ASSIGNMENTS IN MUTABLE SEQUENCES

Assigning Values via Indexes, Slices and Extended Slices

We have seen how we can extract elements from a sequence by using indexing, slicing, and extended slicing

```
[i]
[i:j]       slice(i, j)
[i:j:k]     slice (i, j, k)     k
```

(if k=1 then it's just a standard slice)

Mutable sequences support assignment via a specific index

and they also support assignment via slices

The value being assigned via slicing and extended slicing **must** to be an iterable

(any iterable, not just a sequence type)

Replacing a Slice

A slice can be replaced with another iterable

For regular slices (non-extended), the slice and the iterable need not be the same length

```
l = [1, 2, 3, 4, 5]          l[1:3] → [2, 3]

l[1:3] = (10, 20, 30)        l → [1, 10, 20, 30, 4, 5]
```

The list `l` was mutated → `id(l)` did not change

With extended slicing, the extended slice and the iterable must have the same length

```
l = [1, 2, 3, 4, 5]          l[0:4:2] → [1, 3]

l[0:4:2] = [10, 30]          l → [10, 2, 30, 4, 5]
```

The list `l` was mutated

Deleting a Slice

Deletion is really just a special case of replacement

We simply assign an empty iterable → works for standard slicing only
(extended slicing replacement needs same length)

```
l = [1, 2, 3, 4, 5]
```

```
l[1:3] → [2, 3]
```

```
l[2:3] = []
```

```
l → [1, 4, 5]
```

The list `l` was mutated

Insertions using Slices

We can also insert elements using slice assigment

The trick here is that the slice must be empty

otherwise it would just replace the elements in the slice

```
l = [1, 2, 3, 4, 5]                l[1:1] → []

l[1:1] = 'abc'                     l[1:1] → [1, 'a', 'b', 'c', 2, 3, 4, 5]
```

The list `l` was mutated

Obviously this will also not work with extended slices

extended slice assignment requires both lengths to be the same

but for insertion we need the slice to be empty,
and the iterable to have some values

Code Exercises