# Training Report Day-2

## 7 June 2024

## LISTS IN PYTHON

Lists are used to store multiple items 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
#Create a List:
FRUITS = ["apple", "banana", "cherry", 24, 78.90, True, ["ram", "sham"]]
print(FRUITS)
```

## Characteristics of list:

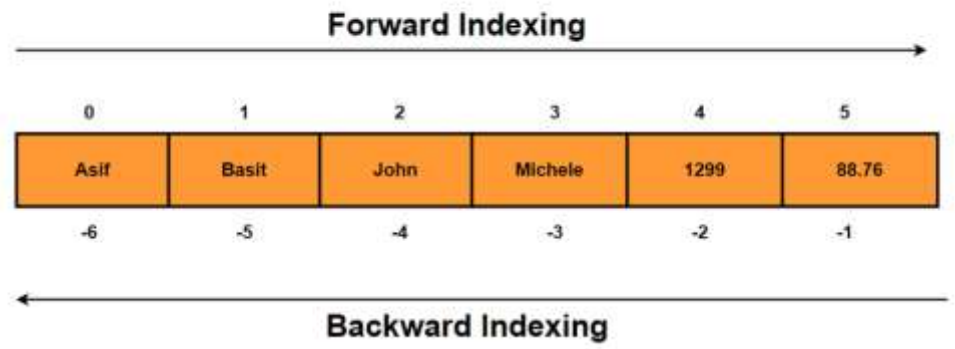
Ordered
Mutable
Heterogeneous
Nestable
Sliceable

```
#Example
#Lists allow duplicate values:

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

```
#Example
#Print the number of items in the list:

UNI = ["CTU", "KTU", "DTU"]
print(len(UNI))
```

```
#Example
#Using the list() constructor to make a List:
thislist = list(("apple", "banana", "cherry")) # note the double round-brackets
print(thislist)
```

List Indexing




## List methods:

```
#LIST METHODS
#APPEND: Used to new elements to the end of an existing list.
#syntax = list_name.append("item")

UNI = ["CTU", "KTU", "DTU"]
UNI.append("CU")
print(UNI)
```

```
#appending another list at the end of one list
# we can also append another list in an already existing list in order to make nested lists.
#syntax = list1_name.append(list2_name)
UNI = ["CTU", "KTU", "DTU"]
Courses = ["BCA", "MCA"]
UNI.append(Courses)
print(UNI)
```

```
#clear() method: It removes all the elements from the list, meaning makes the list empty.

#syntax = list_name.clear()
UNI = ["CTU", "KTU", "DTU"]
UNI.clear()
print(UNI)
```

```
#copy() method: It creates another copy of existing list.
UNI = ["CTU", "KTU", "DTU"]
print( UNI.copy())

colleges = UNI.copy()
print(colleges)
```

```
#count(): it counts the occurrence of particular element in a list
#syntax = list_name.count("element")
UNI = ["CTU", "KTU", "DTU", "CTU"]
UNI.count("CTU")
```

```
#extend() method: It is used to concatenate 2 existing lists().
UNI = ["CTU", "KTU", "DTU", "CTU"]
UNI2 = ["CU", "PU"]
UNI2.extend(UNI)
print(UNI2)
```

```
#remove() : It is also used to remove an element from an existing list like pop() method. But the difference is that it takes element itself as arguement.
FUR = ["TABLE", "CHAIR", "STOOL", "BED"]
FUR.remove("CHAIR")
FUR
```

```
#reverse method
FUR = ["TABLE", "CHAIR", "STOOL", "BED"]
FUR.reverse()
FUR
```

## List slicing:

List slicing in Python allows you to extract a portion of a list by specifying a range of indices**.**

## Syntax:

list[start:stop:step]

- **start**: The index where the slice begins (inclusive). If omitted, it defaults to the beginning of the list.
- **stop**: The index where the slice ends (exclusive). If omitted, it defaults to the end of the list.
- **step**: The step size, which determines how many items to skip between indices. If omitted, it defaults to 1.

```
#example 1
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
# Extracting a slice from index 2 to 5 (exclusive)
slice1 = numbers[2:6]
print(slice1)
```

```
#Example 2: Omitting start and stop
# Omitting start, defaults to the beginning of the list
slice2 = numbers[:4]
print(slice2)

# Omitting stop, defaults to the end of the list
slice3 = numbers[5:]
print(slice3)
```

```
#Example 3: Using a Negative Index
# Negative index, counts from the end of the list
slice4 = numbers[-5:]
print(slice4)

slice5 = numbers[:-3]
print(slice5)
```

```
#Example 4: Step Parameter

# Using step to get every second item
slice6 = numbers[::2]
print(slice6)
# Using step to get every third item
slice7 = numbers[1::3]
print(slice7)
```