



PRACTICA DE JAVA DE ENCRIPCIÓN ASIMETRICA

Enunciat

En aquesta segona pràctica provarem d'encriptar i desencriptar arxius amb programes escrits en Java. En el moodle del mòdul 9 trobaràs arxius per crear claus RSA, per encriptar i desencriptar, sino et funcionen, busca'ls per internet.

Pràctica

1. Busca per internet codi font en Java per crear claus pública i privada.

Clase Java Crear Clases

```
package practica_criptografia_asimetrica;

import java.security.*;
import javax.crypto.*;
import javax.crypto.spec.*;
import java.util.*;
import java.io.*;

public class CrearClaves implements Constantes {

    public static void main(String[] args) throws Exception {

        // Generamos las claves publica/privada
        SecureRandom sr = new SecureRandom();
        sr.setSeed(new Date().getTime());
        System.out.println("Generando claves...");
        KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
        kpg.initialize(TAMANO_CLAVE_RSA, sr);
        KeyPair par_claves = kpg.generateKeyPair();
        System.out.println("Claves generadas");

        // Generamos el fichero de la clave publica
        System.out.print("Indique fichero para" + " la clave publica:");
        BufferedReader teclado = new BufferedReader(new
        InputStreamReader(System.in));
        String fichero_publica;
        fichero_publica = teclado.readLine();
        FileOutputStream fos = new FileOutputStream(fichero_publica);
        fos.write(par_claves.getPublic().getEncoded());
        fos.close();
        System.out.println("Fichero con clave publica generado");

        // Generamos el fichero de clave privada
```



Generalitat de Catalunya
Departament d'Educació
Institut d'Educació Secundària i Superior
d'Ensenyaments Professionals
Esteve Terradas i Illa
Departament d'Informàtica

```
System.out.print("Indique fichero para la clave privada:");
String fichero_privada;
fichero_privada = teclado.readLine();
System.out.print("La clave privada debe estar encriptada, indique
password con la que encriptarla:");
char[] password;
password = teclado.readLine().toCharArray();

// Encriptamos con un PBE
byte[] salt = new byte[TAMANO_SALT_BYTES];
sr.nextBytes(salt);
PBEKeySpec clave_pbe = new PBEKeySpec(password);
SecretKey clave_secreta_pbe =
SecretKeyFactory.getInstance("PBEWithMD5AndDES").generateSecret(clave_pbe);
PBEPParameterSpec pbe_param = new PBEPParameterSpec(salt, ITERACIONES_PBE);
Cipher cifrador_pbe = Cipher.getInstance("PBEWithMD5AndDES");
cifrador_pbe.init(Cipher.ENCRYPT_MODE, clave_secreta_pbe, pbe_param);
byte[] clave_privada_cifrada =
cifrador_pbe.doFinal(par_claves.getPrivate().getEncoded());
fos = new FileOutputStream(fichero_privada);
fos.write(salt);
fos.write(clave_privada_cifrada);
fos.close();
System.out.println("Fichero con clave privada generado");
}
}
```

Crear Implements Constants

```
package practica_criptografia_asimetrica;

public interface Constantes
{
    public static final int ITERACIONES_PBE = 1000;
    public static final int TAMANO_CLAVE_RSA = 1024;
    public static final int TAMANO_CLAVE_SESSION = 128;
    public static final int TAMANO_SALT_BYTES = 8;
    public static final int TAMANO_IV_BYTES = 8;
}
```



Generalitat de Catalunya
Departament d'Educació
Institut d'Educació Secundària i Superior
d'Ensenyaments Professionals
Esteve Terradas i Illa
Departament d'Informàtica

2. Executa el programa i fes una demostració amb captures de:

- Generació de clau publica

```
1 package practica_criptografia_asimetrica;
2
3 import java.security.*;
4 import javax.crypto.*;
5 import javax.crypto.spec.*;
6 import java.util.*;
7 import java.io.*;
8
9 public class CrearClaves implements Constantes {
10
11     public static void main(String[] args) throws Exception {
12
13         // Generamos las claves publica/privada
14         SecureRandom sr = new SecureRandom();
15         sr.setSeed(new Date().getTime());
16         System.out.println("Generando claves...");
17         KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
18         kpg.initialize(TAMANO_CLAVE_RSA, sr);
19         KeyPair par_claves = kpg.generateKeyPair();
20         System.out.println("Claves generadas");
21
22         // Generamos el fichero de la clave publica
23         System.out.print("Indique fichero para " + " la clave publica:");
24         BufferedReader teclado = new BufferedReader(new InputStreamReader(System.in));
25         String fichero_publica;
26         fichero_publica = teclado.readLine();
27         FileOutputStream fos = new FileOutputStream(fichero_publica);
28     }
29 }
```

Problems Javadoc Declaration Console

<terminated> CrearClaves [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (19 dic. 2019 16:31:30)

Generando claves...
Claves generadas
Indique fichero para la clave publica:ficheroClavePublica
Fichero con clave publica generado
Indique fichero para la clave privada:ficheroClavePrivada
La clave privada debe estar encriptada, indique password con la que encriptarla:1234
Fichero con clave privada generado

←

→

⌵

⬆

AMS2 > M9_Programacion_de_servicios_y_procesos > UF1 > Practica_Criptografia_Asimetrica_VALLEJ

Examen_UF1_VA

MPDUAL_Videoj

Practica_Cifratxe

UF1

OneDrive

Este equipo

Descargas

Documentos

Escritorio

Imágenes

Música

Objetos 3D

Vídeos

Disco local (C:)

Nombre

Fecha de modificación

Tipo

Tamaño

.settings

bin

src

.classpath

.gitignore

.project

ficheroClavePrivada

ficheroClavePublica

19/12/2019 16:14

19/12/2019 16:15

19/12/2019 16:15

19/12/2019 16:14

19/12/2019 16:14

19/12/2019 16:14

19/12/2019 16:31

19/12/2019 16:31

Carpeta de archivos

Carpeta de archivos

Carpeta de archivos

Archivo CLASSPATH

Archivo de origen ...

Archivo PROJECT

Archivo

Archivo

1 KB

1 KB

1 KB

1 KB

1 KB

8 elementos | 1 elemento seleccionado 162 bytes |



Generalitat de Catalunya
Departament d'Educació
Institut d'Educació Secundària i Superior
d'Ensenyaments Professionals

Esteve Terradas i Illa

Departament d'Informàtica

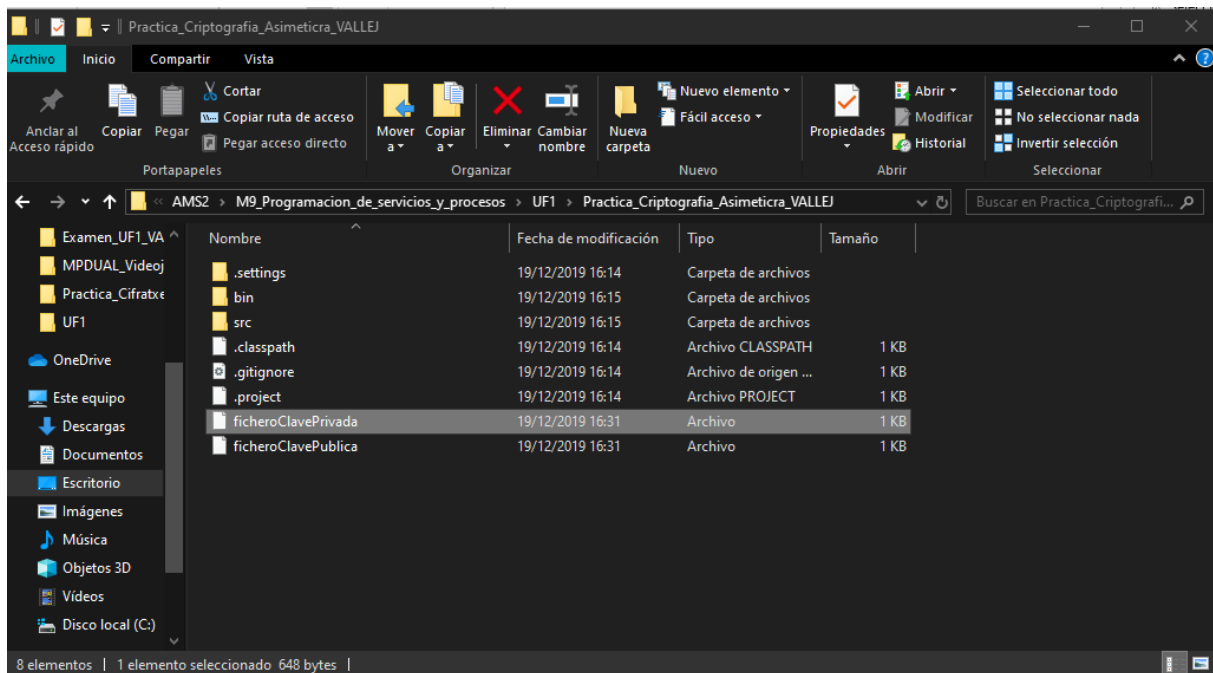
- Generació de clau privada

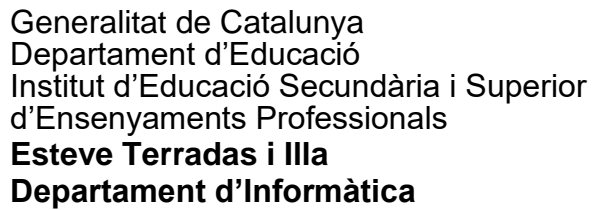
```
J EncriptaFichero.java J DesencriptaFichero.java J DesencriptaFichero.java J EncriptaFichero.java J MainClass.java J CrearClaves.java J Constantes.java
1 package practica_criptografia_asimetrica;
2
3 import java.security.*;
4 import javax.crypto.*;
5 import javax.crypto.spec.*;
6 import java.util.*;
7 import java.io.*;
8
9 public class CrearClaves implements Constantes {
10
11     public static void main(String[] args) throws Exception {
12
13         // Generamos las claves publica/privada
14         SecureRandom sr = new SecureRandom();
15         sr.setSeed(new Date().getTime());
16         System.out.println("Generando claves...");
17         KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA");
18         kpg.initialize(TAMANO_CLAVE_RSA, sr);
19         KeyPair par_claves = kpg.generateKeyPair();
20         System.out.println("Claves generadas");
21
22         // Generamos el fichero de la clave publica
23         System.out.print("Indique fichero para " + " la clave publica:");
24         BufferedReader teclado = new BufferedReader(new InputStreamReader(System.in));
25         String fichero_publica;
26         fichero_publica = teclado.readLine();
27         FileOutputStream fos = new FileOutputStream(fichero_publica);
28     }
29 }
```

Problems Javadoc Declaration Console

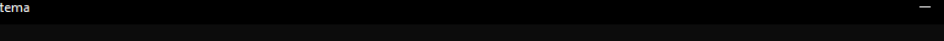
terminated> CrearClaves [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (19 dic. 2019 16:31:30)

Generando claves...
Claves generadas
Indique fichero para la clave publica:ficheroClavePublica
Fichero con clave publica generado
Indique fichero para la clave privada:ficheroClavePrivada
La clave privada debe estar encriptada, indique password con la que encriptarla:1234
Fichero con clave privada generado





- ```
EncrptFichero.java DescriptFichero.java DescriptFichero.java EncrptFichero.java MainClass.java CrearClaves.java Constantes.java EncrptFichero
11
12 public static void main(String[] args) throws Exception {
13
14 // Pedimos el fichero a encriptar
15 // y fichero de clave publica a usar
16 BufferedReader teclado = new BufferedReader(new InputStreamReader(System.in));
17 System.out.print("Indique fichero a encriptar:");
18 String fichero_encriptar = teclado.readLine();
19 if (!new File(fichero_encriptar).exists()) {
20 System.out.println("El fichero " + fichero_encriptar + " no existe");
21 return;
22 }
23 String fichero_encriptado = fichero_encriptar + ".crypto";
24 System.out.print("Indique que fichero tiene la" + " clave publica a usar:");
25 String fichero_publica = teclado.readLine();
26
27 // Recuperamos la clave publica
28 FileInputStream fis = new FileInputStream(fichero_publica);
29 byte[] buffer = new byte[fis.available()];
30 fis.read(buffer);
31 X509EncodedKeySpec clave_publica_spec = new X509EncodedKeySpec(buffer);
32 KeyFactory kf = KeyFactory.getInstance("RSA");
33 PublicKey clave_publica = kf.generatePublic(clave_publica_spec);
34
35 // Generamos el fichero encriptado
36 SecureRandom sr = new SecureRandom();
37 sr.setSeed(new Date().getTime());
38 fis = new FileInputStream(fichero_encriptar);
39
40 Problems
41 Javadoc
42 Declaration
43 Console
44 terminated: EncrptFichero [Java Application] C:\Program Files\Java\jre1.8.0_191\bin\javaw.exe (19 dic. 2019 16:42:00)
45 Indique fichero a encriptar: ficheroParaEncriptar.txt
46 Indique que fichero tiene la clave publica a usar: ficheroClavePublica
47 Generando clave de sesion...
48 Guardando la clave de sesion encriptada...
49 Guardando ficheroParaEncriptar.txt en el fichero encriptadoficheroParaEncriptar.txt.crypto
50 Fichero encriptado correctamente
```



```
C:\Users\vcj\on\Desktop\AMS2\M9_Programacion_de_servicios_y_procesos\UF1\Practica_Criptografia_Asimetrica_VALLEJ>type ficheroParaEncriptar.txt.crypt
C]p÷X4âãUÖEP";xT%±ää³ñ%0B¡aÁ!üÖ 9 V+CB!a÷("P3B÷0müeoü!Bb/añsëÇÜw-EXt!■ēŠñ`T^3;]!ðüâ2c ¶Ö!BEBÜ m%1't['Z'g!B!afU
r8È!i*pBÜ !e;"iôt'ÖBø%ÊÖ/a
C:\Users\vcj\on\Desktop\AMS2\M9_Programacion_de_servicios_y_procesos\UF1\Practica_Criptografia_Asimetrica_VALLEJ>
```



Generalitat de Catalunya  
Departament d'Educació  
Institut d'Educació Secundària i Superior  
d'Ensenyaments Professionals  
**Esteve Terradas i Illa**  
Departament d'Informàtica

- Desenscriptació d'un text o arxiu

```
34 // Recuperamos la clave privada
35 System.out.println("Recuperando clave privada...");
36 SecureRandom sr = new SecureRandom();
37 sr.setSeed(new Date().getTime());
38 FileInputStream fis = new FileInputStream(fichero_privada);
39 byte[] buffer = new byte[TAMANO_SALT_BYTES];
40 fis.read(buffer);
41 PBEKeySpec clave_pbe_spec = new PBEKeySpec(password);
42 SecretKey clave_pbe = SecretKeyFactory.getInstance("PBKDF2WithHMACSHA512").generateSecret(clave_pbe_spec);
43 PBEParameterSpec param_pbe_spec = new PBEParameterSpec(buffer, ITERACIONES_PBE);
44 Cipher descifrador_pbe = Cipher.getInstance("PBKDF2WithHMACSHA512");
45 descifrador_pbe.init(Cipher.DECRYPT_MODE, clave_pbe, param_pbe_spec, sr);
46 buffer = new byte[fis.available()];
47 fis.read(buffer);
48 buffer = descifrador_pbe.doFinal(buffer);
49 PKCS8EncodedKeySpec clave_privada_spec = new PKCS8EncodedKeySpec(buffer);
50 KeyFactory kf = KeyFactory.getInstance("RSA");
51 PrivateKey clave_privada = kf.generatePrivate(clave_privada_spec);
52 System.out.println("Clave secreta recuperada");
53
54 // Generamos el fichero desenscriptado
55 DataInputStream dis = new DataInputStream(new FileInputStream(fichero_encryptado));
56 FileOutputStream fos = new FileOutputStream(fichero_desenscriptado);
57
58 // 1. Recuperamos la clave de sesion
59 System.out.println("Generando el fichero desenscriptado...");
60 int longitud = dis.readInt();
61 byte[] buffer = new byte[longitud];
62 dis.read(buffer);
63 fos.write(buffer);
64 fos.close();
65 System.out.println("Fichero desenscriptado correctamente");
```

Problems | Javadoc | Declaration | Console

<terminated> DesenscriptarFichero [Java Application] C:\Program Files\Java\jre1.8.0\_191\bin\javaw.exe (19 dic. 2019 16:44:49)

Indique fichero a desenscriptar: ficheroParaEncryptar.txt.crypto  
Indique que fichero tiene la clave privada a usar: ficheroClavePrivada  
Indique password con que se encrypto el fichero ficheroClavePrivada:1234  
Recuperando clave privada...  
Clave secreta recuperada  
Generando el fichero desenscriptado...  
Guardando ficheroParaEncryptar.txt.crypto en el fichero encryptado ficheroParaEncryptar.txt  
Fichero desenscriptado correctamente

```
Simbolo del sistema

C:\Users\vcjon\Desktop\AMS2\M9_Programacion_de_servicios_y_procesos\UF1\Practica_Criptografia_Asimetrica_VALLEJ>type ficheroParaEncryptar.txt
Hola soy Jonatan, estudiante del Esteve Terrades.
C:\Users\vcjon\Desktop\AMS2\M9_Programacion_de_servicios_y_procesos\UF1\Practica_Criptografia_Asimetrica_VALLEJ>
```



Generalitat de Catalunya  
Departament d'Educació  
Institut d'Educació Secundària i Superior  
d'Ensenyaments Professionals

**Esteve Terradas i Illa**

**Departament d'Informàtica**

### **3. Llista el codi font i afegeix comentaris de que fan les diferents parts.**

#### **CODIGO CREAR CLAVES COMENTADO:**

```
public class CrearClaves implements Constantes {

 public static void main(String[] args) throws Exception {

 // Generamos las claves publica/privada
 SecureRandom sr = new SecureRandom(); // Genera un numero random
 cryptographically
 sr.setSeed(new Date().getTime());
 System.out.println("Generando claves...");
 KeyPairGenerator kpg = KeyPairGenerator.getInstance("RSA"); // Genera
 una clave publica o privada
 kpg.initialize(TAMANO_CLAVE_RSA, sr);
 KeyPair par_claves = kpg.generateKeyPair(); // Genera un simple
 titular de las claves privadas o publicas
 System.out.println("Claves generadas");

 // Generamos el fichero de la clave publica
 System.out.print("Indique fichero para" + " la clave publica:");
 BufferedReader teclado = new BufferedReader(new
 InputStreamReader(System.in)); // Iniciamos el BufferedReader para escribir en el
 fichero sin sobrescribir
 String fichero_publica;
 fichero_publica = teclado.readLine();
 FileOutputStream fos = new FileOutputStream(fichero_publica); //
 FileOutputStream para escribir en el fichero
 fos.write(par_claves.getPublic().getEncoded());
 fos.close();
 System.out.println("Fichero con clave publica generado");

 // Generamos el fichero de clave privada
 System.out.print("Indique fichero para la clave privada:");
 String fichero_privada; // Hacemos los mismo que el de fichero de
 clave publica
 fichero_privada = teclado.readLine();
 System.out.print("La clave privada debe estar encriptada, indique
 password con la que encriptarla:");
 char[] password; // Aqui almacenamos la contraseña para la clave
 privada
 password = teclado.readLine().toCharArray();

 // Encriptamos con un PBE
 byte[] salt = new byte[TAMANO_SALT_BYTES]; // Hacemos un array de
 Bytes
 sr.nextBytes(salt);
 PBEKeySpec clave_pbe = new PBEKeySpec(password); // Encriptamos la
 contraseña
 SecretKey clave_secreta_pbe =
 SecretKeyFactory.getInstance("PBKDF2WithHmacSHA1") // Instancia el algoritmo de
 encriptacion de la clave secreta
 .generateSecret(clave_pbe);
```



Generalitat de Catalunya  
Departament d'Educació  
Institut d'Educació Secundària i Superior  
d'Ensenyaments Professionals  
**Esteve Terradas i Illa**  
**Departament d'Informàtica**

```
PBEParameterSpec pbe_param = new PBEParameterSpec(salt,
ITERACIONES_PBE); // Setea paramatros usados para la contraseña encriptada
Cipher cifrador_pbe = Cipher.getInstance("PBESWithMD5AndDES"); //
Instanciamos el cifrador con el algoritmo de encriptacion
cifrador_pbe.init(Cipher.ENCRYPT_MODE, clave_secreta_pbe, pbe_param);
byte[] clave_privada_cifrada =
cifrador_pbe.doFinal(par_claves.getPrivate().getEncoded());
fos = new FileOutputStream(fichero_privada); // Escribimos en el
fichero lo encriptado.
fos.write(salt);
fos.write(clave_privada_cifrada);
fos.close();
System.out.println("Fichero con clave privada generado");
}
```

## CODIGO DESENCRIPTARFICHERO COMENTADO:

```
public class DescriptaFichero implements Constantes {
 public static void main(String[] args) throws Exception {
 // Pedimos el fichero a desenscriptar y fichero de clave privada a usar
 BufferedReader teclado = new BufferedReader(new InputStreamReader(System.in));
 System.out.print("Indique fichero a desenscriptar:");
 String fichero_encriptado = teclado.readLine();

 // Se comprueba que el fichero exista y que sea valido
 if (!new File(fichero_encriptado).exists()) {
 System.out.println("El fichero " + fichero_encriptado + " no existe");
 return;
 }
 if (!fichero_encriptado.toLowerCase().endsWith(".crypto")) {
 System.out.println("La extension de los " + "ficheros encriptados debe ser
.crypto");
 return;
 }
 String fichero_desenscriptado = fichero_encriptado.substring(0,
 fichero_encriptado.length() - ".crypto".length());
 System.out.print("Indique que fichero tiene la clave privada a usar:");
 String fichero_privada = teclado.readLine();

 // Para continuar, se necesita la password con la que se encripto el fichero que
 // contiene la clave privada
 System.out.print("Indique password con que se encripto el fichero " +
fichero_privada + " :");
 char[] password;
 password = teclado.readLine().toCharArray();

 // Recuperamos la clave privada
 System.out.println("Recuperando clave privada...");
 SecureRandom sr = new SecureRandom();
 sr.setSeed(new Date().getTime());

 // Y cargamos el fichero que contiene la clave privada en un InputStream
 FileInputStream fis = new FileInputStream(fichero_privada);
 byte[] buffer = new byte[TAMANO_SALT_BYTES];
 fis.read(buffer);
 }
}
```





Generalitat de Catalunya  
Departament d'Educació  
Institut d'Educació Secundària i Superior  
d'Ensenyaments Professionals  
**Esteve Terradas i Illa**  
**Departament d'Informàtica**

```
// Se lee el fichero que contiene la clave privada usando la clave introducida
PBEKeySpec clave_pbe_spec = new PBEKeySpec(password);
SecretKey clave_pbe =
SecretKeyFactory.getInstance("PBESWithMD5AndDES").generateSecret(clave_pbe_spec);
PBESParameterSpec param_pbe_spec = new PBESParameterSpec(buffer, ITERACIONES_PBE);
Cipher descifrador_pbe = Cipher.getInstance("PBESWithMD5AndDES");
descifrador_pbe.init(Cipher.DECRYPT_MODE, clave_pbe, param_pbe_spec, sr);
buffer = new byte[fis.available()];
fis.read(buffer);
buffer = descifrador_pbe.doFinal(buffer);
PKCS8EncodedKeySpec clave_privada_spec = new PKCS8EncodedKeySpec(buffer);
KeyFactory kf = KeyFactory.getInstance("RSA");

// Y ya tenemos la clave privada descryptada
PrivateKey clave_privada = kf.generatePrivate(clave_privada_spec);
System.out.println("Clave secreta recuperada");

// Generamos el fichero descryptado
DataInputStream dis = new DataInputStream(new
FileInputStream(fichero_encryptado));
FileOutputStream fos = new FileOutputStream(fichero_descryptado);

// Recuperamos la clave de sesion
System.out.println("Generando el fichero descryptado...");
int longitud = dis.readInt();
buffer = new byte[longitud];
dis.read(buffer);

// Descryptamos la clave de sesion
Cipher descifrador_rsa = Cipher.getInstance("RSA/ECB/PKCS1Padding");
descifrador_rsa.init(Cipher.DECRYPT_MODE, clave_privada, sr);
buffer = descifrador_rsa.doFinal(buffer);
SecretKeySpec clave_sesion = new SecretKeySpec(buffer, "Blowfish");

// Recuperamos el IV
byte[] IV = new byte[TAMANO_IV_BYTES];
dis.read(IV);
IvParameterSpec iv_spec = new IvParameterSpec(IV);

// Descryptamos y vamos generando el fichero descryptado
System.out.println("Guardando " + fichero_encryptado + " en el fichero
encryptado " + fichero_descryptado);
Cipher cifrador_fichero = Cipher.getInstance("Blowfish/CBC/PKCS5Padding");
cifrador_fichero.init(Cipher.DECRYPT_MODE, clave_sesion, iv_spec, sr);
CipherOutputStream cos = new CipherOutputStream(fos, cifrador_fichero);
int b = dis.read();
while (b != -1) {
 cos.write(b);
 b = dis.read();
}
dis.close();
cos.close();
fos.close();
System.out.println("Fichero descryptado correctamente");
}
```



Generalitat de Catalunya  
Departament d'Educació  
Institut d'Educació Secundària i Superior  
d'Ensenyaments Professionals

**Esteve Terradas i Illa**  
**Departament d'Informàtica**

}

CODIGO ENCRIPITARFICHERO COMENTADO:

```
public class EncriptaFichero implements Constantes {
 public static void main(String[] args) throws Exception {
 // Pedimos el fichero a encriptar y fichero de clave publica a usar
 BufferedReader teclado = new BufferedReader(new
InputStreamReader(System.in));
 System.out.print("Indique fichero a encriptar:");
 String fichero_encriptar = teclado.readLine();

 // Se comprueba que el fichero exista y que sea valido
 if (!new File(fichero_encriptar).exists()) {
 System.out.println("El fichero " + fichero_encriptar + " no existe");
 return;
 }
 String fichero_encriptado = fichero_encriptar + ".crypto";
 System.out.print("Indique que fichero tiene la" + " clave publica a usar:");
 String fichero_publica = teclado.readLine();

 // Recuperamos la clave publica
 FileInputStream fis = new FileInputStream(fichero_publica);
 byte[] buffer = new byte[fis.available()];
 fis.read(buffer);
 X509EncodedKeySpec clave_publica_spec = new X509EncodedKeySpec(buffer);
 KeyFactory kf = KeyFactory.getInstance("RSA");
 PublicKey clave_publica = kf.generatePublic(clave_publica_spec);

 // Generamos el fichero encriptado
 SecureRandom sr = new SecureRandom();
 sr.setSeed(new Date().getTime());
 fis = new FileInputStream(fichero_encriptar);
 DataOutputStream dos = new DataOutputStream(new
FileOutputStream(fichero_encriptado));

 // Generamos una clave de sesion
 System.out.println("Generando clave de sesion...");
 KeyGenerator kg = KeyGenerator.getInstance("Blowfish");
 kg.init(TAMANO_CLAVE_SESION, sr);
 SecretKey clave_sesion = (SecretKey) kg.generateKey();

 // Guardamos la clave de sesion encriptada en el fichero
 System.out.println("Guardando la clave de sesion encriptada...");
 Cipher cifrador_rsa = Cipher.getInstance("RSA/ECB/PKCS1Padding");
 cifrador_rsa.init(Cipher.ENCRYPT_MODE, clave_publica, sr);
 buffer = cifrador_rsa.doFinal(clave_sesion.getEncoded());
 dos.writeInt(buffer.length);
 }
}
```



Generalitat de Catalunya  
Departament d'Educació  
Institut d'Educació Secundària i Superior  
d'Ensenyaments Professionals

**Esteve Terradas i Illa**

**Departament d'Informàtica**

```
dos.write(buffer);
```

```
// Generamos un IV aleatorio
```

```
byte[] IV = new byte[TAMANO_IV_BYTES];
```

```
sr.nextBytes(IV);
```

```
IvParameterSpec iv_spec = new IvParameterSpec(IV);
```

```
dos.write(IV);
```

```
// Guardamos los datos encriptados en el fichero
```

```
System.out.println("Guardando " + fichero_encriptar + " en el fichero encriptado" +
fichero_encriptado);
```

```
Cipher cifrador_fichero = Cipher.getInstance("Blowfish/CBC/PKCS5Padding");
```

```
cifrador_fichero.init(Cipher.ENCRYPT_MODE, clave_sesion, iv_spec, sr);
```

```
CipherOutputStream cos = new CipherOutputStream(dos, cifrador_fichero);
```

```
int b = fis.read();
```

```
while (b != -1) {
```

```
 cos.write(b);
```

```
 b = fis.read();
```

```
}
```

```
fis.close();
```

```
cos.close();
```

```
dos.close();
```

```
System.out.println("Fichero encriptado correctamente");
```

```
}
```

```
}
```

#### **4. Indica el protocol de xifratge que empra i les longituds de les claus.**

Com a protocol de xifratge hem emprat PBEWithMD5AndDES, amb una longitud de clau de 1024. La clau de sessió té una longitud 128.