SEQUENCE TYPES

© 2018 MathByte Academy

What is a sequence?

In Math: $S = x_1, x_2, x_3, x_4, \ldots$ (countable sequence)

Note the sequence of indices: 1, 2, 3, 4, …

We can refer to any item in the sequence by using it's index number x_2 or `S[2]`

So we have a concept of the first element, the second element, and so on… → positional ordering

Python lists have a concept of positional order, but sets do not

A list is a sequence type

A set is not

In Python, we start index numbers at 0, not 1 (we'll see why later)

$S = x_0, x_1, x_2, x_3, \ldots$ → `S[2]` is the third element

© 2018 Mathbyte Academy

Built-In Sequence Types

mutable   lists   bytearrays

immutable   strings   tuples   range   bytes

more limited than lists, strings and tuples

in reality a tuple is more than *just* a sequence type

Additional Standard Types: `collections` package   `namedtuple`
`deque`

`array` module   `array`

© 2018 MathByte Academy

Homogeneous vs Heterogeneous Sequences

Strings are homogeneous sequences

each element is of the same type (a character) `'python'`

Lists are heterogeneous sequences

each element may be a different type `[1, 10.5, 'python']`

Homogeneous sequence types are usually more efficient (storage wise at least)

e.g. prefer using a string of characters, rather than a list or tuple of characters

© 2018 Mathbyte Academy

Iterable Type vs Sequence Type

What does it mean for an object to be iterable?

it is a container type of object and we can list out the elements in that object one by one

So any sequence type is iterable

```
l = [1, 2, 3]            for e in l  ✓

                         l[0]  ✓
```

But an iterable is not necessarily a sequence type → iterables are more general

```
s = {1, 2, 3}            for e in s  ✓

                         s[0]  ✗
```

© 2018 Mathbyte Academy

Standard Sequence Methods

Built-in sequence types, both mutable and immutable, support the following methods

`x in s`

`x not in s`

`s1 + s2` concatenation

`s * n (or n * s)` (n an integer) repetition

`len(s)`

`min(s)`

`max(s)`

(if an ordering between elements of s is defined)

This is not the same as the ordering (position) of elements inside the container, this is the ability to compare pairwise elements using an order comparison (e.g. <, <=, etc.)

`s.index(x)`	index of first occurrence of x in s
`s.index(x, i)`	index of first occurrence of x in s at or after index i
`s.index(x, i, j)`	index of first occurrence of x in s at or after index i and before index j

© 2018 MathByte Academy

Standard Sequence Methods

`s[i]`	the element at index `i`
`s[i:j]`	the slice from index `i`, to (but not including) `j`
`s[i:j:k]`	extended slice from index `i`, to (but not including) `j`, in steps of `k`

Note that slices will return in the same container type

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

`range` objects are more restrictive:

no concatenation / repetition

`min`, `max`, `in`, `not in` not as efficient

© 2018 MathByte Academy

Hashing

Immutable sequence types may support hashing `hash(s)`

but not if they contain mutable types!

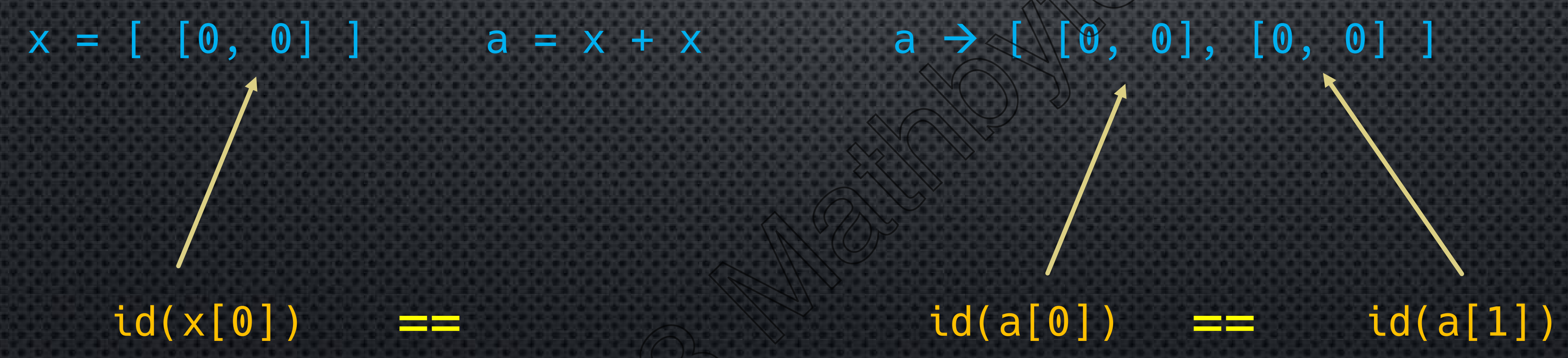
We'll see this in more detail when we look at Mapping Types

© 2018 Mathbyte Academy

Review: Beware of Concatenations

```
x = [1, 2]           a = x + x        a → [1, 2, 1, 2]

x = 'python'         a = x + x        a → 'pythonpython'
```



```
x = [ [0, 0] ]       a = x + x        a → [ [0, 0], [0, 0] ]

    id(x[0])    ==                       id(a[0])   ==    id(a[1])
```

```
a[0] is x[0]
a[1] is x[0]
```

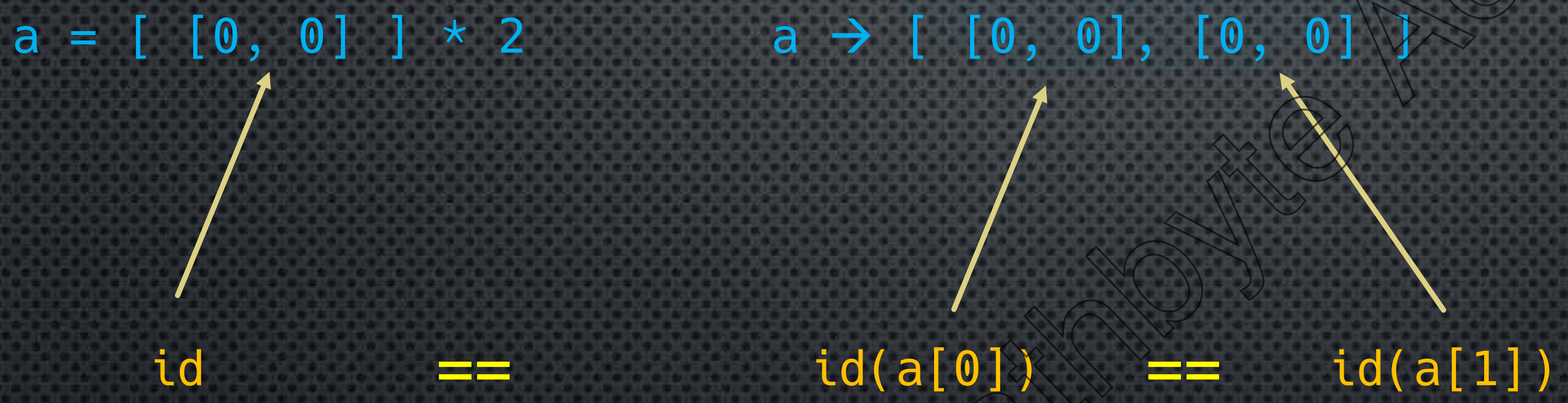
```
a[0][0] = 100       a → [ [100, 0], [100, 0] ]
```

© 2018 Mathbyte Academy

Review: Beware of Repetitions

```
a = [1, 2] * 2                a → [1, 2, 1, 2]

a = 'python' * 2              a → 'pythonpython'
```



```
a = [ [0, 0] ] * 2            a → [ [0, 0], [0, 0] ]

     id          ==               id(a[0])    ==    id(a[1])
```

```
a[0][0] = 100          a → [ [100, 0], [100, 0] ]
```

Same happens here, but because strings are immutable it's quite safe

```
a = ['python'] * 2            a → ['python', 'python']
```

© 2018 Mathbyte Academy

Coding

© 2018 Mathbyte Academy