**ANEXO A: Manual de Usuario**

Instituto Tecnológico de Costa Rica

Ingeniería en Computación

-Curso:

Taller de Programación

-Docente:

Luis Pablo Soto Chaves

-Estudiantes:

José Navarro Acuña

Josué Suárez Campos

-Primer tarea programada:

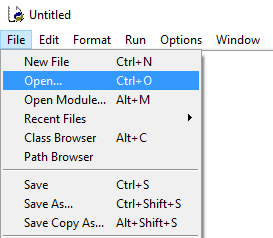
Encriptación clásica y moderna

1er Semestre, 2016.

**Introducción**

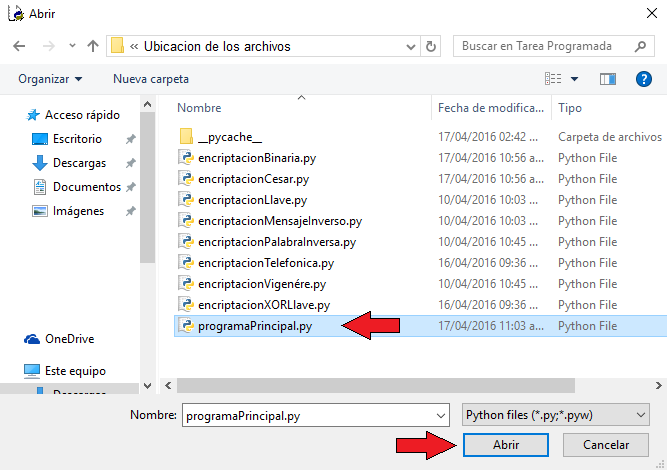
Este software se ha diseñado con el fin de conocer y utilizar los diversos tipos en los que se divide la criptografía clásica y moderna. Seguidamente, para entender mejor la funcionalidad en el momento de utilizar algún algoritmo, es necesario conocer la definición de criptografía, la cual se define como el arte y técnica de escribir con procedimientos o claves secretas de un modo enigmático, de tal forma que lo escrito solamente sea inteligible para quien sepa descifrarlo. Por consiguiente, en el presente software se implementaron las siguientes técnicas de codificación y decodificación del cual el usuario puede hacer uso, entre ellas se encuentran: el cifrado Cesar, el cifrado por llave, sustitución Vigenére, sustitución XOR y llave, palabra inversa, mensaje inverso, cifrado telefónico y por último, el cifrado binario.

Todas las técnicas de codificación y decodificación mencionadas anteriormente, fueron implementadas en el lenguaje de programación, Python. Primeramente, el usuario debe tener instalado en su ordenador dicho software de programación en su versión, únicamente 3.5.1. Una vez dentro del programa, para ejecutar el algoritmo y poder ser utilizado, se debe seleccionar la casilla “*open*” en el apartado “*File*” que se ubica en la posición superior izquierda de la interfaz de Python como se muestra en la figura 1:



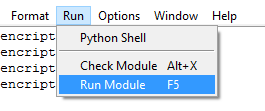
**Figura 1.** Interfaz de Python.

Luego, se desplegará una pequeña ventana, en la cual se debe buscar y seleccionar el archivo llamado “*programaPrincipal.py*” (Ver figura 2).



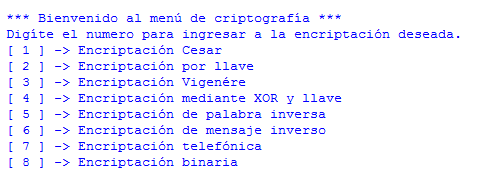
**Figura 2.** Selección del archivo, *“programaPrincipal.py”.*

Una vez realizado lo anterior, para ejecutar el software se debe seleccionar la casilla “*Run Module*” que se encuentra en el apartado “*Run*” de la parte superior de la interfaz de Python como se muestra en la figura 3.



**Figura 3.** Ejecutar el programa.

Finalmente, se desplegará una ventana de consola en la cual, se mostrará el menú de opciones y los respectivos algoritmos. (Ver figura 4). Ahora, solo basta con presionar el número correspondiente a cada encriptación para ejecutar el algoritmo deseado.



**Figura 4.** Ejecución del menú.

**Funcionalidades implementadas del software**

Generalmente, como principal recurso para resolver la codificación y decodificación de todos algoritmos, fue la recursividad, la concatenación de cadenas de caracteres, además del manejo de listas de diferentes maneras (.split(), “ ”.join(), list(), upper(), entre otros) complementándolas con “int” y “strings”. Todo esto efectuado de una muy buena manera, generando estabilidad y continuidad en el software, además se evitó en su totalidad los olores de software y a la vez, evitando inconvenientes en el funcionamiento de las funciones. Lo que se concluye, un estado eficaz en la tarea.

**¿Cómo probar cada algoritmo del software?**

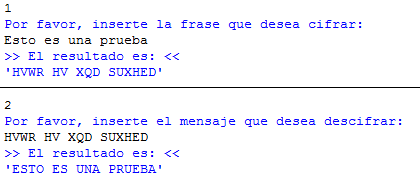
**\*Nota:** Se asume que el usuario ya ha ejecutado el menú.

**\*Nota2:** En caso de digitar un número que no corresponda a ningún algoritmo, se mostrará nuevamente el menú, e igualmente, si el usuario digita un número que no corresponda a codificar (1) o descodificar (2), se retornará el menú nuevamente.

**1) Cifrado Cesar:**

Para su ejecución se debe digitar el número 1, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar.

Para codificar se inserta una frase o un mensaje y en seguida se mostrará su respectivo resultado, luego para descodificar se inserta el resultado obtenido en la codificación (Ver figura 5). En este algoritmo tanto para codificar como para descodificar, únicamente se debe utilizar el alfabeto, excepto la “Ñ”, las vocales tildadas y cualquier otro símbolo (apostrofe, comillas, la coma, el punto, entre otros). En caso contrario, el resultado de codificación y descodificación será incorrecto.

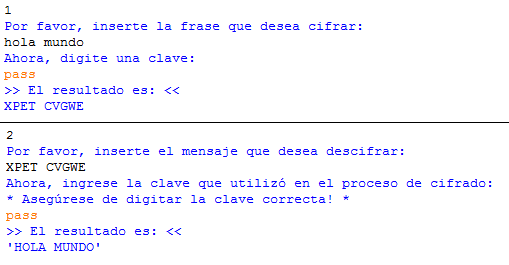


**Figura 5.** Encriptación Cesar: Codificar (1) y descodificar (2).

**2) Cifrado por llave:**

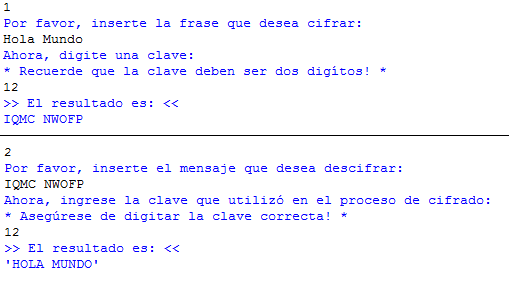
Para su ejecución se debe digitar el número 2, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar. Para codificar, se inserta el mensaje deseado y luego se debe insertar una clave, en el cual la clave estrictamente debe de ser de la misma longitud de la primera palabra del mensaje insertado. Y para la descodificación, se inserta la frase obtenida en la codificación, para luego insertar la misma clave que se utilizó previamente para la codificación. (Ver figura 6). En este algoritmo tanto para codificar como para descodificar, únicamente se debe utilizar el alfabeto, excepto la “Ñ”, las vocales tildadas y cualquier otro símbolo (apostrofe, comillas, la coma, el punto, entre otros). En caso contrario, el resultado de codificación y descodificación será incorrecto.

**Figura 6.** Encriptación por llave: Codificar (1) y descodificar (2).



**3) Cifrado Vigenére:**

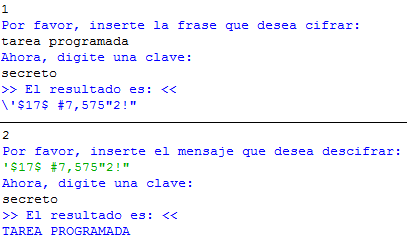
Para utilizar este algoritmo se debe digitar el número 3, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar. Para codificar se debe insertar una frase deseada y luego una clave, dicha clave únicamente debe contener números y preferiblemente una extensión máxima de dos dígitos. Y para descodificar, se inserta la frase obtenida en la codificación previa y se debe utilizar la misma clave (Ver figura 7). En caso de utilizar una contraseña incorrecta, el proceso de codificación o de descodificación mostrará un resultado no deseado. En este algoritmo tanto para codificar como para descodificar, únicamente se debe utilizar el alfabeto, excepto la “Ñ”, las vocales tildadas y cualquier otro símbolo (apostrofe, comillas, la coma, el punto, entre otros). En caso contrario, el resultado de codificación y descodificación será incorrecto.



**Figura 7.** Encriptación Vigenére: Codificar (1) y descodificar (2).

**4) Sustitución XOR y llave:**

Se debe digitar el número 4 para su ejecución, luego se mostrarán dos opciones con su respectivo dígito, 1 para codificar y 2 para descodificar. Para codificar se inserta un mensaje deseado para luego digitar una clave y mostrar el respectivo resultado. Para descodificar se inserta el mensaje obtenido en el proceso de codificación y la misma clave utilizada en dicho proceso (Ver figura 8). Únicamente se debe utilizar el alfabeto, excepto la “Ñ”, las vocales tildadas y cualquier otro símbolo (apostrofe, comillas, la coma, el punto, entre otros.), tanto para el cifrado como para la clave.

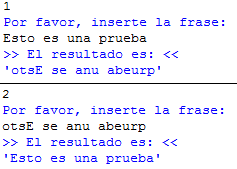


**Figura 9.** Encriptación XOR y llave: Codificar (1) y descodificar (2).

**5) Palabra inversa:**

Se debe digitar el número 5 para su ejecución, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar.

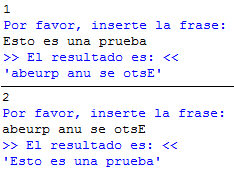
Para codificar se inserta una frase o un mensaje y en seguida se mostrará su respectivo resultado, luego para descodificar se inserta el resultado obtenido en la codificación (Ver figura 9).



**Figura 9.** Palabra inversa: Codificar (1) y descodificar (2).

**6) Mensaje inverso:**

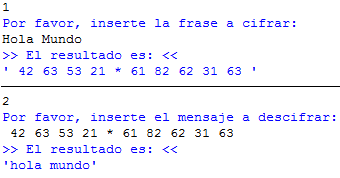
Se debe digitar el número 6 para su ejecución, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar. Para codificar se inserta una frase o un mensaje y luego se mostrará su respectivo resultado y para descodificar, se inserta el resultado obtenido en la codificación (Ver figura 10).



**Figura 10.** Mensaje inverso: Codificar (1) y descodificar (2)

**7) Cifrado telefónico:**

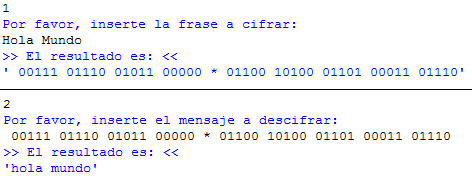
Se debe digitar el número 7 para su ejecución, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar. Para codificar se digita un mensaje deseado, únicamente se debe utilizar el alfabeto, excepto la “Ñ”, las vocales tildadas y cualquier otro símbolo (apostrofe, comillas, la coma, el punto, entre otros.), en caso contrario, se representarán mediante un punto ( **.** ). Para descodificar, se inserta el resultado obtenido en la codificación (Ver figura 11) y de igual manera, si se inserta un número no válido, se representará mediante un punto ( **.** ). Además el espacio está representado mediante un asterisco ( \* ) a la hora de codificar.



**Figura 11.** Encriptación telefónica: Codificar (1) y descodificar (2)

**8) Cifrado binario:**

Se debe digitar el número 8 para su ejecución, luego se mostrarán dos opciones con su respectivo número, 1 para codificar y 2 para descodificar. Para codificar, únicamente se debe utilizar el alfabeto, excepto la “Ñ”, las vocales tildadas y cualquier otro símbolo (apostrofe, comillas, la coma, el punto, entre otros.), en caso contrario, se representarán mediante un punto ( **.** ). Para descodificar, se digita el resultado obtenido en la codificación (Ver figura 12) y de igual manera, si se inserta un número no válido, se representará mediante un punto. Además el espacio está representado mediante un asterisco ( \* ) en el momento de codificar.



**Figura 12.** Encriptación binaria: Codificar (1) y descodificar (2)

**Fuentes consultadas**

Martínez, A. (2013, 13 de junio). Guía Python: Cadenas de texto. *Maestros del Web.* Consultado en <http://www.maestrosdelweb.com/guia-python-cadenas-de-texto/>

Tutorialspoint. (s.f.). Python Lists. Consultado en <http://www.tutorialspoint.com/python/python_strings.htm>

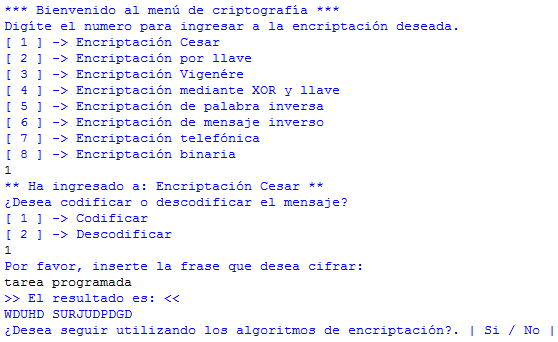
Tutorialspoint. (s.f.). Python Strings. Consultado en <http://www.tutorialspoint.com/python/python_strings.htm>

Tutorialspoint. (s.f.). Python Tuples. Consultado en <http://www.tutorialspoint.com/python/python_tuples.htm>

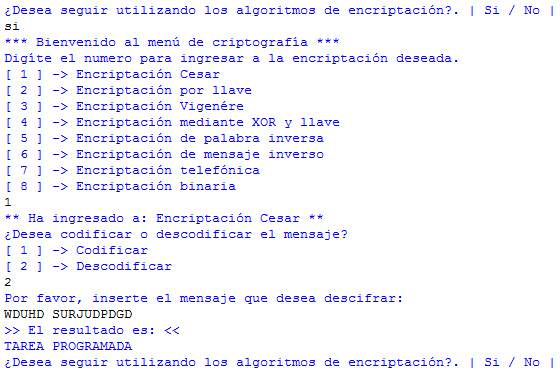
**Anexo B: Pruebas de funcionalidad**

**-Encriptación Cesar.**

Cifrado:

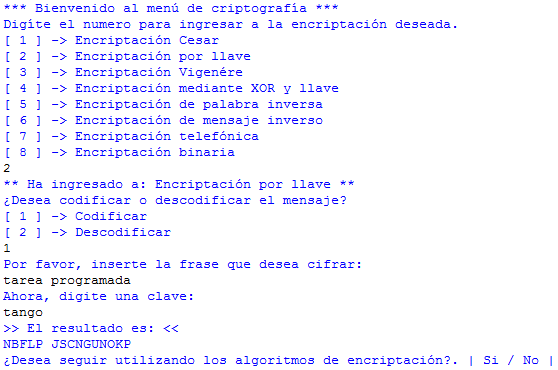


Descifrado:

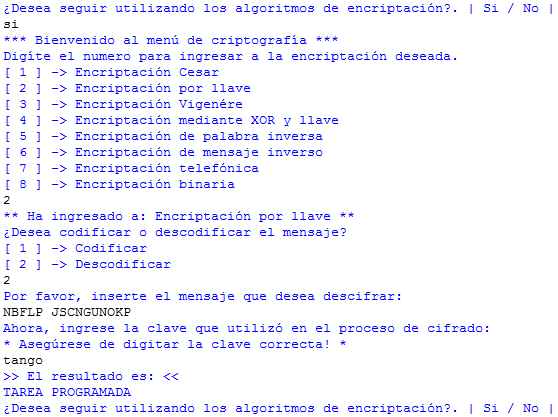


**-Encriptación por llave.**

Cifrado:

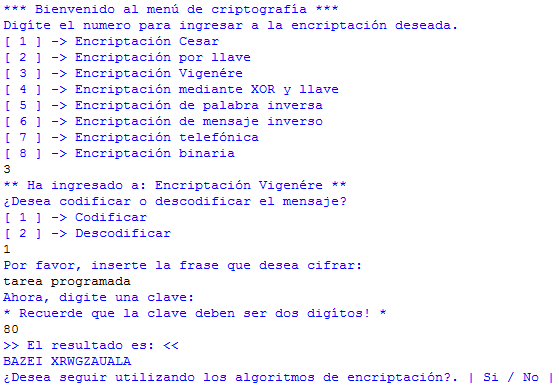


Descifrado:

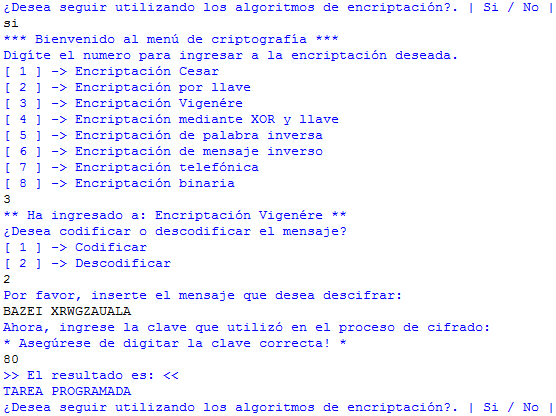


**-Encriptación Vigenére.**

Cifrado:



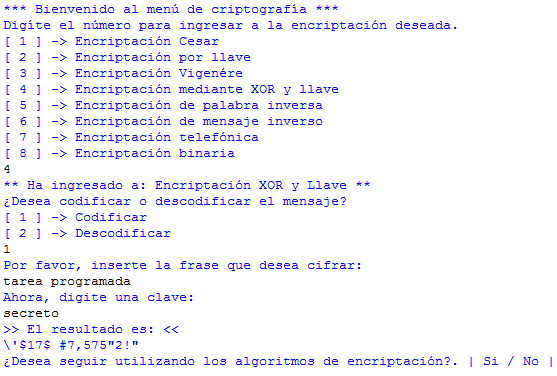
Descifrado:



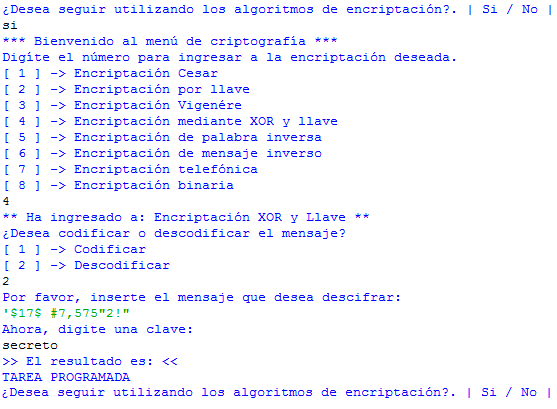
**-Encriptación por XOR y llave.**

Cifrado:

Descifrado:

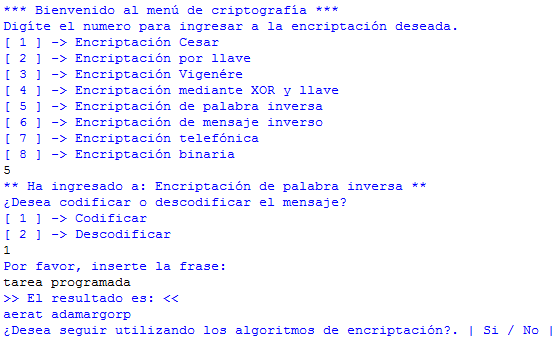


**-**

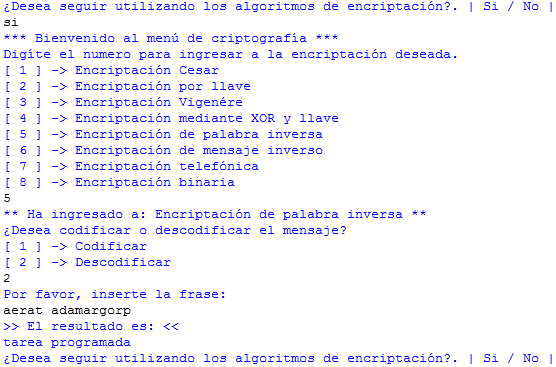


**Encriptación de palabra inversa.**

Cifrado:

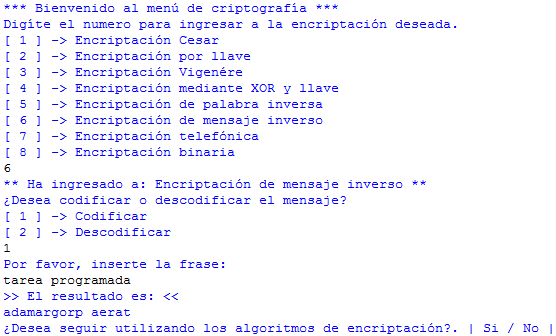


Descifrado:

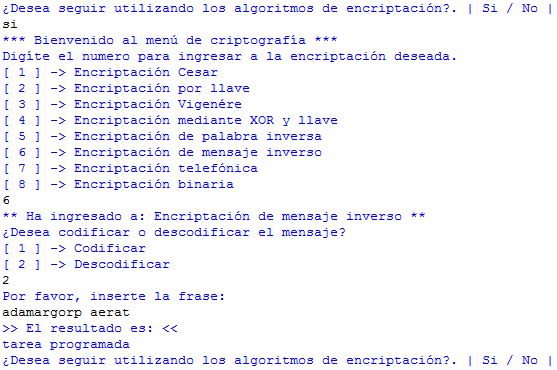


**-Encriptación de mensaje inverso.**

Cifrado:



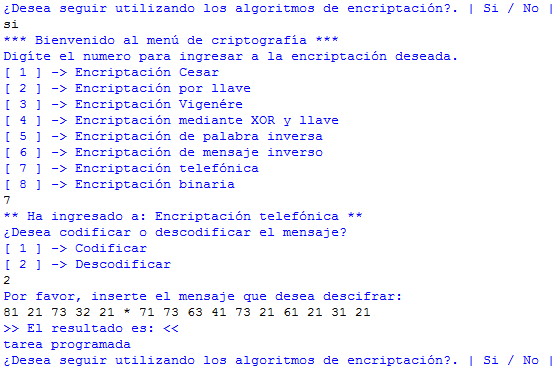
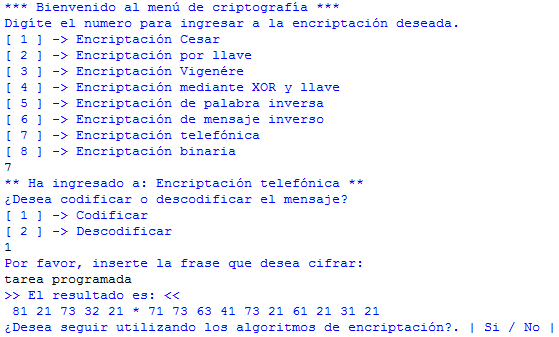
Descifrado:



**-Encriptación telefónica.**

Cifrado:

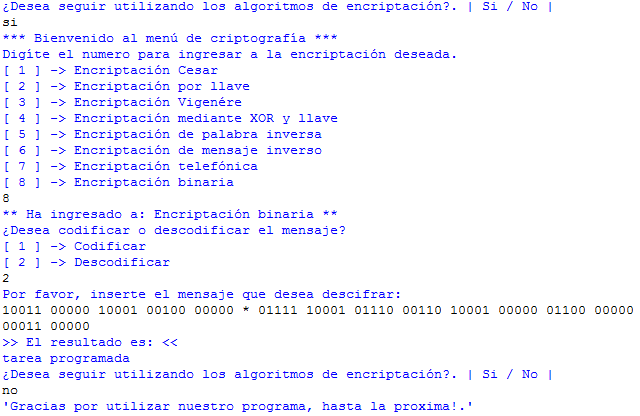
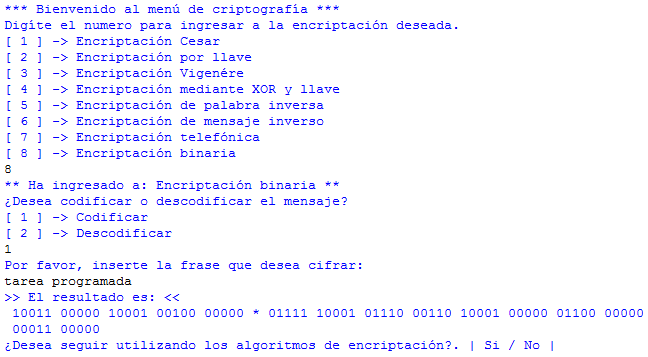
Descifrado:



**Encriptación binaria.**

Cifrado:

Descifrado:

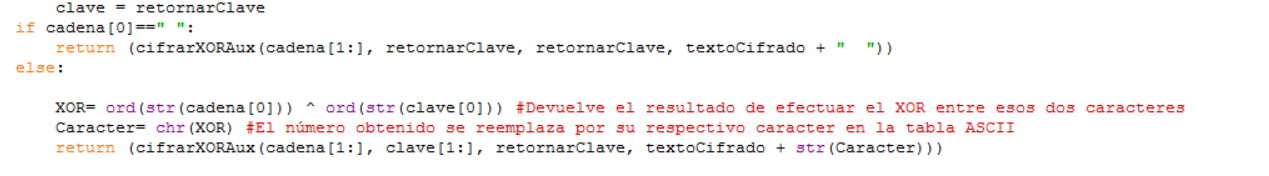


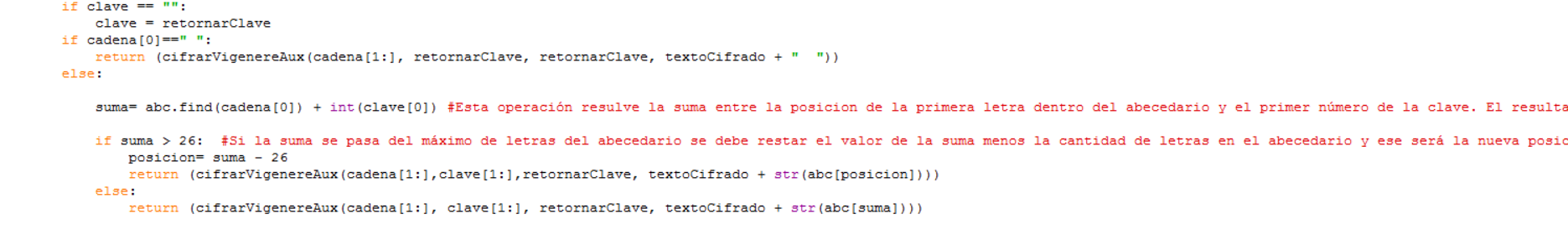
**Anexo C: Lecciones aprendidas.**

* Uso de la función find() para encontrar la posición numérica de un carácter dentro de una cadena.
* Uso de las funciones upper() y lower() para aumentar toda una cadena a letras mayúsculas o minúsculas
* Buscar referencias por internet y las ideas aprendidas utilizarlas en nuestra propia lógica.
* Uso de una variable como un almacén, donde si otra variable se queda sin caracteres pueda tomar caracteres del almacén.
* El uso correcto de las funciones int() y str() y cuando utilizarlas al resolver operaciones y procesos
* Se puede nombrar una cadena de caracteres de manera global donde todas las funciones del programa la utilicen sin estar dentro de la función
* Uso de la función .isalpha() para validación de datos si una cadena es alfabética o no.
* La función len() para averiguar la longitud de una cadena y utilizándola como condición de parada.

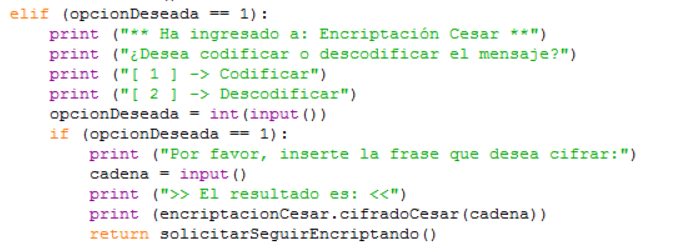
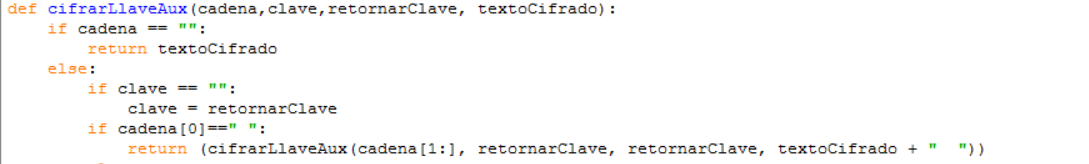
**Anexo D: Olores de software.**

La opacidad es un olor de software, la cual dificulta la lectura del código y su entendimiento. Para evitar este olor de software, en nuestro proyecto utilizamos comentarios para explicar lo que realiza cada operación o proceso.

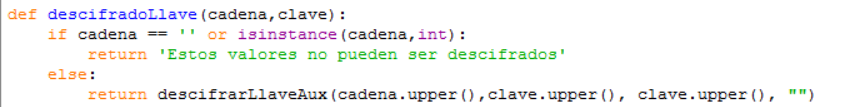




Además de utilizar comentarios para combatir la opacidad como olor de software. Utilizamos nombres significativos a nuestros procesos, variables, métodos y funciones.

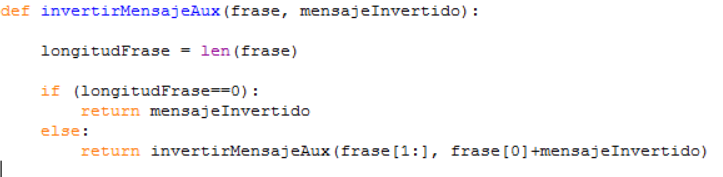
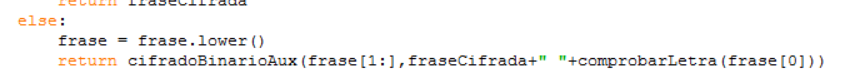


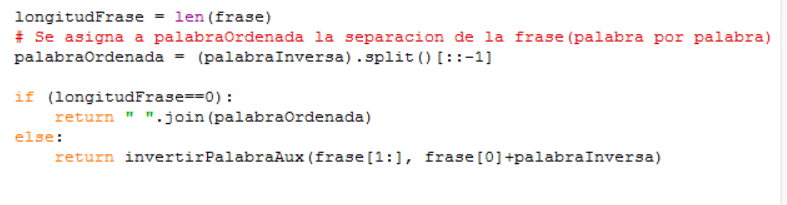
La repetición de métodos innecesarios es aquel olor de software en que líneas de código se pueden escribir en una sola línea.



En este caso, en lugar de hacer una decisión If para cada caso de la línea, utilizamos un solo If para procesar ambas condiciones y anidándolas con el operador OR.

Otra situación evitada es la complejidad innecesaria en el código. En este caso utilizamos funciones predefinidas por Python en lugar de crear nosotros mismos la función y así ahorrarnos tiempo, línea de código y procesamiento de datos.





**Anexo E: Minutas y evidencias de asignación de responsabilidades**

**-Domingo 3 de Abril, (3 horas):**

**Lugar:** Casa de Jose Navarro.

**Temas tratados:** Investigación del manejo y tratamiento de listas en internet y funciones predefinidas en Python. Eso mismo día, nos distribuimos la tarea en partes iguales, o sea 4 encriptaciones cada uno.

**-Martes 5 de Abril, (2 horas):**

**Lugar:** Casa de Josué Suárez

**Temas tratados:** Realizamos la revisión de la encriptación Cesar y la encriptación del mensaje inverso, y empezamos a realizar las encriptaciones por llave y el de mensaje inverso.

**-Sábado 9 de Abril, (3 horas):**

**Lugar:** Casa de José Navarro

**Temas tratados:** Investigación sobre los demás cifrados, para luego realizar una posible implementación en Python de las encriptaciones siguientes, (efectuamos pruebas).

**-Lunes 11 de Abril, (3:30 horas):**

**Lugar:** Casa de Josué Suárez

**Temas tratados:** Realizamos la implementación de la encriptación Vigenére, la encriptación binaria y terminamos de realizar la encriptación por llave y el de mensaje inverso. Luego, nos repartimos las encriptaciones restantes (Encriptación OXR y encriptación telefónica).

**-Jueves 14 de Abril (2:30 horas):**

**Lugar:** Casa de José Navarro

**Temas tratados:** Logramos realizar la encriptación XOR y la encriptación telefónica en los días anteriores, para luego realizar profundas revisiones de todas las encriptaciones y verificar de que todo fuera correcto.

**-Sábado 16 de Abril (2 horas):**

**Lugar:** Casa de José Navarro

**Temas tratados:** Realizamos el “programaPrincipal”, en el cual hicimos un menú que pudiera utilizar todas las encriptaciones creadas.

**-Lunes 18 de Abril (3 horas):**

**Lugar:** Casa de Josué Suarez

**Temas tratados:** Realizamos toda la documentación que implicaba este proyecto (manual de usuario, evidencias de funcionalidad, entre otros), además grabamos en un CD los archivos (documentación y encriptaciones) para la entrega de dicha tarea.