List Functions

There are a number of built-in functions that can be used with lists. In this notebook, we cover some of them.

1. `list()` : used to create lists when supplied with a string or any other iterable object as an argument.
2. `len()` : finding the number of elements in a list using the len() function
3. `range()` : used to generate a sequence of integers
4. `max()` : used to find the largest value in an iterable
5. `sum()` : used to find the sum of all elements in an iterable. Cannot be applied to sequences containing non-numeric values
6. `min()` : used to find the smallest value in a sequence
7. `in` and `not in` membership operators

Lists can be created from strings by using the `list()` function.

In [1]:

```
'''
Note the following:
  1.  When converting a string to a list, each character (including spaces) is considered a
separate element.
  2.  A list can have duplicate elements, that is, a list can have the same element repeated multi
ple times.
'''
my_new_lst = list("Creating a list from a string")
print(my_new_lst)
print(len(my_new_lst))
print(len("Creating a list from a string"))
```

```
['C', 'r', 'e', 'a', 't', 'i', 'n', 'g', ' ', 'a', ' ', 'l', 'i', 's', 't', ' ', 'f', 'r', 'o', 'm
', ' ', 'a', ' ', 's', 't', 'r', 'i', 'n', 'g']
29
29
```

In [2]:

```
'''
Lists can be created from strings by using the list() function.
Note the following:
  1.  When converting a string to a list, each character (including spaces) is considered a
separate element.
  2.  A list can have duplicate elements, that is, a list can have the same element repeated multi
ple times.
'''
my_new_lst = list(range(2, 18, 2))
print(my_new_lst)
print(len(my_new_lst))
```

```
[2, 4, 6, 8, 10, 12, 14, 16]
8
```

In [ ]:

```
'''
Another Example
'''
my_new_lst2 = list((2,3,4,5,6))
print(my_new_lst2)
print('The length of my_lst_2 is:',len(my_new_lst2))
```

In [4]:

```
'''
List traversal can be done in the following manner and is the recommended approach to traversing a
list.
'''
```

```
my_new_lst2 = ['5', 4, 3.3, 2, '1', True, [8,9], (5, 9)]
for ele in my_new_lst2:
    print(type(ele))
```

```
<class 'str'>
<class 'int'>
<class 'float'>
<class 'int'>
<class 'str'>
<class 'bool'>
<class 'list'>
<class 'tuple'>
```

In [6]:

```
'''
In the example below, we print out each element of a list.  If an element is itself a list, the in
dividual
elements of the list are printed out.
'''
my_lst = ['hello', 5, 19, [5,'mix']]
for ele in my_lst:
    if type(ele) == list:
        for i in ele:
            print(i, end=" ")
        print()
    else:
        print(ele)
```

```
hello
5
19
5 mix
```

You can find the number of elements in a list, using the `len()` function. You can then traverse the list using the `for` construct, together with the `range()` function as shown below. However, this is not the recommended approach since it is more ineficient than the approach discussed in the previous cell.

In [3]:

```
my_new_lst2 = list("2345678")

for ele in my_new_lst2:
    print(ele)

for i in range(len(my_new_lst2)):
    print(my_new_lst2[i])
```

```
2
3
4
5
6
7
8
2
3
4
5
6
7
8
```

To find the element in a list with the highest value, use the `max()` function. Note that the `max` function can be applied to both numeric as well as alpha numeric values but not to a list that contains a mix of both types.

In [4]:

```
my_new_lst = list("Creating a list from a string")
my_new_lst1 = [2,3,4,5,6,7,8]
```

```
my_new_lst2 = [2,3,4,5,6,7,8, 'asdsf']
print('The max of my_lst: ',max(my_new_lst))
print('The max of my_lst1: ',max(my_new_lst1))
print('The max of my_lst2: ',max(my_new_lst2))
```

```
The max of my_lst:  t
The max of my_lst1:  8
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-4-d038448956b2> in <module>
      4 print('The max of my_lst: ',max(my_new_lst))
      5 print('The max of my_lst1: ',max(my_new_lst1))
----> 6 print('The max of my_lst2: ',max(my_new_lst2))

TypeError: '>' not supported between instances of 'str' and 'int'
```

In [7]:

```
print(max([True, False]))
```

```
True
```

To find the element in a list with the smallest value, use the `min()` function. Note that the `min` function can be applied to both numeric as well as alpha numeric values. The space character is the smallest in the my_new_lst list.

In [5]:

```
my_new_lst = list("Creating a list from a string")
my_new_lst2 = [2,3,4,5,6,7,8]
my_new_lst3 = list("Creatingalistfromastring")
print('1. ', min(my_new_lst)) # This will display a space
print('2. ', min(my_new_lst2))
print('3. ', min(my_new_lst3))
```

```
1.
2.  2
3.  C
```

To find the total of the elements in a list, use the `sum()` function. The sum function cannot be applied to lists with non-numeric values

In [6]:

```
my_new_lst = list("Creating a list from a string")
my_new_lst2 = [2,3,4,5,6,7,8]

#print(sum(my_new_lst)) #This will result in an error
print(sum(my_new_lst2))
```

```
35
```

The `in` and `not in` operators, referred to as membership operators are used to specify whether or not an element is in a list.

In [2]:

```
my_list = [[10], 20, 30, 40, 50 ]
print(10 in my_list)
print(54 in my_list)

print(10 not in my_list)
print(54 not in my_list)
```

```
False
False
```

```
True
True
```