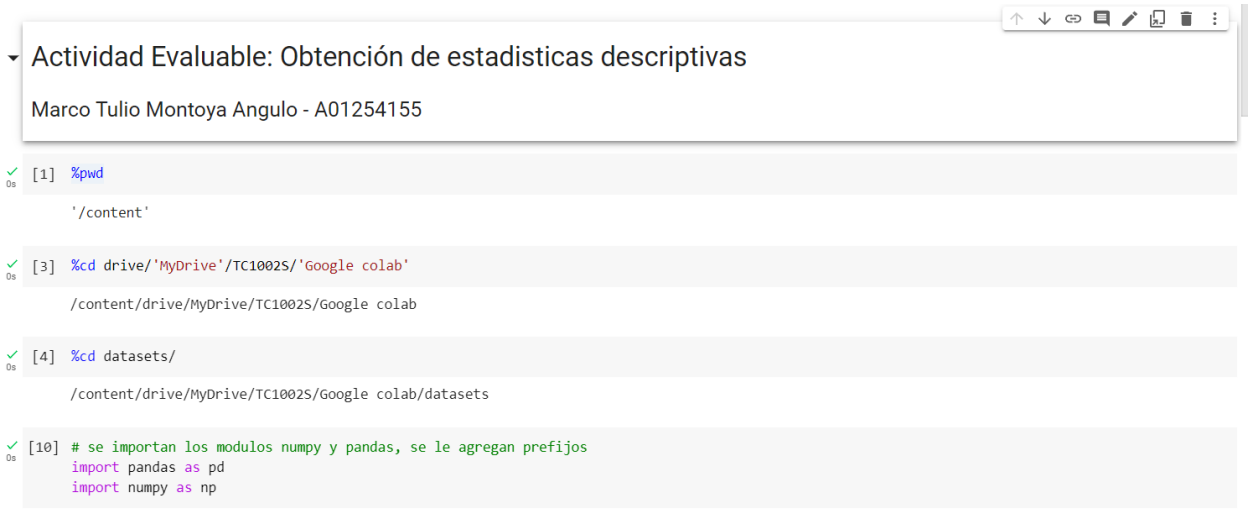
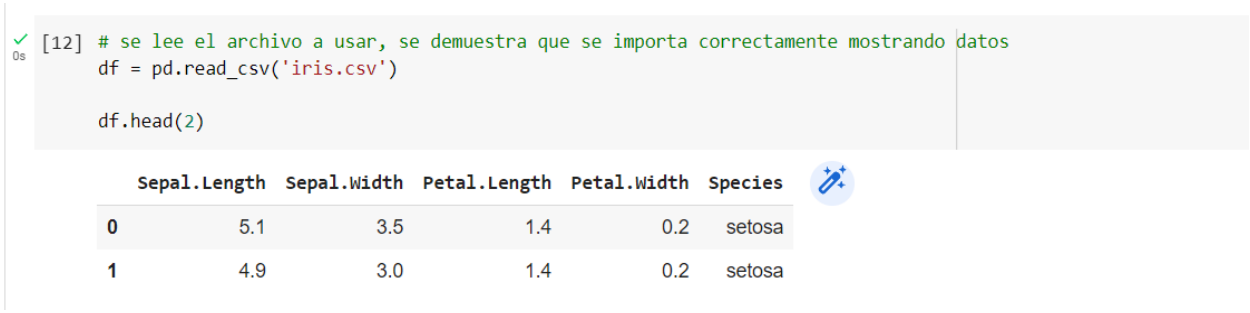
Actividad Evaluable: Obtención de estadísticas descriptivas

Antes de empezar con la actividad, se necesita preparar el entorno de trabajo, en este caso, usando Google Colab se crea un notebook, para después vincular la carpeta donde se encuentra el archivo a trabajar, iris.csv.



```
Actividad Evaluable: Obtención de estadisticas descriptivas

Marco Tulio Montoya Angulo - A01254155

[1] %pwd

'/content'

[3] %cd drive/'MyDrive'/TC1002S/'Google colab'

/content/drive/MyDrive/TC1002S/Google colab

[4] %cd datasets/

/content/drive/MyDrive/TC1002S/Google colab/datasets

[10] # se importan los modulos numpy y pandas, se le agregan prefijos
     import pandas as pd
     import numpy as np
```

1. **Carga los datos usando tu lector de csv o con pandas. Es recomendable hacerlo con pandas.**



```
[12] # se lee el archivo a usar, se demuestra que se importa correctamente mostrando datos
     df = pd.read_csv('iris.csv')

     df.head(2)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa

En este caso se utiliza el comando pd.read, de la librería pandas, y se almacena en la variable df (dataframe).

Para comprobar que los archivos se importaron correctamente, se muestran datos de la tabla con el comando df.head.

2. **Verifica la cantidad de datos que tienes, las variables que contiene cada vector de datos e identifica el tipo de variables.**

Hay varias maneras de comprobar la cantidad de datos que se tienen, en este caso, queremos ver la cantidad de columnas y filas, para eso se utiliza la propiedad shape (df.shape), esto muestra el siguiente resultado:

```
[13] # se define el numero de columnas y renglones del archivo
     df.shape
```

```
(150, 5)
```

Este output muestra que, nuestro dataframe tiene un total de 150 filas y 5 columnas, o sea, imaginando que los datos siempre estén llenos, 750 datos en total. Esto se puede comprobar con la propiedad size:

```
[27] print(df.size)
```

```
750
```

El siguiente paso es mostrar los variables que contiene cada vector y el tipo de variables que acepta, para esto primero usamos la función head, que es una muestra una parte de la tabla.

```
[28] df.head()
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Aquí se ve un poco la estructura de la tabla, para ver el tipo de variables que acepta cada una, se utiliza el método info, este método nos muestra lo siguiente:

```
[15] # se ven las variables y tipos de datos que estas aceptan
     df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Sepal.Length  150 non-null    float64
 1   Sepal.Width   150 non-null    float64
 2   Petal.Length  150 non-null    float64
 3   Petal.Width   150 non-null    float64
 4   Species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

```
[16] # se muestra el indice de columnas
     df.columns
```

```
Index(['Sepal.Length', 'Sepal.Width', 'Petal.Length', 'Petal.Width',
       'Species'],
      dtype='object')
```

Este método nos muestra las variables, en este caso, son lo siguiente:

- Sepal.Length: todos los datos son datos no nulos, es tipo float (float64).
- Sepal.Width: todos los datos son datos no nulos, es tipo float (float64).
- Petal.Length: todos los datos son datos no nulos, es tipo float (float64).
- Petal.Length: todos los datos son datos no nulos, es tipo float (float64).
- Species: todos los datos son datos no nulos, es tipo objeto (object).

También se utiliza el método columns, que muestra los títulos de la tabla.

3. **Analiza las variables para saber que representa cada una y en que rangos se encuentran. Si la descripción del problema no te lo indica, utiliza el máximo y el mínimo para encontrarlo.**

En este caso, primero se utiiza el metedodo describe, que nos da una vista detallada de las variables de la tabla:

```
df.describe()
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Con esta tabla podemos ver que, los datos están llenos, la media, la desviación estándar, el valor mínimo y máximo de cada variable.

Las variables estadísticas que destacan son las siguientes:

```
[20] print('=== DATOS - MEDIA ===')
     print('Media Sepal.Lenght: ', df['Sepal.Length'].mean())
     print('Media Sepal.Width: ', df['Sepal.Width'].mean())
     print('Media Petal.Length: ', df['Petal.Length'].mean())
     print('Media Petal.Width: ', df['Petal.Width'].mean())
```

```
=== DATOS - MEDIA ===
Media Sepal.Lenght:  5.843333333333334
Media Sepal.Width:  3.0573333333333337
Media Petal.Length:  3.7580000000000005
Media Petal.Width:  1.1993333333333336
```

```
=== DATOS - MEDIANA ===
Mediana Sepal.Lenght:  5.8
Mediana Sepal.Width:  3.0
Mediana Petal.Length:  4.35
Mediana Petal.Width:  1.3

=== DATOS - DESVIACIÓN ESTÁNDAR ===
Desviación estándar Sepal.Lenght:  0.828066127977863
Desviación estándar Sepal.Width:  0.4358662849366982
Desviación estándar Petal.Length:  1.7652982332594662
Desviación estándar Petal.Width:  0.7622376689603465
```

4. **Basándose en la media, mediana y desviación estándar de cada variable, que conclusiones puedes entregar de los datos.**

   Con los datos de media, mediana y desviación estándar de cada variable, podemos ver:

   - **Media:** el promedio de los datos en cada serie, dividiendo la suma de ellos entre su cantidad.
   - **Mediana:** El numero intermedio de la serie de datos, se descartan los datos de los lados hasta encontrar el dato central.
   - **Desviación estándar:** La dispersión del conjunto de datos.