



**INSTITUTO POLITÉCNICO
NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO**



Cryptography

Lab 04: Finite fields

Grupo: 3CM12

Alumnos:

2019630426 Cazares Martínez Maximiliano

2019630004 Cipriano Damián Sebastián

2019630531 Murillo Granada Esteban Jair

Profesora: Diaz Santiago Sandra.

Fecha de entrega: 25 de Octubre del 2021

Implementar el algoritmo y diseñar una función que reciba como entrada dos elementos $f, g \in GF(2^6)$. La función debe devolver $f * g$.

El algoritmo se implementó para resolver $f(x) * g(x)$ en $GF(2^6)$ en Python. Se utilizó la biblioteca “bitstring” para poder manipular las cadenas binarias dentro del programa. Existen 2 funciones que se encargan de la multiplicación en $GF(2^6)$ de las cadenas; estas funciones se llaman MultGF26 y product.

```
1 def product(bits):
2     if not bits[0]:
3         return bits << 1
4     else:
5         return (bits << 1) ^ irreducible_polynomial
6
7 def MultGF26(f, g):
8     fx = BitArray(f'0b{f}')
9     gx = BitArray(f'0b{g}')
10
11     resP = []
12     for i in range(0, 5):
13         aux = fx
14         if gx[i]:
15             j = i
16             for j in range(i, 5):
17                 aux = product(aux)
18                 resP.append(aux)
19
20     if gx[5]:
21         resP.append(fx)
22
23     a = resP[0]
24     fgx = []
25     for i in range(1, len(resP)):
26         fgx.append(a ^ resP[i])
27
28     if len(fgx) > 0:
29         return fgx[0]
30     else:
31         return a
```

Este es un ejemplo de la multiplicación de dos polinomios de grado 2.

```
enter the binary coefficients of f(x): 000011
enter the binary coefficients of g(x): 000011
the product of f(x) * g(x) = 000101
```

Usando la función del paso anterior, calcule la tabla de multiplicar para GF (2^6).

En este apartado se hará el cálculo de la tabla de multiplicar para GF(2^6) en donde se implementa el algoritmo propuesto anteriormente y de dos funciones adicionales llamadas CompleteZeros and RightLen. Estas funciones convertirán los índices de los ciclos for en cadenas binarias.

```
1 def CompleteZeros(f):
2     a = ''
3     if len(f) < 7:
4         for i in range(6 - len(f)):
5             a += '0'
6     return f'{a}{f}'
7
8 def RightLen(f,g):
9     f , g = f[2:], g[2:]
10    f, g = CompleteZeros(f), CompleteZeros(g)
11    return f, g
```

```
1 limite = 64
2 a = []
3 for i in range(1, limite):
4     b = []
5     for j in range(1, limite):
6         fx, gx = RightLen(bin(i), bin(j))
7         b.append(MultGF26(fx, gx).bin)
8     a.append(b)
```

La siguiente captura de pantalla muestra la tabla de multiplicar de $GF(2^6)$.

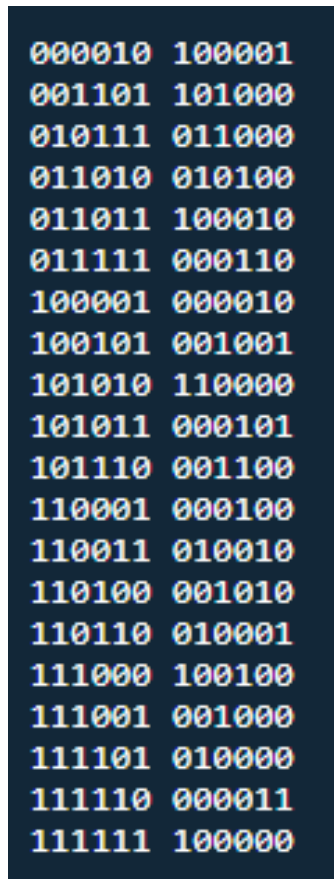
```
C:\Users\sebas\Downloads>multiplication_GF26.py
['000001', '000010', '000011', '000100', '000101', '000110', '000111', '001000', '001001', '001010', '001011', '001100', '001101', '001110', '001111', '010000', '010001', '010010', '010011', '010100', '010101', '010110', '010111', '011000', '011001', '011010', '011011', '011100', '011101', '011110', '011111', '100000', '100001', '100010', '100011', '100100', '100101', '100110', '100111', '101000', '101001', '101010', '101011', '101100', '101101', '101110', '101111', '110000', '110001', '110010', '110011', '110100', '110101', '110110', '110111', '111000', '111001', '111010', '111011', '111100', '111101', '111110', '111111']
['000010', '000100', '000110', '001000', '001010', '001100', '001110', '010000', '010010', '010100', '010110', '011000', '011010', '011100', '011110', '100000', '100010', '100100', '100110', '101000', '101010', '101100', '101110', '110000', '110010', '110100', '110110', '111000', '111010', '111100', '111110', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['000011', '000101', '000111', '001001', '001011', '001101', '001111', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['000100', '000101', '000110', '001000', '001010', '001100', '001110', '010000', '010010', '010100', '010110', '011000', '011010', '011100', '011110', '100000', '100010', '100100', '100110', '101000', '101010', '101100', '101110', '110000', '110010', '110100', '110110', '111000', '111010', '111100', '111110', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['000101', '000110', '000111', '001001', '001011', '001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['000110', '000111', '001001', '001011', '001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001000', '001001', '001010', '001011', '001100', '001101', '001110', '010000', '010001', '010010', '010011', '010100', '010101', '010110', '010111', '011000', '011001', '011010', '011011', '011100', '011101', '011110', '011111', '100000', '100001', '100010', '100011', '100100', '100101', '100110', '100111', '101000', '101001', '101010', '101011', '101100', '101101', '101110', '101111', '110000', '110001', '110010', '110011', '110100', '110101', '110110', '110111', '111000', '111001', '111010', '111011', '111100', '111101', '111110', '111111']
['001001', '001010', '001011', '001100', '001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001010', '001011', '001100', '001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001011', '001100', '001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001100', '001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001101', '001110', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001110', '001111', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['001111', '001111', '010001', '010011', '010101', '010111', '011001', '011011', '011101', '011111', '100001', '100011', '100101', '100111', '101001', '101011', '101101', '101111', '110001', '110011', '110101', '110111', '111001', '111011', '111101', '111111']
['010000', '010001', '010010', '010011', '010100', '010101', '010110', '011000', '011001', '011010', '011011', '011100', '011101', '011110', '011111', '100000', '100001', '100010', '100011', '100100', '100101', '100110', '100111', '101000', '101001', '101010', '101011', '101100', '101101', '101110', '101111', '110000', '110001', '110010', '110011', '110100', '110101', '110110', '110111', '111000', '111001', '111010', '111011', '111100', '111101', '111110', '111111']
['010001', '010010', '010011', '010100', '010101', '010110', '011000', '011001', '011010', '011011', '011100', '011101', '011110', '011111', '100001', '100010', '100011', '100100', '100101', '100110', '100111', '101001', '101010', '101011', '101100', '101101', '101110', '101111', '110001', '110010', '110011', '110100', '110101', '110110', '110111', '111001', '111010', '111011', '111100', '111101', '111110', '111111']
```

Encuentra el inverso multiplicativo para cada elemento en $GF(2^6)$.

La función `multiplicativeInverse` encuentra los inversos multiplicativos de cada elemento en $GF(2^6)$ haciendo uso de la tabla de multiplicar anteriormente calculada.

```
1 def multiplicativeInverse(limite, a):
2     row = limite-1
3     col = limite-1
4     for i in range(0, row):
5         aux = a[i][0]
6         for j in range(1, col):
7             aux2 = a[i][j]
8             if aux2 == '000001':
9                 print(aux, a[0][j])
10            break
```

La siguiente captura de pantalla muestra en cada línea los polinomios que al multiplicarse su producto es 1, es decir, son inversos multiplicativos.



```
000010 100001
001101 101000
010111 011000
011010 010100
011011 100010
011111 000110
100001 000010
100101 001001
101010 110000
101011 000101
101110 001100
110001 000100
110011 010010
110100 001010
110110 010001
111000 100100
111001 001000
111101 010000
111110 000011
111111 100000
```

Como correr el programa:

❖ Requerimientos:

- Python instalado.
- Tener la librería de python "bitstring".
- Directorio con los archivos de la práctica.

❖ Instrucciones:

- Abrir una terminal e ir a la dirección en donde se encuentran los archivos de la práctica.

```
C:\Users\muril>cd C:\Users\muril\Desktop\Todo\Crypto  
C:\Users\muril\Desktop\Todo\Crypto>
```

- Ejecutar alguno de los dos archivos a continuación listados.

```
Directorio de C:\Users\muril\Desktop\Todo\Crypto  
25/10/2021 11:37 p. m. <DIR> .  
25/10/2021 11:37 p. m. <DIR> ..  
25/10/2021 11:09 p. m. 1,471 multiplication_GF26.py  
25/10/2021 09:51 p. m. 292 TableGF26.py  
25/10/2021 10:58 p. m. <DIR> __pycache__  
2 archivos 1,763 bytes  
3 dirs 409,867,726,848 bytes libres  
C:\Users\muril\Desktop\Todo\Crypto>
```

- Ejercicio 2 y 4 - multiplication_GF26.py
- Ejercicio 3 - TableGF26.py

- Ejemplo: Ejecución del archivo multiplication_GF26.py para obtener la multiplicación de $f(x) * g(x)$.

```
enter the binary coefficients of f(x): 000011  
enter the binary coefficients of g(x): 000011  
the product of f(x) * g(x) = 000101
```