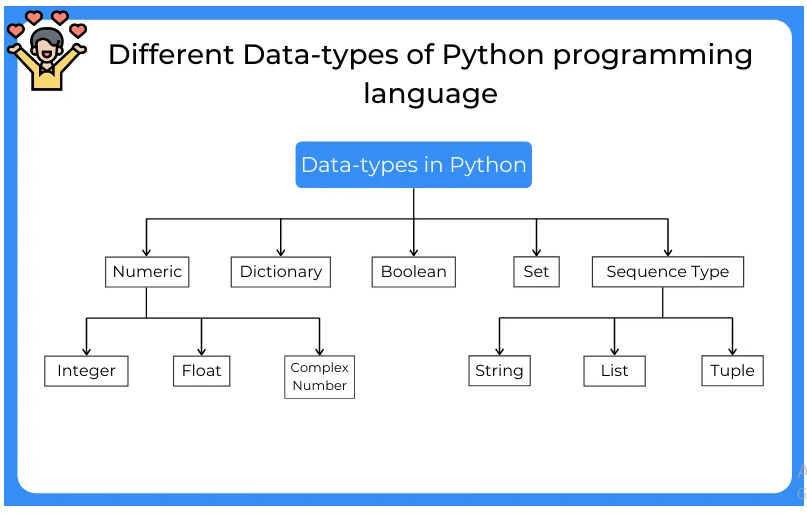
**Different Data-types of Python programming language**

1. Numeric:

Numeric Data-types are the data-type which represent the numeric values. It can be any number from the -infinity to +infinity. Numeric data types are one of them which can store values for an integer, Decimal values, Complex numbers, etc. Numeric data-types are subdivided into three types:

* Integer
* Float
* Complex Number

**Integer**

Integer data types are the one which can hold the integral value of the number. Integer data-types are the most common data types used in every programming language to deal with the operations performed on integer values. When we use loop statements to iterate through some list or any other data type, we use the integer values to get the indexes of the element.

#user input of integer value

var1 = int(input('Enter the integer value :'))

#initialize an integer variable

var2 = 2+56j

#print the integer variables

print(var1)

print(var2)

Output:

Enter the integer value :47

47

2+56j

**Float**

Float Data-types is one which holds the decimal values written after an integral part of the number along with an integral part. Float data-type is used to store accurate values. They can be used defined inputs or obtained by performing some operations. Float data-types can store values up to some exact digits and also can be formatted according to users need.

#user input of float value

var1 = float(input('Enter the float value :'))

#initialize an float variable

var2 = 23.123123

#print the float variables

print(var1)

print(var2)

Output:

Enter the float value :89.00012

89.00012

23.123123

**Complex Number**

Complex numbers are the data-type category that stores the complex numbers in a variable. Complex numbers are a combination of two different parts, one is a real part and another is the imaginary part. Complex numbers are written as Complex\_number = 5+89j. Complex numbers are used to solve advance calculus problems.

#user input of complex value

var1 = complex(input('Enter the complex value :'))

#initialize an complex variable

var2 = 2+56j

#print the complex variables

print(var1)

print(var2)

Output:

Enter the complex value : 45+6j

(45+6j)

(2+56j)

**2. Dictionary :**

Dictionaries are the unordered collection of data-types used to store the key-value pairs. Key in these dictionaries are the values that can be assigned to some data of any data-type, Whereas values are the information of the key. These key-value pairs can be accessed and used in many different ways.

**Creating Dictionaries**

Dictionaries in the Python Programming languages is the set of key holding some value to it and can be of any data-type. Dictionaries are created by placing the sequence of element inside the curly brackets {}, separated by commas. Values are the data stored in the key and can be duplicated, but keys are unique and duplicating keys are not allowed.

#creating dictionary

dictionary = dict({'Name': 'Divyanshu', 'Contact': 875597, 1: 'id\_no'})

#printing dictionary

print(dictionary)

print(dictionary.keys())

print(dictionary.values())

Output:

{'Name': 'Divyanshu', 'Contact': 875597, 1: 'id\_no'}

dict\_keys(['Name', 1,  'Contact'])

dict\_values(['Divyanshu', 875597, 'id\_no'])

**3. Boolean :**

Boolean data-type is used to store values with True and False. True value is stored the 1 and False stores the 0. Boolean data-types are used to check conditions. In conditional statements, boolean data-types are used to check if the value is correct or not. If the output is 1 or true it means the condition is satisfied and if the value is 0 or false it means the condition is not satisfied.

Boolean data-types are used to check the condition whether it is TRUE or FALSE. To check any condition we used if-else statements and loops. If the condition is True then the code inside the condition will get executed otherwise the code will stop and come out of the loop.

#Boolean in is-else conditions

if True:

print('I am always True')

else:

print('I am False')

#boolean in while loop

i = 1

while i == True:

print('I am True')

i = 0

Output:  
I am always True  
I am True

**4. Set :**

Sets are Data-type like the list. In this type of data type, we can store different values which are non-repeating. Sets remove the duplicates of the elements and rearrange them in an unpredictable arrangement. We cannot define the position of the element in the set but sets help find if the element is available in it or not.

Sets are used when we need to search if the item is available or not. It is the optimized version of the list in some ways. List and sets both can store values but in sets duplicate values are removed and operations are performed. Sets are also iterable and mutable and contain a huge set of elements.

#user defined set

Set = set([1,2,3,4,3,1,6])

print(Set)

#check for element

if 1 in Set:

print('True')

Output:

{1, 2, 3, 4, 6}

True

**5. Sequence :**

The sequence in a python programming language is a collection of data of same data-type or different data-type. In sequences, we can store items or elements and can efficiently perform operations. The sequence elements are used in performing various operations. The sequence is subdivided in three parts:

* List
* String
* Tuple

**List**

The list is a subpart of the datatype sequence. Lists are used to store values of same and different data types like integer, string, float, etc. The list is one of the most efficient and mutable data-type in the python programming language. The list can perform several operations like append the new elements, remove the item, reverse the elements, etc. All these functions are available in the python library.

#initializing a list

List = [1,2,3,4,2,1]

#print the list elements

print(List)

#Printing type of sequence

print(type(List))

Output:

[1, 2, 3, 4, 2, 1]

<class 'list'>

**Strings**

Strings in python programming is a subpart of sequence data type which contains character values. The string is a very-strong data-type used in python to perform operations. It is immutable data-type but has the property of iterations. We can perform many different operations on strings. In python, there is no datatype as a character, and a character is treated as the string of length 1.

#initializing an string

STRING = "Have a Good Day"

print(STRING)

#user defined strings

STR = str(input('Enter the string :'))

print(STR)

Output:

Have a Good Day

Enter the string :PrepInsta

PrepInsta

**Tuple**

Tuples belong to the data-type class classed as a sequence. Like List and set, a tuple can also store elements of same or different data-type. The tuple is an immutable collection of elements. we can perform operations on tuple but we cannot remove or append elements in the tuple. Tuples are hashable whereas lists are not. Tuples elements are enclosed in brackets ().

#initializing a tuple

Tuple = tuple([1,2,3,1,2])

print(Tuple)

#Printing type of sequence

print(type(Tuple))

Output:

(1, 2, 3, 1, 2)

<class 'tuple'>