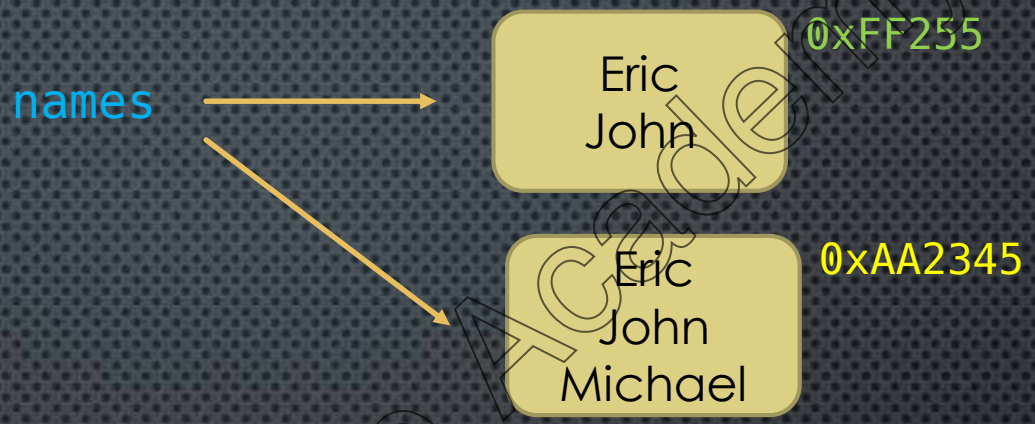
MUTABLE SEQUENCE TYPES

Mutating Objects

```
names = ['Eric', 'John']
```




```
names = names + ['Michael']
```

This is NOT mutation!

Mutating an object means changing the object's state without creating a new object

```
names = ['Eric', 'John']
names.append('Michael')
```

Mutating Using []

`s[i] = x`	element at index `i` is replaced with `x`
`s[i:j] = s2`	slice is replaced by the contents of the iterable `s2`
`del s[i]`	removes element at index `i`
`del s[i:j]`	removes entire slice

We can even assign to extended slices: `s[i:j:k] = s2`

We will come back to mutating using slicing in a lot more detail in an upcoming video

Some methods supported by mutable sequence types such as lists

`s.clear()`	removes all items from `s`
`s.append(x)`	appends `x` to the end of `s`
`s.insert(i, x)`	inserts `x` at index `i`
`s.extend(iterable)`	appends contents of `iterable` to the end of `s`
`s.pop(i)`	removes and returns element at index `i`
`s.remove(x)`	removes the first occurrence of `x` in `s`
`s.reverse()`	does an in-place reversal of elements of `s`
`s.copy()`	returns a shallow copy

and more…

Coding Exercises