

**Alumno:** Kevin Zamora Amela  
**Tarea:** PSP07  
**Asignatura:** Programación de Servicios y Procesos

## Ejercicio 1.

De igual manera a lo visto en el tema, ahora te proponemos un ejercicio que genere una cadena de texto y la deje almacenada en un fichero encriptado, en la raíz del proyecto hayas creado, con el nombre fichero.cifrado.

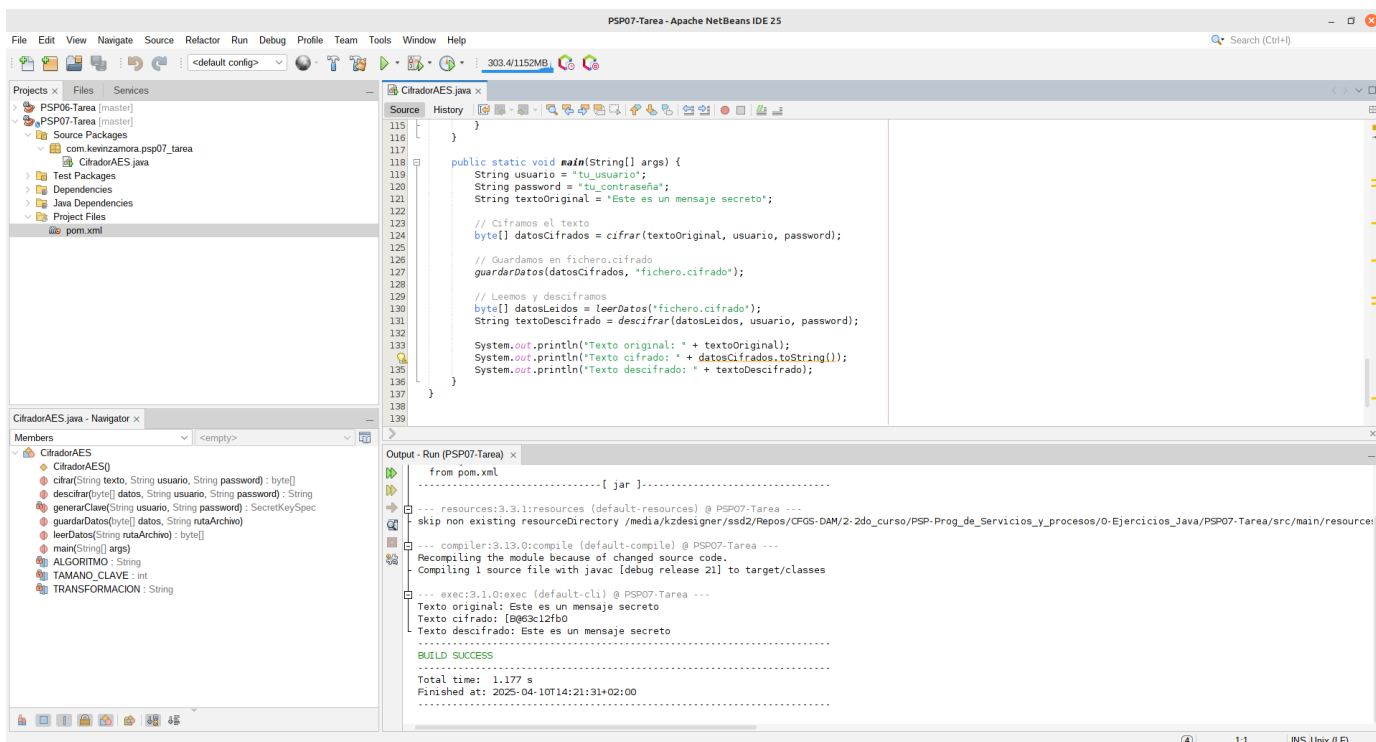
Para encriptar el fichero, utilizarás el algoritmo Rijndael o AES, con las especificaciones de modo y relleno siguientes: Rijndael/ECB/PKCS5Padding.

La clave, la debes generar de la siguiente forma:

- A partir de un número aleatorio con semilla la cadena del nombre de usuario + password.
- Con una longitud o tamaño 128 bits.

Para probar el funcionamiento, el mismo programa debe acceder al fichero encriptado para desencriptarlo e imprimir su contenido.

## Respuesta:



The screenshot shows an IDE window titled "PSP07-Tarea - Apache NetBeans IDE 25". The main editor displays the source code for "CifradorAES.java". The code implements a simple encryption and decryption process using AES. It defines a main method that takes command-line arguments for a user, a password, and a text to be encrypted. The text is encrypted using Rijndael/ECB/PKCS5Padding and saved to a file named "fichero.cifrado". The file is then read back and decrypted to show the original text.

```
115 }
116
117
118 public static void main(String[] args) {
119     String usuario = "tu_usuario";
120     String password = "tu_contraseña";
121     String textoOriginal = "Este es un mensaje secreto";
122
123     // Ciframos el texto
124     byte[] datosCifrados = cifrar(textoOriginal, usuario, password);
125
126     // Guardamos en fichero.cifrado
127     guardarDatos(datosCifrados, "fichero.cifrado");
128
129     // Leemos y desciframos
130     byte[] datosLeidos = leerDatos("fichero.cifrado");
131     String textoDescifrado = descifrar(datosLeidos, usuario, password);
132
133     System.out.println("Texto original: " + textoOriginal);
134     System.out.println("Texto cifrado: " + datosCifrados.toString());
135     System.out.println("Texto descifrado: " + textoDescifrado);
136 }
137
138
139 }
```

The left sidebar shows the project structure with "CifradorAES.java" selected. The bottom output window shows the execution results, including the compilation and execution of the program, and the printed output of the encryption and decryption process.

```
from pom.xml
[ jar ]-----
--- resources:3.1:resources (default-resources) @ PSP07-Tarea ---
skip non existing resourceDirectory /media/kzdesigner/ssd2/Repos/CFGS-DAM/2-2do_curso/PSP-Prog_de_Servicios_y_procesos/0-Ejercicios_Java/PSP07-Tarea/src/main/resources
--- compiler:3.13.0:compile (default-compile) @ PSP07-Tarea ---
Recompiling the module because of changed source code.
Compiling 1 source file with javac [debug release 21] to target/classes
--- exec:3.1.0:exec (default-cli) @ PSP07-Tarea ---
Texto original: Este es un mensaje secreto
Texto cifrado: [0069c12fb0
Texto descifrado: Este es un mensaje secreto
BUILD SUCCESS
Total time: 1.177 s
Finished at: 2025-04-10T14:21:31+02:00
```

## Validación del funcionamiento requerido:

```
Output - Run (PSP07-Tarea) x
from pom.xml
-----[ jar ]-----
--- resources:3.3.1:resources (default-resources) @ PSP07-Tarea ---
skip non existing resourceDirectory /media/kzdesigner/ssd2/Repos/CFGS-DAM/2-2do_curso/PSP-Prog_de_Servicios_y_procesos/0-Ejercicios_Java/PSP07-Tarea/src/main/resources
--- compiler:3.13.0:compile (default-compile) @ PSP07-Tarea ---
Recompiling the module because of changed source code.
Compiling 1 source file with javac [debug release 21] to target/classes
--- exec:3.1.0:exec (default-cli) @ PSP07-Tarea ---
Texto original: Este es un mensaje secreto
Texto cifrado: [B@63c12fb0
Texto descifrado: Este es un mensaje secreto
BUILD SUCCESS
Total time: 1.177 s
Finished at: 2025-04-10T14:21:31+02:00
```

## Programa/Aplicación implementado/a:

```
/**
 * Clase que maneja las operaciones de cifrado/descifrado usando AES
 */
public class CifradorAES {

    private static final String ALGORITMO = "AES";
    private static final String TRANSFORMACION = "AES/ECB/PKCS5Padding";
    private static final int TAMANO_CLAVE = 128; // bits

    public static void main(String[] args) {
        String usuario = "tu_usuario";
        String password = "tu_contraseña";
        String textoOriginal = "Este es un mensaje secreto";

        // Ciframos el texto
        byte[] datosCifrados = cifrar(textoOriginal, usuario, password);

        // Guardamos en fichero.cifrado
        guardarDatos(datosCifrados, "fichero.cifrado");

        // Leemos y desciframos
        byte[] datosLeidos = leerDatos("fichero.cifrado");
        String textoDescifrado = descifrar(datosLeidos, usuario, password);

        System.out.println("Texto original: " + textoOriginal);
        System.out.println("Texto cifrado: " + datosCifrados.toString());
        System.out.println("Texto descifrado: " + textoDescifrado);
    }
}
```

## Importaciones de los paquetes o bibliotecas necesarias:

```
package com.kevinzamora.psp07_tarea;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
```

← Métodos de entrada y salida

← Método para generar “clave” fija

← Bibliotecas de métodos de cifrado

## Métodos implementados:

```
/**
 * Cifra un texto usando AES
 * @param texto Texto a cifrar
 * @param usuario Usuario para generar la clave
 * @param password Password para generar la clave
 * @return Datos cifrados
 */
public static byte[] cifrar(String texto, String usuario, String password) {
    try {
        SecretKeySpec clave = generarClave(usuario, password);
        Cipher cipher = Cipher.getInstance(TRANSFORMACION);
        cipher.init(Cipher.ENCRYPT_MODE, clave);

        return cipher.doFinal(texto.getBytes(StandardCharsets.UTF_8));
    } catch (Exception e) {
        throw new RuntimeException("Error al cifrar", e);
    }
}

/**
 * Descifra datos previamente cifrados
 * @param datos Datos cifrados
 * @param usuario Usuario usado para generar la clave
 * @param password Password usada para generar la clave
 * @return Texto descifrado
 */
public static String descifrar(byte[] datos, String usuario, String password) {
    try {
        SecretKeySpec clave = generarClave(usuario, password);
        Cipher cipher = Cipher.getInstance(TRANSFORMACION);
        cipher.init(Cipher.DECRYPT_MODE, clave);

        return new String(cipher.doFinal(datos), StandardCharsets.UTF_8);
    } catch (Exception e) {
        // throw new RuntimeException("Error al descifrar", e);
        e.printStackTrace(); // Imprime el error original
        throw new RuntimeException("Error al descifrar", e);
    }
}

/**
 * Guarda datos cifrados en un archivo
 * @param datos Datos cifrados
 * @param rutaArchivo Ruta del archivo
 */
public static void guardarDatos(byte[] datos, String rutaArchivo) {
    try (FileOutputStream fos = new FileOutputStream(rutaArchivo)) {
        fos.write(datos);
    } catch (IOException e) {
        throw new RuntimeException("Error al guardar el archivo", e);
    }
}

/**
 * Lee datos cifrados desde un archivo
 * @param rutaArchivo Ruta del archivo
 * @return Datos leídos
 */
public static byte[] leerDatos(String rutaArchivo) {
    try (FileInputStream fis = new FileInputStream(rutaArchivo)) {
        byte[] datos = fis.readAllBytes();
        return datos;
    } catch (IOException e) {
        throw new RuntimeException("Error al leer el archivo", e);
    }
}
```

**- Comentarios relacionados con el proceso de generación de la documentación con JavaDoc de ambos proyectos/ejercicios:**

- Primeramente y como ya deberíamos saber, para ejecutar la herramienta JavaDoc y proceder a generar la documentación de nuestro proyecto, debemos hacer clic derecho sobre el directorio raíz de nuestro proyecto con java y seleccionar la opción “Generar JavaDoc/Generate JavaDoc”, según tengamos en Español o en Inglés, respectivamente, la interfaz de nuestro IDE Netbeans.
- Al ejecutar la citada opción “Generar JavaDoc”, nos aparecerá/apareció un mensaje de error comentándonos que nuestra versión es demasiado antigua (siendo la 5 y en consecuencia, siendo anterior a la 8) y por ende, la herramienta de JavaDoc no se encuentra disponible ni resulta compatible.
- Para solucionarlo, simplemente hemos creado un nuevo proyecto de Java 21 (como versión instalada en nuestro sistema) con Maven y hemos “migrado”(o copiado y pegado) las clases de nuestros programas, a dos proyectos de Java nuevos e independientes. Al ejecutar la opción de “Generar JavaDoc”, ahora ya se nos generará/generó correctamente la documentación de nuestros proyectos/ejercicios.