**ACTIVIDAD 2**

**Enunciado**

Realiza un programa en Java que verifique la validez de un fichero de firma previamente generado. A modo orientativo, la estructura del programa podría ser la siguiente:

* Leer la clave pública del fichero a verificar. Dicha clave está dentro de un fichero de nombre *Clave.publica* disponible en la plataforma.
* Recuperar la clave pública desde los datos codificados en formato X509 y mediante el algoritmo *DSA*.
* Leer el fichero que contiene la firma. Dicho fichero se llama *FICHERO.FIRMA* y está disponible en la plataforma.
* Inicializar el objeto *Signature* con la clave pública que se va a usar para verificar la validez del fichero y mediante el algoritmo *SHA1withDSA.*
* Leer el fichero que contiene los datos a verificar (*FICHERO.DAT*), el cual está disponible en la plataforma.
* Verificar la firma de los datos leídos del fichero *FICHERO.DAT.*
* Comprobar la verificación de los datos leídos. Si la verificación es correcta se mostrará el mensaje “LOS DATOS SE CORRESPONDEN CON SU FIRMA”. En caso contrario, se mostrará este otro mensaje: “LOS DATOS NO SE CORRESPONDEN CON SU FIRMA”.

El programa gestionará las posibles excepciones que se puedan producir durante su ejecución.

El nombre de la clase principal será Actividad2.

**Solución**

package paquete;

import java.io.FileInputStream;

import java.security.KeyFactory;

import java.security.PublicKey;

import java.security.Signature;

import java.security.spec.X509EncodedKeySpec;

public class Actividad2 {

public static void main(String[] args) {

try {

// Leer la clave pública

FileInputStream keyfis = new FileInputStream("src/paquete/Clave.publica");//La ruta del fichero en me caso src/paquete...

byte[] encKey = keyfis.readAllBytes();

keyfis.close();

X509EncodedKeySpec pubKeySpec = new X509EncodedKeySpec(encKey);

KeyFactory keyFactory = KeyFactory.*getInstance*("DSA");

PublicKey pubKey = keyFactory.generatePublic(pubKeySpec);

// Leer la firma del fichero

FileInputStream sigfis = new FileInputStream("src/paquete/FICHERO.FIRMA");//La ruta del fichero en me caso src/paquete...

byte[] sigToVerify = sigfis.readAllBytes();

sigfis.close();

// Inicializar el objeto Signature

Signature sig = Signature.*getInstance*("SHA1withDSA");

sig.initVerify(pubKey);

// Leer el fichero de datos y suministrarlo a Signature

FileInputStream datafis = new FileInputStream("src/paquete/FICHERO.DAT");//La ruta del fichero en me caso src/paquete...

byte[] buffer = new byte[1024];

int len;

while (datafis.available() != 0) {

len = datafis.read(buffer);

sig.update(buffer, 0, len);

}

datafis.close();

// Verificar la firma

boolean verifies = sig.verify(sigToVerify);

System.***out***.println("LOS DATOS " + (verifies ? "SE CORRESPONDEN CON SU FIRMA" : "NO SE CORRESPONDEN CON SU FIRMA"));

} catch (Exception e) {

e.printStackTrace();

}

}

}

**Resolución de problemas**

**CRITERIOS DE CORRECCIÓN**

1. Nivel de ajuste a la nomenclatura de objetos definida en el enunciado de la actividad y nivel de ajuste al almacenamiento de la información requerida en las subcarpetas definidas para la actividad y unidad didáctica correspondientes (15%).
2. Hacer lo que se indica en el enunciado (55%).
3. Claridad del código Java (15%).
4. Documentación del código Java (15%).
5. Explicación detallada de cómo se han resuelto problemas que hayan podido surgir en la realización de la actividad (hasta un 15% adicional sobre la nota obtenidaVariables no usadas (-15%).
6. Librerías no usadas (-15%).