



EXAMEN HASH ENCRYPTACIÓN

MARIO JIMÉNEZ MARSET

ÍNDICE

1. ENUNCIADO – OBJETIVOS	3
2. DESARROLLO – PROCEDIMIENTOS.....	3

1. ENUNCIADO – OBJETIVOS

En este examen se pedía desarrollar una clase donde, a partir del fichero de texto dado, se cree otro fichero encriptado mediante el algoritmo de cifrado asimétrico RSA, mostrando las claves públicas y privadas por pantalla. Además, se pedía que, el receptor, a partir de la clave privada y del fichero creado, se desenscriptase el contenido y se obtuviese el fichero original.

También se pedía desarrollar otra clase que crease un hash a partir del texto original, mostrándolo por pantalla.

Finalmente, se pedía comparar el hash creado entre este fichero y cualquier otro introducido por pantalla (mensaje comparativo).

2. DESARROLLO – PROCEDIMIENTOS

Se muestra el código de las dos clases utilizadas (comentado):

Código ClaseUnoUno:

```
package examen;
import java.io.*;
import java.security.*;
import java.security.spec.InvalidKeySpecException;
import javax.crypto.*;

public class ClaseUnoUno {
    public static void main(String[] args) throws NoSuchAlgorithmException,
        NoSuchPaddingException, InvalidKeyException, IllegalBlockSizeException,
        BadPaddingException, InvalidKeySpecException {
        //se crea el cifrador RSA perteneciente a la clase Cipher
        Cipher cifradorRsa;
        //se generan el par de claves RSA
        KeyPairGenerator keyPairGenerator =
        KeyPairGenerator.getInstance("RSA");
        KeyPair keyPair = keyPairGenerator.generateKeyPair();
        //se guardan el par de claves pública y privada en variables
        PublicKey publicKey = keyPair.getPublic();
        PrivateKey privateKey = keyPair.getPrivate();
        //se imprimen por pantalla el formato de las dos claves
        System.out.println("La clave pública es "+publicKey.getFormat());
        System.out.println("La clave privada es "+privateKey.getFormat());
        //se establece el tipo de cifrado (por parámetro se establece el cifrado
        RSA)
        cifradorRsa = Cipher.getInstance("RSA/ECB/PKCS1Padding");
        //se escribe la ruta del fichero cuyo contenido va a ser codificado
        String ficheroTexto="C:\\Users\\jimen\\Downloads\\mensaje texto.txt";
        String contenido=null;
        try {
            //se crea el objeto de tipo File que va a recoger por parámetro
            la ruta insertada
            File f=new File(ficheroTexto);
            //se crea el FileReader y BufferedReader encargados de leer el
            contenido del fichero
            FileReader fr=new FileReader(f);
            BufferedReader br=new BufferedReader(fr);
```

```

//dentro del bucle que lee el contenido, se hará el cifrado y
descifrado
while((contenido=br.readLine())!=null) {
    //cada línea se almacena en otro String diferente
    String texto=contenido;
    cifradorRsa.init(Cipher.ENCRYPT_MODE, publicKey);
    //se almacena en un array de bytes la información del
    fichero
    byte[] encriptado =
    cifradorRsa.doFinal(texto.getBytes());
    System.out.print("Encriptado-> ");
    //por cada byte guardado, se realiza una conversión
    hexadecimal
    for (byte b : encriptado) {
        System.out.print(Integer.toHexString(0xFF & b));
    }
    //se imprime el texto encriptado (la colección de bytes)
    System.out.println("\nEl texto encriptado es:
"+encriptado.toString());
    cifradorRsa.init(Cipher.DECRYPT_MODE, privateKey);
    //en otro array de bytes, se almacena la información
    desencriptada
    byte[] bytesDesencriptados =
    cifradorRsa.doFinal(encriptado);
    //se muestra el array de bytes, convirtiéndose a String
    String textoDesencriptado = new
    String(bytesDesencriptados);
    //se imprime por pantalla cada línea desencriptada del fichero
    System.out.println("El texto desencriptado es:
"+textoDesencriptado);
    }
    br.close();
} catch (FileNotFoundException e1) {
    System.out.println(e1.getMessage());
} catch (IOException e2) {
    System.out.println(e2.getMessage());
}
}
}

```

Código ClaseDosUno:

```

package examen;
import java.security.*;
import java.util.Scanner;
import org.apache.commons.codec.binary.Hex;

public class ClaseDosUno {
    public static void main(String[] args) {
        //se crean dos métodos: el primero, muestra el hash del fichero original
        //el segundo gestiona el hash del fichero insertado y el mensaje final
        System.out.print("ALGORITMO HASH SHA-512-> ");
        hash();
        hashNuevo(hash());
    }
    public static String hash() {
        //a partir de la clase MessageDigest, se gestiona la creación de hash
        MessageDigest md = null;

```

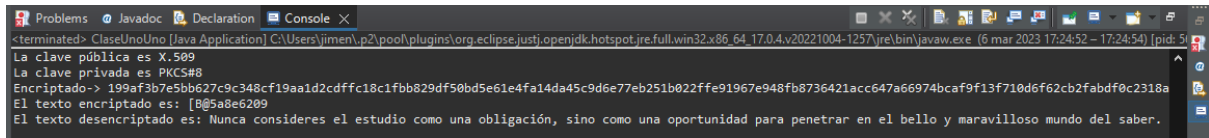
```

String fichero="C:\\Users\\jimen\\Downloads\\mensaje texto.txt";
char[] hash=null;
try {
    //se llama al método getInstance, con el algoritmo elegido
    md=MessageDigest.getInstance("SHA-512");
    //se llama al método actualizar, consiguiendo los bytes del
    fichero
    md.update(fichero.getBytes());
    byte[] mb=md.digest(fichero.getBytes());
    hash=Hex.encodeHex(mb);
    //se imprime el hash finalmente creado
    System.out.println(Hex.encodeHex(mb));
}catch(NoSuchAlgorithmException e) {
    System.out.println(e.getMessage());
}
return String.valueOf(hash);
}

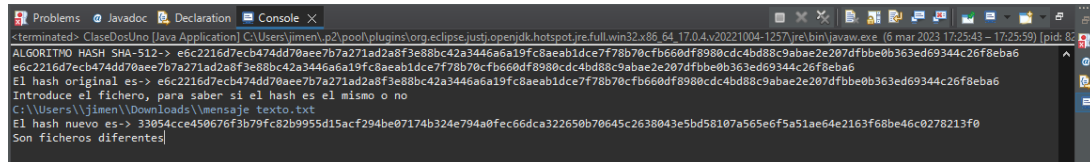
public static void hashNuevo(String hash) {
    //se guarda en una variable el hash creado por el anterior método
    String hashTextoDado=hash;
    System.out.print("El hash original es-> ");
    System.out.println(hashTextoDado);
    //se siguen los mismos pasos que en el anterior método para crear el
    hash del fichero introducido
    char[] hashNuevo=null;
    Scanner entrada=new Scanner(System.in);
    System.out.println("Introduce el fichero, para saber si el hash es el
    mismo o no");
    MessageDigest md = null;
    String fichero=entrada.nextLine();
    try {
        md=MessageDigest.getInstance("SHA-512");
        md.update(fichero.getBytes());
        byte[] mb=md.digest(fichero.getBytes());
        hashNuevo=Hex.encodeHex(mb);
        System.out.print("El hash nuevo es-> ");
        System.out.println(Hex.encodeHex(mb));
        //lo único nuevo es que se comparan las cadenas (el hash
        original y el nuevo)
        //saldrá un mensaje u otro dependiendo de si es el mismo hash
        if(hashTextoDado.equals(String.valueOf(hashNuevo))) {
            System.out.println("Es el mismo fichero. Coinciden los
            hashes");
        }else {
            System.out.println("Son ficheros diferentes");
        }
    }catch(NoSuchAlgorithmException e) {
        System.out.println(e.getMessage());
    }
}
}

```

CAPTURAS RESULTADOS:



```
<terminated> ClaseUnoUno [Java Application] C:\Users\jimen\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20221004-1257\jre\bin\javaw.exe (6 mar 2023 17:24:52 - 17:24:54) [pid: 5128]
La clave pública es X.509
La clave privada es PKCS#8
Encriptado-> 199af3b7e5bb627c9c348cf19aa1d2cdfc18c1fbb829df50bd5e61e4fa14da45c9d6e77eb251b022ffe91967e948fb8736421acc647a66974bcaf9f13f710d6f62cb2fabdf0c2318a
El texto encriptado es: [B@5a8e6209
El texto desencriptado es: Nunca consideres el estudio como una obligación, sino como una oportunidad para penetrar en el bello y maravilloso mundo del saber.
```



```
<terminated> ClaseDosUno [Java Application] C:\Users\jimen\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.4.v20221004-1257\jre\bin\javaw.exe (6 mar 2023 17:25:43 - 17:25:59) [pid: 8128]
ALGORITMO HASH SHA-512-> e6c2216d7ecb474dd70aee7b7a271ad2a8f3e88bc42a3446a6a19fc8aeab1dce7f78b70cfb660df8980cdc4bd88c9abae2e207dfbbe0b363ed69344c26f8eba6
e6c2216d7ecb474dd70aee7b7a271ad2a8f3e88bc42a3446a6a19fc8aeab1dce7f78b70cfb660df8980cdc4bd88c9abae2e207dfbbe0b363ed69344c26f8eba6
El hash original es-> e6c2216d7ecb474dd70aee7b7a271ad2a8f3e88bc42a3446a6a19fc8aeab1dce7f78b70cfb660df8980cdc4bd88c9abae2e207dfbbe0b363ed69344c26f8eba6
Introduce el fichero, para saber si el hash es el mismo o no
C:\Users\jimen\Downloads\Mensaje texto.txt
El hash nuevo es-> 33054cce450676f3b79fc82b9955d15acf294be07174b324e794a0fec66dca322650b70645c2638043e5bd58107a565e6f5a51ae64e2163f68be46c9278213f0
Son ficheros diferentes
```