



Tecnológico
de Monterrey

| Educación
Continua

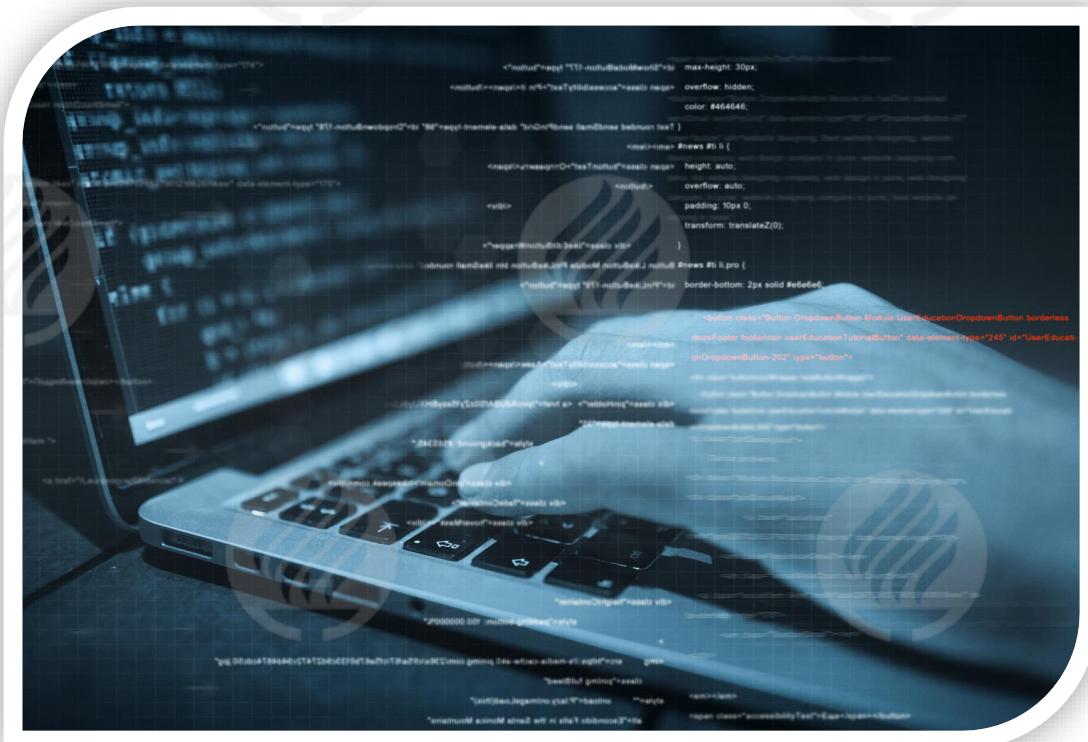
Plataformas de Aplicación sobre Python

Programas LIVE

SS1 | Aprender

Bienvenida y presentación

Bienvenida



“Saber dónde encontrar la información y cómo usarla, ese es el secreto del éxito”

-Albert Einstein

Objetivo principal del módulo



- **Diseñarás programas** en las plataformas de **Pandas y Numpy para Python** sobre el ambiente de programación **Colab notebook**, para que se cumpla con los requerimientos de la **aplicación de ciencia de datos requerida**

Objetivos particulares de la sesión



- Identificarás las bondades de usar una **plataforma** para apoyar el **proceso de soluciones con Python**
- Reconocerás cuándo trabajar con los **frameworks** NumPy y Pandas, así como sus **estructuras de datos y operaciones fundamentales**
- Integrarás los conocimientos adquiridos para plantear **soluciones** complejas de una **manera rápida y eficiente**

Agenda

1

Sesión Síncrona 1

Aprender

2

Trabajo asíncrono 1

Profundizar | Ruta de Aprendizaje

3

Sesión Síncrona 2

Preparar para Aplicar

4

Trabajo asíncrono 2

Aplicar en el trabajo | Reto



2



3



4



Dinámica de participación y convivencia

Sesión grupal:

- Las dudas durante la sesión se manejarán a través del chat. Éstas serán respondidas en los momentos definidos para ello
- Participar en las dinámicas propuestas durante la sesión de acuerdo con las instrucciones dadas para cada una

Rooms:

- Los equipos se harán de manera aleatoria y serán sólo para las actividades de práctica
- Estar atentos a la notificación de zoom para unirse al room
- Seguir al pie de la letra las instrucciones para la dinámica en rooms
- Informar al moderador si te encuentras sólo en el room para reasignarte
- Te pedimos no salir del room que tienes asignado. Cualquier reasignación será realizada por parte del tutor o staff

Conociendo más de los participantes



 **Mentimeter**

Please enter the code

<https://www.menti.com>



Introducción

- El desconocimiento de la **integración de plataformas** en nuestros desarrollos **incrementa la complejidad** y el tiempo de programación, así como las probabilidades de error
- Utilizar plataformas como **NumPy** y **Pandas** en Python es una forma inteligente y rápida de obtener **resultados confiables en nuestras aplicaciones de ciencia de datos**



Tecnológico
de Monterrey

Educación
Continua

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Plataformas / Frameworks

Tema 2: Arreglos y matrices con NumPy

Tema 3: Estructuras de datos en Pandas

Tema 4: Manipulando datos Tabulares con Pandas

Cierre de sesión

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Plataformas / Frameworks

Tema 2: Arreglos y matrices con NumPy

Tema 3: Estructuras de datos en Pandas

Tema 4: Manipulando datos Tabulares con Pandas

Cierre de sesión



- En la actualidad existen muchas plataformas que incrementan las **capacidades** de los **lenguajes de programación**
- El esfuerzo derivado de su **omisión** se incrementa considerablemente en un **desarrollo por la cantidad de código a generar y la propensión a cometer errores**
- La integración de estas plataformas sobre un lenguaje de programación como **Python** **mejorará la experiencia del programador**
- **Agilizando la implementación de nuestras aplicaciones**, disminuyendo los fallos y proporcionando **confianza y seguridad** en nuestros resultados

Instalación y Uso | Tema 1



- Una plataforma o marco de trabajo es una **estructura conceptual** y **tecnológica de asistencia definida**
- Cuenta normalmente, con artefactos o **módulos concretos de software**, que pueden servir de base para la organización y **desarrollo de software**



Debería utilizarse por los siguientes motivos:

- **Acelerar** el proceso de desarrollo del software
- **Evitar** errores de codificación
- **Reutilizar** código mediante su modularización
- **Utilizar** buenas prácticas de programación
- **Eliminar** vulnerabilidades en las aplicaciones

Al utilizar Colab de Google podemos:

- **Verificar** si una plataforma / framework está instalada
 - ✓ `!pip show nombre_plataforma`
- **Instalar** una plataforma
 - ✓ `!pip install nombre_plataforma`

```
1 #!/usr/bin/env python
2 import sys
3 import os
4 import simpleknn
5 from bigfile import BigFile
6
7 if __name__ == "__main__":
8     trainCollection = 'toydata'
9     nimages = 2
10    feature = 'f1'
11    dim = 3
12
13    testCollection = trainCollection
14    testset = testCollection
15
16    featureDir = os.path.join(rootpath, trainCollection)
17    searcher = simpleknn.load_model(os.path.join(featureDir, 'model'))
```

Para utilizar una plataforma debemos:

- Importarla
 - ✓ `import numpy as np`
- Llamar a los **métodos, atributos u objetos** para trabajar con ellos
 - ✓ `np.arange(15)`

Actividad | Aprendizaje activo



1. Entra a **Google Colab** en la liga:
<https://colab.research.google.com>
2. Crea un nuevo **Notebook**
3. Verifica si en tu **entorno de Colab** ya están disponibles las plataformas:
 - numpy
 - pandas

En caso de que alguna no esté disponible,
instálala

Duración: **5 minutos**



- Como pudiste darte cuenta, hay muchas **plataformas** que enriquecen el ecosistema de Python y aportan **poder y simplicidad** en la creación de nuestras aplicaciones.
- ¿Habías utilizado alguna **plataforma** anteriormente?
- Ahora que conoces sus beneficios, antes de programar desde cero alguna aplicación **¿Buscarías una plataforma que te ayude en la solución?**

Cierre: Concepto clave | Tema 1

Las plataformas agilizan el desarrollo de aplicaciones y potencializan las capacidades de nuestras soluciones con código estandarizado, optimizado y seguro.

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

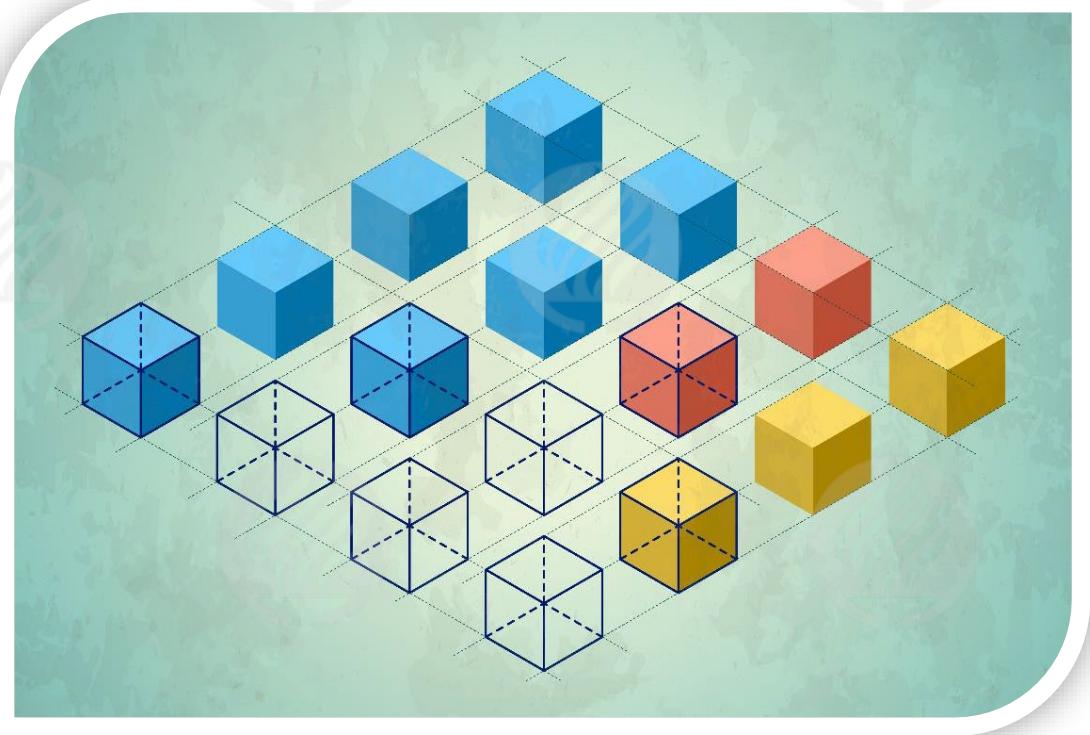
Tema 2: Arreglos y matrices con NumPy

Tema 3: Estructuras de datos en Pandas

Tema 4: Manipulando datos Tabulares con Pandas

Cierre de sesión

Propósito | Tema 2



- El **crecimiento acelerado** de los volúmenes de información en todos los ámbitos ha impulsado la **creación de estructuras y mecanismos adecuados** para su almacenamiento y procesamiento
- Saber manipular **estructuras potentes**, como los **arreglos y matrices** de la plataforma **Numpy**, nos permitirá **explotar grandes cantidades de datos con poco esfuerzo**

Arreglos | Tema 2.1

Subíndices	Arreglo
0	37
1	65
2	23
3	56
4	12

- **NumPy (Numerical Python)** es una plataforma para cómputo científico con Python
- Contiene la clase **ndarray** para trabajar con: arreglos de una o más dimensiones
- Para manipularlo se necesita un **subíndice**
- Sus **atributos** más importantes son: **ndim, shape, size, dtype, itemsize** y **data**

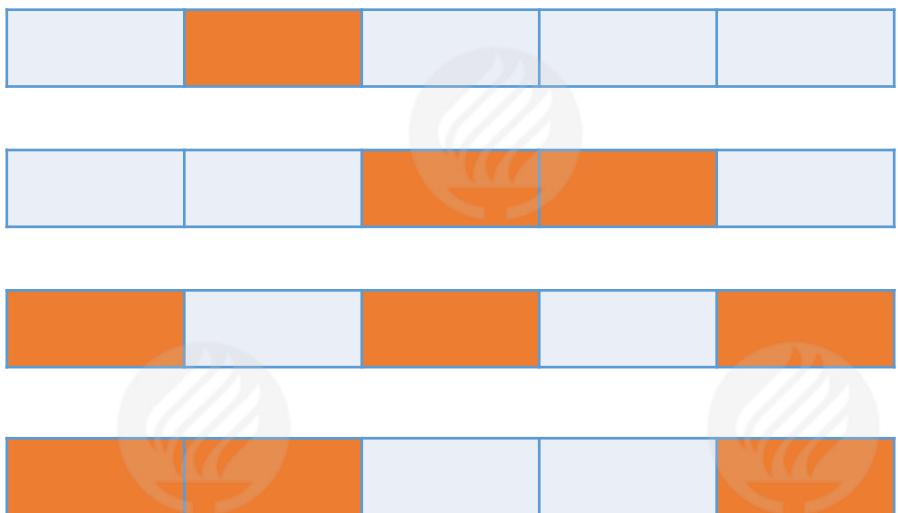
Arreglos | Tema 2.1



Podemos crear arreglos por medio de:

- Una lista
- Una tupla
- Con **funciones** como: **zeros**, **ones**, **arange** y **linspace**

Arreglos | Tema 2.1



**Podemos acceder a sus
elementos por:**

- Indexación (un solo elemento)
- Rebanado (varios elementos)
- Iteración (varios elementos)
- Condicionando sus valores
(varios elementos)

Arreglos | Tema 2.1

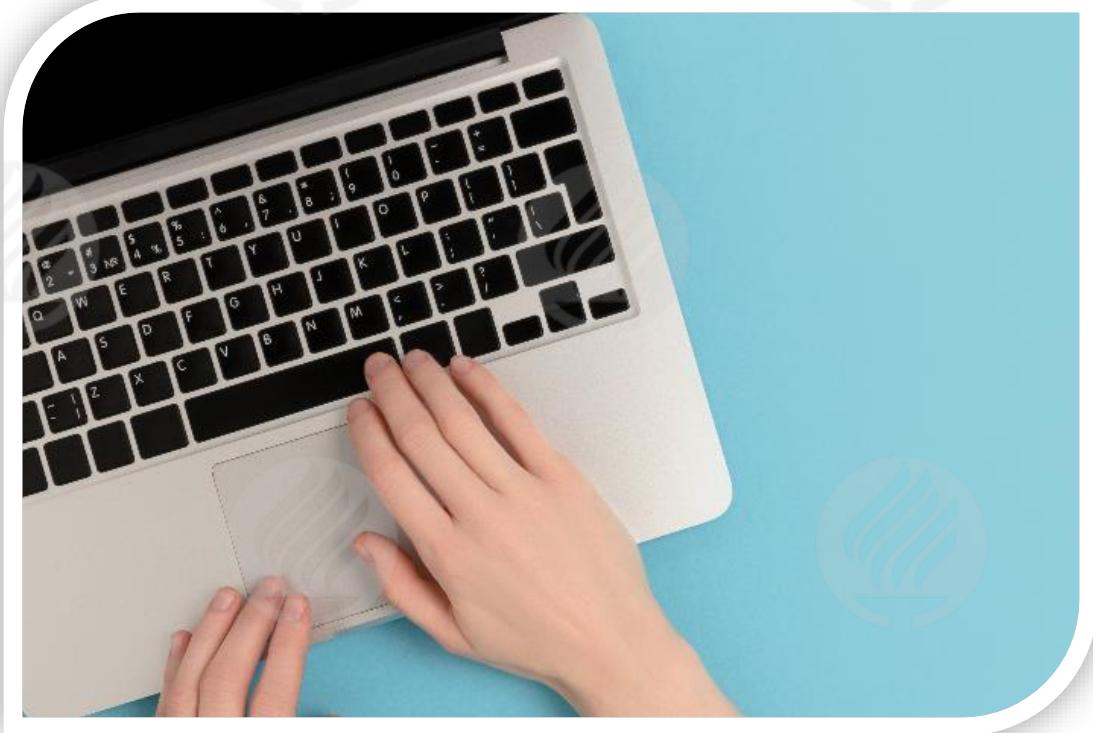
5	8	23	56	3
---	---	----	----	---

5	8	23	56	3
+				
7	26	15	8	13

Las operaciones con los arreglos pueden dividirse en:

- **Operaciones unarias** o sobre un único arreglo. Por ejemplo:
 - ✓ Obtener la suma de sus elementos, sacar el **valor máximo**, ordenarlo, etc.
- **Operaciones con otro arreglo** (siempre y cuando sea del mismo tamaño). Por ejemplo:
 - ✓ Obtener la **suma de arreglos**

Actividad | Aprendizaje activo



Crea una nueva libreta en Colab para:

1. Crear **dos arreglos** de 7 números enteros con la función arange
2. Obtener el **promedio** de los mínimos de ambos arreglos
3. Obtener los elementos de las **posiciones pares** del primer arreglo
4. Sumar **ambos arreglos**, ordenar el resultado y obtener los **tres primeros elementos del resultado**

Duración: **10 minutos**



- Los **arreglos** son muy útiles para **manejar** y **procesar** información por medio de una única variable y con poco código. Representan muy bien **estructuras de la vida real**, como una lista de calificaciones, una lista de compras, entre otras
- Piensa en dos ejemplos donde usaste o podrías usar **arreglos** para **manipular tu información**

¿Te imaginas cómo es el funcionamiento de un **arreglo con **más de una dimensión**?**

Matrices | Tema 2.2

Matriz

Subíndices	0	1	2
0	37	65	23
1	56	12	67
2	34	98	52

- Una **matriz** es un arreglo de dos dimensiones
- Al igual que en los arreglos de una dimensión, se utiliza la clase **ndarray** para trabajar con matrices
- Su manipulación requiere dos **subíndices**
- Se pueden emplear los **mismos atributos** que en los arreglos unidimensionales: **ndim, shape, size, dtype, itemsize y data**

Matrices | Tema 2.2



Podemos crear matrices por medio de:

- Listas de listas
- Las funciones: **zeros, ones y arange**

37	65	23
56	12	67
34	98	52

37	65	23
56	12	67
34	98	52

37	65	23
56	12	67
34	98	52

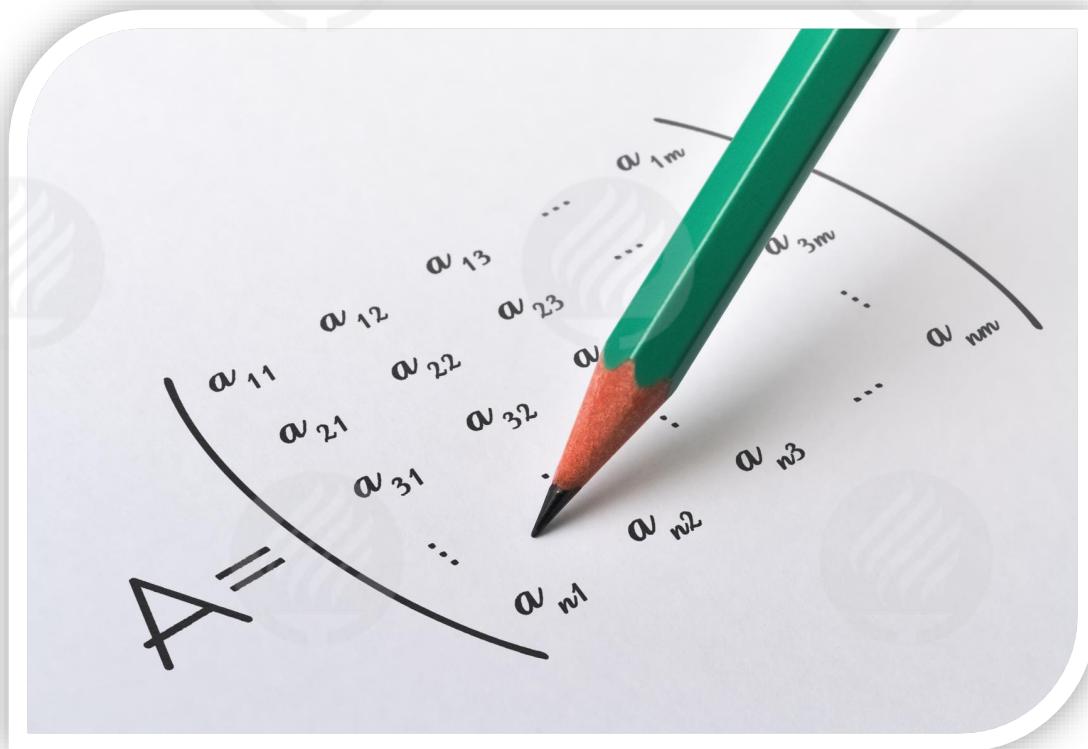
37	65	23
56	12	67
34	98	52

37	65	23
56	12	67
34	98	52

Podemos acceder a sus elementos por medio de:

- Sus dos subíndices (1 solo elemento)
- Rebanado de una fila (varios elementos)
- Iteración en una fila (varios elementos)
- Condicionando los valores de una fila (varios elementos)
- Condicionando los valores de toda la matriz (varios elementos)

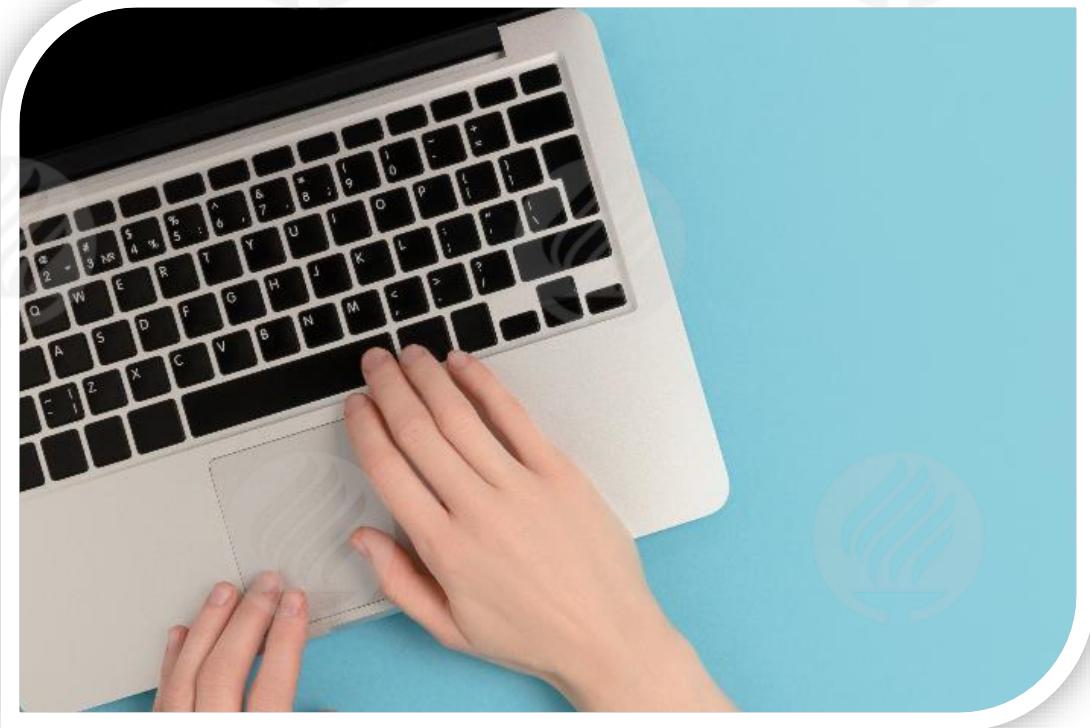
Matrices | Tema 2.2



Hay muchas formas de ejecutar operaciones con matrices:

1. Usar **funciones básicas** para todos los elementos de la matriz. Por ejemplo: la **suma de sus elementos, obtener el valor mínimo, el máximo**, entre otras
2. Aplicar las **funciones** únicamente sobre las **columnas**
3. Aplicar las **funciones** únicamente sobre las **filas**
4. Realizar **operaciones** con otra matrix. Por ejemplo: **suma, resta y multiplicación de matrices**

Actividad | Aprendizaje activo



Duración: 10 minutos

Crea una nueva libreta en Colab para:

Definir dos **matrices de 5x4** con elementos en serie y obtener:

- El **mínimo y máximo** de la **primera matriz**
- La **suma** de todas las columnas de la **primera matriz**
- La **suma** de todos los elementos de **la segunda y tercera fila de la primera matriz**
- La **suma** de los elementos de la **tercera y quinta fila de la primera matriz**
- La **suma** de las **dos matrices**

Duración **10 minutos**



- Las **matrices** son estructuras que permiten la manipulación de grandes cantidades de datos en formato de tabla. Muchas organizaciones almacenan su **información** dispuesta como **tablas en archivos o bases de datos**
- ¿Puedes ubicar dos escenarios donde tengas información que puedes vaciar a una **matriz**?
- De los escenarios anteriores, **¿cuáles son los campos numéricos que emplearías de la matriz para obtener información relevante?**

Cierre: Concepto clave | Tema 2

El trabajo con arreglos de una o más dimensiones en NumPy para almacenar y manipular información es fundamental por la claridad de la representación y la rapidez de su procesamiento.

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Plataformas / Frameworks

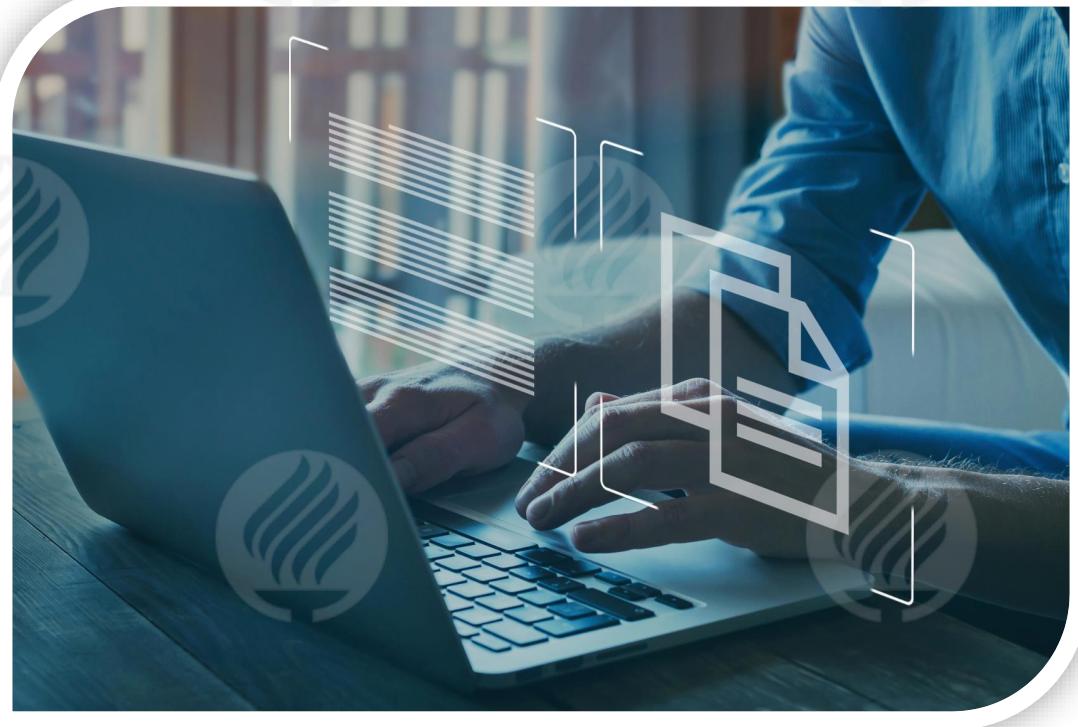
Tema 2: Arreglos y matrices con NumPy

Tema 3: Estructuras de datos en Pandas

Tema 4: Manipulando datos Tabulares con Pandas

Cierre de sesión

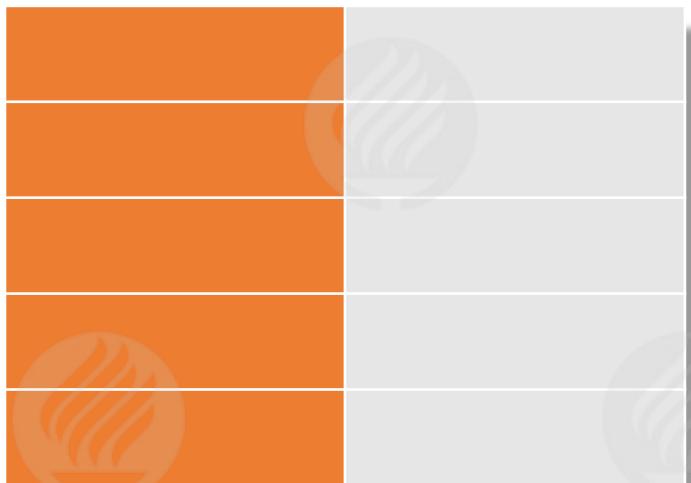
Propósito | Tema 3



- El auge en las **aplicaciones de ciencia de datos** requiere herramientas más intuitivas que las matrices para manipular estructuras en **dos dimensiones**
- No extender la funcionalidad de **NumPy** con este fin a través de la plataforma Pandas, limita la **expresividad y comprensión** de nuestros códigos
- Saber manipular las **estructuras de datos** de la plataforma Pandas nos permitirá **recuperar información** para procesarla rápidamente, de manera **segura, confiable y con muy poco esfuerzo**

Series y Dataframe | Tema 3.1

Serie



Etiquetas
o
subíndices

Elementos
o
valores

Las **Series** en Pandas son **arreglos de una dimensión** que pueden etiquetarse y permiten la manipulación de cualquier tipo de dato (numérico, texto u objetos de Python)

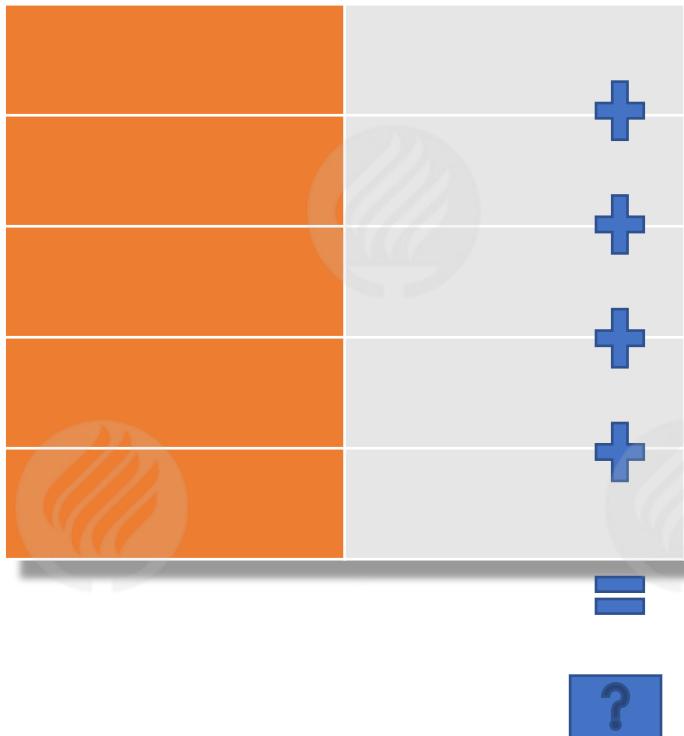
Podemos **crear una serie** con el constructor Series de Pandas, pasándole como primer parámetro:

- un **ndarray**
- un diccionario
- o un valor escalar

En el **segundo parámetro** se especifica, de manera opcional, **los índices de la serie**

Series y Dataframe | Tema 3.1

Calculando la sumatoria



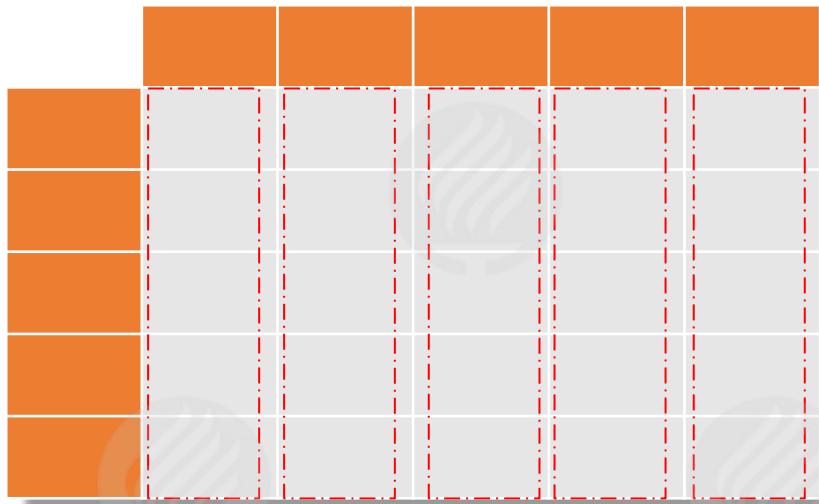
Acceso a una serie

- Si es etiquetada, se puede acceder a sus elementos a través de los **subíndices o de las etiquetas**
- Si **no** es etiquetada, el acceso a sus elementos sólo es posible **a través de los subíndices**

Operaciones

- Podemos aplicar **funciones de NumPy**
- Operaciones entre **series**

Dataframe



Series

El **dataframe** es la estructura **más importante** en Pandas. Tiene dos dimensiones de datos etiquetados. Las columnas pueden almacenar cualquier tipo de datos, Cada columna en un dataframe es una serie.

Podemos **crear un dataframe** a partir de:

- a. Un proceso de lectura de información
- b. Series
- c. Otro dataframe
- d. Un arreglo de dos dimensiones de NumPy
- e. Un arreglo estructurado o de registros ndarray
- f. Un Diccionario de ndarrays, listas, diccionarios o Series

Series y Dataframe | Tema 3.1

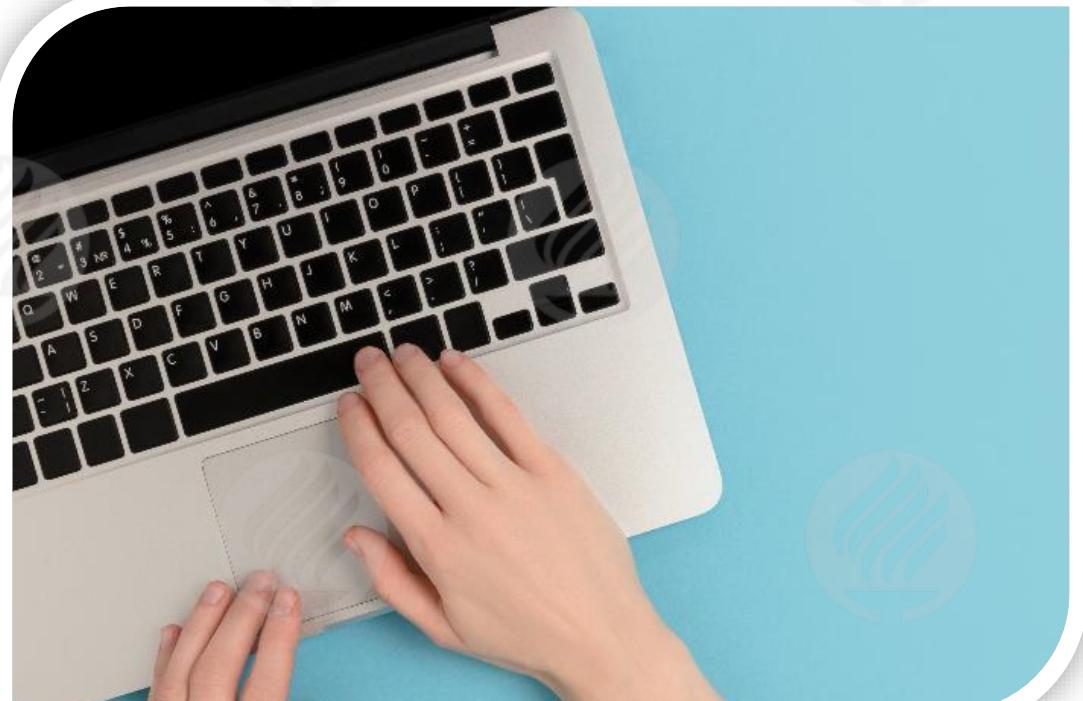


Eliminando una fila y columna

Entre las operaciones que pueden realizarse en un dataframe están:

- Agregar una columna
- Eliminar una columna
- Agregar una fila
- Eliminar una fila

Actividad | Aprendizaje activo



Crea una nueva libreta en Colab para:

1. Crear un **dataframe** que contenga **5 registros** de deportistas famosos con su nombre y el deporte que practican
2. Agrega una **columna** de edad con sus valores correspondientes
3. Agrega otra columna con nombre “**estado**”, donde todos los registros contengan el valor “**activo**”

Duración: **10 minutos**



- Las **series y los dataframes** de Pandas permiten **manipular información de manera sencilla e intuitiva**. Además, proporcionan muchos métodos para **procesar la información**
- ¿Comprendes las similitudes de las estructuras de **NumPy** y de **Pandas**? ¿Qué ventajas te aporta el uso de Pandas?
- ¿Puedes identificar **dos ejemplos** en los cuales ocuparías una **serie** y dos ejemplos en los que ocuparías un **dataframe**?

Cierre: Concepto clave | Tema 3

Las **series** y **dataframes** de Pandas son excelentes estructuras de datos de **alto rendimiento** para manipular **grandes volúmenes** de información con tipos de **datos diversos** y con **posibilidad de etiquetado**.

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Plataformas / Frameworks

Tema 2: Arreglos y matrices con NumPy

Tema 3: Estructuras de datos en Pandas

Tema 4: Manipulando datos Tabulares con Pandas

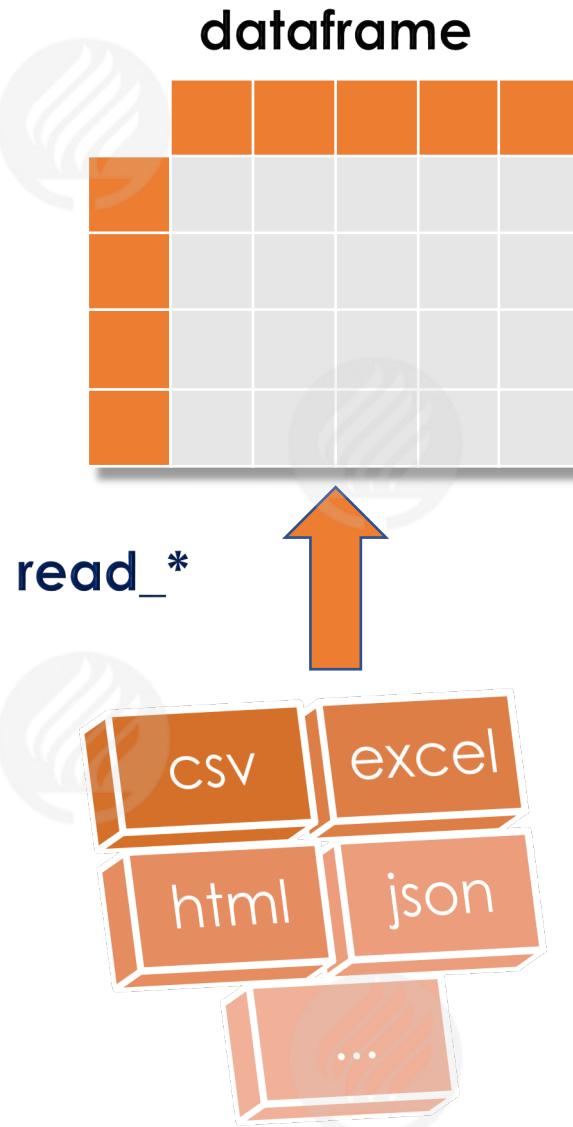
Cierre de sesión

Propósito | Tema 4



- Mucha de la **información almacenada** hoy en día tiene **estructura tabular y orígenes de datos diversos**
- De ahí la necesidad de conocer procesos de **lectura y escritura** para los scripts de análisis de dicha información
- El conocer cómo trabajar con **datos tabulares** es fundamental para manipular el **contenido de archivos y bases de datos**
- La similitud en la representación de un **dataframe** de Pandas con una hoja de cálculo o una tabla de SQL y su rapidez y eficiencia **contribuyen al éxito de nuestros desarrollos**

Leyendo datos tabulares con Pandas | Tema 4.1



- La lectura de **datos tabulares** es la forma más frecuente de **encontrar y obtener información**, independientemente de su origen (archivos o bases de datos) o formato
- Para esta tarea, la plataforma Pandas tiene muchas **ventajas**. Entre ellas:
 - a. Ágil **intercambio de datos** entre el almacenamiento permanente y memoria
 - b. La **indexación** integrada
 - c. El manejo de **datos faltantes**
 - d. Las operaciones sobre **conjuntos de datos**

Leyendo datos tabulares con Pandas | Tema 4.1



- Para la lectura se utilizan las funciones `read_*`
- Donde el `*` representa el formato del **origen de la información**
- Por ejemplo: si deseamos leer de un archivo separado por comas, utilizaremos `read_csv`
- El resultado de la lectura queda almacenado en un **dataframe**
- Es recomendable consultar la documentación oficial para conocer todos los **orígenes de datos**

Leyendo datos tabulares con Pandas | Tema 4.1

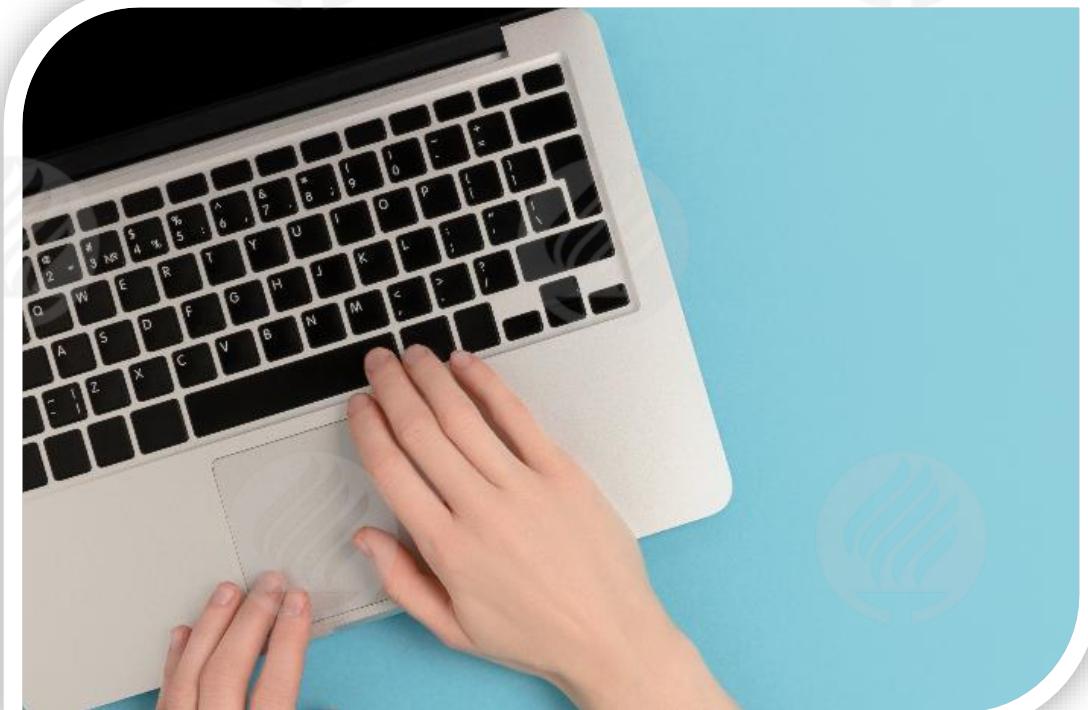
	id	name	height	mass	hair_color	skin_color
0	1	Luke Skywalker	172	77.0	blond	
1	2	C-3PO	167	75.0	NaN	
2	3	R2-D2	96	32.0	NaN	white,
3	4	Darth Vader	202	136.0	none	black
4	5	Leia Organa	150		personajesSW.dtypes	
5	6	Owen Lars	178		id	int64
6	7	Beru Whitesun lars	165		name	object
7	8	R5-D4	97		height	int64
					mass	float64
					hair_color	object
					skin_color	object
					eye_color	object
					birth_year	object
					gender	object
					homeworld	object
					species	object
					dtype:	object

Una vez que la información extraída se encuentra en el **dataframe**, se pueden ocupar algunas funciones o atributos útiles para **mostrar parte de la misma o su estructura**

Entre estos, se encuentran:

- **head()** – Para mostrar los **primeros registros**
- **tail()** – Para mostrar los **últimos registros**
- **info()** – Para obtener **información técnica** del dataframe
- El atributo **dtypes** – Para obtener el **tipo de dato de las columnas**

Actividad | Aprendizaje activo



Sube a tu drive el archivo cvs que tu profesor te proporcionará

Crea una nueva libreta en **Colab** para:

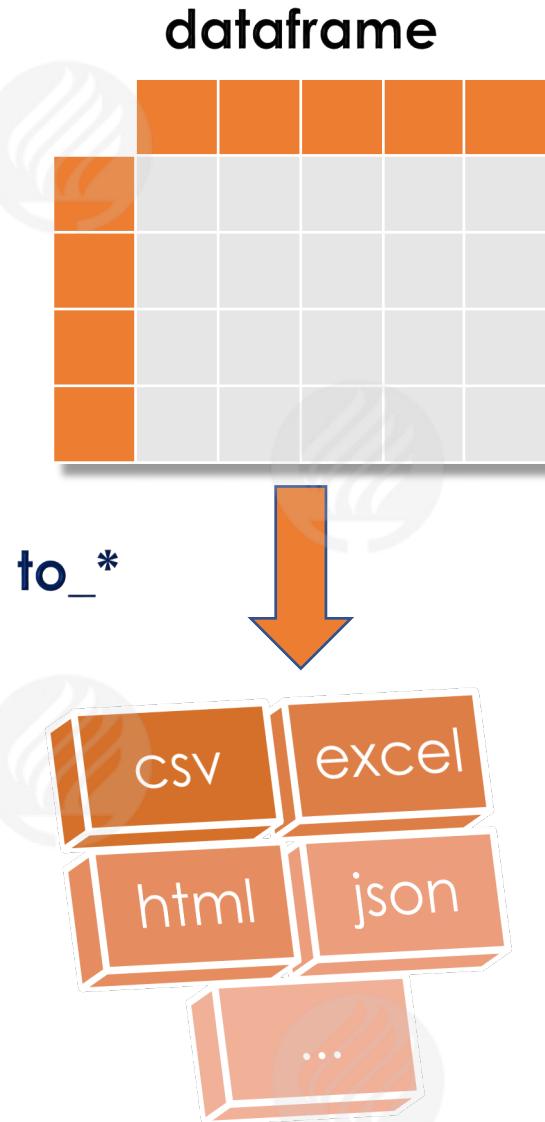
1. Leer dicho archivo
2. Mostrar los primeros **7 registros**
3. Mostrar los últimos **4 registros**
4. Mostrar la estructura del **dataframe** y comentar la cantidad de **columnas y el tipo de información** que contiene

Duración: 12 minutos



- Las funciones **read_*** nos permiten extraer datos tabulares de **diversos orígenes de datos, sin preocuparnos por el formato**
- En los programas que has realizado, ¿de qué tipos de **fuentes de datos** has tenido que extraer información?
- ¿Cuánto tiempo te ha tomado realizar los procedimientos de **extracción de información**?
- **¿Puedes mencionar archivos con información tabular que tengas la necesidad de leer?**

Escribiendo datos tabulares con Pandas | Tema 4.2

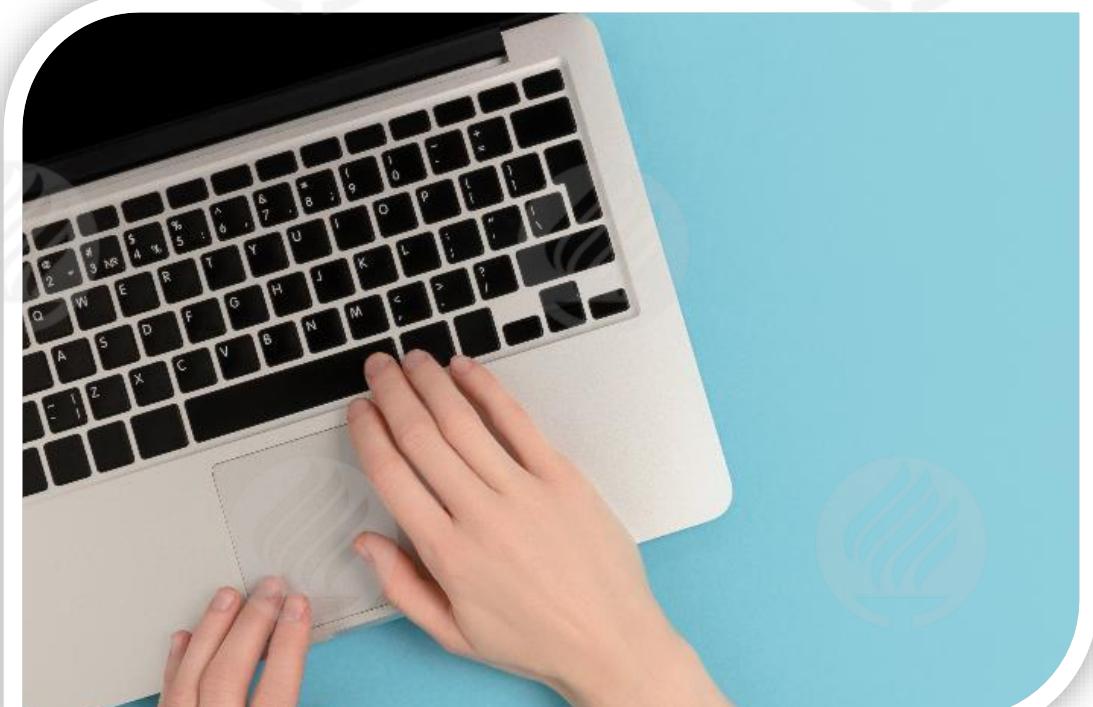


- Muchas veces necesitaremos **guardar en almacenamiento permanente**, las modificaciones que realicemos en memoria sobre un **dataframe**
- Para ello, Pandas provee mecanismos para **guardar el dataframe en diferentes formatos**



- Similar a las funciones **read_*** existen las funciones **to_***, donde el ***** denota el **formato que utilizaremos** para guardar la información

Actividad | Aprendizaje activo



Crea una nueva libreta en Colab para:

1. **Leer el archivo** que tu profesor te proporcionó en la actividad anterior
2. **Eliminar la mitad de los registros del dataframe** y las columnas necesarias para sólo mantener cuatro de ellas
3. Almacenar el nuevo **dataframe** en:
 - Un archivo separado por comas llamado “**someCharacters.csv**” y
 - Un archivo de excel llamado “**someCharacters.xlsx**”

Duración: **6 minutos**



- Las funciones `to_*`* nos permiten almacenar los **datos tabulares** contenidos en un dataframe en archivos con diferentes formatos, **sin preocuparnos por implementaciones particulares**
- En los programas que has realizado, ¿qué **tipos de archivos** has tenido que crear para **almacenar información**?
- **¿Cuánto tiempo te ha tomado realizar los procedimientos de almacenamiento de información?**

Cierre: Concepto clave | Tema 4

Pandas proporciona una amplia variedad de funciones para la lectura y escritura de datos tabulares en diversos formatos, lo que nos permite centrarnos en la manipulación y procesamiento de la información.

Panorámica de la sesión

1

Sesión Síncrona 1
Aprender

Tema 1: Plataformas / Frameworks

Tema 2: Arreglos y matrices con NumPy

Tema 3: Estructuras de datos en Pandas

Tema 4: Manipulando datos Tabulares con Pandas

Cierre de sesión

Cierre de la sesión

Cierre | Sesión Sincrónica 1 [Aprender]



- El utilizar las plataformas **NumPy** y **Pandas** y las estructuras de datos que éstas disponen es una gran opción para los procesos de **extracción, procesamiento y almacenamiento de la información**

Cierre | Sesión Sincrónica 1 [Aprender]



- ¿Identificas qué **estructura(s) de datos** utilizar dependiendo de la **información con la que cuentas**?
- ¿Puedes **leer y escribir** información tabular en distintos formatos?
- ¿Eres capaz de **realizar operaciones** con las estructuras estudiadas, incluso combinándolas?

Cierre: Concepto clave

Las plataformas **NumPy** y **Pandas** y sus estructuras favorecen el enfoque en la aplicación de ciencia de datos y nos alejan de la implementación subyacente de los métodos de lectura, selección, procesamiento y almacenamiento de la información.

Siguientes pasos | Semana de trabajo asíncrono [Profundizar]



1 Sesión Síncrona 1
Aprender

2 Trabajo asíncrono 1
Profundizar | Ruta de Aprendizaje

3 Sesión Síncrona 2
Preparar para Aplicar

4 Trabajo asíncrono 2
Aplicar en el trabajo | Reto

- Todos los conocimientos adquiridos en esta sesión te permitirán adentrarte en los siguientes conceptos de las plataformas **NumPy y Pandas** que se presentan en la sesión **Profundizar**
- En este momento eres capaz de utilizar las **plataformas NumPy y Pandas** como extensiones del lenguaje **Python**, sus estructuras de datos y las funciones más comunes para realizar desarrollos que **manipulen grandes volúmenes de información**



Tecnológico de Monterrey | 2021

Prohibida la reproducción total o parcial de esta
obra sin expresa autorización del Tecnológico
de Monterrey

Gracias | Programas LIVE