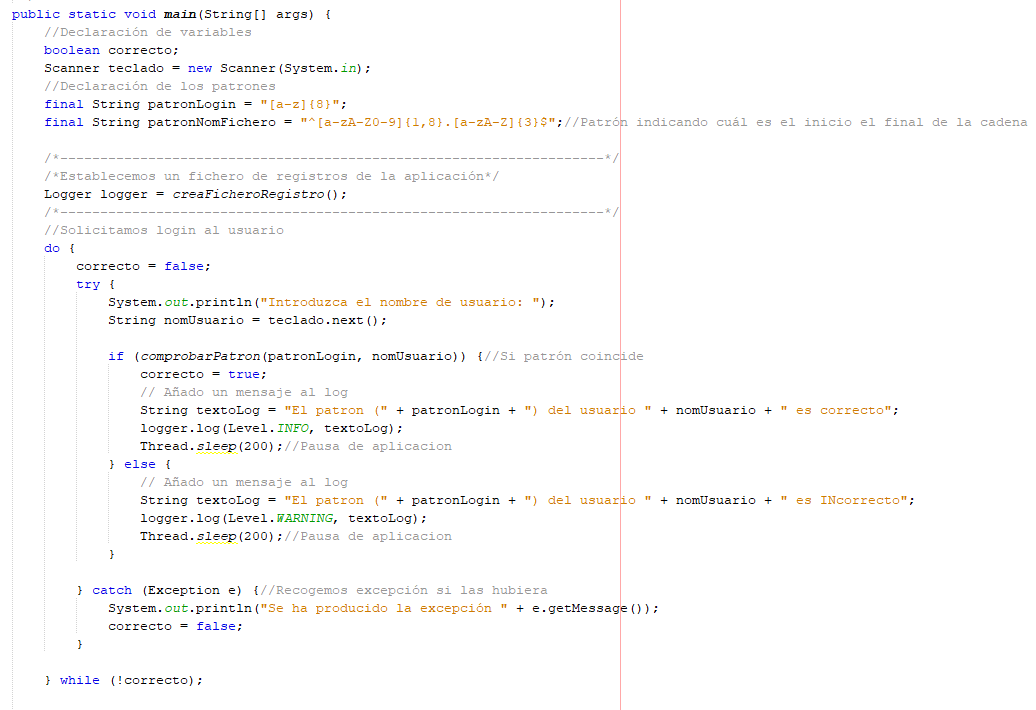
**ACTIVIDAD 6.1**

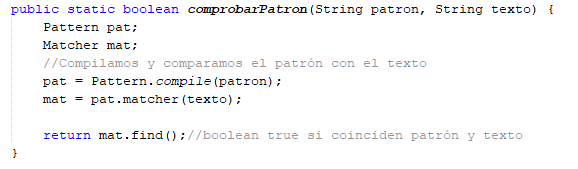
En este apartado de la tarea se nos pide que realicemos una aplicación que cumpla los siguientes requerimientos:

* **Solicita el nombre del usuario que va a utilizar la aplicación. El login tiene una longitud de 8 caracteres y está compuesto únicamente por letras minúsculas.**

Dentro del proyecto que hemos creado para la tarea, añadimos una clase principal a la que llamaremos “Aplicación Principal” y que contendrá la lógica del programa. Dentro del método main vamos a incluir un bloque de código para solicitar el login al usuario, y evaluar si este tiene el formato requerido según un patrón dado:



*Solicitamos el login, lo recogemos en una variable y este se lo pasamos como parámetro al método creado “comprobarPatron” que nos devolverá true o false según coincida o no el login dado con el patrón. A su vez contemplamos también cualquier excepción metida dentro de un bucle para que el programa regrese a solicitar usuario en caso de que salte alguna excepción en la recepción de los datos y escribimos en el log la información cada vez que se realiza alguna acción importante.*



*Este es el método que nos compara patrón con texto recibido, en este caso el login*

* **Solicita al usuario el nombre de un fichero que quiere mostrar. El nombre del fichero es como máximo de 8 caracteres y tiene una extensión de 3 caracteres.**

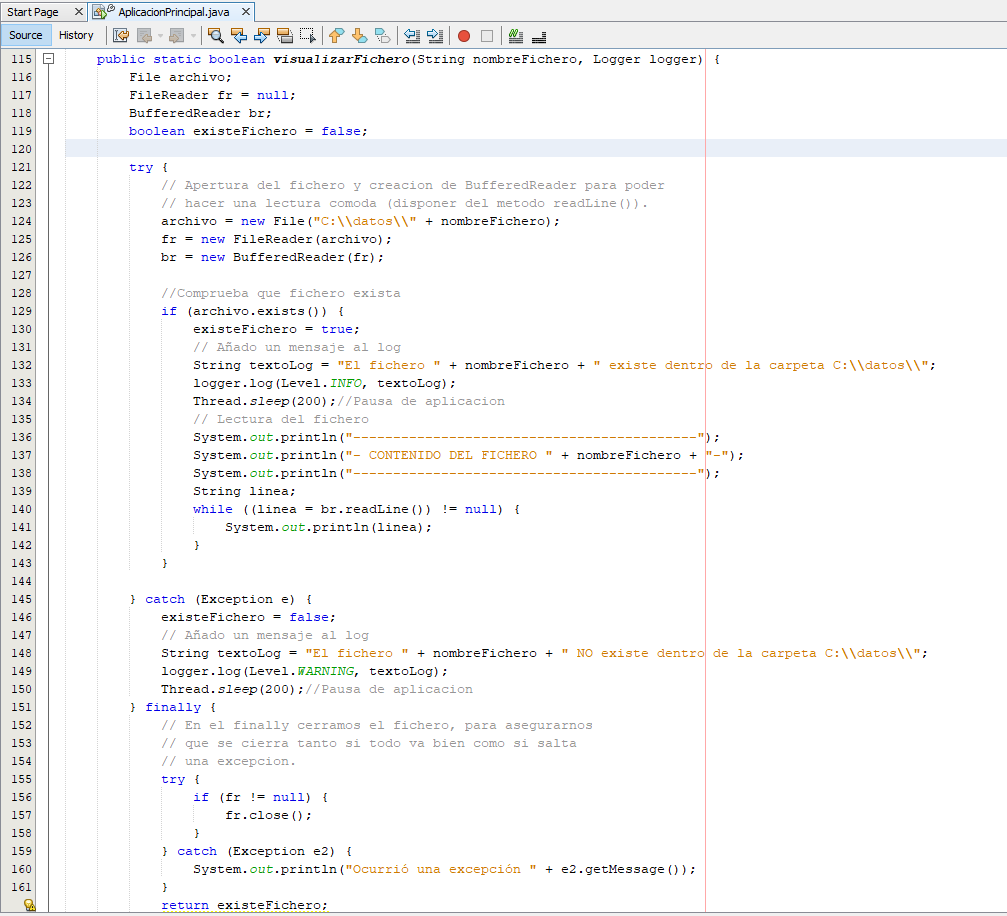
Creamos este bloque a continuación del anterior:



*Como en caso anterior solicitamos el nombre del fichero, lo recogemos y se lo pasamos como parámetro al método “comprobarPatron” junto con el patrón de fichero propuesto en la tarea. Si el patrón es correcto llamamos al método “visualizarFichero” para que muestre su contenido. Contempla también cualquier excepción controlada por un bucle para que el programa sepa reponerse del error y escribe en el fichero log cualquier información importante que se vaya produciendo.*

* **Visualiza en pantalla el contenido del fichero**

Este apartado lo he metido en un método al que he llamado “visualizarFichero”, recoge como parámetro el Logger y el nombre del fichero:



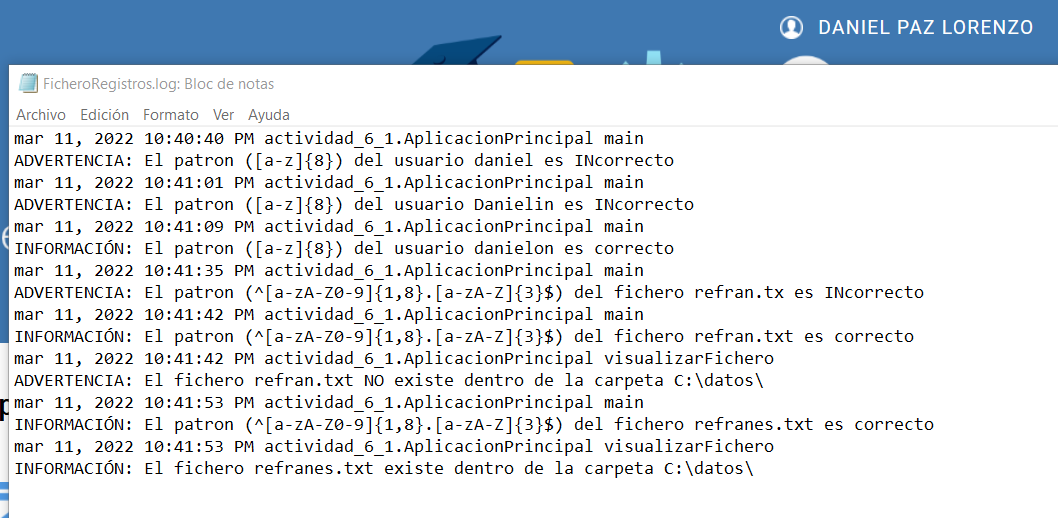
*Evalúa si el fichero existe o no, si es que sí lo muestra por pantalla. Se recoge cualquier tipo de excepción y se recoge en el fichero log las acciones más importantes que van sucediéndose*

PRUEBAS DE LA APLICACIÓN

Ejecutamos, nos va a solicitar nombre de usuario y nombre de fichero, probamos con 3 intentos (2 incorrectos y 1 correcto), así podemos hacernos una idea del comportamiento y si funciona correctamente:



*Después de introducir los datos en rojo se muestran los mensajes de log para mostrar información de las acciones que se van sucediendo. A su vez estos mensajes se van a registrar en el fichero de log que hemos creado. Finalmente se muestra el contenido del fichero por pantalla*

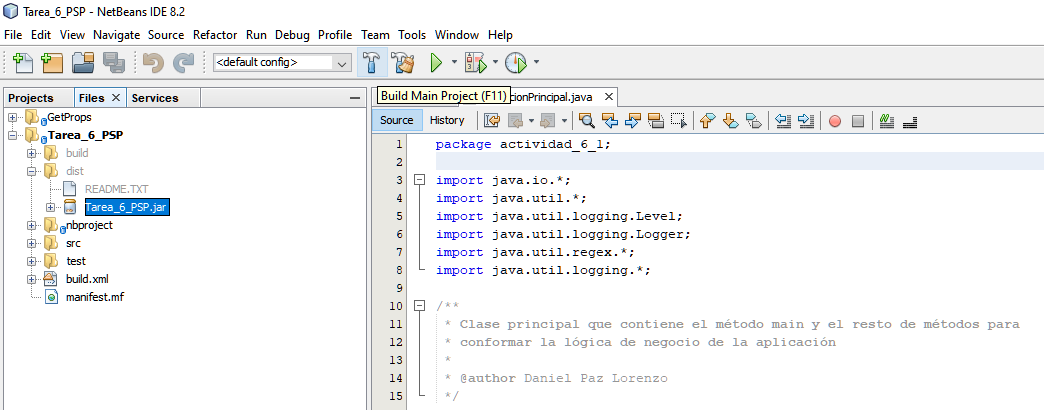


*Este es el fichero de log que se ha ido generando cuando corríamos la aplicación*

**ACTIVIDAD 6.2**

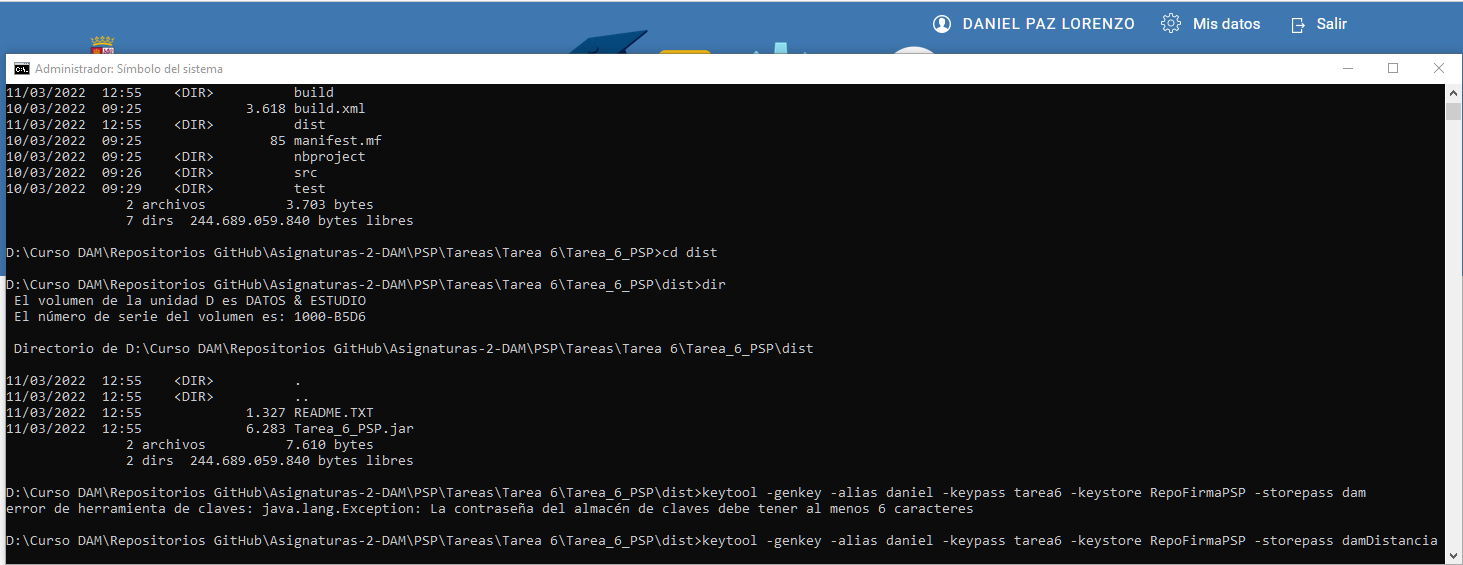
En este apartado se nos pide que utilizando la aplicación que hemos creado anteriormente, la firmemos digitalmente y que solo pueda leer los datos del directorio “c:/datos”.

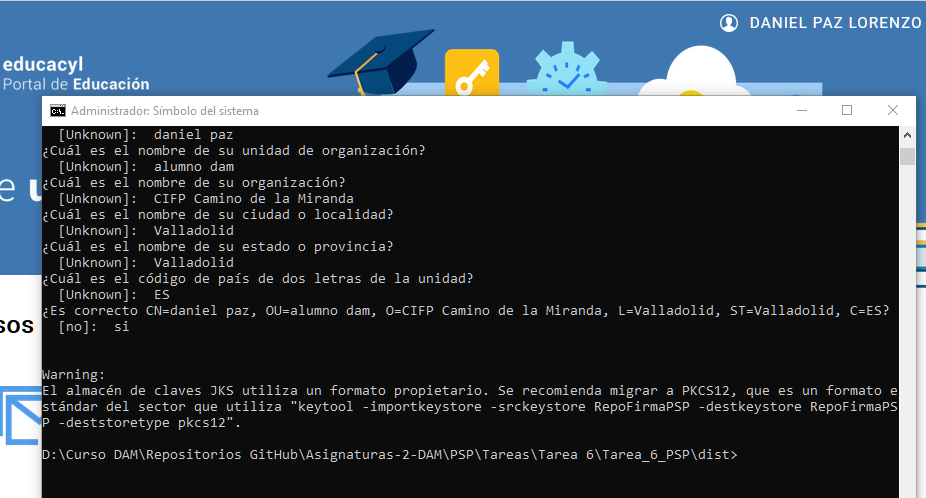
Comenzamos compilando la aplicación en NetBeans:



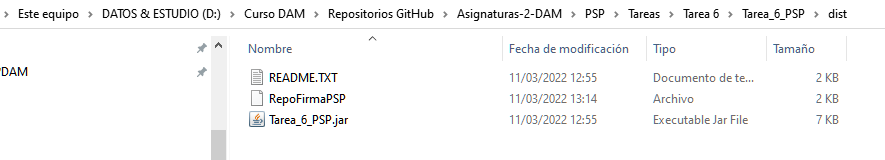
*Podemos ver en el navegador de la izquierda como se nos ha generado el paquete .jar*

A continuación, abrimos la consola de comandos de Windows en modo administrador, nos dirigimos al directorio del proyecto y generamos un par de claves (privada y pública) para firmar el fichero con el comando “keytool”:



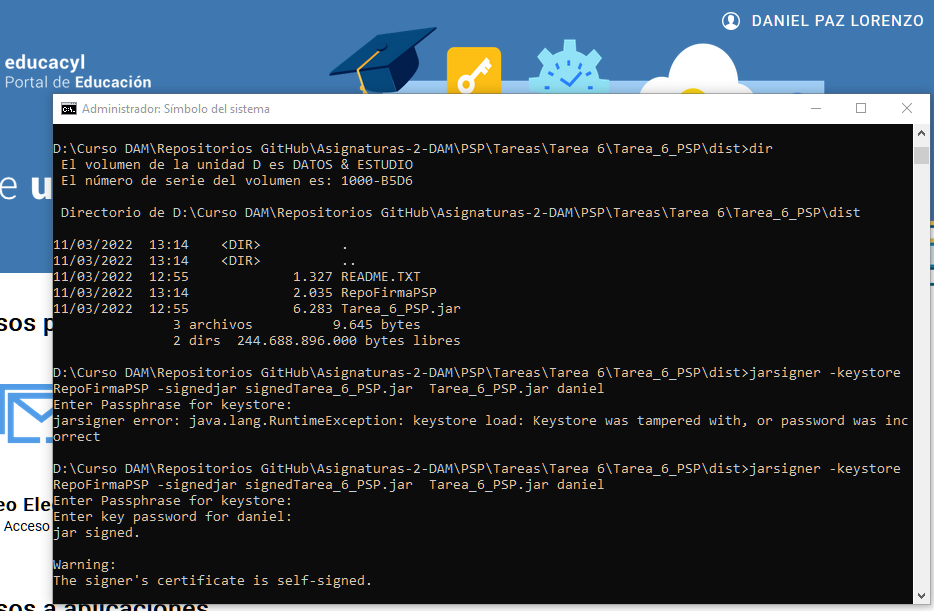


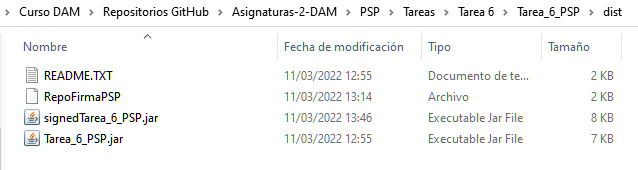
*Rellenamos los datos que nos van solicitando*



*Si entramos en el directorio ya vemos que se nos ha creado el almacenamiento de claves*

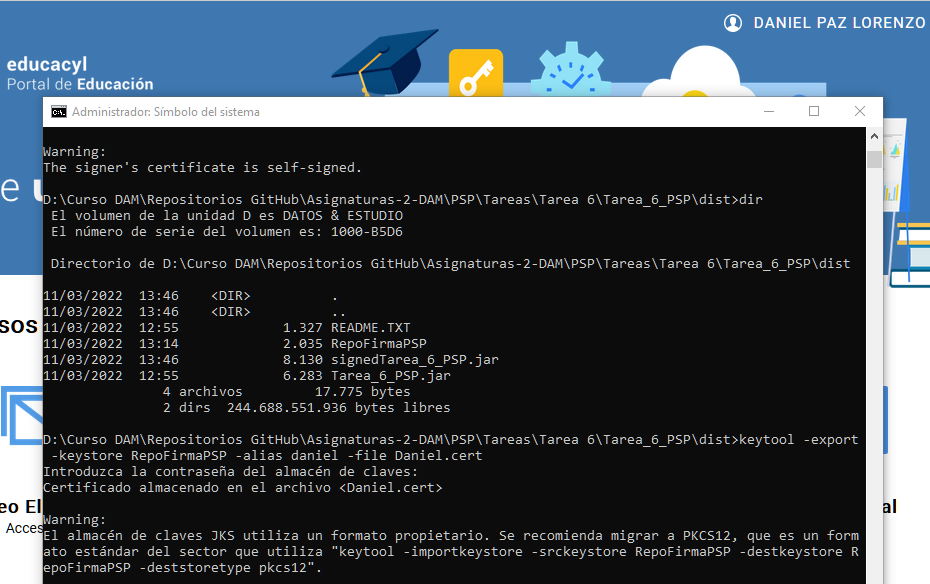
Por último, vamos a firmar la aplicación con el certificado que hemos creado con el comando “jarsigner”:



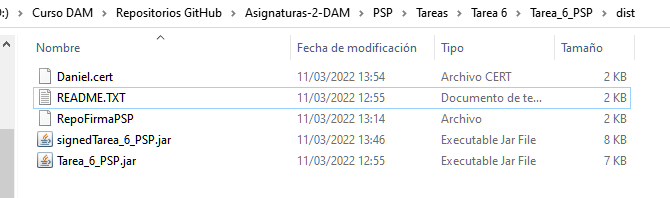


*Comprobamos que dentro del directorio se nos ha creado un nuevo paquete .jar ya firmado digitalmente*

Ya solo nos queda exportar la llave pública de nuestro certificado para que un usuario externo pueda utilizar la aplicación validando mi firma digital. Para ello utilizamos nuevamente el comando “keytool” con el atributo “-export” de la siguiente manera:

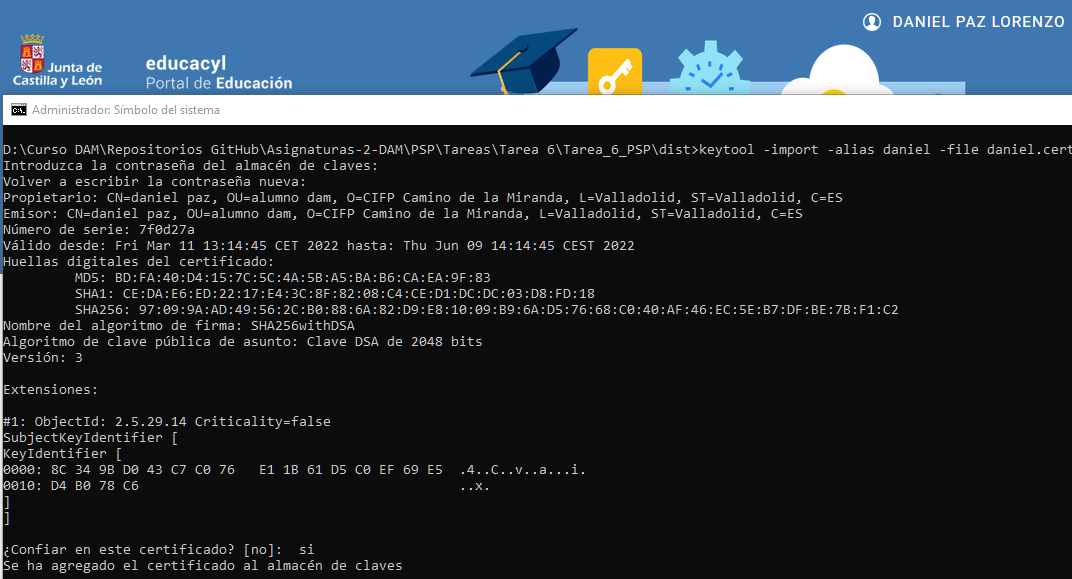


Volvemos al directorio para comprobar que se nos ha generado el certificado:



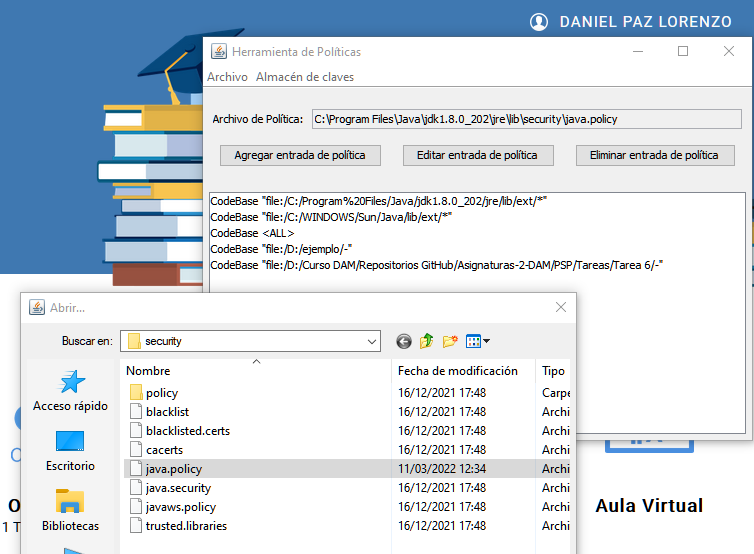
El siguiente apartado nos pide que la aplicación solo pueda leer los datos del directorio “c:/datos”, para ello tendremos que configurar la política de seguridad de nuestra aplicación firmada.

Lo primero que haremos será importar el certificado para validar la firma del fichero utilizando el comando “keytool -import”:

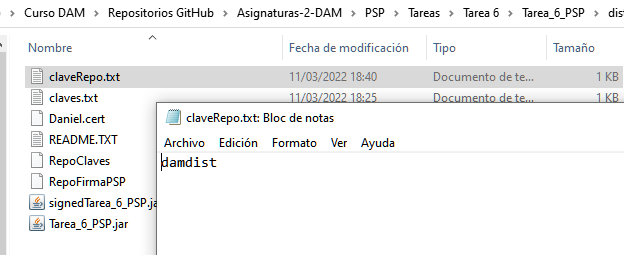


*Al ejecutar nos solicita contraseña para el nuevo almacén de claves creado*

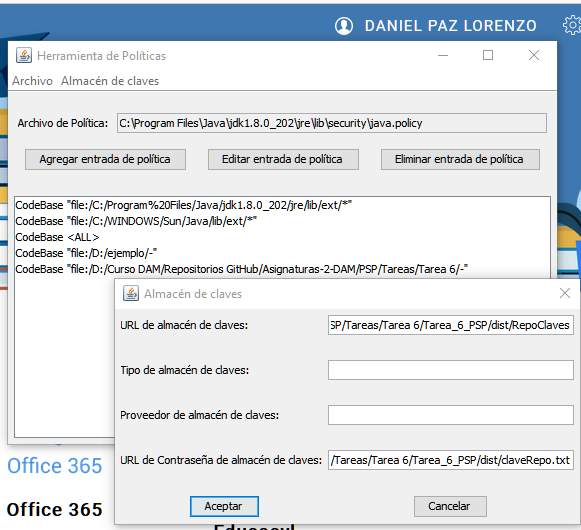
Pasamos a configurar la política de seguridad para que permita ejecutar el fichero firmado:



*Abrimos la herramienta policitytool, pulsamos Archivo->Abrir, seleccionamos java.policy y lo abrimos*

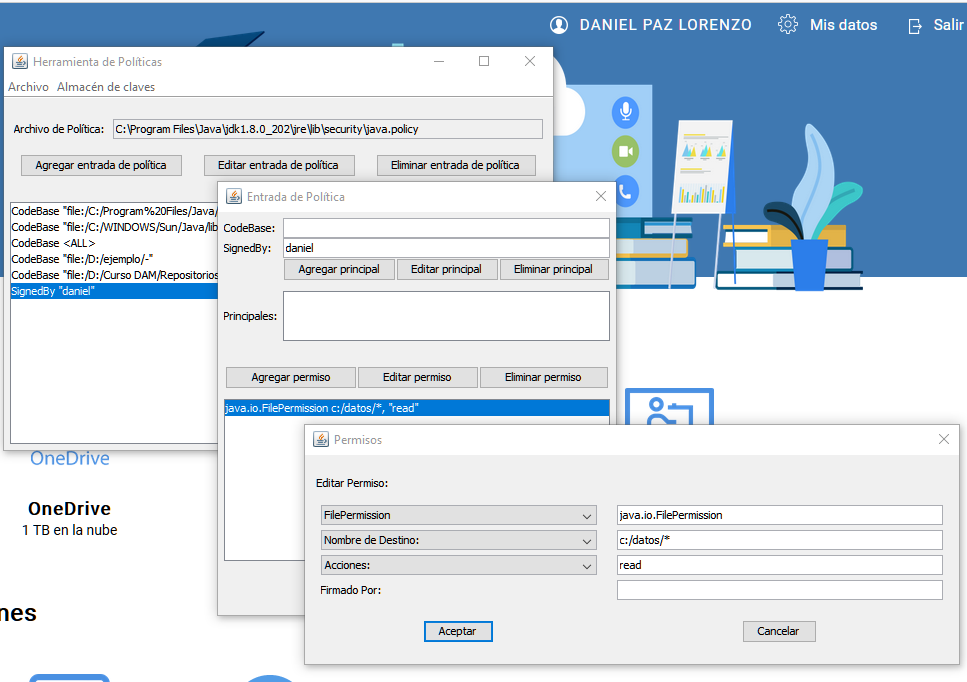


*Creamos un .txt dentro del mismo directorio que contenga la clave del repositorio de claves*



*Abrimos el almacén de claves y metemos las 2 url del repositorio de las claves y del fichero con la contraseña del repositorio*

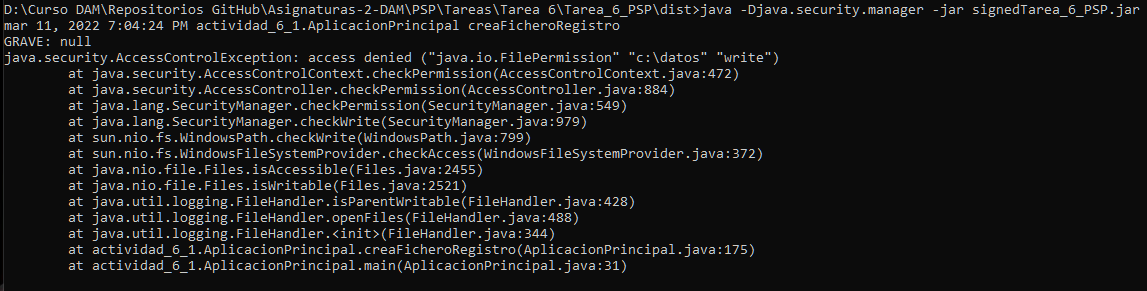
Una vez tenemos importado el almacén de claves vamos a agregar una entrada de política para que la aplicación pueda acceder a la carpeta C:/datos:



*Establecemos los parámetros necesarios y guardamos*

Ya solo nos queda comprobar que hemos configurado correctamente la política de seguridad ejecutando la aplicación.

1. Este primer pantallazo corresponde a una ejecución anterior a configurar las políticas de seguridad donde podemos ver que tenemos denegado el acceso a c:/datos:



1. Después de configurar y guardar la nueva política de seguridad ejecutamos la aplicación y vemos como ahora si no nos muestra ningún mensaje de error y podemos ejecutar la aplicación sin restricciones:

