MODELOS Y SIMULACION

¡Bienvenidos a Modelos y Simulacion! Algunas cuestiones de logística:

- Las clases prácticas serán todos los Lunes de 16:00 a 17:30
- Las clases teóricas serán todos los Martes de 16:00 a 17:30

En esta materia contaremos a su vez con Office Hours / Horas de consulta. Podrán reservar espacios de tiempo a través de este link: https://calendly.com/tomas-nozica/15min .Podran venir a hablar sobre temas de clase o cualquier cosa que quieran preguntar. Las mismas me tendran a mi (Tomas) como host.

Cada semana al terminar la clase teórica les enviaremos práctica para ejercitar lo visto en clase.

Practica 0: Elementos de Python

En esta seccion práctica vamos a cubrir los siguientes temas:

- Manejo de listas, ciclos y flow control en Python
- Las librerias Panda, Numpy y Matplot: herramientas escenciales en ciencia de datos

En esta clase se seguirán los lineamientos de styling sugeridos por la comunidad: https://www.python.org/dev/peps/pep-0008

Parte 1: Recordando Python

In [ ]:
```python
array = [1, 2, 3, 4, 5]
```

In [ ]:
```python
#Ciclo for para recorrer un array
for i in array:
    print(i)
```

In [ ]:
```python
#Ciclo while para recorrer un array
i = 0
while i < len(array):
    print(array[i])
    i += 1
```

In [ ]:
```python
#Obtengamos los elementos pares de una lista [Para clase escribir esta fun de cero]
resul = []
for i in array:
    if i % 2 == 0:
        resul.append(i)
print(resul)
```

In [ ]:
```python
array = [[6, 4], [7, 1], [5, 10]]
```

In [ ]:
```python
#Sumemos uno a todos los pares para que se vuelvan impares [Para clase escribir esta fun de cero]
for r in range(0, len(array)):
    for c in range(0, len(array[r])):
        if array[r][c] % 2 == 0:
            array[r][c] += 1
print(array)
```

In [ ]:
```python
array = [['a', '3'], ['5', 'd'], ['c', '%']]
```

In [ ]:
```python
#Si es letra, hagamoslo mayus
for r in range(0, len(array)):
    for c in range(0, len(array[r])):
        if array[r][c].isalpha():
            array[r][c] = array[r][c].upper()
print(array)
```

Panda

Panda es una librería muy popular en el mundo de la ciencia de datos. Nos permite explorar, limpiar y procesar información. En Panda, llamamos DataFrame a las tablas que contienen la data. Para cualquier consulta dirigirse a la documentación: https://pandas.pydata.org/ docs/

In [ ]:
```python
import pandas as pd
source = {"Alumno": ["Abel", "Belen", "Carlos"],
         "Ingreso": [2016, 2018, 2016],
          "Aprobadas": [50, 35, 45]}
```

In [ ]:
```python
data_frame = pd.DataFrame(source)
data_frame
#.head() para ver las primeras 10
```

In [ ]:
```python
data_frame.describe()
```

In [ ]:
```python
aprobadas = data_frame["Aprobadas"]
aprobadas.describe()
```

In [ ]:
```python
nombre = data_frame["Alumno"]
nombre
```

In [ ]:
```python
aprobadas.max()
```

In [ ]:
```python
class_2016 = data_frame.loc[data_frame["Ingreso"] == 2016]
```

In [ ]:
```python
class_2016.max()
```

In [ ]:
```python
class_2016.min()
```

NumPy

NumPy es una libreria que contiene funciones matematicas de "alto nivel" para operar con matrices y vectores.

¿Que significa de alto nivel? Quiere decir que son faciles de entender para el ser humano.

Vamos a explorar algunas funciones...

Para acceder a toda la documentacion: https://numpy.org

In [ ]:
```python
import numpy as np
#Crear una matriz de 15 elementos, del 0 al 14, donde la matriz sea de 3x5
a = np.arange(15).reshape(3, 5)
a
#Si solo aplicaramos np.arrange(15) tendriamos un vector de 15 elementos
```

In [ ]:
```python
#Podemos preguntar que tamaño tiene X matriz
a.shape
```

In [ ]:
```python
#Para crear un array con determinados valores
array = np.array([10,21,23,22])
array
```

In [ ]:
```python
#Podemos convertir en matriz a un vector
matriz_cuadrada = array.reshape(2,2)
matriz_cuadrada
```

In [ ]:
```python
#Podemos convertir en vector a una matriz (tecnicamente se convierte en una matriz de una sola fila)
vector = matriz_cuadrada.reshape(1,-1)
vector
```

In [ ]:
```python
#Las operaciones entre matrices son muy simples con NumPy

A = np.array( [[1,1],[0,1]] )
B = np.array( [[2,0], [3,4]] )
#Producto de elemento por elemento
A * B
```

In [ ]:
```python
#Producto de matrices
A @ B
```

In [ ]:
```python
#Accediendo a los datos
vector = np.arange(5)
#El indice va desde 0 hasta n-1, como los arreglos que ya conociamos
print(vector[1])
#Para acceder a una porcion
print(vector[0:2])
print(vector[2:5])
#Para acceder al ultimo elemento si no conocemos el tamaño
print(vector[-1])
#Para acceder a una porcion desde el final, en este caso los dos ultimos
print(vector[-2:])
#Para acceder a una porcion desde el final, en este caso todos menos los dos ultimos
print(vector[:-2])
```

MATPLOT

Matplotlib es una biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy

In [ ]:
```python
import matplotlib.pyplot as plt
#Grafiquemos X contra Y
X = [1,2,3,4]
Y = [2,4,6,8]

plt.plot(X,Y)
```

In [ ]:
```python
#Podemos customizarlo un poco
plt.plot(X,Y,'r')
```

In [ ]:
```python
plt.plot(X,Y,'ro')
```

Aqui podras encontrar todas las opciones de formato https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html#matplotlib.pyplot.plotb

In [ ]:
```python
#Creemos la funcion X^3, generamos los valores X de 0 al 19, luego los Y seran X*X. Con numpy esta multiplicacion
#es muy sencilla y encima matplot la entiende!
X = np.arange(20)
Y = np.power(X, 3)
plt.plot(X,Y)
```

In [ ]:
```python
X = np.arange(20)
Y = X*X
W = X*2
#Ahora vamos a graficar X contra Y y X contra W en el mismo grafico
plt.plot(X,Y,W)
```

In [ ]:
```python
#Ahora podemos agregar una dispersion de puntos
plt.scatter(X,Y)
plt.plot(X,Y)
```