# **Python Basics**[**¶**](http://localhost:8889/notebooks/wk1_d1/exercises/01_python_basic_exercises.ipynb#Python-Basics)

Exercise 1: Basic Data Types and Strings

1. Create a variable of each data type (int, float, string, boolean).
2. Concatenate two strings and print the result.
3. Convert an integer to a string and concatenate it with another string.

Exercise 2: Working with Dates & Times

1. Get the current date and time.
2. Format the date as YYYY-MM-DD.
3. Calculate the number of days until your next birthday.

Exercise 3: String Manipulation

1. Convert a string to uppercase.
2. Find the position of a substring within a string.
3. Replace all instances of a substring with another substring.

Exercise 4: Lists and Basic Operations

1. Create a list of integers.
2. Append an integer to the list.
3. Remove an integer from the list.
4. Sort the list in ascending order.

Exercise 5: Dictionaries and Sets

1. Create a dictionary with three key-value pairs.
2. Add a new key-value pair to the dictionary.
3. Create a set and add three elements to it.
4. Check if an element exists in the set.

Exercise 6: Date and Time Arithmetic

1. Calculate the difference in days between two dates.
2. Add 10 days to the current date.
3. Subtract 5 hours from the current time.

Exercise 7: Advanced String Operations

1. Split a string into a list of words.
2. Join a list of words into a single string with pipes separating them.
3. Count the number of occurrences of a character in a string.

Exercise 8: List Comprehensions and Filtering

1. Use a list comprehension to create a list of squares of integers from 1 to 10.
2. Filter a list to include only even numbers.
3. Use a dictionary comprehension to create a dictionary from two lists.

Exercise 9: Nested Collections and Data Aggregation

1. Create a list of dictionaries representing students with their names and grades.
2. Calculate the average grade of the students.
3. Find the student with the highest grade.

Exercise 10: Complex Date and Time Operations

1. Parse a string into a datetime object.
2. Create a function that takes two dates and returns the number of weekdays between them.
3. Create a function that accepts a start date, end date, meeting datetime, and interval and returns all occurrences of the event within that timeframe.