# CÓMPUTO FORENSE

## PRÁCTICA

PEDRO OCTAVIO CULEBRO PRADO
JOSÉ MARIA PADILLA FERNANDEZ

UNIVERSIDAD AUTÓNOMA DE CHIAPAS


*FACULTAD DE NEGOCIOS - CAMPUS IV*

*INGENIERÍA EN DESARROLLO Y TECNOLOGÍAS DE SOFTWARE*
*SÉPTIMO SEMESTRE "D"*

*MATERIA: CÓMPUTO FORENSE*
*IMPARTIDA POR: RENÉ SERVANDO RIVERA ROBLERO*

# *PRÁCTICA: GENERACIÓN Y VALIDACIÓN DE FIRMA ELECTRÓNICA*

*REALIZADO POR:*
*PEDRO OCTAVIO CULEBRO PRADO*
*JOSÉ MARIA PADILLA FERNANDEZ*
*MATRÍCULAS: B200227, B200003*

## Resumen:

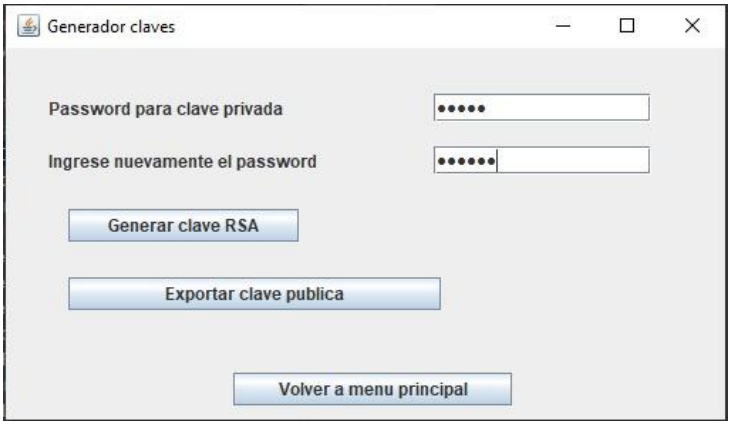
El programa implementa un sistema de seguridad asimétrica utilizando Java, donde se generan pares de claves públicas y privadas mediante un algoritmo establecido para garantizar la confidencialidad y autenticación. La clave privada se emplea para crear una firma digital sobre un documento específico, mientras que la clave pública se utiliza para verificar la autenticidad de la firma y confirmar que coincide con el documento original. Se aplican algoritmos de protección para asegurar la integridad y seguridad de la información, siguiendo estándares de seguridad en la generación y validación de firmas digitales.

## Introducción:

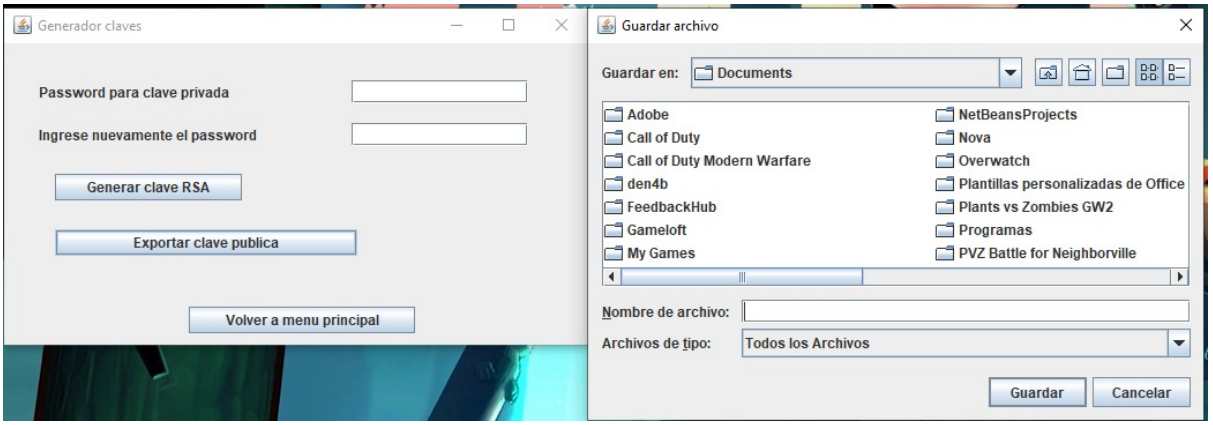
La firma electrónica, también conocida como eFirma, es una técnica ampliamente utilizada para garantizar la autenticidad e integridad de los datos en el mundo digital. Este informe se enfoca en la implementación de un programa en Java que tiene dos componentes principales: uno para generar una eFirma basada en una contraseña y otro para leer y validar dicha firma.

## Generación de Clave Pública

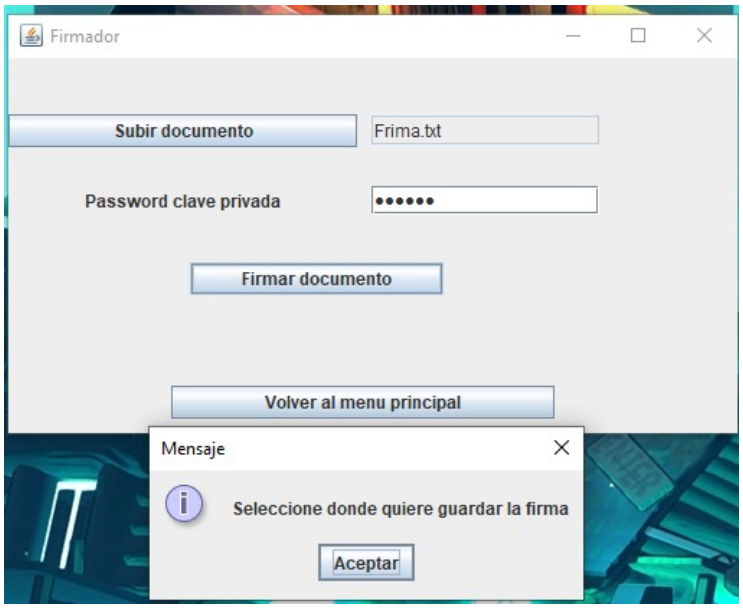
En esta ventana pedirá al usuario ingresar una contraseña para asignarla a la clave pública que después usaremos para validar y firmar algún documento.







# Generación de eFirma




**Descripción General**
El primer componente de nuestro programa se encarga de generar una eFirma basada en una contraseña proporcionada por el usuario. Esta eFirma se utiliza para asegurar que los datos no han sido alterados durante la transmisión o almacenamiento.

Acá nos pedirá que subamos el documento que querramos darle una eFirma, posteriormente nos pedira una contraseña la cual ingresamos en el paso anterior.

Una vez registrado esos datos, le daremos a firmar documento para generar un archivo con la eFirma propia generada automaticamente.

**Algoritmo de Generación**
Para lograr esto, se utiliza un algoritmo de generación de firma que involucra los siguientes pasos:

*Obtención de la Contraseña: El programa solicita al usuario que ingrese su contraseña.*

*Hashing de la Contraseña: La contraseña ingresada se pasa a través de una función de hash segura para generar un valor único e irreversible.*

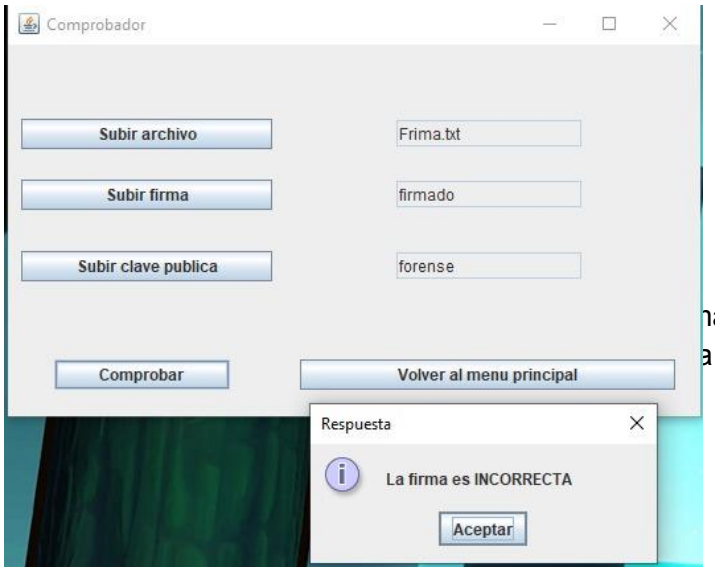
*Firma con Clave Privada: La contraseña hash se utiliza para firmar digitalmente los datos utilizando una clave privada almacenada de manera segura.*

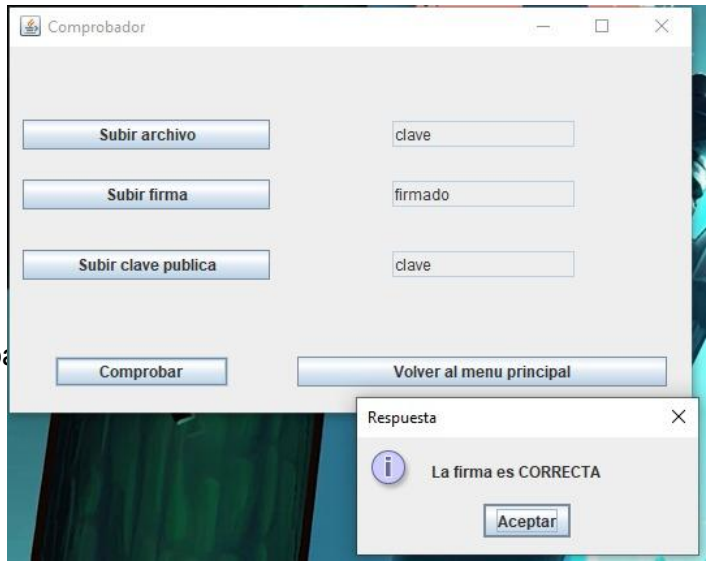
*Generación de eFirma: La firma digital resultante se almacena como la eFirma generada.*

## Seguridad

Es esencial destacar que la seguridad de la eFirma depende en gran medida de la seguridad de la contraseña y de la protección de la clave privada. Se deben tomar medidas adicionales para garantizar la confidencialidad de la clave privada y la complejidad de la contraseña.

Para ello se piden 3 datos para comprobar la veracidad de la eFirma de algún documento. El primero subir el archivo que firmamos, como segundo parámetro es la eFirma que se generó anteriormente y por último nos pide la clave pública que generamos en el primer paso. De ser todo correcto nos marcará si la firma es válida o no.




**Algoritmo de Validación**
El proceso de validación implica los siguientes pasos:

*Obtención de la eFirma y Datos Originales: El programa solicita al usuario la eFirma y los datos originales que se firmaron previamente.*

*Verificación de la Firma: Utilizando la clave pública correspondiente, el programa verifica la autenticidad de la firma digital.*

*Hashing de los Datos Originales: Los datos originales se pasan a través de la misma función de hash utilizada en la generación de la firma.*

*Comparación de Hashes: Se compara el hash de los datos originales con el hash almacenado en la eFirma. Si coinciden, la firma se considera válida.*

## Importancia de la Validación

La validación de la eFirma es fundamental para asegurarse de que los datos no han sido manipulados durante la transmisión o el almacenamiento. Garantiza la integridad de la información y la autenticidad del remitente.

## Conclusiones:

El programa Java desarrollado para la generación y validación de firmas electrónicas es una herramienta esencial para garantizar la seguridad de la comunicación electrónica. Proporciona una forma eficaz de verificar la autenticidad e integridad de los mensajes transmitidos, lo que es crucial en entornos donde la seguridad de la información es primordial.

Este sistema ofrece una interfaz de usuario amigable que permite a los usuarios sin experiencia técnica utilizarlo de manera efectiva. Además, la implementación de prácticas de seguridad sólidas garantiza la protección de las claves privadas y la confidencialidad de los mensajes.

En resumen, el programa en Java para generar y validar firmas electrónicas es una herramienta valiosa para la seguridad de la información en la era digital y se recomienda su implementación en entornos donde la seguridad de la comunicación es esencial.

## Referencias:

[1] Documentación de Java Cryptography Architecture. Oracle. Disponible en: https://docs.oracle.com/en/java/javase/11/security/intro-secure-communication.html