9.6. Cloning Lists

If we want to modify a list and also keep a copy of the original, we need to be able to make a copy of the list itself, not just the reference. This process is sometimes called **cloning**, to avoid the ambiguity of the word copy.

The easiest way to clone a list is to use the slice operator.

Taking any slice of `a` creates a new list. In this case the slice happens to consist of the whole list.

Save & Run 5/13/2021, 2:02:49 PM - 2 of 2 Show in CodeLens

```python
a = [81,82,83]

b = a[:]       # make a clone using slice
print(a == b)
print(a is b)

b[0] = 5

print(a)
print(b)

```

```
True
False
[81, 82, 83]
[5, 82, 83]
```

Activity: 1 -- ActiveCode (clens8_5_1)

Now we are free to make changes to `b` without worrying about `a`. Again, we can clearly see in codelens that `a` and `b` are entirely different list objects.

Check your understanding

seqmut-5-1: What is printed by the following statements?

```python
alist = [4,2,8,6,5]
blist = alist * 2
blist[3] = 999
print(alist)
```

A. [4,2,8,999,5,4,2,8,6,5]
B. [4,2,8,999,5]
[✓ C. [4,2,8,6,5]]

Check me Compare me

✓ Yes, alist was unchanged by the assignment statement. blist was a copy of the references in alist.

Activity: 2 -- Multiple Choice (question8_5_1)

You have attempted 3 of 2 activities on this page

✓ Completed. Well Done!

9.5. Aliasing">
ising">
9.7. Mutating Methods">

| Back to top