Universidad Nacional Autónoma de México

Facultad de Ciencias

Criptografía y Seguridad

Punto extra

Fecha de entrega: 27/09/2022

Equipo:

Criptonianos

Acosta Arzate Rubén - 317205776

Bernal Marquez Erick - 317042522

Deloya Andrade Ana Valeria - 317277582

Marco Antonio Rivera Silva - 318183583

1. Usando la clave SILVIOR genera un cuadro de Playfair y cifra la frase: "quedamos los que puedan sonreir"

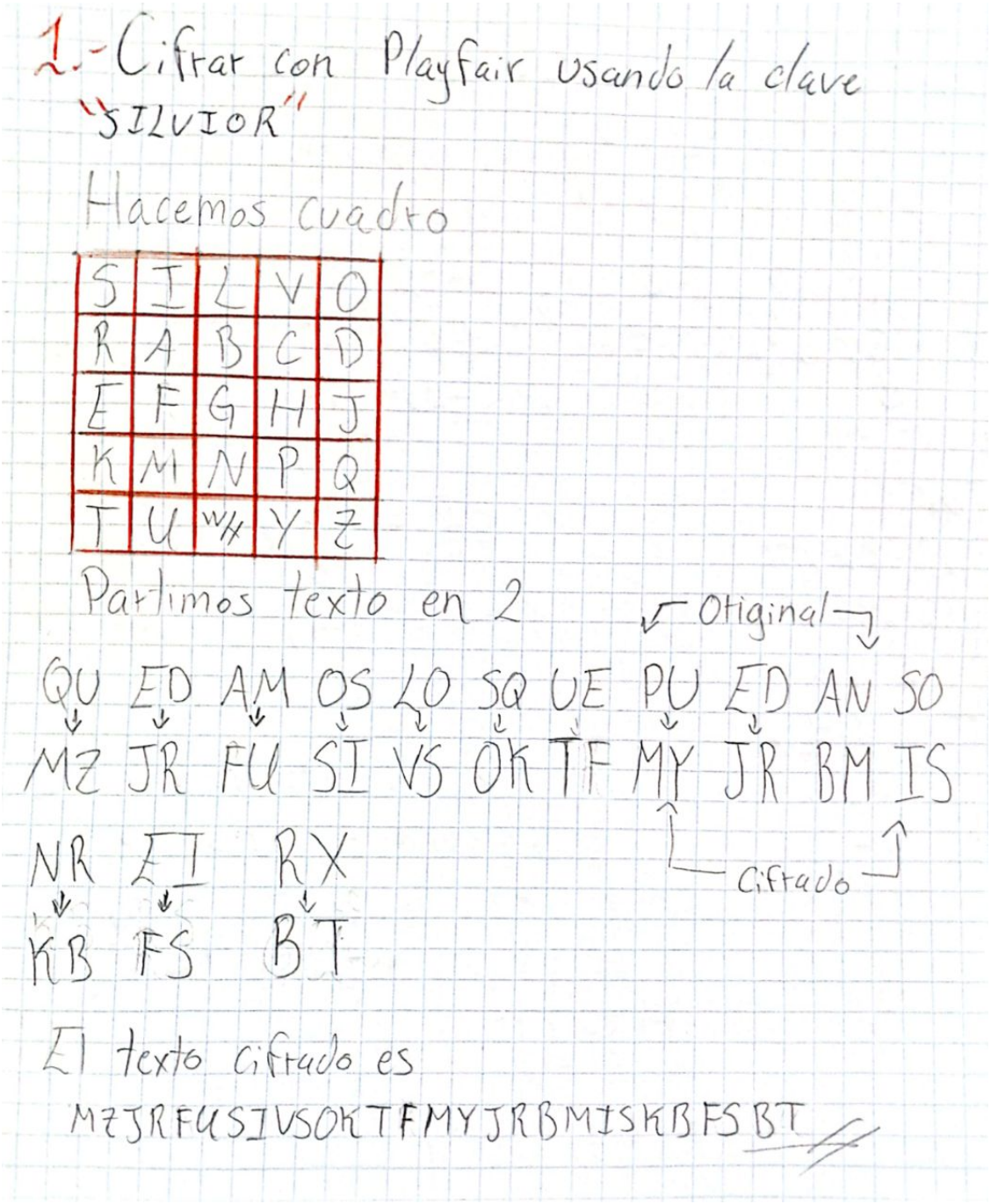

Figura 1: Cifrado Playfair

2. Crea una matriz de Hill y cifra la palabra "viernes". Da la inversa de esta matriz sin usar la fórmula vista en clase, pero sí el procedimiento

Tomamos la siguiente correspondencia de letras vista en clase.

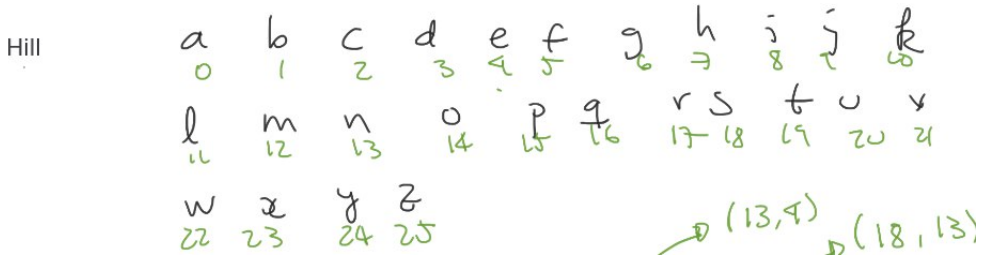

Figura 2: Correspondencia de letras a numeros

Ciframos

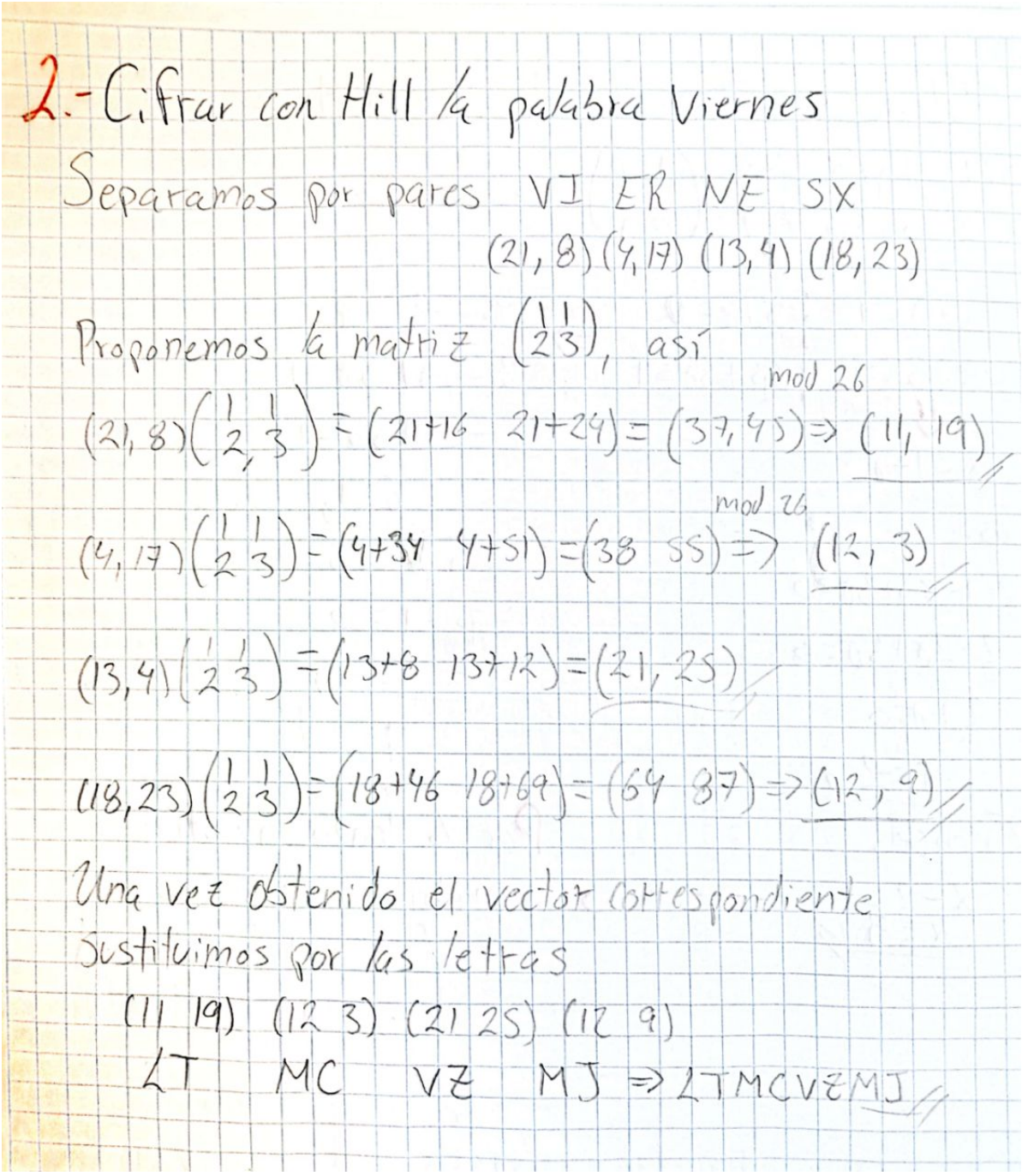

Figura 3: Cifrado Playfair

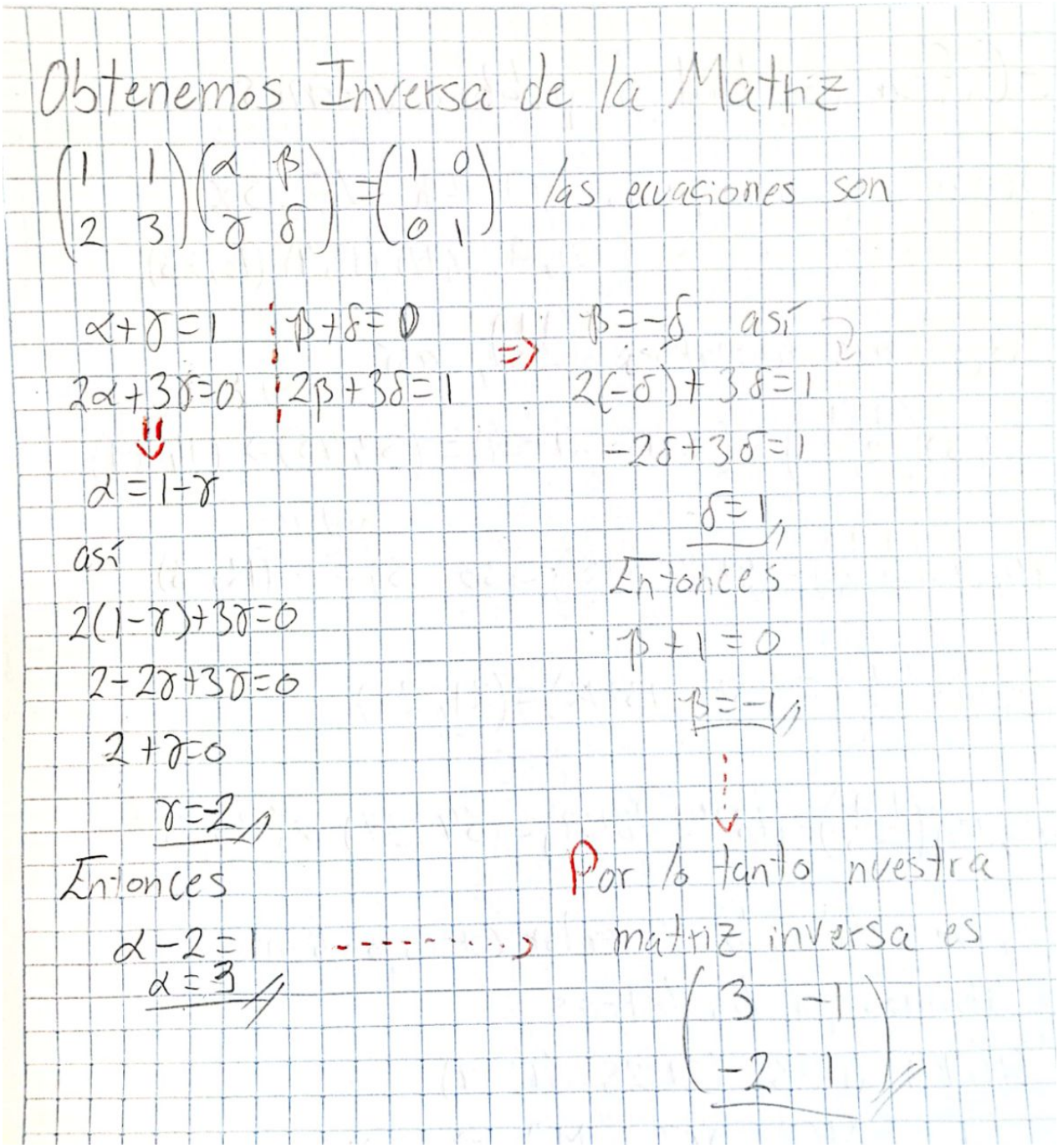

Figura 4: Cifrado Playfair

3. Pídele a *ChatGPT* que genere el código para cifrar con Playfair y Hill y córrelo. ¿Da los mismos resultados? Si no ¿Puedes mejorar la instrucción para que lo haga bien?

El código generado por ChatGPT para el cifrado de Playfair es el siguiente.

```python
def create_playfair_matrix(key):
    # Crear la matriz Playfair a partir de la clave
    alphabet = "ABCDEFGHIKLMNOPQRSTUVWXYZ"  # Excluimos la letra 'J'
    key = key.upper().replace("J", "I")  # Convertir a mayúsculas y reemplazar 'J' por 'I'
    key = "".join(dict.fromkeys(key))  # Eliminar letras duplicadas
    matrix = [[None] * 5 for _ in range(5)]
    key_index = 0
    alphabet_index = 0

    for row in range(5):
        for col in range(5):
            if key_index < len(key):
                matrix[row][col] = key[key_index]
                key_index += 1
            else:
                while alphabet[alphabet_index] in key:
                    alphabet_index += 1
                matrix[row][col] = alphabet[alphabet_index]
                alphabet_index += 1

    return matrix
```

Figura 5: Cifrado Playfair parte 1

```python
def find_char(matrix, char):
    # Encontrar las coordenadas de un carácter en la matriz Playfair
    for row in range(5):
        for col in range(5):
            if matrix[row][col] == char:
                return (row, col)
```

Figura 6: Cifrado Playfair parte 2

```python
def playfair_encrypt(plaintext, key):
    # Cifrar un mensaje utilizando el cifrado Playfair
    matrix = create_playfair_matrix(key)
    plaintext = plaintext.upper().replace("J", "I")
    plaintext = "".join(filter(str.isalpha, plaintext))  # Eliminar caracteres no alfabéticos
    if len(plaintext) % 2 != 0:
        plaintext += "X"  # Agregar 'X' si la longitud es impar
    ciphertext = []
    for i in range(0, len(plaintext), 2):
        char1, char2 = plaintext[i], plaintext[i + 1]
        row1, col1 = find_char(matrix, char1)
        row2, col2 = find_char(matrix, char2)
        if row1 == row2:
            ciphertext.append(matrix[row1][(col1 + 1) % 5])
            ciphertext.append(matrix[row2][(col2 + 1) % 5])
        elif col1 == col2:
            ciphertext.append(matrix[(row1 + 1) % 5][col1])
            ciphertext.append(matrix[(row2 + 1) % 5][col2])
        else:
            ciphertext.append(matrix[row1][col2])
            ciphertext.append(matrix[row2][col1])
    return "".join(ciphertext)
```

Figura 7: Cifrado Playfair parte 3

```python
# Ejemplo de uso
key = "SILVIOR"
plaintext = "quedamos los que puedan sonreir"
print(create_playfair_matrix(key))
ciphertext = playfair_encrypt(plaintext, key)
print("Mensaje cifrado:", ciphertext)
```

```
[['S', 'I', 'L', 'V', 'O'], ['R', 'A', 'B', 'C', 'D'], ['E', 'F', 'G', 'H', 'K'], ['M', 'N', 'P', 'Q', 'T'], ['U', 'W', 'X', 'Y', 'Z']]
Mensaje cifrado: MYKRRNSIVSVMSMMXKRFWISMAFSBU
```

Figura 8: Resultados de cifrado Playfair

La primer salida muestra las filas de la matriz de Playfair, notemos que los primeros dos corresponden a nuestra implementación, sin embargo el tercer renglon *[E, F, G, H, K]* difiere en nuestra implementación respecto a la ultima letra, *ChatGPT* quita la letra J y en el último renglón deja las letras W y X en diferentes casillas, mientras que nosotros ponemos ambas letras en una sola.

El cifrado es correcto pero al tener una ligera variación el resultado puede ser diferente, de hecho algunas letras cifradas de ChatGPT sí corresponen a nuestro cifrado, como las 4 últimas.

Si queremos que *ChatGPT* nos arroje exactamente el mismo resultado que nosotros tenemos habrá que especificar “No quites la J del alfabeto” y “Pon las letras W y X en una sola casilla”

El código generado por ChatGPT para el cifrado de Hill es el siguiente.

```python
import numpy as np

def prepare_plaintext(plaintext, key_size):
    # Prepara el texto plano para ser compatible con la matriz de clave
    plaintext = plaintext.replace(" ", "").upper()
    while len(plaintext) % key_size != 0:
        plaintext += "X"
    return plaintext

def text_to_matrix(text, key_size):
    # Convierte el texto en una matriz de números correspondientes a las letras
    matrix = []
    for char in text:
        matrix.append(ord(char) - ord('A'))
    return np.array(matrix).reshape(-1, key_size)

def matrix_to_text(matrix):
    # Convierte una matriz de números en texto cifrado
    text = ""
    for row in matrix:
        for num in row:
            text += chr(num + ord('A'))
    return text
```

Figura 9: Cifrado de Hill parte 1

```python
def hill_encrypt(plaintext, key_matrix):
    key_size = key_matrix.shape[0]
    plaintext = prepare_plaintext(plaintext, key_size)
    plaintext_matrix = text_to_matrix(plaintext, key_size)
    encrypted_matrix = np.dot(plaintext_matrix, key_matrix) % 26
    ciphertext = matrix_to_text(encrypted_matrix)
    return ciphertext

# Ejemplo de uso
key_matrix = np.array([[6, 24, 1], [13, 16, 10], [20, 17, 15]])
plaintext = "viernes"
ciphertext = hill_encrypt(plaintext, key_matrix)
print("Mensaje cifrado:", ciphertext)
```

```
Mensaje cifrado: YYFNIZJVV
```

Figura 10: Cifrado de Hill y Resultados de cifrado Hill

Similar al cifrado anterior, *ChatGPT* cifra bien el texto sin embargo utiliza diferentes métodos y diferentes llaves, sí multiplica las matrices de manera correcta pero no son las mismas que nosotros utilizamos. *ChatGPT* utiliza una matriz como llave de 3x3 mientras que nosotros la utilizamos de 2x2 afectando que los resultados coindican.

Podemos corregir este resultado especificando a *ChatGPT* que la matriz que usamos como llave tenga tamaño 2x2