Q. List datatypes and Tuples.

⇒ List :→

Lists are used to store multiple item in a single variable.

Lists are one of 4 built-in data types in python used to store collections of data, the other 3 are Tuple, Set and Dictionary; all with different qualities and usage.

Lists are created using square brackets:

Example :→

```
this list = ["apple", "banana", "cherry"]
print (thislist)
```

List Items :-

List items are ordered, changeable, and allow duplicate values.

List items are indexed, the first item has index [0], 2nd item has index [1] etc.

Ordered :→.

When we say that lists are ordered, it means that the items have a defined order, and that order will not change

If you add new items to a list, the new items will be placed at the end of the list.

Note:→ There are some list methods that will change the order, but in general: the order of the items will not change.

~~If yo~~

Changeable :→ List is changeable, meaning that we can change, add, and remove items in a list after it has been created.

Allow Duplicates

Since lists are indexed, lists can have items with the same value.

e.g:→
```
this list = ["apple", "banana", "cherry", "apple",
             "cherry"]
```

(3) List Length:→ To determine how many items a list has use the len() function.

e.g:→
```
this list = ["apple", "banana", "cherry"]
print (len (this list))
```
→ 3.

e.g→ String, int & boolean data types:-

```
lists1 = ["apple", "banana", "cherry"]
lists2 = [1,5,7,9,3]
lists3 = [True, False, False]
print(list1)
print(list2)
print(lists3)
```

⟹ Output:→ ['apple', 'banana', 'cherry']
[1,5,7,9,3]
[True, False, False]

④ A lists can contain different data types:-

A list with strings, integers and boolean values:-

```
lists1 = ["abc", 34, True, 40, "male"]
print(lists)
```

⟹ ['abc', 34, True, 40, 'male']

⑤ type() :→ From python perspective, lists are defined as objects with the data-type 'list':

<class 'list'>

e.g:

```
mylist = ["apple", "banana", "cherry"]
print(type(mylist))
```

⟹ <class 'list'>

* Namespace in Python.

- In Python, a namespace is a mapping from name to objects. Its essentially a system that ensures that all names in a program are unique and can be used without any conflict. Namespace are used to organize and control the visibility and accessibility of names (variables, functions, classes etc) in Python programs.

Types :→.

(i) Local namespaces. (local scope)

→ local namespaces when a function is called and immediately destroyed when the function exits.
→ It contains names of local variables, parameters ; and any name defined within the function.
→ Each function call creates a new local namespaces

(ii) Global Namespaces. (Global scope) :→

→ Global namespaces are created when a module is imported or when a script is executed. It contain names from various import modules, global variables functions, classes etc.

(iii). Built in Namespaces (built-in scope):-

→ Built in namespaces contain built in function exception and types are available to all modules by default.
- These names are always available in Python without needing to import anything.

Namespace Lookup :→.

When Python encounters a name in a script or function it searches for the name in the following order.

1. Local Namespace :- If the name is found in the local namespace (local variables, parameters) it is used.

2. Enclosing (non-local) function's local Namespace
- If the name is not found locally, Python searches the namespaces of the ending function (if any).

3. Global Namespace :- If the name is not found in the local or enclosing function namespaces, Python searches the global namespaces (module-level variables, function, classes).

4. Built in Namespaces :- If the name is still not found Python searches the built-in-namespace which contains built in function, exception and types.

Difference Between.

Q.	== operator	is operator
(i)	The == operator checks for equality of values between two objects.	The 'is' operator checks for identity between two objects.
(ii)	It compares the ~~memory~~ values ~~address~~ of the objects, meaning it checks if both objects ~~points to the same~~ on both side have the same content	It compares the memory address of the objects meaning it checks if both objects points to the same memory location.
(iii)	It is used to compare the values of variables or the content of data structure (like lists, dictionaries, etc).	It is used to determine if two variables reference the same object in memory.
(iv)	e.g.:- a = [1, 2, 3] b = [1, 2, 3] c = a print(a == b) # True print(a == c) # True.	e.g:- a = [1, 2, 5] b = [1, 2, 5] c = a print(a is b) # False bcoz a & b are different obj in memory print(a is c) # True, bcoz a & c point to same obj in memory

Difference Between.

	List	Tuple
(i)	List in Python are data structure that can hold a collection of items.	Tuple in Python are data structure that can hold a collection of items.
(ii)	List are mutable, i.e they can be changed or modified after creation. Elements can be added, removed or modified using various list methods.	Tuples are the immutable, i.e once they are created, their contents cannot be changed or ~~upload~~ updated. You cannot add, remove or modify elements in Tuple.
(iii)	Syntax :→ my_list = [1, 2, 3, 4, 5]	Syntax :→ my_tuple = (1, 2, 3, 4)
(iv)	Use lists when you have a coll^n of data that may need to be altered throughout the life of a program e.g.:- scores of students you might to update	Use Tuples when you have a collection of data that should not change such as a coll^n of constants or when data that shouldn't modified accidentally.

i) List consume more memory compared to tuples because of the overhead of the additional functionality (mutability).

ii) Lists are typically used for collection of homogeneous items where the order & number of items may change

Tuples are generally faster to access than lists, especially when iterating through them due to their immutable nature.

Tuples are used for heterogeneous collection where the structure (number and type of elements) should remain fixed.