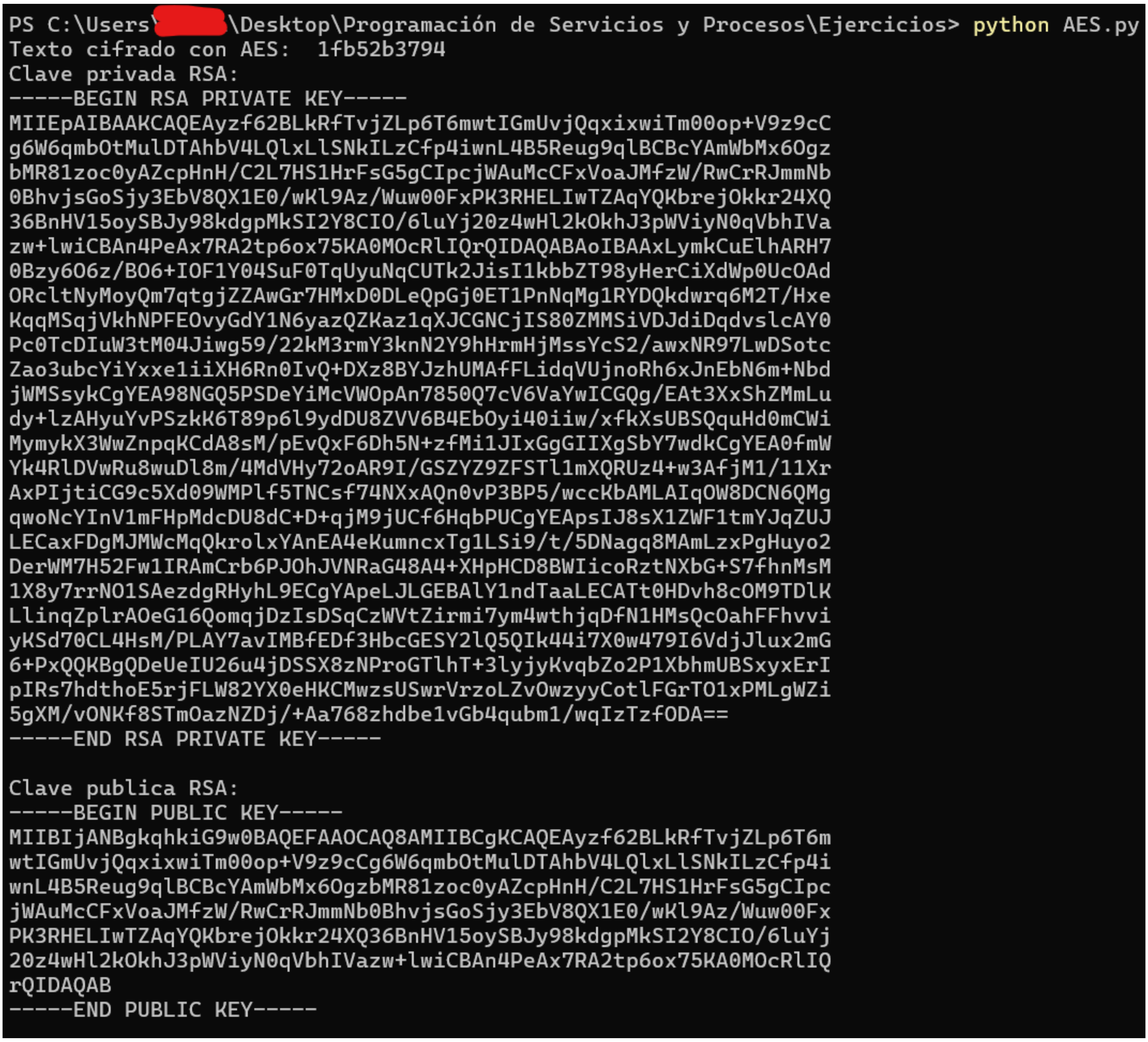
Tarea Individual 26 - Cifrados simetricos y asimetricos

La salida muestra:

- La generación de claves.
- El proceso de cifrado y descifrado con AES y RSA.

Capturas de la salida:



```
PS C:\Users\[redacted]\Desktop\Programación de Servicios y Procesos\Ejercicios> python AES.py
Texto cifrado con AES:  1fb52b3794
Clave privada RSA:
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAyzf62BLkRfTvjZLp6T6mwtIGmUvjQqxixwiTm00op+V9z9cC
g6W6qmbOtMulDTAhbV4LQlxLlSNkILzCfp4iwnL4B5Reug9qlBCBcYAmWbMx6Ogz
bMR81zoc0yAZcpHnH/C2L7HS1HrFsG5gCIpcjWAuMcCFxVoaJMfzW/RwCrRJmmNb
0BhvjsGoSjy3EbV8QX1E0/wKl9Az/Wuw00FxPK3RHELIwTZAqYQKbrejOkkr24XQ
36BnHV15oySBJy98kdgpMkSI2Y8CIO/6luYj20z4wHl2kOkhJ3pWViyN0qVbhIVa
zw+lwiCBAn4PeAx7RA2tp6ox75KA0MOcRlIQrQIDAQABAoIBAAxLymkCuElhARH7
0Bzy6O6z/BO6+IOF1Y04SuF0TqUyuNqCUTk2JisI1kbbZT98yHerCiXdWp0UcOAd
ORcltNyMoyQm7qtgjZZAwGr7HMxD0DLeQpGj0ET1PnNqMg1RYDQkdwrq6M2T/Hxe
KqqMSqjVkhNPFEOvyGdY1N6yazQZKaz1qXJCGNCjIS80ZMMSiVDJdiDqdvslcAY0
Pc0TcDIuW3tM04Jiwg59/22kM3rmY3knN2Y9hHrmHjMssYcS2/awxNR97LwDSotc
Zao3ubcYiYxxe1iiXH6Rn0IvQ+DXz8BYJzhUMAfFLidqVUjnoRh6xJnEbN6m+Nbd
jWMSsykCgYEA98NGQ5PSDeYiMcVWOpAn7850Q7cV6VaYwICGQg/EAt3XxShZMmLu
dy+lzAHyuYvPSzkK6T89p6l9ydDU8ZVV6B4EbOyi40iiw/xfkXsUBSQquHd0mCWi
MymykX3WwZnpqKCdA8sM/pEvQxF6Dh5N+zfMi1JIxGgGIIXgSbY7wdkCgYEA0fmW
Yk4RlDVwRu8wuDl8m/4MdVHy72oAR9I/GSZYZ9ZFSTl1mXQRUz4+w3AfjM1/11Xr
AxPIjtiCG9c5Xd09WMPlf5TNCsf74NXxAQn0vP3BP5/wccKbAMLAIqOW8DCN6QMg
qwoNcYInV1mFHpMdcDU8dC+D+qjM9jUCf6HqbPUCgYEApsIJ8sX1ZWF1tmYJqZUJ
LECaxFDgMJMWcMqQkrolxYAnEA4eKumncxTg1LSi9/t/5DNagq8MAmLzxPgHuyo2
DerWM7H52Fw1IRAmCrb6PJOhJVNRaG48A4+XHpHCD8BWIicoRztNXbG+S7fhnMsM
1X8y7rrNO1SAezdgRHyhL9ECgYApeLJLGEBAlY1ndTaaLECATt0HDvh8cOM9TDlK
LlinqZplrAOeG16QomqjDzIsDSqCzWVtZirmi7ym4wthjqDfN1HMsQcOahFFhvvi
yKSd70CL4HsM/PLAY7avIMBfEDf3HbcGESY2lQ5QIk44i7X0w479I6VdjJlux2mG
6+PxQQKBgQDeUeIU26u4jDSSX8zNProGTlhT+3lyjyKvqbZo2P1XbhmUBSxyxErI
pIRs7hdthoE5rjFLW82YX0eHKCMwzsUSwrVrzoLZvOwzyyCotlFGrTO1xPMLgWZi
5gXM/vONKf8STmOazNZDj/+Aa768zhdbe1vGb4qubm1/wqIzTzfODA==
-----END RSA PRIVATE KEY-----

Clave publica RSA:
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAyzf62BLkRfTvjZLp6T6m
wtIGmUvjQqxixwiTm00op+V9z9cCg6W6qmbOtMulDTAhbV4LQlxLlSNkILzCfp4i
wnL4B5Reug9qlBCBcYAmWbMx6OgzbMR81zoc0yAZcpHnH/C2L7HS1HrFsG5gCIpc
jWAuMcCFxVoaJMfzW/RwCrRJmmNb0BhvjsGoSjy3EbV8QX1E0/wKl9Az/Wuw00Fx
PK3RHELIwTZAqYQKbrejOkkr24XQ36BnHV15oySBJy98kdgpMkSI2Y8CIO/6luYj
20z4wHl2kOkhJ3pWViyN0qVbhIVazw+lwiCBAn4PeAx7RA2tp6ox75KA0MOcRlIQ
rQIDAQAB
-----END PUBLIC KEY-----
```

Código:

```
from Crypto.Cipher import AES
from cryptography.hazmat.primitives.asymmetric import rsa
from cryptography.hazmat.primitives import serialization

#PRIMERO -------------------------------------------------------------

#Definir una clave de 16 bytes (128 bits)
key = b'Sixteen byte key'

#Crear el objeto de cifrado AES en modo EAX
cipher = AES.new(key, AES.MODE_EAX)

#Mensaje a cifrar
plaintext = b'hello'

#Cifrar y obtener el ciphertext y el tag de autenticación
ciphertext, tag = cipher.encrypt_and_digest(plaintext)

print("Texto cifrado con AES: ", ciphertext.hex())


#SEGUNDO -------------------------------------------------------------

#Generar un par de claves RSA (clave privada y publica)

private_key = rsa.generate_private_key(
    public_exponent = 65537,
    key_size = 2048
)

#Obtener la clave publica
public_key = private_key.public_key()

#Serializar las claves en formato PEM para guardarlas
private_pem = private_key.private_bytes(
    encoding=serialization.Encoding.PEM,
    format=serialization.PrivateFormat.TraditionalOpenSSL,
    encryption_algorithm=serialization.NoEncryption()
)

public_pem = public_key.public_bytes(
    encoding=serialization.Encoding.PEM,
    format=serialization.PublicFormat.SubjectPublicKeyInfo

)
```

```
print("Clave privada RSA: ")
print(private_pem.decode())

print("Clave publica RSA: ")
print(public_pem.decode())
```

Hecho por Jorge Varela Zamora, de 2º del C.F.G.S. de D.A.M, curso 2024-2025, I.E.S. Vista Alegre