TAREA PARA PSP07

Enunciado de la tarea

fpdistancia Usuarios Español - Internacional (es) EBERT JAHNS CRISTOBAL SANCHEZ

Tarea para PSP07.

Detalles de la tarea de esta unidad.

Enunciado.
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.

Criterios de puntuación. Total 10 puntos.
Total 10 puntos.

Se tendrá en cuenta:

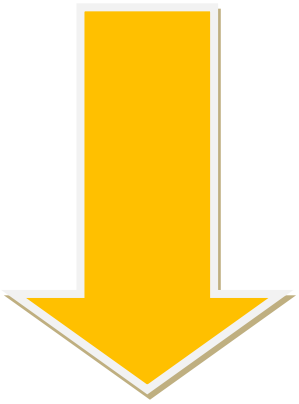
- El funcionamiento correcto del programa.
- El uso adecuado del API criptográfico.
- Tratamiento adecuado de posibles excepciones.

Recursos necesarios para realizar la Tarea.
Los contenidos y ejemplos realizados en la Unidad.
Consejos y recomendaciones.
Ninguno en particular.
Indicaciones de entrega.
Elabora un documento con un procesador de texto donde expliques cómo has realizado los dos ejercicios de la tarea. El documento debe tener tamaño de página A4, estilo de letra Times New Roman, tamaño 12 e interlineado normal.

Debes enviar el informe, y los dos proyectos, comprimidos en un fichero.

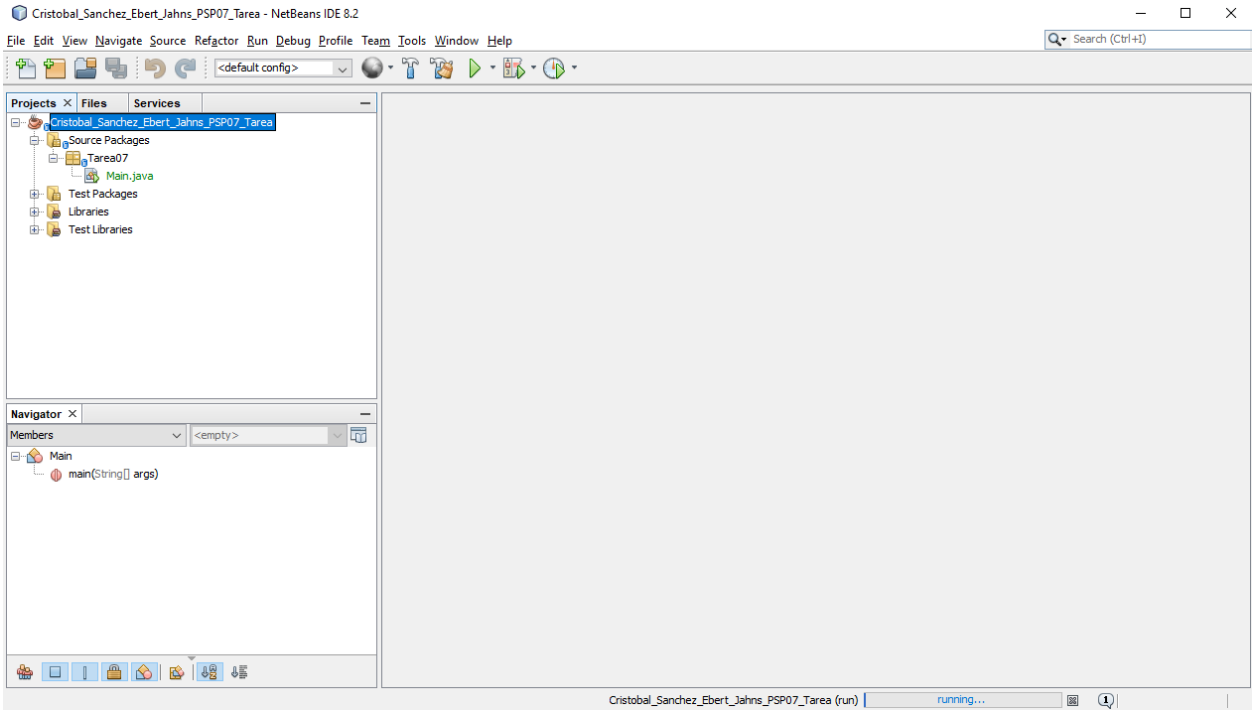
El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:

Esta tarea solo cuenta con un ejercicio. En la siguiente página comenzaremos a desarrollar esta tarea.

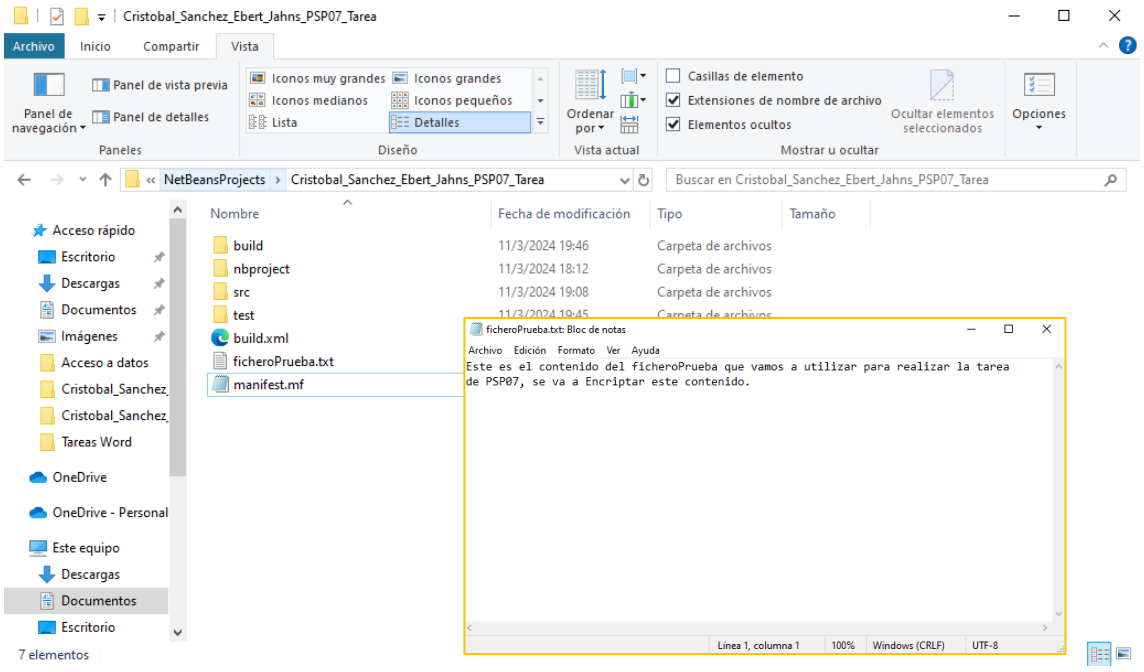


Para realizar este ejercicio se ha utilizado **NetBeans 8.2** y **JDK 1.8**

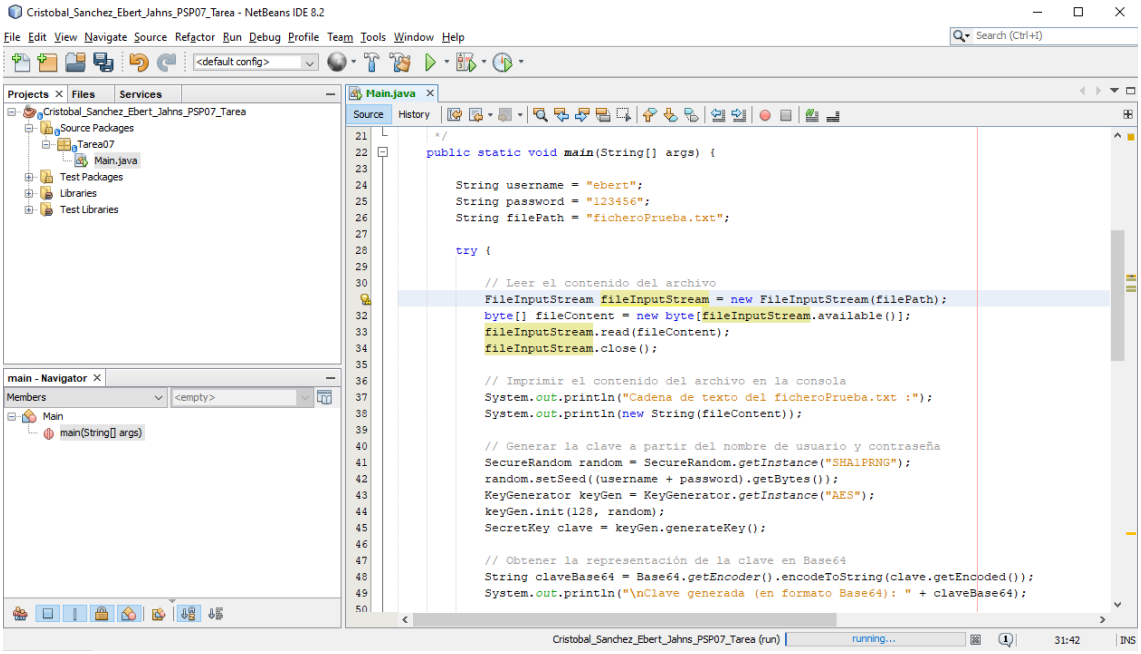
Se ha creado el proyecto **Cristobal_Sanchez_Ebert_Jahns_PSP07_Tarea**



En la imagen siguiente se muestra la carpeta del proyecto, y dentro de ella se ha creado un fichero de nombre **ficheroPrueba.txt** que es el fichero que vamos a encriptar para la solución de esta tarea.



Solo se ha utilizado una clase. En la clase Main se encuentra todo el código necesario que va a permitir cumplir con los requerimientos de la tarea.



Código Main.java

```
import java.io.*;
import java.security.*;
import java.util.Base64;
import javax.crypto.*;

/**
 *
 * @author Ebert
 */
public class Main {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        //Definición de variables para el nombre de usuario, contraseña y la ruta del archivo
        String username = "ebert";
        String password = "123456";
        String filePath = "ficheroPrueba.txt";

        try {

            // Leer el contenido del archivo
            FileInputStream fileInputStream = new FileInputStream(filePath);
            byte[] fileContent = new byte[fileInputStream.available()];
            fileInputStream.read(fileContent);
            fileInputStream.close();

            // Imprimir el contenido del archivo en la consola
            System.out.println("Cadena de texto del ficheroPrueba.txt :");
            System.out.println(new String(fileContent));

            // Generar la clave a partir del nombre de usuario y contraseña
            SecureRandom random = SecureRandom.getInstance("SHA1PRNG");
            random.setSeed((username + password).getBytes());
            KeyGenerator keyGen = KeyGenerator.getInstance("AES");
            keyGen.init(128, random);
            SecretKey clave = keyGen.generateKey();

            // Obtener la representación de la clave en Base64
            String claveBase64 = Base64.getEncoder().encodeToString(clave.getEncoded());
            System.out.println("\nClave generada (en formato Base64): " + claveBase64);
```

```
            // Crear el cifrador AES con el modo y relleno especificados
            Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE, clave);
            // Encriptar el contenido del archivo
            byte[] textoEncriptado = cipher.doFinal(fileContent);

            // Escribir el texto encriptado en un archivo
            FileOutputStream fos = new FileOutputStream("fichero.cifrado");
            fos.write(textoEncriptado);
            fos.close();

            System.out.println("Texto encriptado y almacenado en el archivo 'fichero.cifrado'");

            // Leer el archivo encriptado y desencriptarlo
            FileInputStream fis = new FileInputStream("fichero.cifrado");
            byte[] textoEncriptadoLeido = new byte[fis.available()];
            fis.read(textoEncriptadoLeido);
            fis.close();

            System.out.println("\nEl contenido del archivo encriptado es:");
            System.out.println(new String(textoEncriptadoLeido));

            // Desencriptar el texto leído
            cipher.init(Cipher.DECRYPT_MODE, clave);
            byte[] textoDesencriptado = cipher.doFinal(textoEncriptadoLeido);

            // Imprimir el texto desencriptado leído en la consola
            System.out.println("\nEl archivo se ha desencriptado y la cadena de texto es:");
            System.out.println(new String(textoDesencriptado));
            System.out.println("\nGenerado el fichero de desencriptación
'fichero_desencriptado.txt'");

            // Escribir el texto desencriptado en un archivo
            FileOutputStream fos2 = new FileOutputStream("fichero_desencriptado.txt");
            fos2.write(textoDesencriptado);
            fos2.close();

        } catch (NoSuchAlgorithmException | NoSuchPaddingException | InvalidKeyException |
IllegalBlockSizeException | BadPaddingException | IOException e) {
            e.printStackTrace();
        }
    }
}
```

Explicación de cada bloque de código:

```
//Definición de variables para el nombre de usuario, contraseña y la ruta del archivo
String username = "ebert";
String password = "123456";
String filePath = "ficheroPrueba.txt";
```

Username y password son las credenciales que se utilizan para generar la clave, y **filePath** es la ruta del archivo que se va a leer y encriptar

```
// Leer el contenido del archivo
FileInputStream fileInputStream = new FileInputStream(filePath);
byte[] fileContent = new byte[fileInputStream.available()];
fileInputStream.read(fileContent);
fileInputStream.close();
```

Se abre un **FileInputStream** para el archivo especificado en **filePath**. Se lee el contenido del archivo en un array de bytes. Luego se cierra el **FileInputStream**.

```
//Imprimir el contenido del archivo en la consola
System.out.println("Cadena de texto del ficheroPrueba.txt :");
System.out.println(new String(fileContent));
```

Este bloque imprime el contenido del archivo en la consola.

```java
// Generar la clave a partir del nombre de usuario y contraseña
SecureRandom random = SecureRandom.getInstance("SHA1PRNG");
random.setSeed((username + password).getBytes());
KeyGenerator keyGen = KeyGenerator.getInstance("AES");
keyGen.init(128, random);
SecretKey clave = keyGen.generateKey();
```

Se genera la clave utilizando el algoritmo **SHA1PRNG** y el algoritmo de cifrado **AES**. La semilla para el generador de números aleatorios **(SecureRandon)** se basa en la concatenación del **username y password**

```java
// Obtener la representación de la clave en Base64
String claveBase64 = Base64.getEncoder().encodeToString(clave.getEncoded());
System.out.println("\nClave generada (en formato Base64): " + claveBase64);
```

Este bloque convierte la clave generada a su representación en formato Base64 y la imprime en la consola.

```java
// Crear el cifrador AES con el modo y relleno especificados
Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, clave);
// Encriptar el contenido del archivo
byte[] textoEncriptado = cipher.doFinal(fileContent);
```

Se instancia un objeto **Cipher** con el algoritmo de cifrado AES y el modo de operación **ECB** con relleno **PKCS5**. Luego se inicializa el cifrador en modo de encriptación con la clave generada. El contenido del archivo se encripta utilizando el método **doFinal()**

```java
// Escribir el texto encriptado en un archivo
FileOutputStream fos = new FileOutputStream("fichero.cifrado");
fos.write(textoEncriptado);
fos.close();
```

El texto encriptado se escribe en un archivo llamado **"fichero.cifrado"**. que se crea en la carpeta del proyecto

```java
// Leer el archivo encriptado y desencriptarlo
FileInputStream fis = new FileInputStream("fichero.cifrado");
byte[] textoEncriptadoLeido = new byte[fis.available()];
fis.read(textoEncriptadoLeido);
fis.close();
```

Se lee el contenido del archivo encriptado recién creado.

```java
// Desencriptar el texto leído
cipher.init(Cipher.DECRYPT_MODE, clave);
byte[] textoDesencriptado = cipher.doFinal(textoEncriptadoLeido);
```

El cifrador se reinicia en modo de desencriptación y se desencripta el contenido del archivo encriptado.

```java
// Imprimir el texto desencriptado leído en la consola
System.out.println("\nEl archivo se ha desencriptado y la cadena de texto es:");
System.out.println(new String(textoDesencriptado));
System.out.println("\nGenerado el fichero de desencriptación 'fichero_desencriptado.txt'");
```

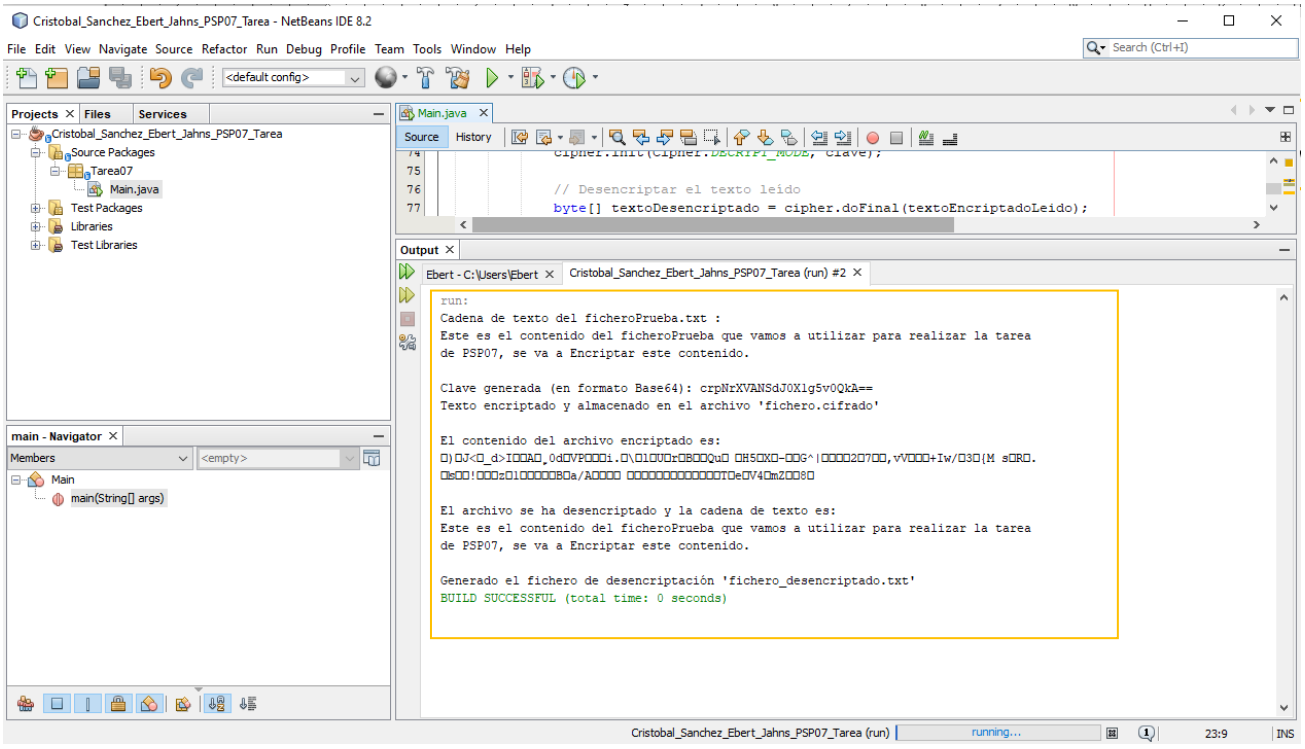
El texto desencriptado se imprime en consola

```java
// Escribir el texto desencriptado en un archivo
FileOutputStream fos2 = new FileOutputStream("fichero_desencriptado.txt");
fos2.write(textoDesencriptado);
fos2.close();
```

El texto desencriptado se escribe en un archivo llamado **"fichero_desencriptado.txt".**

Ejecutamos el programa

El resultado de ejecutar **Main.java** es el siguiente:



A continuación, vamos a ver cómo se muestra el contenido de la carpeta del proyecto, en ella debe crearse un archivo de nombre **"fichero.cifrado"** y otro archivo de nombre

“fichero_desencriptado.txt”, donde el contenido de este último debe coincidir con el contenido del archivo “**ficheroPrueba.txt**”.