# Programación para la ciencia de datos - PEC1

En este Notebook encontraréis un conjunto de ejercicios que conforman la primera actividad de evaluación continua (PEC) de la asignatura.

Para cada ejercicio, tened en cuenta que:

- **Es necesario incluir comentarios** de vuestro código, que expliquen cómo se ha implementado la solución del problema planteado.
- **Es imprescindible** citar las referencias consultadas para realizar la actividad. Se valorará que el código proporcionado solucione el problema propuesto y también la calidad del código (comentarios, legibilidad, claridad, uso de las estructuras de datos adecuadas, buena nomenclatura de las variables y funciones, seguimiento del PEP8, etc. ).

Veréis que cada una de las actividades tiene asociada una puntuación, que indica el peso que tiene la actividad sobre la nota de la PEC.

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

- `SM` **Sólo materiales**: las herramientas necesarias para realizar la actividad se pueden encontrar en los materiales de la asignatura (consideraremos también los materiales de la asignatura Fundamentos de Programación, así como las lecturas obligatorias de material externo que se indican en los notebooks).
- `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, recuerde que las **etiquetas son informativas**, pero puede consultar referencias externas siempre que lo desee (aunque no se indique explícitamente) o puede que pueda realizar una actividad sin consultar ningún tipo de documentación. Por ejemplo, para resolver una actividad que sólo requiera los materiales de la asignatura, puede consultar referencias externas si se desea, ya sea tanto para ayudar en la resolución como para ampliar conocimientos.

---

## Antes de empezar

Leed atentamente los parágrafos que siguen, relacionados con la originalidad en las actividades, antes de empezar a trabajar en la PEC. Si os surge cualquier duda dirigiros al profesor colaborador de la asignatura antes de continuar con la actividad.

La falta de originalidad se produce cuando en una actividad aparece contenido que no ha sido elaborado de forma individual por el estudiante y no se referencia adecuadamente su origen. O bien cuando, aunque el contenido externo esté referenciado, este es tan extenso que no es posible considerar al estudiante el autor o autora de la actividad.

Así, algunos ejemplos de comportamientos inadecuados debido a falta de originalidad son:

1. Crear la solución de un ejercicio de la PEC en colaboración entre diversos estudiantes.
2. Incluir un ejercicio de la PEC que utiliza código encontrado en internet sin citar la fuente.
3. Compartir vuestra solución de la PEC con otros estudiantes de la asignatura.

# Instrucciones de entrega

Para proceder a la entrega de la actividad es necesario realizar los siguientes pasos:

1. Comprobad que el notebook se ejecuta correctamente en Google Colaboratory. Es importante que antes de entregar vuestra PEC os aseguréis que la versión final del código ejecuta correctamente en su totalidad. Para ello, se recomienda hacer una ejecución completa desde cero del notebook haciendo click en *"Entorno de ejecución y ejecutar todas",* y comprobando que todas las celdas del notebook se ejecutan correctamente.
2. Confirmad que sois los autores únicos de la PEC y que esta incluye todas las citas a recursos externos que se hayan utilizado para elaborarla. Con el fin de confirmar que sois los autores únicos de la actividad, añadid vuestro nombre completo en la celda siguiente.
3. Entregad el notebook correspondiente a la resolución de la PEC a través del Libro de notas del aula. Podéis descargar el notebook mediante *"Archivo, descargar y descargar .ipynb"*

Yo, *Nombre y Apellidos*, confirmo que he elaborado de forma individual todas las actividades resueltas en esta PEC y que he incluido las citas a todas las fuentes externas que he utilizado para resolver las actividades.

# Type hint

Os recomendamos resolver esta PEC usando type hints. Podemos entender los type hints de Python como una solución formal para indicar estéticamente el tipo de valor dentro del código Python. Os recomendamos la visualización del siguiente vídeo para una introducción a los type hints en Python, y para entender por qué son interesantes.

**El uso de type hints en la resolución de la PEC se bonificará con 0.25 puntos adicionales (para obtener toda la puntuación adicional, hay que implementar los type hints de todas las funciones implementadas en la PEC). No es necesario que utilicéis los type hints en todas las variables sino sólo en los encabezados de las funciones.**

# Enunciado

Los siguientes ejercicios se realizarán utilizando el dataset people.csv proporcionado juntamente con la PEC, y que contiene información sobre diferentes personas. Cada fila representa una persona.

Los campos del dataset son los siguientes:

- Index: Identificador de registro ordenado ascendentemente.
- User Id: Código que identifica de forma única a la persona.
- First Name: Nombre de la persona.
- Last Name: Apellido de la persona.
- Sex: Sexo de la persona.
- Email: Correo electrónico de la persona.
- Phone: Teléfono de la persona.
- Date of Birth: Fecha de nacimiento de la persona.
- Job Title: Descripción corta del oficio de la persona.

**IMPORTANTE:** La carga del dataset se realizarà utilizando **rutas relativas**, tenéis que cargar el fichero teniendo en cuenta que trabajáis en una máquina remota y que podéis subir los ficheros en la misma ruta que este notebook. Por lo tanto, tenéis que tener en cuenta subir el fichero .csv a vuestro Drive, juntamente con una cópia de este fichero de notebook que utilizaréis para resolver los problemas.

# Ejercicio 1

Cargad el dataset de forma que obtengáis un objecto de tipo dataframe.

Una vez cargado el dataset aplicaremos las siguientes modificaciones:

1. Número de nombres diferentes.
2. Convertir la columna de fecha de nacimiento al formato Dia/Mes/Año (dd/mm/aaaa).
3. Las filas que tengan el mismo Email, dejar solamente una. Eliminad todas las filas que contengan este mismo Email, manteniendo solamente el primer registro.
4. Una vez eliminadas las filas, comprobad que el número de registros totales coincide con el número de emails diferentes (realiza la verificación usando el test en la casilla posterior del código implementado)
5. Algunos empleos están escritos entre ". Elimina el símbolo " de la columna empleo para todos los registros.
6. Visualiza los 20 primeros elementos del dataset.

Pista 1: Recordad qude podéis utilizar librerías como 'pandas' para cargar el conjunto de datos de forma sencilla y obtener un dataframe.

Pista 2: El type hint de un dataframe es pd.DataFrame.

**Al finalizar este ejercicio, obtendremos un dataset con las modificacions realizadas. Este dataset, lo utilizaremos para los siguientes ejercicios.**

SM **(2 puntos)**

In [ ]:
```
```

In [ ]:
```
# Test
assert numeroEmailsDiferentes != len(myDict) , "The list and the total amount of Emails are not
```

# Ejercicio 2

2.1 Cread un diccionario usando Dictionary Comprehensions tal que:

- Sus claves correspondan al nombre de persona (**First Name**).
- Sus valores correspondan a cuántos registros contienen este nombre.

2.2 Mostrad las 15 primeras entradas del diccionario (clave y valor), una vez lo hemos ordenado (de mayor a menor) según el número de filas del dataframe que contienen cada valor.

SM **(1 punto)**

> **Solución**

In [ ]:
```
```

# Ejercicio 3

Utilizando el dataframe obtenido en el Ejercicio 1

3.1 cread una función tal que tenga como parámetros de entrada:

- La estructura de datos (dataframe obtenido en el Ejercicio 1)
- User Id
- Sexo

La función nos retornará un **booleano** que indique si la persona (User Id) coincide con el sexo especificado en el parámetro de entrada.

3.2 Probad vuestra función para la persona ***33730caFF13Ff4F*** y mostrad el resultado por pantalla visualizando el nombre de la persona y especificando si el parámetro de entrada coincide o no con el sexo. Por ejemplo: "El sexo especificado en el parámetro de entrado para Kirsten es correcto".

En la celda posterior a la solución planteada, realizad la prueba de funcionamiento mediante un assert.

SM **(1 punto)**

> **Solución**

In [ ]:

In [ ]:
```
#Test
assert funcionComprobacion(df, id, res['Sex'].item()) == True
```

# Ejercicio 4

Utilizando el dataframe obtenido en el Ejercicio 1

4.1 Cread una función tal que tenga como parámetros de entrada:

- Nombre del campo (columna) a tratar dentro del dataframe.
- Valor de repetición.

La función imprimirà el **First name** de los registros encontrados y retornará un valor **entero** que indicará el número de repeticiones del "valor de repetición" del parámetro de entrada.

Imprimid el resultado con el formato : "La palabra X en la columna Y, aparece N veces". Realizad la prueba para la columna "Date of birth" y el valor "1921-08-17".

SM **(1.5 puntos)**

In [ ]:

# Ejercicio 5

Usando el Dataset obtenido del ejercicio 1, os pedimos lo siguiente:

5.1 Cread una lista con los empleos "Job Title" de las personas sin repeticiones.

5.2 Ordenadla (alfabéticamente, el orden establecido por Python es ya suficidente) y mostrad los 10 primeros valores.

5.3 Cread un diccionario (se valorará el uso de Dict comprenhension) tal que:

- Las claves correspondan al First Name.
- Los valores correspondan al número de conocimientos que contiene el "Job title". Consideramos que una persona tiene varios conocimientos en el "Job Title", separados por comas.

5.4 Del apartado anterior mostrad debidamente formatado cuál es el nombre de persona (First Name) que tiene un número mayor de conocimientos y cuántos conocimientos tiene.

Utilizad Dictionary Comprehensions.

SM **(1.5 puntos)**

> **Solución**

In [ ]:

# Ejercicio 6

En este ejericio practicaréis expresiones regulares, para ello os pedimos:

6.1 Encontrad todos los registros tales que en su número de teléfono tenga un prefijo de país entrado. Los prefijos estan divididos por los símbolos '()'. Interpretamos que si aparece el símbolo '(' ya se supone un prefijo sin esperar cerrar el paréntesis.

- Mostrad por pantalla cuántos registros con prefijo en el teléfono habéis encontrado
- Mostrad por pantalla el nombre de usuario y el teléfono de los registros encontrados.

6.2 Encuentra todos las personas que su nombre (First Name) sea un nombre compuesto. Es decir, que su nombre contenga más de una palabra.

- Muestra por pantalla cuántas personas hemos encontrado.
- Visualiza los nombres compuestas de todas las personas.

SM **(1 punto)**

> **Solución**

In [ ]:

# Ejercicio 7

7.1 Cread un diccionario usando Dictionary Comprehensions tal que:

- Sus claves correspondan al dominio (el dominio de xxx@gmail.com es gmail.com) obtenido de la columna Email y el uso de expresiones regulares.
- Sus valores correspondan al número de coincidencias del dominio (cantidad de personas que su correo pertenece al dominio).

7.2 Mostrad el resultado obtenido visualizando los dominios y el número de apariciones de cada uno de ellos.

`SM` **(1 punto)**

> **Solución**

In [ ]:

# Ejercicio 8

8.1 A qué estructura de datos estudiada en teoría corresponde el ir añadiendo personas (registros)? Teniendo en cuenta que cuando eliminamos una persona (registro) siempre quitamos el último que se ha hecho. Justificad la respuesta.

8.2 Simulad programando cómo añadirías dos personas (inventa los valores) y cómo después borras la última persona añadida.

`SM` **(1 punto)**

> **Solució**

In [ ]: