**Python for data science, AI and development**

**Module 1 Python basics**

* 1. **About the course**

1. This course was designed to provide the building blocks for Python programming and data collection for those choosing a career in Data Science, Data Engineering, AI or Application Development.
2. Initially conceived as a foundation course for Data Science and AI it has been refreshed several times to keep pace with emerging career options. Additional content has been added which is applicable to Data Science, Data Engineering, AI or Application Development.
3. After completing this course you will have learned foundational skills in Python programming which you can then go on to apply in the Python Project course for your chosen career.  The Python Project courses involve real world scenarios where you are in charge of a final project as a Data Scientist, a Data Engineer, or in AI and Application Development. By finishing this course and your follow-on Python Project, you will gain the basic skills to continue the steps on your chosen career path.

**Course overview**

Welcome to the Python for Data Science, AI, and Development course. After completing this course, you'll possess the basic knowledge of Python and acquire a good understanding of different data types. You’ll also learn to use lists and tuples, dictionaries, and Python sets. Additionally, you’ll acquire the concepts of condition and branching and will know how to implement loops, create functions, perform exception handling, and create objects. Furthermore, you’ll be proficient in reading and writing files and will be able to implement unique ways to collect data using APIs and web scraping. In addition to the module labs, you'll prove your skills in a peer-graded project and your overall knowledge with the final quiz.

Course Content

This course is divided into five modules. You should set a goal to complete at least one module per week.

Module 1: Python Basics

* About the Course
* Types
* Expressions and Variables
* String Operations

Module 2: Python Data Structures

* Lists and Tuples
* Dictionaries
* Sets

Module 3: Python Programming Fundamentals

* Conditions and Branching
* Loops
* Functions
* Exception Handling
* Objects and Classes
* Practice with Python Programming Fundamentals

Module 4: Working with Data in Python

* Reading and Writing Files with Open
* Pandas
* Numpy in Python

Module 5: APIs and Data Collection

* Simple APIs
* REST APIs, Web Scraping, and Working with Files
* Final Exam

The course contains a variety of learning assets: Videos, activities, labs, projects, practice, graded quizzes, and readings. The videos and readings present the instruction. Labs and activities support that instruction with hands-on learning experiences. Discussions allow you to interact and learn from your peers. A peer-review project that mimics real-world scenarios encourage you to showcase your skills, Practice quizzes enable you to test your knowledge of what you learned. Finally, graded quizzes indicate how well you have learned the course concepts.

**1.2 Getting started with python and Jupiter**

**Introduction to Jupyter**

Jupyter is a freely available web application that enables creation and sharing of documents containing equations, live coding, visualizations, and narrative text. Jupyter provides an interactive computing environment that supports multiple programming languages, including Python, R, Julia, and more, but it shines brightest when used with Python. Jupyter revolves around notebooks, documents containing a mix of code, visualizations, narrative text, equations, and multimedia content. These notebooks allow users to create, share, and collaborate on computational projects seamlessly.

Why Jupyter?

Jupyter's popularity stems from its flexibility and ease of use. Regardless of your level of programming expertise, whether you're an experienced coder or embarking on your data science journey, Jupyter offers an intuitive platform for writing, testing, and sharing code. Its interactive interface enables data exploration, algorithm experimentation, and result visualization—all seamlessly integrated within a unified environment.

Key Features of Jupyter

Here are some key features and advantages of Jupyter:

1. Interactive Computing: Jupyter notebooks enable users to write and execute code interactively. This means you can run code cells individually and see the output immediately, fostering an iterative approach to coding and experimentation.
2. Support for Multiple Languages: While Jupyter was initially developed for Python (hence the name, which stands for Julia, Python, and R), it now supports various programming languages through its kernel system. This flexibility makes Jupyter suitable for various computational tasks and interdisciplinary collaboration.
3. Rich Output: Jupyter Notebooks support rich media integration, allowing users to generate interactive plots, charts, images, videos, and more directly within the document. This makes visualizing data, communicating findings, and creating compelling narratives easier.
4. Integration with Data Science Libraries: Jupyter seamlessly integrates with popular libraries and frameworks used in the data science ecosystem, such as NumPy, Pandas, Matplotlib, sci-kit-learn, TensorFlow, and PyTorch. This allows users to leverage the full power of these tools within the notebook environment for tasks like data manipulation, visualization, machine learning, and deep learning.
5. Collaboration and Sharing: Jupyter promotes collaboration and reproducibility by allowing users to share their notebooks with others via email, GitHub, or the Jupyter Notebook Viewer. This facilitates knowledge sharing, peer review, and interdisciplinary collaboration, as users can easily exchange ideas, code snippets, and best practices.

Jupyter in Data Science

Jupyter has become an indispensable tool for researchers, analysts, and developers in data science. Its seamless integration with popular libraries such as NumPy, pandas, and sci-kit-learn makes it the go-to choice for data manipulation, analysis, and machine learning. Jupyter provides a user-friendly interface, interactive capabilities, and robust collaboration features, making it an essential tool for anyone involved in data analysis, scientific research, education, or software development. Whether you're exploring data, building machine learning models, teaching a class, or conducting research, Jupyter empowers you to work more efficiently and share your insights with others.

Getting Started with Jupyter

Now that you can glimpse what Jupyter offers, it's time to dive in and experience its capabilities firsthand. Our Getting Started with Jupyter video will walk you through the basics of setting up and using Jupyter, empowering you to unleash the full potential of Python and embark on your data science journey with confidence.

**1.3 Types**

**1.4 Expressions and variables**

**1.4 String operations**

**Summary**

* Python can distinguish among data types such as integers, floats, strings, and Booleans.
* Integers are whole numbers that can be positive or negative.
* Floats include integers as well as decimal numbers between the integers.
* You can convert integers to floats using typecasting, but you cannot convert a float to an integer.
* You can convert integers and floats to strings.
* You can convert an integer or float value to True (1) or False (0).
* Expressions in Python are a combination of values and operations used to produce a single result.
* Expressions perform mathematical operations such as addition, subtraction, multiplication, and so on.
* We use"//" to round off integer divisions, resulting in float values.
* Python follows the order of operations (BODMASS) to perform operations with multiple expressions.
* Variables store and manipulate data, allowing you to access and modify values throughout your code.
* The assignment operator "=" assigns a value to a variable.
* ":" denotes the value of the variable within the code.
* Assigning another value to the same variable overrides the previous value of that variable.
* You can perform mathematical operations on variables using the same or different variables.
* While performing operations with various variables, modifying a value in one variable will lead to changes in the other variables.
* Python string operations involve manipulating text data using tasks such as indexing, concatenation, slicing, and formatting.
* A string is usually written within double quotes or single quotes, including letters, white space, digits, or special characters.
* A string attaches to another variable and is an ordered sequence of characters.
* Characters in a string identify their index numbers, which can be positive or negative.
* We use strings as a sequence to perform sequence operations.
* You can input a stride value to perform slicing while operating on a string.
* Operations like finding the length of the string, combining, concatenating, and replicating, result in a new string.
* You cannot modify an existing string; they are immutable.
* You can perform escape sequences using " " to change the layout of the string.
* In Python, you perform tasks such as searching, modifying, and formatting text data with its pre-built string methods functions.
* You apply a method to a string to change its value, resulting in another string.
* You can perform actions such as changing the case of characters in a string, replacing items in a string, finding items in a string, and so on using pre-built string methods.

**Link:**

[**https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL0NoZWF0X1NoZWV0X1dlZWstMS5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg1OTl9.Y\_C9kDKasuzH63vioeVvOEqVpsInCG9DLVj53MUyJg0**](https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL0NoZWF0X1NoZWV0X1dlZWstMS5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg1OTl9.Y_C9kDKasuzH63vioeVvOEqVpsInCG9DLVj53MUyJg0)

**Module 2 Data structor**

**2.1 Lists and Tuples**

**Cheatsheet**

**Link:**

<https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstMi5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg1OTl9.4L5yEN_BpE8odg36ZiLmkRTf3clYER2hiGe0Bu74ljo>

**2.2 Dictionary**

**2.3 Set**

**Summary**

* In Python, we often use tuples to group related data together.Tuples refer to ordered and immutable collections of elements.
* Tuples are usually written as comma-separated elements in parentheses “()".
* You can include strings, integers, and floats in tuples and access them using both positive and negative indices.
* You can perform operations such as combining, concatenating, and slicing on tuples.
* Tuples are immutable, so you need to create a new tuple to manipulate it.
* Tuples, termed nesting, can include other tuples of complex data types.
* You can access elements in a nested tuple through indexing.
* Lists in Python contain ordered collections of items that can hold elements of different types and are mutable, allowing for versatile data storage and manipulation.
* A list is an ordered sequence, represented with square brackets "[]".
* Lists possess mutability, rendering them akin to tuples.
* A list can contain strings, integers, and floats; you can nest lists within it.
* You can access each element in a list using both positive and negative indexing.
* Concatenating or appending a list will result in the modification of the same list.
* You can perform operations such as adding, deleting, splitting, and so forth on a list.
* You can separate elements in a list using delimiters.
* Aliasing occurs when multiple names refer to the same object.
* You can also clone a list to create another list.
* Dictionaries in Python are key-value pairs that provide a flexible way to store and retrieve data based on unique keys.
* Dictionaries consist of keys and values, both composed of string elements.
* You denote dictionaries using curly brackets.
* The keys necessitate immutability and uniqueness.
* The values may be either immutable or mutable, and they allow duplicates.
* You separate each key-value pair with a comma, and you can use color highlighting to make the key more visible.
* You can assign dictionaries to a variable.
* You use the key as an argument to retrieve the corresponding value.
* You can make additions and deletions to dictionaries.
* You can perform an operation on a dictionary to check the key, which results in a true or false output.
* You can apply methods to obtain a list of keys and values in a dictionary.
* Sets in Python are collections of unique elements, useful for tasks such as removing duplicates and performing set operations like union and intersection. Sets lack order.
* Curly brackets "{}" are helpful for defining elements of a set.
* Sets do not contain duplicate items.
* A list passed through the set function generates a set containing unique elements.
* You use “Set Operations” to perform actions such as adding, removing, and verifying elements in a set.
* You can combine sets using the ampersand "&" operator to obtain the common elements from both sets.
* You can use the Union function to combine two sets, including both the common and unique elements from both sets.
* The sub-set method is used to determine if two or more sets are subsets.

**Cheatsheet**

[**https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstMl9QYXJ0LTIubWQiLCJ0b29sX3R5cGUiOiJpbnN0cnVjdGlvbmFsLWxhYiIsImFkbWluIjpmYWxzZSwiaWF0IjoxNzExNjM4NTk2fQ.luBpuMXe9qHg0AyG0CeoUDtX9HzrpDcLZMm6gP-p5Dw**](https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstMl9QYXJ0LTIubWQiLCJ0b29sX3R5cGUiOiJpbnN0cnVjdGlvbmFsLWxhYiIsImFkbWluIjpmYWxzZSwiaWF0IjoxNzExNjM4NTk2fQ.luBpuMXe9qHg0AyG0CeoUDtX9HzrpDcLZMm6gP-p5Dw)

**Module 3**

**3.1 Conditions and Branching**

**3.2 Loops**

**3.3 Functions**

**3.4 Exception Handling**

[**https://docs.python.org/3/library/exceptions.html**](https://docs.python.org/3/library/exceptions.html)

**Summary**

* Python conditions use “if” statements to execute code based on true/false conditions created by comparisons and Boolean expressions.
* Comparison operations require using comparison operators equal to "=", greater than ">", less than "<".
* An exclamation mark "!" is used to define inequalities of a variable.
* You can compare integers, strings, and floats.
* Python branching directs program flow by using conditional statements (for example, if, else, elif) to execute different code blocks based on conditions or tests.
* You can use the "if" statement with conditions to define actions if true.
* To perform actions based on true or false output, you can use the "else" statement with conditions.
* The elif statement allows for additional checks only if the initial condition is false.
* To execute various operations on Boolean values, we use Boolean logic operators.
* Python loops are control structures that automate repetitive tasks and iterate over data structures like lists or dictionaries.
* The range() function generates a sequence of numbers with a specified start, stop, and step value for loops in Python.
* A for loop in Python iterates over a sequence, such as a list, tuple, or string, and executes a block of code for each item in the sequence.
* A while loop in Python executes a block of code as long as a specified condition remains true.
* Python functions are reusable code blocks that perform specific tasks, take input parameters, and often return results, enhancing code modularity and reusability.
* You may or may not have written the codes that are often included in functions.
* Python has a set of built-in functions such as "len" to find the length of a sequence or "sum" to find the total sum of a sequence.
* The "sorted" function creates a new sorted list, while "sort" sorts items in the original list.
* You can also create your own functions in Python.
* To ensure clarity and organization and facilitate understanding and maintenance of the code, developers must document functions using a documentation string enclosed in three quotes.
* The help command will return the documentation defined for a particular function.
* A function can have multiple parameters.
* “No return” statement in the function means that the function will return nothing.
* The "No work" function does not execute any task. You can use the "pass" keyword to meet the requirement of a non-empty body.
* A function will usually perform more than one task.
* In Python, the scope of a variable determines where you can access or modify that variable. Global scope allows access from anywhere, while local scope restricts it to a block or function.
* In Python, a programmer defines a local variable within a specific block or function, which can only be accessed or modified within that block or function.
* In Python, a global variable is a variable defined at the top level of a program that any part of the code can access or modify.
* Exception handling in Python is a mechanism for managing and responding to errors and exceptions that may occur during program execution, preventing them from crashing the program.
* In Python, you use the "try-except" statement to attempt a block of code and specify alternative actions to execute if an error occurs, allowing you to handle exceptions.
* In Python, you use the "try-except-else" statement to attempt a block of code, handle exceptions in the "except" block, and execute code in the "else" block when no exceptions occur.
* Python developers use the "try-except-else-finally" statement to attempt a block of code, catch exceptions in the "except" block, execute code in the "else" block when no exceptions occur, and ensure that the "finally" block always runs, regardless of whether an exception raised or not.
* In Python, objects are instances of classes that encapsulate data and behavior, serving as the foundation for creating and working with various data types and custom data structures.
* To determine the type of an object in Python, you can use the `type()` command.
* Any changes made within the method of the object may result in a change in object type.
* Classes in Python are blueprints for creating objects, defining their attributes and methods, enabling code organization, and object-oriented programming.
* Function "init" is a special method used to initialize data attributes.
* We can create instances of a class in Python.
* Data attributes consist of the data defining the objects.
* Methods are functions that interact and change the data attributes.
* The method has a function that requires the self as well as other parameters.

**Cheatsheet**

[**https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstMy5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg2MDF9.3wu94h79vCdZMomR4hNKjY9JDxWK6FXB1hbKYV7RHWg**](https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstMy5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg2MDF9.3wu94h79vCdZMomR4hNKjY9JDxWK6FXB1hbKYV7RHWg)

**Module 4**

* 1. **Reading and writing files with open**

| **Mode** | **Syntax** | **Description** |
| --- | --- | --- |
| **‘r’** | **'r'** | **Read mode. Opens an existing file for reading. Raises an error if the file doesn't exist.** |
| **‘w’** | **'w'** | **Write mode. Creates a new file for writing. Overwrites the file if it already exists.** |
| **‘a’** | **'a'** | **Append mode. Opens a file for appending data. Creates the file if it doesn't exist.** |
| **‘x’** | **'x'** | **Exclusive creation mode. Creates a new file for writing but raises an error if the file already exists.** |
| **‘rb’** | **'rb'** | **Read binary mode. Opens an existing binary file for reading.** |
| **‘wb’** | **'wb'** | **Write binary mode. Creates a new binary file for writing.** |
| **‘ab’** | **'ab'** | **Append binary mode. Opens a binary file for appending data.** |
| **‘xb’** | **'xb'** | **Exclusive binary creation mode. Creates a new binary file for writing but raises an error if it already exists.** |
| **‘rt’** | **'rt'** | **Read text mode. Opens an existing text file for reading. (Default for text files)** |

**Summary**

* Python uses the open() function and allows you to read and write files, providing access to the content within the file for reading. It also allows overwriting it for writing and specifies the file mode (for example, r for reading, w for writing, a for appending).
  + To read a file, Python uses an open function along with *r.*
  + Python uses theopen with function to read and process a file attribute, that is, from open to close.
  + In Python, you use the open method to edit or overwrite a file.
  + To write a file, Python uses the open function along with *w.*
  + In Python, "a" indicates that the program has appended to the file.
  + In Python, “\n” signifies that the code should start on a new line.
  + Python uses various methods to print lines from attributes.
* Pandas is a powerful Python library for data manipulation and analysis, providing data structures and functions to work with structured data like data frames and series.
  + You import the file (panda) by using the import command followed by the file name.
  + In Python, you use the as command to provide a shorter name for the file.
  + In Pandas, you use a data frame (df) to specify the files to read.
  + DataFrames consist of rows and columns.
  + You can create new DataFrames by using the column or columns of a specific DataFrame.
  + We can work with data in a DataFrames and save the results in different formats.
  + In Python, you use the Unique method to determine unique elements in a column of the DataFrames.
  + You use the inequality operator along with df to assign a Boolean value to the selected column in DataFrames.
  + You save a new DataFrame as a different DataFrame, which may contain values from an earlier DataFrame.
* NumPy is a Python library for numerical and matrix operations, offering multidimensional array objects and a variety of mathematical functions to work with data efficiently.
  + NumPy is a basis for Pandas.
  + A NumPy array or ND array is similar to a list, usually of a fixed size with the same kind of element.
* A one-dimensional NumPy array is a linear sequence of elements with a single axis, like a traditional list, but optimized for numerical computations and array operations.
  + You can access elements in a NumPy using an index.
  + You use the attribute dtype to get the data type of the array elements.
  + You use nsize and ndim to get the size and dimension of the array, respectively.
  + You can use indexing and slicing methods in NumPy.
  + Vector additions are widely used operations in Python.
  + Representing vector addition with line segments or arrows is useful.
  + NumPy codes work much faster, which is helpful with lots of data.
  + You perform vector subtraction by replacing the addition sign with a negative sign.
  + Multiplying an array by a scalar in Python entails multiplying each element of the array by the scalar value, leading to a new array in which each element scales by the scalar.
  + Hadamard product refers to the element-wise multiplication of two arrays of the same shape, resulting in a new array where each element is the product of the corresponding elements in the input arrays.
  + The dot product in Python is the sum of the element-wise products of two arrays, often used for vector and matrix operations to find the scalar result of multiplying corresponding elements and summing them.
  + When working with NumPy, it is common to utilize libraries like Matplotlib to create graphs and visualizations from numerical data stored in NumPy arrays.
* A two-dimensional NumPy array is a grid-like structure with rows and columns suitable for representing data as a matrix or a table for numerical computations.
  + In NumPy, "shape" refers to an array's dimensions (number of rows and columns), indicating its size and structure.
  + You use the attribute "size" to obtain the size of an array.
  + You use rectangular attributes to access the various elements in an array.
  + You use a scalar to multiply elements in NumPy.

**Cheatsheet**

[**https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstNC5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg2MDN9.IJuVPT8a0WmMscsxZAgRT0EvHa-vLngdkX-wwrGz7tE**](https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstNC5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg2MDN9.IJuVPT8a0WmMscsxZAgRT0EvHa-vLngdkX-wwrGz7tE)

**Module 5**

**5.1 Simple API – NBA-API**

**5.2 Rest API, Web scraping and working with files.**

**Custom data extraction**

Web scraping allows you to navigate the HTML structure and extract specific information based on your requirements. This process may involve finding specific tags, attributes, or text content within the HTML document.

**Using BeautifulSoup for HTML parsing**

Beautiful Soup is a powerful tool for navigating and extracting specific web page parts. It allows you to find elements based on their tags, attributes, or text, making extracting the information you're interested in easier.

**Using pandas read\_html for table extraction**

Pandas, a Python library, provides a function called read\_html, which can automatically extract data from websites' tables and present it in a format suitable for analysis. It’s similar to taking a table from a webpage and importing it into a spreadsheet for further analysis.

**Get Methods**

* get\_cell()
* get\_city()
* get\_dob()
* get\_email()
* get\_first\_name()
* get\_full\_name()
* get\_gender()
* get\_id()
* get\_id\_number()
* get\_id\_type()
* get\_info()
* get\_last\_name()
* get\_login\_md5()
* get\_login\_salt()
* get\_login\_sha1()
* get\_login\_sha256()
* get\_nat()
* get\_password()
* get\_phone()
* get\_picture()
* get\_postcode()
* get\_registered()
* get\_state()
* get\_street()
* get\_username()
* get\_zipcode()

**Links Free API:**

[**https://mixedanalytics.com/blog/list-actually-free-open-no-auth-needed-apis/**](https://mixedanalytics.com/blog/list-actually-free-open-no-auth-needed-apis/)

**Web Scraping with Python**

1. BeautifulSoup: BeautifulSoup is a Python library used for web scraping purposes to pull the data out of HTML and XML files. It creates a parse tree from page source code that can be used to extract data in a hierarchical and more readable manner.
2. from bs4 import BeautifulSoup
3. import requests
4. URL = "http://www.example.com"
5. page = requests.get(URL)
6. soup = BeautifulSoup(page.content, "html.parser")
7. Scrapy: Scrapy is an open-source and collaborative web crawling framework for Python. It is used to extract the data from the website.
8. import scrapy
9. class QuotesSpider(scrapy.Spider):
10. name = "quotes"
11. start\_urls = ['http://quotes.toscrape.com/tag/humor/',]
12. def parse(self, response):
13. for quote in response.css('div.quote'):
14. yield {'quote': quote.css('span.text::text').get()}
15. Selenium: Selenium is a tool used for controlling web browsers through programs and automating browser tasks.
16. from selenium import webdriver
17. driver = webdriver.Firefox()
18. driver.get("http://www.example.com")

**Module 5 Summary: APIs and Data Collection**

* Simple APIs in Python are application programming interfaces that provide straightforward and easy-to-use methods for interacting with services, libraries, or data, often with minimal configuration or complexity.
  + An API lets two pieces of software talk to each other.
  + Using an API library in Python entails importing the library, calling its functions or methods to make HTTP requests, and parsing the responses to access data or services provided by the API.
  + Pandas API processes the data by communicating with the other software components.
  + An Instance forms when you create a dictionary and then use the DataFrames constructor to create a Pandas object.
  + Method “head()” will display the mentioned number of rows from the top (default 5) of DataFrames, while method “mean()” will calculate the mean and return the values
* Rest APIs allow you to communicate through the internet, taking advantage of resources like storage, access more data, AI algorithms, and so on.
  + HTTP methods transmit data over the internet.
  + An HTTP message typically includes a JSON file with instructions for operations.
  + HTTP messages containing JSON files are returned to the client as a response from web services.
  + Dealing with time series data involves using the Pandas time series function.
  + You can get data for daily candlesticks and plot the chart using Plotly with the candlestick plot.
* The HTTP (HyperText Transfer Protocol) transfers data, including web pages and resources, between a client (a web browser) and a server on the World Wide Web.
  + The HTTP protocol is commonly used for implementing various types of REST APIs.
  + An HTTP response includes information like the type of resource, length of resource, and so on
  + Uniform resource locator (URL) is the most popular way to find resources on the web.
  + URL is divided into three parts: scheme, internet address or base URL, and route
  + The GET method is one of the popular methods of requesting information. Some other methods may also include the body.
  + Response method contains the version and body of the response.
  + POST submits data to the server, PUT updates data already on the server, DELETE deletes data from the server
* Requests is a Python library that allows you to send HTTP/1.1 requests easily
  + You can modify the results of your query with the GET method.
  + You can obtain multiple requests from a URL like name, ID, and so on with a Query string.
* Web scraping in Python involves extracting and parsing data from websites to gather information for various applications, using libraries like Beautiful Soup and requests.
  + HTML comprises text surrounded by blue text elements enclosed in angular brackets called tags.
  + You can select an HTML element on a web page to inspect the webpage.
  + Web pages may also contain CSS and JavaScript along with HTML elements.
  + Each HTML document is like an HTML Tree, which may contain strings and other tags.
  + Each HTML table is comprised of table tags and is structured with elements such as rows, headers, body and so on.
* Tabular data can also be extracted from web pages using the `read\_html` method in Pandas.
* Beautiful Soup in Python is a library for parsing and navigating HTML and XML documents, making extracting, and manipulating data from web pages more accessible.
* To parse a document, pass it through the Beautiful Soup constructor to get a beautiful soup object representing the document as a nested data structure.
* Beautiful soup represents HTML as a set of tree-like objects with methods to parse the HTML.
* Navigable string is like a Python string that supports beautiful soup functionality.
* find\_all is a method used to extract content based on the tag’s name, its attributes, the text of a string, or some combination of these.
* The find\_all method looks through a tag’s descendants and retrieves all descendants that match your filters.
* The result is a Python iterable like a list.
* File formats refer to the specific structure and encoding rules used to store and represent data in files, such as .txt for plain text or .csv for comma-separated values.
* Python works with different file formats such as CSV, XML, JSON, xlsx, and so on
* The extension of a file name will let you know what type of file it is and what it needs to open with.
* To access data from CSV files, we can use Python libraries such as Pandas.
* Similarly, different methods help parse JSON, XML, and other files.

**Cheatsheet**

[**https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstNS5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg1OTZ9.ZQlNT\_yAcq8g5LaQiL9FZqF7w0EMTFEFUTY8edNA8mA**](https://author-ide.skills.network/render?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJtZF9pbnN0cnVjdGlvbnNfdXJsIjoiaHR0cHM6Ly9jZi1jb3Vyc2VzLWRhdGEuczMudXMuY2xvdWQtb2JqZWN0LXN0b3JhZ2UuYXBwZG9tYWluLmNsb3VkL0lCTURldmVsb3BlclNraWxsc05ldHdvcmstUFkwMTAxRU4tU2tpbGxzTmV0d29yay9sYWJzL2hhbmRvdXRzL0NoZWF0X1NoZWV0X1dlZWstNS5tZCIsInRvb2xfdHlwZSI6Imluc3RydWN0aW9uYWwtbGFiIiwiYWRtaW4iOmZhbHNlLCJpYXQiOjE3MTE2Mzg1OTZ9.ZQlNT_yAcq8g5LaQiL9FZqF7w0EMTFEFUTY8edNA8mA)