**Python**

By: Harrison Corbin

# Introduction:

Not long ago in the late 80s, Guido van Rossum began creating what would become the largest known programming language of the modern day, Python. It started out as a hobby project, fixing issues and flaws with the ABC language that he helped develop at Centrum Wiskunde & Informatica (CWI) in the Netherlands. He named it after the hit show “Monty Python’s Flying Circus” and was finally released in 1991, version 1.0 later released in 1994. Compared to other programming languages like Java, C++, or C it was able to express the same concepts in many fewer lines of code. This is why Python is one of the top three most recommended languages for new programmers. Python supports four main programming paradigms: imperative, functional, procedural, and object oriented, each with its own specific uses for the different types of programs. Python has many uses as well, from Web Development to Databases to Game Design, all having various uses for Python.

# Data Abstractions:

Python has four primitive data types: Integers, Floats, Strings, and Booleans. There’s also the non-primitive data types such as Arrays and Lists. Due to being a dynamically typed language, Python checks its variables during runtime instead of once it’s compiled. Along with being dynamic, Python is also an interpreted language This also increases its portability, allowing for a Python program to be run on any computer without the need for any recompiling. Python is an implicit language, not needing to have its types defined when initially using a variable. However, when converting one variable to a specific type, Python is explicit such as in the following example:

x = “4”

y = int(x)

return y

These are just how the variables are defined, there are various different keywords and expressions used within Python functions to dictate what a program needs to do.

# Control Abstractions:

Python has thirty five different keywords used for its different purposes, from async to yield. Some are basic like the average true or false, while others are for more complex processes like await. Await is used in an asynchronous function to specify when to return to the loop that registers asynchronous tasks and functions to be executed. Python mostly uses the basic operators and expressions like +,-, and other calculator functions. However there are also bitwise operators in Python that can either shift the bits of binary number left or right, or and and two binary numbers together, and invert the bits within a given binary value.

These operators can be used in Python’s three different control structures: Conditional Statements, Loops, and Exception Handling. While these are all basic concepts, the syntax for how they’re used slightly differs from other programming languages. Else ifs are defined as elif and for loops run through a range rather than constantly looping until a condition is met. There are two different types of parameters that can be passed through these conditionals, keyword and positional. A keyword-only argument uses parameter names to pass the argument during the function call.

A positional argument uses arguments that are passed in the order of parameters. The order of the parameters are defined in the order function declaration. While loops are still conditional, but for loops require the breaking conditional to be placed within the loop itself. These are both examples of the iteration within Python, definitive iteration and Indefinite Iteration. Definitive iteration has a set number of repetitions while indefinite constantly executes until a condition or exception is met. Some languages handle their function arguments as references to variables that already exist, which is known as pass by reference. Other languages handle them as independent values, known as pass by value. Python has its own unique way of handling its function arguments.

Python passes its values by assignment, meaning that when you call a function in Python each function becomes a variable that the passed value is assigned to. If the assignment target is a name, like x, then this name is bound to an object like the number 2 or the string “word”. If the same has already been bound like before, then it’s rebound to the new object if we use x = 3 after the previous binding. Exception handling has its own keywords like as well like raise, finally, and except. Despite being neither pass-by-reference nor a pass-by-value language, Python isn’t any more difficult than languages that use either of those two function arguments. Because of its object oriented nature and flexibility it is more than capable of functioning in any desired way.

# Paradigms:

Python is an object-oriented programming language, almost everything from classes to functions are considered to be an object with distinct properties. Python also supports two other paradigms: Procedure Oriented and Functional Programming, which we’ll delve more into later. The object oriented style of programming allows the concept of inheritance to represent real world relationships easily as well as a large amount of reusability for the code. Instead of writing the same blocks of code over and over, they can be relegated to their own class which adds a form of modularity. The object oriented structure also allows for the usage of polymorphism, which is the characteristic of being able to assign a different meaning or usage to something in different contexts. In Python, polymorphism defines methods in the child classes that have the same name as the methods in a parent class. However this does add the issue that your code’s hierarchy can become a large spider web where an error in one class ripples out and crashes large chunks of the program.

Procedural Programming is a programming model derived from structural programming and based off of the concept of calling a procedure. Procedures, routines, subroutines or functions consist of a series of steps that need to be carried out. During a program’s execution, a procedure can be called at any point by other procedures or even itself. Within Python, tasks are treated as step-by-step chunks that are called when needed. This relates to the previously mentioned modularity that Python implements.

With Functional Programming, every statement is treated as an equation where any forms of state are avoided. State is anything we are asking the computer to remember in a program. Python uses Functional Programming by including the core concepts of functional programming: pure functions, recursion, first-class functions, and immutable variables. Pure functions have two properties: it always creates the same output from the and immutability, the act of not changing or modifying the input variable. For example: if you input a list into a for loop and create a new list by squaring each value, the resulting new list will always be the same as long as the initial input of the loop is the same. Like in most programming languages, you can recursively call a function within itself instead of running a for or while loop. Lambdas are used in this language to create simple expressions in fewer lines of code. For example, instead of:

def f(x):

return x \* 2

f(3)

We can simply use a lambda expression and do in one line what we did in three.

(lambda x : x \* 2)(x)

First-class functions are a function that is an instance of an object, can be stored in a variable, and can pass the function as a parameter to another function. Python has implemented some commonly used first-class and higher-order functions such as .map(), .filter(), and any other basic lambda functions. Finally, immutability is a functional programming paradigm that can be used for debugging as it throws an error where the variable being changed instead of the value. Some examples of immutable data types in Python include strings, tuples, and numerics.

Within Python an abstract class is used to define a collection of methods that are required for all subclasses that are derived from the abstract class. It’s a sort of basic requirements list of resources that the child classes need to use. Abstract types generally aren't used in dynamically typed languages unless they’re more object oriented like Python. They also include one or more abstract methods, a method that has a declaration but no implementation. These kinds of classes are used to create a standard interface for various implementations of any of the components within a program. Python has a module called ABC that serves as the foundation for building Abstract Base Classes. ABC adds a keyword @abstractmethod that denotes a class as abstract and then registers the other classes as implementations of the abstract base.

# Distinctive Features:

The most distinctive features of Python are its easy writability, easy readability, and its extensive standard library. We’ve talked about most of the other distinctive features previously such as it being interpreted, dynamically typed, and its high portability. Python is a very high-level programming language, which is why it’s often recommended as a first programming language. Its syntax is more basic compared to other languages like C,C++ or Java. Python code doesn’t use brackets, semicolons, and the indentations define the code block. Lastly, because of the extensive standard library included with Python programmers simply need to import certain tasks and functions rather than coding every single task. There are libraries for image manipulation, databases, unit testing, expressions and a lot of other functions.

# Overall Evaluation:

Python is one of the most recommended programming languages for a good reason. It’s simple, its syntax relying more on basic English words than more advanced syntax like Java or C++. Python is used as a scripting language to glue existing components together. Python includes an extensive standard library with tons of different modules. It’s like a box of Legos, providing the programmer with a wide variety of parts and pieces to build whatever kind of program they may need. The downside of using Python is based on how new users feel with its more unconventional style compared to other languages.

In Python, indentation and whitespace matters instead of using curly braces to define when each function begins and ends. There’s also Python’s slow speed and high memory usage to account for. This is due to being object oriented, as everything in Python is stored as an object and objects take up more memory than other variables. Another reason that Python is so well liked is its orthogonality, where a relatively small set of primitive constructs can be assembled to control different data structures within the language.

Bibliography:

*Python® – the language of today and Tomorrow*. About Python. (n.d.). Retrieved April 15, 2023, from <https://pythoninstitute.org/about-python>

GeeksforGeeks. (2022, November 26). *History of python*. GeeksforGeeks. Retrieved April 15, 2023, from <https://www.geeksforgeeks.org/history-of-python/>

Grover, J. (n.d.). *Perceiving python programming paradigms*. Opensource.com. Retrieved April 15, 2023, from <https://opensource.com/article/19/10/python-programming-paradigms>

Applications for python. (n.d.). Retrieved April 15, 2023, from <https://legacy.python.org/about/apps/>

Jaiswal, S. (2017, December 8). *Python data structures with Primitive & non-primitive examples*. DataCamp. Retrieved April 15, 2023, from <https://www.datacamp.com/tutorial/data-structures-python>

GeeksforGeeks. (2021, October 11). *Functional programming in python*. GeeksforGeeks. Retrieved April 17, 2023, from <https://www.geeksforgeeks.org/functional-programming-in-python/>

Team, G. L. (2022, October 31). *Abstraction in python*. Great Learning Blog: Free Resources What Matters to shape your Career! Retrieved April 17, 2023, from <https://www.mygreatlearning.com/blog/abstraction-in-python/>

Real Python. (2022, December 23). *Python keywords: An introduction*. Real Python. Retrieved April 17, 2023, from <https://realpython.com/python-keywords/#the-async-keyword>

*Python user defined Data Types*. GKIndex. (n.d.). Retrieved April 17, 2023, from <https://www.gkindex.com/python-tutorial/python-user-defined-datatypes.jsp>

*Type conversion in Python-Intellipaat*. Intellipaat Blog. (n.d.). Retrieved April 17, 2023, from <https://intellipaat.com/blog/tutorial/python-tutorial/type-conversion-in-python/>

Real Python. (2021, June 26). *Pass by reference in Python: Background and best practices*. Real Python. Retrieved April 17, 2023, from <https://realpython.com/python-pass-by-reference/#replicating-pass-by-reference-with-python>