Tarea 07

fl.gomez10 at uniandes.edu.co

20 de marzo de 2019

Horario de atención: Principalmente de 2:00pm a 5:00pm en la oficina i-109. También se pueden enviar dudas al correo electrónico.

Entregar la carpeta de trabajo en un archivo comprimido `hw07-username.tar` antes de finalizar la clase.

Trabaje iniciando sesión en la máquina virtual en línea mybinder.org/ [1]. Aparte, cree un archivo de texto llamado `bitacora.txt`

1. Ejercicio 1 (30 puntos) Trabajo en Casa

Cree un notebook llamado `ejercicio01.ipynb`. En la primera celda puede incluir `%pylab inline` para cargar `numpy` y `matplotlib` de una vez.

1.1. A (6pts)

Graficar la función coseno

- (2 pts) Cree un array unidimensional `a` que tenga 30 números desde -2π hasta 2π igualmente distanciados usando `np.linspace()`
- (2 pts) Cree un array unidimensional `b` que sea el coseno de `a`. Créelo directamente operando sobre el array `a` como un todo, no elemento por elemento.
- (2 pts) Grafique `b` vs. `a`.

1.2. B (14 pts)

Crear un array de 8×8 que tenga el patrón del tablero de ajedréz.

- (2pts) Cree un array de ceros de 8×8 usando `np.zeros()`.
- (4pts) Con un doble `for` (uno para barrer filas y otro para barrer columnas), recorra el array y coloque unos cada tanto siguiendo el patrón del tablero de ajedréz intercalando unos y ceros en ambos ejes (0 y 1).

[1] https://mybinder.org/v2/gh/ComputoCienciasUniandes/FISI2026-201910/master?urlpath=lab

- (4pts) Grafique usando `plt.imshow`.
- (4pts) Cree e imprima un array de $8 \times 8 \times 4$ con un patrón de ajedréz 3D. Esto es, si uno se desplaza en cualquier eje (0, 1 o 2) va a encontrar intercalados unos y ceros.

1.3. C (10 pts)

Cree una variable $N = 4$. Cree un array de $N \times N$ donde la matriz diagonal superior sean ceros, la diagonal sean unos y la matriz diagonal inferior se llene incrementando del siguiente modo:

$$\begin{matrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 4 & 1 & 0 \\ 5 & 6 & 7 & 1 \end{matrix} \tag{1}$$

Debe funcionar bien para los casos $N = 0, 1, ..., 10$. Con esto se calificará. (10 pts). Puede empezar creando un array de ceros usando `np.zeros()`

2. Ejercicio 2 (30 pts)

En un notebook llamado `ejercicio02.ipynb` copie el siguiente fragmento de código.

```
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

img=mpimg.imread('https://github.com/ComputoCienciasUniandes/FISI2026-201910/raw/master/Talleres/Grupo_1/sorpresa_hubble.png')

imgplot = plt.imshow(img)
```

Listing 1: grafica-cos.py

- (10 pts) Escriba en una celda la forma (shape) de `img`. ¿Es un solo array? ¿Son varias capas? ¿Qué representa cada capa?
- (10 pts) Cree una variable $k > 2{,}0$. Con esta cree un nuevo array de la forma `img**k`. Grafique con `plt.imshow()`. En otra celda indique qué tipo de operación se está realizando.
- (10 pts) Aumente el valor de `k` hasta poder ver claramente la imágen. ¿Cuál es el valor crítico de "k" que le permite ver el resultado?

Guarde el notebook con el resultado.

3. Ejercicio 03 (40 pts) El Juego de la Vida de Conway

Cada celda puede estar viva o muerta. Este será el estado de la celda. Cada celda tiene ocho celdas vecinas.

Todas las celdas evolucionan al mismo tiempo según tres reglas:

Una celda viva con 2 o 3 celdas vecinas vivas sigue viva. Una celda muerta con exactamente 3 celdas vecinas vivas se convierte en una celda viva. En los demás casos las celdas mueren o siguen muertas.

- (20 pts) Cree un array de ceros de 10×10. Defina una función que haga evolucionar el sistema según las reglas de Conway. Pruebe con

```
A[3,3] = 1
A[3,4] = 1
A[3,5] = 1

A[6,6] = 1

```

Listing 2: Array de prueba.

  y déjelo evolucionar durante 10 ciclos.

- (10 pts) Ahora extienda el sistema a un array de 20×20, inicie aleatoriamente con una probabilidad p=0.4 de obtener un uno.

- (10 pts) Deje evolucionar el sistema por 40 ciclos.