Universidad Nacional Autónoma de México
Facultad de Estudios Superiores Aragón

Ingeniería en Computación

SEGURIDAD INFORMATICA

Grupo: 1910

Profesora
MARIANA VERDUZCO RODRIGUEZ

Proyecto

Equipo
Cruz Cervantes Guadalupe Sugeily
Curiel Martinez Luis Angel
Siordia Olvera Fernanda Vidkar

Índice

Introducción ... 3

Descarga de código mediante GitHub. ... 3

Desarrolle un programa que permita encriptar y desencriptar un texto mediante los siguientes tipos de cifrado: ... 4

- Cifrado César ... 4
- Cifrada sustitución ... 6
- Cifrado Vigenère ... 9
- Función principal ... 11

Conclusión ... 14

Referencias ... 15

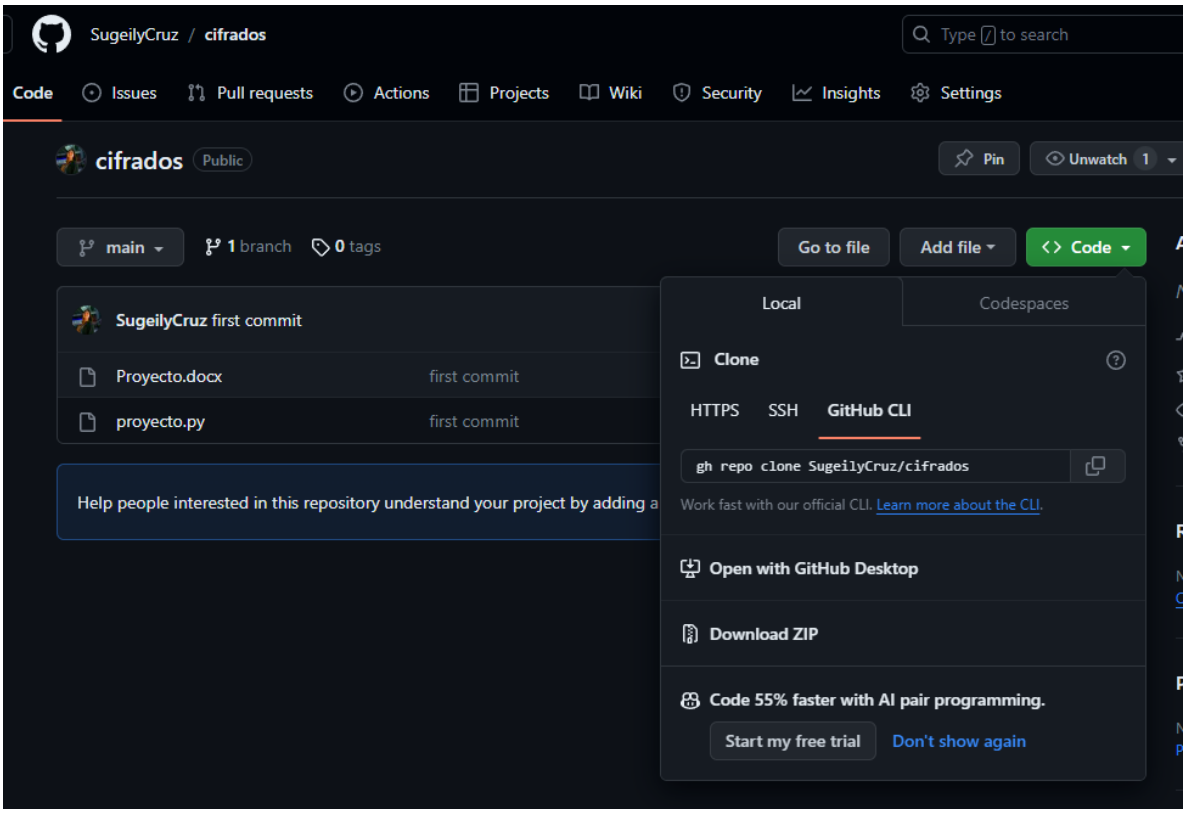
Introducción

La criptografía, desde tiempos inmemoriales, ha sido el baluarte de la privacidad y la confidencialidad en las comunicaciones. Entre sus técnicas más célebres y fundamentales se encuentran los cifrados de sustitución, el cifrado César y el cifrado Vigenère. En este proyecto, nos adentraremos en el fascinante mundo de la criptografía clásica para comprender, implementar y analizar estos métodos que han resistido el paso del tiempo.

El cifrado de sustitución, la piedra angular de muchos métodos criptográficos se basa en reemplazar cada elemento del mensaje original por otro siguiendo reglas específicas. Por su parte, el cifrado César, un caso particular de sustitución, opera mediante un desplazamiento fijo de caracteres a lo largo del alfabeto. Mientras tanto, el cifrado Vigenère, más sofisticado, utiliza una clave para aplicar una serie de desplazamientos variables, lo que añade una capa adicional de seguridad y complejidad.

Descarga de código mediante GitHub.

https://github.com/SugeilyCruz/cifrados



Desarrolle un programa que permita encriptar y desencriptar un texto mediante los siguientes tipos de cifrado:

- **Cifrado César**

```python
#CIFRADO CESAR
def cifraCesar(cad, llave):
    cad = cad.lower()
    cifrado = ""
    alfabeto = "abcdefghijklmnopqrstuvwxyz"
    for c in cad:
        if (c in alfabeto):
            pos = alfabeto.index(c)
            pos2 = (pos+llave)%26
            cifrado = cifrado + alfabeto[pos2]
        else:
            cifrado = cifrado + c
    return cifrado

def descifraCesar(cad, llave):
    cad = cad.lower()
    cifrado = ""
    alfabeto = "abcdefghijklmnopqrstuvwxyz"
    for c in cad:
        if (c in alfabeto):
            pos = alfabeto.index(c)
            pos2 = (pos-llave)%26
            cifrado = cifrado + alfabeto[pos2]
        else:
            cifrado = cifrado + c
    return cifrado
```

Explicación del código línea por línea:

- **def cifraCesar(cad, llave):**: Define una función llamada **cifraCesar** que toma dos argumentos: **cad** (cadena de texto a cifrar) y **llave** (número entero que representa la cantidad de desplazamiento en el cifrado César).
- **cad = cad.lower()**: Convierte toda la cadena de texto **cad** a minúsculas para asegurar coherencia en el procesamiento.
- **cifrado = ""**: Inicializa una cadena vacía llamada **cifrado** que almacenará el texto cifrado.
- **alfabeto = "abcdefghijklmnopqrstuvwxyz"**: Define una cadena que contiene todas las letras del alfabeto en minúsculas.
- **for c in cad:**: Inicia un bucle que recorre cada carácter en la cadena de texto **cad**.
- **if (c in alfabeto):**: Verifica si el carácter **c** está presente en el alfabeto.

- **pos = alfabeto.index(c)**: Obtiene la posición del carácter **c** dentro del alfabeto.
- **pos2 = (pos+llave)%26**: Calcula la nueva posición (**pos2**) aplicando el desplazamiento (**llave**) al carácter actual. Se utiliza el módulo 26 para asegurar que el índice esté dentro del rango del alfabeto.
- **cifrado = cifrado + alfabeto[pos2]**: Agrega el carácter cifrado correspondiente a la cadena **cifrado**.

- **else: cifrado = cifrado + c**: Si el carácter **c** no está en el alfabeto (por ejemplo, espacios, signos de puntuación, etc.), simplemente se agrega tal como está a la cadena **cifrado**.
- **return cifrado**: Devuelve la cadena de texto cifrada.

La función **descifraCesar** es similar a **cifraCesar**, pero en lugar de desplazar los caracteres hacia adelante, los desplaza hacia atrás (**pos-llave**) para realizar el descifrado del mensaje.

```
MA DE MÉXICO\9 sem\Seguridad\proyecto.py'
Seleccione el tipo de cifrado:
1. Cifrado Cesar
2. Cifrado de Sustitucion
3. Cifrado Vigenere

Seleccione el tipo de descifrado:
4. desCifrado Cesar
5. desCifrado de Sustitucion
6. desCifrado Vigenere

7. Salir

Ingrese el numero correspondiente al tipo de cifrado: 1
Ingrese el texto a cifrar: Hola mundo
Por favor, ingrese un numero para el desplazamiento en el cifrado Cesar: 3
Texto final: krod pxqgr
```

```
Seleccione el tipo de cifrado:
1. Cifrado Cesar
2. Cifrado de Sustitucion
3. Cifrado Vigenere

Seleccione el tipo de descifrado:
4. desCifrado Cesar
5. desCifrado de Sustitucion
6. desCifrado Vigenere

7. Salir

Ingrese el numero correspondiente al tipo de cifrado: 4
Ingrese el texto a descifrar: krod pxqgr
Por favor, ingrese un numero para el desplazamiento en el cifrado Cesar: 3
Texto final: hola mundo
```

- **Cifrada sustitución**

```python
#CIFRADO SUSTITUCION
def cifraSustituye (cad, llave):
    alfabeto = "abcdefghijklmnopqrstuvwxyz"
    cad = cad.lower()
    cifrado = ""
    for c in cad:
        if (c in alfabeto):
            pos=alfabeto.index(c)
            cifrado=cifrado+llave[pos]
        else:
            cifrado=cifrado+c
    return cifrado

def descifraSustituye (cad, llave):
    alfabeto = "abcdefghijklmnopqrstuvwxyz"
    cad = cad.lower()
    texto = ""
    for c in cad:
        if (c in alfabeto):
            pos=llave.index(c)
            texto=texto+alfabeto[pos]
        else:
            texto=texto+c
    return texto
```

```python
def creaLlaveAleatoria():
    alfabeto = "abcdefghijklmnopqrstuvwxyz"
    cad =""
    while (len(cad)<len(alfabeto)):
        ale = random.randint(0, len(alfabeto)-1)
        c=alfabeto[ale]
        if not (c in cad):
            cad = cad+c
    return cad
```

Explicación del código:

Función cifraSustituye(cad, llave)

- **def cifraSustituye(cad, llave):**: Define una función llamada **cifraSustituye** que toma dos argumentos: **cad** (cadena de texto a cifrar) y **llave** (clave de sustitución que se utilizará para cifrar).

- **alfabeto = "abcdefghijklmnopqrstuvwxyz"**: Define una cadena que contiene todas las letras del alfabeto en minúsculas.
- **cad = cad.lower()**: Convierte toda la cadena de texto **cad** a minúsculas para asegurar coherencia en el procesamiento.
- **cifrado = ""**: Inicializa una cadena vacía llamada **cifrado** que almacenará el texto cifrado.
- **for c in cad:**: Inicia un bucle que recorre cada carácter en la cadena de texto **cad**.
    - **if (c in alfabeto):**: Verifica si el carácter **c** está presente en el alfabeto.
        - **pos = alfabeto.index(c)**: Obtiene la posición del carácter **c** dentro del alfabeto.
        - **cifrado = cifrado + llave[pos]**: Agrega el carácter correspondiente a la posición **pos** en la cadena **llave** a la cadena **cifrado**. La **llave** es una cadena que contiene la sustitución de letras del alfabeto.
    - **else: cifrado = cifrado + c**: Si el carácter **c** no está en el alfabeto (por ejemplo, espacios, signos de puntuación, etc.), simplemente se agrega tal como está a la cadena **cifrado**.
- **return cifrado**: Devuelve la cadena de texto cifrada.

Función descifraSustituye(cad, llave)

- **def descifraSustituye(cad, llave):**: Define una función llamada **descifraSustituye** que toma dos argumentos: **cad** (cadena de texto cifrado) y **llave** (clave de sustitución utilizada para el cifrado).
- **cad = cad.lower()**: Convierte toda la cadena de texto **cad** a minúsculas para asegurar coherencia en el procesamiento.
- **texto = ""**: Inicializa una cadena vacía llamada **texto** que almacenará el texto descifrado.
- **for c in cad:**: Inicia un bucle que recorre cada carácter en la cadena de texto **cad**.
    - **if (c in alfabeto):**: Verifica si el carácter **c** está presente en el alfabeto.
        - **pos = llave.index(c)**: Obtiene la posición del carácter **c** dentro de la cadena **llave**.
        - **texto = texto + alfabeto[pos]**: Agrega el carácter correspondiente a la posición **pos** en el alfabeto original a la cadena **texto**. Esto sirve para descifrar el mensaje.

- **else: texto = texto + c**: Si el carácter **c** no está en el alfabeto (por ejemplo, espacios, signos de puntuación, etc.), simplemente se agrega tal como está a la cadena **texto**.

- **return texto**: Devuelve la cadena de texto descifrada.

Función creaLlaveAleatoria()

- **def creaLlaveAleatoria():**: Define una función llamada **creaLlaveAleatoria** que genera una clave aleatoria para el cifrado de sustitución.
- **alfabeto = "abcdefghijklmnopqrstuvwxyz"**: Define una cadena que contiene todas las letras del alfabeto en minúsculas.
- **cad = ""**: Inicializa una cadena vacía llamada **cad** que almacenará la clave aleatoria.
- **while (len(cad) < len(alfabeto)):**: Inicia un bucle que se ejecutará mientras la longitud de **cad** sea menor que la longitud del alfabeto.
    - **ale = random.randint(0, len(alfabeto)-1)**: Genera un número aleatorio entre 0 y la longitud del alfabeto - 1.
    - **c = alfabeto[ale]**: Obtiene el carácter correspondiente al índice aleatorio en el alfabeto.
    - **if not (c in cad): cad = cad + c**: Si el carácter no está presente en la cadena **cad**, se agrega a la cadena.
- **return cad**: Devuelve la clave aleatoria generada.

```
Seleccione el tipo de cifrado:
1. Cifrado Cesar
2. Cifrado de Sustitucion
3. Cifrado Vigenere

Seleccione el tipo de descifrado:
4. desCifrado Cesar
5. desCifrado de Sustitucion
6. desCifrado Vigenere

7. Salir

Ingrese el numero correspondiente al tipo de cifrado: 2
Ingrese el texto a cifrar: hola mundo
Si desea que la llave sea generada por el programa coloque la palabra 'default'
Por favor, ingrese una llave para el cifrado por sustitución: default
Su llave para el cifrado por sustitución es: mjpzdiyfnrbwqavgtlcxheksuo
Texto final: fvwm qhazv
```

```
Seleccione el tipo de cifrado:
1. Cifrado Cesar
2. Cifrado de Sustitucion
3. Cifrado Vigenere

Seleccione el tipo de descifrado:
4. desCifrado Cesar
5. desCifrado de Sustitucion
6. desCifrado Vigenere

7. Salir

Ingrese el numero correspondiente al tipo de cifrado: 5
Ingrese el texto a descifrar: fvwm qhazv
Por favor, ingrese una llave para el descifrado por sustitución: mjpzdiyfnrbwqavgtlcxheksuo
Texto final: hola mundo
```

- **Cifrado Vigenère**

```python
#CIFRADO Vigenère
abc = "abcdefghijklmnopqrstuvwxyz "
def cifrarVigenere(cad, clave):
    cadena = cad.lower()
    text_cifrar = ""
    i = 0
    for letra in cadena:
        suma = abc.find(letra) + abc.find(clave[i % len(clave)])
        modulo = int(suma) % len(abc)
        text_cifrar = text_cifrar + str(abc[modulo])
        i = i + 1
    return text_cifrar

def descifrarVigenere(cad, clave):
    cadena = cad.lower()
    text_cifrar = ""
    i = 0
    for letra in cadena:
        suma = abc.find(letra) - abc.find(clave[i % len(clave)])
        modulo = int(suma) % len(abc)
        text_cifrar = text_cifrar + str(abc[modulo])
        i = i + 1
    return text_cifrar
```

abc = "abcdefghijklmnopqrstuvwxyz ": Define una cadena que contiene todas las letras del alfabeto en minúsculas, incluyendo un espacio al final que representa los caracteres no alfabéticos.

Función cifrarVigenere(cad, clave)

- **def cifrarVigenere(cad, clave):**: Define una función llamada **cifrarVigenere** que toma dos argumentos: **cad** (cadena de texto a cifrar) y **clave** (clave que se utilizará para el cifrado Vigenère).
- **cadena = cad.lower()**: Convierte toda la cadena de texto **cad** a minúsculas para asegurar coherencia en el procesamiento.
- **text_cifrar = ""**: Inicializa una cadena vacía llamada **text_cifrar** que almacenará el texto cifrado.
- **i = 0**: Inicializa un contador **i** para iterar sobre la clave.
- **for letra in cadena:**: Inicia un bucle que recorre cada letra en la cadena de texto **cadena**.
    - **suma = abc.find(letra) + abc.find(clave[i % len(clave)])**: Obtiene la posición de la letra en el alfabeto (**abc.find(letra)**) y suma la posición de la letra correspondiente en la clave (**clave[i % len(clave)]**) para obtener la nueva posición de la letra cifrada.
    - **modulo = int(suma) % len(abc)**: Calcula el residuo de la división de **suma** entre la longitud del alfabeto **len(abc)**, asegurando que la posición resultante esté dentro del rango del alfabeto.
    - **text_cifrar = text_cifrar + str(abc[modulo])**: Agrega la letra cifrada correspondiente al residuo calculado a la cadena **text_cifrar**.
    - **i = i + 1**: Incrementa el contador **i** para avanzar a la siguiente letra de la clave.
- **return text_cifrar**: Devuelve la cadena de texto cifrada.

Función descifrarVigenere(cad, clave)

- **def descifrarVigenere(cad, clave):**: Define una función llamada **descifrarVigenere** que toma dos argumentos: **cad** (cadena de texto cifrado) y **clave** (clave que se utilizó para cifrar el texto).

La función **descifrarVigenere** realiza un proceso similar al de cifrado, pero en lugar de sumar las posiciones de las letras, resta la posición de la letra cifrada con la posición correspondiente en la clave para obtener la posición original del carácter en el alfabeto.

```
MA DE MEXICO\9 sem\Seguridad\proyecto.py
Seleccione el tipo de cifrado:
1. Cifrado Cesar
2. Cifrado de Sustitucion
3. Cifrado Vigenere

Seleccione el tipo de descifrado:
4. desCifrado Cesar
5. desCifrado de Sustitucion
6. desCifrado Vigenere

7. Salir

Ingrese el numero correspondiente al tipo de cifrado: 3
Ingrese el texto a cifrar: hola mundo
Por favor, ingrese una clave para el cifrado Vigenère: arbol
Texto final: hemokmkorz
```

```
Seleccione el tipo de cifrado:
1. Cifrado Cesar
2. Cifrado de Sustitucion
3. Cifrado Vigenere

Seleccione el tipo de descifrado:
4. desCifrado Cesar
5. desCifrado de Sustitucion
6. desCifrado Vigenere

7. Salir

Ingrese el numero correspondiente al tipo de cifrado: 6
Ingrese el texto a descifrar: hemokmkorz
Por favor, ingrese una clave para el descifrado Vigenère: arbol
Texto final: hola mundo
```

- **Función principal**

```python
# Función principal para seleccionar el tipo de cifrado
def main():
    while True:
        print("Seleccione el tipo de cifrado:")
        print("1. Cifrado Cesar")
        print("2. Cifrado de Sustitucion")
        print("3. Cifrado Vigenere")
        print("\nSeleccione el tipo de descifrado:")
        print("4. desCifrado Cesar")
        print("5. desCifrado de Sustitucion")
        print("6. desCifrado Vigenere")
        print("\n7. Salir\n")

        opcion = int(input("Ingrese el numero correspondiente al tipo de cifrado: "))

        if opcion == 1:
            texto = input("Ingrese el texto a cifrar: ")
            desplazamiento = int(input("Por favor, ingrese un numero para el desplazamiento en el cifrado Cesar: "))
            texto_cifrado = cifraCesar(texto, desplazamiento)
            print("Texto final:", texto_cifrado, "\n")

            time.sleep(1)
```

```
        elif opcion == 2:
            texto = input("Ingrese el texto a cifrar: ")
            print("Si desea que la llave sea generada por el programa coloque la palabra 'default' ")
            llave = input("Por favor, ingrese una llave para el cifrado por sustitución: ")
            if llave == "default":
                llave = creaLlaveAleatoria()
                print(f"Su llave para el cifrado por sustitución es: {llave}")
            texto_cifrado = cifraSustituye(texto, llave)
            print("Texto final:", texto_cifrado, "\n")
            time.sleep(1)
        elif opcion == 3:
            texto = input("Ingrese el texto a cifrar: ")
            clave = input("Por favor, ingrese una clave para el cifrado Vigenère: ")
            texto_cifrado = cifrarVigenere(texto, clave)
            print("Texto final:", texto_cifrado, "\n")
            time.sleep(1)
        elif opcion == 4:
            texto = input("Ingrese el texto a descifrar: ")
            desplazamiento = int(input("Por favor, ingrese un numero para el desplazamiento en el cifrado Cesar: "))
            texto_cifrado = descifraCesar(texto, desplazamiento)
            print("Texto final:", texto_cifrado, "\n")
            time.sleep(1)
```

```
        elif opcion == 5:
            texto = input("Ingrese el texto a descifrar: ")
            llave = input("Por favor, ingrese una llave para el descifrado por sustitución: ")
            texto_cifrado = descifraSustituye(texto, llave)
            print("Texto final:", texto_cifrado, "\n")
            time.sleep(1)
        elif opcion == 6:
            texto = input("Ingrese el texto a descifrar: ")
            clave = input("Por favor, ingrese una clave para el descifrado Vigenère: ")
            texto_cifrado = descifrarVigenere(texto, clave)
            print("Texto final:", texto_cifrado, "\n")
            time.sleep(1)
        elif opcion == 7:
            print("Saliendo del programa...")
            break
        else:
            print("Opción no válida. Por favor, ingresa un número del 1 al 7.")

# Llamada a la función principal
if __name__ == "__main__":
    main()
```

Este bloque de código es una implementación de un menú interactivo que permite al usuario seleccionar entre diferentes tipos de cifrado y descifrado: Cifrado César, Cifrado de Sustitución y Cifrado Vigenère.

Explicación del código:

1. **Función main()**:
    - Inicia un bucle infinito con **while True**.
    - Muestra un menú con opciones para cifrar y descifrar mediante diferentes métodos de cifrado.

- Solicita al usuario que ingrese un número correspondiente a la opción deseada.
- Según la opción seleccionada, solicita al usuario que ingrese el texto a cifrar/descifrar, así como los parámetros específicos requeridos para cada tipo de cifrado.
- Realiza el cifrado/descifrado utilizando las funciones respectivas para cada tipo de algoritmo.
- Imprime el resultado del proceso y espera un segundo ante de volver a mostrar el menú.

2. **Bloques if**:
    - Cada bloque **if** verifica la opción seleccionada por el usuario.
    - En cada caso, solicita la información necesaria para el cifrado o descifrado correspondiente.
    - Llama a la función respectiva para realizar el cifrado o descifrado según la opción seleccionada.
3. **Condición if __name__ == "__main__":**:
    - Esta condición verifica si el script está siendo ejecutado directamente (no importado como módulo).
    - En este caso, llama a la función **main()** para iniciar el programa.

Funciones utilizadas:

- **cifraCesar()**, **descifraCesar()**, **cifraSustituye()**, **descifraSustituye()**, **cifrarVigenere()**, **descifrarVigenere()**: Estas son funciones definidas previamente que implementan los diferentes tipos de cifrado y descifrado.

Uso de time.sleep(1):

- La función **time.sleep(1)** se utiliza para generar una pausa de 1 segundo después de mostrar el resultado del cifrado o descifrado. Esto da tiempo al usuario para ver la información antes de que el menú vuelva a mostrarse.

En resumen, este bloque de código constituye un programa interactivo que permite al usuario elegir entre diferentes opciones de cifrado y descifrado, utilizando las funciones previamente definidas para cada método de cifrado clásico.

Conclusión

En este proyecto, hemos realizado la criptografía clásica, explorando y comprendiendo los principios fundamentales detrás de tres de los cifrados más emblemáticos: el Cifrado de Sustitución, el Cifrado César y el Cifrado Vigenère.

Hemos descubierto que el Cifrado de Sustitución actúa como un pilar fundamental en la historia de la criptografía, estableciendo la base para técnicas más complejas. A través de la sustitución de cada elemento del mensaje original por otro según reglas específicas, se logra ocultar el contenido de manera sencilla pero limitada en su seguridad.

El Cifrado César, un ejemplo simple de sustitución, demuestra cómo un desplazamiento fijo de caracteres a lo largo del alfabeto puede ser eficaz para ocultar el mensaje, aunque es vulnerable a ataques de fuerza bruta debido a su limitado número de combinaciones.

Por otro lado, el Cifrado Vigenère, al introducir una clave que permite desplazamientos variables a lo largo del alfabeto, añade una capa adicional de complejidad. Esto dificulta los ataques de criptoanálisis y refuerza la seguridad del cifrado, haciendo que sea más resistente frente a varios métodos de descifrado tradicionales.

Con este proyecto, hemos apreciado la belleza y la lógica detrás de estos métodos clásicos, así como sus limitaciones y vulnerabilidades en un entorno moderno de seguridad de la información. Además, hemos observado cómo estas técnicas han allanado el camino hacia el desarrollo de sistemas criptográficos más avanzados y sofisticados que se utilizan en la actualidad.

Referencias

Cifrado César. (2018, July 8). EduEscapeRoom. https://eduescaperoom.com/cifrado-cesar/

González, A. (2020, June 10). ¿Qué es el cifrado César y cómo funciona? Ayuda Ley Protección Datos; AyudaLeyProteccionDatos. https://ayudaleyprotecciondatos.es/2020/06/10/cifrado-cesar/

El cifrado de Cesar. (2023). Www.ugr.es. http://www.ugr.es/~anillos/textos/pdf/2012/EXPO-1.Criptografia/02a04.htm

fran. (2006, October 23). Critpografía: Cifrado por sustitución - Gaussianos. Gaussianos. https://www.gaussianos.com/critpografia-cifrado-por-sustitucion/

Timur : miembro planetcalc. (2021). Calculadora en línea: Herramienta de cifrado por sustitución. Planetcalc.com. https://es.planetcalc.com/7984/

Cifrado de sustitución _ AcademiaLab. (2013). Academia-Lab.com. https://academia-lab.com/enciclopedia/cifrado-de-sustitucion/

El cifrado de Vigenère. (2023). Www.ugr.es. https://www.ugr.es/~anillos/textos/pdf/2012/EXPO-1.Criptografia/02a11.htm

Alfonso Jesús Población. (2018, November 19). La cifra Vigenère: el misterioso código que se tardó tres siglos en descifrar. Diario ABC; ABC.es. https://www.abc.es/ciencia/abci-cifra-vigenere-misterioso-codigo-tardo-tres-siglos-descifrar-201811192150_noticia.html