Python Data Structures

- List
- Tuple
- Dictionary
- Set
- Range

1. List

- List is a collection used to store multiple items/elements in a single variable
- List always use square brackets []

- Example:

In [7]:
```
azurevm_names = ['dev-vm1', 'prod-vm2']
print (azurevm_names)
```

```
['dev-vm1', 'prod-vm2']
```

In [8]:
```
print (type (azurevm_names))
```

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

In [9]:
```
print (len (azurevm_names))
```

```
2
```

1.1 append ()

- append () is a built-in list method used to add one new item to the end of an existing list.

- Example:

In [10]:
```
azurevm_names.append('qa-vm3')
```

In [11]:
```
print (azurevm_names)
```

```
['dev-vm1', 'prod-vm2', 'qa-vm3']
```

In [12]:
```
len (azurevm_names)
```

Out[12]:
```
3
```

1.2 copy ()

- copy () method is used to create a duplicate (independent) copy of a list.
- it makes new copy with the same items but changing one won't affect the other.

- Example:

```
In [15]: new_azurevms = azurevm_names.copy()
```

```
In [16]: print (new_azurevms)
```

```
['dev-vm1', 'prod-vm2', 'qa-vm3']
```

```
In [17]: azurevm_names == new_azurevms
```

```
Out[17]: True
```

Slicing

- colon : is a symbol used for slicing
- colon : helps us extract a specific part of a list - like cutting a piece from a cake.
- syntax >>> list [start:end:step]
-
    - start >> where the slice begins (index)
    - end >> where the slice stops (n-1)
    - step >> how many elements to skip each time

```
In [18]: M = [1, 2, 3, 4, 5, 6]
```

```
In [19]: print (M)
```

```
[1, 2, 3, 4, 5, 6]
```

```
In [20]: M
```

```
Out[20]: [1, 2, 3, 4, 5, 6]
```

```
In [21]: M [:]
```

```
Out[21]: [1, 2, 3, 4, 5, 6]
```

```
In [22]: M [0]
```

```
Out[22]: 1
```

```
In [23]: M [-1]
```

```
Out[23]: 6
```

```
In [24]: M [7]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
Cell In[24], line 1
----> 1 M [7]

IndexError: list index out of range
```

update element

```
In [25]: M
```

```
Out[25]: [1, 2, 3, 4, 5, 6]
```

```
In [26]: M [0] = 7
         M
```

```
Out[26]: [7, 2, 3, 4, 5, 6]
```

```
In [34]: M [-1] = 'Meera'
         M
```

```
Out[34]: [7, 2, 3, 4, 5, 'Meera', 'Vali', 'Meera']
```

```
In [35]: M = [1, 2, 3, 4, 5]
         M
```

```
Out[35]: [1, 2, 3, 4, 5]
```

```
In [36]: M.append('Meera')
         M
```

```
Out[36]: [1, 2, 3, 4, 5, 'Meera']
```

```
In [37]: M.append('Vali')
         M
```

```
Out[37]: [1, 2, 3, 4, 5, 'Meera', 'Vali']
```

Clear ()

- clear () method is used to remove all items from a list

```
In [39]: M.clear()
         M
```

```
Out[39]: []
```

1.3 count ()

- count () method is used to count how many times a specific value appears in a list.

```
In [42]: M = [1, 2, 3, 4, 5, 6]
```

```
In [43]: M
```

```
Out[43]: [1, 2, 3, 4, 5, 6]
```

```
In [44]: M.count(100)
```

```
Out[44]: 0
```

```
In [45]: M.count(2)
```

```
Out[45]: 1
```

List memebership

```
In [47]: M
```

```
Out[47]: [1, 2, 3, 4, 5, 6]
```

```
In [48]: 6 in M
```

```
Out[48]: True
```

```
In [49]: 7 in M
```

```
Out[49]: False
```

1.4 extend ()

- extend () method is used to add multiple elements (from another list) to the end of an existing list.

```
In [50]: M
```

```
Out[50]: [1, 2, 3, 4, 5, 6]
```

```
In [52]: N = [7, 8, 9]
         N
```

```
Out[52]: [7, 8, 9]
```

```
In [53]: N.extend(M)
         N
```

```
Out[53]: [7, 8, 9, 1, 2, 3, 4, 5, 6]
```

```
In [54]: N = [7, 8, 9]
         N
```

```
Out[54]: [7, 8, 9]
```

```
In [55]: M.extend(N)
         M

Out[55]: [1, 2, 3, 4, 5, 6, 7, 8, 9]

In [56]: M

Out[56]: [1, 2, 3, 4, 5, 6, 7, 8, 9]

In [57]: M [::]

Out[57]: [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Forward Index vs Forward Slicing

- Examples

```
In [ ]: - Element = 1 2 3 4 5 6 7 8 9
        - Index   = 0 1 2 3 4 5 6 7 8

In [60]: print (M[0])
         print (M[3])
         print (M[8])

1
4
9

In [61]: print (M[0:4])
         print (M[2:6])
         print (M[:5])
         print (M[5:])

[1, 2, 3, 4]
[3, 4, 5, 6]
[1, 2, 3, 4, 5]
[6, 7, 8, 9]
```

Backward Index vs Backward Slicing

- Examples

```
In [ ]: - Element =  1  2  3  4  5  6  7  8  9
        - Index   = -9 -8 -7 -6 -5 -4 -3 -2 -1

In [62]: M

Out[62]: [1, 2, 3, 4, 5, 6, 7, 8, 9]

In [63]: print(M[-1])
         print(M[-2])
         print (M [-5])
```

```
9
8
5
```

reverse the entire list [::-1]

```
In [64]: print (M[::-1])
```

```
[9, 8, 7, 6, 5, 4, 3, 2, 1]
```

```
In [65]: M
```

```
Out[65]: [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
In [67]: print (M[-1:-9:-3])
```

```
[9, 6, 3]
```

```
In [ ]:
```