Data Types in Python

In programming, data type is an important concept. Variables can store data of different types, and different types can perform different operations. Every value in Python has a datatype. Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.

Python has the following data types built-in by default, in these categories:

A. List

1. A list is created by placing all the items (elements) inside square brackets [], separated by commas.
2. List items are ordered, changeable and allows duplicate values as well.
3. List items are indexed, the first item has index [0], the second item has index [1] etc.
4. It can have any number of items and they may be of different types (integer, float, string etc.).
5. A list can also have another list as an item. This is called a nested list.
6. We can access a range of items in a list by using the slicing operator :(colon).
7. We can also use + operator to combine two lists. This is also called concatenation.
8. The * operator repeats a list for the given number of times.

B. Tuple

1. A tuple is created by placing all the items (elements) inside parentheses (), separated by commas.
2. To create a tuple with only one item, you have to add a comma after the item, otherwise Python will not recognize it as a tuple.
3. Tuple can contain different data types.
4. We cannot change the elements of a tuple once it is assigned
5. Methods that add items or remove items are not available with tuple.
6. Since tuples are immutable, iterating through a tuple is faster.
7. Tuples can be used as a key for a dictionary.

C. Set

1. Since they are unordered, indexing has no meaning.
2. A set is created by placing all the items (elements) inside curly braces {}, separated by comma, or by using the built-in set() function.
3. Set items are unchangeable, and do not allow duplicate values.
4. However, a set itself is mutable. We can add or remove items from it.
5. A set can contain different data types.
6. Sets can also be used to perform mathematical set operations like union, intersection, symmetric difference, etc.
7. To make a set without any elements, we use the set() function without any argument.

D.Data Frame

1. Data Frame contains labeled axes (rows and columns).
2. Arithmetic operations align on both row and column labels.
3. Along with the data, you can optionally pass **index** (row labels) and **columns** (column labels) arguments.
4. Data Frame is a mutable, potentially heterogeneous tabular data structure.
5. It acts as a as a dictionary like container for Series objects.

E. Strings

1. Strings in python are surrounded by either single quotation marks, or double quotation marks.
2. Assigning a string to a variable is done with the variable name followed by an equal sign and the string.
3. You can assign a multiline string to a variable by using three quotes.
4. Strings in Python are arrays of bytes representing Unicode characters.
5. A single character is simply a string with a length of 1.
6. Square brackets can be used to access elements of the string.
7. Since strings are arrays, we can loop through the characters in a string, with a for loop.

F. Dictionaries

1. Dictionaries are used to store data values in key: value pairs.
2. Keys must be of immutable type and must be unique.
3. A dictionary is a collection which is unordered, changeable.
4. Dictionaries are written with curly brackets, and have keys and values.
5. Dictionary items are presented in key: value pairs, and can be referred to by using the key name.
6. If the key is already present, then the existing value gets updated.
7. While indexing is used with other data types to access values, a dictionary uses keys.
8. We can remove a particular item in a dictionary by using the pop() method.