Index

Index is a stored position of an object from ordered collections like string, list, tuple.

For Example, if we have a string "HELLO", we can access the first letter "H" using its index 0 by using the square bracket notation: string[0]

```
                              |----> positive indexing
                              |
   <-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|----->
        -5    -4    -3    -2    -1    0     1     2     3     4     5
                              |
        Negative indexing<----|
```

S. No	Negative indexing	Positive indexing
1	`-ve` indexing start with -1	`+ve` index start with 0
2	`-ve` indexing read from R to L	`+ve` index read from L to R
3	`-ve` indexing write from L to R	`+ve` indexing write from L to R
4	`-ve` indexing stop point in (stop+1)	`+ve` indexing stop point in (stop-1)

Python's built-in index() function is a useful tool for finding the index of a specific element in a sequence. This function takes an argument representing the value to search for and returns the index of the first occurrence of that value in the sequence.

If the value is not found in the sequence, the function raises a ValueError. For Example, if we have a list [1, 2, 3, 4, 5], we can find the index of the value 3 by calling list.index(3), which will return the value 2 (since 3 is the third element in the list, and indexing starts at 0).

```
l = [1, 2, 3, 4, 5]
print(l.index(3))

# Output
2
```

but if we want to findout 10 with the help of index(10), then it gives ValueError

```
l = [1, 2, 3, 4, 5]
print(l.index(10))

# Output
ERROR!
Traceback (most recent call last):
  File "<main.py>", line 2, in <module>
ValueError: 10 is not in list
```

The method index() returns the lowest index in the list where the element searched for appears. If any element which is not present is searched, it returns a ValueError.

```
list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
element = 3
print(list.index(element))

# Output
2
```

```
list = [1, 'two', 3, 4, 5, 6, 7, 8, 9, 10]
element = 'two'
print(list.index(element))

# Output
1
```

Find element with particular start and end point:--

Syntax:-

1. **iterator/collection.index(element, start, stop)**

2. **iterator/collection.index(element)**

3. **iterator/collection.index(element, start)**

Example :

```
list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
element = 7
print(list.index(element, 5, 8))

#output
6
```

Example :

```
list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
element = 5
print(list.index(element))


#output
4
```

Example :

```
list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
element = 5
print(list.index(element,5))


#output
ERROR!
Traceback (most recent call last):
  File "<main.py>", line 3, in <module>
ValueError: 5 is not in list
# Raise Error because it start from 5th position of index.
```