**Taller 9 – Cifrado Asimétrico y Control de Integridad**

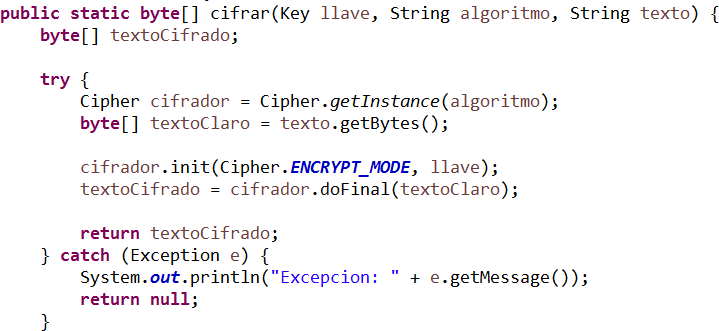
El propósito de este taller es familiarizarse con las librerías Java (javax.crypto.\*) para el cifrado de información usando algoritmos criptográficos de llave asmiétrica y control de integridad.

# Parte 1: Cifrar y descifrar con cifrado asimétrico

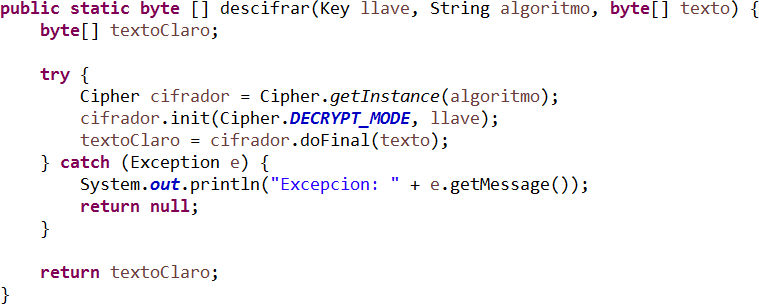
1. Escriba una clase llamada **Asimetrico** en la cual vamos a incluir los métodos **cifrar()**

y **descifrar()**.

Cree el método **cifrar()** usando el siguiente código.



Cree el método **descifrar()** usando el siguiente código.



2. Escriba una clase **Main** para crear una aplicación que use los métodos **cifrar()** y

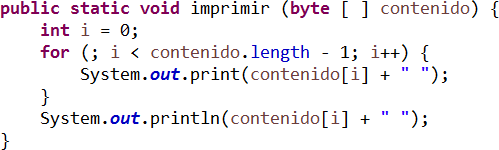
**descifrar()** de la clase **Asimetrico**.

En el método **main()**:

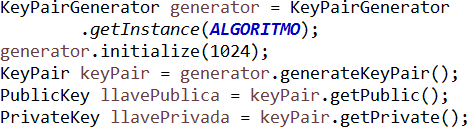
1. Utilice la siguiente constante para indicar que utilizará el algoritmo **RSA** para cifrar y descifrar sus mensajes:



1. Reciba por teclado la entrada de un texto.
2. Imprima el texto recibido por el teclado.
3. Imprima texto claro en **byte []**. Utilice el método **getBytes()** de la clase **String** para convertir el mensaje a **byte[]**. Para imprimr el contenido del **byte[]** utilice el método **imprimir()**.



1. Genere un par de llaves asimétricas: una pública y otra privada, las cuales empleará para cifrar y descifrar. Utilice el siguiente fragmento de código:

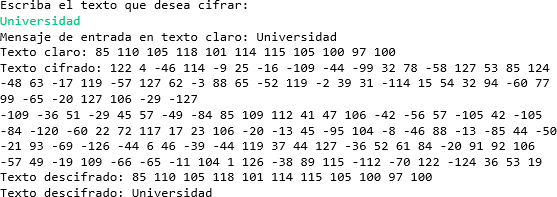


1. Obtenga un **byte []** con el texto cifrado, invocando al método **cifrar()** de la clase

**Asimetrico**. Utilice la llave pública para cifrar.

1. Imprima el texto cifrado en **byte []**. Utilice el método **imprimir()**.
2. Obtenga un **byte []** con el texto descifrado, invocando al método **descifrar()** de la clase **Asimetrico**. Utilice la llave privada para descifrar.
3. Imprima el texto descifrado en **byte []**. Utilice el método **imprimir()**.
4. Convierta el **byte[]** con el texto descifrado a **String**. Utilice el método constructor de la clase **String**.

Un ejemplo de la salida en consola de la ejecución del programa se puede ver a continuación:



1. Escriba el método **main()**.

**try**

{

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(System.***in***));

System.***out***.println("Escriba el texto que desea cifrar:");

String texto = in.readLine();

System.***out***.println("El texto escrito es: " + texto);

System.***out***.print("Texto original: ");

*imprimir* (texto.getBytes());

KeyPairGenerator keygen = KeyPairGenerator.*getInstance*(***ALGORITMO***);

keygen.initialize(1024);

KeyPair keyPair = keygen.generateKeyPair();

PublicKey publicK = keyPair.getPublic();

PrivateKey privateK = keyPair.getPrivate();

**long** ini = System.*nanoTime*();

**byte**[] cifrado = *cifrar*(publicK, ***ALGORITMO***, texto);

**long** fin = System.*nanoTime*();

System.***out***.print("Tardó " + (fin-ini) + "ns Texto cifrado: ");

*imprimir* (cifrado);

ini = System.*nanoTime*();

**byte**[] descifrado = *descifrar* (privateK, ***ALGORITMO***, cifrado);

fin = System.*nanoTime*();

System.***out***.print("Tardó " + (fin-ini) + "ns Texto descifrado: ");

*imprimir*(descifrado);

System.***out***.println("Texto original: " + **new** String(descifrado));

}

**catch** (Exception e)

{

System.***out***.println("Exception :" + e.getMessage());

e.printStackTrace();

}

3. Responda las siguientes preguntas:

* 1. ¿Cuál es el propósito del llamado al método **initialize()** con el parámetro **1024**?

Determinar el tamaño de la llave con la que se cifrará

* 1. Describa brevemente el funcionamiento del algoritmo RSA.

Un mensaje se convierte en números los cuales son elevados con la llave pública y luego se realiza modulo del producto de dos números primos, esto es el cifrado.

Para descifrar, se toma el número cifrado se le eleva con la llave privada y se le realiza modulo con el mismo producto usado en el cifrado.

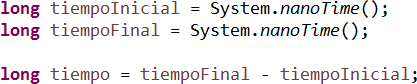
* 1. ¿Si desea confidencialidad, con cuál llave debe cifrar y con cuál llave debe descifrar?

Se debe cifrar con la pública, así solo la persona dueña de la llave privada puede descifrar.

* 1. ¿Si desea autenticación del emisor, con cuál llave debe cifrar y con cuál llave debe descifrar?

Se cifra con su llave privada y así solo con su llave pública se podrá descifrar.

* 1. ¿Cuánto tiempo tarda cifrar y descifar el mensaje de entrada? Utilice el siguiente fragmento de código:



255475717ns -> 0,26 segundos

4. Cree una clase llamada **Main2** y en ella cree el método **main()** de acuerdo con las siguientes modificaciones al método **main()** inicial y responda:

* + 1. Utilice el método **cifrar()** con la llave privada y el método **descifrar()** con la llave pública. Explique el resultado obtenido.

Al descifrar se llaga al mismo resultado

* + 1. Escriba el método **main()**.

**try**

{

BufferedReader in = **new** BufferedReader(**new** InputStreamReader(System.***in***));

System.***out***.println("Escriba el texto que desea cifrar:");

String texto = in.readLine();

System.***out***.println("El texto escrito es: " + texto);

System.***out***.print("Texto original: ");

*imprimir* (texto.getBytes());

KeyPairGenerator keygen = KeyPairGenerator.*getInstance*(***ALGORITMO***);

keygen.initialize(1024);

KeyPair keyPair = keygen.generateKeyPair();

PublicKey publicK = keyPair.getPublic();

PrivateKey privateK = keyPair.getPrivate();

**long** ini = System.*nanoTime*();

**byte**[] cifrado = *cifrar*(privateK, ***ALGORITMO***, texto);

**long** fin = System.*nanoTime*();

System.***out***.print("Tardó " + (fin-ini) + "ns Texto cifrado: ");

*imprimir* (cifrado);

ini = System.*nanoTime*();

**byte**[] descifrado = *descifrar* (publicK, ***ALGORITMO***, cifrado);

fin = System.*nanoTime*();

System.***out***.print("Tardó " + (fin-ini) + "ns Texto descifrado: ");

*imprimir*(descifrado);

System.***out***.println("Texto original: " + **new** String(descifrado));

}

**catch** (Exception e)

{

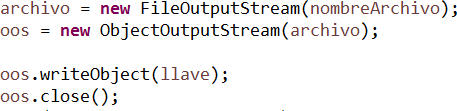
System.***out***.println("Exception :" + e.getMessage());

e.printStackTrace();

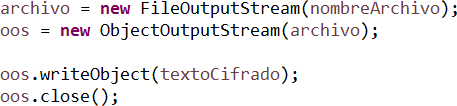
}

5. Cree una clase llamada **Main3** en ella cree el método **main()** de acuerdo con las siguientes modificaciones al método **main()** inicial y responda

1. Genere en un archivo la llave pública y en otro la llave privada. Utilice el siguiente fragmento de código:

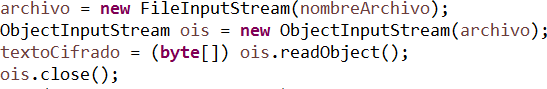


1. Cifre un mensaje de entrada. Almacene el texto cifrado en un archivo. Utilice el siguiente fragmento de código:

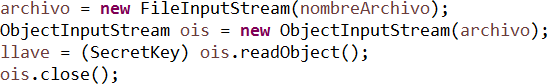


6. Cree una clase llamada **Main4** en ella cree el método **main()** de acuerdo con las siguientes modificaciones al método **main()** inicial y responda

1. Recupere la llave pública que está en un archivo. Recupere la llave privada que está en otro archivo. Utilice el siguiente fragmento de código:



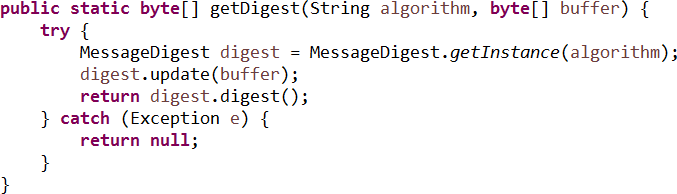
1. Recupere el texto cifrado que está en el archivo. Utilice el siguiente fragmento de código:



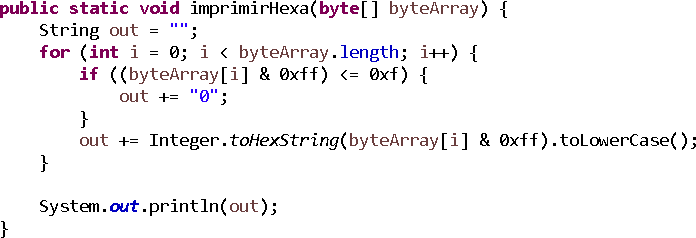
# Parte 2: Control de Integridad

1. Escriba una clase llamada **Digest** en la cual vamos a incluir el método **getDigest()**.

1. Cree el método **getDigest()** usando el siguiente código.



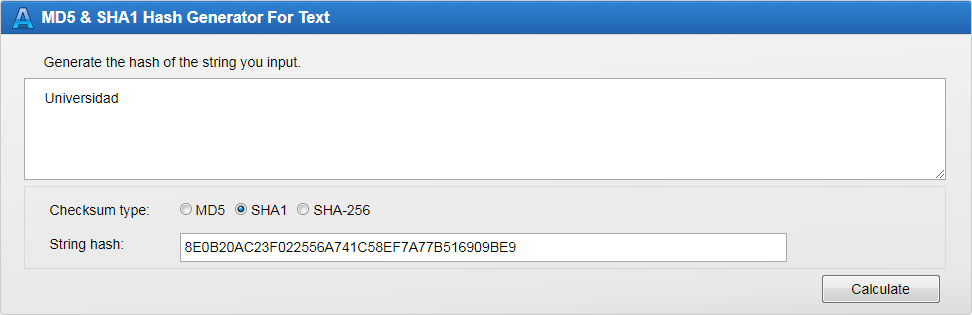
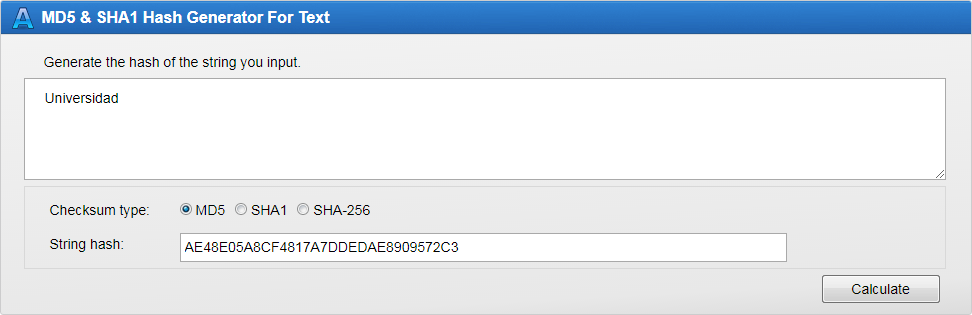
1. Imprima el digest obtenido. Utilice el método **imprimirHexa()**.



3. Escriba una clase **Main** para crear una aplicación que use el método **getDigest()**.

En el método **main()**:

1. Reciba por teclado la entrada de un texto.
2. Imprima el texto recibido por el teclado.
3. Convierta el **String** a **byte []** usando el método **getBytes()** de la clase **String**.
4. Utilizando el método **getDigest()** calcule el digest MD5 del mensaje e imprima el valor. Guarde una imagen de la captura de pantalla.
5. Utilizando el método **getDigest()** calcule el digest SHA-1 del mensaje e imprima el valor. Guarde una imagen de la captura de pantalla. Guarde una imagen de la captura de pantalla.
6. Utilizando la aplicación disponible en la página [**http://onlinemd5.com/**](http://onlinemd5.com/) calcule y compare los resultados obtenidos. Guarde una imagen de la captura de pantalla.



1. Desarrolle un método **verificar()** que reciba dos parámetros de tipo **byte []** y determine si su contenido es idéntico o no.

Un ejemplo de la salida en consola de la ejecución del programa con digest MD5 se puede ver a continuación:



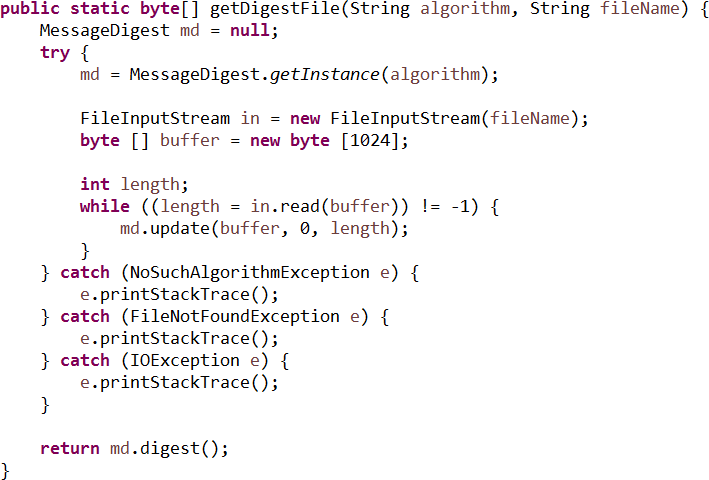
Un ejemplo de la salida en consola de la ejecución del programa con digest SHA-1 se puede ver a continuación:



1. Escriba el método **main()**.

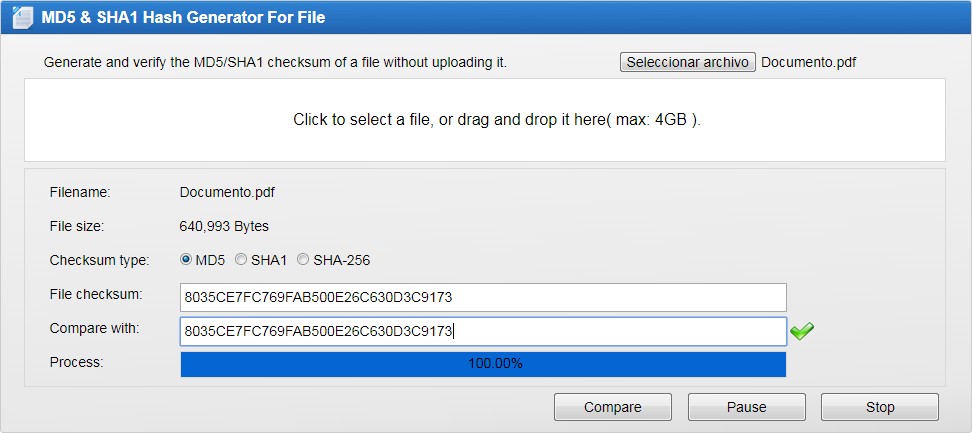
4. Agregue el método **getDigestFile()** a la clase **Digest** para obtener el digest de un archivo, y cree una nueva clase **Main2** con un método **main ()** en la que se pueda probar.

1. Cree el método **getDigestFile()** usando el siguiente código.



En el método **main()**:

1. Reciba por teclado el nombre de un archivo que esté en la carpeta raíz del proyecto.
2. Calcule el digest MD5 del archivo e imprima el valor obtenido.
3. Calcule el digest SHA-1 del archivo e imprima el valor obtenido.
4. Utilizando la aplicación disponible en la página [**http://onlinemd5.com/**](http://onlinemd5.com/) calcule y compare los resultados obtenidos. Guarde una imagen de la captura de pantalla.



Un ejemplo de la salida en consola de la ejecución del programa con digest MD5 se puede ver a continuación:



Un ejemplo de la salida en consola de la ejecución del programa con digest SHA-1 se puede ver a continuación:



1. Escriba el método **main()**.