Data: 29/04/2024

Nome: Victor Furtado Estrada

RA: 22006647

1° Prova de Ciber Segurança

1- Questão:

```
Questão: def decifra_cesar(texto_cifrado, chave):
    texto_decifrado = ""
    for c in texto_cifrado:
        if c.isalpha():
            ascii_offset = ord('a') if c.islower() else ord('A')
            c_index = ord(c) - ascii_offset
            novo_c_index = (c_index - chave) % 26
            novo_c = chr(novo_c_index + ascii_offset)
            texto_decifrado += novo_c
        else:
            texto_decifrado += c
    return texto_decifrado

texto_cifrado = "Kjqne fvzjqj vzj ywfsxkjwj t vzjxfgj j fuwjsij t vzj
jsxnsf.Htwf Htwfqnsf."
for chave in range(26):
    print(f"Chave={chave}, Texto={decifra_cesar(texto_cifrado, chave)}")
```

Conforme foi rodado o código a palavra decifrada veio na Chave 5 com o texto: Feliz aquele que transfere o que sabe e aprende o que ensina. Cora Coralina.
Este código define uma função chamada decifra_cesar que aceita um texto_cifrado e uma chave como entrada. Esta função percorre cada caractere no texto_cifrado. Se o caractere é uma letra, ele calcula o novo caractere deslocado e adiciona ao texto_decifrado. Se o caractere não é uma letra, ele simplesmente adiciona o caractere ao texto_decifrado. Depois o código então define uma string como o texto_cifrado que contém o texto cifrado e então percorre todas as possíveis chaves de 0 a 25 e decifra o texto_cifrado usando cada chave e imprime a chave e o texto decifrado.

2- Questão:

```
3- cifra_feistel(texto_binario, rodadas):
4-     def funcao_F(bloco, chave):
5-         return ''.join(str(int(b) & int(k)) for b, k in zip(bloco,
   chave))
6-
7-     def deslocamento_circular_esquerda(bloco):
8-         return bloco[1:] + bloco[0]
9-
10-    L, R = texto_binario[:len(texto_binario)//2],
   texto_binario[len(texto_binario)//2:]
11-    for _ in range(rodadas):
12-        L, R = R, ''.join(str(int(l) ^ int(r)) for l, r in zip(L,
   funcao_F(R, L)))
13-        L = deslocamento_circular_esquerda(L)
14-    return L + R
15-
16-print(cifra_feistel("01101100", 2))
```

O texto 01101100 é cifrado conforme este número gerado pelo código 0100100.

1. A função cifra_feistel aceita um texto_binario e um número de rodadas como entrada.
2. A função funcao_F é definida dentro de cifra_feistel. Ela aceita um bloco e uma chave como entrada, e retorna uma string que é o resultado da operação AND bit a bit entre o bloco e a chave.
3. A função deslocamento_circular_esquerda também é definida dentro de cifra_feistel. Ela aceita um bloco como entrada e retorna uma nova string que é o bloco deslocado para a esquerda (o primeiro caractere é movido para o final).
4. O texto_binario é dividido em duas metades, L e R.
5. O código então entra em um loop que executa um número de rodadas vezes. Em cada rodada, L e R são transformados e trocados. A transformação envolve a aplicação da funcao_F a R e L, e então fazendo a operação XOR bit a bit entre L e o resultado da funcao_F. L é então deslocado para a esquerda.
6. Após todas as rodadas terem sido executadas, a função retorna a concatenação de L e R como o texto binário criptografado.
7. Finalmente, o código chama cifra_feistel com o texto_binario "01101100" e 2 rodadas, e imprime o resultado.
   Portanto, a saída deste código será um texto binário criptografado.

3- Questão:

```
s1 = [
    [14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7],
    [0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8],
    [4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 9, 3, 10, 5, 0],
    [14, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 3, 10, 0, 6, 13]
]

entrada = "011010"

# Dividir a entrada em blocos de 3 bits
blocos = [entrada[i:i+3] for i in range(0, len(entrada), 3)]

# Realizar a substituição para cada bloco
saida = []
for bloco in blocos:
    # Converter o bloco de binário para decimal
    linha = int(bloco[0], 2)
    coluna = int(bloco[1:], 2)

    # Usar os valores decimais para indexar a caixa S1
    substituicao = s1[linha][coluna]

    # Converter a substituição de volta para binário e adicionar à saída
    saida.append(bin(substituicao)[2:].zfill(3))

# Juntar os blocos de saída para formar a saída final
saida_final = "".join(saida)

print("Saída:", saida_final)
```

A saída deste código é 0011101. Este é o código explicado passo a passo:

1. Define a caixa de substituição s1 como uma lista de listas, onde cada lista interna representa uma linha da matriz.
2. Define uma entrada binária.
3. Divide a entrada em blocos de 3 bits. Ele faz isso usando uma compreensão de lista para criar uma nova lista blocos, onde cada elemento é uma substring de 3 caracteres da entrada.
4. Inicializa uma lista vazia saida para armazenar os resultados da substituição.
5. Para cada bloco na lista blocos, ele faz o seguinte:
    - Converte o primeiro bit do bloco para um número inteiro para obter o índice da linha na caixa S1.

- Converte o restante do bloco para um número inteiro para obter o índice da coluna na caixa S1.
- Usa os índices da linha e da coluna para obter um valor de substituição da caixa S1.
- Converte o valor de substituição de volta para uma string binária, preenche com zeros à esquerda para garantir que seja uma string de 3 caracteres e adiciona à lista saida.

6. Junta os blocos de saída para formar a saída final e imprime a saída.

Portanto, a saída deste código será uma string binária que é o resultado da aplicação da caixa S1 à entrada.