

# ***Práctica 1: API de cifrado de Java (JCA: Java Cryptographic Architecture)***

**Miguel Ferreiro Díaz  
José Domínguez Vázquez**

# Contenidos

1. Descripción breve de la práctica
2. Descripción y justificación de las estrategias criptográficas empleadas
3. Descripción del formato/estructura del "paquete" resultante
4. Descripción breve de la implementación
5. Instrucciones de compilación y ejemplos de uso
6. Breve comentario sobre las simplificaciones realizadas y sus consecuencias en una aplicación real
7. Resultados obtenidos y conclusiones

## ***1. Descripción breve de la práctica***

Se trata de desarrollar una herramienta para el empaquetado y distribución de exámenes que sea fiable y segura y garantice las restricciones de entrega en plazo.

Para ello se contará con una especie de Autoridad de sellado de tiempo simplificada.

Los alumnos podrán generar su Examen Empaquetado que será remitido a la autoridad de sellado para vincularle el timestamp con la fecha de entrega.

Finalmente, el profesor podrá validar este Examen Empaquetado para extraer los datos aportados por el alumno y validar la autenticidad del sello emitido por la autoridad de sellado.

## ***2. Descripción y justificación de las estrategias criptográficas empleadas***

***Requisitos básicos a cumplir por el esquema criptográfico propuesto en la práctica:***

**R1.**

Asegurar la confidencialidad del contenido incluido en la Examen Empaquetado por parte del ALUMNO (sólo el PROFESOR podrá tener acceso a estos contenidos).

- Opcionalmente se puede asegurar también la confidencialidad de la información incluida en el sello de la AUTORIDAD SELLADO (en caso de hacerlo así, sólo el PROFESOR podría tener acceso a estos contenidos), aunque no es imprescindible.

Para la funcionalidad requerida en este proyecto bastará con dar soporte al sellado y esos datos (timestamp) pueden dejarse en claro en el paquete que da soporte físico al Examen Empaquetado.

**R2.**

Garantizar que el PROFESOR tenga la seguridad de que el ALUMNO que presenta el Examen Empaquetado es quien dice ser.

**R3.**

Asegurar que el contenido del paquete con el Examen Empaquetado (datos del ALUMNO y sello de AUTORIDAD SELLADO) que se ha recibido no haya sido modificado.

**R4.**

Asegurar que ni el ALUMNO ni la AUTORIDAD SELLADO podrán repudiar el contenido incluido por ellos en el Examen Empaquetado

**R5.**

Asegurar que el PROFESOR no podrá realizar cambios en el contenido del Examen Empaquetado que ha recibido.

**R6.**

Contar con un mecanismo mediante el cual la AUTORIDAD pueda garantizar la fecha en que fue recibido el Examen Empaquetado generado por un determinado ALUMNO.

Se pretende que esta vinculación entre Examen Empaquetado y sello pueda ser validada por el PROFESOR y que no pueda ser falsificada ni por el ALUMNO, ni por la AUTORIDAD SELLADO (o por otras AUTORIDADES SELLADO ajenas), ni por el propio PROFESOR.

#### **R7.**

Asegurar un coste computacional reducido en la creación, sellado y validación del Examen Empaquetado minimizando el uso de criptografía asimétrica

#### ***Estrategias seguidas para cumplir los requisitos descritos para formar el esquema criptográfico propuesto en la práctica:***

**R1.** Se asegura cifrando el examen con una clave a través de cifrado simétrico y cifrando esa clave con la clave pública del profesor mediante cifrado asimétrico. Esto asegura que solo puede descifrar la clave aquella persona que tenga la clave privada del profesor.  
(Cifrado Híbrido)

**R2.** Se garantiza debido a que el alumno ha hecho un resumen con su clave privada, por lo que el profesor podrá comprobarlo al comparar el resumen descifrado con la clave pública del alumno con el resumen creado por el profesor de los datos del examen que vienen en el paquete cifrados con la clave pública del profesor y que el profesor tendrá que descifrar con su clave privada (Firma digital).

**R3.** Se garantiza comprobando si los resúmenes producidos por el alumno y la autoridad de sellado se corresponde con el que el profesor construye para comprobar su validez.  
(Firma digital)

**R4.** Se garantiza debido a que cifran los resúmenes del examen con sus claves privadas.  
(Firma digital)

**R5.** Se garantiza debido a que al concatenar la fecha establecida por la autoridad de sellado y el resumen de alumno, esto debería coincidir con el resumen que ha generado la autoridad de sellado. Por lo que si modifica algo del examen, esta igualdad no se daría  
(Firma digital)

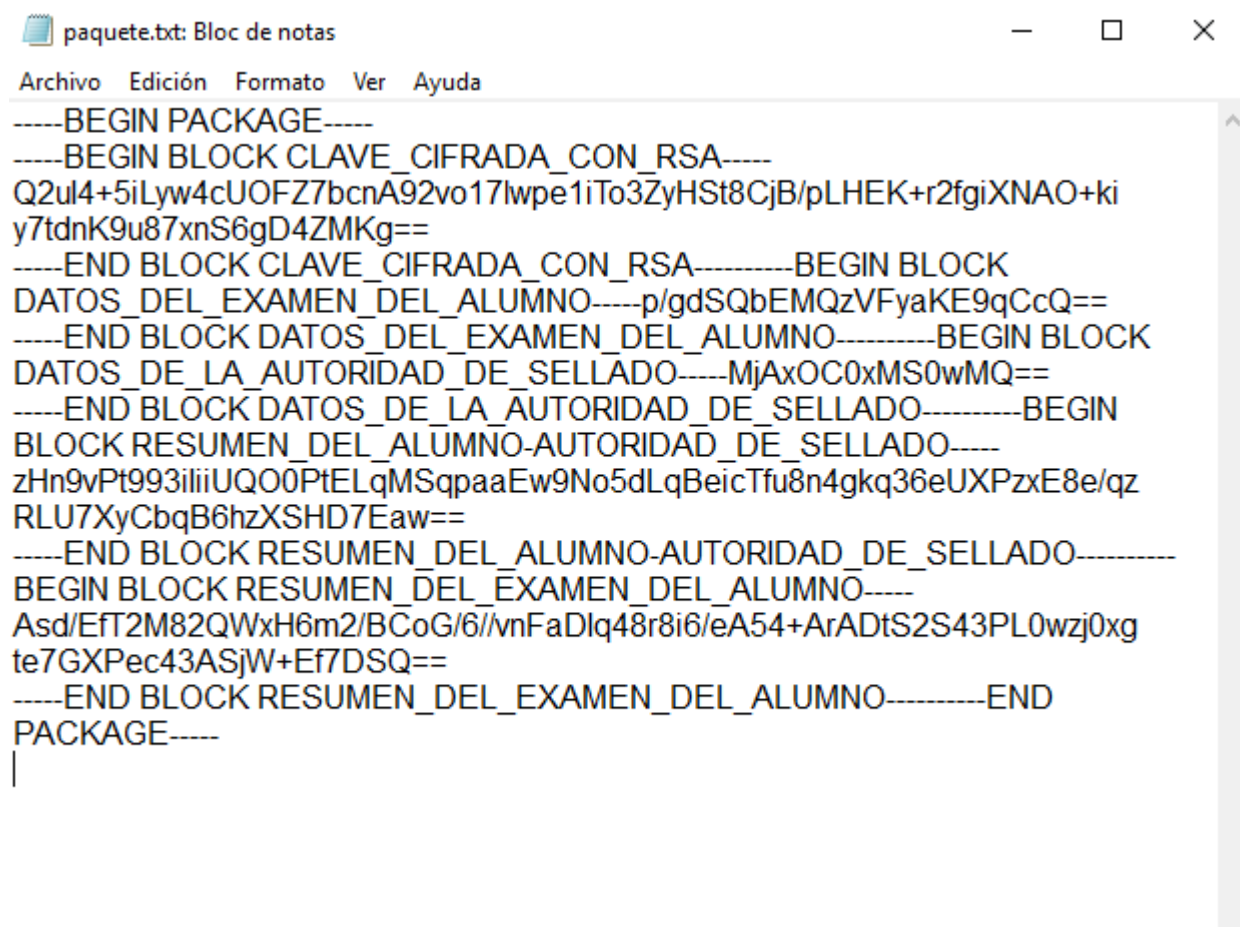
**R6.** Para cumplir con este punto, la autoridad de sellado utiliza el resumen generado por el alumno con su clave privada y le añade la fecha en la que se realizó el sellado y lo cifra con su clave privada. Esto asegura que únicamente la autoridad de sellado pudo cifrar este resumen (Firma digital).

**R7.** Se utilizará cifrado híbrido, ya que se utiliza cifrado simétrico para cifrar el examen, utilizando DES, y el cifrado asimétrico para cifrar las claves necesarias para cifrar y

descifrar el examen, utilizando RSA. Además la Autoridad de sellado, cifra su resumen con RSA utilizando como clave su clave privada, por lo que el profesor podrá obtener el resumen generado en la autoridad de sellado con la clave pública de la autoridad de sellado. (Cifrado Híbrido)

### 3. Descripción del formato/estructura del "paquete" resultante

El paquete resultante tiene la siguiente estructura:



```
paquete.txt: Bloc de notas
Archivo Edición Formato Ver Ayuda
-----BEGIN PACKAGE-----
-----BEGIN BLOCK CLAVE_CIFRADA_CON_RSA-----
Q2ul4+5iLyw4cUOFZ7bcnA92vo17lwpe1iTo3ZyHSt8CjB/pLHEK+r2fgiXNAO+ki
y7tdnK9u87xnS6gD4ZMKg==
-----END BLOCK CLAVE_CIFRADA_CON_RSA-----BEGIN BLOCK
DATOS_DEL_EXAMEN_DEL_ALUMNO-----p/gdSQbEMQzVFyaKE9qCcQ==
-----END BLOCK DATOS_DEL_EXAMEN_DEL_ALUMNO-----BEGIN BLOCK
DATOS_DE_LA AUTORIDAD_DE_SELLADO-----MjAxOC0xMS0wMQ==
-----END BLOCK DATOS_DE_LA AUTORIDAD_DE_SELLADO-----BEGIN
BLOCK RESUMEN_DEL_ALUMNO-AUTORIDAD_DE_SELLADO-----
zHn9vPt993iliiUQ00PtELqMSqpaaEw9No5dLqBeicTfu8n4gkq36eUXPzxE8e/qz
RLU7XyCbqB6hzXSHD7Eaw==
-----END BLOCK RESUMEN_DEL_ALUMNO-AUTORIDAD_DE_SELLADO-----
BEGIN BLOCK RESUMEN_DEL_EXAMEN_DEL_ALUMNO-----
Asd/Eft2M82QWxH6m2/BCoG/6/vnFaDIq48r8i6/eA54+ArADtS2S43PL0wzj0xg
te7GXPec43ASjW+Ef7DSQ==
-----END BLOCK RESUMEN_DEL_EXAMEN_DEL_ALUMNO-----END
PACKAGE-----
```

Este paquete contiene los datos cifrados después de ejecutar SellarExamen.

### 4. Descripción breve de la implementación

La implementación se hace en java y concretamente utilizando JCA(Java Cryptographic Architecture). El archivo está formado por los siguientes archivos:

- **Bloque.java:** encapsula cada uno de los bloques que forman parte de un paquete.
- **Paquete.java:** encapsula un paquete formado por varios bloques. Cada bloque

tiene un nombre (tipo String) y un contenido (tipo byte[]). La clase provee de métodos para añadir, listar y eliminar los bloques que conforman el paquete.

- **PaqueteDAO.java:** método estáticos que dan soporte a las operaciones de lectura y escritura de paquetes empleando un formato similar al PGP ASCII armor que usa la codificación Base64 para representar datos binarios mediante caracteres imprimibles.

Esta clase usa los objetos org.bouncycastle.util.encoders.Base64 de la librería BouncyCastle para la conversión a/desde Base64, por lo que para compilar y ejecutar es necesario incluir en el CLASSPATH el paquete JAR correspondiente.

- **GenerarClaves.java:** Usado para generar los pares de claves de los participantes: ALUMNO, AUTORIDAD SELLADO y PROFESOR. Se le pasa como argumento de línea de comando un identificador que se usará para componer los nombres de los archivos que se generarán. Genera dos ficheros: identificador.publica e identificador.privada, conteniendo, respectivamente, las claves pública y privada de ese usuario.
- **CifradorRSA:** se encarga del proceso de cifrado y descifrado de las claves.
- **CifradorDES:** se encarga del proceso de cifrado y descifrado utilizando el método DES.
- **EmpaquetarExamen:** Se le pasa en línea de comandos un fichero de texto con el contenido del examen a enviar, el nombre del paquete resultante y los ficheros con las claves necesarias para el empaquetado.
- **SellarExamen:** Se le pasa en línea de comandos el fichero el paquete a sellar y los ficheros con las clave/s criptográficas necesaria/s.

Al paquete recibido como argumento le vincula (añade) los bloques que correspondan para incorporar los datos aportados por la AUTORIDAD SELLADO y para garantizar la autenticidad de los datos de sellado.

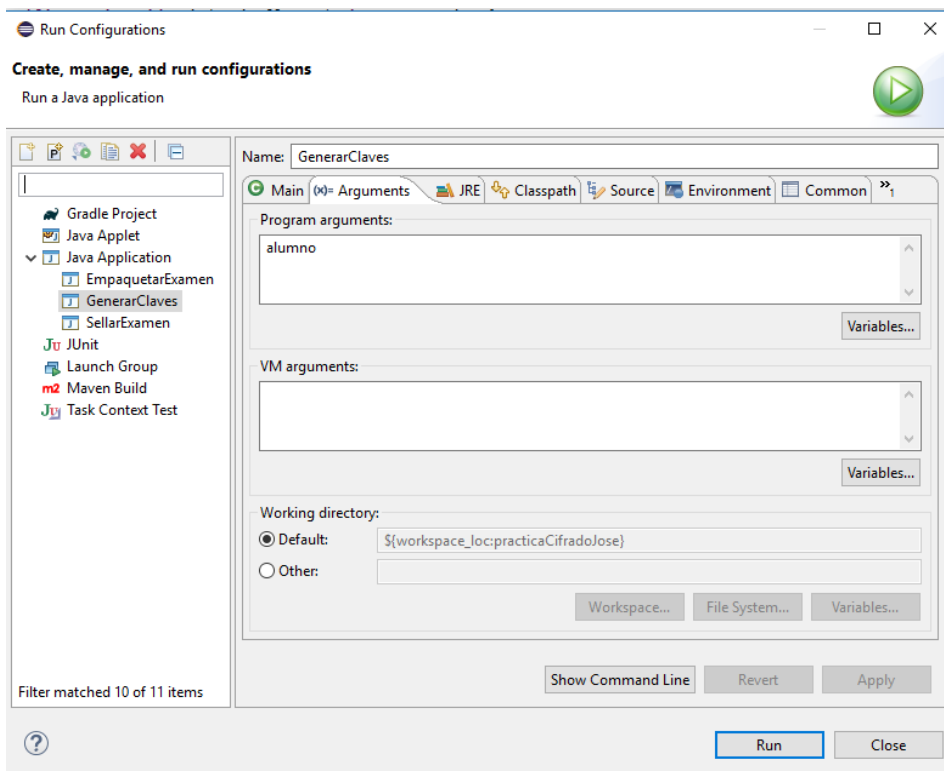
- **DesempaquetarExamen:** Se le pasa en línea de comandos el fichero con el paquete que representa al Examen Empaquetado (donde se incluyen los datos aportados por el ALUMNO y los datos de la AUTORIDAD SELLADO), el nombre del fichero donde se almacenará el examen en claro y los nombres de los ficheros con las claves que sean necesarias para desempaquetar y verificar la información que contiene el mencionado paquete.

Al usuario (PROFESOR) se le indicará por pantalla el resultado de las comprobaciones que se hayan realizado sobre el Examen Empaquetado y se mostrarán los datos que incluye.

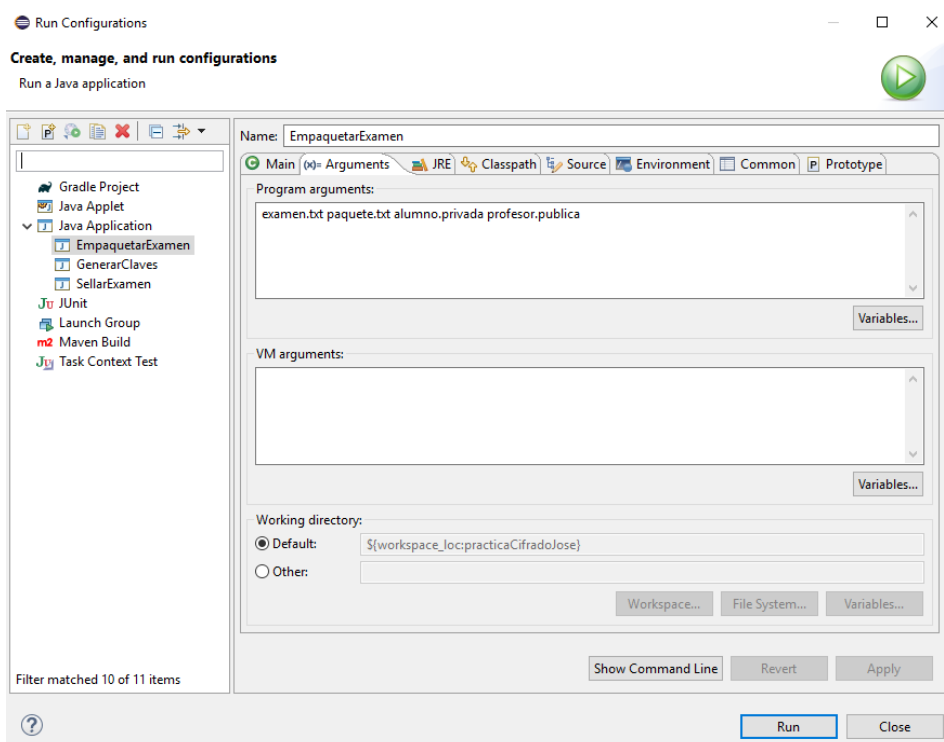
## ***5. Instrucciones de compilación y ejemplos de uso.***

Hay que ejecutarlo en orden para que funcione.

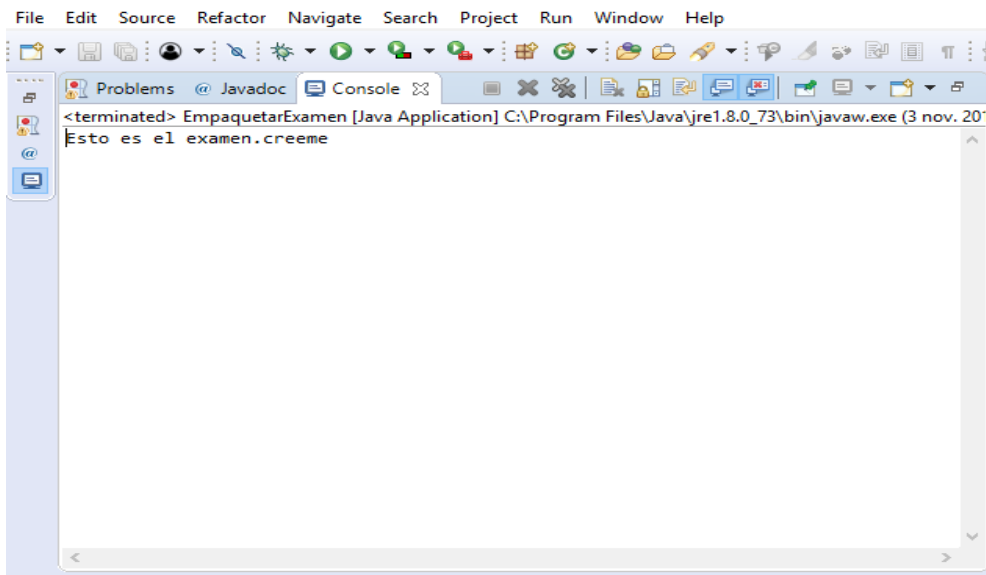
Primero habría que utilizar GenerarClave.java tres veces para generar las claves públicas y privadas de ALUMNO, PROFESOR y AUTORIDADSELLADO.



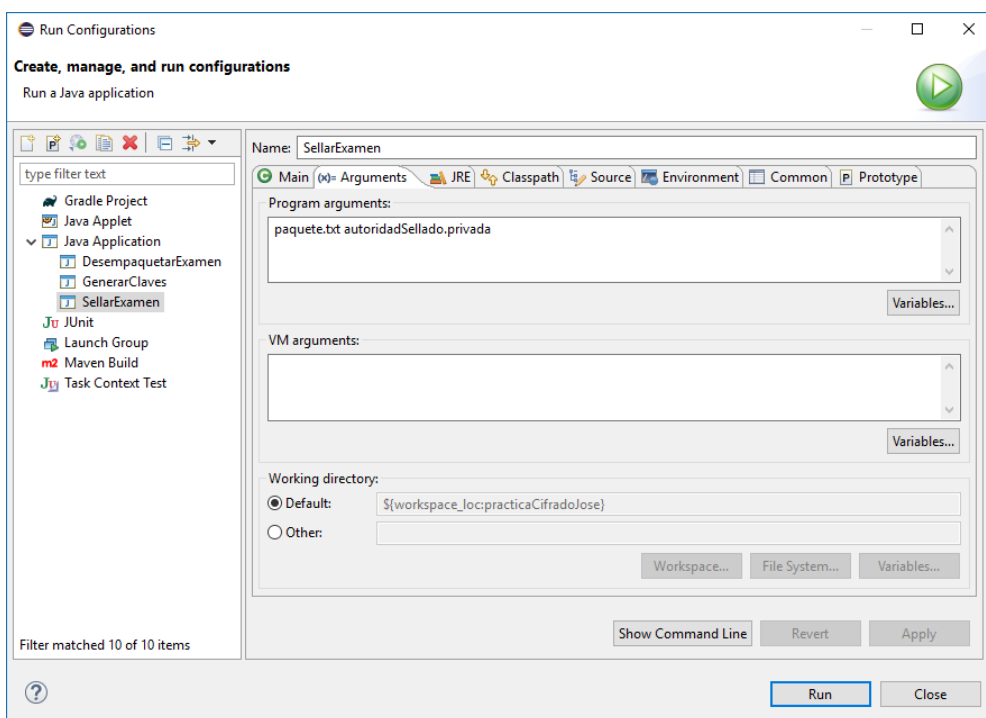
El siguiente paso será empaquetar el examen con EmpaquetarExamen.java. Para eso hay que pasarle cuatro argumentos:



Estos argumentos son el examen, el paquete donde se guardará el resultado, la clave privada del alumno y la clave pública del profesor. Al ejecutarse se muestra el contenido del examen.



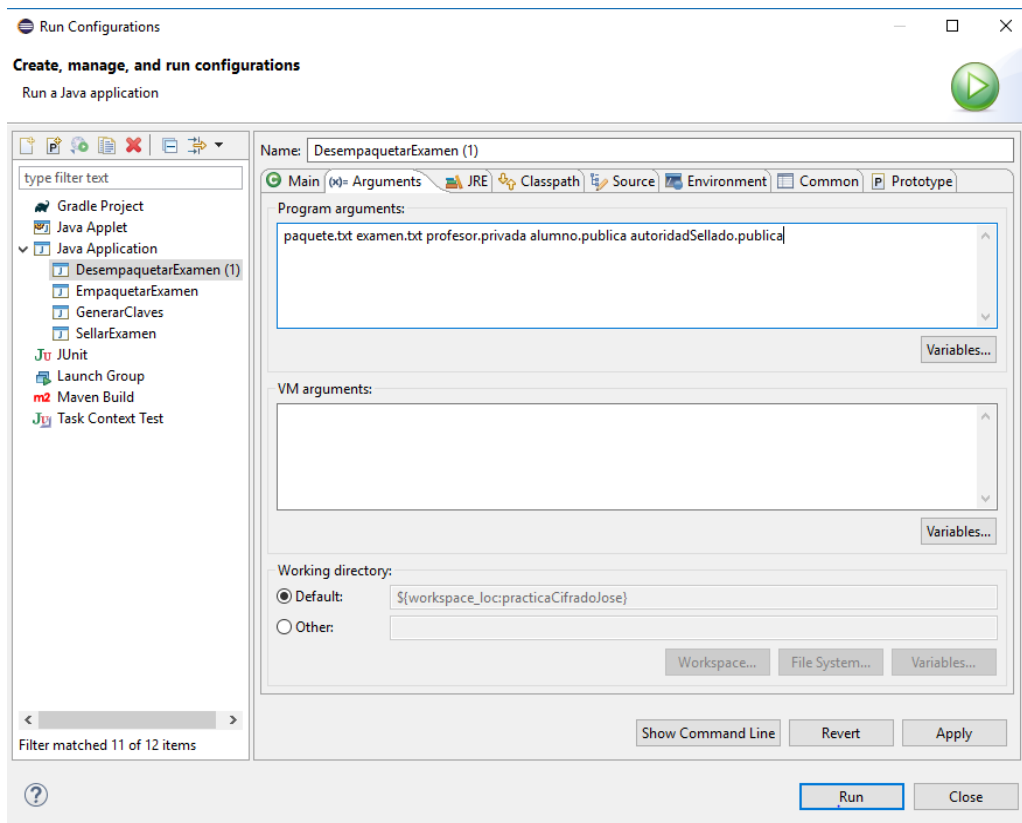
Al tener el paquete se procede a sellarlo con `SellarExamen.java`. Tenemos que pasarle el paquete anteriormente generado y la clave privada de `autoridadSellado`.



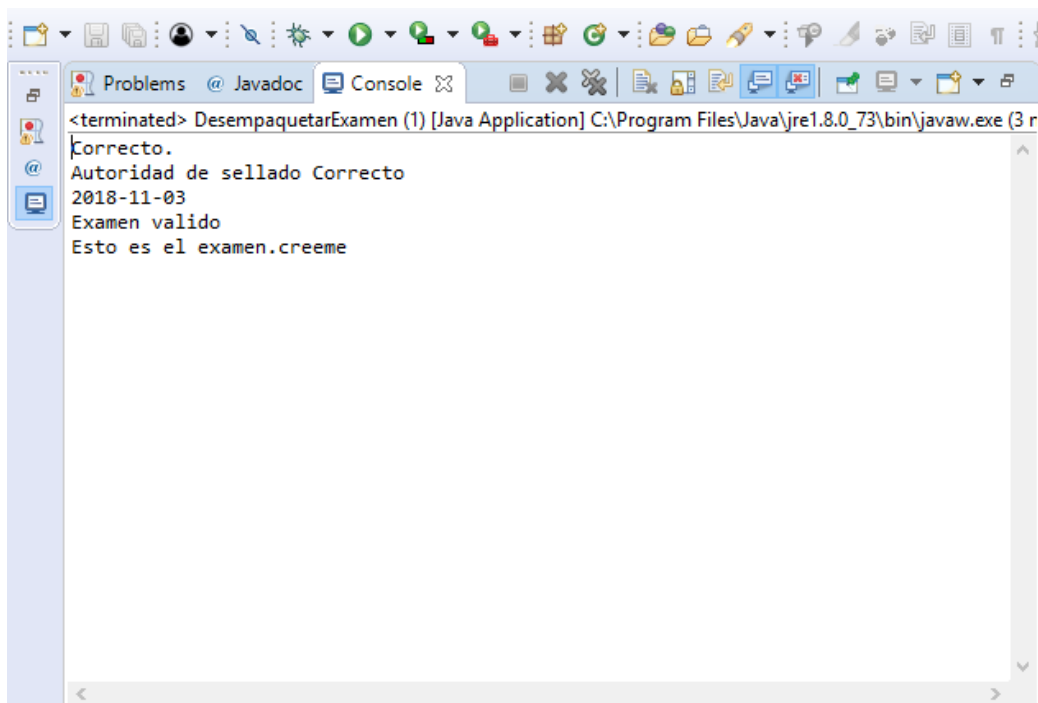
Por último faltaría ejecutar `DesempaquetarPaquete.java` para comprobar que todo ha funcionado correctamente.

Para esto hay que pasar el paquete, el archivo del examen, la clave privada del profesor, la clave pública del alumno y la clave pública de `AutoridadSellado`.

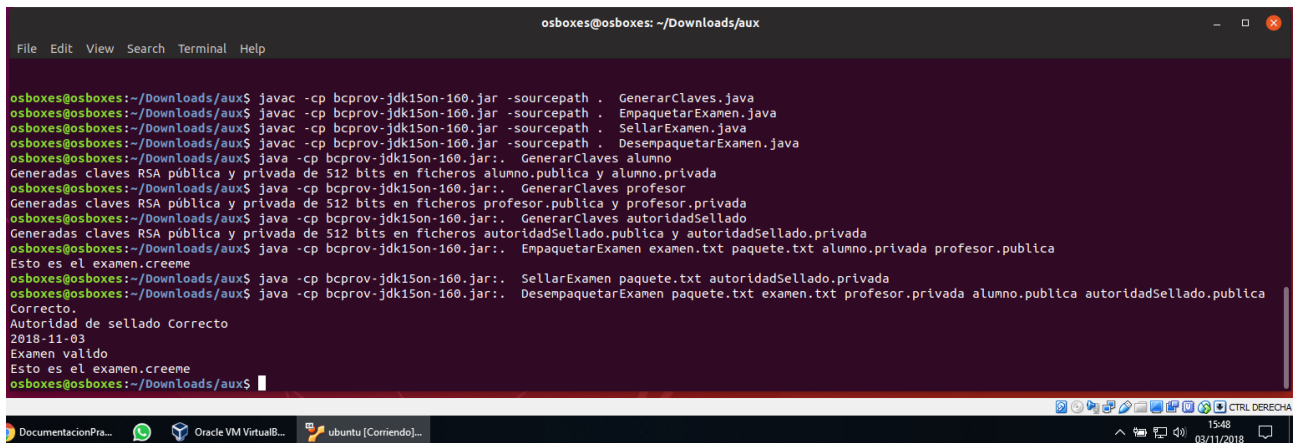




Si todo se ha ejecutado correctamente se muestra la siguiente salida con el contenido del examen para comprobar que no se ha modificado y la fecha.



Si utilizamos un sistema operativo Linux y queremos ejecutarlo por línea de comandos el orden sería así:



```
osboxes@osboxes: ~/Downloads/aux
File Edit View Search Terminal Help

osboxes@osboxes:~/Downloads/aux$ javac -cp bcprov-jdk15on-160.jar -sourcepath . GenerarClaves.java
osboxes@osboxes:~/Downloads/aux$ javac -cp bcprov-jdk15on-160.jar -sourcepath . EmpaquetarExamen.java
osboxes@osboxes:~/Downloads/aux$ javac -cp bcprov-jdk15on-160.jar -sourcepath . SellarExamen.java
osboxes@osboxes:~/Downloads/aux$ javac -cp bcprov-jdk15on-160.jar -sourcepath . DesempaquetarExamen.java
osboxes@osboxes:~/Downloads/aux$ java -cp bcprov-jdk15on-160.jar:. GenerarClaves alumno
Generadas claves RSA pública y privada de 512 bits en ficheros alumno.publica y alumno.privada
osboxes@osboxes:~/Downloads/aux$ java -cp bcprov-jdk15on-160.jar:. GenerarClaves profesor
Generadas claves RSA pública y privada de 512 bits en ficheros profesor.publica y profesor.privada
osboxes@osboxes:~/Downloads/aux$ java -cp bcprov-jdk15on-160.jar:. GenerarClaves autoridadSellado
Generadas claves RSA pública y privada de 512 bits en ficheros autoridadSellado.publica y autoridadSellado.privada
osboxes@osboxes:~/Downloads/aux$ java -cp bcprov-jdk15on-160.jar:. EmpaquetarExamen examen.txt paquete.txt alumno.privada profesor.publica
Esto es el examen.creeme
osboxes@osboxes:~/Downloads/aux$ java -cp bcprov-jdk15on-160.jar:. SellarExamen paquete.txt autoridadSellado.privada
osboxes@osboxes:~/Downloads/aux$ java -cp bcprov-jdk15on-160.jar:. DesempaquetarExamen paquete.txt examen.txt profesor.privada alumno.publica autoridadSellado.publica
Correcto.
Autoridad de sellado Correcto
2018-11-03
Examen valido
Esto es el examen.creeme
osboxes@osboxes:~/Downloads/aux$
```

## ***6. Breve comentario sobre las simplificaciones realizadas y sus consecuencias en una aplicación real***

Cada uno de los participantes (ALUMNO, AUTORIDAD SELLADO, PROFESOR) podrá generar sus propios pares de claves privada y pública que se almacenarán en ficheros (no se considera una protección del fichero con la clave privada, como sí ocurriría en una aplicación real). Si este fuera el caso en la vida real, todos los actores de esta situación podría acceder a la clave privada de los otros y realizar una suplantación de identidad.

No se contemplan los mecanismos de distribución de claves públicas. Se asumirá que todas las claves públicas necesarias estarán en poder del usuario que las necesite (ALUMNO, AUTORIDAD SELLADO o PROFESOR) de forma confiable (en una aplicación real se haría uso de certificados digitales y de una autoridad de certificación común).

El examen empaquetado se materializará físicamente en un fichero o paquete que contendrá toda la información que vayan incorporando los distintos participantes.

Las distintas piezas de información que aporte cada participante (ALUMNO o AUTORIDAD SELLADO) al Examen Empaquetado tendrán (antes del cifrado/firma y después del descifrado) la forma de Strings con codificación UTF8.

Las respuestas del ALUMNO estarán almacenadas inicialmente en un fichero de texto.

Por tanto, el principal problema de seguridad que tiene la aplicación es el almacenamiento de las claves y que debería utilizar certificados digitales y una autoridad de certificación común. De esta forma se parecería mucho más a una aplicación de uso real.

## ***7.Resultados obtenidos y conclusiones***

Obtenemos como resultado una aplicación que nos permite generar claves, empaquetar un fichero, utilizar una autoridad de sellado para sellar el paquete y por último poder desempaquetarlo utilizando las claves necesarias.

Es una práctica útil para saber cómo funciona el cifrado, y concretamente para utilizar la API de cifrado de java JCA, pero por las simplificaciones vistas anteriormente se puede decir que no sería una aplicación segura en el mundo real. Una manera de mejorarla sería implementar el uso de certificados digitales y utilizar una autoridad de certificación común.