```
In [3]: txt = " abc def ghi "
        txt.lstrip()

Out[3]: 'abc def ghi '
```

```
In [5]: txt = " abc def ghi "
        txt.strip()

Out[5]: 'abc def ghi'
```

# Using Escape Character

```
In [8]: #Using double quotes in the string is not allowed.
        mystr = "My favourite TV Series is "Game of Thrones""
```

```
  Cell In[8], line 2
    mystr = "My favourite TV Series is "Game of Thrones""
                                        ^

SyntaxError: invalid syntax
```

```
In [12]: #Using escape character to allow illegal characters
         mystr = "My favourite series is \"Game of Thrones\""
         print(mystr)

My favourite series is "Game of Thrones"
```

# List

1. List is an ordered sequence of items.
2. We can have different data types under a list. E.g we can have integer, float and string items in a same list.

# List Creation

```
In [24]: list1 = [] # Empty List
```

```
In [26]: print(type(list1))

<class 'list'>
```

```
In [28]: list2 = [10,30,60] # List of integers numbers
```

```
In [30]: list3 = [10.77,30.66,60.89] # List of float numbers
```

```
In [32]: list4 = ['one','two' , "three"] # List of strings
```

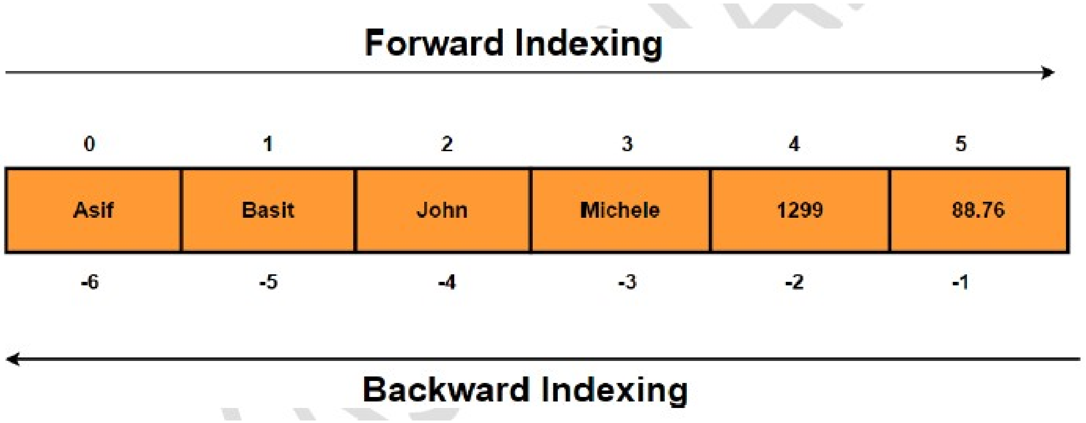
```
In [34]: list5 = ['Asif', 25 ,[50, 100],[150, 90]] # Nested Lists
```

```
In [36]: list6 = [100, 'Asif', 17.765] # List of mixed data types
In [38]: list7 = ['Asif', 25 ,[50, 100],[150, 90] , {'John' , 'David'}]
```

```
In [40]: len(list6) #Length of list
```

```
Out[40]: 3
```

# List Indexing



```
In [44]: list2[0] # Retreive first element of the list
```

```
Out[44]: 10
```

```
In [46]: list4[0] # Retreive first element of the list
```

```
Out[46]: 'one'
```

```
In [50]: list4[0][0] # Nested indexing - Access the first character of the first list ele
```

```
Out[50]: 'o'
```

```
In [52]: list4[-1] # Last item of the list
```

```
Out[52]: 'three'
```

```
In [54]: list5[-1] # Last item of the list
```

```
Out[54]: [150, 90]
```

# List Slicing

```
In [1]: mylist = ['one' , 'two' , 'three' , 'four' , 'five' , 'six' , 'seven' , 'eight']
```

```
In [3]: mylist[0:3] # Return all items from 0th to 3rd index location excluding the item
```

```
Out[3]: ['one', 'two', 'three']
```

```
In [5]: mylist[2:5] # List all items from 2nd to 5th index location excluding the item a
```

```
Out[5]: ['three', 'four', 'five']
```

```
In [7]: mylist[:3] # Return first three items
```

```
Out[7]: ['one', 'two', 'three']
```

```
In [9]: mylist[:2] # Return first two items
```

```
Out[9]: ['one', 'two']
```

```
In [11]: mylist[-3:] # Return last three items
```

```
Out[11]: ['six', 'seven', 'eight']
```

```
In [13]: mylist[-2:] # Return last two items
```

```
Out[13]: ['seven', 'eight']
```

```
In [15]: mylist[-1] # Return last item of the list
```

```
Out[15]: 'eight'
```

```
In [17]: mylist[:] # Return whole list
```

```
Out[17]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

# Add , Remove & Change Items

```
In [20]: mylist   # Return whole list
```

```
Out[20]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [22]: mylist.append('nine') # Add an item to the end of the list
         mylist
```

```
Out[22]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

```
In [24]: mylist.insert(9,'ten') # Add item at index location 9
         mylist
```

```
Out[24]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', 'ten']
```

```
In [26]: mylist.insert(1,'ONE') # Add item at index location 1
         mylist
```

```
Out[26]: ['one',
          'ONE',
          'two',
          'three',
          'four',
          'five',
          'six',
          'seven',
          'eight',
          'nine',
          'ten']
```

```
In [28]: mylist.remove('ONE') # Remove item "ONE"
         mylist
```

```
Out[28]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', 'ten']
```

```
In [30]: mylist.pop() # Remove last item of the list - Execute only once else it will rem
         mylist
```

```
Out[30]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

```
In [32]: mylist.pop(8) # Remove item at index location 8
         mylist
```

```
Out[32]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [34]: del mylist[7] # Remove item at index location 7
         mylist
```

```
Out[34]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven']
```

```
In [36]: # Change value of the string
         mylist[0] = 1
         mylist[1] = 2
         mylist[2] = 3
         mylist
```

```
Out[36]: [1, 2, 3, 'four', 'five', 'six', 'seven']
```

```
In [38]: mylist.clear() # Empty List / Delete all items in the list
         mylist
```

```
Out[38]: []
```

```
In [40]: del mylist # Delete the whole list
         mylist
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[40], line 2
      1 del mylist # Delete the whole list
----> 2 mylist

NameError: name 'mylist' is not defined
```

# Copy List

```
In [45]: mylist = ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine'
```

```
In [47]: mylist1 = mylist # Create a new reference "mylist1"
```

```
In [49]: id(mylist) , id(mylist1) # The address of both mylist & mylist1 will be the same
```

```
Out[49]: (2212437486720, 2212437486720)
```

```
In [51]: mylist2 = mylist.copy() # Create a copy of the list
```

```
In [53]: id(mylist2) # The address of mylist2 will be different from mylist because mylis
```

```
Out[53]: 2212396773056
```

```
In [55]: mylist[0] = 1
```

```
In [57]: mylist
```

```
Out[57]: [1, 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

```
In [59]: mylist1 # mylist1 will be also impacted as it is pointing to the same list
```

```
Out[59]: [1, 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

```
In [61]: mylist2 # Copy of list won't be impacted due to changes made on the original lis
```

```
Out[61]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine']
```

# Join Lists

```
In [64]: list1 = ['one', 'two', 'three', 'four']
         list2 = ['five', 'six', 'seven', 'eight']
```

```
In [66]: list3 = list1 + list2 # Join two lists by '+' operator
         list3
```

```
Out[66]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [68]: list1.extend(list2) #Append list2 with list1
         list1
```

```
Out[68]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

# List Membership

```
In [71]: list1
```

```
Out[71]: ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [73]: 'one' in list1 # Check if 'one' exist in the list
```

```
Out[73]:  True
```

```
In [75]:  'ten' in list1 # Check if 'ten' exist in the list
```

```
Out[75]:  False
```

```
In [81]:  if 'three' in list1: # Check if 'three' exist in the list
              print('Three is present in the list')
          else:
              print('Three is not present in the list')

          Three is present in the list
```

```
In [85]:  if 'eleven' in list1: # Check if 'eleven' exist in the list
              print('eleven is present in the list')
          else:
              print('eleven is not present in the list')

          eleven is not present in the list
```

# Reverse & Sort List

```
In [88]:  list1

Out[88]:  ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [90]:  list1.reverse() # Reverse the list
          list1

Out[90]:  ['eight', 'seven', 'six', 'five', 'four', 'three', 'two', 'one']
```

```
In [92]:  list1 = list1[::-1] # Reverse the list
          list1

Out[92]:  ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [94]:  list1 = list1[::] # Simply Prints the Copy of list
          list1

Out[94]:  ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [96]:  list1 = list1[:]  # Simply Prints the Copy of list as above
          list1

Out[96]:  ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [98]:  mylist3 = [9,5,2,99,12,88,34]
          mylist3.sort() # Sort list in ascending order
          mylist3

Out[98]:  [2, 5, 9, 12, 34, 88, 99]
```

```
In [100…  mylist3 = [9,5,2,99,12,88,34]
          mylist3.sort(reverse=True) # Sort list in descending order
          mylist3
```

```
Out[100…  [99, 88, 34, 12, 9, 5, 2]
```

```
In [102…  mylist4 = [88,65,33,21,11,98]
          sorted(mylist4)
          # Returns a new sorted list and doesn't change original list
```

```
Out[102…  [11, 21, 33, 65, 88, 98]
```

```
In [104…  mylist4  # Above step didn't changed mylist4
```

```
Out[104…  [88, 65, 33, 21, 11, 98]
```

# Loop through a list

```
In [107…  list1
```

```
Out[107…  ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight']
```

```
In [111…  for i in list1:
              print(i)
```

```
one
two
three
four
five
six
seven
eight
```

```
In [113…  for i in enumerate(list1):
              print(i)
```

```
(0, 'one')
(1, 'two')
(2, 'three')
(3, 'four')
(4, 'five')
(5, 'six')
(6, 'seven')
(7, 'eight')
```

# Count

```
In [116…  list10 =['one', 'two', 'three', 'four', 'one', 'one', 'two', 'three']
```

```
In [118…  list10.count('one') # Number of times item "one" occurred in the list.
```

```
Out[118…  3
```

```
In [120…  list10.count('two') # Occurence of item 'two' in the list
```

```
Out[120…  2
```

In [122…

```
list10.count('four') #Occurence of item 'four' in the list
```

Out[122…

```
1
```

# All / Any

The all() method returns:

- True - If all elements in a list are true
- False - If any element in a list is false

The any() function returns:

- True - if any element in the list is True.
- False - if not, any() returns False.

In [125…

```
L1 = [1,2,3,4,0]
```

In [127…

```
all(L1) # Will Return false as one value is false (Value 0)
```

Out[127…

```
False
```

In [129…

```
any(L1) # Will Return True as we have items in the list with True value
```

Out[129…

```
True
```

In [131…

```
L2 = [1,2,3,4,True,False]
```

In [133…

```
all(L2) # Returns false as one value is false
```

Out[133…

```
False
```

In [135…

```
any(L2) # Will Return True as we have items in the list with True value
```

Out[135…

```
True
```

In [137…

```
L3 = [1,2,3,True]
```

In [139…

```
all(L3) # Will return True as all items in the list are True
```

Out[139…

```
True
```

In [ ]:

```

```