Guia Ágil

TechGuide - Alura

Python

Nivel 1

☐ **Python - Fundamentals:**

- Python is a high-level, general-purpose programming language that is widely used in web applications, software development, data science, and machine learning. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected.

- Knowing the primitive types

- Declaring variables, considering the different types

- Using conditional structures ('if', 'else')

- Knowing the assignment and comparison operators

- Using repetition structures and loops ('while', 'for')

- Using functions, passing parameters and arguments

- Manipulating methods

- Manipulating arrays and lists

- Getting data from an API

- Creating constructors

☐ **Object-oriented Programming Concepts:**

- Object-oriented programming (OOP) is a programming paradigm based on the concept of 'objects', which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods). A common feature of objects is that procedures (or methods) are attached to them and can access and modify the object's data fields. Some of the main concepts are classes and instances, inheritance, and encapsulation.

- How objects work

- Creating and using constructors

- What classes are

- Creating and using Methods

- How encapsulation works

- What inheritance is

- What polymorphism is

- How interfaces work

- What abstractions are

☐ **Data Structures**:

- In the context of computers, the data structure is a specific way of storing and organizing data in the computer's memory so that these data can be easily retrieved and efficiently used when needed later.

- Knowing the main data structures (linked list, stack, queue, tree, etc)

- Implementing the main data structures

☐ **Python - Collections**:

- A collection represents a group of objects, known as its elements. They are like containers that group multiple items into a single unit. Some collections allow duplication of elements and some do not. Some are ordered and some are unordered.

- Using lists and tuples

- Using polymorphism in collections

- Using sets and dictionaries

☐ **Python - Testing**:

- Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. The benefits of testing include preventing bugs, reducing development costs and improving performance.
- Using unit tests
- Using integration testing
- Using behavioral testing
- Using mocks

☐ **Python - Communication with APIs**:

- An API is an interface that software developers use to program interaction with software components or features outside their own code. An even simpler definition is that an API is the part of a software component that is accessible to other components.
- Understanding what a REST API is
- Knowing the basic HTTP communication commands
- Understanding what a REST API is
- Knowing how to make authenticated requests
- Converting objects to JSON and vice-versa
- Knowing how to use the tools of the Requests package

Nivel 2

☐ **Flask**:

- Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However,

Flask supports extensions that can add application features as if they were implemented in Flask itself.

- Creating web applications

- Defining routes, redirects and templates

- Validating forms

☐ **Python - Advanced OOP**:

- Mixin is a class that provides implementations of methods for reuse by multiple related child classes.

- Operator overloading means giving extended meaning beyond its predefined operational meaning.

☐ **Django**:

- Django is a high-level Python web framework that enables rapid development of secure and maintainable websites.

- Creating web applications

- Understanding the architecture of an application made with Django

- Creating a page admin

- Using templates and routes

- Creating forms

- Writing views

☐ **Django Rest Framework**:

- Django REST framework is a powerful and flexible toolkit for building Web APIs.

- Developing APIs

- Working with models, serializers and views

- Including filters, searches and sorting

- Limiting the number of requests

☐ **Python - MVC and MTV**:

- MVC and MTV are two design patterns used for implementing web interfaces and applications.

- Understanding the MVC pattern

- Understanding the MTV pattern

- Understanding the difference between the MVC and MTV patterns

☐ **Python - Lambdas and Closures**:

- Lambda functions are nothing more than anonymous functions. While normal functions can be created using def as a prefix, lambda functions are created using lambda.

- A closure in Python is an inner function object, a function that behaves like an object, that remembers and has access to variables in the local scope where it was created, even after the outer function has finished executing. It can also be defined as a means of binding data to a function without passing it as a parameter.

Nivel 3

☐ **Microservices architecture**:

- Microservices are an architectural approach in which software consists of small independent services that communicate with each other and are organized according to their business domains.

- Learning the concept of planned architecture for microservices

- Performing communication using APIs

- Improving the scalability of a system

☐ **Containers**:

- Containers are software packages that contain all the elements needed to run in any environment.

- Kubernetes (also known as k8s or "kube") is an open source container orchestration platform that automates many of the manual processes involved in deploying, managing, and scaling containerized applications.

- Isolating your software to run independently
- Deploying software in clusters
- Modularizing your system into smaller packages
- Getting to know the Docker platform
- Getting to know Kubernetes

☐ **Python - Static typing**:

- Python is a dynamically typed language, that is, there is no need to think about data types. Statically typed languages (such as C or Java) perform type checks upon compilation. It may feel safer, as you can immediately tell the type of parameter going into each function.
- Getting to know type hinting

☐ **Python - Generators**:

- Generator functions allow you to declare a function that behaves like an iterator, for example, it can be used in a 'for' loop.
- Creating iterative objects
- Using lazy evaluation
- Running concurrent tasks
- Using the reserved word 'yield'

☐ **Python - Asynchronous**:

- In asynchronous programming, the functions are not executed in order. We can interrupt the code to get some other information needed to continue execution. This means that the code waits for another part of the code, and while it waits, it can execute the other parts.
- Learning about coroutines
- Coroutines are a generalization of subroutines. They are used for cooperative multitasking where a process yields control voluntarily, periodically or when idle, in order to allow multiple applications to run simultaneously.

- Dealing with concurrency

- Knowing the concept of waitable objects

- Creating concurrent tasks

- Knowing the 'asyncio' library

☐ **Python - args & kwargs**:

- The magic variables *args and **kwargs are commonly used in function definitions, and are used to pass an unknown number of arguments to a function.

- Understanding the difference between *args and **kwargs

☐ **Python - Dunder methods**:

- Special methods or magic methods in Python are methods that are predefined in all objects, with automatic invocation under special circumstances. They are usually not called directly by the user but can be overloaded (overwritten and changed). Their names begin and end with double underscores called 'dunder' (an expression derived from double underscore).

- Understanding the concept of special (or magic) methods

- Knowing the main magic methods and how to use them

☐ **Python - Metaprogramming**:

- Metaprogramming is a programming technique in which computer programs have the ability to treat other programs as their data. It means that a program can be designed to read, generate, analyze or transform other programs, and even modify itself while running.

- Writing a program that manipulates other programs

- Using metaclasses

☐ **Python - Multiprocessing**:

- In Python, the multiprocessing module includes a very simple and intuitive API for dividing up work among multiple processes.

- Running processes in parallel

- Getting to know the Pool class

☐ **Reflection and attributes**:

- Reflection objects are used for obtaining type information at runtime. The classes that give access to the metadata of a running program are in the System.Reflection namespace.

- Writing code that reads object information and metadata at runtime

- Getting class names at runtime and creating objects of a class

Habilidade Auxiliar: Infrastructure

☐ **Git & GitHub - Fundamentals**:

- Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

- GitHub is a hosting service for software development and version control using Git.

- Creating a repository

- Cloning a repository

- Committing, pushing and pulling to and from the repository

- Reversing a commit

- Creating branches and pull requests

- Handling merge and conflicts

☐ **HTTP - Fundamentals**:

- HTTP stands for Hyper Text Transfer Protocol. Communication between client computers and web servers is done by sending HTTP Requests and receiving HTTP Responses.

- Understanding the difference between HTTP verbs

- Testing requests and checking the status codes in the browser

- Learning how to make a HTTP request on the command line with WGET

- Downloading an image with WGET

- Performing a POST

☐ **JSON**:

- JSON stands for JavaScript Object Notation. It is a text format for storing and transporting data.

- Creating an object

- Transforming an object into a string

- Transforming a string into an object

- Manipulating an object

☐ **Command Line - Fundamentals**:

- CLI is a command line program that accepts text input to execute operating system functions.

- Knowing the most important commands

☐ **Cloud - Fundamentals**:

- Cloud, or cloud computing, is the distribution of computing services over the Internet using a pay-as-you-go pricing model. A cloud is composed of various computing resources, ranging from the computers themselves (or instances, in cloud terminology) to networks, storage, databases, and everything around them. In other words, everything that is normally needed to set up the equivalent of a server room, or even a complete data center, will be ready to use, configured, and run.

- Knowing the difference between IaaS, PaaS and SaaS

- Knowing the largest cloud providers

- Specializing in a specific provider of your choice

☐ **SQL - Fundamentals**:

- Structured Query Language (SQL) is a standardized programming language that is used to manage relational databases and perform various operations

on the data in them.

- Knowing the most common SQL commands
- Using SELECT to query a table
- Using INSERT to insert data into a table
- Using UPDATE to update a table
- Using DELETE to remove data from a table
- Using JOIN to connect data from multiple tables
- Knowing the clauses (FROM, ORDER BY, etc.)

Habilidade Auxiliar: Good practices and tools

☐ **SOLID**:

- SOLID has five principles that are considered best practices in software development that help programmers write cleaner code by separating responsibilities, reducing docking, easing refactoring, and encouraging code reuse.

☐ **Clean Architecture**:

- Clean architecture is a way of developing software, such that just by looking at the source code of a program, you should be able to tell what the program does.

☐ **Design Patterns**:

- In software engineering, a Design Pattern is a general, reusable solution to a commonly occurring problem within a given context in software design. It is a description or template for how to solve a problem that can be used in many different situations. Design Patterns are formalized best practices that the programmer can use to solve common problems when designing an application or system.
- Getting familiarized with and applying the main Design Patterns

☐ **Data Extraction and Processing**:

- Data extraction is the process of collecting or retrieving disparate types of data from a variety of sources, many of which may be poorly organized or completely unstructured.

- Obtaining the data to be analyzed

- Treating the obtained data, transforming it, changing its structure and values in order to make the database more coherent and ensure that the data to be worked on is in the best conditions to be analyzed

- Using the Pandas and Scikit-learn libraries to treat the data

---