Fundamentos de Programación

PEC 4 - Librerías Científicas en Python

En este Notebook se encontraréis el conjunto de actividades evaluables como PEC de la asignatura. Veréis 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 algún 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 PEC sin necesidad de hacer estos ejercicios. El objetivo de estos ejercicios 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.

NumPy - Ejercicios

Ejercicio 1

(a) Crea una matriz A de dimensiones 6x6 formada por números aleatorios comprendidos entre 0 y 9. **(0.25 puntos)** NM

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

(b) Vamos ahora a calcular ciertas operaciones útiles sobre esta matriz. Calcula:

- La traza de la matriz, $tr(A)$.
- El determinante de la matriz, $det(A)$.
- La traspuesta de A, denotada por A^T.
- La norma de Frobenius de la matriz, $||A||_F$.
- La inversa de la matriz, A^{-1}.
- La multiplicacion de A por su inversa A^{-1}, es decir, $A \cdot A^{-1}$.

Y muestra los resultados en pantalla. Para realizar estas operaciones, tendrás que sumergirte en la documentación oficial de `numpy` , en particular, el apartado de operaciones de álgebra lineal (https://numpy.org/doc/stable/reference/routines.linalg.html), y encontrar las funciones que necesitas. Una de estas funciones no se encuentra en esa sección, pero seguro que puedes

encontrarla utilizando el buscador **(1 punto)** EG

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

(c) En el último subapartado tenías que hacer el producto de las matrices $A \cdot A^{-1}$. ¿Qué ocurre cuando haces este producto de una matriz con su inversa? ¿Qué ocurre si multiplicas las matrices haciendo lo siguiente y cuál es la diferencia con la operación del subapartado anterior? **(0.25 puntos)** EI

```
B = A*A_inv
```

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

Ejercicio 2

Vamos a intentar reproducir algunos de los resultados del ejercicio anterior utilizando el *slicing* de matrices.

(a) Crea una función llamada *transpuesta* que calcule manualmente la traspuesta de una **matriz cuadrada, independientemente de su dimensión**. Para este primer subapartado, podéis seguir estos pasos:

- Define una función que tome una matriz A como input.
- Dentro de la función, crea una matriz A_t de ceros.
- Como esta función debe funcionar independientemente de la dimensión de la matriz A, debes extraer la dimensión de esta matriz cuadrada y almacenarla en una variable, por ejemplo llamada *dim*.
- A continuación, tendrás que iterar utilizando un bucle *for* por las filas y las columnas, utilizando lo que se conoce como *for anidado*. Puedes guiarte según este ejemplo (https://www.tutorialspoint.com/transpose-a-matrix-in-python).

Comprueba que da lo mismo que en el apartado anterior utilizando la matriz A definida en el primer ejercicio. Puedes recordar la definición de la matriz transpuesta aquí (https://es.wikipedia.org/wiki/Matriz_transpuesta). **(0.5 puntos)** EG

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

(b) Esta vez, de forma independiente (sin guia), crea una función llamada *traza* que calcule manualmente la traza de una **matriz cuadrada, independientemente de su dimensión**. Para ello, utiliza un bucle *for* y *slicing* de matrices. **Comprueba que da lo mismo que en el apartado anterior utilizando la matriz A definida en el primer ejercicio.** Recuerda que la traza de una matriz se define como la suma de sus elementos en diagonal. Más información (https://es.wikipedia.org/wiki/Traza_(%C3%A1lgebra_lineal)). **(0.5 puntos)** NM

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

(c) Crea una función llamada *frobenius* que calcule manualmente la norma de Frobenius de una **matriz cuadrada, independientemente de su dimensión**. Para ello, utiliza un bucle *for* y *slicing* de matrices. **Comprueba que da lo mismo que en el apartado anterior utilizando la matriz A definida en el primer ejercicio.** Recuerda que la norma de Frobenius se define como la raíz cuadrada de la suma del cuadrado de todos los elementos:

$$||A||_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2}$$

Más información (https://es.wikipedia.org/wiki/Norma_matricial). **(0.5 puntos)** NM

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

(d) En el link anterior podéis ver la definición de la norma de Frobenius y, según la Wikipedia, es igual a la raíz cuadrada de la traza de $A \cdot A^T$, donde A^T es la traspuesta de A:

$$||A||_F = \sqrt{traza(A \cdot A^T)}$$

Como no nos fiamos de la Wikipedia, queremos comprobar si esto es verdad. Utiliza las funciones que has definido anteriormente para lo siguiente:

- Llama a la función *transpuesta()* para calcular la transpuesta de A.
- Llama a la función *traza()* para calcular la traza de $A \cdot A^T$. Luego cálcula la raíz cuadrada del resultado.
- Llama a la función *frobenius()* para comprobar si el resultado es el mismo.

(0.5 puntos) NM

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

(e) Crea dos funciones que calculen la multiplicación de matrices (o producto escalar) y la multiplicación elemento a elemento

para dos matrices no necesariamente cuadradas. **(Opcional)** EI

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

Pandas - Ejercicios

Ejercicio 3

En este ejercicio vamos a trabajar con un conjunto de datos sobre la **brecha salarial de género** en países europeos durante una serie de años. Encontraréis el dataset llamado `brecha_salarial_europa.csv` en la carpeta `data`.

(a) Carga el conjunto de datos especificando el separador entre columnas (https://pandas.pydata.org/docs/reference/api/pandas.read_csv.html) más pertinente. **(0.25 puntos)** EG

NOTA: Los valores de las columnas de los sectores representan la brecha salarial no ajustada, calculada como la diferencia entre el salario bruto medio por hora de los empleados masculinos y las empleadas femeninas como un porcentaje del salario bruto medio por hora de los empleados masculinos.

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

(b) A partir del `DataFrame` anterior, muestra las 10 primeras filas, las 10 últimas y muestra información estadística básica del dataset utilizando las funciones `describe` y `info`. Viendo estos resultados, ¿cuál es el menor año y mayor año que aparece en este dataset? **(0.5 puntos)** NM

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

(c) Almacena en una lista el nombre de todos los países que aparecen en este dataset de manera única (es decir, sin repeticiones). Almacena en otra lista los nombres de los sectores profesionales que aparecen en las columnas (es decir, "Industry", "Business", ... hasta "Other", sin incluir las que no son sectores). **Nota:** investiga independientemente cómo extraer los nombres de las columnas de un dataset con `pandas`. **(0.5 puntos)** EG

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

(d) Filtra los datos para obtener sólo los datos de *España*. ¿Cuál fue el sector con menor brecha de género en 2014 y cuál fue el valor? ¿Y el que tuvo mayor brecha en 2021, y cuál fue su valor? **Nota:** debes contestar estas preguntas utilizando elementos de programación, no observando visualmente los datos. Puedes investigar las funciones `max`, `min`, `idmax`, e `idmin` de *pandas*, las cuales puedes encontrar aquí (https://pandas.pydata.org/docs/reference/frame.html). **(0.5 puntos)** EG

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

(e) Agrupa los datos por *país*, mostrando en columnas el valor medio de su GDP (Producto Interior Bruto en inglés) y el porcentaje de población urbana. Ordénalos de mayor porcentaje de población urbana a menor. Las funciones necesarias las puedes encontrar en la documentación oficial de *pandas* (https://pandas.pydata.org/docs/reference/frame.html). **(0.75 puntos)** EG

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

Matplotlib - Ejercicios

Ejercicio 5

En este ejercicio vamos a utilizar la librería `matplotlib` para trazar la evolución temporal de los datos de brecha salarial. En particular, vamos a graficar la brecha salarial media en España (agregando todos los sectores), junto a su desviación estándar, en función del año. Tendrás que hacer lo siguiente:

- Calcular la media y desviación estándar de **todos los sectores**, sólo para España.
- Grafica la evolución de esta media en función del año utilizando `matplotlib`.
- Añade un sombreado con la desviación estándar utilizando la función fill_between (https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.fill_between.html) de `matplotlib`.
- Añade una línea horizontal en $y = 0$, para visualizar mejor el corte entre brecha positiva y negativa.

- Añade un título y etiquetas en el eje x e y.

Para que no haya dudas, esperamos una grafica similar a ésta (https://matplotlib.org/stable/plot_types/basic/fill_between.html) **(1.5 puntos)** EG

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

Ejercicio 6

Como habrás visto, esta gráfica no es muy informativa, ya que todos los sectores están agregados y no podemos ver si cada uno sigue un patrón diferente. Por tanto, vamos a graficarlos por separado en subplots (https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.subplots.html).

- Crea una gráfica tipo subplot con 20 columnas y 2 filas, de tamaño `(30x10)` y compartiendo el eje x y el eje y.
- Haz un bucle por cada una de estas sub-gráficas y sub-ejes y grafica en cada una de ellas uno de los 20 sectores. Como son muchas gráficas, para no hacerlo a mano, te será útil revisar ejemplos como éste (https://matplotlib.org/devdocs/gallery/lines_bars_and_markers/markevery_demo.html#sphx-glr-gallery-lines-bars-and-markers-markevery-demo-py), éste (https://matplotlib.org/devdocs/gallery/shapes_and_collections/hatch_style_reference.html#sphx-glr-gallery-shapes-and-collections-hatch-style-reference-py) o éste (https://matplotlib.org/devdocs/gallery/images_contours_and_fields/interpolation_methods.html#sphx-glr-gallery-images-contours-and-fields-interpolation-methods-py)
- Asegúrate de añadir un título a cada sub-gráfica para poder diferenciarlas.

En la documentación de subplots (https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.subplots.html) hay muchos ejemplos que te pueden ayudar a resolver este ejercicio.

(1.5 puntos) EG

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

SciPy - Ejercicios

Ejercicio 7

(a) Utiliza la función linregress (https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.linregress.html) de la librería `stats de scipy` para realizar un ajuste o fitting lineal de la brecha en función del tiempo en uno de los sectores de tu elección en España (¿por qué no probar el sector en el que trabajas o te gustaría trabajar?). **(0.5 puntos)** EG

(b) ¿Qué información puedes sacar de los valores como la pendiente, el punto de intersección, el coeficiente de correlación de Pearson y el p-value que te devuelve la función? ¿Ha sido un buen fitting? ¿Está mejorando o empeorando la brecha de género? **(0.5 puntos)** NM

(c) Grafica el fitting encima de los datos originales del sector para comprobar si los datos se ajustan o no. **(Opcional)** EG

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