**PYTHON NOTES FOR BEGINNERS**

In This Part We Will Learn Below Topics :-

**1) What Is Python ?**

**2) Why Is Python Popular ?**

**3) What Are The Features Of Python ?**

**4) Where Is Python Used In The Industry ?**

**5) What Are Comments In Python ?**

**6) What Are Variable And Data Types ?**

**1) What Is Python ?**

* Python Is A General Purpose **High Level, Interpreted Language** With Easy Syntax And Dynamic Semantics.
* Created By **Guido Van Rossum In 1989.**

**2) Why Is Python Popular ?**

* **Easy -** Python Has Easy Syntax.
* **Free -** It Is Open Source Programming Language And It Is Free For Anybody To Use Every Day.
* **Applications -** By Using Python You Can Develop Web Applications, AI, Machine Learning, GUI’s, Web Scripting Etc.
* **Library And Support -** Python Has A Huge Community To Support.

**3) What Are The Features Of Python ?**

* **Simplicity -** Python Has Simple Syntax You Enjoy When You Write Code In Python.
* **Open Source -** Python Is Free For Anybody To Use.
* **Portability -** Python Support Portability Means You Write Code And Share Anybody That You Want And It Would Work Same Way.
* **Interpreted -** Python Is Interpreted Line By Line Which Means Management Of Memory And CPU And Debugging Of Code Much More Simple.
* **Huge Libraries -** Python Has Huge Library Support Which Handles Your Problems Easier.
* **Object Orientation -** Python Supports Oops Concepts.

**4) Where Is Python Used In The Industry ?**

* **Google -** Better Search Results Are Provided Based On Ranking Of The Websites And Much More.
* **Dropbox -** The Server And Client Applications Are Coded Using Python.
* **Netflix -** Machine Learning Is Used To Cluster Based On Their Interest On Shows And Retain Them For Longer.
* **BitTorrent -** Transfer Of Files Between Peer To Peer Which Started Out As A Normal Python File.
* **NASA -** Scientific Calculations Are Computed Using Python.

**5) What Are Comments In Python ?**

* Comments In Python Are The Lines In The Code That Are Ignored By The Interpreter During The Execution Of The Program.
* Comments Enhance The Readability Of The Code And Help The Programmers To Understand The Code Very Carefully.
* Comments In Python Start With A **# Character.**
* There Are **Three Types** Of Comments In Python :-

**1) Single Line Comments :-**

* Python Single Line Comments Starts With The Hashtag Symbol # With No White Spaces And Lasts Till The End Of The Line.

**Example :-**

# print hello world

print(“hello world”)

**2) Multi Line Comments :-**

* We Can Multiple Hashtag # To Write Multiline Comments In Python. Each And Every Line Will Be Considered As A Single Line Comment.

**Example:-**

# multiline comment

# hey there

# hello python

print(“hello python”)

**3) Docstring Comments :-**

* Documentation Strings Written Within Triple Quotes (“”” “””) That Act A Comments.

**Example:-**

“””

Using Docstring As A Comment,

This Code Divides 2 Numbers

“””

x = 8

y = 4

z = x / y

print(z)

**6) What Are Variable And Data Types ?**

* Python Variable Is Containers Which Store Values. A Python Variable Is A Name Given To A Memory Location. It Is The Basic Unit Of Storage In A Program.

**Example :-**

X = 100

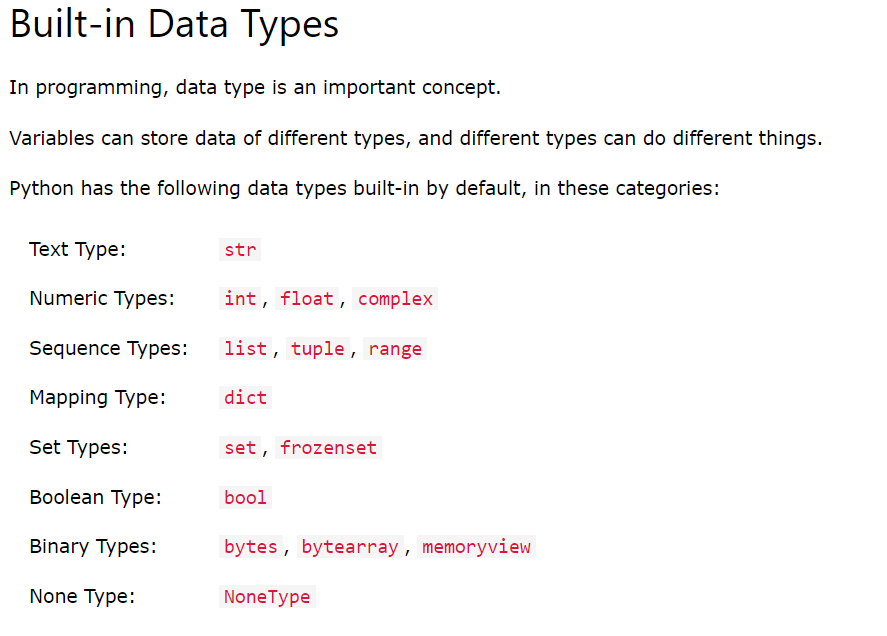
Y = “Python”

print(x)

print(y)

**Variable Data Types :-**

1. **Numbers**
2. **String**
3. **List**
4. **Dictionary**
5. **Tuple**
6. **Set**
7. **Range**

****

**Reference :** [**https://www.w3schools.com/python/python\_datatypes.asp**](https://www.w3schools.com/python/python_datatypes.asp)

[**https://realpython.com/lessons/sets-frozenset-multiset/#:~:text=A%20set%20is%20a%20collection,frozenset%20provides%20an%20immutable%20implementation**](https://realpython.com/lessons/sets-frozenset-multiset/#:~:text=A%20set%20is%20a%20collection,frozenset%20provides%20an%20immutable%20implementation)**.**

1. **Numbers :-**

* Numbers Have Mainly Four Data Types :-

**Integer :**

X = 10

**Float :**

X = 10.32

**Complex :**

X = 25j

**Boolean :- It Returns Only True And False.**

**NOTE :- To Check Type Of Variables Use type() Method.**

1. **String :-**

* A String Is A Collection Of One Or More Characters Put In A Single Quote, Double Quote Or Triple Quote.

**Example :-**

String1 = ‘Hello There !’

String2 = “Hello World !”

String3 = “”” Hello Python Using Me ! “””

print(String1)

print(String2)

print(String3)

1. **List :-**

* List Is An Ordered Collection Of Data. List Can Be Changed. List Allowed Duplicate Entries.
* List In Python Can Be Created By Just Placing The Sequence Inside The **Square Brackets [ ] .**

**Example :-**

Fruits = [‘apple’ , ‘kiwi’ , ‘banana’]

print(Fruits)

1. **Dictionary :-**

* Dictionary In Python Is An Unordered Collection Of Data Values. Can Be Changed And No Duplicate Entries Are Present.
* Dictionary holds the **key:value pair.**

**Example :-**

Animals = {

‘ reptiles ’ = ‘ snakes ’ ,

‘ mammals ’ = ‘ whale’ ,

‘ amphibians ’ = ‘ frogs ’

}

print(Animals)

1. **Tuple :-**

* Just Like List, Tuple Is Also An Ordered Collection Of Python Objects.
* The Only Difference Between Tuple And List Is That Tuples Are Immutable That Is Tuples Can Not Be Modified After It Is Created.
* In Python, tuples are created by placing a sequence of values separated by ‘comma’ with or without the use of parentheses for grouping of the data sequence. Tuples can contain any number of elements and of any datatype (like strings, integers, list, etc.).

**Example :-**

Animals = (‘ lion ’ , ‘ tiger ’ , 20 )

print(Animals)

1. **Set :-**

* In Python, Set is an unordered collection of data types that is iterable, mutable and has no duplicate elements.
* Sets Are Written With **{ } Curly** Brackets.

**Example :-**

Animals = { ‘ lion ’ , ‘ monkey ’ , ‘ snake ’}

print(Animals)

1. **Range :-**

* The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.
* **Syntax - range*(start, stop, step*)**
* **Example :-**

**x = range(3, 6)**

**for n in x:**

**print(n)**

Week 1: Introduction to Python

* Introduction to the Python programming language
* Setting up a development environment
* Basic data types (numbers, strings, lists, dictionaries)
* Control flow (if/else statements, for/while loops)
* Functions and modules

Week 2: Advanced Python

* Object-oriented programming in Python
* Error handling
* Reading and writing files
* Working with external libraries (e.g. NumPy, pandas)

Week 3: Introduction to Django

* Setting up a Django development environment
* Understanding the Model-View-Controller (MVC) architecture
* Creating a simple Django project
* Working with views and templates

Week 4: Advanced Django

* Working with models and the database
* Creating and managing forms
* Handling user authentication and authorization
* Implementing RESTful APIs with Django

Week 5: Building a Real-world Web Application with Django

* Building a complete web application with Django, including a database and user authentication
* Deploying the application to a hosting service
* Debugging and troubleshooting
* Maintaining and updating the application

Week 6: Extending and Refining

* Optimizing web performance with caching and database optimization
* Integrating with third-party services
* Best practices for security
* Using other popular libraries with Django
* Tips and tricks for debugging and troubleshooting

Week 1: Introduction to React

* Introduction to the React library
* Setting up a development environment
* JSX and components
* Props and state
* Handling events

Week 2: Advanced React

* Understanding the component lifecycle
* Managing component state
* Creating reusable components
* Integrating with external libraries (e.g. axios)
* Routing with React Router

Week 3: Introduction to ES6

* Understanding the modern JavaScript syntax
* Using let and const
* Arrow functions
* Destructuring
* Template literals

Week 4: Advanced ES6

* Understanding the concept of promises and async/await
* Iterators and generators
* Classes and inheritance
* Modules
* Spread and rest operators

Week 5: Building a Real-world Web Application with React and ES6

* Building a complete web application with React, including a router, and integrating with a backend
* Understanding best practices for performance and maintainability
* Deployment
* Debugging and troubleshooting

Week 6: Extending and Refining

* Optimizing performance with React hooks and Higher-Order Components
* Tips and tricks for debugging and troubleshooting
* Creating custom hooks
* Integrating with Redux and other state management tool