

Manual del Programador

Cifrado Chase

Shunya Zhan

Lunes, 14 de octubre de 2024



ÍNDICE

1. Introducción
2. Diseño del Sistema
3. Descripción de componentes
4. Fuentes
5. Conclusión
6. Anexo



Introducción

El Cifrado de Chase es un criptosistema de clave simétrica que destaca por su facilidad de uso y seguridad. Este sistema representa las letras del alfabeto mediante coordenadas numéricas, lo que lo clasifica como un cifrado fraccionado o tomográfico. La fortaleza del Cifrado de Chase radica en su capacidad para representar cada letra con dos números, aumentando así la complejidad y seguridad del cifrado.

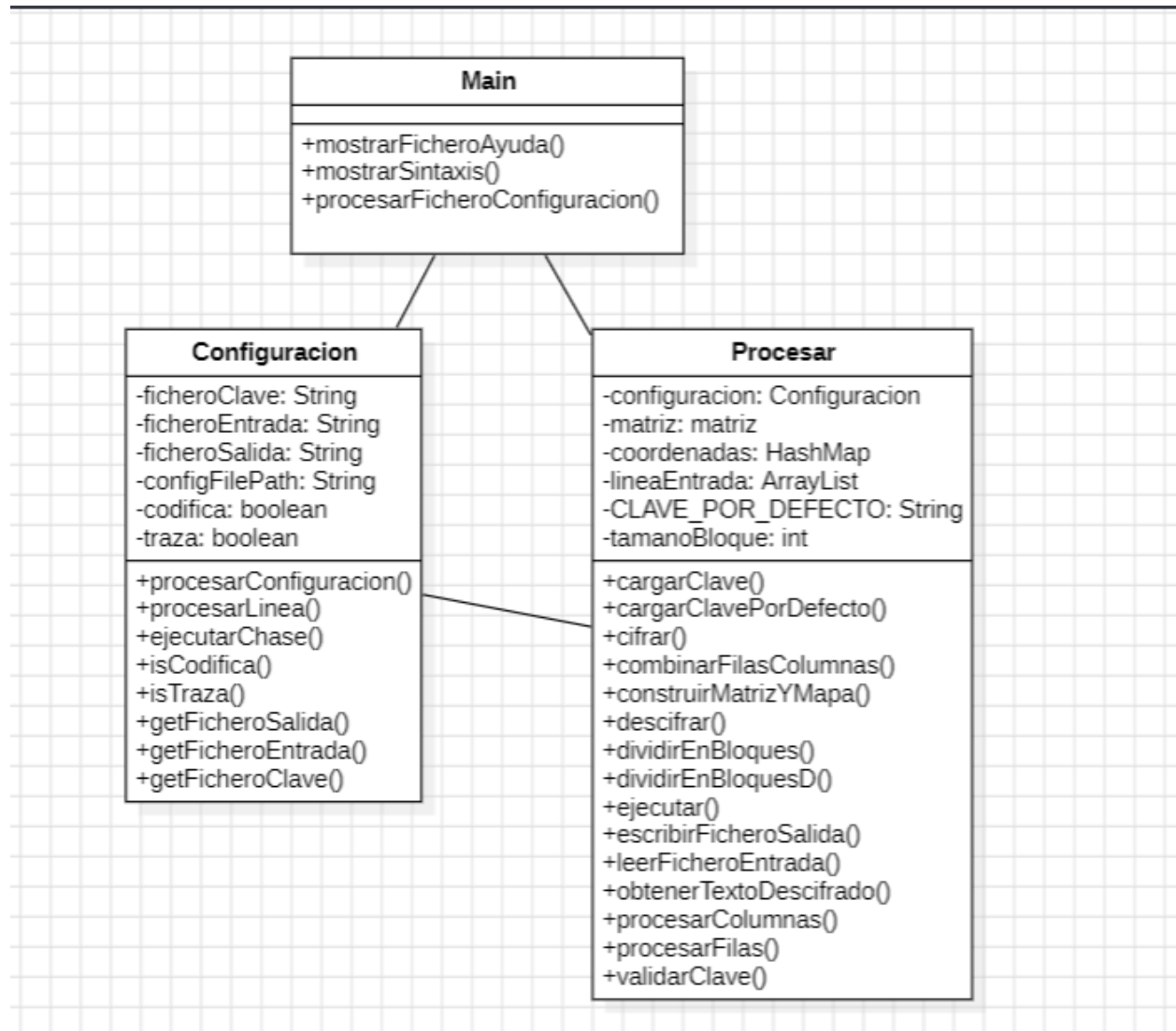
Esta práctica tiene como objetivo implementar el Cifrado de Chase en un programa escrito en Java. A través de esta implementación, aprenderán a manejar la entrada y salida de información mediante ficheros TXT, desarrollar un criptosistema de clave simétrica y aplicar técnicas de cifrado en bloque. La práctica es modular e incremental, permitiendo añadir distintos algoritmos de cifrado conforme se avanza en la parte teórica de la asignatura.

El programa tomará los datos de entrada desde ficheros especificados en los parámetros del programa y generará los resultados en ficheros de salida. Además, se incluirá una funcionalidad de traza para informar en pantalla de los procesos que se están llevando a cabo, facilitando así la depuración y comprensión del funcionamiento del programa.

La clave utilizada en el Cifrado de Chase se formará mediante una matriz de 3×10 , donde cada columna se rellenará con tres caracteres según un orden especificado en la clave. Esta clave también incluirá un valor que indica el tamaño del bloque para el cifrado y descifrado, asegurando que tanto el emisor como el receptor están sincronizados en el proceso de cifrado.

En resumen, esta práctica no solo permite aplicar conceptos teóricos de seguridad de la información, sino también desarrollar habilidades prácticas en programación y manejo de datos.

Diseño del Sistema



Descripción de componentes

Clase Main:

La clase Main es la clase principal que inicia la ejecución del programa. Su función principal es procesar los argumentos de la línea de comandos y delegar las tareas a las clases Configuración y Procesar

- Atributos: No tiene atributos
- Métodos:
 1. `public static void main(String[] args)`: Método principal que inicia el programa y procesa los argumentos de la línea de comandos.
 2. `private static void mostrarFicheroAyuda(String filePath)`: Muestra el contenido del fichero de ayuda.
 3. `private static void procesarFicheroConfiguracion(String configFilePath)`: Procesa el fichero de configuración y ejecuta el proceso de cifrado o descifrado.
 4. `private static void mostrarSintaxis()`: Muestra la sintaxis correcta del programa.

Clase Procesar:

La clase Procesar se encarga de realizar el cifrado y descifrado de los textos. Utiliza la configuración proporcionada por la clase Configuración para leer los ficheros de entrada, procesar los datos y escribir los resultados en los ficheros de salida.

- Atributos:
 1. `private Configuracion configuracion`: Objeto que contiene la configuración del programa.
 2. `private char[][] matriz`: Matriz de caracteres utilizada para almacenar la clave de cifrado.
 3. `private Map<Character, String> coordenadas`: Mapa que asocia cada carácter con sus coordenadas en la matriz.
 4. `private List<String> lineasEntrada`: Lista que contiene las líneas de texto leídas del fichero de entrada.
 5. `private static final String CLAVE_POR_DEFECTO`: Clave por defecto utilizada si no se proporciona una clave específica.
 6. `private int tamanoBloque`: Tamaño del bloque utilizado para el cifrado y descifrado.
- Métodos:




1. `public Procesar(Configuracion configuracion)`: Constructor que recibe un objeto `Configuracion`.
2. `public void ejecutar()`: Ejecuta el proceso de lectura, cifrado/descifrado y escritura de archivos.
3. `private void cargarClave(String claveFilePath)`: Carga la clave desde un fichero o utiliza la clave por defecto.
4. `private void cargarClavePorDefecto()`: Carga la clave por defecto.
5. `private void construirMatrizYMapa(String clave)`: Construye la matriz y el mapa de coordenadas a partir de una clave proporcionada.
6. `private void leerFicheroEntrada(String entradaFilePath)`: Lee el fichero de entrada y almacena las líneas en `lineasEntrada`.
7. `private void escribirFicheroSalida(String salidaFilePath)`: Escribe las líneas procesadas en un fichero de salida.
8. `private void cifrar()`: Cifra las líneas de texto de `lineasEntrada`.
9. `private void descifrar()`: Descifra las líneas de texto de `lineasEntrada`.
10. `private List<String> dividirEnBloques(String coordenadasTexto, int tamanoBloque)`: Divide una cadena de texto en bloques de tamaño especificado.
11. `private static StringBuilder obtenerTextoDescifrado(String descifradoTexto, char[][] matriz)`: Convierte una cadena de texto descifrada en el texto original utilizando la matriz de clave.
12. `private static StringBuilder combinarFilasColumnas(List<String> filas, List<Long> columnas, int tamanoBloque)`: Combina las filas y columnas procesadas en una cadena de texto descifrada.
13. `private static List<String> procesarFilas(List<String> bloquesFilas)`: Procesa una lista de bloques de filas eliminando el primer dígito de cada bloque.
14. `private static List<Long> procesarColumnas(List<String> bloquesColumnas)`: Procesa una lista de bloques de columnas dividiendo cada número por 9.
15. `private static List<String> dividirEnBloquesD(String texto, int tamanoBloque)`: Divide una cadena de texto en bloques de tamaño especificado.

Clase Configuración:

La clase `Configuracion` se encarga de leer y procesar el fichero de configuración. Configura los atributos del programa según las instrucciones especificadas en el fichero de configuración.

- Atributos:

1. `private String configFilePath`: Ruta del fichero de configuración.
2. `private boolean codifica`: Indica si el modo de codificación está activado.

- 
3. `private boolean traza`: Indica si el modo de traza está activado.
 4. `private String ficheroEntrada`: Ruta del fichero de entrada.
 5. `private String ficheroSalida`: Ruta del fichero de salida.
 6. `private String ficheroClave`: Ruta del fichero de clave.
- Métodos:
1. `public Configuracion(String configFilePath)`: Constructor que recibe la ruta del fichero de configuración.
 2. `public void procesarConfiguracion()`: Lee y procesa el fichero de configuración línea por línea.
 3. `private void procesarLinea(String line)`: Procesa una línea del fichero de configuración.
 4. `private void ejecutarChase()`: Crea una instancia de la clase Procesar y ejecuta su método ejecutar.
 5. `public boolean isCodifica()`: Devuelve el estado del modo de codificación.
 6. `public boolean isTraza()`: Devuelve el estado del modo de traza.
 7. `public String getFicheroEntrada()`: Devuelve la ruta del fichero de entrada.
 8. `public String getFicheroSalida()`: Devuelve la ruta del fichero de salida.
 9. `public String getFicheroClave()`: Devuelve la ruta del fichero de clave.



Fuentes

1. Campus Virtual (Seguridad de información)
2. Google

Conclusión

El proyecto de implementación del Cifrado de Chase ha sido una experiencia integral que ha permitido aplicar y consolidar conocimientos teóricos y prácticos en el ámbito de la seguridad de la información. A continuación, se presenta un resumen de los aspectos más destacados del proyecto:

Objetivos

Aprender el manejo básico del entorno de programación: Utilizando Java para desarrollar un criptosistema.

Estudiar la entrada/salida de información: A través de ficheros TXT.

Desarrollar un criptosistema de clave simétrica: Implementando el Cifrado de Chase.

Cifrar y descifrar datos: Aplicando técnicas de cifrado en bloque.

Componentes Principales

Clase Main:

- Inicia la ejecución del programa.
- Procesa los argumentos de la línea de comandos.
- Delegar tareas a las clases Configuración y Procesar.

Clase Procesar:

- Realiza el cifrado y descifrado de textos.
- Gestiona la lectura de ficheros de entrada y la escritura de ficheros de salida.
- Utiliza una matriz de caracteres (`char[][]`) para almacenar la clave de cifrado.
- Implementa métodos para dividir el texto en bloques, procesar filas y columnas, y combinar coordenadas.

Clase Configuración:

- Lee y procesa el fichero de configuración.
- Configura los atributos del programa según las instrucciones especificadas.
- Gestiona las rutas de los ficheros de entrada, salida y clave.
- Proceso de Cifrado y Descifrado


Cifrado:

- Sustituir caracteres especiales y espacios en blanco.
- Obtener coordenadas de cada letra y dividir el texto en bloques.
- Modificar las coordenadas y combinar filas y columnas para formar el texto cifrado.

Descifrado:

- Convertir caracteres en sus coordenadas correspondientes.
- Separar las coordenadas en filas y columnas.
- Restaurar las coordenadas originales y convertirlas en texto descifrado.



Anexo

Dentro del ZIP, en la carpeta Documentación:

- JavaDoc
- Diagrama de clase
- Prueba de ejecución

