Nombre: Ortega Silva Jorge Eduardo

Como ejecutar los programas

Son tres programas los que hay que ejecutar, explicare cual es la función de cada uno a continuación

BinarioBase64.py: Las funciones se prueban directo desde el código fuente, al final de este, se encuentra comentareado donde se llaman a cada función, en los parámetros se puede probar entre la generación de diferentes n números pseudoaleatorios y dada una cadena binaria como seria convertida a base 64

CifradoTOP.py: Al correr el programa pedirá escribir el texto a cifrar como entrada e imprimirá tu llave generada junto con el mensaje cifrado, ambos en base 64

DescifradoTOP.py: Al correr el programa se pedirá el texto a descifrar, seguido de la clave, ambos en base 64, e imprimirá el mensaje descifrado

Funciones realizadas

Ejercicio 1

```
def bitsaleatorios(n):
    print("Bits pseudoaleatorio")
    for i in range(0, n+1):
        print("Bit pseudoaleatorio ",i,": ",secrets.randbits(1),"\n")#numero aleatorio de un bit
```

Dado un número en un ciclo for se generan dichos n números aleatorios con la función secrets.randbits que ofrece Python para trabajar a nivel criptográfico, donde pasando el parámetro uno solo generara valores aleatorios de un bit (1-0)

Ejercicio 2

```python
#funcion para convertir de binario a base64
def binarioBase64(bnr):
    #variables para la conversion de binario a decimal
    posicion = 0
    decimal = 0
    start=0
    long=len(bnr)# longitud de la cadena binaria
    binario="" #bloques de 8 bits
    texto="" #cadena de texto
    if(long%8!=0):#agregar ceros faltantes a la cadena
        ceros=8-long%8
        bnr=bnr[::-1]
        for i in range (0, ceros):
            bnr+='0'
        bnr=bnr[::-1]
    long=len(bnr)
    while(start!=long):#Convertir de binario a decimal
        for i in range(start, start+8):
            binario+=str(bnr[i])
        binario = binario[::-1]#invertir la cadena
        for digito in binario:
            decimal += int(digito) * 2**posicion # Elevar 2 a la posición actual y multiplicar por el bit
            posicion += 1

        texto+=chr(decimal)#convertir de decimal a caracter

        start=start+8
        binario=""
        decimal=0
        posicion=0
    mensaje_bytes = texto.encode('UTF-8') #mensaje en tipo bytes
    base64_bytes = base64.b64encode(mensaje_bytes)#arreglo de bytes en base 64
    base64_mensaje = base64_bytes.decode('UTF-8') #texto en base 64
    return base64_mensaje
```

La cadena en binario la tratamos de tal forma que su longitud sea múltiplo de 8, siendo que si faltan bits se rellenan con 0, esto se hace para su mejor manejo. Después dicha cadena se toma en bits de 8 y se convierte a decimal cada decimal se pasa a su correspondiente carácter en ASCII, esto se le aplica a toda la cadena de bits y cada resultado se guarda en la variable 'texto'. La variable texto se pasa a un arreglo de bytes para después mandarlo como parámetro a la función proporcionada por Python que nos hace la conversión a base 64, esta nos regresa un arreglo de bytes, esta cadena la pasamos a tipo String y este valor seria el resultado de nuestra conversión

Ejercicio 3

```python
def cifradoTOP(texto, clave):
    clave=base64Binario(clave)#convertir la base 64 en binario
    #operacion xor
    long=len(texto)
    cifrado=""# guardar el texto cifrado
    for i in range(0, long):#aplicar xor
        if(texto[i]==clave[i]):
            cifrado+='0'
        else:
            cifrado+='1'
    return binarioBase64(cifrado)  #pasar a base 64 el cifrado binario
```

Dados el mensaje en binario y la clave en base 64, la cual solo es generar los bits aleatorios de la misma longitud del mensaje, cabe aclara que el mensaje ya debe estar tratado como en el ejercicio 3, donde a cada carácter se le pone dentro de un bloque de 8 bits. Se realiza la operación lógica xor con cada bit del mensaje y la llave, cada resultado se guarda en la variable 'cifrado' esta será pasada a base 64 y nuestro resultado será el mensaje cifrado

Ejercicio 4

```
def descifradoTOP(texto, clave):
    #operacion xor
    textoBin=base64Binario(texto)#convertir la base 64 en binario
    clave=base64Binario(clave)#convertir la base 64 en binario
    long=len(textoBin)
    descifrado=""# guardar el texto cifrado
    for i in range(0, long):#aplicar xor para descifrar
        if(textoBin[i]==clave[i]):
            descifrado+='0'
        else:
            descifrado+='1'
    txtBase64=binarioBase64(descifrado)#convertir el binario descifrado a base 64
    base64_bytes = txtBase64.encode("UTF-8")#pasar de base 64 a caracteres
    mensaje_bytes = base64.b64decode(base64_bytes)
    mensaje = mensaje_bytes.decode("UTF-8")
    return mensaje #mensaje final
```

Dado el mensaje cifrado y la llave en base 64, ambos se pasarán a binario, se realizara la operación lógica xor con cada bit de ambas cadenas, el resultado se guarda en la variable 'descifrado', dicha cadena binaria se pasa a base 64, para después pasarla a tipo bytes, seguido de decodificarla de la base 64, finalmente esta cadena se pasara a tipo String, este será nuestro mensaje descifrado.

Capturas de pantalla de pruebas

Ejercicio 1

```
Bits pseudoaleatorio
Bit pseudoaleatorio  0 :  1

Bit pseudoaleatorio  1 :  0

Bit pseudoaleatorio  2 :  1

Bit pseudoaleatorio  3 :  0

Bit pseudoaleatorio  4 :  0

Bit pseudoaleatorio  5 :  0

Bit pseudoaleatorio  6 :  0

Bit pseudoaleatorio  7 :  1

Bit pseudoaleatorio  8 :  0
```

```
Bits pseudoaleatorio
Bit pseudoaleatorio  0 :  0

Bit pseudoaleatorio  1 :  0

Bit pseudoaleatorio  2 :  1

Bit pseudoaleatorio  3 :  1

Bit pseudoaleatorio  4 :  1

Bit pseudoaleatorio  5 :  1
```

Ejercicio 2

```
Texto en base 64:  Sm9yZ2U=
```

```
Texto en base 64:  wpTDnsOyZ2U=
```

Ejercicio 3 y 4

Prueba 1

```
Escriba el texto a cifrar: Criptografía
Llave:  RkcUwqXDosKbw4nDqhtZQmI=
Mensaje cifrado en Base 64:  BTV9w5XClsO0wq7CmHo/wq8D
```

```
Ingrese el texto a descifrar: BTV9w5XClsO0wq7CmHo/wq8D

Ingrese la llave: RkcUwqXDosKbw4nDqhtZQmI=
Mensaje descifrado:  Criptografía
```

Prueba 2

```
Escriba el texto a cifrar: Jorge Ortega
Llave:  wqrCjcK8V8K6wozCn2McYGvCtQ==
Mensaje cifrado en Base 64:  w6DDosOOMMOfwqzDkBFoBQzDlA==
```

```
Ingrese el texto a descifrar:
w6DDosOOMMOfwqzDkBFoBQzDlA==

Ingrese la llave: wqrCjcK8V8K6wozCn2McYGvCtQ==
Mensaje descifrado:  Jorge Ortega
```