Assignment no 1

1. Who developed Python programming language?

Python was created by **Guido van Rossum**, and first released on February 20, 1991.

1. Which type of programming language Python supports?

Python is an interpreted programming language, supporting **object-oriented, structured, and functional programming**.

1. Is python case sensitive when dealing with identifiers?

Yes, Python is a case-sensitive language, i.e., it treats uppercase and lowercase characters differently. This applies to identifiers too. You must avoid using the same name with different cases while naming identifiers.

1. What is the correct extension of Python file?

The correct file extension for Python files is **.py**

1. Is python code compiled or interpreted?

**Python is an interpreted language**, which means the source code of a Python program is converted into bytecode that is then executed by the Python virtual machine.

1. Name a few blocks of code used to define in python language?

**Indentation** is used to define a block of code in python.

1. State a character used to give single line comment in python?

We can write a single-line comment by adding a single # character before any statement or line of code.

1. Mention function which can helps us to find the version of python that we are currently working on?

The function sys.version can help us to find the version of python that we are currently working on.

1. Python supports the creation of anonymous functions at runtime, using a construct called

Python supports to make anonymous functions which are not bound to a name(@ runtime). It uses a construct called "**lambda**"

1. what does pip stand for python?

PIP is a recursive acronym for “**Preferred Installer Program**” or PIP Installs Packages.

1. Mention a few built-in function in python?

|  |  |
| --- | --- |
| Function | Description |
| abs() | Returns the absolute value of a number |
| all() | Returns True if all items in an iterable object are true |
| any() | Returns True if any item in an iterable object is true |
| ascii() | Returns a readable version of an object. Replaces none-ascii characters with escape character |
| bin() | Returns the binary version of a number |
| bool() | Returns the boolean value of the specified object |
| bytearray() | Returns an array of bytes |
| bytes() | Returns a bytes object |
| callable() | Returns True if the specified object is callable, otherwise False |
| chr() | Returns a character from the specified Unicode code. |
| classmethod() | Converts a method into a class method |
| compile() | Returns the specified source as an object, ready to be executed |
| complex() | Returns a complex number |
| delattr() | Deletes the specified attribute (property or method) from the specified object |
| dict() | Returns a dictionary (Array) |
| dir() | Returns a list of the specified object's properties and methods |
| divmod() | Returns the quotient and the remainder when argument1 is divided by argument2 |
| enumerate() | Takes a collection (e.g. a tuple) and returns it as an enumerate object |
| eval() | Evaluates and executes an expression |
| exec() | Executes the specified code (or object) |
| filter() | Use a filter function to exclude items in an iterable object |
| float() | Returns a floating point number |
| format() | Formats a specified value |
| frozenset() | Returns a frozenset object |
| getattr() | Returns the value of the specified attribute (property or method) |
| globals() | Returns the current global symbol table as a dictionary |
| hasattr() | Returns True if the specified object has the specified attribute (property/method) |
| hash() | Returns the hash value of a specified object |
| help() | Executes the built-in help system |
| hex() | Converts a number into a hexadecimal value |
| id() | Returns the id of an object |
| input() | Allowing user input |
| int() | Returns an integer number |
| isinstance() | Returns True if a specified object is an instance of a specified object |
| issubclass() | Returns True if a specified class is a subclass of a specified object |
| iter() | Returns an iterator object |
| len() | Returns the length of an object |
| list() | Returns a list |
| locals() | Returns an updated dictionary of the current local symbol table |
| map() | Returns the specified iterator with the specified function applied to each item |
| max() | Returns the largest item in an iterable |
| memoryview() | Returns a memory view object |
| min() | Returns the smallest item in an iterable |
| next() | Returns the next item in an iterable |
| object() | Returns a new object |
| oct() | Converts a number into an octal |
| open() | Opens a file and returns a file object |
| ord() | Convert an integer representing the Unicode of the specified character |
| pow() | Returns the value of x to the power of y |
| print() | Prints to the standard output device |
| property() | Gets, sets, deletes a property |
| range() | Returns a sequence of numbers, starting from 0 and increments by 1 (by default) |
| repr() | Returns a readable version of an object |
| reversed() | Returns a reversed iterator |
| round() | Rounds a numbers |
| set() | Returns a new set object |
| setattr() | Sets an attribute (property/method) of an object |
| slice() | Returns a slice object |
| sorted() | Returns a sorted list |
| staticmethod() | Converts a method into a static method |
| str() | Returns a string object |
| sum() | Sums the items of an iterator |
| super() | Returns an object that represents the parent class |
| tuple() | Returns a tuple |
| type() | Returns the type of an object |
| vars() | Returns the \_\_dict\_\_ property of an object |
| zip() | Returns an iterator, from two or more iterators |

1. **What is the maximum possible length of an identifier in python?**

**An identifier can have a maximum length of 79 characters in Python.**

1. **What are the benefits of using python?**

* **Presence of third-party modules**
* **Extensive support libraries(NumPy for numerical calculations, Pandas for data analytics, etc.)**
* **Open source and large active community base**
* **Versatile, Easy to read, learn and write**
* **User-friendly data structures**
* **High-level language**
* **Dynamically typed language(No need to mention data type based on the value assigned, it takes data type)**
* **Object-Oriented and Procedural Programming language**
* **Portable and Interactive**
* **Ideal for prototypes – provide more functionality with less coding**
* **Highly Efficient(Python’s clean object-oriented design provides enhanced process control, and the language is equipped with excellent text processing and integration capabilities, as well as its own unit testing framework, which makes it more efficient.)**
* **Internet of Things(IoT) Opportunities**
* **Interpreted Language**
* **Portable across Operating systems**

1. **How is memory managed in python?**

**Memory management in Python involves a private heap containing all Python objects and data structures. The management of this private heap is ensured internally by the *Python memory manager*. The Python memory manager has different components which deal with various dynamic storage management aspects, like sharing, segmentation, preallocation or caching.**

1. **How to install python on windows and set path variables?**

**Installing Python on Windows takes a series of few easy steps.**

**Step 1 − Select Version of Python to Install**

**Python has various versions available with differences between the syntax and working of different versions of the language. We need to choose the version which we want to use or need. There are different versions of Python 2 and Python 3 available.**

**Step 2 − Download Python Executable Installer**

**On the web browser, in the official site of python (**[**www.python.org**](https://www.tutorialspoint.com/www.python.org)**), move to the Download for Windows section.**

**All the available versions of Python will be listed. Select the version required by you and click on Download. Let suppose, we chose the Python 3.9.1 version.**

**Table

Description automatically generated**

**On clicking download, various available executable installers shall be visible with different operating system specifications. Choose the installer which suits your system operating system and download the instlaller. Let suppose, we select the Windows installer(64 bits).**

**The download size is less than 30MB.**

**Graphical user interface, text, application, email, Teams

Description automatically generated**

**Step 3 − Run Executable Installer**

**We downloaded the Python 3.9.1 Windows 64 bit installer.**

**Run the installer. Make sure to select both the checkboxes at the bottom and then click Install New.**

**Graphical user interface, text, application

Description automatically generated**

**On clicking the Install Now, The installation process starts.**

**Graphical user interface, text, application

Description automatically generated**

**The installation process will take few minutes to complete and once the installation is successful, the following screen is displayed.**

**Graphical user interface, text, application

Description automatically generated**

**Step 4 − Verify Python is installed on Windows**

**To ensure if Python is succesfully installed on your system. Follow the given steps −**

* **Open the command prompt.**
* **Type ‘python’ and press enter.**
* **The version of the python which you have installed will be displayed if the python is successfully installed on your windows.**

**Graphical user interface, text

Description automatically generated**

**Step 5 − Verify Pip was installed**

**Pip is a powerful package management system for Python software packages. Thus, make sure that you have it installed.**

**To verify if pip was installed, follow the given steps −**

* **Open the command prompt.**
* **Enter pip –V to check if pip was installed.**
* **The following output appears if pip is installed successfully.**

**Text

Description automatically generated**

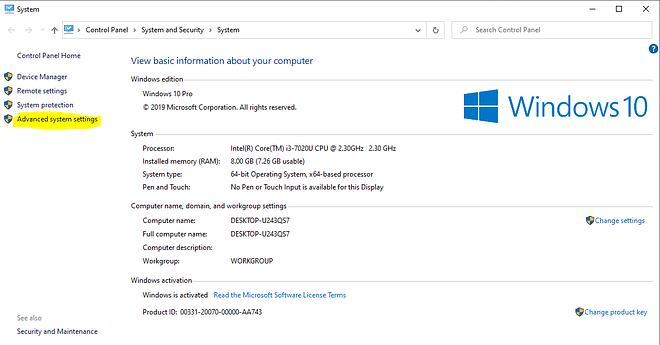
**We have successfully installed python and pip on our Windows system.**

**Setting the Python Environment Variable PYTHONPATH on Windows**

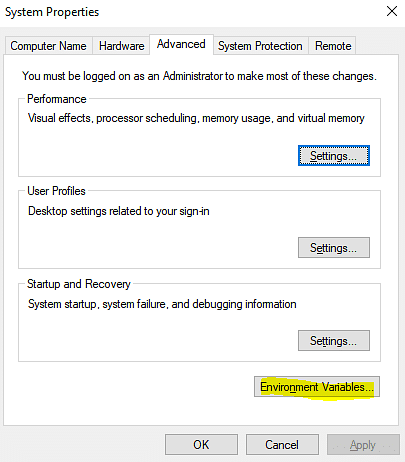
**Here is how to set PYTHONPATH on a windows machine:**

**Step 1: Open My Computer or This PC and right-click on it. Then click on properties.**

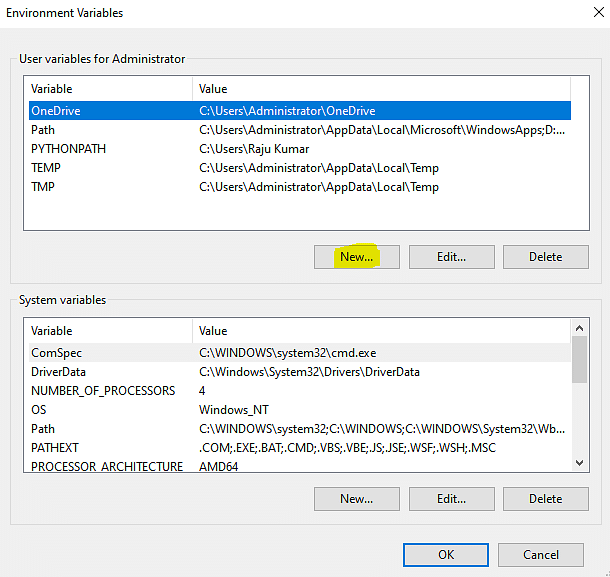
**Step 2: When the properties window pops up, click on the Advance System Settings.**

****

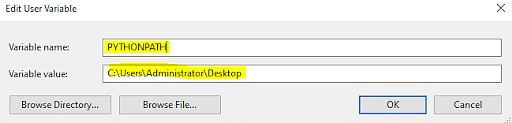
**Step 3: Click on the environment variable button that appears in the new popped-up window. Here is how it looks:**

****

**Step 4: In the dialog box- new Environment Variable, click on New.**

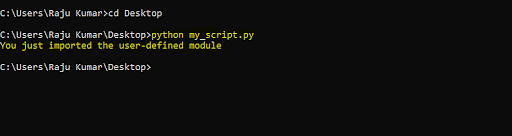
****

**Step 5: In the variable dialog box, add the variable's name as PYTHONPATH. Add the location that you want Python to check every time as a value to the module directory.**

****

**Step 6: Open the command prompt, execute the**[**python file**](https://www.simplilearn.com/tutorials/python-tutorial/how-to-open-a-file-in-python)**using the given command:**

**python my\_script.py**

****

1. **Is indentation required in python?**

**Indentation is mandatory in python to define the blocks of statements.**

**The number of spaces must be uniform in a block of code.**

**It is preferred to use whitespaces instead of tabs to indent in python. Also, either use whitespace or tabs to indent; intermixing of tabs and whitespaces in indentation can cause wrong indentation errors.**