```python
# bob.py
# g. Cargar la clave privada de Bob y la clave pública de Alice.
# h. Cargar     el texto cifrado y la firma     digital.
# i. Descifrar el texto cifrado y mostrarlo por pantalla.
# j. Comprobar la validez de la firma digital

from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
from Crypto.Signature import pss
from Crypto.Hash import SHA256

def crear_RSAKey():
    key = RSA.generate(2048)
    return key

def guardar_RSAKey_Privada(fichero, key, password):
    key_cifrada = key.export_key(passphrase=password, pkcs=8, protection="scryptAndAES128-CBC")
    file_out = open(fichero, "wb")
    file_out.write(key_cifrada)
    file_out.close()

def cargar_RSAKey_Privada(fichero, password):
    key_cifrada = open(fichero, "rb").read()
    key = RSA.import_key(key_cifrada, passphrase=password)
    return key

def guardar_RSAKey_Publica(fichero, key):
    key_pub = key.publickey().export_key()
    file_out = open(fichero, "wb")
    file_out.write(key_pub)
    file_out.close()

def cargar_RSAKey_Publica(fichero):
    keyFile = open(fichero, "rb").read()
    key_pub = RSA.import_key(keyFile)
    return key_pub

def cifrarRSA_OAEP(cadena, key):

    datos = cadena.encode("utf-8")
    engineRSACifrado = PKCS1_OAEP.new(key)
    cifrado = engineRSACifrado.encrypt(datos)
    return cifrado

def descifrarRSA_OAEP(cifrado, key):
    engineRSADescifrado = PKCS1_OAEP.new(key)
    datos = engineRSADescifrado.decrypt(cifrado)
    cadena = datos.decode("utf-8")
    return cadena

def firmarRSA_PSS(texto, key_private):
# La firma se realiza sobre el hash del texto (h)
    h = SHA256.new(texto.encode("utf-8"))
    print(h.hexdigest())
    signature = pss.new(key_private).sign(h)
    return signature

def comprobarRSA_PSS(texto, firma, key_public):
# Comprobamos que la firma coincide con el hash (h)
    h = SHA256.new(texto.encode("utf-8"))
```

```
    print(h.hexdigest())
    verifier = pss.new(key_public)
    try:
        verifier.verify(h, firma)
        return True
    except (ValueError, TypeError):
        return False

def savefile (file, data):
    file_out = open (file, "wb")
    file_out.write = (data)
    file_out.close()

def loadFile(file):
    return open(file, "rb").read()

password = "1234"

# g. Cargar la clave privada de Bob y la clave pública de Alice.
bobPrivKey = cargar_RSAKey_Privada("B_priv.pkcs",password)
alicePubKey = cargar_RSAKey_Publica("A_pub.pkcs")

# h. Cargar el texto cifrado y la firma digital.
cipherText = loadFile("cadenaCifradaAliceBob.bin")
signedText = loadFile("cadenaFirmadaAlice.bin")

# i. Descifrar el texto cifrado y mostrarlo por pantalla.
text = descifrarRSA_OAEP(cipherText, bobPrivKey)
print(text)

# j. Comprobar la validez de la firma digital
if comprobarRSA_PSS(text, signedText, alicePubKey):
    print("Texto validado con la firma de Alice")
else:
    print("Error de firma")
```