

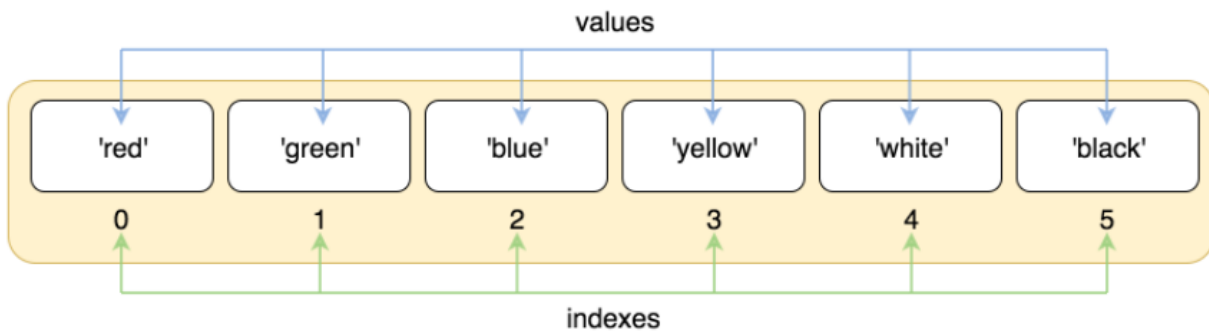
Tutorials on List structures, and Tuple

Indexing:

Let's take a simple example:

```
colors = ['red', 'green', 'blue', 'yellow', 'white', 'black']
```

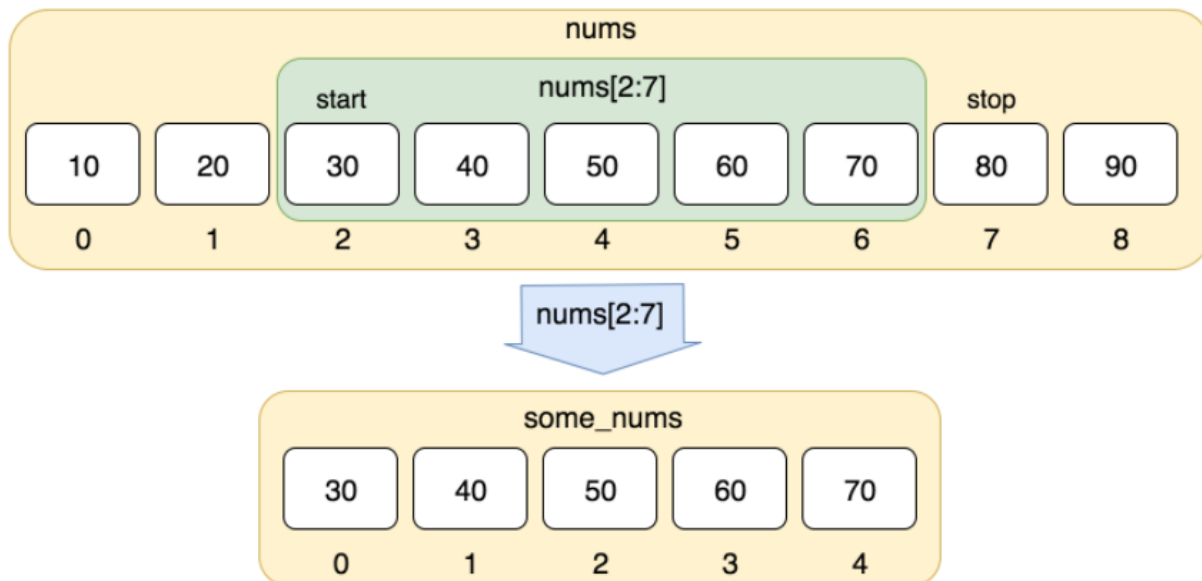
Python uses zero-based indexing. That means, the first element(value 'red') has an index 0, the second(value 'green') has index 1, and so on.



To address this requirement there is negative indexing. So, instead of using indexes from zero and above, we can use indexes from -1 and below.



In negative indexing system -1 corresponds to the last element of the list(value 'black'), -2 to the penultimate (value 'white'), and so on.

Slicing:

```
[start : stop : steps]
```

which means that slicing will start from index `start`
will go up to **stop** in **step** of steps.
Default value of `start` is 0, `stop` is last index of list
and for `step` it is 1


append() and extend() in Python

Last Updated: 01-04-2020


Append: Adds its argument as a single element to the end of a list. The length of the list increases by one.

syntax:

```
# Adds an object (a number, a string or a  
# another list) at the end of my_list  
my_list.append(object)
```




```
my_list = ['geeks', 'for']  
my_list.append('geeks')  
print my_list
```




Output:

```
['geeks', 'for', 'geeks']
```

NOTE: A list is an object. If you append another list onto a list, the parameter list will be a single object at the end of the list.



```
my_list = ['geeks', 'for', 'geeks']  
another_list = [6, 0, 4, 1]  
my_list.append(another_list)  
print my_list
```




Output:

```
['geeks', 'for', 'geeks', [6, 0, 4, 1]]
```

Tutorials on List structures, and Tuple

extend(): Iterates over its argument and adding each element to the list and extending the list. The length of the list increases by number of elements in it's argument.

syntax:
Each element of an iterable gets appended
to my_list
my_list.extend(iterable)




```
my_list = ['geeks', 'for']  
another_list = [6, 0, 4, 1]  
my_list.extend(another_list)  
print my_list
```

Output:

```
['geeks', 'for', 6, 0, 4, 1]
```

NOTE: A string is an iterable, so if you extend a list with a string, you'll append each character as you iterate over the string.



```
my_list = ['geeks', 'for', 6, 0, 4, 1]  
my_list.extend('geeks')  
print my_list
```

Output:

```
['geeks', 'for', 6, 0, 4, 1, 'g', 'e', 'e', 'k', 's']
```

Tuple

Tuples are used to store multiple items in a single variable.

Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Set, and Dictionary, all with different qualities and usage.

A tuple is a collection which is ordered and **unchangeable**.

Tuples are written with round brackets.

Example

Create a Tuple:

```
thistuple = ("apple", "banana", "cherry")  
print(thistuple)
```

Tutorials on List structures, and Tuple

1. Printing a empty list:

```
a=[]  
print(a)
```

2. Check index:

```
a=[10, 20, 30, 40]  
c=a[1:2]  
print(c)
```

3. Reverse a string

```
L = [1,2,3]  
print(list(reversed(L)))
```

4. Check any number

```
L = [1,2,3]  
print(5 in L)
```

5. Sort the list

```
L= [1,3,5,2,7,8]  
print(sorted(L))
```

6. Value at a index:

```
L= [1,3,5,2,7,8]  
print(L.index(5))
```

7. Pop the value

```
L= [1,2,3]  
L.pop()  
Print(L)
```

8. Copy the list:

```
a= [10,20,30,40]  
c=a  
print(c)
```

9. Pick the last index:

```
L= [1,2,3]  
print(L[-1])
```

10. I= (40,20,30,50)

```
print(I)  
  
print(I[0])
```

Tutorials on List structures, and Tuple

```
print(l.index(20))
```

```
print(sorted(l))
```

11. Predict the output:

```
x= "Bennett"
```

```
print(x[3])
```

```
x= ["I" , "Am" , "Ironman"]
```

```
print(x[2])
```

```
x= [["I", "Am"] , ["Ironman"]]
```

```
print(x[0][1][1])
```

```
print(x[1][0])
```

12. Predict the output:

```
List = [['Python ', 'is' ], ['Easy']]
```

```
print("\nValue in Multi-Dimensional List: ")
```

```
print(List[0][0], List[1][0][0])
```

```
List = [1, 2, 'Python', 4, 'is', 6, 'Easy']
```

```
print("\nList with the use of Mixed Values: ")
```

```
print(List[1], List[2][2])
```

13. Predict The Output:

```
List = ["Bennett"]
```

```
List.append("University")
```

```
print(List)
```

```
List.extend([ 'First', 'year'])
```

```
print(List)
```

Tutorials on List structures, and Tuple

```
List.insert(2, "CSE")
```

```
print(List)
```

14. Predict the output

```
List = [1, 2, 3, 9, 5, 6, 4, 7, 8, 7, 10, 11, 12]
```

```
List.remove(7)
```

```
print(List)
```

```
for i in range(3, 5):
```

```
    List.remove(i)
```

```
print(List)
```

```
List = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
```

```
List.remove(7)
```

```
print(List)
```

```
List.pop(7)
```

```
print(List)
```

15. Predict the output

```
from collections import Counter
```

```
list = ['blue', 'pink', 'green', 'green', 'yellow', 'pink', 'orange']
```

```
print(Counter(list))
```

16. Write a python program to find the frequency of each element in the list without using pre-defined function.

17. Create a matrix and display the elements of it, using python program.

18. Depict how can we concatenate two tuples using Python program.