



PRÁCTICA ALGORITMOS HASH

MARIO JIMÉNEZ MARSET

ÍNDICE

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

1. ENUNCIADO – OBJETIVOS

En esta práctica se pedía realizar, a partir del código proporcionado, pasar a partir de algoritmos hash, una contraseña en específico (como variable String) y ser codificada. Se muestran los resultados de los diferentes algoritmos hash.

2. DESARROLLO – PROCEDIMIENTOS

Se muestra el código comentado de la clase y los resultados. Las modificaciones realizadas han sido cambiar la funcionalidad de la contraseña (en vez de ser una contraseña estática, se ha implementado un Scanner para que el usuario introduzca la que más le plazca). Además, se ha introducido otro tipo de algoritmo hash que no estaba presente en el código proporcionado.

Código Clase DigestExample:

```
package hash;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.*;
import org.apache.commons.codec.binary.Hex;

public class DigestExample {
    public static void main(String[] args) {
        //se crean las variables a utilizar en el programa
        //se crea un objeto MessageDigest, el cual permite "digerir" un mensaje
        MessageDigest md = null;
        //se crea un objeto Scanner, el cual permite escribir una contraseña en
        la variable String declarada
        Scanner entrada=new Scanner(System.in);
        System.out.println("Escribe una password cualquiera:");
        String password=entrada.nextLine();
        //bloque try-catch para el manejo de excepciones
        try {
            //en primer lugar, se crea el algoritmo hash SHA-512
            //se inicializa el objeto MessageDigest con el método
            getInstance, estableciendo como String el nombre del algoritmo hash
            md=MessageDigest.getInstance("SHA-512");
            //se actualiza este objeto con la cadena establecida por consola
            anteriormente
            md.update(password.getBytes());
            //el objeto MessageDigest se "digiere" dentro de un array de
            bytes
            byte[] mb=md.digest();
            //se imprime por pantalla cómo queda el algoritmo SHA-512
            System.out.print("ALGORITMO HASH SHA-512-> ");
            System.out.println(Hex.encodeHex(mb));
            //con el algoritmo SHA-1, al igual que en el anterior, se llama al
            método getInstance con el nombre del algoritmo
            md=MessageDigest.getInstance("SHA-1");
            //se actualiza el objeto MessageDigest con la cadena
            establecida por consola
            md.update(password.getBytes());
            //se digiere el objeto dentro del array de bytes
            mb=md.digest();
            //se imprime el resultado
```

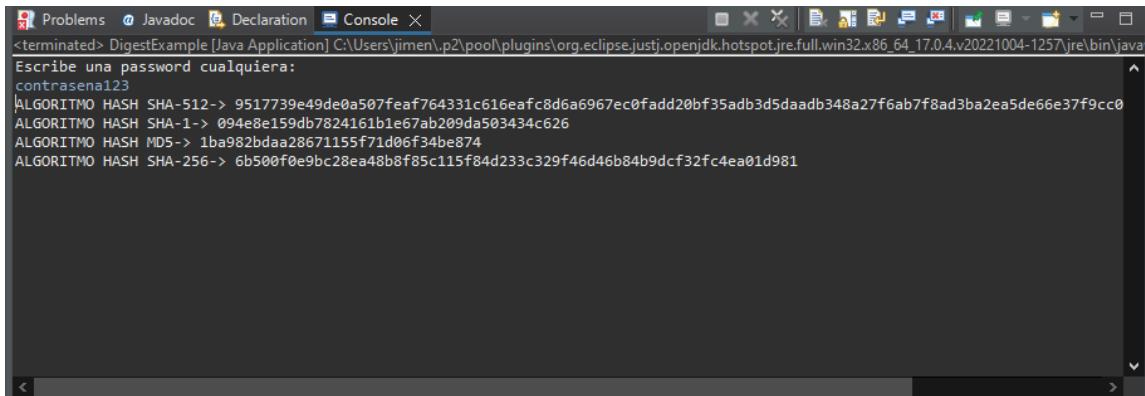
```

        System.out.print("ALGORITMO HASH SHA-1-> ");
        System.out.println(Hex.encodeHex(mb));
        //con el algoritmo MD5 también se llama al método getInstance
        y se hace lo mismo que en los casos anteriores
        md=MessageDigest.getInstance("MD5");
        //se actualiza el objeto MessageDigest con la cadena
        establecida por consola
        md.update(password.getBytes());
        //se digiere el objeto dentro del array de bytes
        mb=md.digest();
        //se imprime el resultado
        System.out.print("ALGORITMO HASH MD5-> ");
        System.out.println(Hex.encodeHex(mb));
        //se realiza el mismo procedimiento que en el algoritmo SHA-
256

        md=MessageDigest.getInstance("SHA-256");
        md.update(password.getBytes());
        byte[] mc=md.digest();
        //se imprime el resultado
        System.out.print("ALGORITMO HASH SHA-256-> ");
        System.out.println(Hex.encodeHex(mc));
    }catch(NoSuchAlgorithmException e) {
        System.out.println(e.getMessage());
    }
    entrada.close();
}
}

```

CAPTURA RESULTADO CON CONTRASEÑA DE EJEMPLO:



The screenshot shows a Java application window titled "DigestExample [Java Application]". The console output is as follows:

```

<terminated> DigestExample [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\java
Escribe una password cualquiera:
contrasena123
ALGORITMO HASH SHA-512-> 9517739e49de0a507feaf764331c61eafc8d6a6967ec0fadd20bf35adb3d5daadb348a27f6ab7f8ad3ba2ea5de66e37f9cc0
ALGORITMO HASH SHA-1-> 094e8e159db7824161b1e67ab209da503434c626
ALGORITMO HASH MD5-> 1ba982bdaa28671155f71d06f34be874
ALGORITMO HASH SHA-256-> 6b50f0e9bc28ea48b8f85c115f84d233c329f46d46b84b9dcf32fc4ea01d981

```

Se muestran las diferentes codificaciones de los algoritmos hash (incluyendo el último como extra). Como conclusión, se ha aprendido una nueva clase de Java (MessageDigest), además de los métodos relacionados con el jar externo de commons code de Apache.