Nicholas Hamakami

180D Wiki Article Initial Draft

2/9/23

Introduction:

Python is one of the most used programming languages today, with over 17 million users across the globe. However, as large as the user base of this language is, there are still many people who do not know the syntax, advantages, and disadvantages of using Python. That is why this article will give an overview of how to code the most basic Python scripts, as well as when and why to use Python.

History:

But before diving into some of the most basic syntax and structures in Python, it's important to give context as to how Python came to be. Firstly, Python was developed by the Dutch programmer Guido Van Rossam in 1991 due to his frustrations with the programming language ABC (Munro 2024). Additionally, when Python was being developed, some of its key features include it being an object-oriented language, a high-level language, and executable without the use of a compiler. For clarification, an object-oriented language is designed around objects, which are instances of a class that correspond to either real-world objects or abstract entities (Gillis). Additionally, a high-level programming language is one that is portable, maintainable, and easier for programmers to understand and debug, in exchange for programmers having less control over memory management and efficiency (Beal). From these qualities, it is clear that Python was developed to be a language with simplicity and a good user experience.

Syntax:

Now, diving into the syntax of Python, one key feature of Python that differentiates it from other languages is its use of indents as part of its syntax. Where other languages would use indents purely as a means of increasing user readability, indents are essential in Python, as they indicate where blocks of code begin and end. For example, after any if statement, loop, or function declaration, an indent must be made in order to show which code belongs to the respective statement (W3 Schools 2024).

Ex.

```
if 5 > 2:
  print("Five is greater than two!")
```

Another important aspect of Python are variable instantiations. To instantiate a variable, one should follow such a syntax:

```
var1 = 10
var2 = "Hello!"
```

There are a couple key pieces of information that should be noted from these variable instantiations. For one, a variable can be named any combination of alphanumeric characters (including "_"), as long as the variable name does not start with a number. Additionally, a variable can store any data type. Finally, one should note that a variable cannot be declared without a value (W3 Schools).

Another key aspect of Python is its ability to instantiate classes. These classes are key to the object-oriented functionality of Python code. To instantiate a class, one should follow the syntax:

```
class Animal:
  def __init__(self, species, name):
      self.species = species
      self.name = name
```

Again, there are a couple of important key points to take note of. For one, the __init__ function occurs every time a class is instantiated. Additionally, the “self” variable that is passed into __init__ represents an instance of the object itself. While other languages would hide this method, Python does not do that.

A variety of functionality can occur within the __init__ statement, but the most common functionality is to instantiate class variables that will be used later. It’s also important to note that Python does not have private variables like other languages do. However, if one were to unofficially declare that a variable is private, they would lead its name with an underscore “_”.

Additionally, Python classes can have methods in addition to variables. To define a new method in Python, do the following:

```
class Animal:
  def __init__(self, species, name):
      self.species = species
      self.name = name
  def get_name(self):
    print("This animal’s name is: " + self.name)
```

In order to access these methods and variables, one would follow a syntax like so:

```
cat = Animal(“cat”, 36)
print(cat.species)
cat.get_name()
```

As for how to actually run Python code, one can either execute the commands listed above directly in the command line, or by creating a .py file and running that file in the command line. If one were to create a .py file, the syntax would look like:

```
C:\path python myfile.py
```

Works Cited

Beal, V. (2022, July 14). *What is a high-level programming language?*. Webopedia. https://www.webopedia.com/definitions/high-level-language/

Gillis, A. S., & Lewis, S. (2021, July 13). *What is object-oriented programming (OOP)?*. App Architecture. https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP

Munro, A. (2024, January 11). *Python*. Encyclopædia Britannica. https://www.britannica.com/technology/Python-computer-language

W3Schools. (2024). *Python Syntax*. Python syntax. https://www.w3schools.com/python/python_syntax.asp