Fundamentos de Programación

PAC 8 - Enunciado

En este Notebook se encontrarás el conjunto de actividades evaluables como PEC de la asignatura. Verás que cada una de ellas tiene asociada una puntuación, que indica el peso que tiene la actividad sobre la nota final de la PEC. Adicionalmente, hay un ejercicio opcional, que no tiene puntuación dentro de la PEC, pero que se valora al final del semestre de cara a conceder las matrículas de honor y redondear las notas finales. Podréis sacar la máxima nota de la PAC sin necesidad de hacer este ejercicio. El objetivo de este ejercicio es que sirva como pequeño reto para los estudiantes que quieran profundizar en el contenido de la asignatura.

Veréis que todas las actividades de la PEC tienen una etiqueta, que indica los recursos necesarios para llevarla a cabo. Hay tres posibles etiquetas:

- `NM` **Sólo materiales**: las herramientas necesarias para realizar la actividad se pueden encontrar en los materiales de la asignatura.
- `EG` **Consulta externa guiada**: la actividad puede requerir hacer uso de herramientas que no se encuentran en los materiales de la asignatura, pero el enunciado contiene indicaciones de dónde o cómo encontrar la información adicional necesaria para resolver la actividad.
- `EI` **Consulta externa independiente**: la actividad puede requerir hacer uso de herramientas que no se encuentran en los materiales de la asignatura, y el enunciado puede no incluir la descripción de dónde o cómo encontrar esta información adicional. Será necesario que el estudiante busque esta información utilizando los recursos que se han explicado en la asignatura.

Es importante notar que estas etiquetas no indican el nivel de dificultad del ejercicio, sino únicamente la necesidad de consulta de documentación externa para su resolución. Además, recordad que las **etiquetas son informativas**, pero podréis consultar referencias externas en cualquier momento (aunque no se indique explícitamente) o puede ser que podáis hacer una actividad sin consultar ningún tipo de documentación. Por ejemplo, para resolver una actividad que sólo requiera los materiales de la asignatura, puedéis consultar referencias externas si queréis, ya sea tanto para ayudaros en la resolución como para ampliar el conocimiento!

En cuanto a la consulta de documentación externa en la resolución de los ejercicios, recordad **citar siempre la bibliografía utilizada** para resolver cada actividad.

Ejercicio 1

Resuelve el siguiente ejercicio mediante las librerías de Seaborn para la visualización.

El año 2016 se celebraron los Juegos Olímpicos en Río de Janeiro. Utilizando la página web de Rio2016, hemos descargado el fichero `athletes.csv` donde podemos ver las algunos datos de sus participantes.

a) Carga los datos en un dataframe de pandas, obtiene:

- Los nombres de las columnas
- El número de atletas
- El número de países
- Muestra las 5 primeras entradas del dataframe

`NM` **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

b) A partir del dataset, representa mediante seaborn:

- Un diagrama de barras (barplot) con el total de medallas de oro por deporte, ordenando las barras de mayor a menor, con el número de medallas en el eje horizontal.
- Un diagrama de barras (barplot) con el total de medallas de oro por país (mostrando solo los países que hayan obtenido más de dos medallas de oro), ordenando las barras de mayor a menor, con el número de medallas en el eje horizontal.

`NM` **(1 punto)**

In [ ]:
```
# Respuesta
```

c) Visualiza ahora

- Un histograma de la altura de los participantes. (utilizando `displot`)
- Un histograma del peso de los participantes. (utilizando `displot`)
- Un diagrama de puntos donde veamos la altura en un eje y el peso en el otro, mostrando cada punto en un color diferente en función del género, y una distribución marginal en función del género en cada eje. (utilizando `jointplot`)

Explica que información podemos obtener de los tres gráficos realizados.

`EG` **(1 punto)**

In [ ]:
```
# Respuesta
```

d) Crea una nueva variable en el dataframe que nos enseñe el número de medallas ganadas (independientemente de su tipo), y representa un *violinplot* contra el BMI, mostrando de forma diferenciada el género.

Nota: BMI se calcula con el peso de una persona en kilogramos dividido por el cuadrado de la estatura en metros

`EI` **(1 punto)**

In [ ]:
```
# Respuesta
```

e) Seguramente hemos visto cierta correlación entre algunos atributos. Utiliza un mapa de calor pera visualizar las posibles correlaciones entre los atributos numéricos de este último dataframe. ¿Qué información podemos extraer del mapa de calor generado?

`NM` **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 2 (NETWORKX)

Considera los siguientes pares de conexiones:

```
('St. Louis', 'Miami'), ('St. Louis', 'San Diego'),
('St. Louis', 'Chicago'),
('San Diego', 'Chicago'), ('San Diego', 'San Francisco'),
('San Diego', 'Minneapolis'),
('San Diego', 'Boston'),   ('San Diego', 'Portland'),
('San Diego', 'Seattle'),
('Tulsa', 'New York'),  ('Tulsa', 'Dallas'),  ('Phoenix', 'Cleveland'),
('Phoenix', 'Denver'),   ('Phoenix', 'Dallas'),
('Chicago', 'New York'),   ('Chicago', 'Los Angeles'),
('Miami', 'New York'),   ('Miami', 'Philadelphia'),
('Miami', 'Denver'),
('Boston', 'Atlanta'),
('Dallas', 'Cleveland'),  ('Dallas', 'Albuquerque'),
('Philadelphia', 'Atlanta'),
('Denver', 'Minneapolis'),
('Denver', 'Cleveland'),
('Albuquerque', 'Atlanta'),
('Minneapolis', 'Portland'),
('Los Angeles', 'Seattle'),
('San Francisco', 'Portland'), ('San Francisco', 'Seattle'),
('San Francisco', 'Cleveland'),
('Seattle', 'Portland')`
```

a) Usando `networkx`, crea un grafo no dirigido con las conexiones declaradas anteriormente. Representa este grafo con `spring_layout`, y con los nombres de las ciudades representadas en cada nodo.

`EI` **(1 punto)**

In [ ]:
```
# Respuesta
```

b) Calcula utilitzando las herramientas de NetworkX todas las posibles rutas entre `Philadelphia` y `San Francisco`, y muéstra la primera (la que cruza menos ciudades) y la última (la que cruza más ciudades). ¿Cuantas rutas aparecen? ¿Cuantas ciudades contiene la más larga ?

NOTA Puedes consultar en shortest_simple_paths cómo generar caminos entre nodos.

`EG` **(1 punto)**

In [ ]:
```
# Respuesta
```

c) ¿Puedes mostrar el nombre de las estaciones que son de paso? (aquellas que solo tienen dos vecinos) utilitzando las funciones que nos proporcionan NetworkX?

`EI` **(1 punto)**

In [ ]:
```
# Respuesta
```

Ejercicio 3

Vamos a intentar representar las rutas que hemos encontrado anteriormente sobre un mapa. Para eso realizaremos los siguientes pasos:

a) Crea un GeoPandas dataframe, con el nombre de cada ciudad y su longitud y latitud.

Nota: debereis instalar la libreria geopandas utilizando *"pip install geopandas"*

Nota: puedes extraer la localización de cada ciudad mediante geopy

`EI` **(1 punto)**

In [ ]:
```
# Respuesta
```

b) Representa un mapa de Estados Unidos con las ciudades marcadas en forma de puntos de color azul.

Nota: quizas necesitareis instalar la libreria mapclassify utilizando *"pip install mapclassify"*

`EI` **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

Ejercicio 4

En la carpeta de datos verás un fichero con nombre `data/provincias_covid19_datos_sanidad_nueva_serie.csv`.

a) Importa los datos en un objeto pandas, de forma que la fecha esté como índice del `data.frame`. Lista las primeras 5 entradas, los nombres de las columnas y las diferentes provincias que contienen los datos.

`NM` **(0.5 puntos)**

In [ ]:
```
# Respuesta
```

b) Realiza una visualización interactiva mediante `ipywidgets` con las siguientes propiedades.

- Se pueda seleccionar la variable a representar entre `Casos`, `Fallecidos` y `Hospitalizados`, de forma que se permitan selecciones múltiples (representar una o más variables).
- Se represente la evolución de todas las variables seleccionadas en el eje vertical, contra el tiempo en el eje horizontal. La provincia debe aparecer en el título de la representación.
- Se permita seleccionar la provincia a visualizar mediante un segundo selector.
- La representación debe incluir una rejilla o `grid`.
- Si incorpore una leyenda con la información de las trazas representadas.

`NM` **(1 punto)**

In [ ]:
```
# Respuesta
```

Ejercicio opcional

Dada la serie

$$Z_n = z_{n-1}^3 + c$$

1. Escribe una función $f(c, z_0, A, N_max)$, que devuelva el número iteraciones N de la serie anterior necesarias para que el módulo $|z| > A$, siendo $A \in \mathbb{R}$.
2. Calcula y representa una imagen 2D representando N sobre el dominio de z_0 comprendido por $Re(z_0) \subset [-2, 2]$, $Im(z_0) \subset [-1.2, 1.2]$ con los parámetros ($A = 2, n_{max} = 100, c = 0.8 + 0.6j$), Resolución de la imagen = (800,1200) pixels).

Nota: Te puede ser útil la función `meshgrid` de `numpy`.

`EI`

In [ ]:
```
# Respuesta
```