25. The correct decryption is found at **Shift 1**.

Summary of Results:

- **Encryption** of "KHOOR" with **Shift 1** gives "**LIPPS**".
- **Decryption** of "LIPPS" with **Shift 1** gives back "**KHOOR**".
- The **brute-force decryption** finds the correct plaintext "**KHOOR**" at **Shift 1**.

This matches exactly what you observed in your program's output.

4. Write a program to implement encryption and decryption using Hill cipher.

Let's go through the calculations of the Hill cipher encryption and decryption step-by-step using your provided key matrix and message.

Given:

- **Message**: "help"
- **Key Matrix**:

$$K = \begin{bmatrix} 3 & 3 \\ 2 & 5 \end{bmatrix}$$

- **Alphabet Modulus**: 26 (since there are 26 letters in the alphabet)

We will first convert the message to numerical values, encrypt it, then calculate the modular inverse of the key matrix and use it to decrypt the encrypted message.

Step 1: Convert Message to Numerical Vector

We convert each letter of the message "help" into its corresponding numerical value:

- `h = 7`, `e = 4`, `l = 11`, `p = 15`

Since our key matrix is 2×2, we'll split the message into pairs of two letters:

- First pair: "he" → $\begin{bmatrix} 7 \\ 4 \end{bmatrix}$
- Second pair: "lp" → $\begin{bmatrix} 11 \\ 15 \end{bmatrix}$

Step 2: Encrypt Each Pair

For each pair, we perform matrix multiplication with the key matrix K and take modulo 26.

Encrypting "he" ($\begin{bmatrix}7\\4\end{bmatrix}$):

1. Matrix multiplication:

$$K \cdot \begin{bmatrix}7\\4\end{bmatrix} = \begin{bmatrix}(3\times 7 + 3\times 4)\\(2\times 7 + 5\times 4)\end{bmatrix} = \begin{bmatrix}21+12\\14+20\end{bmatrix} = \begin{bmatrix}33\\34\end{bmatrix}$$

2. Apply modulo 26:

$$\begin{bmatrix}33\\34\end{bmatrix} \mod 26 = \begin{bmatrix}7\\8\end{bmatrix}$$

3. Convert back to letters:
   - `7 = h`, `8 = i`

   So, "he" encrypts to "hi".

Encrypting "lp" ($\begin{bmatrix}11\\15\end{bmatrix}$):

1. Matrix multiplication:

$$K \cdot \begin{bmatrix}11\\15\end{bmatrix} = \begin{bmatrix}(3\times 11 + 3\times 15)\\(2\times 11 + 5\times 15)\end{bmatrix} = \begin{bmatrix}33+45\\22+75\end{bmatrix} = \begin{bmatrix}78\\97\end{bmatrix}$$

2. Apply modulo 26:

$$\begin{bmatrix}78\\97\end{bmatrix} \mod 26 = \begin{bmatrix}0\\19\end{bmatrix}$$

3. Convert back to letters:
   - `0 = a`, `19 = t`

   So, "lp" encrypts to "at".

The **encrypted message** is "hiat".

Step 3: Calculate the Modular Inverse of the Key Matrix K

To decrypt, we need the inverse of K modulo 26.

1. Calculate the determinant of K:

$$\det(K) = (3 \times 5 - 3 \times 2) = 15 - 6 = 9$$

2. Find the modular inverse of the determinant (9) modulo 26. We use the Extended Euclidean Algorithm for this:
   - The modular inverse of 9 modulo 26 is 3 (since $9 \times 3 \equiv 1 \mod 26$).
3. Multiply the adjugate of K by the modular inverse of the determinant:
   - Adjugate of K: Swap the diagonal elements and change the signs of the off-diagonals:

$$\text{adj}(K) = \begin{bmatrix} 5 & -3 \\ -2 & 3 \end{bmatrix} \equiv \begin{bmatrix} 5 & 23 \\ 24 & 3 \end{bmatrix} \mod 26$$

   - Multiply by 3 (modular inverse of determinant):

$$K^{-1} = 3 \cdot \begin{bmatrix} 5 & 23 \\ 24 & 3 \end{bmatrix} \mod 26 = \begin{bmatrix} 15 & 17 \\ 20 & 9 \end{bmatrix} \mod 26$$

The inverse key matrix K^{-1} is:

$$K^{-1} = \begin{bmatrix} 15 & 17 \\ 20 & 9 \end{bmatrix}$$

Step 4: Decrypt Each Encrypted Pair

Decrypting "hi" ($\begin{bmatrix} 7 \\ 8 \end{bmatrix}$):

1. Matrix multiplication with K^{-1}:

$$K^{-1} \cdot \begin{bmatrix} 7 \\ 8 \end{bmatrix} = \begin{bmatrix} (15 \times 7 + 17 \times 8) \\ (20 \times 7 + 9 \times 8) \end{bmatrix} = \begin{bmatrix} 105 + 136 \\ 140 + 72 \end{bmatrix} = \begin{bmatrix} 241 \\ 212 \end{bmatrix}$$

2. Apply modulo 26:

$$\begin{bmatrix} 241 \\ 212 \end{bmatrix} \mod 26 = \begin{bmatrix} 7 \\ 4 \end{bmatrix}$$

3. Convert back to letters:
   - `7 = h`, `4 = e`

   So, "hi" decrypts to "he".

Decrypting "at" ($\begin{bmatrix} 0 \\ 19 \end{bmatrix}$):

1. Matrix multiplication with K^{-1}:

$$K^{-1} \cdot \begin{bmatrix} 0 \\ 19 \end{bmatrix} = \begin{bmatrix} (15 \times 0 + 17 \times 19) \\ (20 \times 0 + 9 \times 19) \end{bmatrix} = \begin{bmatrix} 0 + 323 \\ 0 + 171 \end{bmatrix} = \begin{bmatrix} 323 \\ 171 \end{bmatrix}$$

2. Apply modulo 26:

$$\begin{bmatrix} 323 \\ 171 \end{bmatrix} \mod 26 = \begin{bmatrix} 11 \\ 15 \end{bmatrix}$$

3. Convert back to letters:

   - `11 = l`, `15 = p`

   So, "at" decrypts to "lp".

The **decrypted message** is "help", which matches the original message, verifying that encryption and decryption were successful.

Playfair Cipher – Algorithm Step 1

- **Generate the key Square (5×5):** The key square is a 5×5 grid of alphabets that acts as the key for encrypting the plaintext.
- Each of the 25 alphabets must be unique and one letter of the alphabet (usually J) is omitted from the table (as the table can hold only 25 alphabets).
- If the plaintext contains J, then it is replaced by I.
- The initial alphabets in the key square are the unique alphabets of the key in the order in which they appear followed by the remaining letters of the alphabet in order.

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Cipher – Algorithm Step 2

- **Algorithm to encrypt the plain text:** The plaintext is split into pairs of two letters (digraphs).
- If there is an odd number of letters, a Z is added to the last letter.
- **For example:**
- **PlainText**: "instruments" **After Split:** 'in' 'st' 'ru' 'me' 'nt' 'sz'
- Pair cannot be made with same letter. Break the letter in single and add a bogus letter to the previous letter.
- **Plain Text:** "hello" **After Split:** 'he' 'lx' 'lo'
- Here 'x' is the bogus letter.

Playfair Cipher – Algorithm Rules for Encryption

- **If both the letters are in the same column**: Take the letter below each one (going back to the top if at the bottom).
- **For example:**
- **Diagraph:** "me"

  Encrypted Text: cl

  Encryption:

  m -> c

  e -> l

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Cipher – Algorithm Rules for Encryption

- **If both the letters are in the same row**: Take the letter to the right of each one (going back to the leftmost if at the rightmost position).
- **For example:**
- **Diagraph:** "st"

  Encrypted Text: tl

  Encryption:

  s -> t

  t -> l

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Cipher – Algorithm Rules for Encryption

- **If neither of the above rules is true**: Form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle.
- **For example:**
- **Diagraph:** "nt"

  Encrypted Text: rq

  Encryption:

  n -> r

  t -> q

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

5. Write a program to implement encryption using Playfair cipher.

Playfair Cipher – Algorithm Example

Plain Text: "instrumentsz"

Encrypted Text: gatlzmclrqtx

Encryption:

in -> ga

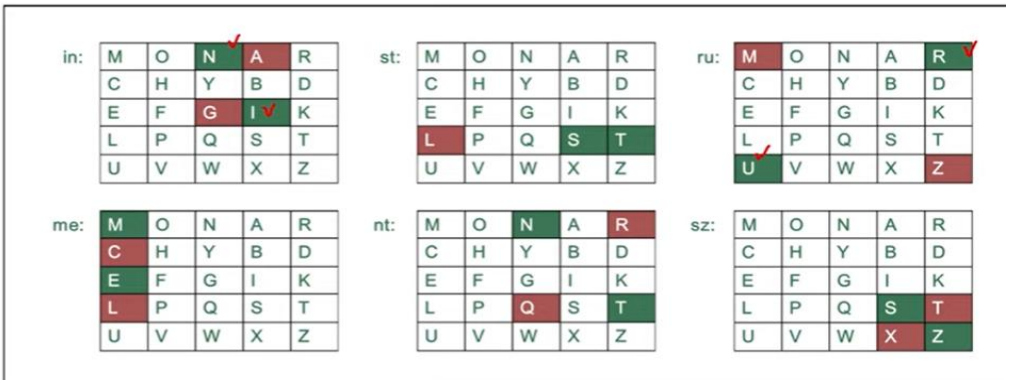
st -> tl

ru -> zm

me -> cl

nt -> rq

sz -> tx



Playfair Cipher – Algorithm Rules for Decryption

- **If both the letters are in the same column**: Take the letter above each one (going back to the top if at the bottom).
- **For example:**
- **Diagraph:** "cl"

  Decrypted Text: me

  Encryption:

  c -> m

  l -> e

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Cipher – Algorithm Rules for Decryption

- **If both the letters are in the same row**: Take the letter to the left of each one (going back to the leftmost if at the rightmost position).
- **For example:**
- **Diagraph:** "tl"

  Decrypted Text: st

  Encryption:

  t -> s

  l -> t

M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

Playfair Cipher – Algorithm Rules for Decryption

- **If neither of the above rules is true**: Form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle.
- **For example:**
- **Diagraph:** "rq"

  Decrypted Text: nt

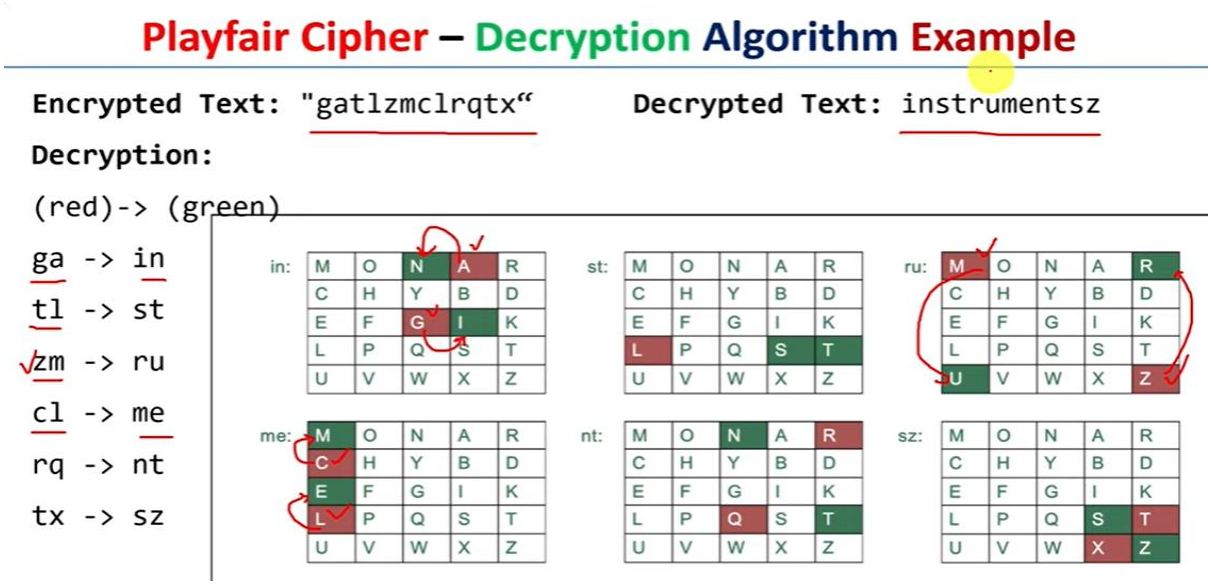
  Encryption:

  r -> n

  q -> t

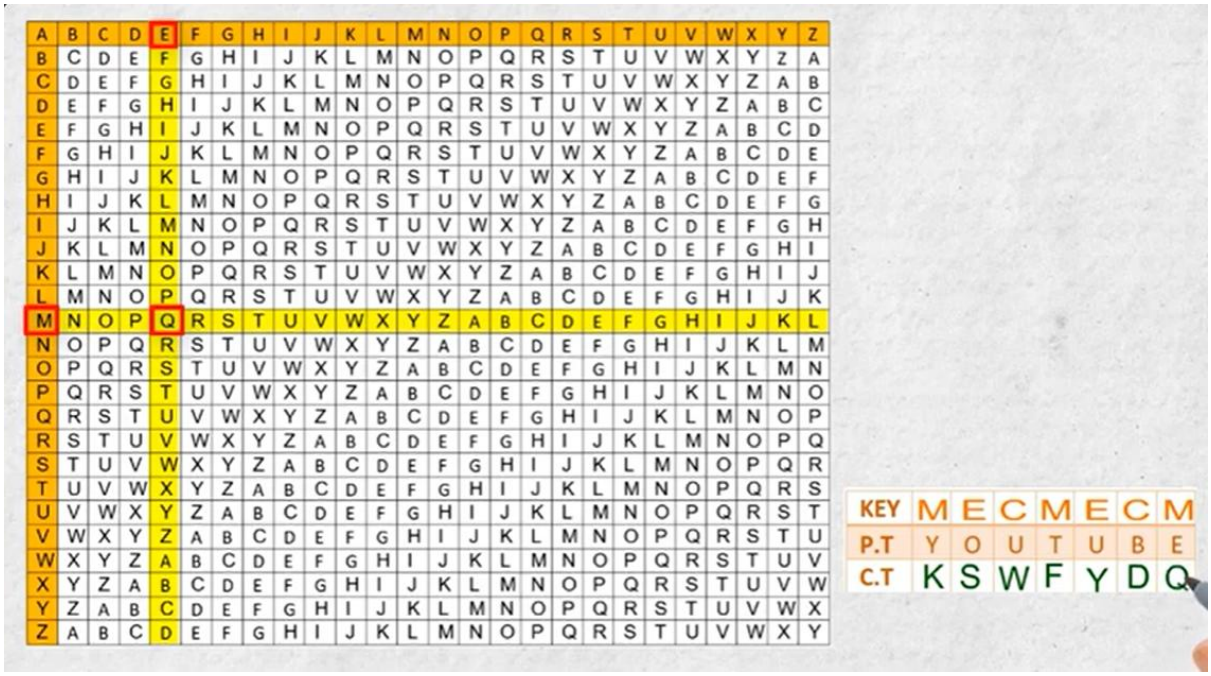
M	O	N	A	R
C	H	Y	B	D
E	F	G	I	K
L	P	Q	S	T
U	V	W	X	Z

6.Write a program to implement decryption using Playfair cipher.



7.Write a program to implement encryption using Poly-Alphabetic cipher.

8.Write a program to implement decryption using Poly-Alphabetic cipher.



Given the message "YOUTUBE" and the key "MEC," we'll calculate the encryption using the Vigenère (Poly-Alphabetic) Cipher step by step.

Step 1: Setup and Mapping

1. **Message**: `YOUTUBE`
   - Convert each letter to its numeric index:
     $Y = 24, O = 14, U = 20, T = 19, U = 20, B = 1, E = 4.$
2. **Key**: `MEC`
   - Repeat the key to match the length of the message: `MECMECM`.
   - Convert each letter in the repeated key to its numeric index:
     $M = 12, E = 4, C = 2, M = 12, E = 4, C = 2, M = 12.$

Step 2: Encrypt Each Letter

Using the encryption formula $C_i = (P_i + K_i) \mod 26$:

1. **Encrypt Y with M**:
$$C_1 = (24 + 12) \mod 26 = 10 \quad (\text{K})$$
2. **Encrypt O with E**:
$$C_2 = (14 + 4) \mod 26 = 18 \quad (\text{S})$$
3. **Encrypt U with C**:
$$C_3 = (20 + 2) \mod 26 = 22 \quad (\text{W})$$
4. **Encrypt T with M**:
$$C_4 = (19 + 12) \mod 26 = 5 \quad (\text{F})$$
5. **Encrypt U with E**:
$$C_5 = (20 + 4) \mod 26 = 24 \quad (\text{Y})$$
6. **Encrypt B with C**:
$$C_6 = (1 + 2) \mod 26 = 3 \quad (\text{D})$$
7. **Encrypt E with M**:
$$C_7 = (4 + 12) \mod 26 = 16 \quad (\text{Q})$$

Result

The **encrypted text** for the message "YOUTUBE" with the key "MEC" is: `KSWFYDQ`.

To decrypt the encrypted message "KSWFYDQ" with the key "MEC" using the Vigenère (Poly-Alphabetic) Cipher, we'll reverse the encryption process.

Step 1: Setup and Mapping

1. **Ciphertext**: `KSWFYDQ`
   - Convert each letter to its numeric index:
     $K = 10, S = 18, W = 22, F = 5, Y = 24, D = 3, Q = 16.$
2. **Key**: `MEC`
   - Repeat the key to match the length of the ciphertext: `MECMECM`.
   - Convert each letter in the repeated key to its numeric index:
     $M = 12, E = 4, C = 2, M = 12, E = 4, C = 2, M = 12.$

Step 2: Decrypt Each Letter

Using the decryption formula $P_i = (C_i - K_i + 26) \mod 26$:

1. **Decrypt K with M**:
$$P_1 = (10 - 12 + 26) \mod 26 = 24 \quad (\text{Y})$$
2. **Decrypt S with E**:
$$P_2 = (18 - 4 + 26) \mod 26 = 14 \quad (\text{O})$$
3. **Decrypt W with C**:
$$P_3 = (22 - 2 + 26) \mod 26 = 20 \quad (\text{U})$$
4. **Decrypt F with M**:
$$P_4 = (5 - 12 + 26) \mod 26 = 19 \quad (\text{T})$$
5. **Decrypt Y with E**:
$$P_5 = (24 - 4 + 26) \mod 26 = 20 \quad (\text{U})$$
6. **Decrypt D with C**:
$$P_6 = (3 - 2 + 26) \mod 26 = 1 \quad (\text{B})$$

7. **Decrypt Q with M**:

$$P_7 = (16 - 12 + 26) \mod 26 = 4 \quad (\text{E})$$

Result

The **decrypted text** for the ciphertext "KSWFYDQ" with the key "MEC" is: `YOUTUBE`.

This confirms that the decryption successfully retrieves the original message "YOUTUBE".

Alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Plain Text	H	O	W	A	R	E	Y	O	U
	7	14	22	0	17	4	24	14	20
Key	N	C	B	T	Z	Q	A	R	X
	13	2	1	19	25	16	0	17	23
Total	20	16	23	19	42	20	24	31	43
Subtract 26 if >25	20	16	23	19	16	20	24	5	17
Cipher Text	U	Q	X	T	Q	U	Y	F	R

Let's go step-by-step through the encryption and decryption processes for the following:

- **Plaintext**: `"howareyou"`
- **Key**: `"ncbtzqarx"`

We will first calculate the encryption and then proceed with the decryption, showing the corresponding mathematical steps.

Encryption Process

Step 1: Convert Plaintext and Key to Numeric Values

Each letter in the plaintext and key is converted to its corresponding numeric position in the alphabet (A=0, B=1, ..., Z=25).

- **Plaintext** `"howareyou"`:
  - `h -> 7`
  - `o -> 14`
  - `w -> 22`
  - `a -> 0`
  - `r -> 17`

    - `e -> 4`
    - `y -> 24`
    - `o -> 14`
    - `u -> 20`
- Key `"ncbtzqarx"`:
    - `n -> 13`
    - `c -> 2`
    - `b -> 1`
    - `t -> 19`
    - `z -> 25`
    - `q -> 16`
    - `a -> 0`
    - `r -> 17`
    - `x -> 23`

Step 2: Add Corresponding Values (Plaintext + Key)

For each character in the plaintext, add the corresponding character in the key. If the result exceeds 25 (i.e., the alphabet's range), we subtract 26 to wrap around.

- `h + n -> 7 + 13 = 20` → Corresponds to `u`
- `o + c -> 14 + 2 = 16` → Corresponds to `q`
- `w + b -> 22 + 1 = 23` → Corresponds to `x`
- `a + t -> 0 + 19 = 19` → Corresponds to `t`
- `r + z -> 17 + 25 = 42` → `42 - 26 = 16` → Corresponds to `q`
- `e + q -> 4 + 16 = 20` → Corresponds to `u`
- `y + a -> 24 + 0 = 24` → Corresponds to `y`
- `o + r -> 14 + 17 = 31` → `31 - 26 = 5` → Corresponds to `f`
- `u + x -> 20 + 23 = 43` → `43 - 26 = 17` → Corresponds to `r`

Step 3: Encrypted Text

After adding the corresponding values, the **Encrypted Text** becomes:

```mathematica
Encrypted Text: "uqxtquyfr"
```

Decryption Process

Step 1: Convert Ciphertext and Key to Numeric Values

For decryption, we first convert the ciphertext and key to numeric values just like we did for encryption.

- **Ciphertext** `"uqxtquyfr"`:
  - `u -> 20`
  - `q -> 16`
  - `x -> 23`
  - `t -> 19`
  - `q -> 16`
  - `u -> 20`
  - `y -> 24`
  - `f -> 5`
  - `r -> 17`

- Key `"ncbtzqarx"`:
  - `n -> 13`
  - `c -> 2`
  - `b -> 1`
  - `t -> 19`
  - `z -> 25`
  - `q -> 16`
  - `a -> 0`
  - `r -> 17`
  - `x -> 23`

Step 2: Subtract Corresponding Values (Ciphertext - Key)

For each character in the ciphertext, subtract the corresponding character in the key. If the result is negative, we add 26 to get a positive value and ensure that it stays within the alphabet range.

- `u - n -> 20 - 13 = 7` → Corresponds to `h`
- `q - c -> 16 - 2 = 14` → Corresponds to `o`
- `x - b -> 23 - 1 = 22` → Corresponds to `w`
- `t - t -> 19 - 19 = 0` → Corresponds to `a`
- `q - z -> 16 - 25 = -9` → `-9 + 26 = 17` → Corresponds to `r`
- `u - q -> 20 - 16 = 4` → Corresponds to `e`
- `y - a -> 24 - 0 = 24` → Corresponds to `y`
- `f - r -> 5 - 17 = -12` → `-12 + 26 = 14` → Corresponds to `o`
- `r - x -> 17 - 23 = -6` → `-6 + 26 = 20` → Corresponds to `u`

Step 3: Decrypted Text

After subtracting the corresponding values, the **Decrypted Text** becomes:

mathematica Copy code

```
Decrypted Text: "howareyou"
```