20 MAY TASK

LIST DOCUMENT

In [9]:

```
txt = "    abc def ghi    "
txt.lstrip()
```

Out[9]:

```
'abc def ghi    '
```

In [30]:

```
txt = " abc def ghi "
txt.strip()
```

Out[30]:

```
'abc def ghi'
```

Using escape character

In [32]:

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

```
  Cell In[32], line 2
    mystr = "My favourite TV Series is "Game of Thrones""
                                        ^
SyntaxError: invalid syntax
```

In [34]:

```
#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 [47]:

```
list1 = [] # Empty List
```

In [49]:

```
print(type(list1))
```

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

In [51]:

```
list2 = [10,30,60] #List of integers numbrs
```

In [53]:

```
list3 = [10,77,30,66,60,89]   #List of float numbers
```

In [57]:

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

In [59]:

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

In [61]:

```
list6 = [100, 'Asif', 17.765] # List of mixed data types
```

In [63]:

```
list7 = ['Asif', 25 ,[50, 100],[150, 90] , {'John' , 'David'}]
```

In [65]:

```
len(list6) #Length of list
```

Out[65]:

```
3
```

In [67]:

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

Out[67]:

```
10
```

In [69]:

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

Out[69]:

```
'one'
```

In [71]:

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

Out[71]:

```
'o'
```

In [73]:

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

Out[73]:

```
'three'
```

In [77]:

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

Out[77]:

```
[150, 90]
```

List Slicing

In [80]:

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

In [82]:

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

Out[82]:

```
['one', 'two', 'three']
```

In [84]:

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

Out[84]:

```
['three', 'four', 'five']
```

In [86]:

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

Out[86]:

```
['one', 'two', 'three']
```

In [88]:

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

Out[88]:

```
['one', 'two']
```

In [90]:

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

Out[90]:

```
['six', 'seven', 'eight']
```

In [92]:

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

Out[92]:

```
['seven', 'eight']
```

In [94]:

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

Out[94]:

```
'eight'
```

In [96]:

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

Out[96]:

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

Add , Remove & Change Items

In [99]:

```
mylist
```

Out[99]:

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

In [101]:

```
mylist.append('nine') # Add an item to the end of the list
mylist
```

Out[101]:

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

In [103]:

```
mylist.insert(9,'ten') # Add item at index location 9
mylist
```

Out[103]:

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

In [105]:

```
mylist.insert(1,'ONE') # Add item at index location 1
mylist
```

Out[105]:

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

In [107]:

```
mylist.remove('ONE') # Remove item "ONE"
mylist
```

Out[107]:

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

In [109]:

```
mylist.pop() # Remove last item of the list
mylist
```

Out[109]:

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

In [111]:

```
mylist.pop(8) # Remove item at index location 8
mylist
```

Out[111]:

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

In [114]:

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

Out[114]:

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

In [116]:

```
# change the value of string
mylist[0] = 1
mylist[1] = 2
mylist[2] = 3
mylist
```

Out[116]:

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

In [118]:

```
mylist.clear() # Empty List / Delete all items in the list
mylist
```

Out[118]:

```
[]
```

In [120]:

```
del mylist # Delete the whole list
mylist
```

```
---------------------------------------------------------------------
----------
NameError                                 Traceback (most recent
call last)
Cell In[120], line 2
      1 del mylist # Delete the whole list
----> 2 mylist

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

Copy List

In [125]:

```
mylist = ['one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine'
```

In [127]:

```
mylist1 = mylist # Create a new reference "mylist1"
```

In [129]:

```
id(mylist) , id(mylist1) # The address of both mylist & mylist1 will be the same
```

Out[129]:

```
(2201910497664, 2201910497664)
```

In [131]:

```
mylist2 = mylist.copy() # Create a copy of the list
```

In [133]:

```
id(mylist2) # The address of mylist2 will be different from mylist because mylis
```

Out[133]:

```
2201910507840
```

In [135]:

```
mylist[0] = 1
```

In [137]:

```
mylist
```

Out[137]:

```
[1, 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nin
e']
```

In [139]:

```
mylist1 # mylist1 will be also impacted as it is pointing to the same list
```

Out[139]:

```
[1, 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nin
e']
```

In [141]:

```
mylist2 # Copy of list won't be impacted due to changes made on the original lis
```

Out[141]:

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

Join Lists

In [146]:

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

In [148]:

```
list3 = list1 + list2 # Join two lists by '+' operator
list3
```

Out[148]:

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

In [150]:

```
list1.extend(list2) #Append list2 with list1
list1
```

Out[150]:

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

List Membership

In [153]:

```
list1
```

Out[153]:

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

In [155]:

```
'one' in list1 # Check if 'one' exist in the list
```

Out[155]:

```
True
```

In [157]:

```
'ten' in list1 # Check if 'ten' exist in the list
```

Out[157]:

```
False
```

In [165]:

```
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')
```

```
  Cell In[165], line 2
    print('Three is present in the list')
    ^
IndentationError: expected an indented block after 'if' statement
on line 1
```

In [167]:

```
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')
```

```
  Cell In[167], line 2
    print('eleven is present in the list')
    ^
IndentationError: expected an indented block after 'if' statement
on line 1
```

Reverse & Sort List

In [170]:

```
list1
```

Out[170]:

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

In [172]:

```
list1.reverse() # Reverse the list
list1
```

Out[172]:

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

In [174]:

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

Out[174]:

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

In [180]:

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

Out[180]:

```
[2, 5, 9, 12, 34, 88, 99]
```

In [178]:

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

Out[178]:

```
[99, 88, 34, 12, 9, 5, 2]
```

In [182]:

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

Out[182]:

```
[11, 21, 33, 65, 88, 98]
```

In [184]:

```
mylist4
```

Out[184]:

```
[88, 65, 33, 21, 11, 98]
```

Loop through a list

In [187]:

```
list1
```

Out[187]:

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

In [191]:

```
for i in list1:
    print(i)
```

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

In [193]:

```
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 [198]:

```
list10 =['one', 'two', 'three', 'four', 'one', 'one', 'two', 'three']
```

In [200]:

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

Out[200]:

```
3
```

In [202]:

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

Out[202]:

```
2
```

In [204]:

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

Out[204]:

```
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. If not, any() returns False.

In [207]:

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

In [209]:

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

Out[209]:

False

In [211]:

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

Out[211]:

True

In [213]:

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

In [215]:

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

Out[215]:

False

In [217]:

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

Out[217]:

True

In [219]:

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

In [221]:

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

Out[221]:

```
True
```

In [ ]:

```
```