```
# RSA

!pip install pycryptodome

from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
from binascii import hexlify

key = RSA.generate(1024)

private_key = key

public_key = key.publickey()

data_to_encrypt = b"Hello, this is a message to be encrypted."
cipher_rsa = PKCS1_OAEP.new(public_key)

encrypted = cipher_rsa.encrypt(data_to_encrypt)

print("Encrypted:", hexlify(encrypted))

cipher_rsa = PKCS1_OAEP.new(private_key)
decrypted = cipher_rsa.decrypt(encrypted)

print("Decrypted:", decrypted.decode("utf-8"))
```

```
Requirement already satisfied: pycryptodome in /usr/local/lib/python3.10/dist-packages (3.20.0)
Encrypted: b'670fcef9ba40959b39ea0d074e0b57bbdcbab3db8580f7a2abb9aea54a79f9c45688ab15a6fb8d3c04eba45d70b05b284637f9365e906afa0bf73f7
Decrypted: Hello, this is a message to be encrypted.
```

```
# AES

from Crypto.Cipher import AES
from Crypto.Hash import HMAC, SHA256
from Crypto.Random import get_random_bytes

data = 'secret data to transmit'.encode()

#encode

aes_key = get_random_bytes(16)
hmac_key = get_random_bytes(16)

cipher = AES.new(aes_key, AES.MODE_CTR)
ciphertext = cipher.encrypt(data)
nonce = cipher.nonce

hmac = HMAC.new(hmac_key, digestmod=SHA256)
tag = hmac.update(cipher.nonce + ciphertext).digest()

print(ciphertext)
```

```
b'\xd84\x06\xf0>\xe1\x7fS 5\xc9\x87\xffy\x9ciZ|\xe6\x16\xbf\x90\x81'
```

```
#decode

hmac = HMAC.new(hmac_key, digestmod=SHA256)

cipher = AES.new(aes_key, AES.MODE_CTR, nonce=nonce)
message = cipher.decrypt(ciphertext)

print("Message:", message.decode())
```

```
Message: secret data to transmit
```