# RETO 3 - FUNDAMENTOS DE PROGRAMACIÓN

Usted desea implementar un sistema de comunicación segura en el que los mensajes se transformen en un artefacto no reconocible por aquellos que no cuenten con la clave de desencriptado pero que pueda ser traducido por aquellos que sí cuenten con ella. Un método posible para encriptar un mensaje consiste en tomar cada caracter de él, convertirlo en un número entero usando alguna regla, “desorganizar” los números resultantes y escribir esta nueva secuencia de números en una matriz cuadrada, para compactar el mensaje.

Para el desencriptado del mensaje, se tomaría el mensaje encriptado (en una matriz cuadrada) y una clave de encriptado (algo que sugiera cómo se desorganizaron los números inicialmente) para revertir con ellos dos todos los procesos involucrados en el encriptado y obtener el mensaje original. Recordemos que una matriz cuadrada es una matriz en la que la cantidad de filas en ella es igual a la cantidad de columnas.

Este método no serviría para aquellos mensajes que tengan una longitud que no sea igual al cuadrado de algún entero (que no sean de 1 carácter de largo, de 4 caracteres de largo, de 9 caracteres de largo, ...) porque una matriz cuadrada siempre tendrá una cantidad de entradas igual a alguna potencia de un entero y quedarían o faltando o sobrando caracteres en el llenado de esa matriz cuadrada.

Pero si se le agrega al mensaje que se va a encriptar la cantidad suficiente de caracteres extra para que la longitud de este mensaje termine siendo igual a la potencia de algún entero, se puede utilizar este método de encriptado. A modo de ejemplo, supongamos que se desea encriptar este mensaje:

Today is Caturday!

Este mensaje contiene 18 caracteres, incluyendo los espacios. Con ellos no se podría llenar completamente ninguna matriz cuadrada porque ellas tendrán 4, 9, 16, 25, ... entradas; no obstante, si se “rellena” el mensaje con caracteres adicionales así:

Today is Caturday!_____

Se obtendrá un nuevo mensaje de 25 caracteres que sí llenará completamente una matriz cuadrada, en este caso, una matriz de 5 × 5. Este llenado se vería así:

$$\begin{vmatrix} T & o & d & a & y \\ & i & s & & C \\ a & t & u & r & d \\ a & y & ! & \_ & \_ \\ \_ & \_ & \_ & \_ & \_ \end{vmatrix}$$

Con el mensaje dispuesto así en una matriz cuadrada, se puede iniciar la encriptación cambiando cada letra por un número y continuar con los demás pasos hasta obtener un mensaje encriptado dentro de una matriz cuadrada.

Python contiene una pareja de métodos: El método “ord(caracter)” que recibe un caracter y retorna el código en Unicode de dicho carácter; y el método “chr(código)” que recibe un número entero (código) y retorna el caracter correspondiente a dicho valor en Unicode. Este método puede usarse para iniciar el encriptado del mensaje; pero por sí solo no ofrecería protección suficiente del mensaje. En la figura a continuación se muestra una tabla que muestra las equivalencias de las letras mayúsculas y minúsculas con sus códigos en Unicode.

| CARACTER | CÓDIGO UNICODE | CARACTER | CÓDIGO UNICODE | CARACTER | CÓDIGO UNICODE | CARACTER | CÓDIGO UNICODE | CARACTER | CÓDIGO UNICODE |
|---|---|---|---|---|---|---|---|---|---|
| A | 65 | N | 78 | a | 97 | n | 110 | ! | 33 |
| B | 66 | O | 79 | b | 98 | o | 111 | " | 34 |
| C | 67 | P | 80 | c | 99 | p | 112 | & | 38 |
| D | 68 | Q | 81 | d | 100 | q | 113 | , | 44 |
| E | 69 | R | 82 | e | 101 | r | 114 | - | 45 |
| F | 70 | S | 83 | f | 102 | s | 115 | . | 46 |
| G | 71 | T | 84 | g | 103 | t | 116 | : | 58 |
| H | 72 | U | 85 | h | 104 | u | 117 | ; | 59 |
| I | 73 | V | 86 | i | 105 | v | 118 | ? | 63 |
| J | 74 | W | 87 | j | 106 | w | 119 | _ | 95 |
| K | 75 | X | 88 | k | 107 | x | 120 | ¡ | 161 |
| L | 76 | Y | 89 | l | 108 | y | 121 | ¿ | 191 |
| M | 77 | Z | 90 | m | 109 | z | 122 | ' | 39 |

Fuente: Creación propia

Para alcanzar el objetivo que se planteó, usted se propuso seguir este protocolo de encriptado:

- Agregar al mensaje tantos guiones de pie (_) como sean necesarios para que la cantidad total de caracteres en este sea igual a la potencia de un número, en particular, la potencia más cercana al número original de caracteres en el mensaje.
- Insertar todos los caracteres del mensaje y los guiones de relleno en una lista para después “desordenar” dicha lista, haciendo secreto el mensaje

original; así, el nuevo orden de los caracteres del mensaje no lo va a revelar directamente. Si se cuenta con un "patrón de desorden", este será la CLAVE que permitirá la traducción del mensaje encriptado.

- Convertir cada caracter a su respectivo código de Unicode usando la función "ord"; así en lugar de letras, signos de puntuación y números se tendrán códigos de Unicode representando el mensaje

El "patrón de desorden" sería la lista de todos los índices de los caracteres del mensaje original, aleatorizada usando alguna herramienta como la librería "random" de Python.

```
1  codigo_unicode = ord('a')
2  caracter_unicode = chr(97)
```

| | |
|---|---|
| codigo_unicode | 97 |
| caracter_unicode | "a" |

La pareja de funciones ord('carácter') y chr(código)

Terminado este proceso ya se tiene un mensaje encriptado en una matriz cuadrada que, aparte del mensaje original, va a contener al final unos caracteres de relleno (en nuestro caso, guiones de pie "_") A continuación se ilustran visualmente los pasos para encriptar el mensaje "Today is Caturday!"

Today is Caturday!→Today is Caturday!_____

→ [T o d a y  i s  C a t u r d a y ! _ _ _ _ _ _]

Supóngase que se tiene esta aleatorización de los índices de la lista anterior, es decir, esta CLAVE:

[1 7 9 8 22 14 23 16 19 12 21 5 3 24 20 13 18 2 17 10 4 0 15 6 11]

Que en total son 25 porque el mensaje a encriptar tiene 25 caracteres (originales + relleno) Si se usa esta aleatorización de los índices para reordenar o "desordenar" los caracteres se tendría:

| ÍNDICES | 1 | 7 | 9 | 8 | 22 | 14 | 23 | 16 | 19 | 12 | 21 | 5 | 3 | 24 | 20 | 13 | 18 | 2 | 17 | 10 | 4 | 0 | 15 | 6 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CARACTER | o | s | C | | _ | d | _ | y | _ | u | _ | | a | _ | _ | r | _ | d | ! | a | y | T | a | i | t |

Se traducen los caracteres de esta nueva lista a su respectivo código de Unicode usando ord() y se tendría:

| ÍNDICES | 1 | 7 | 9 | 8 | 22 | 14 | 23 | 16 | 19 | 12 | 21 | 5 | 3 | 24 | 20 | 13 | 18 | 2 | 17 | 10 | 4 | 0 | 15 | 6 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CARACTER | o | s | C | | _ | d | _ | y | _ | u | _ | | a | _ | _ | r | _ | d | ! | a | y | T | a | i | t |
| CÓDIGO EN UNICODE | 111 | 115 | 67 | 32 | 95 | 100 | 95 | 121 | 95 | 117 | 95 | 32 | 97 | 95 | 95 | 114 | 95 | 100 | 33 | 97 | 121 | 84 | 97 | 105 | 116 |

Esta nueva lista sería el mensaje encriptado que, puede ser convertido en una matriz cuadrada porque tiene una cantidad de elementos igual al cuadrado de un número entero ($25 = 5^2$)

$$\begin{vmatrix} 111 & 115 & 67 & 32 & 95 \\ 100 & 95 & 121 & 95 & 117 \\ 95 & 32 & 97 & 95 & 95 \\ 114 & 95 & 100 & 33 & 97 \\ 121 & 84 & 97 & 105 & 116 \end{vmatrix}$$

El proceso de desencriptado requiere como parámetros de entrada a la matriz con el mensaje encriptado y de la lista clave, y procedería como sigue:

- Recuperar los caracteres de la matriz a partir de sus códigos de Unicode
- Reorganizar la lista clave y con ella los caracteres recién recuperados
- Eliminar todos los guiones de pie usados para rellenar el mensaje y así, lograr el desencriptado.

Siendo así y basados en el ejemplo anterior, el proceso de desencriptado se vería así:

| CÓDIGO EN UNICODE | 111 | 115 | 67 | 32 | 95 | 100 | 95 | 121 | 95 | 117 | 95 | 32 | 97 | 95 | 95 | 114 | 95 | 100 | 33 | 97 | 121 | 84 | 97 | 105 | 116 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CARACTER | o | s | C | | _ | d | _ | y | _ | u | _ | | a | _ | _ | r | _ | d | ! | a | y | T | a | i | t |
| ÍNDICES | 1 | 7 | 9 | 8 | 22 | 14 | 23 | 16 | 19 | 12 | 21 | 5 | 3 | 24 | 20 | 13 | 18 | 2 | 17 | 10 | 4 | 0 | 15 | 6 | 11 |
| DESENCRIPTADO | T | o | d | a | y | | | | | | | | | | | | | | | | | | | | |
| ÍNDICES | 0 | 1 | 2 | 3 | 4 | | | | | | | | | | | | | | | | | | | | |

## TAREAS

Realizar un programa en Python con dos funciones, una de encriptado y otra de desencriptado, que le permita a usted procesar cualquier mensaje enviado a la función de encriptado y recuperar el mensaje original con la función de desencriptado.

La ejecución de la función de encriptado debe realizarse a través de una función denotada así:

encriptado(texto)

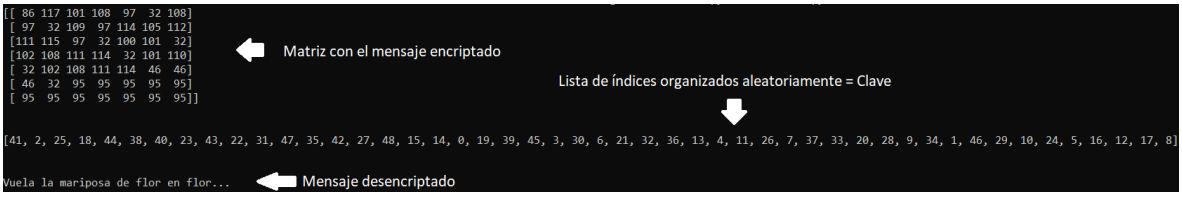
Que recibirá como parámetro alguna cadena de caracteres y que retornará la matriz con el mensaje encriptado y la lista de índices ordenados de manera aleatoria, que será la clave de desencriptado.

La ejecución de la función de desencriptado debe realizarse a través de una función denotada así:

desencriptado(matriz_encriptado, clave)

Que recibirá como parámetros la matriz resultante del encriptado y la lista clave. Esta función deberá retornar el mensaje original que se encriptó sin los caracteres de relleno usados para obtener matrices cuadradas.

En este reto no habrá impresiones por consola ni ingreso de información por ella, los datos se transferirán por llamado de funciones y de retornos; pero a modo de ejemplo ilustrativo, se muestra a continuación los dos retornos esperados de la función encriptado(texto) y el retorno esperado de la función desencriptado(matriz_encriptado, clave) para el texto “Vuela la mariposa de flor en flor... “ (espacio en blanco después de los puntos suspensivos)




**PARA LA IMPLEMENTACIÓN DE LA SOLUCIÓN NO DEBE USAR EL INGRESO DE INFORMACIÓN POR CONSOLA.** Debe crear su programa de tal forma que toda la información que se usará para validar su solución se proporcione a través de un sólo parámetro de entrada (el texto a encriptar)

**RECURRA A LA LIBRERÍA NUMPY Y SUS FUNCIONES.** Es esta librería y sus funciones principales las que le permitirán manipular matrices y listas como lo necesita para solucionar el problema. La creación de arreglos de ceros o de unos puede servir para el llenado de listas y de matrices a través del reemplazo de entradas en ellas.

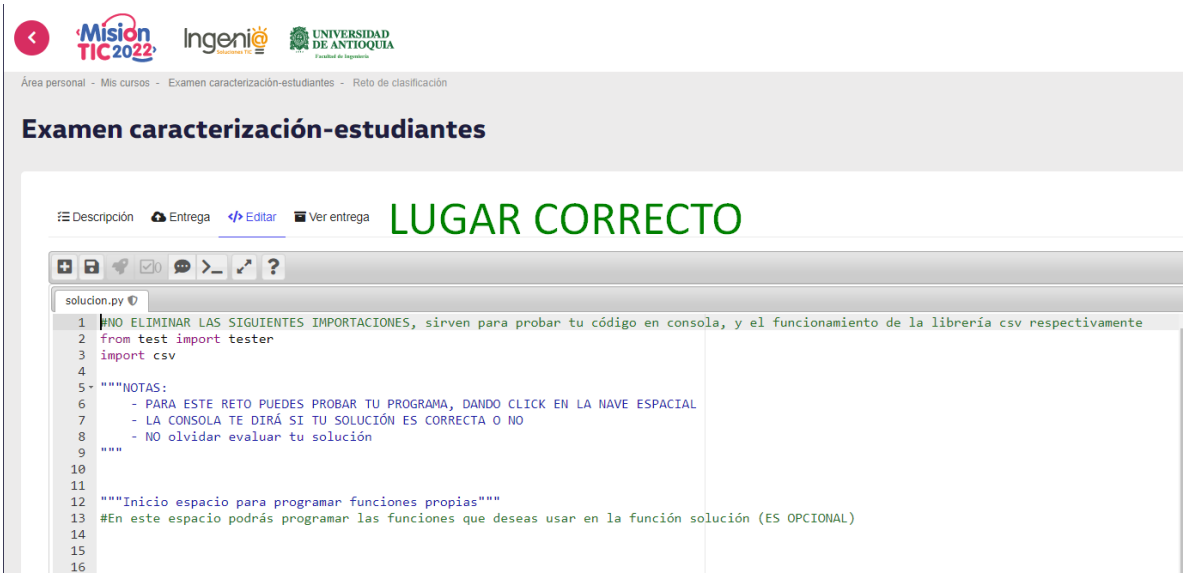
El trabajo con listas y arreglos de ceros y unos es mucho más fácil por lo tanto procure usarlos en la medida de lo posible excepto en aquellas operaciones matriciales en las que invariablemente debe recurrir a objetos de la librería Numpy.

## NOTA ACLARATORIA

Se recomienda desarrollar la prueba en un IDE como G Colab, VSCode, PyCharm, Spyder, etc. Al final debe copiar y pegar el código en la herramienta VPL, pero **NO** deberá subir archivos, es decir:

**Modo incorrecto:**

## Modo correcto:




**TRIPULANTE, ¡MUCHOS ÉXITOS EN EL DESARROLLO DEL RETO 3!**