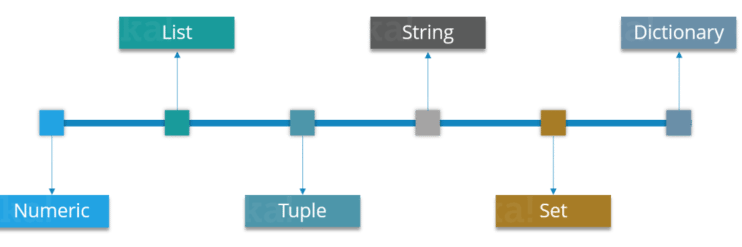
**Data Types in Python:**

Python supports various data types, these data types defines the operations possible on the variables and the storage method. Below is the list of standard data types available in Python:



Let’s discuss each of these in detail. In this Python tutorial, we’ll start with ‘Numeric’ data type.

**Numeric:**

Just as expected Numeric data types store numeric values. They are immutable data types, this means that you cannot change it’s value. Python supports three different Numeric data types:

**Integer type:** It holds all the integer values i.e. all the positive and negative whole numbers, example – 10.

**Float type:**It holds the real numbers and are represented by decimal and sometimes even scientific notations with E or e indicating the power of 10 (2.5e2 = 2.5 x 102 = 250), example – 10.24.

**Complex type:**These are of the form a + bj, where a and b are floats and J represents the square root of -1 (which is an imaginary number), example – 10+6j.

Now you can even perform type conversion. For example, you can convert the integer value to a float value and vice-versa. Consider the example below:

|  |  |
| --- | --- |
|  | A = 10  # Convert it into float type  B = float(A)  print(B) |

The code above will convert an integer value to a float type. Similarly you can convert a float value to integer type:

|  |  |
| --- | --- |
| 1  2  3  4 | A = 10.76  # Convert it into float type  B = int(A)  print(B) |

**List**:

* You can consider the Lists as Arrays in C, but in List you can store elements of different types, but in Array all the elements should of the same type.
* List is the most versatile data type available in Python which can be written as a list of comma-separated values (items) between square brackets. Consider the example below:

|  |  |
| --- | --- |
|  | Subjects = ['BMW', 'Benz', 'Audi', 2]  print(Subjects) |

Notice that the Subjects List contains both words as well as numbers. Now, let’s perform some operations on our Subjects List.

Let’s look at few operations that you can perform with Lists:

|  |  |  |
| --- | --- | --- |
| **Syntax** | **Result** | **Description** |
| Subjects [0] | BMW | This will give the index 0 value from the Subjects List. |
| Subjects [0:2] | BMW, Benz | This will give the index values from 0 till 2, but it won’t include 2 the Subjects List. |
| Subjects [3] = ‘Volvo’ | [‘BMW’, ‘Benz’, ‘Audi’, ‘Volvo’] | It will update the List and add ‘Biology’ at index 3 and remove 2. |
| del Subjects [2] | [‘BMW’, ‘Benz’,  2] | This will delete the index value 2 from Subjects List. |
| len (Subjects) | [‘BMW’, ‘Benz’, ‘Audi’, 2, 1, 2, 3] | This will return the length of the list |
| Subjects \* 2 | [‘BMW’, ‘Benz’, ‘Audi’, 2]  [‘BMW’, ‘Benz’, ‘Audi’, 2] | This will repeat the Subjects List twice. |
| Subjects [::-1] | [2, ‘Audi’, ‘Benz’, ‘BMW’] | This will reverse the Subjects List |

### ****Tuples:****

A Tuple is a sequence of immutable Python objects. Tuples are sequences, just like Lists. The differences between tuples and lists are:

* Tuples cannot be changed unlike lists
* Tuples use parentheses, whereas lists use square brackets. Consider the example below:

|  |  |
| --- | --- |
|  | Chelsea = ('Hazard', 'Lampard', 'Terry') |

Now you must be thinking why Tuples when we have Lists?

So the simple answer would be, Tuples are faster than Lists. If you’re defining a constant set of values which you just want to iterate, then use Tuple instead of a List.

Guys, all Tuple operations are similar to Lists, but you cannot update, delete or add an element to a Tuple.

Now, stop being lazy and don’t expect me to show all those operations, try it yourself.

Next in Python Tutorial, let’s understand Strings.

### ****Strings:****

Strings are amongst the most popular data types in Python. We can create them simply by enclosing characters in quotes. Python treats single and double quotes in exactly the same fashion. Consider the example below:

|  |  |
| --- | --- |
|  | S = "Welcome To brainovision"  D = 'Brainovision' |

|  |  |
| --- | --- |
| **Syntax** | **Operation** |
| print (len(String\_Name)) | String Length |
| print (String\_Name.index(“Char”)) | Locate a character in String |
| print (String\_Name.count(“Char”)) | Count the number of times a character is repeated in a String |
| print (String\_Name[Start:Stop]) | Slicing |
| print (String\_Name[::-1]) | Reverse a String |
| print (String\_Name.upper()) | Convert the letters in a String to upper-case |
| print (String\_Name.lower()) | Convert the letters in a String to lower-case |

### ****Set****:

* A Set is an unordered collection of items. Every element is unique.
* A Set is created by placing all the items (elements) inside curly braces {}, separated by comma. Consider the example below:

|  |  |
| --- | --- |
|  | Set\_1 = {1, 2, 3} |

In Sets, every element has to be unique. Try printing the below code:

|  |  |
| --- | --- |
|  | Set\_2 = {1, 2, 3, 3} |

Here 3 is repeated twice, but it will print it only once.

#### ****Union:****

Union of A and B is a set of all the elements from both sets. Union is performed using | operator. Consider the below example:

|  |  |
| --- | --- |
|  | A = {1, 2, 3, 4}  B = {3, 4, 5, 6}  print ( A | B) |

Output = {1, 2, 3, 4, 5, 6}

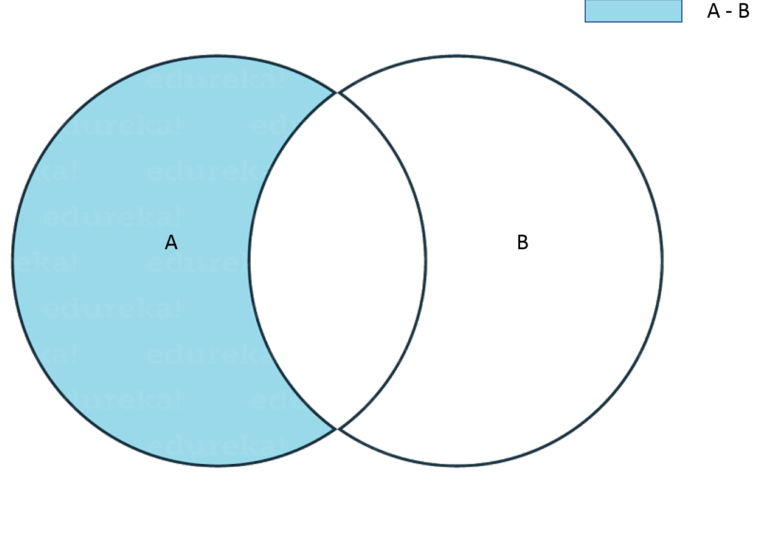
#### ****Intersection:****

Intersection of A and B is a set of elements that are common in both sets. Intersection is performed using & operator. Consider the example below:

|  |  |
| --- | --- |
|  | A = {1, 2, 3, 4}  B = {3, 4, 5, 6}  print ( A & B ) |

Output = {3, 4}

#### ****Difference:****

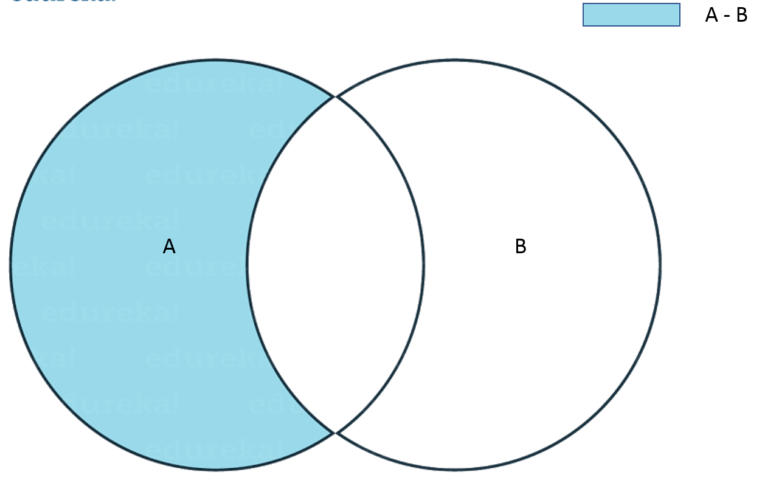


Difference of A and B (A – B) is a set of elements that are only in A but not in B. Similarly, B – A is a set of element in B but not in A. Consider the example below:

|  |  |
| --- | --- |
|  | A = {1, 2, 3, 4, 5}  B = {4, 5, 6, 7, 8}  print(A - B) |

Output = {1, 2, 3}

#### ****Symmetric Difference:****



Symmetric Difference of A and B is a set of elements in both A and B except those that are common in both. Symmetric difference is performed using ^ operator. Consider the example below:

|  |  |
| --- | --- |
|  | A = {1, 2, 3, 4, 5}  B = {4, 5, 6, 7, 8}  print(A ^ B) |

Output = {1, 2, 3, 6, 7, 8}

### ****Dictionary:****

Now let me explain you Dictionaries with an example.

I am guessing you guys know about Aadhaar Card. For those of you who don’t know what it is, it is nothing but a unique ID which has been given to all Indian citizen. So for every Adhaar number, there is a name and few other details attached.

Now you can consider the Adhaar number as a ‘Key’ and the person’s detail as the ‘Value’ attached to that Key.

Dictionaries contains these ‘Key Value’ pairs enclosed within curly braces and Keys and values are separated with ‘:’. Consider the below example:

|  |  |
| --- | --- |
|  | Dict = {'Name' : 'Saurabh', 'Age' : 23} |

You know the drill, now comes various Dictionary operations.

#### Access elements from a dictionary:

|  |  |
| --- | --- |
|  | Dict = {'Name' : 'Saurabh', 'Age' : 23}  print(Dict['Name']) |

Output = Saurabh

#### Changing elements in a Dictionary:

|  |  |
| --- | --- |
|  | Dict = {'Name' : 'Saurabh', 'Age' : 23}  Dict['Age'] = 32  Dict['Address'] = 'Starc Tower' |

Output = {'Name' = 'Saurabh', 'Age' = 32, 'Address' = 'Starc Tower'}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | List | Tuples | Set | Dictionary |
| Access List Item | print(list[1]) | print(tuple[1]) | print("banana" in thisset) | x= dict ["model"]  x=dict.get("model") |
| Negative Indexing | print(list[-1]) | print(tuple[-1]) | NA |  |
| Range of Indexes | print(list[2:5]) | Print(tuple[2:5]) | NA |  |
| Range of Negative Indexes | print(list[-4:-1]) | print(tuple[-4:-1]) | NA |  |
| Change Item value | list[1] = "blackcurrant" |  | NA |  |
| List Length | print(len(list)) | print(len(tuple)) | print(len(thisset)) | print(len(dict)) |
| Add Item | list.append("orange")  list.insert(1, "orange") | NA | thisset.add("orange")  thisset.update(["mango", "grapes"]) | dict["color"]="red" |
| Remove Item | list.remove("banana")  list.pop()  del list[0]  list.clear() | NA  del thistuple | set.remove("banana")  set.discard("banana")  set.clear()  del thisset | dict.pop("model")  dict.popitem()  del dict["model"]  del dict |
| Copy a List | mylist = list.copy()  mylist = list(thislist) | NA | NA | mydict = dict.copy()  mydict =dict(dict) |
| Join Two Lists | list3 = list1 + list2  list1.extend(list2) | tuple3 = tuple1 + tuple2 | set3 = set1.union(set2)  set1.update(set2) |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |