

Data Analysis and Visualization

CentraleDigitalLab@Nice

Deborah Dore - ddore@i3s.unice.fr

MARIANNE, I3S, CNRS, INRIA, Université Côte d'Azur

PART 1

Lesson 1: Introduction to Python

Why Python?

- Simple and easy to use
- Flexible
- Popular and widely used
- Extensive libraries for data manipulation and visualization
- Large community and resources for support

Lesson 1: Introduction to Python

Python Basics

- Variables and data types
- Conditional statements (if-else)
- Loops (while/for)
- Writing and running scripts

Lesson 1: Introduction to Python

Python Basics

Variables and data types


```
✓ 0 s ▶ 1 # Define variables
2 x = 10 # Integer
3 pi = 3.14 # Float
4 name = "Alice" # String
5 is_student = True # Boolean
6
7 # Print variables
8 print("x:", x)
9 print("pi:", pi)
10 print("name:", name)
11 print("is_student:", is_student)
12
13 # Check data types
14 print(type(x))
15 print(type(pi))
16 print(type(name))
17 print(type(is_student))
18
19 # List, Tuple, Dictionary, Set
20 my_list = [1, 2, 3, 4]
21 my_tuple = (5, 6, 7, 8)
22 my_dict = {"key1": "value1", "key2": "value2"}
23 my_set = {9, 10, 11}
24
25 print("List:", my_list)
26 print("Tuple:", my_tuple)
27 print("Dictionary:", my_dict)
28 print("Set:", my_set)
```

5

Lesson 1: Introduction to Python

Python Basics

Conditional statements (if-else)



```
1 # Example of if-else
2 x = 15
3 if x > 10:
4     print("x is greater than 10")
5 elif x == 10:
6     print("x is equal to 10")
7 else:
8     print("x is less than 10")
```

Lesson 1: Introduction to Python

Python Basics

Loops (while/for)

```
1 ## Loops
2
3 # For loop example
4 for i in range(5):
5     print("For loop iteration:", i)
6
7 # While loop example
8 counter = 0
9 while counter < 5:
10     print("While loop iteration:", counter)
11     counter += 1
```

Lesson 1: Introduction to Python

Python Basics

Functions



```
1 # Define a simple function
2 def greet(name):
3     return f"Hello, {name}!"
4
5 # Test the function
6 print(greet("Alice"))
```


Lesson 1: Introduction to Python

Python Basics

Writing and running scripts



```
1 # Example of writing a script (this part would normally go in a .py file)
2 # Save this content as script.py
3 # Then run it in the terminal with: python script.py
4 if __name__ == "__main__":
5     print("This script is being run directly.")
```

Setting up Google Collab

Lesson 1: Introduction to Python

Hands-on activity

1. Launch Google Colab
2. Define a numerical variable
3. Check if the variable is a prime number using if-else
4. Write a loop to print all prime numbers between 0 and 100
5. <optional> Create a function that when called prints the next prime number based on the one that has been passed

Lesson 1: Introduction to Python

Summary

- Python is a powerful tool for data analysis
- Basic Python syntax includes variables, loops, and conditionals
- Jupyter Notebooks provide an interactive environment for coding

PART 2

Lesson 1: Python libraries for Data Analysis

Fundamental python libraries

- **Numpy**
- **Pandas**
- **Matplotlib**

Lesson 1: Python libraries for Data Analysis

Numpy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects, and an assortment of routines for fast operations on arrays and much more.



Lesson 1: Python libraries for Data Analysis

What is an array?

An Array is a linear data structure where all elements are arranged sequentially.

✓
0 s



```
1 # Creating a 1D array
2 array_1d = np.array([1, 2, 3, 4, 5])
3 print("1D Array:", array_1d)
```



1D Array: [1 2 3 4 5]

✓
0 s

[3]

```
1 # Creating a 2D array
2 array_2d = np.array([[1, 2, 3], [4, 5, 6]])
3 print("\n2D Array:\n", array_2d)
```



2D Array:
[[1 2 3]
[4 5 6]]

Lesson 1: Python libraries for Data Analysis

Pandas

Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the [Python](#) programming language.



Lesson 1: Python libraries for Data Analysis

Pandas

With pandas we can create *dataframes* that help us collect and analyze our data

✓
js



```
1 # Creating a DataFrame from a dictionary
2 data = {
3     "Name": ["Alice", "Bob", "Charlie"],
4     "Age": [25, 30, 35],
5     "City": ["New York", "Los Angeles", "Chicago"]
6 }
7 df = pd.DataFrame(data)
8 print("DataFrame:\n", df)
```



DataFrame:

	Name	Age	City
0	Alice	25	New York
1	Bob	30	Los Angeles
2	Charlie	35	Chicago

Lesson 1: Python libraries for Data Analysis

Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations.



Lesson 1: Python libraries for Data Analysis

Hands-on activity

1. Create a NumPy array containing odd numbers from 1 to 10 (call it *odd*).
2. Create a NumPy array containing even numbers from 1 to 10 (call it *even*).
3. Print non-prime numbers from *odd*.
4. Create a DataFrame containing two columns: odd and even using the respective arrays
5. Add a new column called *total*, which contains the sum of the values at the same index from *odd* and *even*.
6. Add a new column called *is_odd* containing **true** if the value of *total* is odd, **false** otherwise

Lesson 1: Python libraries for Data Analysis

Summary

- NumPy and pandas are essential libraries for data manipulation
- Arrays and DataFrames are the core data structures
- Understanding these libraries is crucial for efficient analysis