README
Arroyo Rivera Juan José 416053223

1. *Definición del problema:*
   Crear un programa que encripte un archivo de cualquier extensión con un cifrado simétrico, tal que la llave generada se convierta a un entero que está asociado al término constante de un polinomio aleatorio de grado t-1. Se generará un archivo shares.frg con n las evaluaciones de este polinomio, tal que a partir de una interpolación de Lagrange se pueda reconstruir, obtener su término constante (la llave original) y de esta forma poder desencriptar el archivo encriptado.

2. *Análisis del problema*:
   Se requiere indicar la opción a elegir: cifrar o descifrar y seguir las instrucciones en el README.

   Para ambos casos de forma general se hizo lo siguiente:

   Cifrar:

   Creamos un objeto Cipher, se recibe un password por parte del usuario el cual se convierte a un hash aplicando sha-256 y este hash se convierte a su representación entera. Esta representación se usa como llave para crear un mecanismo de cifrado AES.

   Después se leen los datos del archivo de entrada, y se utiliza el mecanismo de cifrado generado para encriptar los datos y escribirlos en un archivo de extensión <nombre_archivo_original>.aes.

   La llave en representación entera se usa para crear un polinomio aleatorio con el objeto Polynomial, que también recibe los parámetros n, t y p, el cual es el número primo utilizado para hacer aritmética de campos sobre las operaciones dentro del polinomio.

   Se crea el polinomio a partir de t-1 coeficientes aleatorios más la llave entera. Después se generan n evaluaciones aleatorias sobre este polinomio, utilizando numpy.polyval() que internamente utiliza el método de Horner y se escriben en un archivo shares.frg

   Descifrar:

   Para descifrar se leen las shares en forma de tupla en el archivo shares.frg, con estas se hace la interpolación de Lagrange con el objeto LagrangeInterpolation donde se calcula cada polinomio de Lagrange de base i evaluado en x= 0, por cada punto x_i, y se multiplica por su evaluación en el polinomio hecha con

numpy.polyval(), al final estos productos se suman y se regresa el resultado módulo p, el cual corresponde al término constante del polinomio.

Con este la llave (término constante) recuperada más se crea un mecanismo de descifrado AES, y los datos descifrados se escriben en un archivo de nombre igual al archivo de entrada original

3. *Selección de la mejor alternativa:*

   Se eligió la biblioteca pycryptodome y Crypto para generar los mecanismos de cifrado/descifrado y llaves utilizados.

   Se utilizó el número primo dado por el profesor para definir el campo en el que se llevaran las operaciones del cálculo del polinomio de Lagrange

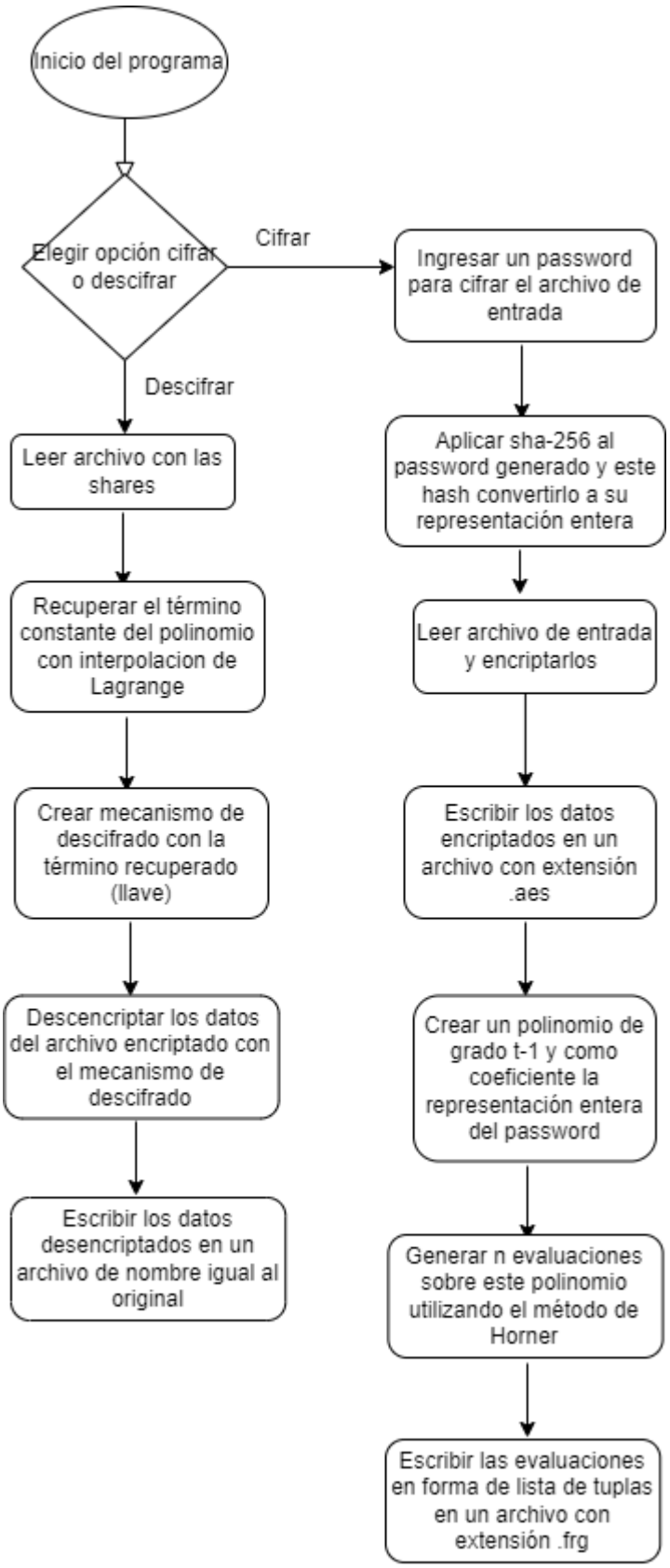
   Para el caso de la evaluación del polinomio se utilizó la biblioteca numpy para hacer uso del método polyval pues con algunas otras implementaciones de Horner que probé había imprecisiones y con una implementación usando potencias puede haber problemas de eficiencia por lo grande que pueden ser los números

   Para el caso de implementar la división en un Campo, usé la biblioteca egcd que calcula el maximo comun divisor entre dos números, lo cual se utiliza para encontrar el inverso multiplicativo para obtener x/y módulo p

4. *Mantenimiento futuro*:

   Quizás se podría añadir un poco más de interfaz de usuario para no depender tanto de saber la forma de ingresar los argumentos.

5. *Diagrama de flujo*:


Inicio del programa
Elegir opción cifrar o descifrar
Cifrar
Descifrar
Ingresar un password para cifrar el archivo de entrada
Aplicar sha-256 al password generado y este hash convertirlo a su representación entera
Leer archivo de entrada y encriptarlos
Escribir los datos encriptados en un archivo con extensión .aes
Crear un polinomio de grado t-1 y como coeficiente la representación entera del password
Generar n evaluaciones sobre este polinomio utilizando el método de Horner
Escribir las evaluaciones en forma de lista de tuplas en un archivo con extensión .frg
Leer archivo con las shares
Recuperar el término constante del polinomio con interpolacion de Lagrange
Crear mecanismo de descifrado con la término recuperado (llave)
Descencriptar los datos del archivo encriptado con el mecanismo de descifrado
Escribir los datos desencriptados en un archivo de nombre igual al original