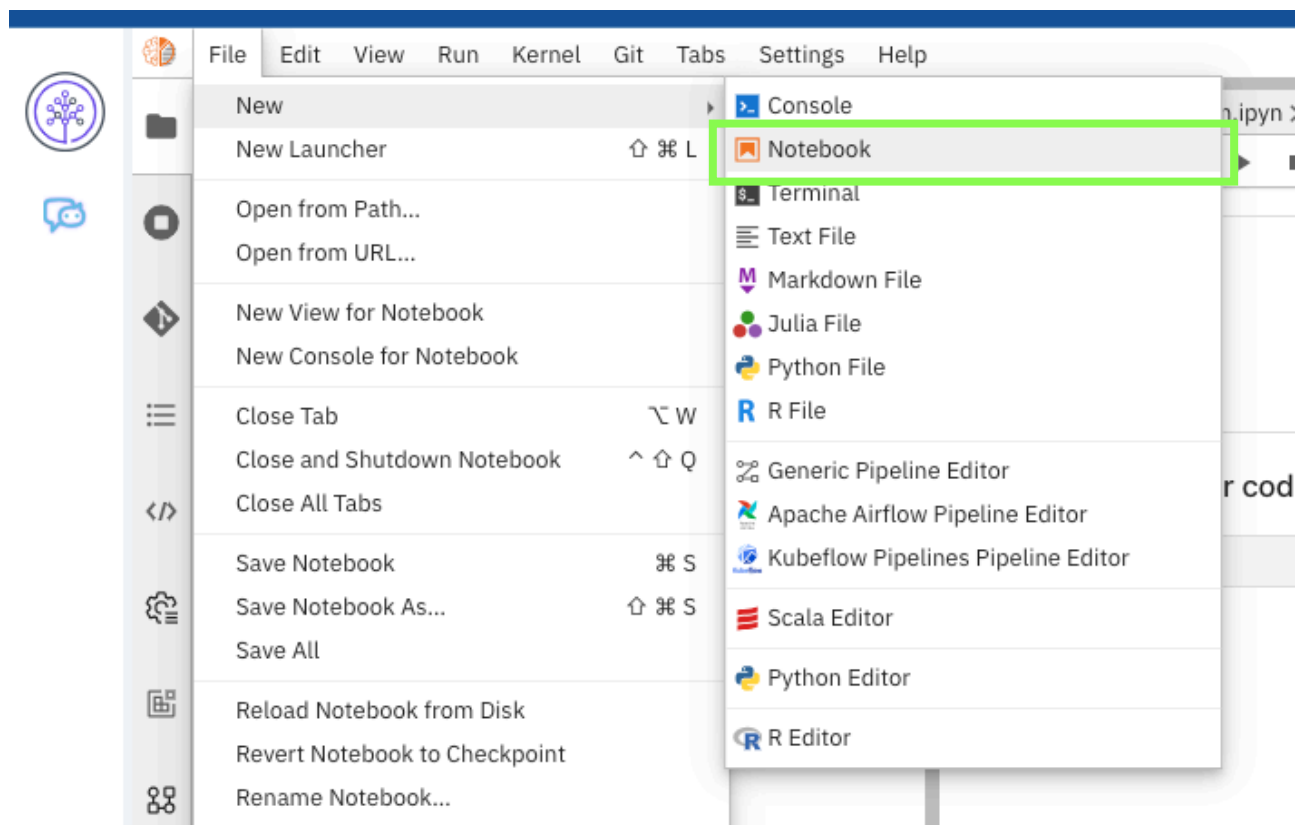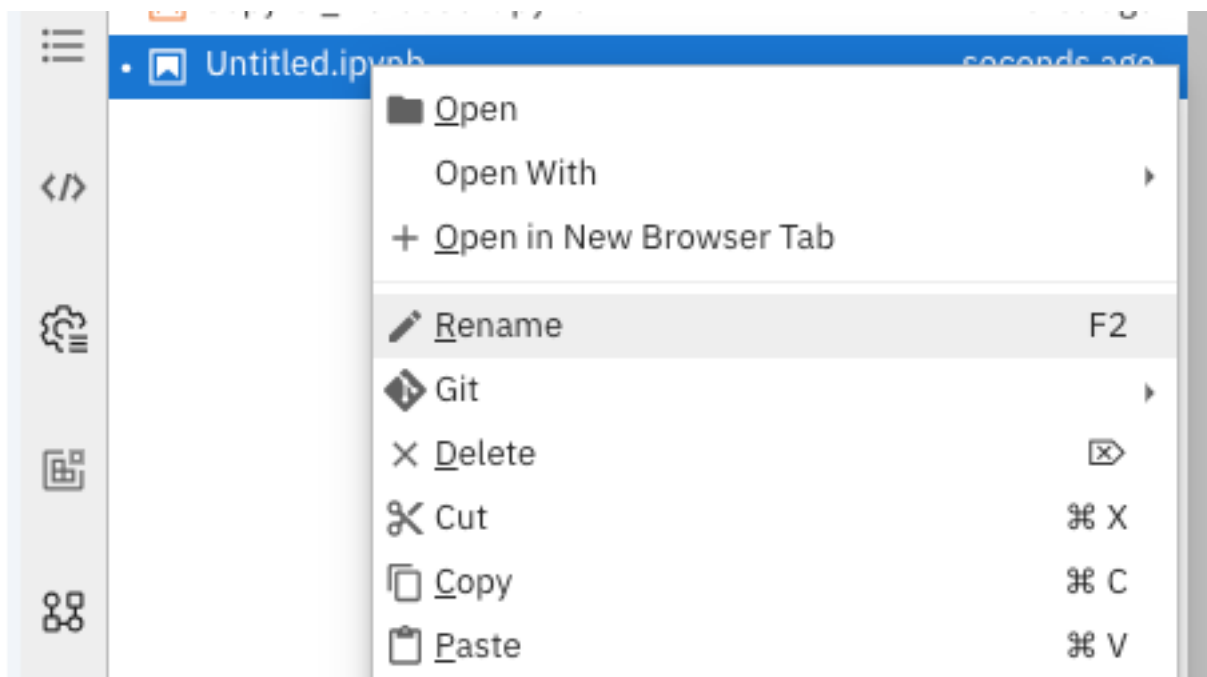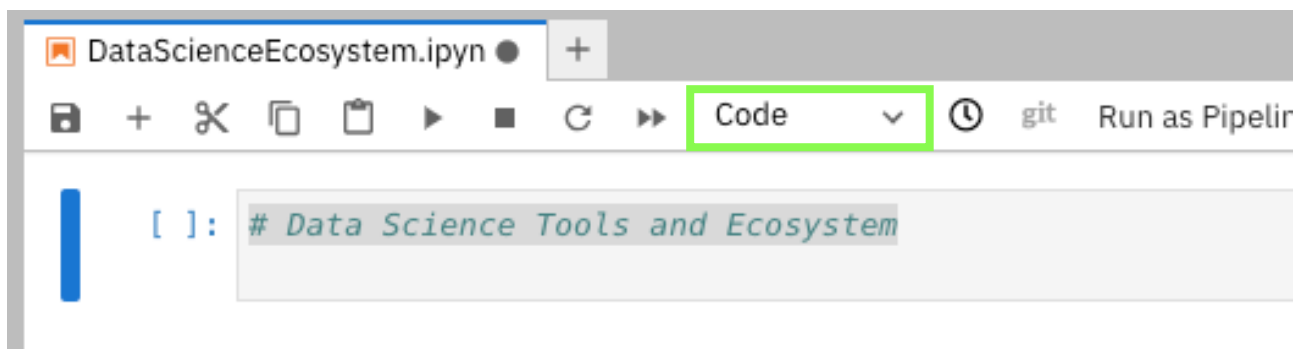# Exercise 1: Create a Jupyter notebook

Then we can change the name by clicking on the created file with the right mouse button.

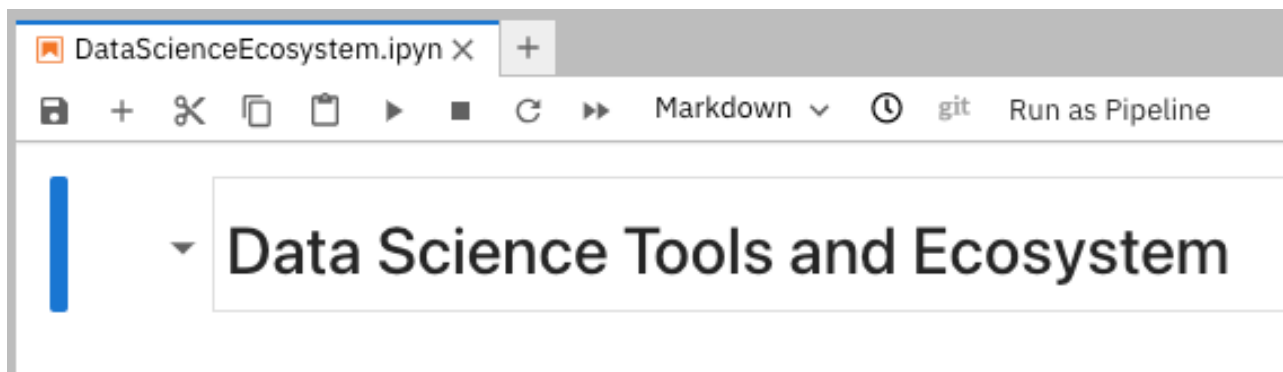# Exercise 2: Create a Markdown cell with a notebook title

Create a Markdown cell with the title Data Science Tools and Ecosystem using the H1 style header.

We select the Markdown option in the tab and then inside the code we press Ctrl + Enter (Win) or Cmd + Enter (Mac), it will become a Markdown cell
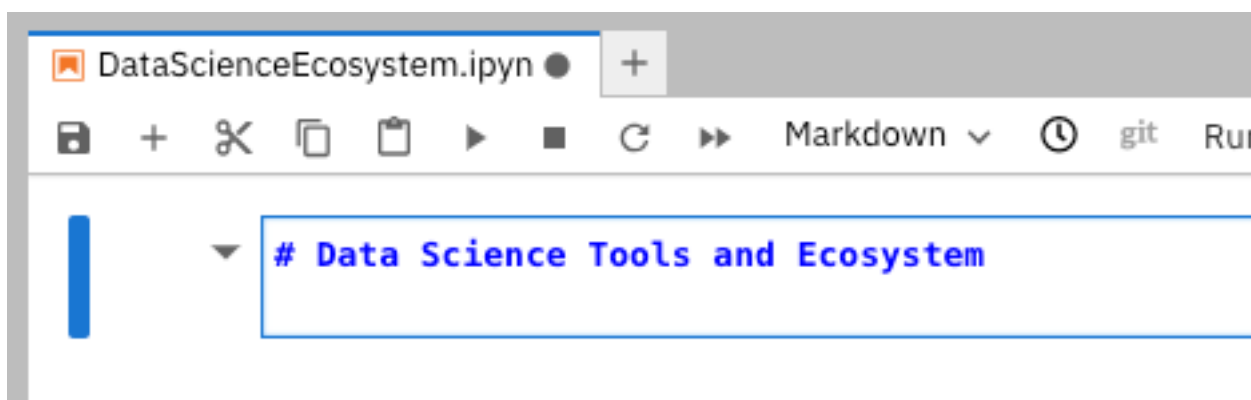
Resultado

Exercise 3
Create a Markdown cell for an introduction

Write an introductory sentence about the notebook like the following:
In this notebook, Data Science Tools and Ecosystem are summarized.

We select the Markdown option in the tab and then inside the code we press Ctrl + Enter (Win) or Cmd + Enter (Mac), it will become a Markdown cell

Then press shift + Enter to create a new cell

Exercise 4: Create a Markdown cell to list data science languages

Start the cell with a general description line like:
Some of the popular languages that Data Scientists use are: Then, create an ordered (numbered) list of 3 (or more) commonly used languages for data science.

With the following code, we will create a numerical list of popular languages in Data Science.

We select the Markdown option in the tab and then inside the code we press Ctrl + Enter (Win) or Cmd + Enter (Mac), it will become a Markdown cell
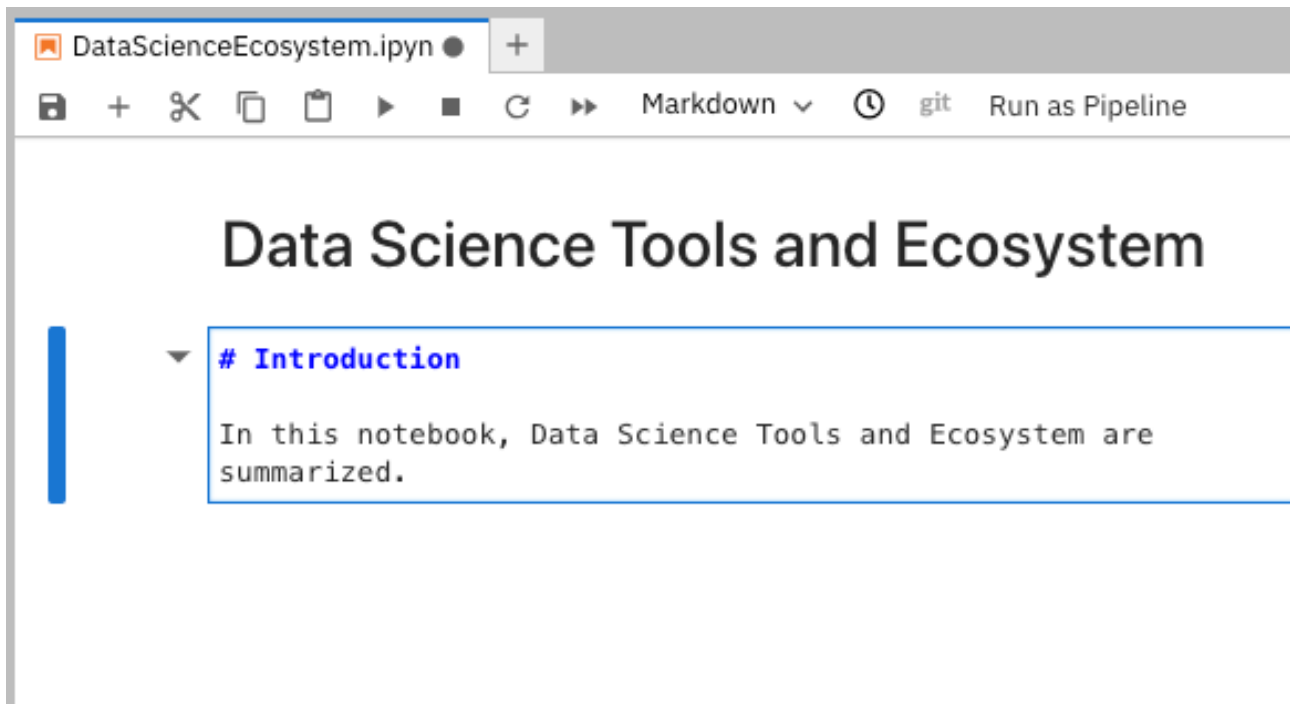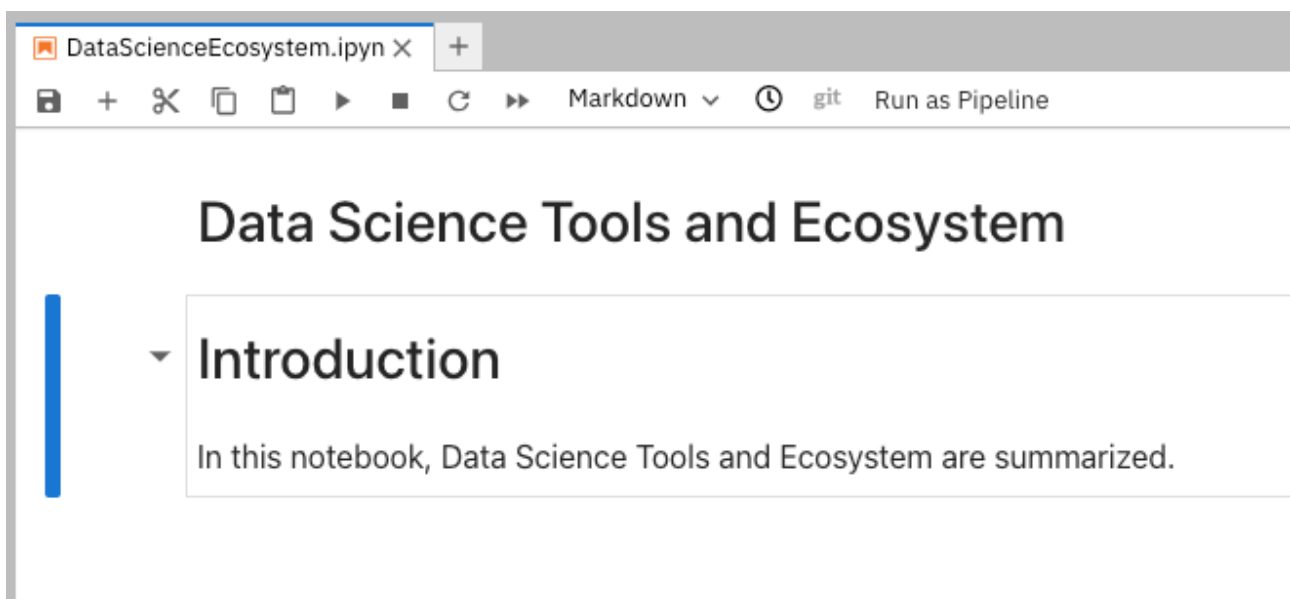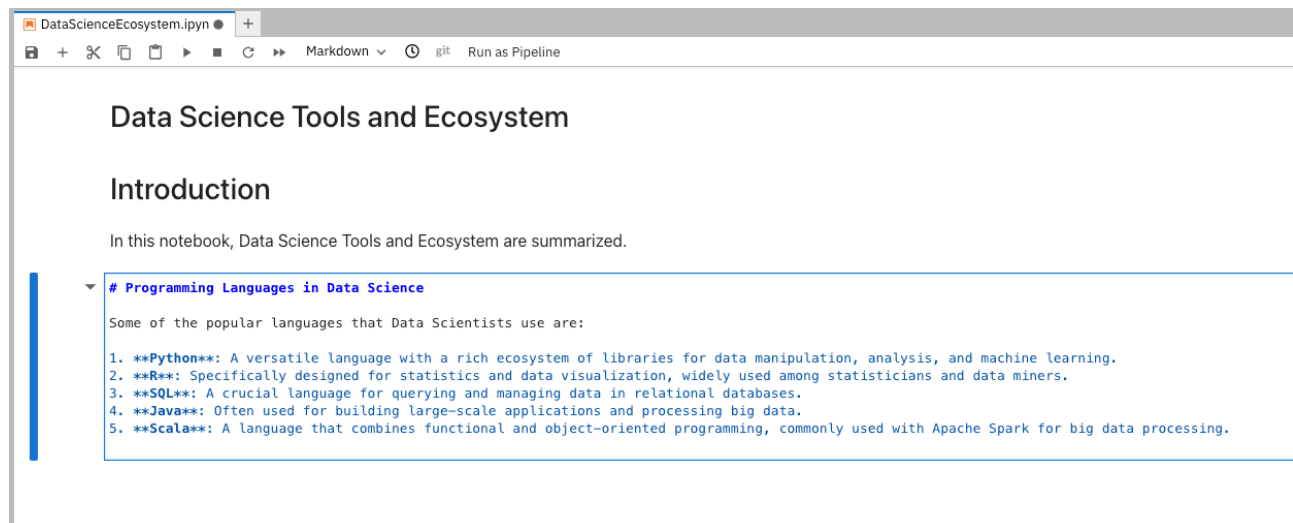

Then press shift + Enter to create a new cell

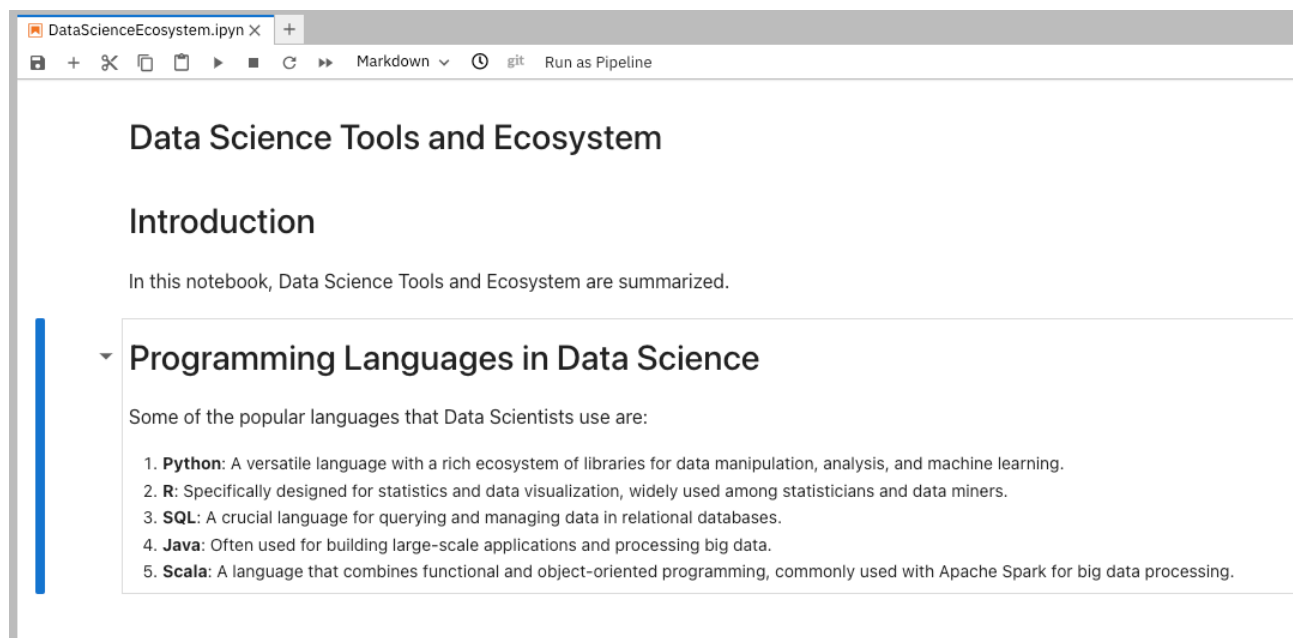# Exercise 5: Create a Markdown cell to list data science libraries

Add an overview line to the cell like:
Some of the commonly used libraries used by Data Scientists
include:
Below this line, add an ordered list of 3 (or more) commonly used libraries in data science.

```
[ ]: # Data Science Libraries

     Some of the commonly used libraries by Data Scientists include:

     1. **Pandas**: A powerful data manipulation and analysis library for Python, ideal for handling structured data.
     2. **NumPy**: A fundamental package for numerical computing in Python, providing support for arrays and a wide range of mathematical functions.
     3. **Matplotlib**: A plotting library for Python that enables the creation of static, interactive, and animated visualizations in Python.
     4. **Scikit-learn**: A machine learning library for Python that provides simple and efficient tools for data mining and data analysis.
     5. **TensorFlow**: An open-source library developed by Google for machine learning and deep learning applications.
```

We select the Markdown option in the tab and then inside the code we press Ctrl + Enter (Win) or Cmd + Enter (Mac), it will become a Markdown cell

Data Science Libraries

Some of the commonly used libraries by Data Scientists include:

1. **Pandas**: A powerful data manipulation and analysis library for Python, ideal for handling structured data.
2. **NumPy**: A fundamental package for numerical computing in Python, providing support for arrays and a wide range of mathematical functions.
3. **Matplotlib**: A plotting library for Python that enables the creation of static, interactive, and animated visualizations in Python.
4. **Scikit-learn**: A machine learning library for Python that provides simple and efficient tools for data mining and data analysis.
5. **TensorFlow**: An open-source library developed by Google for machine learning and deep learning applications.

Then press shift + Enter to create a new cell

Exercise 6: Create a Markdown cell with a table of data science tools

Create a single-column table in this cell, with the first row containing the header Data Science Tools. The next three rows of the table should list three open source development environment tools used in data science.

We select the Markdown option in the tab and then inside the code we press Ctrl + Enter (Win) or Cmd + Enter (Mac), it will become a Markdown cel

Then press shift + Enter to create a new cell

Exercise 7: Create a Markdown cell that presents examples of arithmetic expressions

Add a line in this cell with an H3-style header with text like:
Below are a few examples of evaluating arithmetic expressions in Python

Below are a few examples of evaluating arithmetic expressions in Python in Jupyter

1. Addition: `10 + 2` results in `12`
2. Subtraction: `10 - 2` results in 8
3. Multiplication: `10 * 2` results in `20`
4. Division: `10 / 2` results in `5.0`
5. Exponentiation: `10 ** 2` results in `100`

```python
[11]:  # Define numbers
       a = 10
       b = 2

       # Addition
       result = a + b

       # Print result
       print("The addcition ", a, "and", b, "is:", result)

       # Subtraction
       result = a - b

       # Print result
       print("The Subtraction", a, "and", b, "is:", result)

       # Multiplication
       result = a * b

       # Print result
       print("The Multiplication", a, "and", b, "is:", result)

       # Division
       result = a / b

       # Print result
       print("The Division", a, "and", b, "is:", result)

       # Exponentiation
       result = a ** b

       # Print result
       print("The Exponentiation", a, "and", b, "is:", result)
```

```
The addcition  10 and 2 is: 12
The Subtraction 10 and 2 is: 8
The Multiplication 10 and 2 is: 20
The Division 10 and 2 is: 5.0
The Exponentiation 10 and 2 is: 100
```

Note: The Python code we have to select code

Then press shift + Enter to create a new cell

Exercise 8 - Create a code cell to multiply and add numbers

In this code cell, evaluate the expression (3*4)+5.
Insert a comment line before the expression to explain the operation, e.g. This a simple arithmetic expression to mutiply then add integers. . Then, run the cell to ensure that the expression returns the expected result of 17.

This is a simple arithmetic expression to multiply then add integers

result = (3 * 4) + 5

Print result

print("The result he result of the expression (3 * 4) + 5 is:", result)

```
[15]: # This is a simple arithmetic expression to multiply then add integers
      result = (3 * 4) + 5

      # Print result
      print("The result he result of the expression (3 * 4) + 5 is:", result)

      The result he result of the expression (3 * 4) + 5 is: 17
```

Exercise 9 - Create a code cell to convert minutes to hours

In this code cell, type an expression that converts 200 minutes to hours. Insert a comment line before the expression to explain the operation, for example
This will convert 200 minutes to hours by diving by 60. Run the cell to evaluate the expression.

Print the result

```
print("200 minutes are equivalent to:", hours, "hours")
```

```
[19]:  # This will convert 200 minutes to hours by dividing by 60
       minutes = 200
       hours = minutes / 60

       # Print the result
       print("200 minutes are equivalent to:", hours, "hours")

       200 minutes are equivalent to: 3.3333333333333335 hours
```
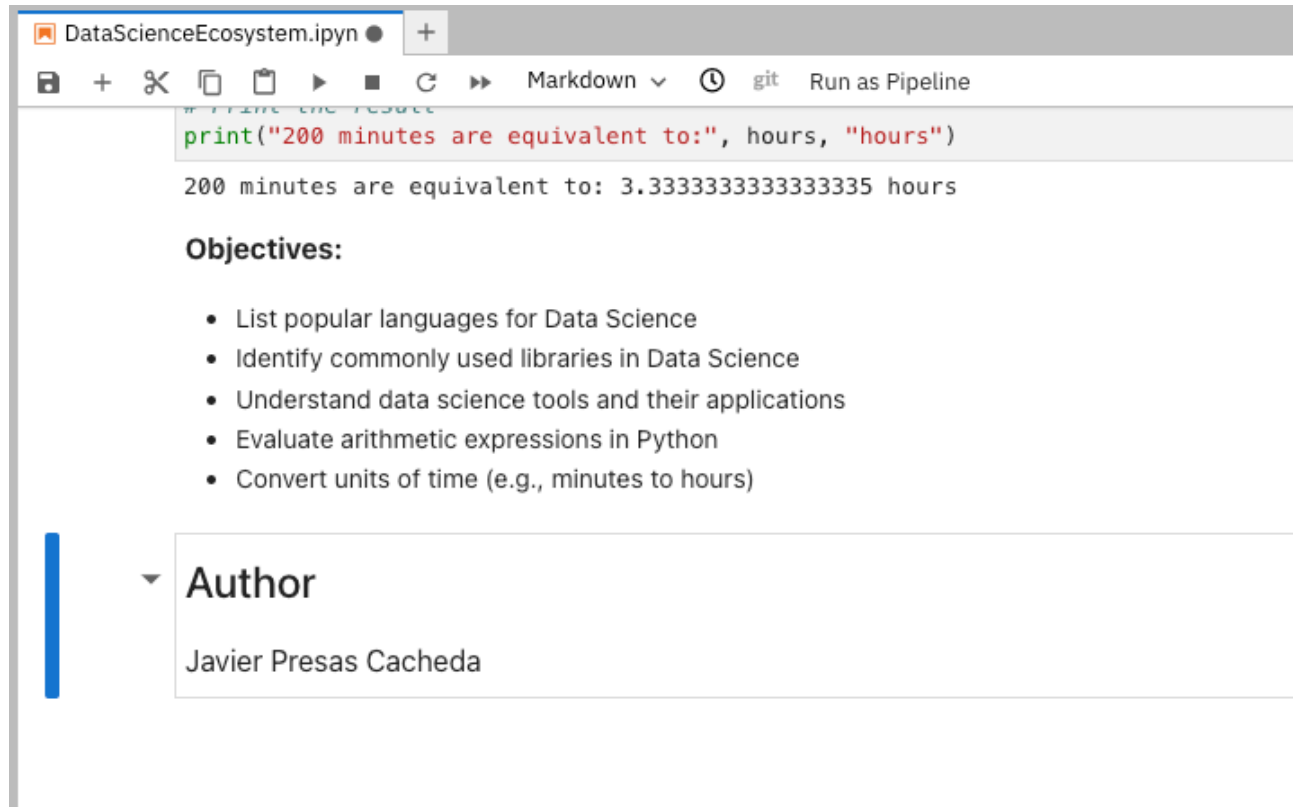
Exercise 10: Insert a Markdown cell to list objectives

Below the introduction cell created in Exercise 3, insert a new Markdown cell to list the objectives that this notebook covered (i.e., some of the key takeaways from the course). In this new cell, start with an introductory line titled: Objectives: in bold. Then, using an unordered list, list 3-5 items covered in this notebook, such as List popular languages for Data Science.
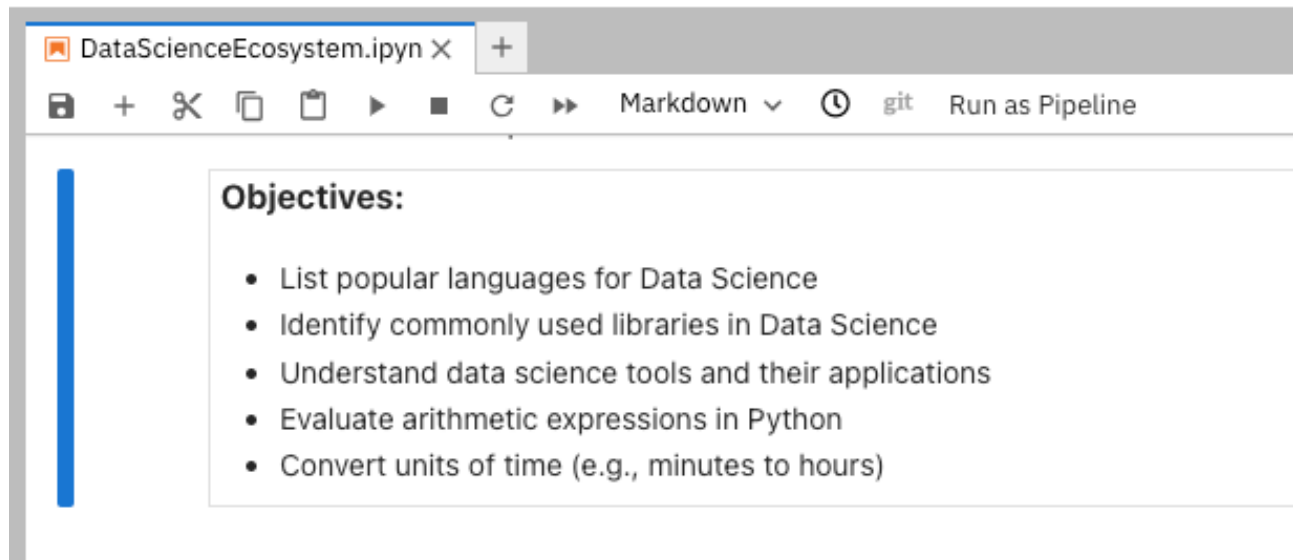
Exercise 11 - Create a Markdown cell to indicate the author's name

In this Markdown cell, include the following text Author in H2 style heading format. Include your normal nametext below the word Author