# Códigos y Criptografía - Curso 2019-2020

## Práctica 2: Cifrado Hill y cifrado de permutación (caso particular del cifrado Hill)

- Usaremos las funciones de la práctica 1:

  **letranumero(texto)** y **inv_modulo(A,m).**

- Recordemos que si queremos trabajar con más caracteres de 27, sólo tendríamos que modificar el alfabeto en la función **letranumero** y tener en cuenta su longitud, *m*.

### 1.- Función cifrado=cifro_hill (A,m,texto)

Construir una función para cifrar un texto dado a partir de una matriz A.

**Entradas**:

*A*: matriz que va a ser la clave. La función debe comprobar que es adecuada para este tipo de cifrado, es decir, que tenga inversa módulo m.

*m*: número de elementos de nuestro alfabeto.

*texto*: texto llano que queremos cifrar.

**Salida**: texto cifrado.

***Ejemplo***

```
>> cifrado=cifro_hill ( [2 1 3;2 1 1;0 4 1] , 27 , 'en un lugar de cordoba')

cifrado =dosselmdrpmrdxueee

>> cifrado=cifro_hill ( [2 1 3;2 1 1;0 4 1] , 27 , 'un lugar de cordoba')

cifrado = hmjuuxxppskxxvhlsh
```

*Ahora tendríamos que hacer una función para descifrar un criptograma obtenido mediante un cifrado Hill, pero ¿hace falta hacerla?*

### 2.- Función permuta=permutacion_v (p)

Función para asegurarnos que la entrada es un vector que de verdad representa una permutación de {1, 2, …, n}.

**Entrada:** El vector que queremos comprobar si es o no una permutación.

**Salida**:

*permuta=0*: si el vector no es una permutación de {1, 2, …, n}.

*permuta=1*: si el vector es una permutación de {1, 2, …, n}.

***Ejemplo***

```
>> permuta=permutacion_v([5 4 3  1 2 ])
efectivamente has introducido una permutación
permuta =
  1
>> permuta=permutacion_v([5 4 6  1 2 ])
error en entrada de de la permutación
permuta = 0
```

**3.- Función  mat_per=matper (p)**

Función para comprobar que *p* es una permutación, y en este caso construir la matriz asociada a ella.

**Entrada:** El vector que supuestamente debe ser una permutación. El programa debe comprobar que lo es.

**Salida**: La matriz asociada en caso de que realmente sea una permutación, o un mensaje de error en caso contrario.

***Ejemplo***

```
>> mat_per=matper([2 1 4 5 3])
disp =la matriz asociada a la permutación introducida es
mat_per =
    0    1    0    0    0
    1    0    0    0    0
    0    0    0    1    0
    0    0    0    0    1
    0    0    1    0    0

>> mat_per=matper([2 1 4 5 3])
No has introducido una permutación
```

**4.- Función cifrado=cifro_permutacion (p,texto)**

Función para cifrar un texto a partir de una permutación con el cifrado Hill en caso de ser posible, y un mensaje de error por lo contrario.

**Entradas**:

*p*: vector que debe ser una permutación. El programa debe comprobarlo.

*texto*: el texto llano a cifrar.

**Salida**: El texto cifrado usando Hill y la permutación si es posible, o un mensaje de error en caso contrario.

***Ejemplo***

>>cifrado= cifro_permutacion ([2 4 6 5 3 1], 'hola me voy de puente')

cifrado = oaemlhodpeyvetwenu

**5.- Función texto=descifro_permutacion (p,cifrado)**

Función para descifrar un texto cifrado conociendo la permutación y sabiendo que se ha cifrado con el método Hill a partir de ésta.

**Entradas**:

*p*: vector que debe ser una permutación. El programa debe comprobarlo.

*cifrado*: el criptograma.

**Salida**: El texto llano descifrado usando Hill y la permutación si es posible, o un mensaje de error en caso contrario.

***Ejemplo***

>> texto=descifro_permutacion([6 4 2 1 3 5],'eaohlmvdovyeoccaaiyhenso')

texto =holamevoydevacacioneshoy

**6.- Función matrizclave=cripto_hill (claro, cripto, d)**

Función para hallar la matriz clave en el cifrado Hill conociendo parte del texto claro, del criptograma y el orden de la matriz, mediante un ataque de tipo Gauss-Jordan.

**Entradas:**

*claro*: un fragmento de texto llano, de longitud al menos $d^2$.

*cripto*: el criptograma, o parte de él, de longitud al menos $d^2$.

*d*: el orden que estamos suponiendo para la matriz clave del cifrado Hill.

**Salida:** la matriz clave del cifrado.

***Ejemplo***

```
>> claro='lasaparienciasengañan';
>> cripto='vovqldhlgzxivñoihccsañdh';
>> matrizclave=cripto_hill(claro,cripto,4)

matrizclave =
    1    3    2    5
    7   24    1    1
    0    0    4    2
    3    5    2    1
```

*Hay que prestar mucha atención al preparar las matrices correspondientes al texto claro y al texto cifrado en la función cripto_hill: ¿sirve cualquier longitud de texto?, ¿qué ocurre si las dos cadenas no tienen la misma longitud?*