Introduction to Python

21 February 2022 - 4 March 2022

Instructor: Lucy Havens

Day 3 Recap

Lists

Dictionaries

Tuples

Sets

Questions about assignment 3? 2? 1?

Assignment 3
Try It & Challenge Cells

DAY 4

Equivalence: == and !=

Equal to: ==

```
True == False
>> False
```

Not equal to: !=

```
True != False
>> True
```

Note: = vs. ==

Assign a value to a variable with a **single** equals sign

```
x = “hello world”
x
>> “hello world”
```

Containment: in, not in

Check whether or not a value is contained in a list, set, or tuple

```
my_list = [2, 4, 8, 16, 32]
2 in my_list
>> True

my_set = {“apples”, “bananas”, “oranges”}
“peaches” not in my_set
>> True
```

Let’s code!

Python Memory Manager

Allocation: process of assigning memory space for a purpose
Deallocation: freeing up memory space
 "garbage collection"

If all your memory space is used up, your programs won't run.
Python automatically makes sure this doesn't happen!

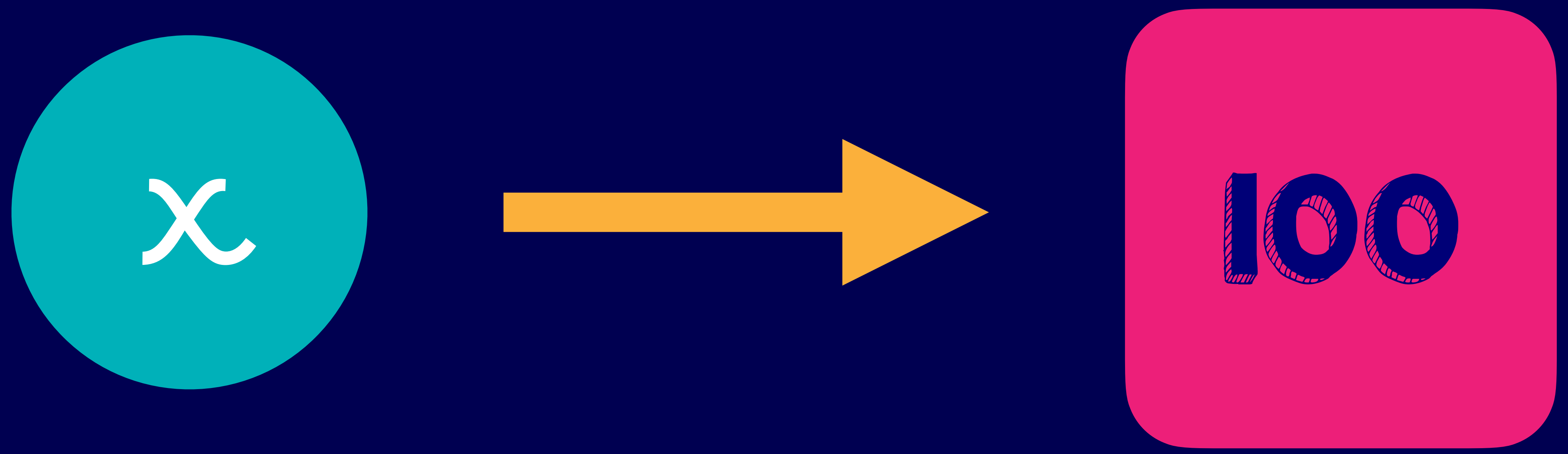
If your code uses memory efficiently, your code will run efficiently.

Memory Management Visualized

KEY

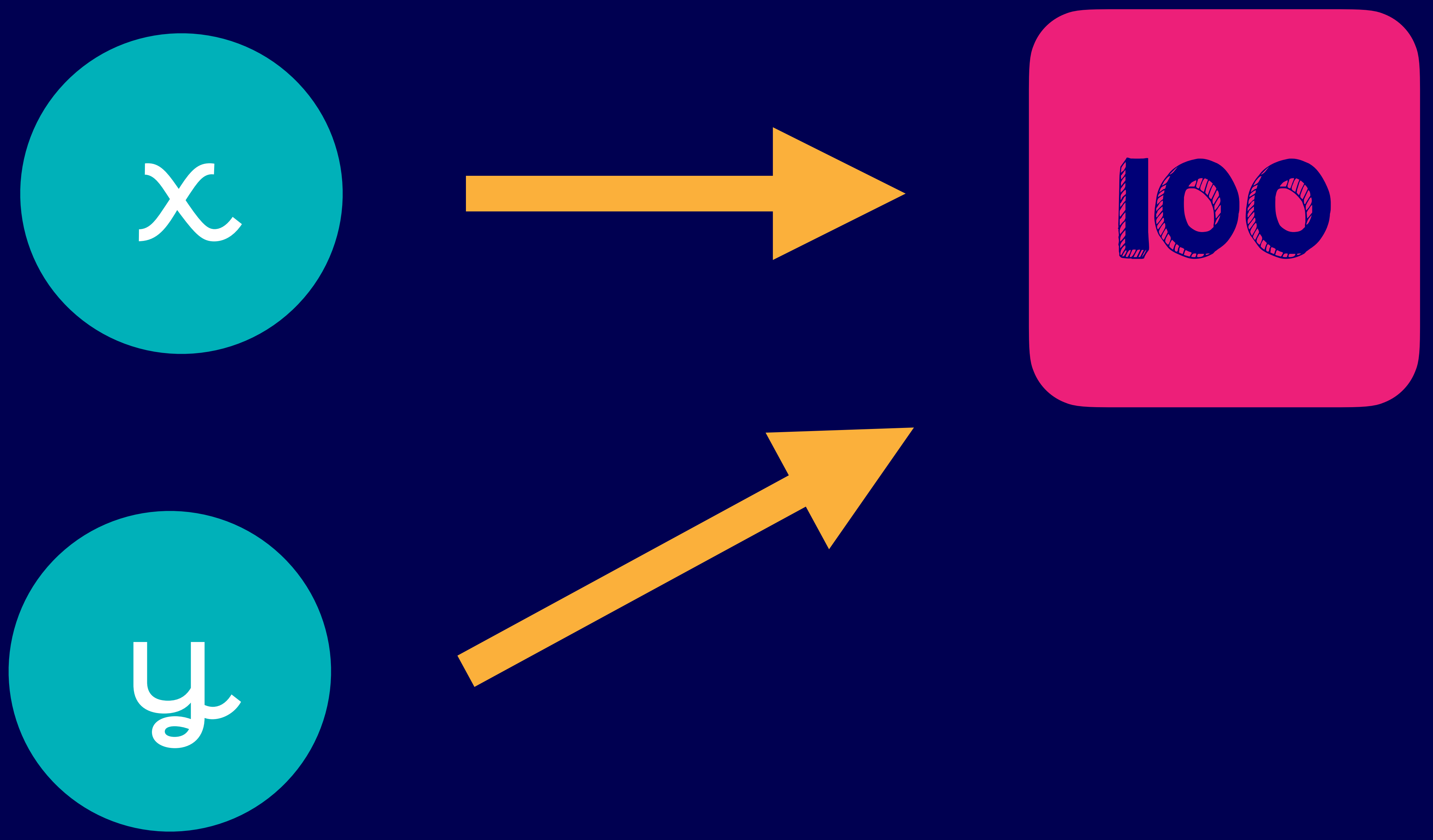
🟥	Object
→	Reference
●	Variable

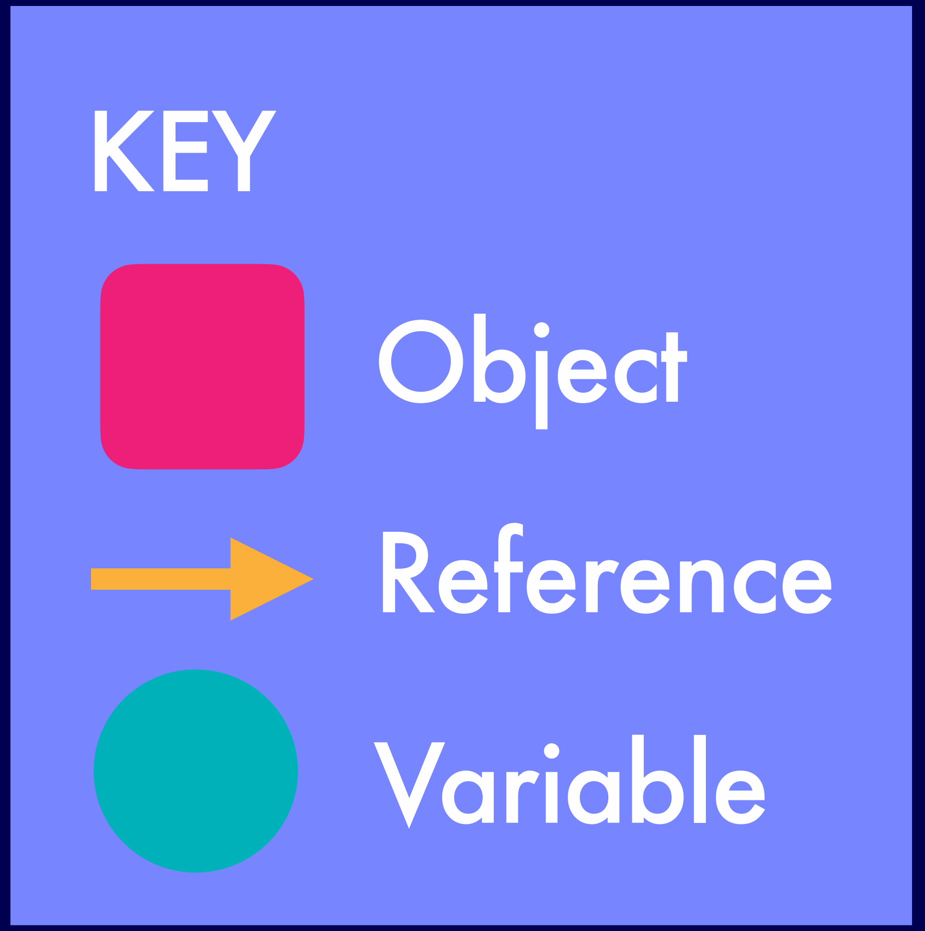
```
x = 100
```



Memory Management Visualized

```
x = 100
y = 100
```



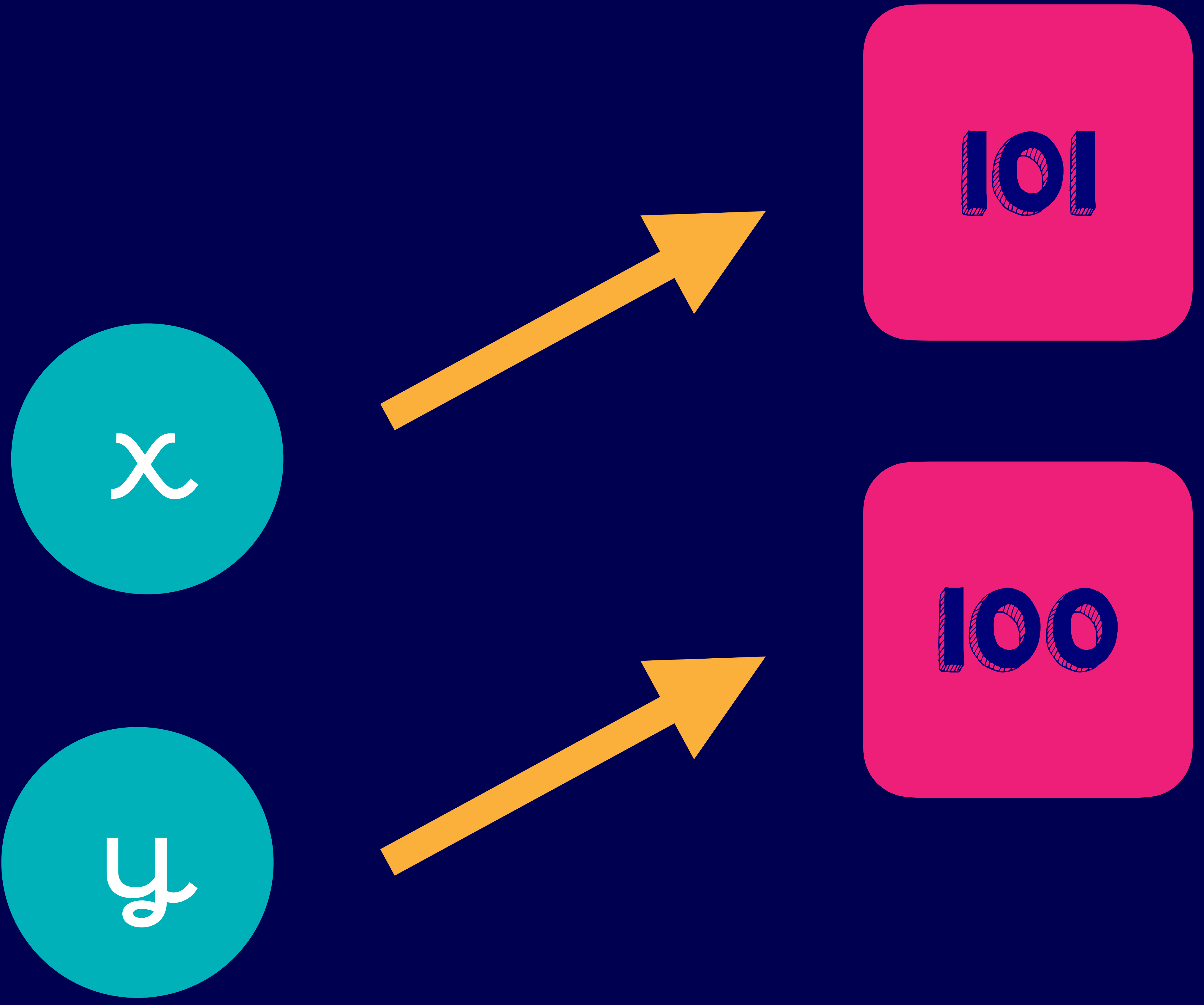


Memory Management Visualized

KEY

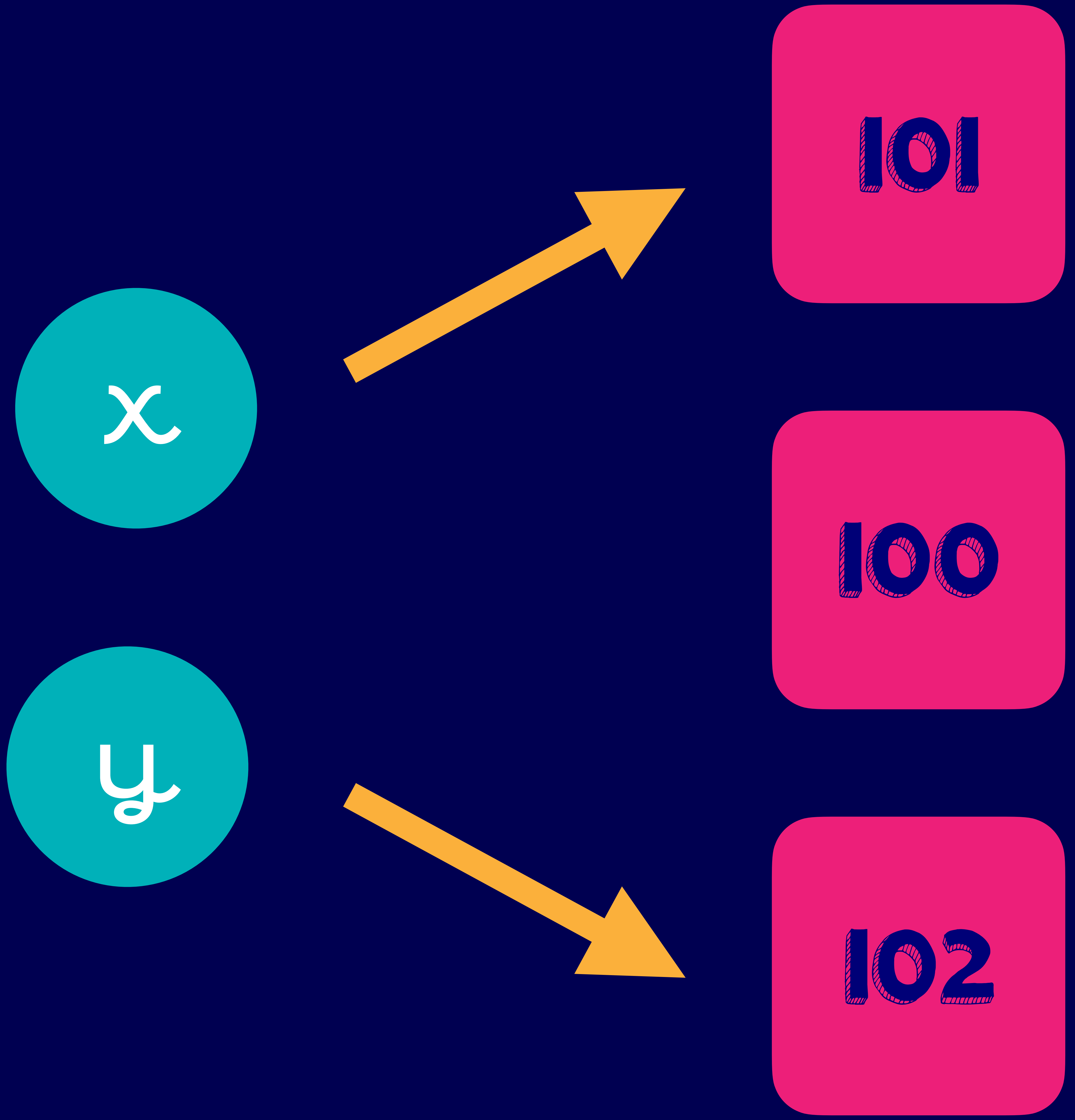
🟥	Object
→	Reference
●	Variable

```
x = 100
y = 100
x = 101
```



Memory Management Visualized

```
x = 100
y = 100
x = 101
y = 102
```

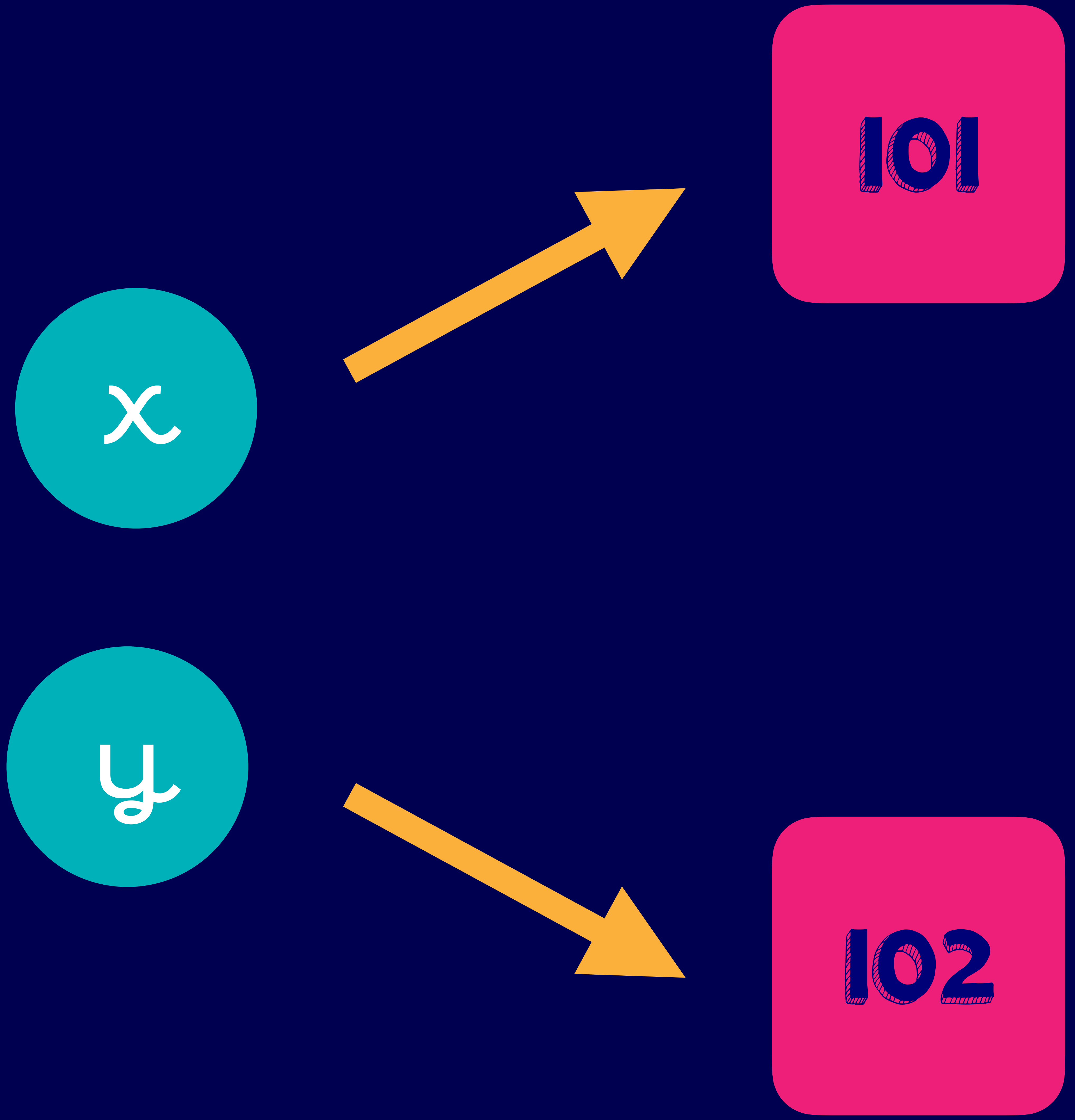


KEY

🟥	Object
→	Reference
●	Variable

Memory Management Visualized

```
x = 100
y = 100
x = 101
y = 102
```



KEY

🟥	Object
→	Reference
●	Variable

Let’s code!

Writing Efficient Code

List Comprehension: a shorter, faster way to iterate than looping

```
x = []
for i in range(10):
    x.append(i*i)
```

```
x = [i*i for i in range(10)]
```

Writing Efficient Code

If you're looping through large amounts of data (i.e., high-resolution images), you can use a generator instead of list comprehension.

```
x = [i*i for i in range(10)]
x = (i*i for i in range(10))
```

Generators create (and delete) items on the fly, rather than storing all items in memory simultaneously.

List Comprehension or Generator?

Generators:

- For large data/lots of memory needed
- Returns an iterable, which isn't mutable or indexable, can't be sliced
- Good when you want to iterate over data *only once*

List Comprehension:

- When you want to iterate over data multiple times
- When you want to access and change the data iterated over
- Returns a list, which is mutable and indexable and can be sliced

Let’s code!

Final Thoughts

Good programming is a balance of…

- Readability
  - Commenting your code (`# like this`)
  - Naming variables intuitively
  - Consistent conventions for naming variables, functions
- Efficiency
  - For you
  - For your machine’s memory

Upcoming CDCS courses that use Python

Analysing Structured and Unstructured Data with Python’s Pandas and ElementTree Libraries

14 March - 25 March (2 days a week)

Machine Learning with Python

23 March - 13 April (2 days a week)

Text Analysis with Python’s NLTK Library

11 April - 22 April (2 days a week)

Further Resources

CDCS Training Resources: https://www.cdcs.ed.ac.uk/training#tab-482

LinkedIn Learning: visit MyEd, search “LinkedIn Learning,” click the LinkedIn Learning link and login to your LinkedIn account to hook it up to your University account

> *This is an amazing FREE resource for all University of Edinburgh students and staff with videos about programming and many other topics*

THANKS EVERYONE!

Please fill out this survey to give feedback on the course

We use your feedback to shape future training sessions!

https://forms.office.com/r/YYNrqvuNr8

Bonus: Regular Expressions

Pattern matching on strings

```
s = “How are you my friend?”
re.findall(“e”, s)
>> [“e”, “e”]
re.search(“\w+\?”, s).group()
 >> “friend?”
```

PYTHEX.ORG

Bonus: Accessing External Data

`os` - read and write files, among other things

`urllib` - access data through a website (a URL)