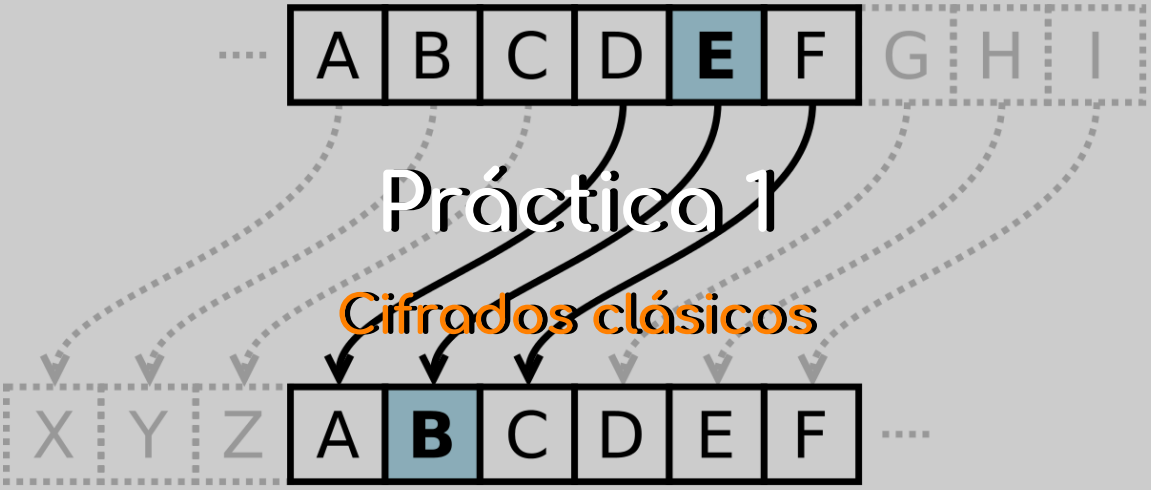

Práctica 1

Cifrados clásicos

Introducción

A tu compa el menos hacker, que le encanta andar descargando videojuegos de internet a través de **Elamigos** fue descuidado y por accidente descargó un *virus* que se activó al momento de ejecutar el *setup.exe*, haciendo que todos los archivos en su carpeta de *Escritorio* fueran **cifrados** y ahora el sistema operativo *no los reconoce*, y como tú estudias en la FC, te ha pedido ayuda para restaurar sus archivos como estaban.

Lo único que sabe tu amigo es que los achivos están cifrados con un *cifrado simétrico*, es decir, es posible que se rompa el cifrado sin conocer la llave original. También recuerda que tenía un *video*, una *canción*, un *PDF* y una *imagen*.

¿Podrías ayudar a tu amigo a recuperar sus archivos? De otra manera tendrá que pagar un rescate de **1,000,000** de bitcoins para poder recuperar sus archivos.

Objetivos

- **Cifrados simétricos**
  Entender cómo funcionan los cifrados clásicos, y programar un descifrador (*al mismo tiempo un cifrador*) de:
  - Cifrado César
  - Cifrado decimado
  - Cifrado afín
  - Base64**
- **Firmas de archivos**
  Conocer los *Magic Bytes* o *File signatures* y cómo es que están relacionados con la extensión de un archivo.
- **Bytes**
  Leer/manipular/escribir bytes en un lenguaje de programación.
- **Base64****
  Programar el algoritmo base64 desde 0. No es un cifrado como tal, es codificacion.

Desarrollo

Para esta práctica necesitarán leer a *nivel de bytes* el contenido de los archivos y hacer el corrimiento de bytes si es necesario. Se recomienda *python* por las funciones predefinidas para extraer **bytes**, listas por compresión, y hacer operaciones de conversión de números en diferentes bases. Sin embargo, **pueden hacerlo en el lenguaje de programación que prefieran** y no pueden usar **ninguna biblioteca**.

Deben desarrollar a grandes rasgos un programa que reciba como entrada un archivo cifrado y pueda rescatar el archivo original a través de la llave correcta. Puedes usar congruencias con los *Magic Bytes* para encontrar las llaves, por fuerza bruta, descarte, o cualquier método que funcione para recuperar los archivos es válido para que tu amigo no tenga que pagar $1,000,000$ de bitcoins. Debemos salvarle su pellejo. **Debes documentar paso a paso el proceso de descifrado de los archivos.**

Hint: Los archivos están en formato *png, pdf, mp4* y *mp3*.

Preguntas

Contesta las siguientes preguntas con tus propias palabras. Serán sometidas a la prueba turing.

1. ¿Cuántos primos relativos hay en $\mathbb{Z}_{256}$?
2. Sea $f : \mathbb{Z}_n \to \mathbb{Z}_n$ tal que $f(i) = k * i$ para $i \in A$, con A un alfabeto cualquiera. ¿qué se debe cumplir para que f sea una función *biyectiva*?
3. ¿Cuántas posibles combinaciones no triviales existen para cifrar bytes con *César*, *Decimado* y *Afin*?
4. ¿Por qué el sistema de archivos de *UNIX*, aunque un archivo tenga una extensión diferente (o incluso no tenga), sigue reconociendo al archivo original?
5. ¿Por qué los archivos descifrados tienen exactamente el mismo tamaño que antes de cifrar, pero no pudimos leerlos? ¿Por qué no tuvimos que agregar/quitar nada?
6. Ya que *base64* no es un cifrado, sino codificación, ¿en qué casos podemos usarlo?

Más por menos

A partir de su *descifrador*, reutilicen el código para crear un *cifrador* de archivos, pasando por banderas o flujo de programa los parámetros para los valores del cifrado, incluyendo **base64**.

Entrega

Para esta entrega, harán un PDF llamado *(PracticaX - NombreEquipo.pdf)* documentando los pasos que consideren importantes. Además el PDF deberá tener la siguiente estructura:

Contenido PDF

- **Portada**: Portada del equipo, con nombres y números de cuenta, además del nombre del equipo.
- **Introducción**: Sobre qué trata la práctica, y una *breve* descripción de la importancia de conocer esta información.
- **Desarrollo**: Explicar de manera *entendible* la forma en la que se realizó la práctica, documentando cada paso ya sea con capturas, salida estándar o con código. También se contestan aquí las preguntas.
- **Conclusiones:** Comentar por equipo, en qué casos es buena idea usar estos cifrados o por qué sería muy mala idea usarlo como candado hoy en día. También pueden poner comentarios de cosas nuevas que aprendieron, trabajo en equipo, en qué se atoraron, retroalimentación de la práctica etc.
- **Referencias:** En caso de utilizar *bibliografia* adicional, citar el link y al autor en caso de ser posible. Debemos dar créditos su autor siempre.

Todo el código que realicen debe ir dentro de una carpeta *src/* compresa en *zip*. No incluir los archivos *cifrados/descifrados* en la entrega, ya que estos serán probados únicamente con el código. El PDF debe ser adjuntado junto a *src/*, **en el classrom** no dentro de la carpeta *src/*.

NOTA: No se aceptan trabajos fuera de la fecha de entrega

Referencias

1. José Galaviz Casas, *Introducción a la criptología* (2010), Notas de clase
2. *Base64 Algorithm Description*, https://medium.com/swlh/base64-encoding-algorithm-42abb929087d
3. *Magic Bytes*, https://en.wikipedia.org/wiki/List_of_file_signatures
4. Spanning Tree, *How to Send a Secret Message.* https://www.youtube.com/watch?v=I6Unxb-PFhs