Passing Arguments by reference

A variable is Python is a reference to the object memory. So, both formal and actual arguments refer to the same object. The following code snippet will confirm this.

We have used the id() function earlier. It returns a unique integer corresponding to identity of an object. In the code displayed above, the id() of a list object before and after passing to a function shows an identical value.

```
>>> def myfunction(newlist):
print("List accessed in function: ", "id:", id(newlist))
return

>>> mylist=[10,20,30,40,50]
>>> print("List before passing to function: ", "id:", id(mylist))
List before passing to function:    id: 2430484894856>>> myfunction(mylist)
List accessed in function:    id: 2430484894856
>>>
```

If we modify the list object inside the function and display its contents after the function is completed, changes are reflected outside the function as well.

The following result confirms that arguments are passed by reference to a Python function.

```
>>> def myfunction(list):
list.append(60)
print("modified list inside function:", list)return

>>> mylist=[10,20,30,40,50]
>>> print("list before passing to function:", mylist)
list before passing to function: [10, 20, 30, 40, 50]
>>> myfunction(mylist)
modified list inside function: [10, 20, 30, 40, 50, 60]
>>> print("list after passing to function:", mylist)
```

```
list after passing to function: [10, 20, 30, 40, 50, 60]
>>>
```