**LAB: 01**

**TITLE:** Implementation and Analysis of the Caesar Cipher using CrypTool

**OBJECTIVES:**

1. To Implement the Caesar Cipher
2. To Analyze Encryption and Decryption Processes

**THEORY:**

It is a type of substitution cipher in which each letter in the plaintext is 'shifted' a certain number of places down the alphabet. For example, with a shift of 1, A would be replaced by B, B would become C, and so on.

**Example:**

Plaintext: HELLO WORLD

Ciphertext: NKRRU CUXRJ (with key/shift=6)

Encryption of a letter by a shift k can be described mathematically as,

c = (m + k) mod 26

m = Plaintext

c = Ciphertext

k = key

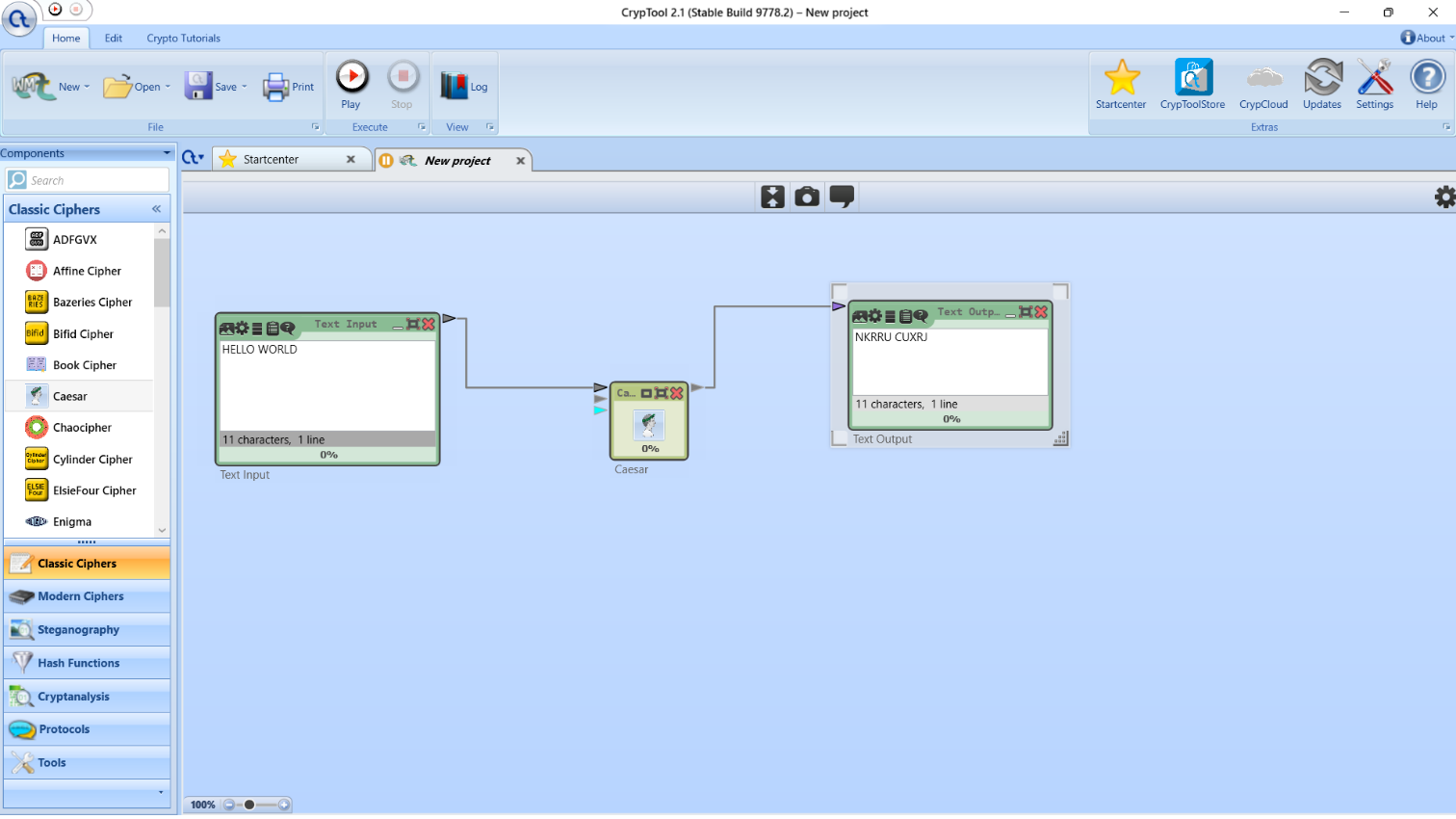
Similarly, Decryption

m = (c – k + 26) mod 26

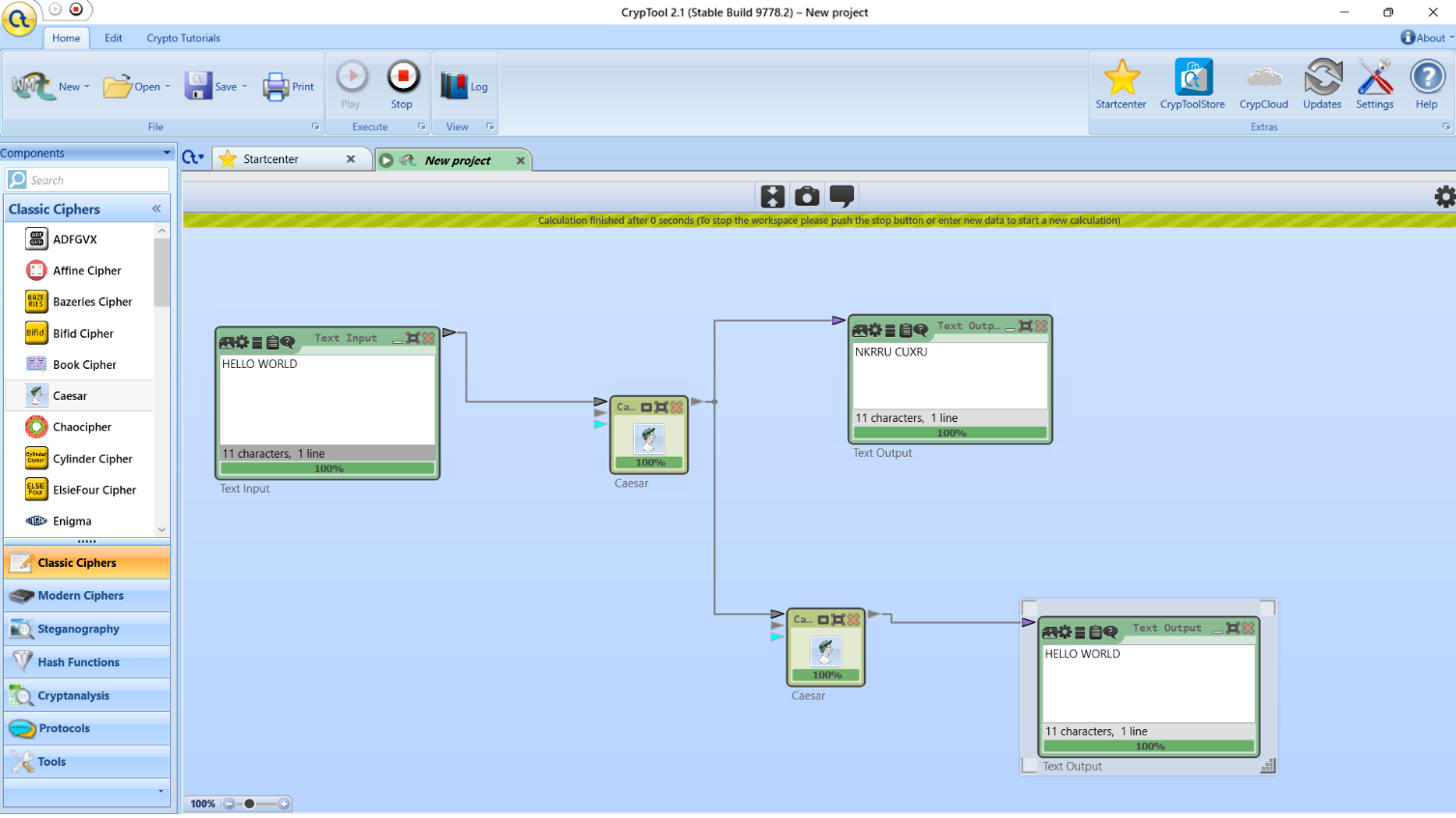
Here, we number each English alphabet starting from 0 (A) to 25 (Z).

**IMPLEMENTATION:**

Encryption Process:



Decryption Process:



**ANALYSIS:**

1. Demonstrates simplicity and educational value.
2. Straightforward encryption and decryption.
3. Vulnerable to brute-force and frequency analysis.
4. Inadequate for securing sensitive information.
5. Less effective compared to modern ciphers.

**CONCLUSION:**

The Caesar Cipher is a straightforward encryption technique that is easy to implement and understand. However, it offers minimal security and is easily broken with modern cryptanalysis methods. While useful for educational purposes, it is not suitable for securing sensitive information. Future work could explore more secure ciphers and key management practices.

**LAB: 02**

**TITLE:** Implementation and Analysis of the Playfair Cipher using CrypTool

**OBJECTIVES:**

1. To Implement the Playfair Cipher
2. To Analyze Encryption and Decryption Processes

**THEORY:**

The Playfair algorithm works on the basis of 5 x 5 matrix.

Matrix construction method:

* Given the keyword, write down its letter from left to right and top to down on the matrix without repeating the letter and fill the remainder of the matrix with the remaining letters in the alphabetic order.
* I and J are considered as same letter.

Encryption process:

* Each time a pair of letters is encrypted.
* The pair cannot contain same letter. If so, separate it with the filler (X).

E.g. HELLO WORLD→HE LX LO WO RL DX

Rules:

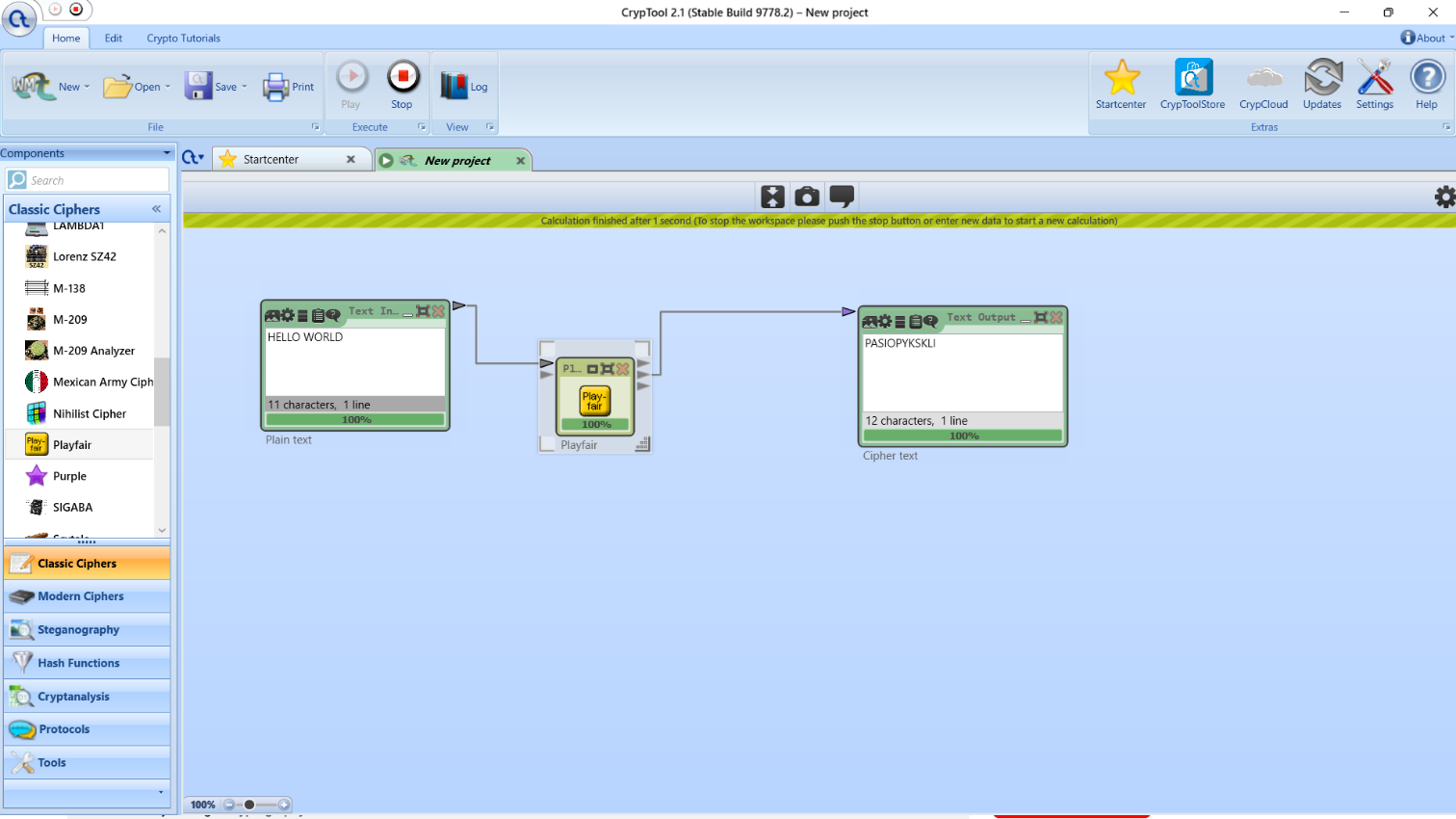
1. If the two letters are in same row, then replace it with the letter one position ahead in the same row (circularly).
2. If the letters are in same column, then replace it with the letter on position below in the same column (circularly).
3. Otherwise, replace it with the letter that lies in its own row and column occupied by the other plaintext letter.

Decryption process:

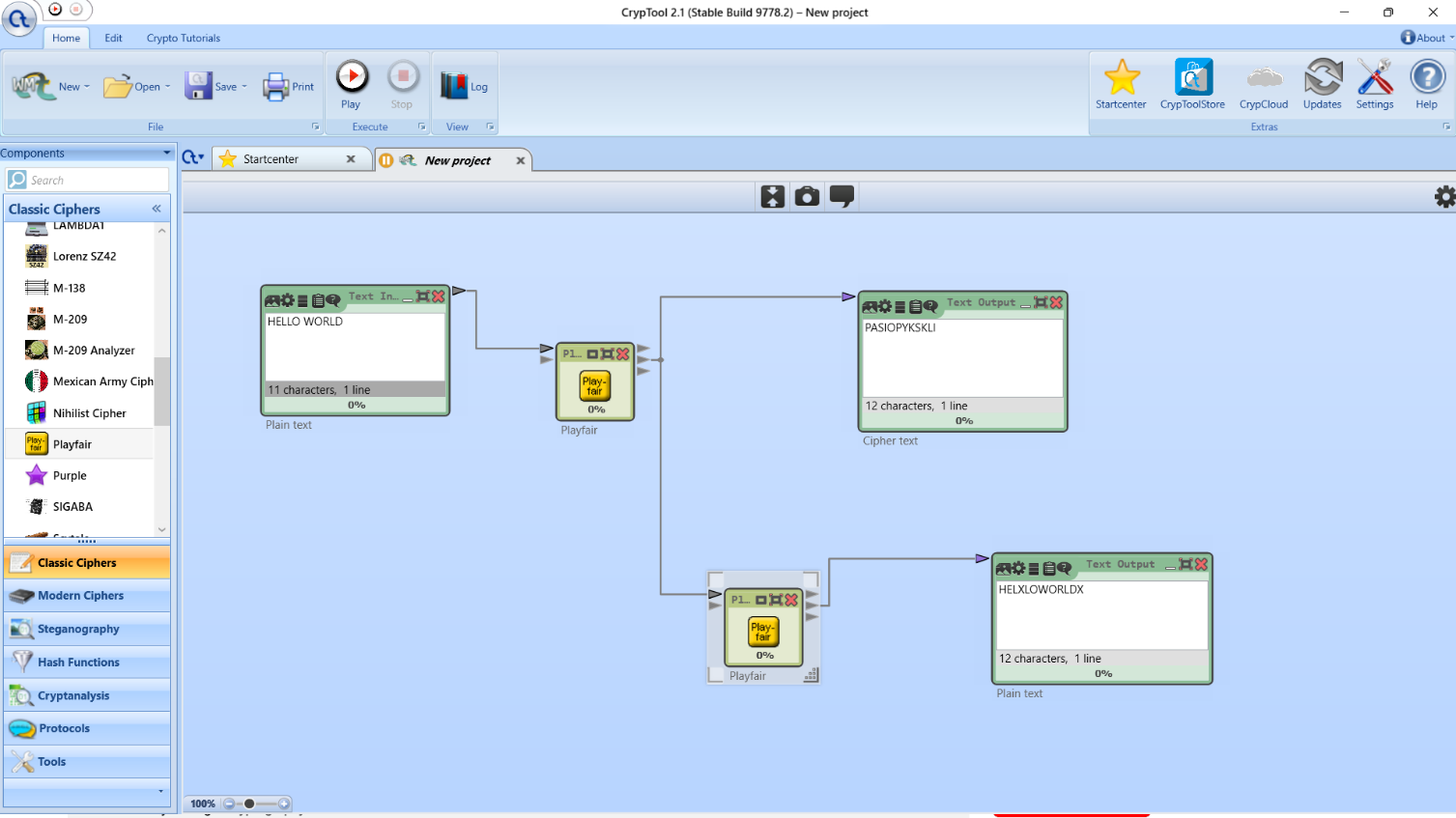
* Decryption is nearly identical to the encryption process, except for rules 1 and 2 we must take the letters to the left and above respectively. Also, we remove any extra filler (X) in the decrypted text to reveal the final plaintext.

**IMPLEMENTATION:**

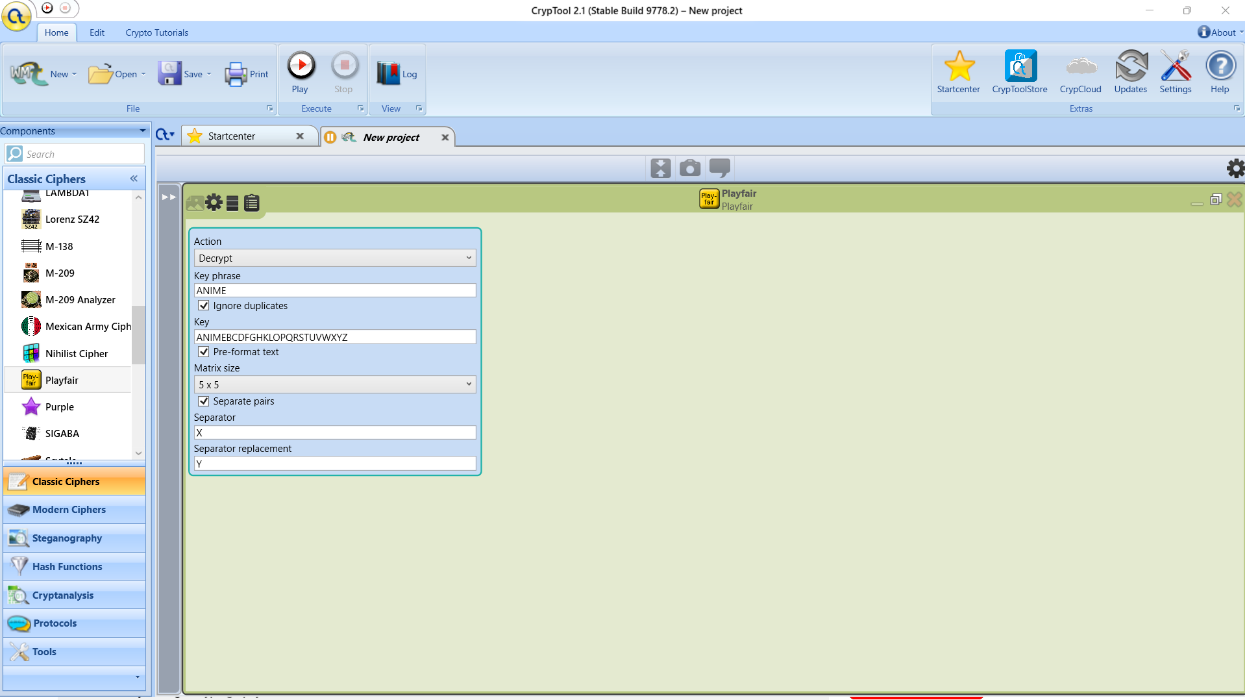
Encryption Process:

****

Decryption Process:

****

Key Phrase:

****

**ANALYSIS:**

* 1. Provides robust encryption with letter pairings.
  2. Effective against simple frequency analysis.
  3. Vulnerable to known-plaintext attacks.
  4. Fixed 5x5 matrix aids pattern recognition.
  5. Suitable for moderate encryption needs.

**CONCLUSION:**

The implementation and analysis of the Playfair Cipher using CrypTool illustrate its effectiveness as a classical encryption method. While providing more security than simpler ciphers like the Caesar Cipher, it has vulnerabilities that limit its application in modern cryptography.

**LAB: 03**

**TITLE:** Implementation and Analysis of the Hill Cipher using CrypTool

**OBJECTIVES:**

1. To Implement the Hill Cipher
2. To Analyze Encryption and Decryption Processes

**THEORY:**

Hill cipher is a polygraphic substitution cipher based on linear algebra. Each letter is represented by a number modulo 26. Often the simple scheme A = 0, B = 1, …, Z = 25 is used, but this is not an essential feature of the cipher. To encrypt a message, each block of n letters (considered as an n-component vector) is multiplied by an invertible n × n matrix, against modulus 26. To decrypt the message, each block is multiplied by the inverse of the matrix used for encryption.  
The matrix used for encryption is the cipher key, and it should be chosen randomly from the set of invertible n × n matrices (modulo 26).

Encryption:

* Generate a key matrix and encode plaintext into vector form.
* Perform matrix multiplication to obtain ciphertext.

C = KP mod 26

Decryption:

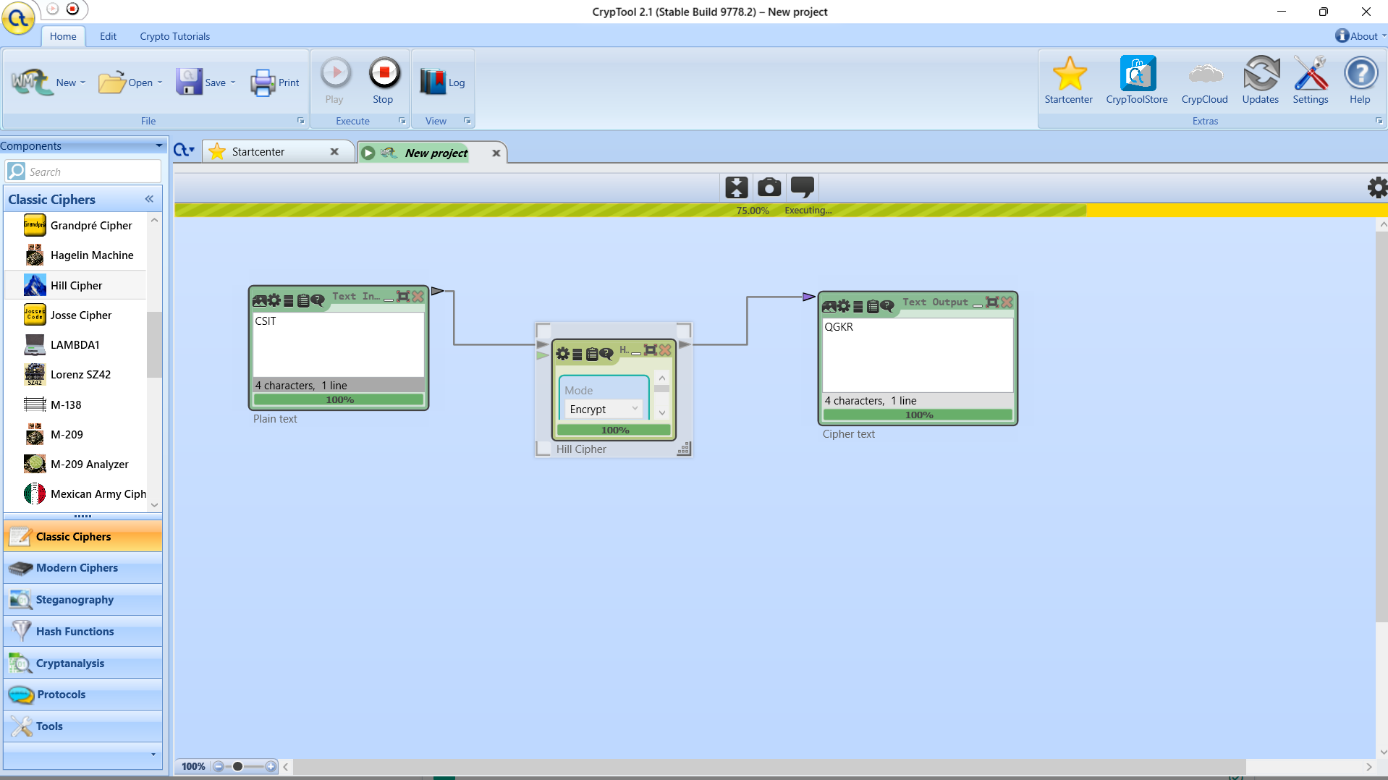
* Utilize the inverse of the key matrix to decrypt ciphertext back into plaintext.

P = K-1C mod 26 [where, K-1 = (detK) -1Adj(K)]

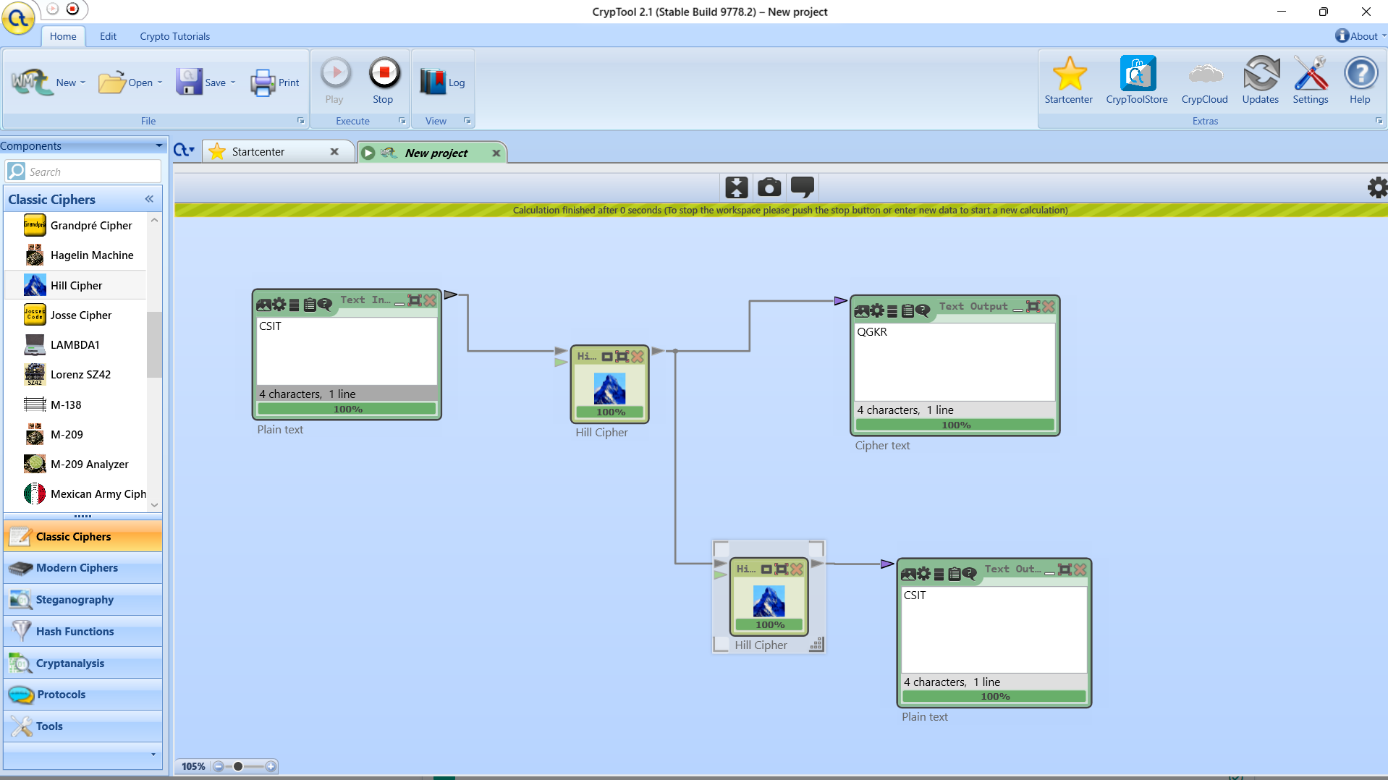
Where, K is n X n key matrix, K-1 is its inverse matrix, P is plaintext matrix and C is ciphertext matrix

**IMPLEMENTATION:**

Encryption Process:

****

Decryption Process:



**ANALYSIS:**

1. Demonstrates accurate encryption and decryption.
2. Stronger encryption due to matrix operations.
3. Vulnerable if the key matrix is compromised.
4. Suitable for applications needing higher encryption strength.
5. Enhances understanding of matrix-based cryptography.

**CONCLUSION:**

The Hill Cipher, analyzed using CrypTool, demonstrates its effectiveness in modern cryptography by using matrix operations to enhance security. This approach overcomes limitations seen in traditional ciphers, making it suitable for applications requiring higher levels of encryption strength and data security.

**LAB: 04**

**TITLE:** Implementation and Analysis of the Rail Fence Cipher using CrypTool

**OBJECTIVES:**

1. To Implement the Rail Fence Cipher
2. To Analyze Encryption and Decryption Processes

**THEORY:**

The Rail Fence Cipher is a form of transposition cipher. It depends on the matrix whose dimension is defined by the length of plaintext (or ciphertext) and a number of rails.

No. of rows = no. of rails (key)

No. of columns = length of plaintext / ciphertext

Encryption:

* The plaintext is written in diagonally, from top to bottom and after reaching rails, it goes bottom to top and so on.
* The ciphertext is written in row-wise from the matrix.

Decryption:

* Construct the rail matrix according to length of ciphertext (columns) and number of rails (rows). Once we've got the matrix we can figure-out the spots where texts should be placed (using the same way of moving diagonally up and down alternatively).
* Then, we fill the cipher-text row wise. After filling it, we traverse the matrix in diagonal manner to obtain the original text.

Example:

Suppose we have to compute the ciphertext for the Rail Fence cipher with rails = 3.

let Plaintext= ABRA KA DABRA

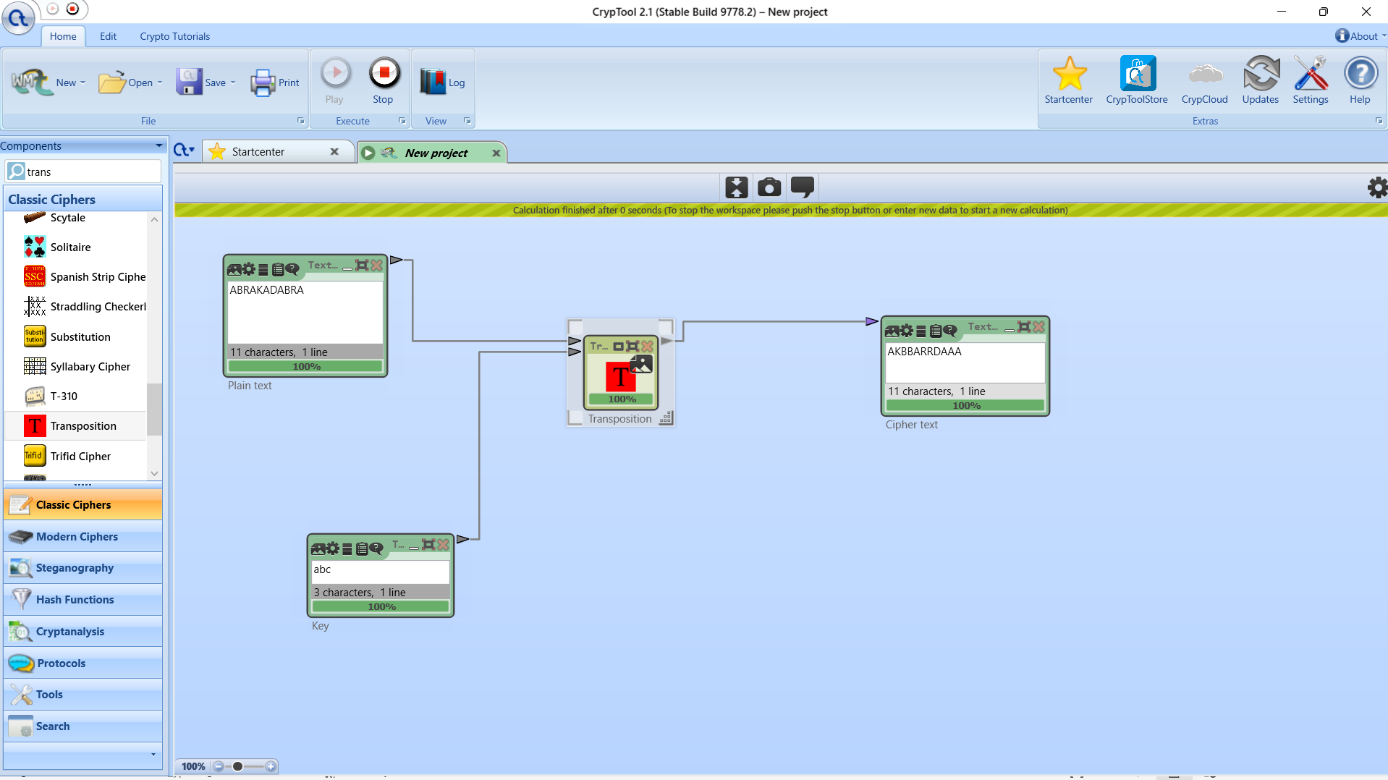
No. of rails = 3

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A |  |  |  | K |  |  |  | B |  |  |
|  | B |  | A |  | A |  | A |  | R |  |
|  |  | R |  |  |  | D |  |  |  | A |

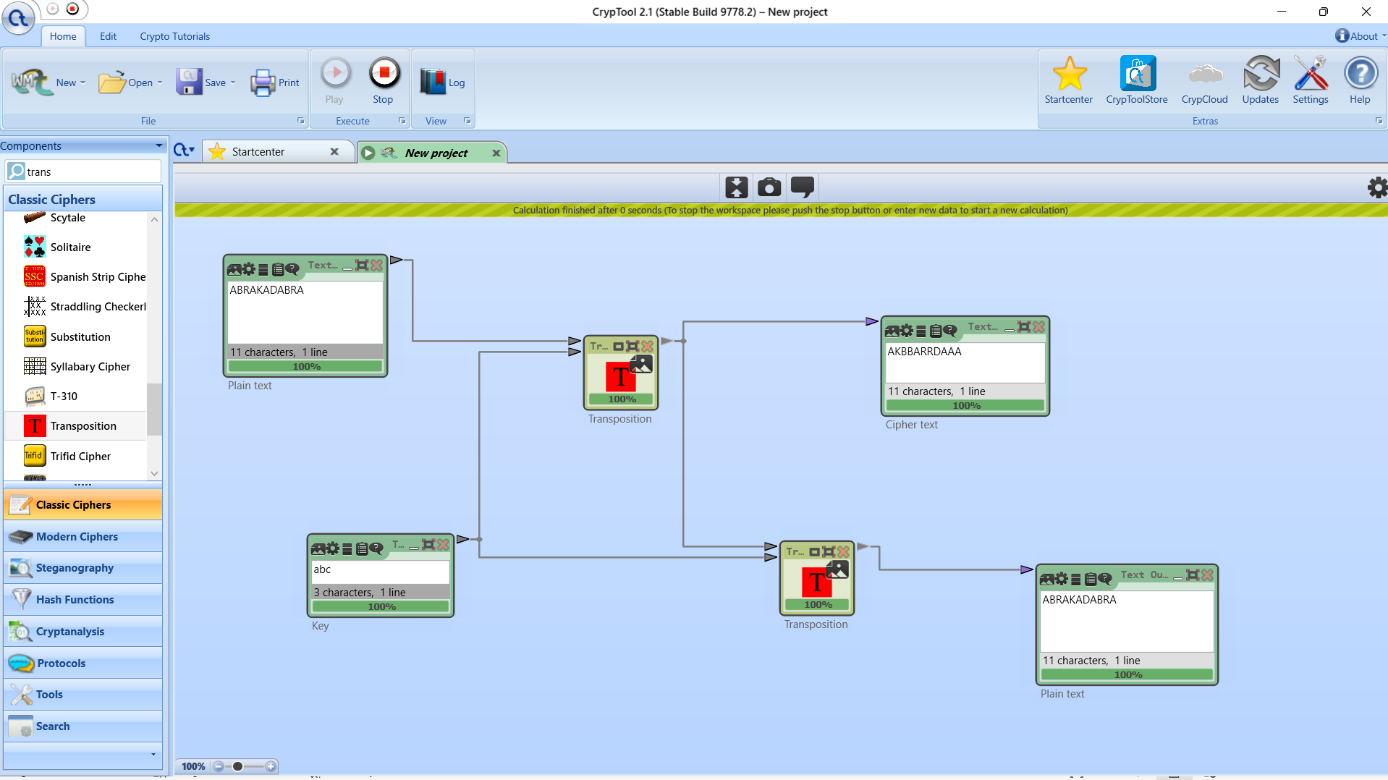
Therefore, Ciphertext = AKBBAAARRDA

**IMPLEMENTATION:**

Encryption Process:



Decryption Process:



**ANALYSIS:**

1. Effectively scrambles plaintext but is relatively simple.
2. Vulnerable to pattern recognition and frequency analysis.
3. Provides a moderate level of security suitable for specific applications.

**CONCLUSION:**

The Rail Fence Cipher is a simple transposition cipher that rearranges plaintext characters. It is useful for educational purposes but lacks substantial security. For protecting sensitive information, more advanced encryption methods are recommended.

**LAB: 05**

**TITLE:** Implementation and Analysis of the Data Encryption Standard (DES) using CrypTool

**OBJECTIVES:**

1. To Implement the DES
2. To Analyze Encryption and Decryption Processes

**THEORY:**

The Data Encryption Standard (DES) is a symmetric-key block cipher that encrypts 64-bit blocks of data using a 56-bit key. The key is originally 64 bits with 8 parity bits that are discarded during key scheduling.

**Key Scheduling:**

* KeyGeneration**:** Creates 16 subkeys from the 56-bit key using permutations and shifts.

**Encryption:**

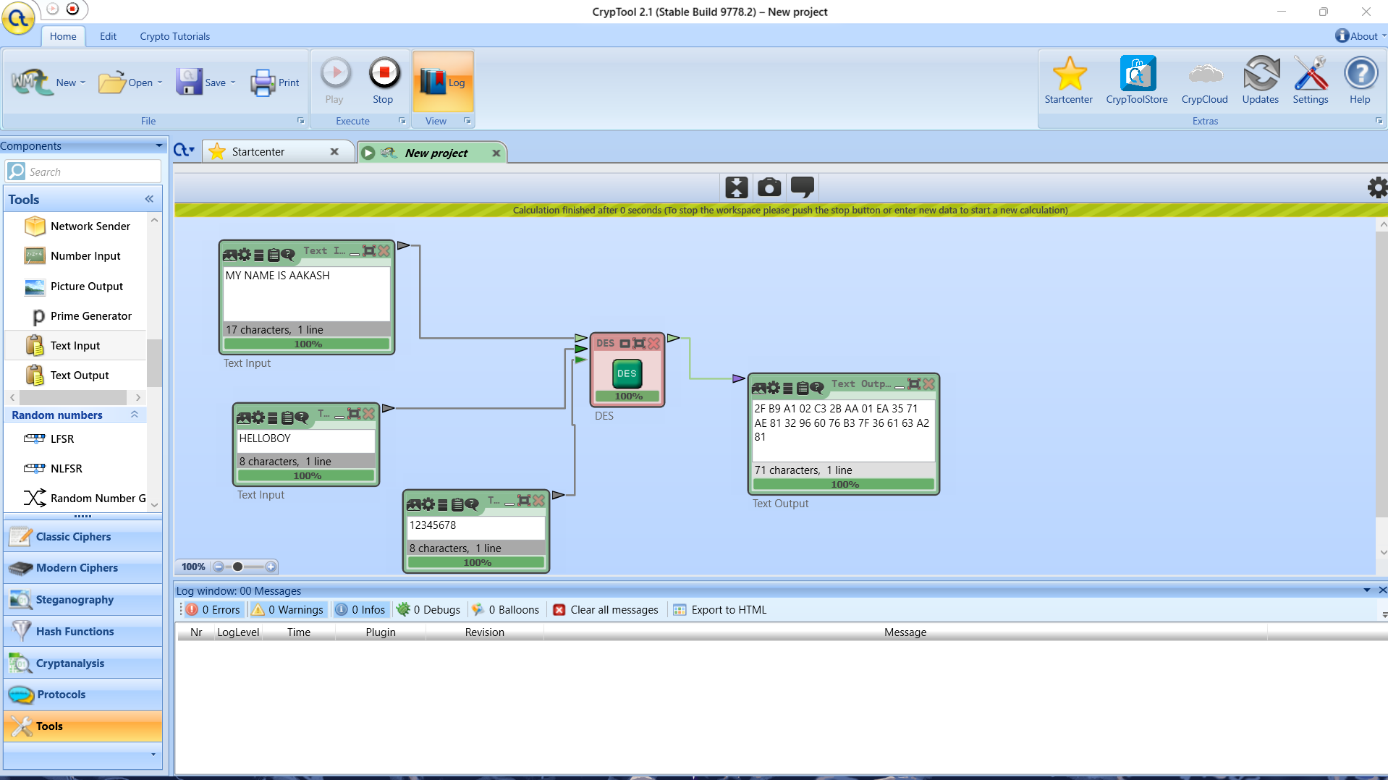
1. Initial Permutation (IP): Rearrange plaintext bits.
2. Key Scheduling: Generate 16 subkeys from the original key.
3. Round Operations (16 rounds):
   * Expand the right half of the block.
   * XOR with the round subkey.
   * Apply S-boxes for substitution.
   * Permute the result.
   * XOR with the left half and swap halves.
4. Final Permutation (FP): Apply FP to get ciphertext.

**Decryption:**

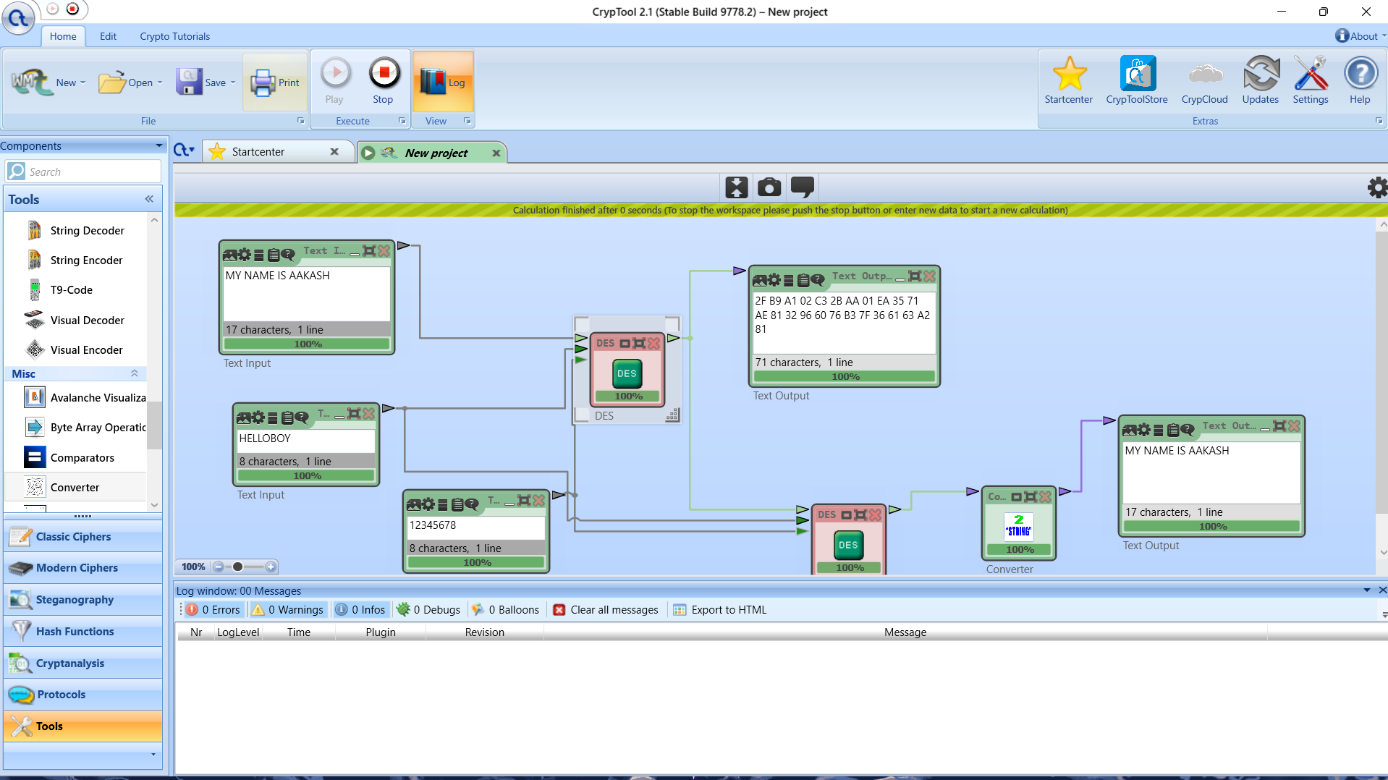
1. Initial Permutation (IP): Apply IP to ciphertext.
2. Round Operations (16 rounds in reverse):
   * Expand the right half.
   * XOR with the round subkey (reversed order).
   * Apply S-boxes for substitution.
   * Permute the result.
   * XOR with the left half and swap halves.
3. Final Permutation (FP): Apply FP to recover plaintext.

**IMPLEMENTATION:**

Encryption Process:

****

Decryption Process:

****

**ANALYSIS:**

1. DES provides robust encryption for its time but is vulnerable to brute-force attacks due to its 56-bit key length.
2. Suitable for historical contexts but less effective with modern computational power.

**CONCLUSION:**

DES was a significant advancement in cryptography with its block cipher approach and multiple rounds. However, it is no longer secure due to its short key length and vulnerability to attacks. Transitioning to more secure standards like AES is essential.

**LAB: 06**

**TITLE:** Implementation and Analysis of the Advance Encryption Standard (AES) using CrypTool

**OBJECTIVES:**

1. To Implement the AES
2. To Analyze Encryption and Decryption Processes

**THEORY:**

AES is a block cipher which operates on block size of 128 bits for both encrypting as well as decrypting. Three key lengths are available: 128, 192, or 256 bits (16, 24, or 32 bytes)

The number of rounds performed by the algorithm strictly depends on the size of key. Key Size (in bits) 128 192 256 and corresponding round 10 12 14. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key. Each round consists of four functions:

1. Sub Bytes

2. Shift Rows

3. Mix Columns, not applied in last round.

4. Add Round Key

**Encryption**:

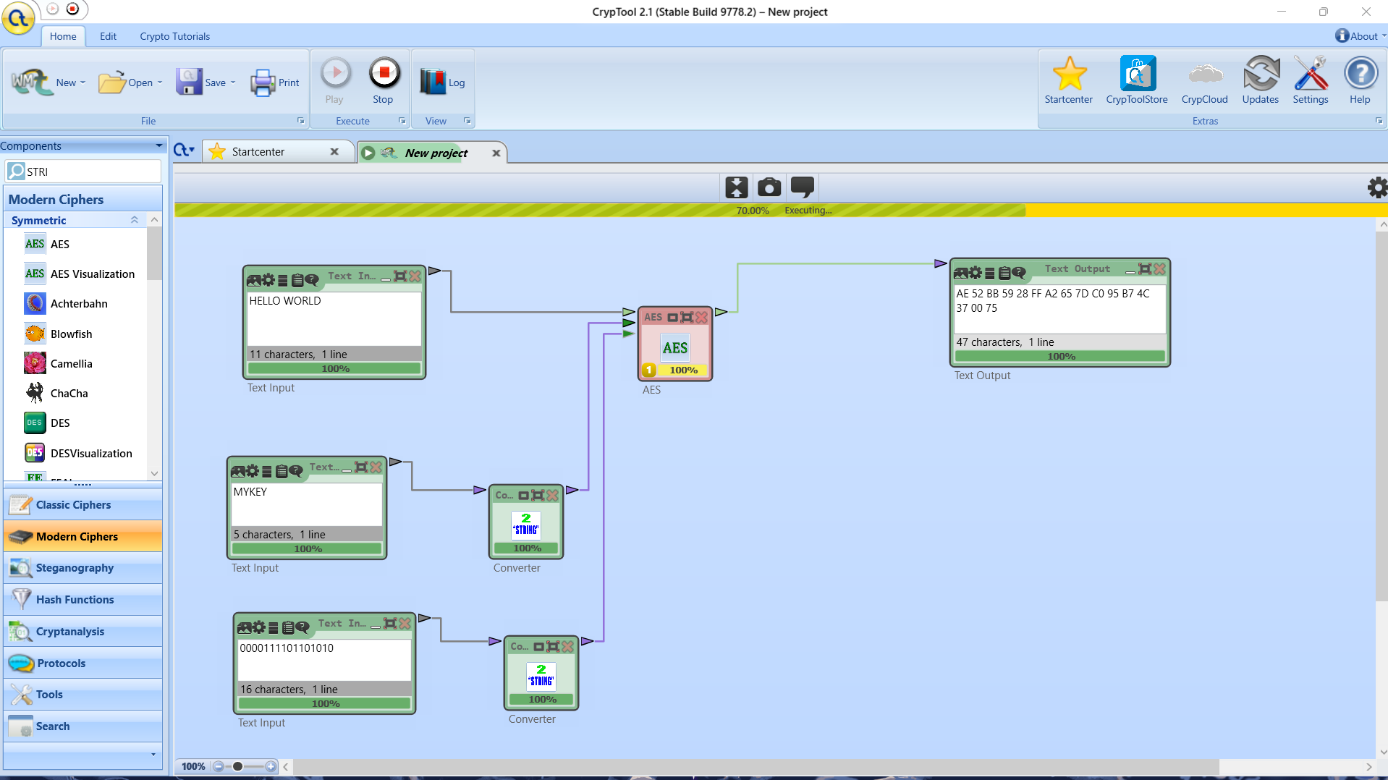
1. Apply the initial round key.
2. Perform multiple rounds of transformation (Sub Bytes, Shift Rows, Mix Columns, Add Round Key).
3. Apply the final round transformations.

**Decryption**:

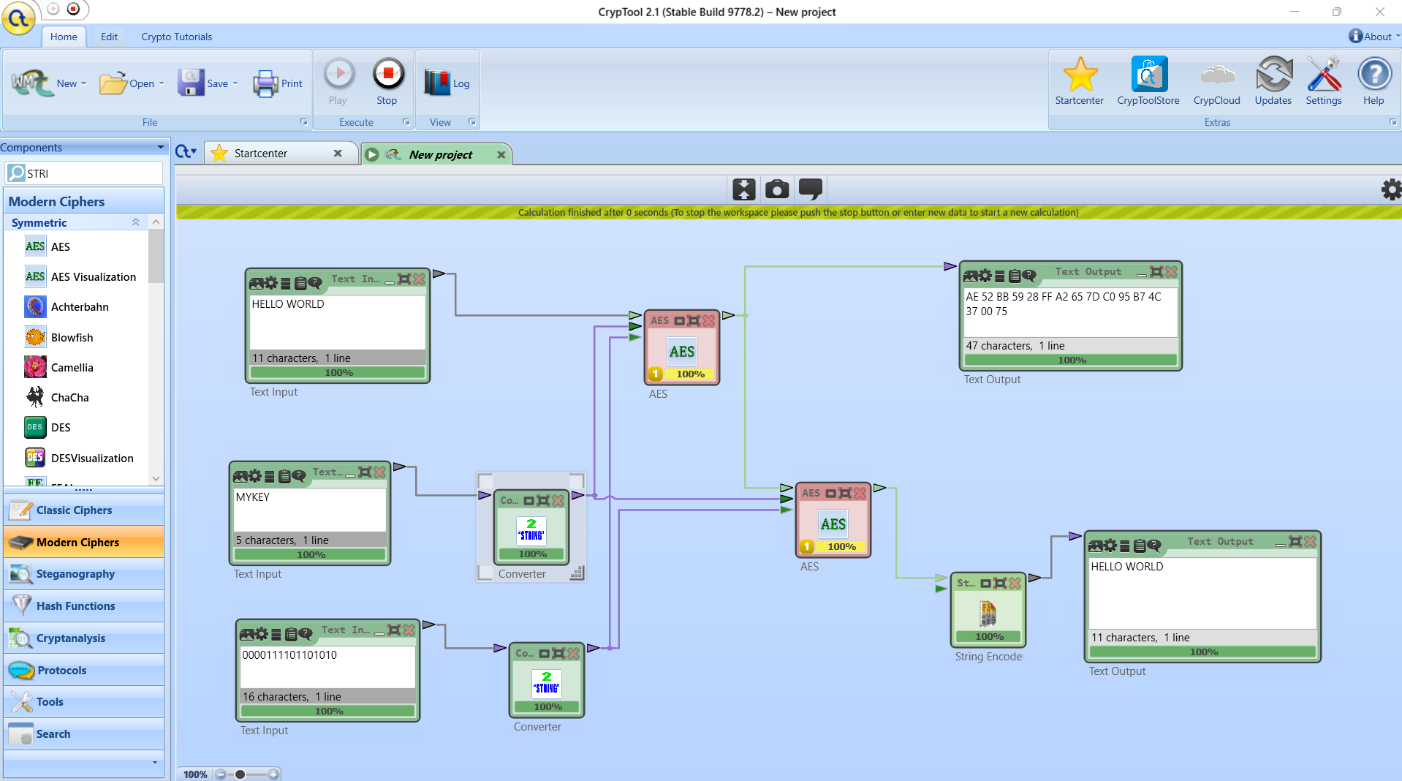
1. Perform inverse transformations of encryption rounds.
2. Apply the final inverse round key operation.

**IMPLEMENTATION:**

Encryption Process:

****

Decryption Process:



**ANALYSIS:**

1. AES provides high security with robust encryption capabilities.
2. Suitable for modern applications requiring strong encryption.
3. Efficient and widely used in various security applications.

**CONCLUSION:**

AES is a highly secure and efficient encryption standard that offers strong protection with variable key lengths and multiple rounds. It is well-suited for modern encryption needs and remains a top choice for securing sensitive data.

**LAB: 07**

**TITLE:** Implementation and Analysis of the Rivest Simmer Algorithm (RSA) using CrypTool

**OBJECTIVES:**

1. To Implement the RSA
2. To Analyze Encryption and Decryption Processes

**THEORY:**

RSA algorithm is public key cryptography i.e. it works on two different keys i.e. public key and private key. The public key can be known to everyone and is used for encrypting message. Message encrypted with the public key can only be decrypted using the private key.

**Algorithm RSA key generation**:

1. Choose two distinct large prime numbers p and q.

2. Compute n = pq, n is used as modulus for both public and private keys.

3. Compute the totient: (n) = (p - 1)(q - 1).

4. Choose an integer e such that 1 < e < (n) and e and (n) are co-prime.

5. Compute d to satisfy ed = 1 (mod (n)).

6. Public key is {e, n}.

7. Private key is (d, n).

**Encryption**:

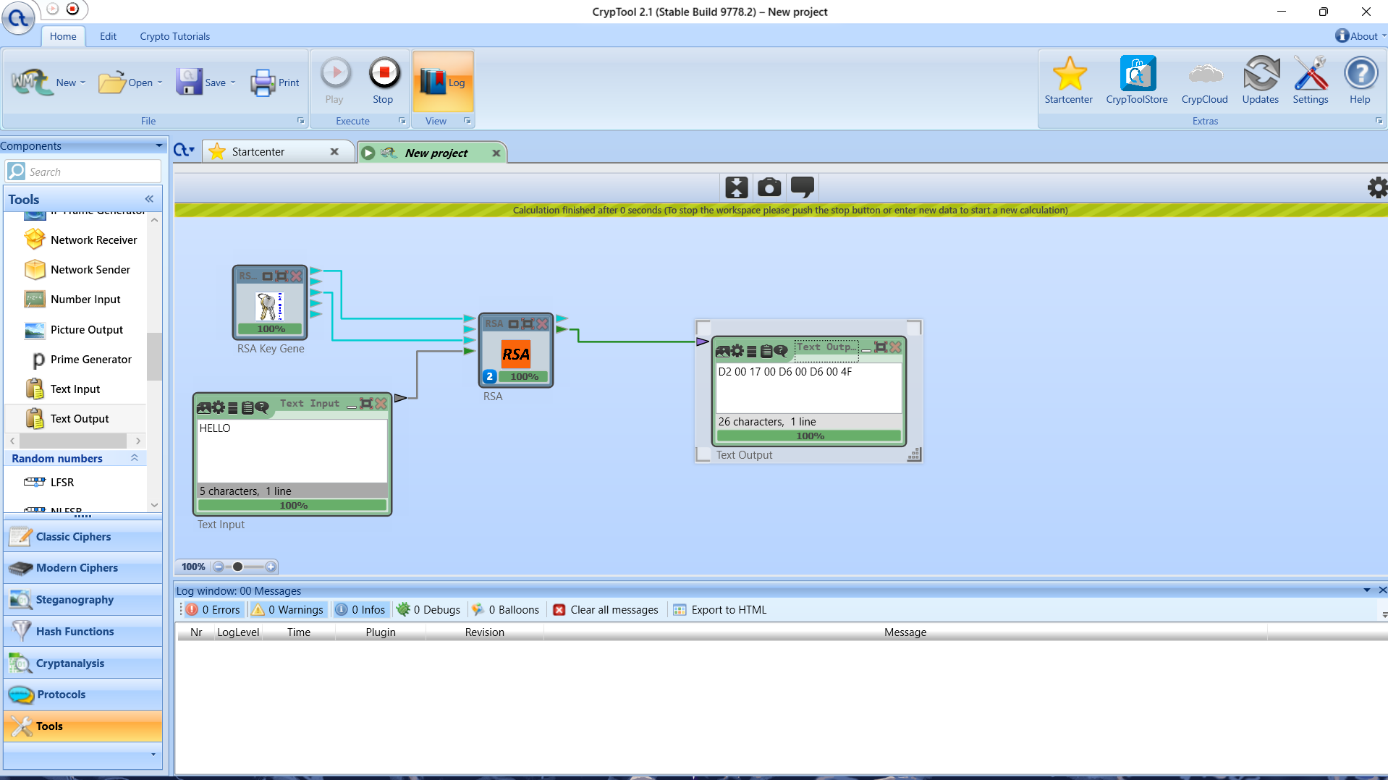
c = me mod n

**Decryption**:

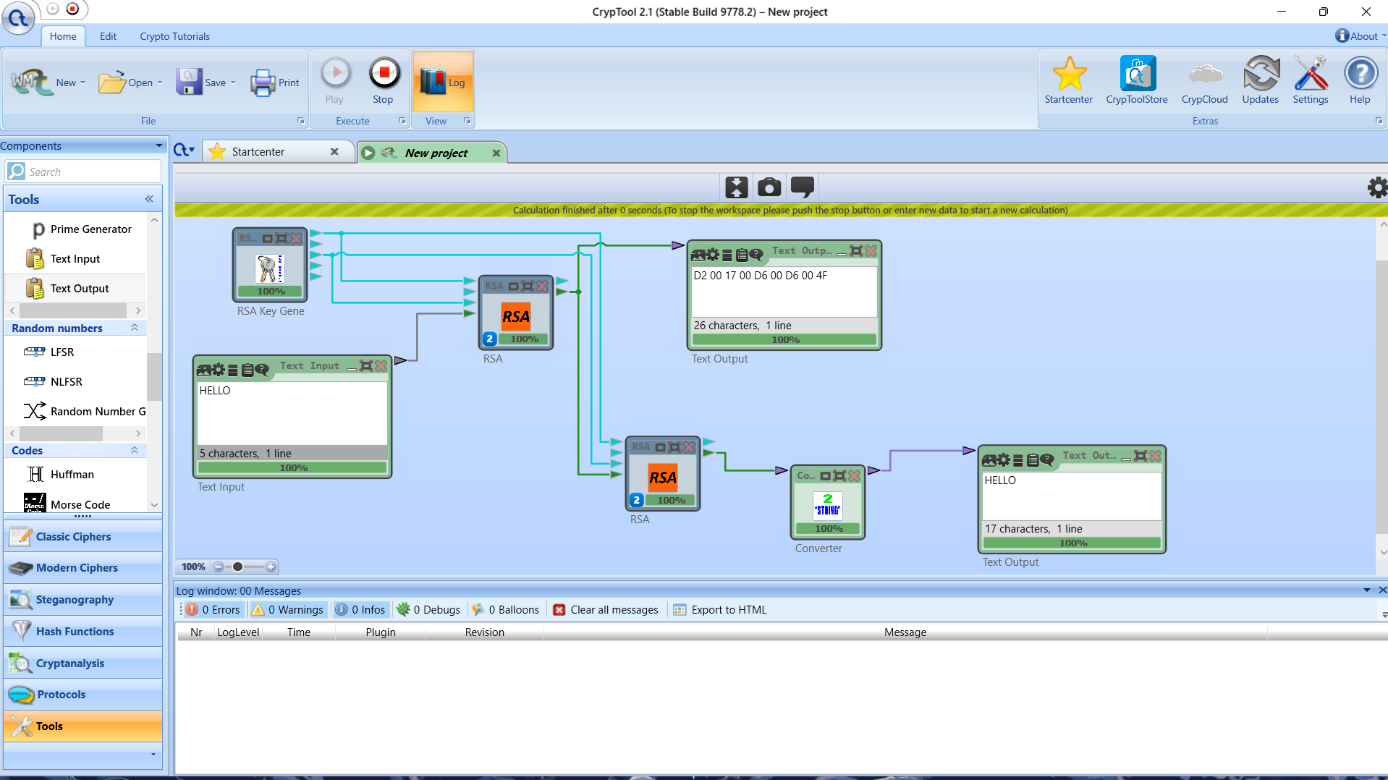
m = cd mod n

**IMPLEMENTATION:**

Encryption Process:

****

Decryption Process:

****

**ANALYSIS:**

1. RSA provides strong security through its use of large key sizes.
2. It is computationally intensive compared to symmetric-key algorithms.
3. Suitable for secure data transmission and digital signatures.

**CONCLUSION:**

RSA is a foundational public key encryption algorithm that provides robust security through asymmetric encryption. While effective, it is important to stay informed about potential future challenges, such as advancements in quantum computing.