Emerging Blockchain Models for Digital Currencies

Name: Shaik Aarif

Regd No: 2200033144

Exp 1: Implementation of the Custom Symmetric Key Encryption Algorithm.

Description of the Algorithm:

1. **Key Generation (GenKey)**:
    - The key is calculated by summing the ASCII values of all characters in the plaintext string.
    - This serves as the basis for encryption and decryption.
2. **Simple XOR Cipher (Encrypt1 and Decrypt1)**:
    - Each character of the plaintext is XOR-ed with the key to produce the ciphertext.
    - Decryption reverses this process using the same key.
3. **Enhanced XOR Cipher with Multiplicative Factor (Encrypt2 and Decrypt2)**:
    - Each character of the plaintext is XOR-ed with the product of the key and a position-based multiplier (j).
    - If the resulting value exceeds the Unicode limit (1114111), it is wrapped around using modulo operation.
    - This adds an additional layer of complexity, making it harder to decode without the exact key and logic.
4. **Encryption Process**:
    - Encrypt1 applies a simple XOR operation.
    - Encrypt2 uses a position-dependent multiplier (j) to modify the XOR operation for each character.
5. **Decryption Process**:
    - The decryption process mirrors the encryption logic, ensuring that the ciphertext is reverted to plaintext using the same key and algorithm.
6. **Output**:
    - The program generates two versions of ciphertext: one with a normal XOR logic (Encrypt1) and another with a custom XOR logic (Encrypt2).

- It then decrypts both ciphertexts back to plaintext to verify the correctness of the algorithms.

7. **Boundary Handling in Encrypt2 and Decrypt2**:
   - If the calculated character value exceeds the Unicode limit, it wraps around using % 1114112.
   - This ensures compatibility with valid Unicode character ranges.
8. **Testing**:
   - The program prints intermediate values for debugging (value during encryption and decryption in Encrypt2).
   - It displays both the encrypted and decrypted outputs for comparison.

Code:

```
def GenKey(plainText):
    key = 0
    for i in plainText:
        key += ord(i)
    return key

def Encrypt1(plainText, key):
    cipherText = ""
    for i in plainText:
        cipherText += chr(ord(i) ^ key)
    return cipherText

def Decrypt1(cipherText, key):
    plainText = ""
    for i in cipherText:
        plainText += chr(ord(i) ^ key)
    return plainText

def Encrypt2(plainText, key):
    cipherText = ""
    j=1
    for i in plainText:
        value = ord(i) ^ key * j
        if value > 1114111:
            value = value % 1114112
        print(value , end=' ')
        cipherText += chr(value)
        j+=1
    return cipherText

def Decrypt2(cipherText, key):
    plainText = ""
    j=1
    for i in cipherText:
        value = ord(i) ^ key * j
        if value > 1114111:
            value = value % 1114112
        print(value , end=' ')
        plainText += chr(value)
        j+=1
    return plainText

plainText = "Emerging BlockChain"
key=GenKey(plainText)
cipherText = Encrypt1(plainText, key)
```

```
46. DecryptText = Decrypt1(cipherText, key)
47.
48. cipherText1 = Encrypt2(plainText, key)
49. DecryptText1 = Decrypt2(cipherText1, key)
50.
51. print("Key: ", key)
52. print("\n")
53. print("Normal XOR Cipher Text: ", cipherText)
54. print("Normal XOR Decrypt Text: ", DecryptText)
55.
56. print("\n")
57.
58. print("Custom XOR Logic Cipher Text: ", cipherText1)
59. print("Custom XOR Logic Decrypt Text: ", DecryptText1)
```

Execution: (Screenshot)