Códigos y criptografía: Curso 2021-2022

Práctica 6: Cifrando una imagen. El mapa de Arnold.

- Pueden ser de utilidad la función de prácticas anteriores *inv_module*.
- NOTA: Todas las funciones que se realicen a lo largo de la práctica deben ser válidas tanto para fotografías en escala de grises como en RGB.

1. Función *pixel_disorder*

```
pixel_disorder (photo,A)
```

Se trata de una función que desordena los píxeles de las matrices asociadas a una imagen de acuerdo a la transformación asociada a la matriz dada. No se pretende que muestre nada, ni imágenes ni matrices, sólo que guarde las matrices obtenidas para usarlas en otras funciones.

Entradas:

photo: imagen de la queremos desordenar sus píxeles. Para poder aplicar el método la imagen debe ser cuadrada.

A: matriz que determina la transformación. Conviene recordar que debe ser cuadrada, de orden 2, con elementos enteros y debe tener inversa módulo el número de filas de *photo*.

Salida: ninguna. Debe guardar las nuevas matrices obtenidas para un posible uso posterior. Para ello puede ser útil la orden *setappdata (gcf,'matrix',matrix)*. Si más adelante necesitamos usar esa matriz puede ser de utilidad *matrix = getappdata (gcf,'matrix')*.

2. Función *arnold*

```
function arnold (photo, A)
```

Se trata de una función que ordena o desordena una foto, de acuerdo a la opción elegida por el usuario. Para elegir una de estas opciones se debe usar un *switch* con dos casos. El **caso 1** para desordenar y el **caso 2** para ordenar.

Entradas:

photo: una fotografía a la que queremos aplicarle la transformación de Arnold. Debe cumplir los requisitos necesarios. NOTA: debe ser la fotografía original en el **caso 1** y la fotografía desordenada según la matriz *A* en el **caso 2**.

A: la matriz que se va a usar para desordenar en el **caso 1** o que ya se haya usado para desordenar en el **caso 2**. Debe cumplir los requisitos necesarios.

Salida: aunque no tenga ningún *output* debe mostrar en cada caso dos imágenes en una misma ventana. En el **caso 1** debe mostrar la foto original (ordenada) junto con la modificada (desordenada). En el **caso 2** debe mostrar la foto original (desordenada) junto con la modificada (ordenada). En ambos casos debe guardar las fotos obtenidas tras la transformación.

Ejemplos:

```
>> arnold('hypatia.bmp',[1 2;1 1])
 Introduce 1 to disorder or introduce 2 to order: 1
```

Foto original

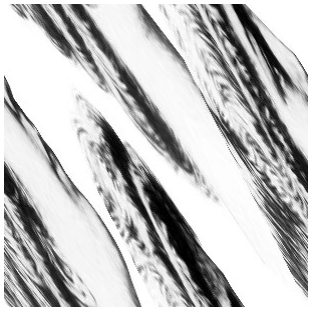

Foto modificada

```
>> arnold('disorder.bmp',[1 3;1 1])
     Introduce 1 to disorder or introduce 2 to order: 2
```

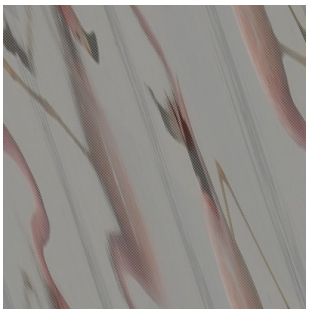

Foto original

Foto modificada

3. Función *power_a*

```
function n = power_a (A, m)
```

Se trata de una función que calcula el mínimo valor del exponente de la potencia de A que módulo m es igual a la matriz identidad.

Entradas:

A: una matriz cuadrada de orden 2, con elementos enteros, y con inversa módulo m.

m: un número natural, que representa nuestro módulo de trabajo.

Salida: un número natural, que se corresponde con el valor del exponente para el que la correspondiente potencia de A es la identidad.

Ejemplo:

```
>> n = power_a ([54 118 ; 260 14], 193)
n   =   96
```

4. Función *arnold_02*

```
function power = arnold_02 (photo, A)
```

Se trata de una función que desordena los pízeles de la imagen *photo* según la matriz A de manera sucesiva. Vamos a usar un *switch* con dos casos. El **caso 1** desordena hasta recuperar la imagen original y el **caso 2** desordena el número de veces que se indique.

Entradas:

photo: la imagen que queremos desordenar. Debe satisfacer los requisitos necesarios.

A: la matriz que determina la transformación. Debe satisfacer los requisitos necesarios.

Salida: el número de veces que hemos transformado la imagen.

Aunque no sea un parámetro de salida, también debe mostrar una animación que comience con la imagen original y muestre todas las imágenes transformadas que se hayan ido realizando.

También debe guardar la última imagen de la sucesión de imágenes modificadas.

Ejemplos: (no se muestran las animaciones, pero hay que incluirlas)

```
>> power = arnold_02('hypatia_peq.bmp', [2 1;1 0])

Introduce 1 to disorder until getting the original image or introduce 2 to apply ...
    a certain number of transformations: 1

power=48
```

```
>> power = arnold_02('hypatia_peq.bmp', [2 1;1 0])

Introduce 1 to disorder until getting the original image or introduce 2 to apply ...
    a certain number of transformations: 2

How many transformations do you want to apply? 10

power=10
```