# Isaac de Melo Souza – 824222167

**Ysadora da Silva Chacon Milani - 824217796**

# Criptografia Simétrica, Assimétrica e Funções Hash

## 1. Criptografia Simétrica - AES com modo CBC

Criptografia simétrica utiliza a mesma chave para criptografar e descriptografar dados. O AES (Advanced Encryption Standard) é um dos algoritmos mais utilizados atualmente por sua segurança e desempenho.

### Implementação e Codificação:

from Crypto.Cipher import AES  
from Crypto.Random import get\_random\_bytes  
from Crypto.Util.Padding import pad, unpad  
  
mensagem = "Mensagem secreta muito importante"  
mensagem\_bytes = mensagem.encode()  
  
chave = get\_random\_bytes(16)  
iv = get\_random\_bytes(16)  
  
cipher = AES.new(chave, AES.MODE\_CBC, iv)  
mensagem\_criptografada = cipher.encrypt(pad(mensagem\_bytes, AES.block\_size))  
print("Criptografada:", mensagem\_criptografada)  
  
cipher\_dec = AES.new(chave, AES.MODE\_CBC, iv)  
mensagem\_descriptografada = unpad(cipher\_dec.decrypt(mensagem\_criptografada), AES.block\_size)  
print("Descriptografada:", mensagem\_descriptografada.decode())

### Verificação:

A mensagem original é criptografada e depois restaurada com sucesso após a descriptografia, confirmando o funcionamento do processo.

## 2. Criptografia Assimétrica - RSA

Na criptografia assimétrica, utiliza-se um par de chaves: uma pública e uma privada. A chave pública é usada para criptografar os dados e a privada para descriptografar. O RSA é um dos algoritmos mais conhecidos nesse tipo de criptografia.

### Implementação e Codificação:

from Crypto.PublicKey import RSA  
from Crypto.Cipher import PKCS1\_OAEP  
  
# Gerar par de chaves  
chave = RSA.generate(2048)  
chave\_publica = chave.publickey()  
  
# Criar objetos de cifra  
cipher\_rsa\_enc = PKCS1\_OAEP.new(chave\_publica)  
cipher\_rsa\_dec = PKCS1\_OAEP.new(chave)  
  
# Mensagem  
mensagem = "Mensagem confidencial".encode()  
  
# Criptografia  
mensagem\_criptografada = cipher\_rsa\_enc.encrypt(mensagem)  
print("Criptografada:", mensagem\_criptografada)  
  
# Descriptografia  
mensagem\_descriptografada = cipher\_rsa\_dec.decrypt(mensagem\_criptografada)  
print("Descriptografada:", mensagem\_descriptografada.decode())

### Verificação:

A criptografia é feita com a chave pública e a descriptografia com a chave privada. A mensagem original é recuperada com sucesso.

## 3. Função Hash - SHA-256

Funções hash transformam qualquer entrada de dados em uma sequência de comprimento fixo. São usadas para verificação de integridade e armazenamento seguro de senhas.

### Implementação e Codificação:

import hashlib  
  
mensagem = "SenhaSuperSecreta".encode()  
hash\_sha256 = hashlib.sha256(mensagem).hexdigest()  
  
print("Hash SHA-256:", hash\_sha256)

### Verificação:

Ao aplicar o hash na mensagem, sempre obtemos o mesmo resultado para a mesma entrada. Se a mensagem mudar, o hash muda completamente.