**Python Introduction:-**

\*Python was created by Guido van Rossum in 1989 (in the late eighties and early nineties) at the National Research Institute for Mathematics and Computer Science in the Netherlands.

\*The name python came from very popular comic show “Monty Python’s Flying Circus”

\* Python is a high-level, interpreted, interactive and object-oriented scripting language. (Is very easy to learn).

\*Python is an interpreter, object-oriented, high-level programming language with dynamic semantics.

**Who Uses Python?**

YouTube, Google, Dropbox, Netflix etc.

**Python features:-**

1. Simple and Easy to learn, read, & write.
2. Free and Open Source.
3. High-level Language.
4. Portable. (Supported by many platforms like Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, PlayStation etc.)
5. Supports different Programming Paradigm.
6. Extensible.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

* + It supports functional and structured programming methods as well as OOP.
  + It can be used as a scripting language or can be compiled to byte-code for building large applications.
  + It provides very high-level dynamic data types and supports dynamic type checking.
  + It supports automatic garbage collection.
  + It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**Operators in Python:-**

1. Arithmetic Operators
2. Assignment Operators
3. Comparison Operators
4. Logical Operators
5. Bitwise Operators
6. Identity Operators
7. Membership Operators

**Arithmetic Operators**

1. **~Negation**
2. **+ Addition**
3. **– Subtraction**
4. **\* Multiplication**
5. **/ Division**
6. **\*\* Exponential (power)**
7. **% Modulus (reminder)**
8. **// Truncating Division (Floor)**

**Assignment Operators**

1. **=**
2. +=
3. **-=**
4. **/=**
5. **%=**
6. **//=**
7. **\*\*=**

**Comparison Operators**

1. **>**
2. **<**
3. **==**
4. **!=**
5. **<=**
6. **>=**

**Logical Operators**

1. **AND**
2. **OR**
3. **NOT**

**Bitwise Operators**

1. **|(OR)**
2. **&(AND)**
3. **^(XOR**
4. **~(NOT)**
5. **a>>b (Shift a right by b bits)**
6. **a<<b (Shift a left by b bits)**

**Identity Operators**

1. **is**
2. **is not**

**Membership Operators**

1. **in**
2. **not in**

**Data type:-**

\*A data type, in programming, is a classification that specifies which type of values a variable has and what type of mathematical, relation or logical operation can be applied to it without causing an error.

\*Python is a loosely typed language. Therefore, no need to define the data type of variables.

\*No need to declare before using them.

**Immutable:-**

1. **Numbers**
2. **Strings**
3. **Tuples**

**Mutable:-**

1. **Lists**
2. **Dictionaries**
3. **Sets**

Numbers:-

1. Integer
2. Float
3. Complex
4. Long Type (3.X version only allowed)

Strings:-

1. Strings are sequences of one-character strings

sample= “Welcome to”

or

sample=’Welcome to’

1. Multi-line strings can be denoted using triple quotes,

‘’’We3c60eh24dfh’’’ or “””4xyfhcvgzdx1hv”””

1. **Sequences Operation:-**

* Concatenation (‘Python’ + ‘Tutorial’ -> ‘Python Tutorial)
* Repetition ( ‘Aamir’ \* 2 -> ‘AamirAamir’)
* Slicing ( string1=’Umair’ -> string1[1:5] -> ‘mair’)
* Indexing ( string1=’Umair’ -> string1[-1] + string[1] -> ‘rm’)

1. **Type Specific method:-**

* find():
* replace()
* split()
* count()
* upper()
* max()
* min
* isalpha()

Tuples:-

* A tuple is sequence of immutable Python object like floating number, string literals etc.
* The tuples can’t be changed unlike list.
* Tuples are defined using curve braces.
* **myTuple = {‘Edureka’, 2.4, 5, ’Python’}**
* **Sequences Operation:-**
* Concatenation [(tup=(’a’,’b’,’c’)-> tup+(‘d’,’f’)-> (‘a’,’b’,’c’,’d’,’f’)]
* Repetition [tup=(’a’,’b’,’c’)-> tup\*2-> (’a’,’b’,’c’,’a’,’b’,’c’)]
* Slicing [tup=(’a’,’b’,’c’)-> tup[1:2]->(‘b’,’c’)]
* Indexing (tup=(’a’,’b’,’c’)-> tup[0]->’a’)

List:-

* A list is sequence of mutable Python object like floating number, string, literals, etc.
* The lists can be modified
* Tuples are defined using square braces
* **myList = [‘Edureka’, 2.4, 5, ’Python’]**
* **Sequences Operation are similar to tuples**
* Type Specific Method:-

1. append(value)
2. extend(list)
3. insert(index, value)
4. pop() & reverse()
5. remove(value) & sort()

Dictionaries:-

* Dictionaries are perhaps the most flexible built-in data type in python
* Dictionaries, items are stored and fetched by key, instead, of by positional offset
* myDict ={1:’John’,2:’Bob’,3:’James’}
* Dictionary Methods:-

1. Accessing Dictionary
2. len()
3. key()
4. values()
5. items()
6. get()
7. update()
8. pop()

Sets:-

* A set is an unordered collection of items
* Every element is unique (no duplicates) and must be immutable ( which can’t be changed)
* **mySet = {1,2,3}**
* Sets Methods:-
* Creating set
* Union
* Intersection
* difference

**Flow Control:-**

**Function in Python:-**

\*A function is a block of organized, reusable sets of instruction that is used to perform some related actions.

\*Function is two types:-

1. Built-in Function 2. User Defined Function

\*Why do we use functions?

🡪Re-usability of code minimizes redundancy

🡪Procedural decomposition makes things organized

**Python Variable:-**

Variable are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Class: - A class is the blueprint from which specific object are created.

Syntax- class Class\_Name:

statement-1

:

:

Statement-N

**Class**

Class Inheritance:-

\*A class can inherit attributes and behavior methods from another class, called superclass.

\*A class which inherits from a superclass is called a subclass, also called heir class or child class.

Abstract Class:-

\*Abstract Class cannot be instantiated.

\*It can only be inherited.

Objects: - Anything that has a state and behavior is object.