Strings, List and Tuple

Session Objectives

- Understand string indexing and slicing
- Explore common string methods and operations
- Understand the meaning of data structures and why we use them.
- Common data structures in Python
- Understand what lists are.
- Understand common methods and operations associated with lists.
- Understand the meaning of list comprehension
- Understand what tuples are.
- Understand common methods and operations associated with tuples.
- Understand the Comparison between Lists and Tuples

```
# String Operations : 'Concatenation' '+'
str1 = 'Coding'
str2 = 'Ninjas'
new_string = str1 + ' ' + str2
print(new_string)
```

```
Coding Ninjas
```

```
name = 'Deepak'
age = 31
print(f"Hi, {name}. You are {age} year old.")
```

```
Hi, Deepak. You are 31 year old.
```

```
# TypeError: can only concatenate str (not "int") to str
name = 'Deepak'
age = 31
print('Hi' + name + ', You are ' + age + ' year old.')
```

```
name = 'Deepak'
age = 31
print('Hi ' + name + ', You are ' + str(age) + ' year old.')
```

```
Hi Deepak, You are 31 year old.
```

```
print(type(age)) # 'int'
```

```
<class 'int'>
```

```
# Repeat ('*')
_echo = 'Python\t'
print(_echo * 5)
```

```
Python  Python  Python  Python  Python
```

```python
# String Comparison
_str1 = 'Python'
_str2 = 'Java'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```

```python
# String Comparison
_str1 = 'Swim'
_str2 = 'Swimming'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # False
print(_str1 <= _str2) # True
```

```
False
True
False
True
```

```python
# String Comparison
_str1 = 'coding'
_str2 = 'Coding'
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```

```python
# String Comparison
_str1 = 'coding' # 99
_str2 = 'Zodiac' # 90
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```

```python
# String Comparison
_str1 = 'cODING' # 99
_str2 = 'Coding' # 67
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```

```python
# String Comparison
_str1 = 'swim' # 115
_str2 = 'Swimming' # 83
print(_str1 == _str2) # False
print(_str1 != _str2) # True
print(_str1 >= _str2) # True
print(_str1 <= _str2) # False
```

```
False
True
True
False
```

```python
# Common String Methods '. operator'
# 'replace'
_str = 'Python is Awesome'
new_str = _str.replace('Awesome','Fantastic')
print(new_str)
```

```
Python is Fantastic
```

```python
print(_str)
```

```
Python is Awesome
```

```python
_str = 'Python is Awesome'
new_str = _str.replace('o','O')
print(new_str)
```

```
PythOn is AwesOme
```

```python
# Split() -> 'Delimeter' '@'
_text = 'Indore@Madhya Pradesh@India'
print(_text.split('@')) # Split and store elements in a list
```

```
['Indore', 'Madhya Pradesh', 'India']
```

```python
topics = 'SQL*Python*Excel*PowerBI'
topic_list = topics.split('*')
print(topic_list)
```

```
['SQL', 'Python', 'Excel', 'PowerBI']
```

```python
topic_list[1] # ['Python']
```

```
'Python'
```

```python
split_str = "ViratKohli MSD RohitSharma Sachin GG"
split_str.split(" ")
```

```
['ViratKohli', 'MSD', 'RohitSharma', 'Sachin', 'GG']
```

```python
file_path = "http://localhost:8970/notebooks/anaconda_projects/CN-Python-Weekend/PythonSession.ipynb?"
path_list = file_path.split("/")
path_list
```

```
['http:',
 '',
 'localhost:8970',
 'notebooks',
 'anaconda_projects',
 'CN-Python-Weekend',
 'PythonSession.ipynb?']
```

```python
print(path_list)
```

```
['http:', '', 'localhost:8970', 'notebooks', 'anaconda_projects', 'CN-Python-Weekend', 'PythonSession.ipynb?']
```

```python
# String Formats -> .format [f_string]
name = 'Aryan'
age = 25
print('Hi, {}!.... You are {} year old.'.format(name, age))
print('Hi, {}!.... You are {} year old.'.format(age, name))
print(f'Hi, {name}!.... You are {age} year old.')
```

```
Hi, Aryan!.... You are 25 year old.
Hi, 25!.... You are Aryan year old.
Hi, Aryan!.... You are 25 year old.
```

```python
# strip() -> Same Like Trim [Trim Leading & Trailing Spaces]
print("      Python Programming        ".strip())
```

```
Python Programming
```

```python
print("********Python Programming**********".strip('*'))
```

```
Python Programming
```

```
print("********Python Programming".strip('*'))
```
```
Python Programming
```
```
print("Python Programming********".strip('*'))
```
```
Python Programming
```
```
print("$$$$$$$$$Python Programming********".strip('*$'))
```
```
Python Programming
```
```
print("$$$$$******$$$$Python Programming**$$$$$******".strip('*$'))
```
```
Python Programming
```
```
print("********Python           Programming**********".strip('* '))
```
```
Python           Programming
```
```
_strip_text = ("********Python           Programming**********".strip('* ').split())
print(_strip_text)
```
```
['Python', 'Programming']
```

```
# Common String Methods '. operator'
# 'replace'
_str = 'Python is Awesome'
new_str = _str.replace('Awesome','Fantastic')
print(new_str)
```
```
Python is Fantastic
```
```
print(_str)
```
```
Python is Awesome
```

Memory

```
_str = 'Python is Awesome'

new_str = _str.replace()
```

```
str1 = _strip_text[0]
str2 = _strip_text[1]
new_str = str1 + ' ' + str2
print(new_str)
```
```
Python Programming
```
```
# .index() -> position
_str = 'This is a Python Course!'
print(_str.index('is')) # First Occurence [2]
```
```
2
```
```
_str = 'This is a Python Course!' # ValueError: substring not found
print(_str.index('python')) # ValueError
```

```python
# 2nd Occurence['is']
_str = 'This is a Python Course!'
print(_str.index('is',3,10)) # .index(substr , start , stop)
print(_str.index('is',3)) # .index(substr , start , stop)
```

```
5
5
```

```python
# Dynamic
print(_str.index('is',_str.index('is') + 1)) # .index(substr , start , stop)
```

```
5
```

```python
# String Check() -> Returns Boolean Results
print("CodingNinjas".isalpha()) # True
print("Ninjas99".isalpha()) # False
print("Ninjas99".isalnum()) # True
print("CodingNinja".isupper()) # False
print("PYTHON".isupper()) # True
print("program".islower()) # True
print("892378432".isnumeric()) # True
print("coding@123".isnumeric()) # False
print("CoDinG NinJAs!007".swapcase()) # lower -> upper and vice-versa
```

```
True
False
True
False
True
True
True
False
cOdINg nINjaS!007
```

```python
# startwith() -> Boolean Returns
_str = "Hey! Welcome to the World of Programming!"
print(_str.startswith('hey')) # False
print(_str.startswith('Hey')) # True
print(_str.startswith('Hello')) # False
print(_str.startswith('Hi')) # False
print(_str.startswith('H')) # True
print(_str.startswith('Welcome' , 5)) # True
```

```
False
True
False
False
True
True
```

```python
# endswith() -> Boolean Returns
_str = "Hey! Welcome to the World of Programming!"
print(_str.endswith('hey')) # False
print(_str.endswith('Program')) # False
print(_str.endswith('Programming!')) # True
print(_str.endswith('!')) # True
print(_str.endswith('g!')) # True
print(_str.endswith('ing!',-4)) # True
print(_str.endswith('ing!')) # True
print(_str.endswith('ing!' , -3)) # False
```

```
False
False
True
True
True
True
True
False
```

```python
# .count() -> 'substring' -> Count the substring
_str = 'This is a Python Course!'
print(_str.count('is')) # 2
```

```
2
```

```python
_str = 'This is a Python Course!'
print(_str.count('python')) # 0 [Case Sensitive]
```

```
0
```

```python
_str = 'This is a Python Course!'
print('is' in _str) # Membership Operators
```

```
True
```

```python
_str = 'This is a Python Course!'
print('Java' in _str) # Membership Operators
```

```
False
```

What is a Data Structure?

A data structure is simply a way to organize and store data in memory, so that we can use it efficiently.

Why do we need to use them?

1. Organizations : Keep the data neat and easy to find.
2. Efficiency - Helps to perform operations (insert, delete, search) quickly.
3. Memory Management - Saves space, avoid waste.
4. Flexibility - Can store all sorts of data (number, text, etc.)
5. Abstraction - Hides the complex details, we just call simple operations
6. Scalability - Good Data Structure lets our program handle more data without slowing down.

Lists in Python: ¶

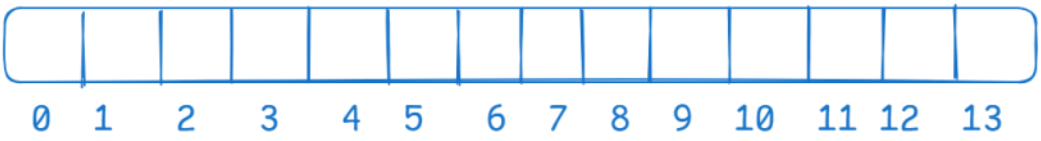
A List is :

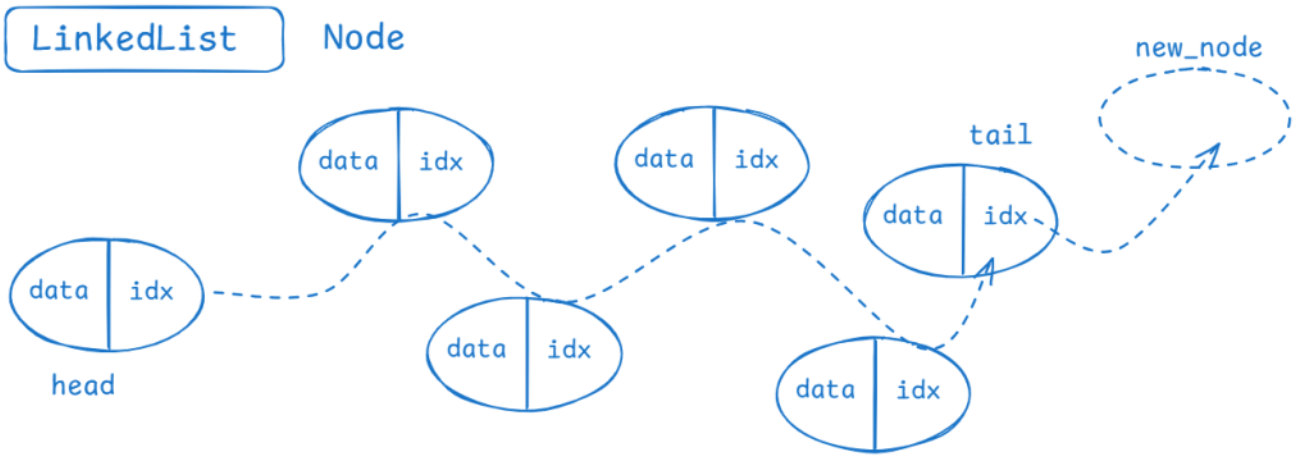
1. An ordered collection (items have a fixed position).
2. mutable (can be changed after creation).
3. Allows Duplicates.
4. Lists are also flexible -> they can store different data types together like integers, string, booleans, etc.

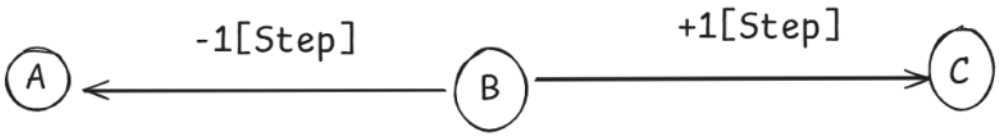
Why use lists?

- To keep a group of related items togethers in a specific order.
- To easily add, remove, change, or access the elements.
- To perform the task like sorting, or summing values.

What is an Array:







```python
_list = []
print(_list) # Empty List
print(type(_list)) # <'list'>
```

```
[]
<class 'list'>
```

```python
_list = [1,2,3,4,5] # Integer List
print(_list)
```

```
[1, 2, 3, 4, 5]
```

```python
_list = [7,9.99,True,"Coding",'k'] # Mixed Data Type
print(_list)
```

```
[7, 9.99, True, 'Coding', 'k']
```

```python
# Duplicate Elements are allowed in a List
country_list = ['India','Russia','America','China','Canada','Japan','Vietnam',
                'Norway','France','Finland','Sweden','Croatia','Australia',
                'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand']
print(country_list)
print(type(country_list))
```

```
['India', 'Russia', 'America', 'China', 'Canada', 'Japan', 'Vietnam', 'Norway', 'France', 'Finland', 'Sweden',
'Croatia', 'Australia', 'India', 'China', 'Pakistan', 'Japan', 'Iran', 'Singapore', 'New-Zealand']
<class 'list'>
```

```python
# List() Constructor
country_list = list(('India','Russia','America','China','Canada','Japan','Vietnam',
                'Norway','France','Finland','Sweden','Croatia','Australia',
                'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand'))
print(country_list)
print(type(country_list))
```

```
['India', 'Russia', 'America', 'China', 'Canada', 'Japan', 'Vietnam', 'Norway', 'France', 'Finland', 'Sweden',
'Croatia', 'Australia', 'India', 'China', 'Pakistan', 'Japan', 'Iran', 'Singapore', 'New-Zealand']
<class 'list'>
```

```python
# List ['String'] -> Extract the characters of a string and add them as an element
_list = list('Coding')
print(_list) # ['C','o','d','i','n','g']
```

```
['C', 'o', 'd', 'i', 'n', 'g']
```

```python
_list = list(('Coding',))
print(_list) # ['Coding']
```

```
['Coding']
```

```python
_list = list(['Coding'])
print(_list) # ['Coding']
```

```
['Coding']
```

```python
val = ('Python')
print(type(val)) # 'str'
```

```
<class 'str'>
```

```
val = ('Python',)
print(type(val)) # 'tuple'
```

```
<class 'tuple'>
```

```
val = ['Python']
print(type(val)) # 'list'
```

```
<class 'list'>
```

```
int('11') -> 11
int(11) -> 11
```

```
# Nested List -> A List Inside a List
nested_list = [
    111,
    'Coding',
    [11,22,33,44,55],
    'Python',
    ['a','b','c','d','e'],
    True
]
nested_list
```

```
[111,
 'Coding',
 [11, 22, 33, 44, 55],
 'Python',
 ['a', 'b', 'c', 'd', 'e'],
 True]
```

```
print(nested_list)
```

```
[111, 'Coding', [11, 22, 33, 44, 55], 'Python', ['a', 'b', 'c', 'd', 'e'], True]
```

```
# 2D Matrix [3][3][rows][cols]
print(nested_list[2][4]) # 55
```

```
55
```

```
print(nested_list[-1][0]) # TypeError: 'bool' object is not subscriptable
```

```
print(nested_list[-1]) # True
```

```
True
```

```
print(nested_list[2]) # [11, 22, 33, 44, 55]
```

```
[11, 22, 33, 44, 55]
```

```
print(nested_list[-2][-5]) # 'a'
```

```
a
```

```
print(nested_list[-2]) # ['a', 'b', 'c', 'd', 'e']
```

```
['a', 'b', 'c', 'd', 'e']
```

```
print(nested_list[-4][2]) # 33
```

```
33
```

```
print(nested_list[-2][5]) # IndexError: list index out of range
```

```
# [111, 'Coding', [11, 22, 33, 44, 55], 'Python', ['a', 'b', 'c', 'd', 'e'], True]
print(nested_list[-4][-1]) # 55
print(nested_list[-2][3]) # 'd'
print(nested_list[2][-5]) # 11
# print(nested_list[7]) # Error : Out of Range
```

```
55
d
11
```

```
country_list = list(('India','Russia','America','China','Canada','Japan','Vietnam',
               'Norway','France','Finland','Sweden','Croatia','Australia',
               'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand'))
print(country_list[-1]) # 'New-Zealand'
print(country_list[-9]) # 'Croatia'
print(country_list[5]) # 'Japan'
print(country_list[-15]) # 'Japan'
print(country_list[13]) # 'India'
print(country_list[-17]) # 'China'
```

```
New-Zealand
Croatia
Japan
Japan
India
China
```

```
# Slicing -> String [Substring]
# Slicing -> List [Sublist]
# Slicing [Start[0] : Stop[len][Non Inclusive] : Step[1][Gear:Forward/Reverse][Magnitude]]
```

```
country_list = list(('India','Russia','America','China','Canada','Japan','Vietnam',
              'Norway','France','Finland','Sweden','Croatia','Australia',
              'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand'))
print(country_list[::]) # [Full List]
```

```
['India', 'Russia', 'America', 'China', 'Canada', 'Japan', 'Vietnam', 'Norway', 'France', 'Finland', 'Sweden',
'Croatia', 'Australia', 'India', 'China', 'Pakistan', 'Japan', 'Iran', 'Singapore', 'New-Zealand']
```

```
print(country_list[2:5]) # ['America','China','Canada']
```

```
['America', 'China', 'Canada']
```

```
print(country_list[9:]) # ['Finland' to 'New-Zealand']
```

```
['Finland', 'Sweden', 'Croatia', 'Australia', 'India', 'China', 'Pakistan', 'Japan', 'Iran', 'Singapore', 'New
-Zealand']
```

```
print(country_list[-5:]) # ['Pak' to 'NZ']
```

```
['Pakistan', 'Japan', 'Iran', 'Singapore', 'New-Zealand']
```

```
print(country_list[-5::-1]) # ['Pak' to 'Ind'][Reverse]
```

```
['Pakistan', 'China', 'India', 'Australia', 'Croatia', 'Sweden', 'Finland', 'France', 'Norway', 'Vietnam', 'Ja
pan', 'Canada', 'China', 'America', 'Russia', 'India']
```

```
print(country_list[-9:2]) # ""
```

```
[]
```

```
print(country_list[-9:2:-1]) # ['Croatia' to 'China'][Reverse]
```

```
['Croatia', 'Sweden', 'Finland', 'France', 'Norway', 'Vietnam', 'Japan', 'Canada', 'China']
```

```
country_list = list(('India','Russia','America','China','Canada','Japan','Vietnam',
                'Norway','France','Finland','Sweden','Croatia','Australia',
                'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand'))
print(country_list[-4:-2]) # ['Japan','Iran']
print(country_list[-11:17:2]) # ['Fin','Cro','Ind','Pak']
```

```
['Japan', 'Iran']
['Finland', 'Croatia', 'India', 'Pakistan']
```

```
print(country_list[-9:2:-3]) # ['Croatia','France','Japan']
```

```
['Croatia', 'France', 'Japan']
```

```
print(country_list[-11:-4:5]) # ['Finland','China']
```

```
['Finland', 'China']
```

```
print(country_list[-8:4:3]) # []
```

```
[]
```

```
print(country_list[-17:21:4]) # ['China','Norway','Croatia','Pakistan','New-Zealand']
```

```
['China', 'Norway', 'Croatia', 'Pakistan', 'New-Zealand']
```

```
print(country_list[-17:3:4]) # []
```

```
[]
```

```
# Basic List Operations:
# len() -> returning the length of a list by calculating the number of elements in a list...
print(len(country_list))
```

```
20
```

```
# min() , max() , sum()
_num_list = [11,22,33,44,55,66,77,88,99,110]
print(len(_num_list)) # 10
print(min(_num_list)) # 11
print(max(_num_list)) # 110
print(sum(_num_list)) # 605
```

```
10
11
110
605
```

```
# min() , max() , sum()
_num_list = [11,22,33,44,55,66,77,88,99,110,False,True] # Mixed Type
print(len(_num_list)) # 12
print(min(_num_list)) # 0[False]
print(max(_num_list)) # 110
print(sum(_num_list)) # 605 + True[1] + False[0] = 606
```

```
12
False
110
606
```

```
print(int(False)) # 0
print(int(True)) # 1
```

```
0
1
```

```
country_list = list(('India','Russia','America','China','Canada','Japan','Vietnam',
               'Norway','France','Finland','Sweden','Croatia','Australia',
               'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand'))
print(len(country_list)) # 20
print(min(country_list)) # America
print(max(country_list)) # 'Vietnam'
```

```
20
America
Vietnam
```

```
country_list = list(('India','Russia','America','China','Canada','Japan','Vietnam',
               'Norway','France','Finland','Sweden','Croatia','australia',
               'India','China','Pakistan','Japan','Iran','Singapore','New-Zealand'))
print(len(country_list)) # 20
print(min(country_list)) # America
print(max(country_list)) # 'ASCII''a' > 'V' ['australia']
```

```
20
America
australia
```

```
_str = 'Hey, Let me know, how you gonna find the second occurence of me'
me_index = _str.index('me') # First Occurence [9]
print(me_index)
```

```
9
```

```
print(_str.count('me')) # 2
```

```
2
```

```
print('me' in _str) # True
```

```
True
```

```
_str.index('me' , me_index + 1) # .index(substr , start, stop)
```

```
61
```