```
Celda de texto <Pk0TDpxtlKtR>
# %% [markdown]


#Big Data
# Introducción a Python



Celda de texto <jNIlov4r9YTe>
# %% [markdown]
## 1) Primer Paso

Este es un primer ejercicio de python



Celda de código <wkktko0ai8aR>
# %% [code]
a=1+1


Celda de texto <w0rXgsqTmids>
# %% [markdown]
que nos imprima la variable, sino print (a)

Celda de código <gaU9BjFo7coT>
# %% [code]
a

Celda de código <8W3fcovE35hb>
# %% [code]
print(a)

Celda de código <tfdKY-k7j60G>
# %% [code]
b="Big Data"

Celda de código <q0Szy-JQjVvX>
# %% [code]
b

Celda de texto <PxZVJ1u1mrWd>
```

```
# %% [markdown]
## Segundo paso!

Explorar un pandas data frame y sus tipos de datos

Celda de código <JXZ-ULORmtgE>
# %% [code]
# cargar librería Pandas

Celda de código <lzxvVMXCpOJQ>
# %% [code]
import pandas as pd

Celda de código <Nb30UPrqpn7Q>
# %% [code]
# crear data frame con 2 columnas (personaje y altura)
d_star_wars = pd.DataFrame({'personaje': ['Luke Skywaker', 'C-3PO',
'Darth Vader'], 'altura':[172, 167, 202]})

Celda de código <wa_yyKmFqIFD>
# %% [code]
d_star_wars

Celda de código <zStFRFcuqJYq>
# %% [code]
# Tipos de datos de pandas data frame, de que tipo son los datos. Que
busco.accion
d_star_wars.dtypes

Celda de texto <Hi93476qnYVU>
# %% [markdown]
otra manera de Crear data set

Celda de código <gho7z3eSquB->
# %% [code]
# Tipo de estructura de dato: la lista
lista_personajes=['Luke Skywaker', 'C-3PO', 'Darth Vader']

Celda de código <n0sSZsVa4OmE>
# %% [code]
lista_personajes

Celda de texto <ABdW2228ndFh>
```

```
# %% [markdown]
crear una variable

Celda de código <EAcxNOCkrepw>
# %% [code]
lista_altura=[172, 167, 202]

Celda de código <ENGuD1nYrjyo>
# %% [code]
lista_altura

Celda de texto <OSgiHZ6Gnhq7>
# %% [markdown]
Otra manera de Crear data frame

Celda de código <LDachckBrk7V>
# %% [code]
# Crear un pandas dataframe basado en 2 listas
dataframe_2l = pd.DataFrame({'personaje': lista_personajes,
                   'altura':lista_altura
                  })

Celda de código <pWgOR8wAsImK>
# %% [code]
dataframe_2l

Celda de texto <RcwFKkcCMrpz>
# %% [markdown]
## Tercer paso!

Filtrar datos en pandas. La lógica de la notación vectorial.

Celda de código <HRem86xYMtP->
# %% [code]
d_star_wars

Celda de código <cPA7uoqjOQwx>
# %% [code]
d_star_wars['personaje'] # devuelve un pandas series, que solo me
muestre la columna que le pido

Celda de texto <5snFxEovoHGB>
# %% [markdown]
```

```
Filtrar por algo específico: iloc

Celda de código <BWK3M7zROv4T>
# %% [code]
# d_star_wars.iloc[indice de fila, indice de columna] --> valor que
esperamos
#: para fila y 0 para columna
d_star_wars.iloc[:, 0] # los dos puntos simboliza los slides

Celda de código <akn758rfO6kk>
# %% [code]
# devuelve todo el registro 0
d_star_wars.iloc[0, :]

Celda de texto <E4adyxUSQiPz>
# %% [markdown]
Que columna quiero usar: loc

Celda de código <E0i-yeT1PjCj>
# %% [code]
d_star_wars.loc[:, 'personaje'] # es equivalente a: d_star_wars.iloc[:,
0]

Celda de código <RFkzb8BaQgAA>
# %% [code]
#d_star_wars.iloc[:, 'personaje'] # error porque el 2do parámetro no es
un entero

Celda de código <iljP9Os0RN5f>
# %% [code]
# índices para las columnas:
d_star_wars.columns

Celda de código <tsmC8avDRZ-6>
# %% [code]
# índices de los registros, para saber que filas tengo
list(d_star_wars.index)

Celda de código <g3VjF13aRfTj>
# %% [code]
d_star_wars.loc[0, :] #: trae todas las filas y columnas, es
equivalente a d_star_wars.iloc[0, :] (porque el índice 0 es la posición
y el nombre)
```

```
Celda de código <bW7eAqQGSNVp>
# %% [code]
# acceder a 1 elemento
d_star_wars.loc[0, 'altura']

Celda de código <TuOWZ5-OSh8n>
# %% [code]
d_star_wars.loc[1, 'altura']

Celda de texto <nyjrYjo7X48j>
# %% [markdown]
## Cuarto paso

Un poco mas de acceso y filtos de datos con condiciones
Rangos de indices

Celda de texto <Bf1zRCGpaZ1H>
# %% [markdown]
![](https://shanelynnwebsite-mid9n9g1q9y8tt.netdna-ssl.com/wp-content/u
ploads/2016/10/Pandas-selections-and-indexing-768x549.png)

Celda de código <HujXQU6BZk73>
# %% [code]
d_star_wars

Celda de código <WTPWCYN-anmZ>
# %% [code]
d_star_wars.iloc[0:2, 0] #[indice de fila, indice de columna] 0:2
significa que queremos la fila desde la posición 0 a la 2

Celda de código <093gE5aZbcVe>
# %% [code]
# en este caso, nos devuelve todo
d_star_wars.iloc[0:3, 0:2]

Celda de texto <qlqqZBwQd-v5>
# %% [markdown]
### Remplazar valores

Celda de código <ofZPOwGmbpTE>
# %% [code]
```

```
# otra manera de filtrar vectorialmente, me devuelve true en donde
busco el dato
d_star_wars['personaje']=='Darth Vader'

Celda de código <gTYNmYf8cK7j>
# %% [code]
# ¿Cómo le cambiamos la altura a Darth Vader?
v_flag = d_star_wars['personaje']=='Darth Vader'

Celda de código <TZuZ7hZFlXS7>
# %% [code]
# ¿Como remplazar un valor basado en una condición?
d_star_wars.loc[v_flag, 'altura'] #buscar en que fila esta para despues
cambiarlo

Celda de código <2z8oLlBXmXCz>
# %% [code]
d_star_wars.loc[v_flag, 'altura'] = 185

Celda de código <JZC8PwcRnrZW>
# %% [code]
# chequeo
d_star_wars

Celda de código <GD2YJFvUoDgS>
# %% [code]
# haciendo todo en la misma linea
d_star_wars.loc[ d_star_wars['personaje']=='Luke Skywaker',
'altura']=200

Celda de código <iJNnY-fWo_GW>
# %% [code]
d_star_wars

Celda de texto <PQJCNkjCiRNm>
# %% [markdown]
## 2) Dominando los datos

Carga de archivos, exploración de un data frame

Celda de código <FHZMAZzE8xsM>
# %% [code]
# URL donde se encuentra el archivo CSV
```

```
url =
'https://raw.githubusercontent.com/fpineyro/homework-0/master/starwars.
csv'

# Lee el CSV
d_star_wars = pd.read_csv(url)


Celda de código <3e_jGCtt9dme>
# %% [code]
# Head: Muestra las primeras 5 filas para ver si efectivamente cargó
bien lso datos
d_star_wars.head()

Celda de texto <EUMCc-1vtKCn>
# %% [markdown]
Cargamos los datos en una tabla

Celda de código <VHJBwGvFpwhh>
# %% [code]
# Exploración con DataTable colab
from google.colab.data_table import DataTable

DataTable(d_star_wars)

Celda de código <zCyLljbDkXku>
# %% [code]
# shape, nos muestra cuantas filas (87) y cuantas columnas tiene el
dataframe (10)
d_star_wars.shape

Celda de código <tG8xSuarkZJc>
# %% [code]
# head / tail: muestra los ultimos registros
d_star_wars.head(3)

Celda de código <DyDHnl40kG7W>
# %% [code]
# funpymodeling / instalar nuevas librerías
!pip install funpymodeling

Celda de código <Pz5-ihpH90ro>
# %% [code]
```

```
Celda de texto <eFME9bXa24Kl>
# %% [markdown]
#Importamos la librería de funpymodeling para explorar los datos


Celda de código <aPVGC1mlkNbh>
# %% [code]
# status
from funpymodeling.exploratory import status

Celda de código <pnxFZzuVkUsT>
# %% [code]
#Mostramos estadísticas del dataframe
status(d_star_wars)

Celda de código <Q8OfKcNnGF1R>
# %% [code]
d_star_wars=pd.read_csv(filepath_or_buffer="https://raw.githubusercontent.com/fpineyro/homework-0/master/starwars.csv", sep=',')
d_star_wars

Celda de texto <V9LlEu0BJTKa>
# %% [markdown]
## Nulos en pandas

Celda de código <azFUKEPlXtPe>
# %% [code]
d_star_wars.head()

Celda de código <9nKzETs7fAzI>
# %% [code]
# None ~ NaN

Celda de código <epuuNwZwjbFr>
# %% [code]
status(d_star_wars)

Celda de código <OofrhDxWJW6j>
# %% [code]
# isna: obtener nulos de una columna
d_star_wars['hair_color'].isna() #isna dice cuales son los nulos
```

```
# creamos un filtro para obtener los registros con valores nulos
d_star_wars[d_star_wars['hair_color'].isna()]

Celda de código <wuHRGrmyJW8l>
# %% [code]
# el origen de los NaN
import numpy as np

Celda de código <wUpYrMdbjb4a>
# %% [code]
a=pd.Series([1,2,3, np.NaN])

Celda de código <eUC2KEOijo1R>
# %% [code]
b=pd.Series([1,2,3, None])

Celda de código <nlPFUvbGhirh>
# %% [code]
a

Celda de código <6wzerJ6IhtJk>
# %% [code]
b

Celda de texto <a15cA4cqv9kF>
# %% [markdown]
### Remplazo de nulos

Variable categórica / object

Celda de código <Xa7Vs_7txajt>
# %% [code]
d_star_wars

Celda de código <J6fgUvzbh9dS>
# %% [code]
values={'homeworld': 'nulo_home', 'species': 'nulo_species'} #cuando
encuentra un valor nulo lo reemplaza por eso

d_star_wars.fillna(value=values)

Celda de código <pL_3yaNVzcFo>
```

```
# %% [code]
# remplazo en el mismo data frame
d_star_wars.fillna(value=values, inplace=True)

Celda de código <nKJEayaPvnQp>
# %% [code]
d_star_wars

Celda de texto <yRu6OxUD2yzx>
# %% [markdown]
## Imputación de nulos para variables numéricas



Celda de código <FfA5M1K4vnSt>
# %% [code]
# Obtención del promedio de la altura, y luego imputamos una columna
d_star_wars['height'].mean()

Celda de código <Nhc4JtzLAsCi>
# %% [code]
d_star_wars['height']=d_star_wars['height'].fillna(d_star_wars['height'
].mean())

d_star_wars

Celda de texto <riuitse2vp-F>
# %% [markdown]
---

## Ejercicios!


<br>

1) Crear dos listas de 5 valores, uno número y otro con string,
llamado lista1 y lista2, con los siguientes valores:

1,2,3,4,5

a,b,c,d,e
```

```
Celda de código <bVCiM4IW0GUB>
# %% [code]
#Ejercicio 1

lista1 =[1,2,3,4,5]
lista2 = ['a','b','c','d','e']

Celda de texto <HN2RIPRF0Gs4>
# %% [markdown]
2) Crear e imprimir en pantalla un pandas data frame llamado `d_test`
basandose en las listas del punto 1). Nombres de las columnas:
`columna_A` y `columna_B`

Celda de código <xnnT8EBXv17s>
# %% [code]
#Ejercicio 2
d_test = pd.DataFrame({'columna_A': lista1, 'columna_B': lista2})
d_test

Celda de texto <1trFt4Mp0Knl>
# %% [markdown]
3) Cargar los datos de heart disease con pandas en una variable llamada
`d_hd`. Imprimir en pantalla.

URL:
`https://raw.githubusercontent.com/fpineyro/homework-0/master/heart_dis
ease.csv`



Celda de código <N5eTxyFw0PE2>
# %% [code]
#Ejercicio 3
import pandas as pd

# Ejercicio 3
url =
'https://raw.githubusercontent.com/fpineyro/homework-0/master/heart_dis
ease.csv'
d_hd = pd.read_csv(url)

print(d_hd)
```

```
Celda de código <vk9DHfAr1hLe>
# %% [code]
#Ejercicio 3.1) Imprimir la variable `age` y `has_heart_disease` al
mismo tiempo.
d_hd[['age','has_heart_disease']]

Celda de código <QRwViZ2I3FpP>
# %% [code]
#Ejericio 3.2) Explorar los datos con DataTable de google colab, usar
filtros y ordenar las columnas haciendo click en ellas
# Activar DataTable en Colab
from google.colab import data_table
data_table.enable_dataframe_formatter()

# Mostrar todo el dataset en modo interactivo
d_hd

# Filtro el DT para mostrar gente mayor a 60
filtered_hd = d_hd[d_hd['age'] > 60]

filtered_hd
#display(filtered_hd)

Celda de código <zpDIuaM_1URJ>
# %% [code]
#Ejercicio 3.3) Imprimir los primeros 3 registros con head, y los
últomos 3 con tail

# head
d_hd.head(3)



Celda de código <_Yqaww6b5z71>
# %% [code]
#tail
d_hd.tail(3)

Celda de código <TPdrF7xeERO2>
# %% [code]
```

```
#Ejercicio 3.4) Acceder con loc a la columna has_heart_disease, y
mostrar todos los registros
d_hd.loc[:, 'has_heart_disease']

Celda de código <Uls4dpJiEgWw>
# %% [code]
#Ejercicio 3.5) Idem 3.4) peor mostrando los primeros 3 registros. Usar
slice, no head.
d_hd.loc[:, 'has_heart_disease'][:3]

Celda de código <yfVqQU1sFBix>
# %% [code]
#Ejercicio 3.6) Usar iloc para obtener las primeras 3 columnas, y 5
registros
d_hd.iloc[0:5, 0:3]

Celda de texto <NFkwzOcw5_1G>
# %% [markdown]
4) Trabajando con nulos


Celda de código <BAiXp62yDtP6>
# %% [code]
# 4.1) Instalar funpymodeling
!pip install funpymodeling

Celda de código <1_ZCEks96CUO>
# %% [code]
# 4.2) Cargar función status
from funpymodeling.exploratory import status

Celda de código <i5PAmADVD4PT>
# %% [code]
#Ejercicio 4.3) ¿Qué variables tienen nulos?  (usar status)
status(d_hd)


Celda de código <V2NRaF1cD4k9>
# %% [code]
#Ejercicio 4.4) Devolver un data frame con todos los registros que
tienen NaN en la variable "thal".
thal_nulls = d_hd[d_hd['thal'].isnull()]
thal_nulls
```

```
Celda de código <Ih_J_WmdD4uj>
# %% [code]
#Ejercicio 4.5) Remplazar los nulos que aparecen en la variable "thal
por 999. *No* guardarlo en el dataframe original. Usar fillna.
thal_filled = d_hd['thal'].fillna(999)
thal_filled

Celda de código <fh4BgTd0Ifmp>
# %% [code]
#Ejercicio 4.6) Remplazar los nulos que aparecen en la variable "thal"
por 999 (como antes),
# y también remplazar los nulos de "num_vessels_flour" por el promedio
de la variable.
# Guardar el resultado en un dataframe nuevo llamado "d_hd2".


d_hd2 = d_hd.copy()


d_hd2['thal'] = d_hd2['thal'].fillna(999)


mean_value = d_hd2['num_vessels_flour'].mean()
d_hd2['num_vessels_flour'] =
d_hd2['num_vessels_flour'].fillna(mean_value)


Celda de código <YMTb6CaYD4zo>
# %% [code]
#Ejercicio 4.7) Chequear resultado con status
status(d_hd2)

Celda de código <Vr5lU2sWDriL>
# %% [code]
#Ejercicio 4.8)
# Eliminar todas los registros con nulos usando `dropna` de pandas
d_hd_dropna = d_hd.dropna()
d_hd_dropna


Celda de código <cce57772>
# %% [code]
```

```
# Load the data first
url =
'https://raw.githubusercontent.com/fpineyro/homework-0/master/heart_dis
ease.csv'
d_hd = pd.read_csv(url)
```