Python data types:

Numeric

Integer 145

Float 12.0, 12.5

Complex number 2+3j

Boolean(True, False)

Dictionary {Key:value} = id:value

Set {} - set() – removes duplicates

Sequence type

Strings ‘ a’, ‘’a ‘’, ‘’’a ‘’’ - if, if else, elif , nested if

List [] – changes, has duplicates

Tuple() – cannot changed (immutable), duplicates

**Numeric:**

Integer <int> a = 5

Float <float> a = 5.0

Complex numbers <complex> a = 5+5j

**Boolean: (True, False),** it is used to check a condition is true or false.

The first letter is always Capital. Mostly used in Looping and conditional statements.

print(type(True))

print(type(False))

Dictionary:{1:”john”, 2:”elder”}

A dictionary is an unordered collection of values, used to store data values like a map, which unlike other data types which stores only value, In dictionary **Key:value\_pair.** The values in the **dictionary** can be of any data type and can be **duplicated,** Whereas the **keys** can’t be **replicated.**

It can be created by the function dict()

Dict = {} #Empty Dictionary.

Dict = {1:"welcome", 2:"to the", 3:"world", 4:"of python"} #dictionary using integer keys

Dict = {"name":"Rahul", 1:[45,32,86,95,83]} #dictionary created using mixed keys.

**#Dictionary created using keyword dict()**

Dict = dict({1:"welcome", 2:"john", 3:"woo"}) #dict using keyword dict and intkeys

Dict = dict([(1, 'Google'), (2, 'yahoo')]) #dict using kwd dict using a pair

Accessing elements from a dictionary.

Dict = {1:"hai", 2:"hello", 3:"raju"}

We can access by using either **keys** or by **get()**

Dict = {1:"hai", 4:"hello", 3:"raju", "kamal":"raj"}

print(dict)

print("Accessing the Dictioanry elements.")

print(Dict[1]) #array value indicates the position value.

print(Dict[4])

print(Dict.get(4)) #references by the key value.

print(Dict.get("kamal"))

Set:

It is an unordered collection of data type, that is iterable, mutable, no duplicates. Order of the elements is undefined , may consists of various elements.

Creating sets:

Sets can be created using the built-in set function, set()with an iterable objects or a sequence by placing the sequence inside curly braces separated by comma. Types of elements in a set need not to be the same, various mixed data type values can also be passed to the set.

Creating a set:

Set1 = set() #blank set

Set2 = set(“Elysium Academy”) #set using Strings

Set3 = set([“hai”, “hello”, “world”) #set created using list.

Set4 = set([1, 2, 3, “hai”, “hello”, “world”, 6, 7, 9])# set using mixed data type.

Accessing the elements of the set:

1. Using the loops (Generally for loops)
2. Checking an element in the loop

set4 = set(['hai', 'hello', 'world',1 , 2, 7, 8, 5])

print("Elements of the set are")

print(set4)

for i in set4:

print(i, end = "\n")

# To check the an element is present in the set

print("hai" in set4)

print("raju" in set4)

Sequence Type:

Sequence is the ordered collection of similar or different data types. Sequence allows to store multiple values in an organized and efficient fashion.

Strings

List

Tuple

Strings:

Strings are arrays of bytes representing Unicode characters. A string is a collection of one or more characters in a single quotes, double quotes or triple quotes. In python there is no character data type, a character is a string of length one. It is represented by str.

A = “hai hello world”

print("string data Type")

a = "hai hello world"

print(a)

a = 'hai hello world'

print(a)

a = """hai hello world"""

print(a)

a = """hai hello world

this is going to be fun"""

print(a)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| H | A | I |  | H | E | L | L | o |  |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| -10 | -9 | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 |

List:

Lists are just like arrays, ordered collection of data. It is very flexible as the items in the list cannot be of the same data type, and declared in square brackets.

Tuple:

Tuple is a collection of ordered set of Python objects, but these are immutable. Cannot be modified

Nestedif

Elif

For loop

Displaying elements from a list using for loop

String splicing , concatenation