Background on Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

Python Overview

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

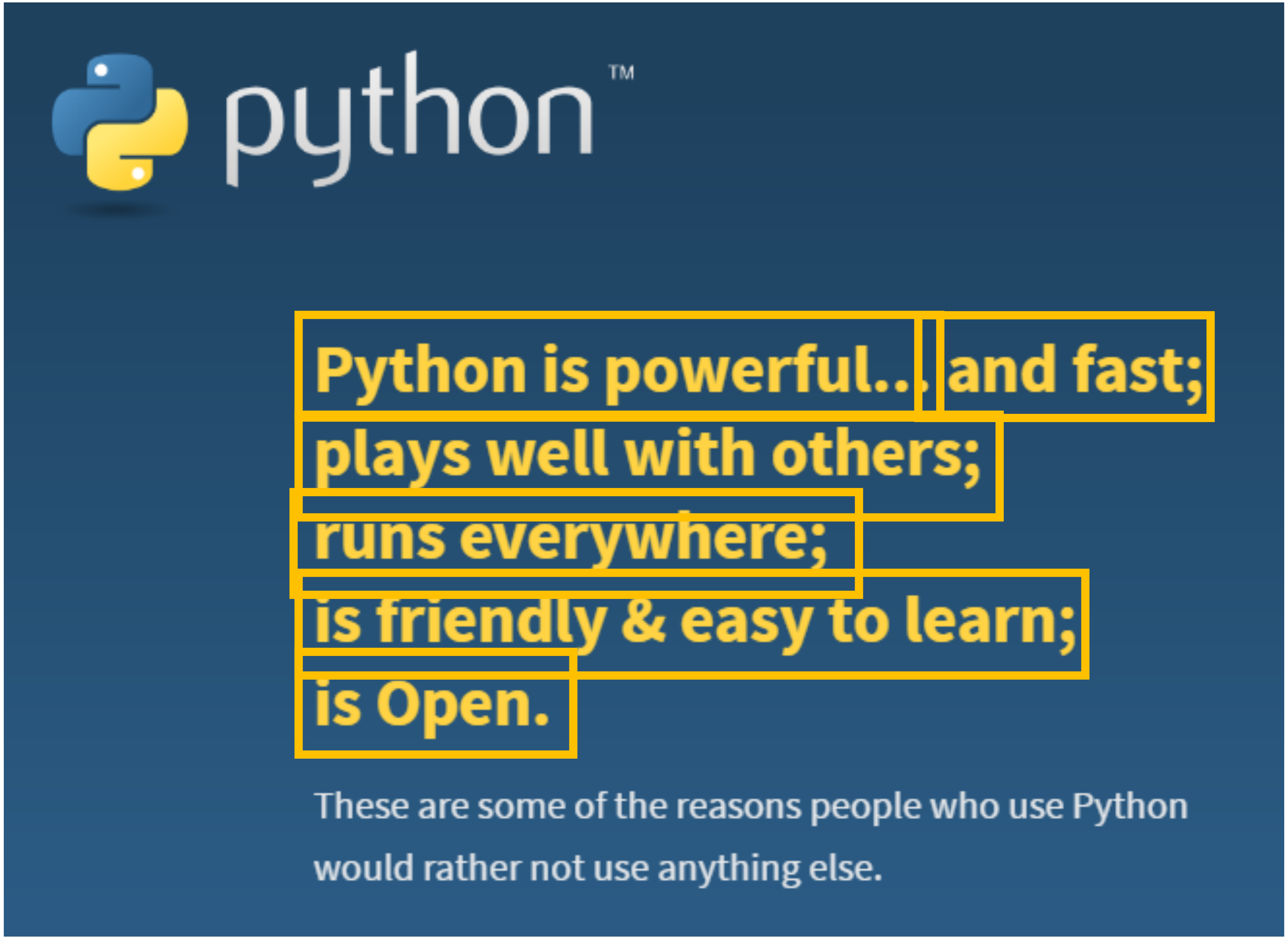
- Articulate the benefits of Python as a programming language

Jupyter

(Ju) Julia

(Pyt) Python

(R) R



Python is powerful... and fast;
plays well with others;
runs everywhere;
is friendly & easy to learn;
is Open.

These are some of the reasons people who use Python would rather not use anything else.

https://www.python.org/about/

Variables in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Write a simple program in python
- Use dynamic typing to assign values to variables

C

```
#include "stdio.h"
int main() {
  printf("Hello\n");
}
```

Java

```
public class Hi {
  public static void main (String [] args) {
    System.out.println("Hello");
  }
}
```

C

```c
#include "stdio.h"
int main() {
  printf("Hello\n");
}
```

python

```python
print("hello")
```

Notice: no ;

Java

```java
public class Hi {
  public static void main (String [] args) {
    System.out.println("Hello");
  }
}
```

C

```
#include "stdio.h"

int main() {
   int x = 3;
   int y = 4;
   printf("%s"\n,x+y);
}
```

python

```
x = 3
y = 4
print(x+y)
```

Notice: no types

Common Types in Python

- **Numeric:** integers, float, complex
- **Sequence:** list, tuple, range
- **Binary:** byte, bytearray
- **True/False:** bool
- **Text:** string

C

```
#include "stdio.h"


int main() {
  int x = 3;
  x = 4.5;
}
```

python

```
x = 3
x = 4.5
```

What happens when we run this in python?

C

```
#include "stdio.h"

int main() {
  int x = 3;
  x = 4.5;
}
```

python

```
x = 3
x = 4.5
```

Dynamic Typing!!

Objects in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Describe an object from a programming perspective
- Recognize that everything in python is an object

Objects

- Can hold data
- Can have actions associated with them

C

```
#include "stdio.h"

int main() {
  int x = 3;
  x = 4.5;
}
```

python

```
x = 3
x = 4.5
print(x+y)
```

Dynamic Typing!!

python

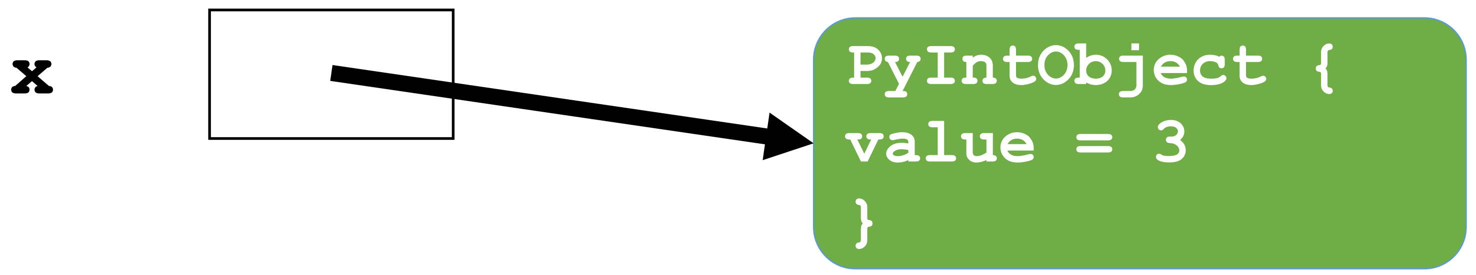
```
x = 3
```

```
PyIntObject {
    value;
    # other bookkeeping features
    # type, num_refs, etc.
}
```

python

```
x = 3
```

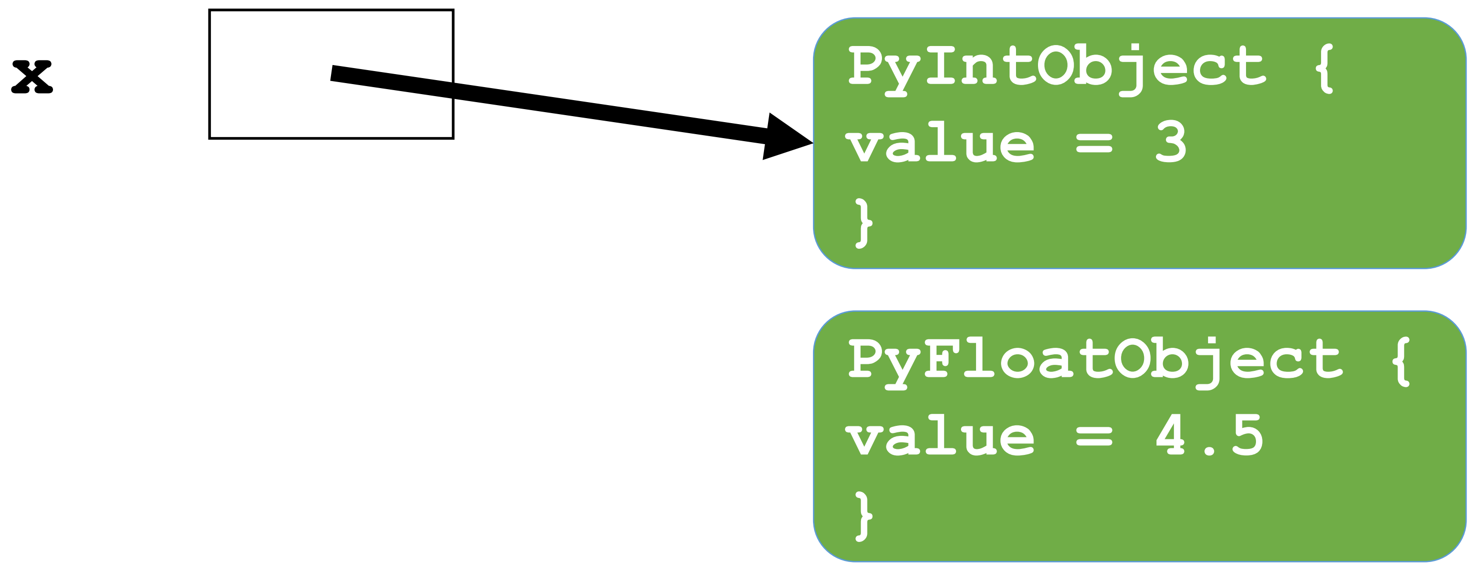
```
PyIntObject {
   value;
   # other bookkeeping features
   # type, num_refs, etc.
}
```



python

```
x = 3
x = 4.5
```

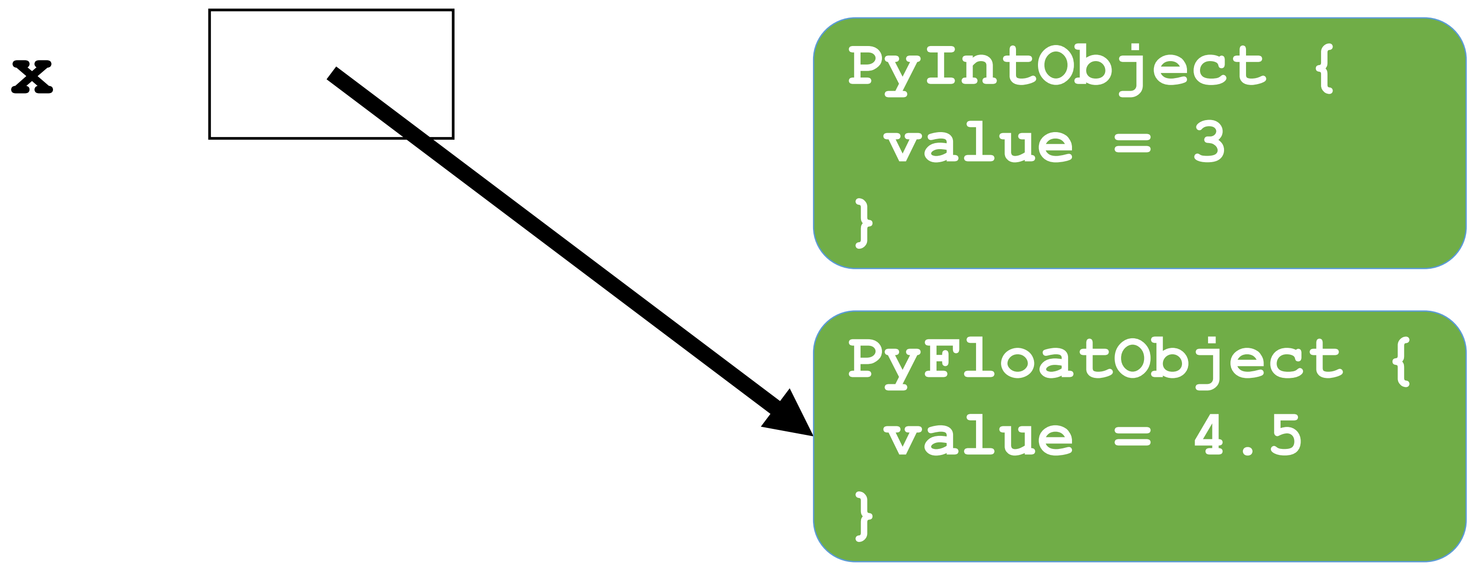
```
PyFloatObject {
   value;
   # other bookkeeping features
   # type, num_refs, etc.
}
```



python

```
x = 3
x = 4.5
```

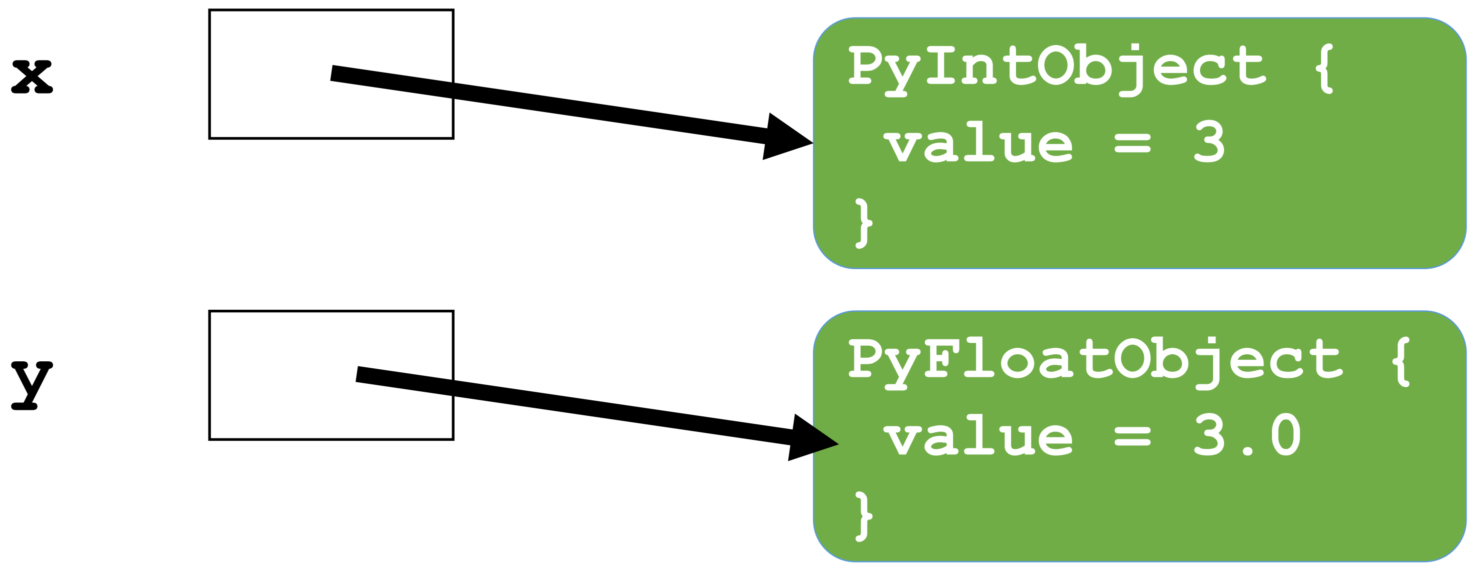
```
PyFloatObject {
   value;
   # other bookkeeping features
   # type, num_refs, etc.
}
```



Python shell

```
>>> x = 3
>>> y = 3.0
>>> x is y
```

```
is returns if the references
point to the same object
```



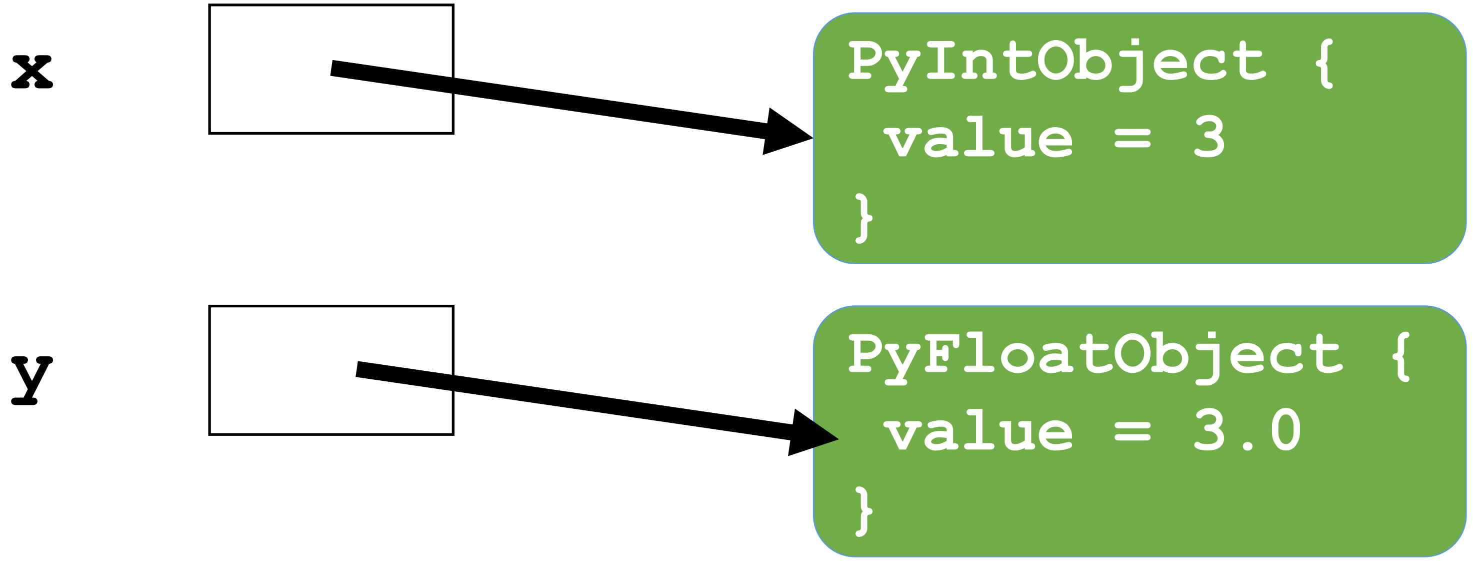
```
PyIntObject {
 value = 3
}
```

```
PyFloatObject {
 value = 3.0
}
```

Python shell

```
>>> x = 3
>>> y = 3.0
>>> x is y
False
```

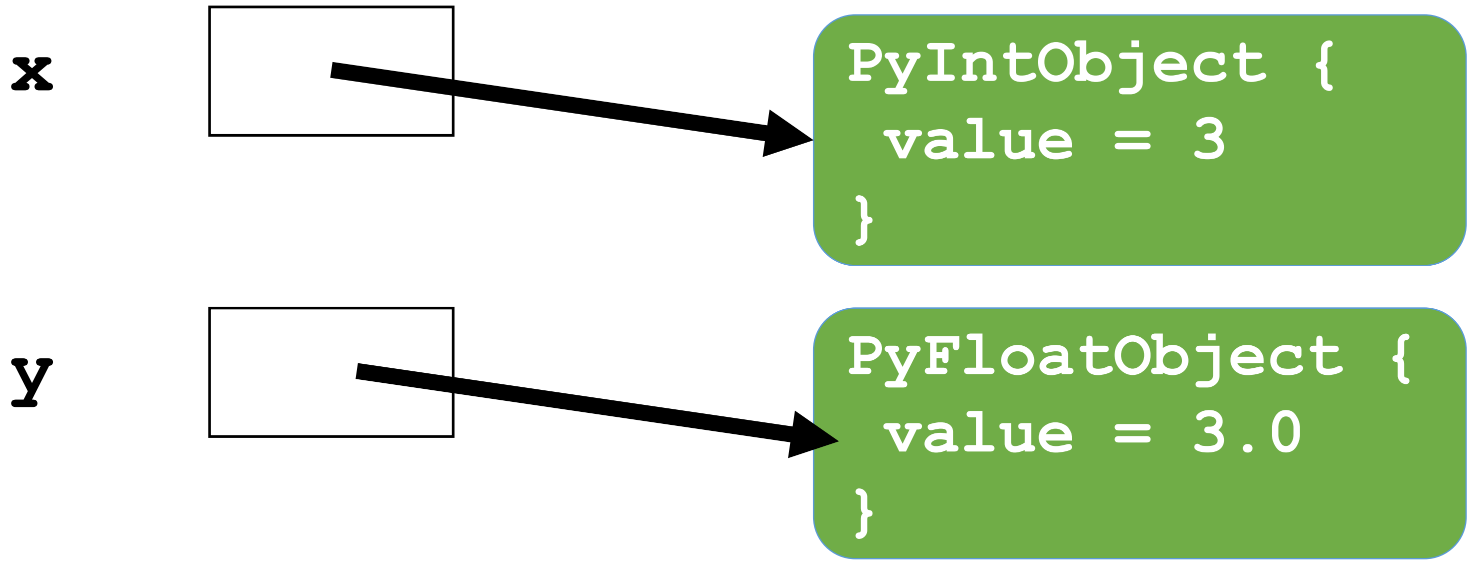
```
is returns if the references
point to the same object
```



Python shell

```
>>> x = 3
>>> y = 3.0
>>> x == y
```

```
is returns if the references
point to the same object
== tests for equality
```

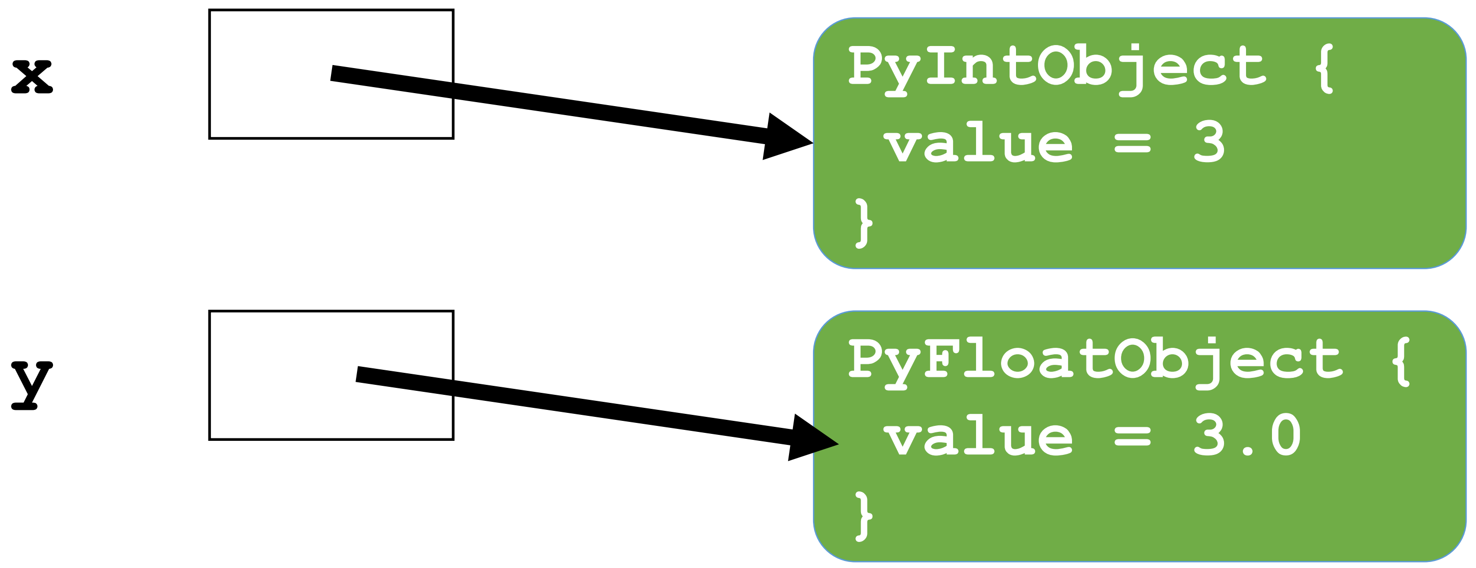


Python shell

```
>>> x = 3
>>> y = 3.0
>>> x == y
True
```

is returns if the references point to the same object

== tests for equality



Feel free to try this yourselves in python using shell

Objects in Python

Dr. Ilkay Altintas and Dr. Leo Porter

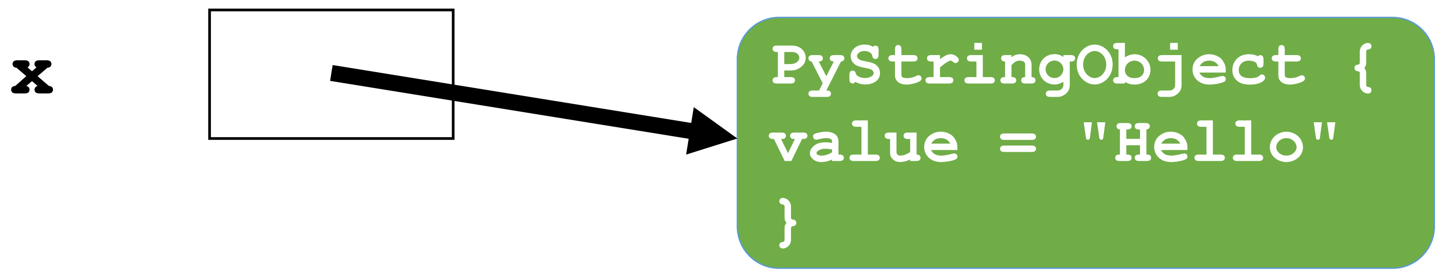
Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Create objects and call methods on objects

python

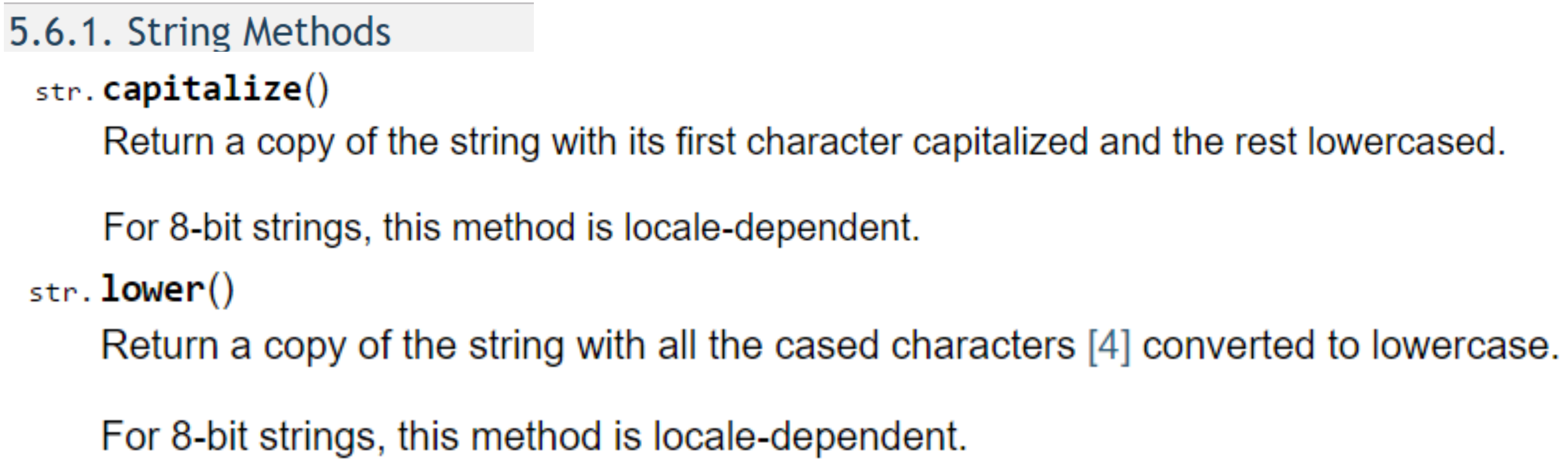
```
>>> x = "Hello"
```



```
PyStringObject {
value = "Hello"
}
```

python

```
>>> x = "Hello"
```



5.6.1. String Methods

`str.capitalize()`

Return a copy of the string with its first character capitalized and the rest lowercased.

For 8-bit strings, this method is locale-dependent.

`str.lower()`

Return a copy of the string with all the cased characters [4] converted to lowercase.

For 8-bit strings, this method is locale-dependent.

https://docs.python.org/2/library/stdtypes.html#string-methods

python

```
>>> x = "Hello"
>>> x.lower()
'hello'
```

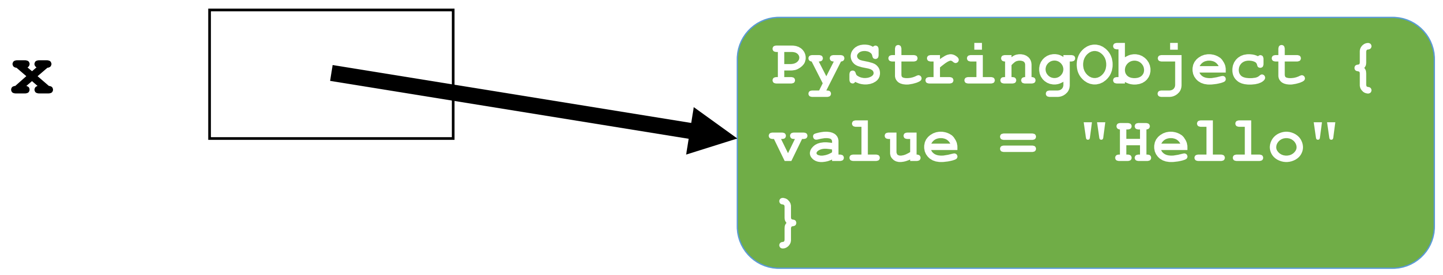
<var_name>.<method_name>(params)

python

```
>>> x = "Hello"
>>> x.lower()
'hello'
```

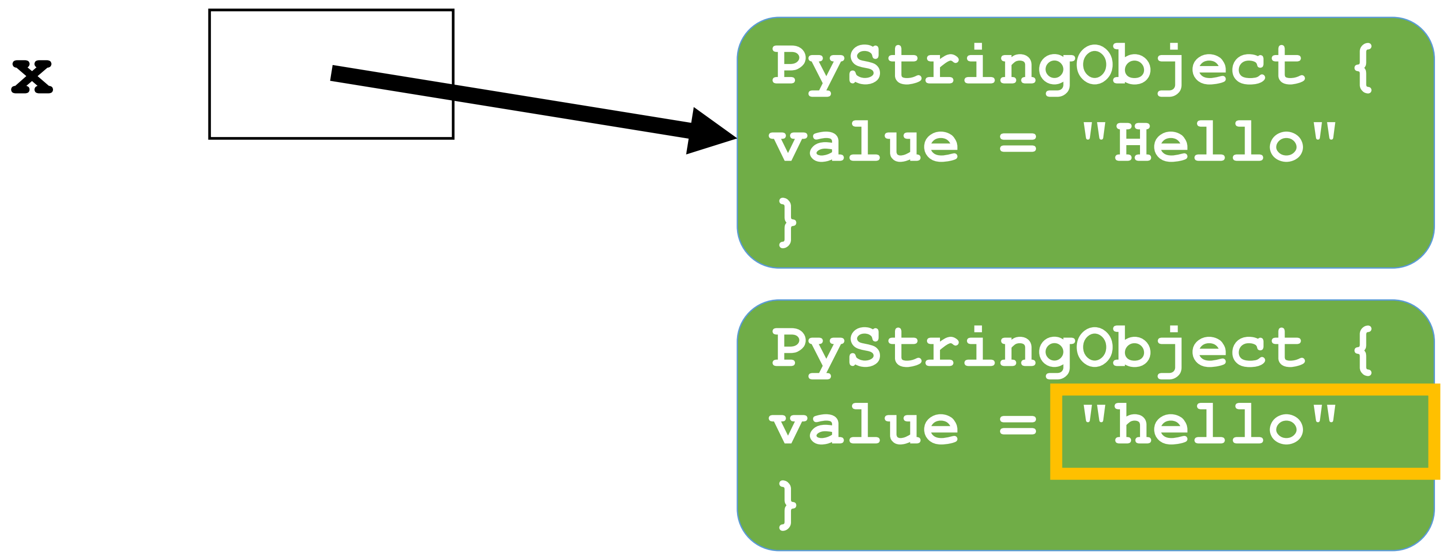
python

```
>>> x = "Hello"
```



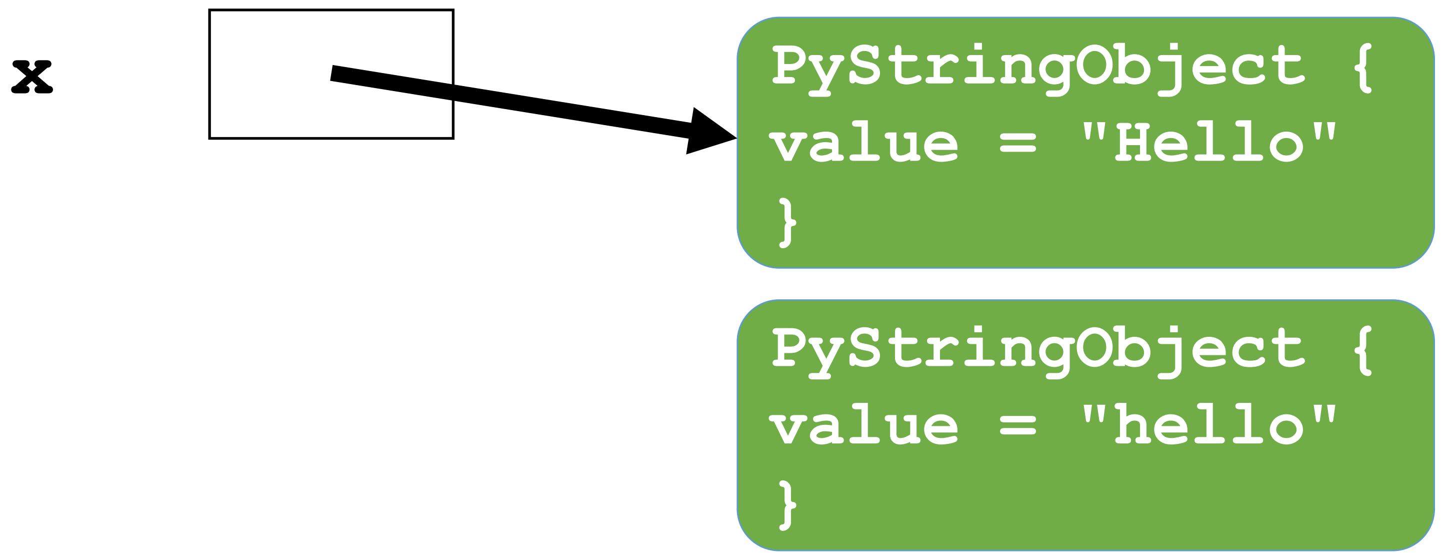
python

```
>>> x = "Hello"
>>> x = x.lower()
```



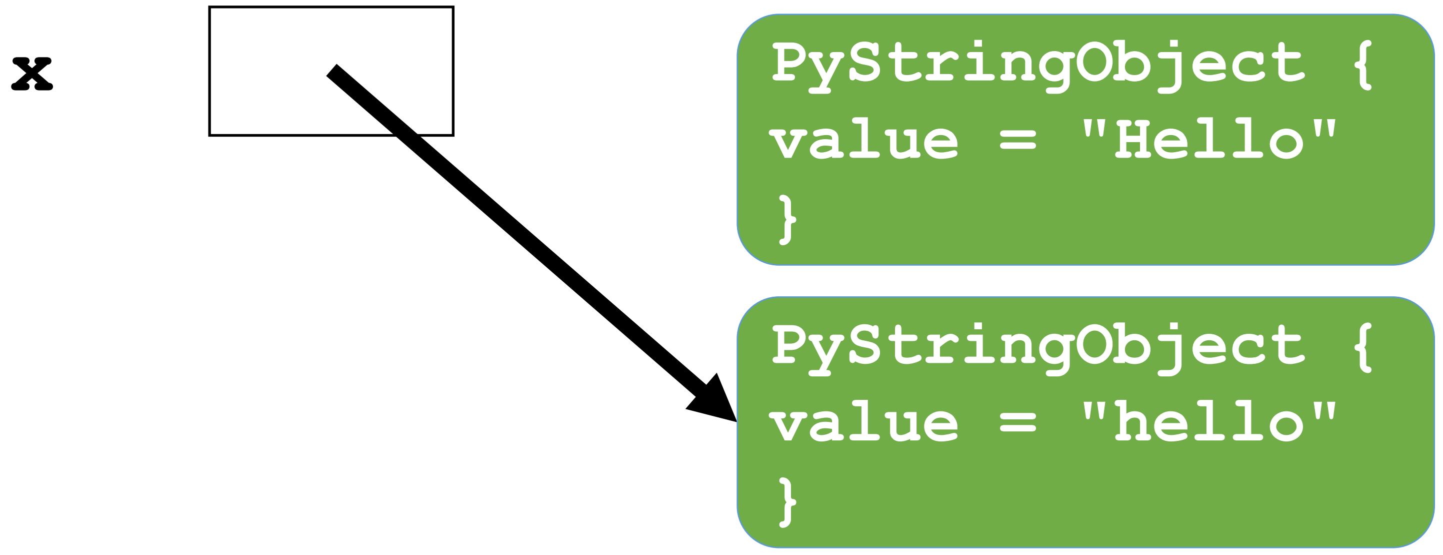
python

```
>>> x = "Hello"
>>> x = x.lower()
```



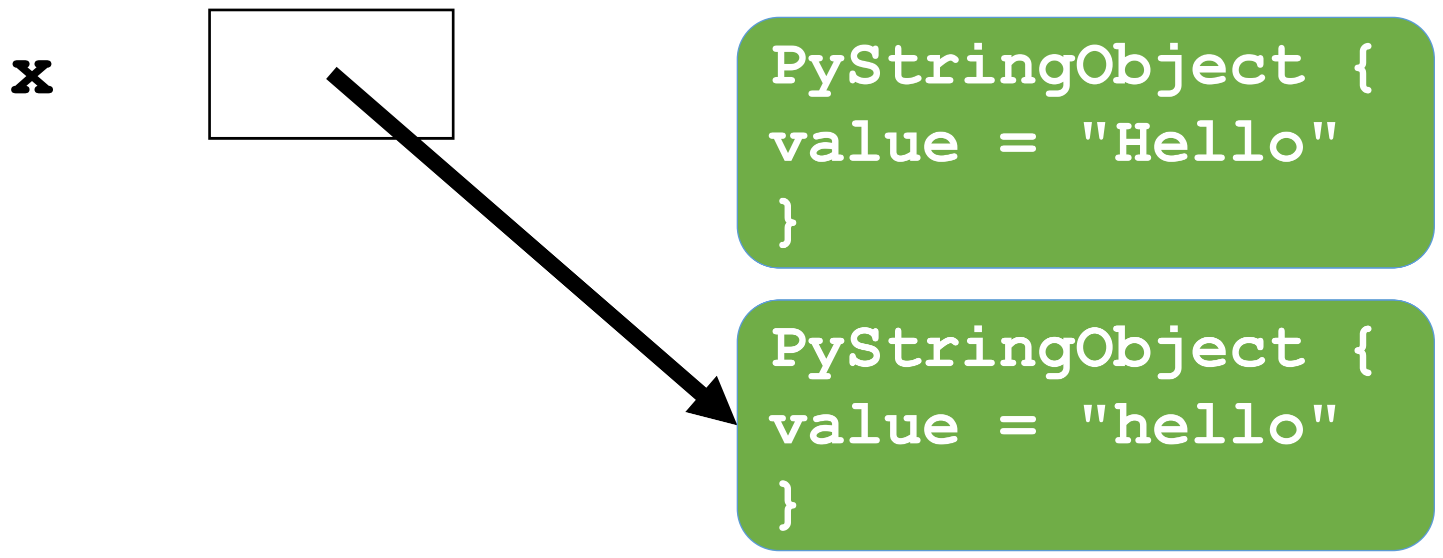
python

```
>>> x = "Hello"
>>> x = x.lower()
```



python

```
>>> x = "Hello"
>>> x = x.lower()
>>> x
'hello'
```



Variable Quiz - Explanation

Dr. Ilkay Altintas and Dr. Leo Porter

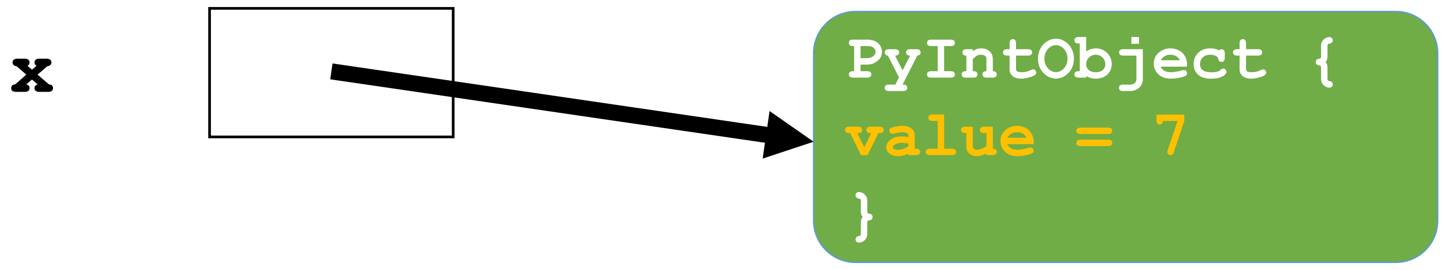
Twitter: #UCSDpython4DS

python

```
>>> x = 7
>>> y = x
>>> x = 3
>>> print(x,",  ",y)
```

python

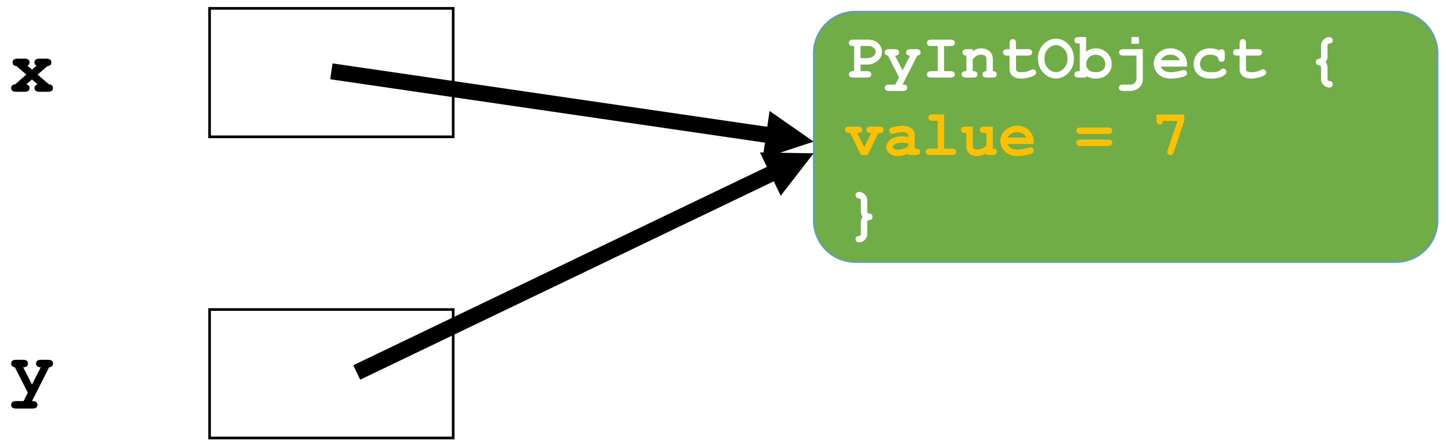
```
>>> x = 7
```



```
PyIntObject {
value = 7
}
```

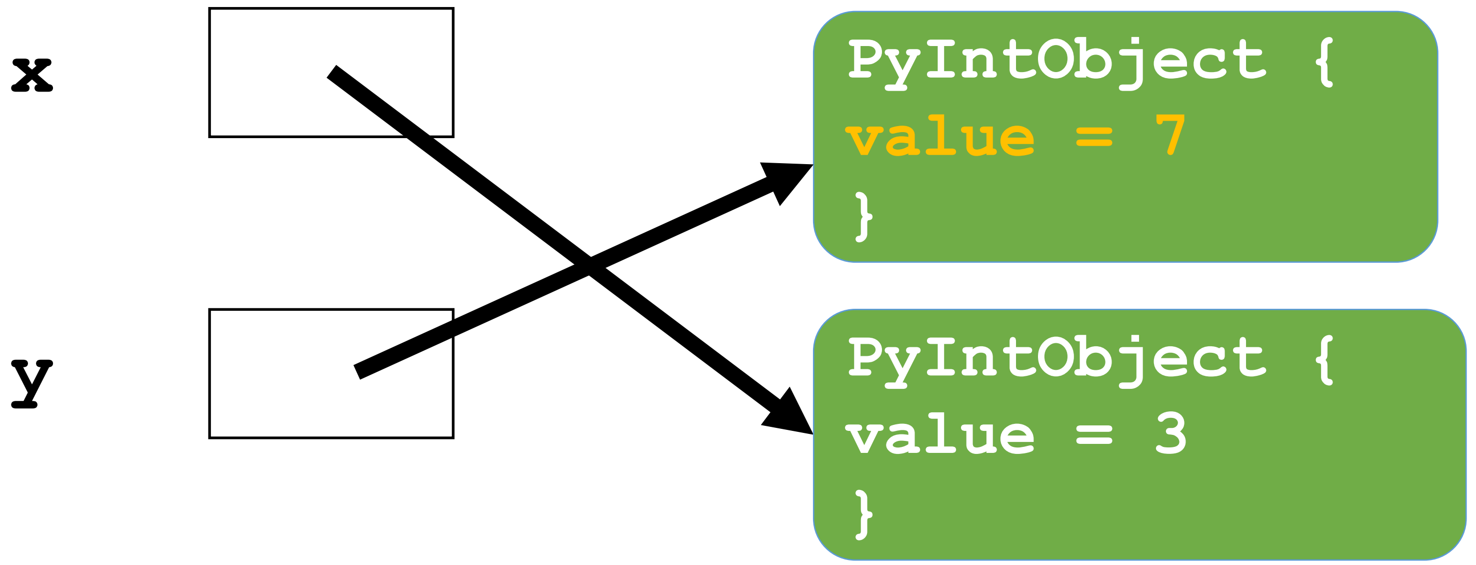
python

```
>>> x = 7
>>> y = x
```



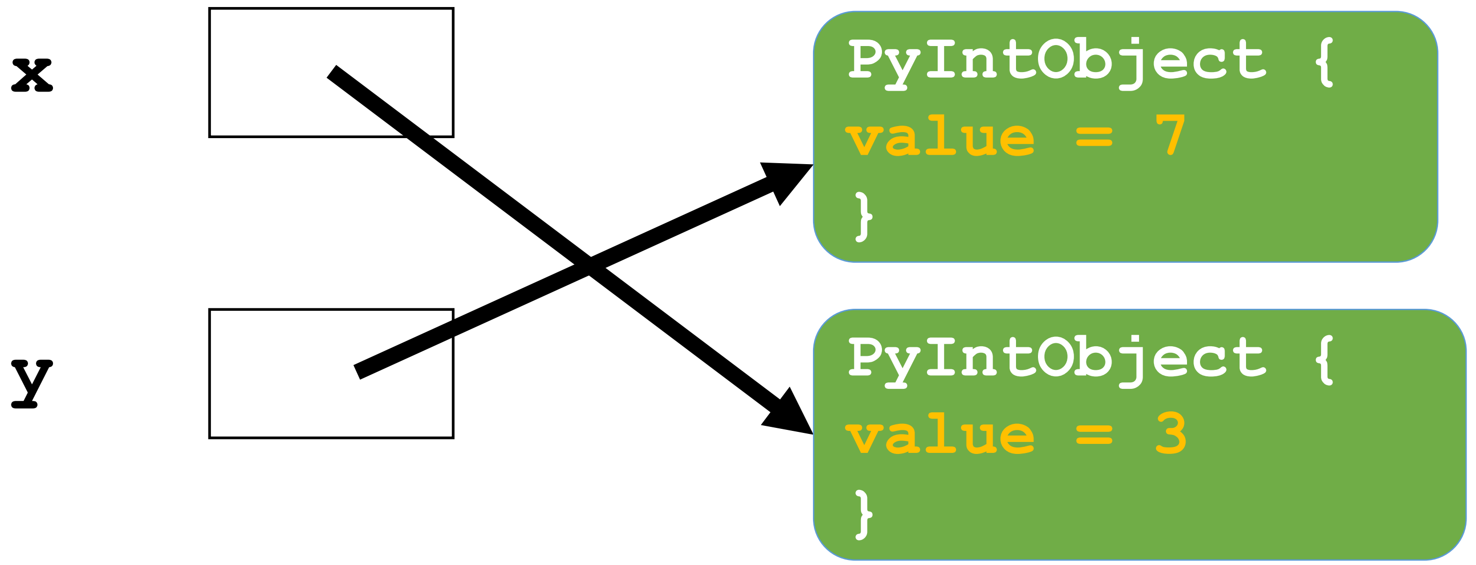
python

```
>>> x = 7
>>> y = x
>>> x = 3
```



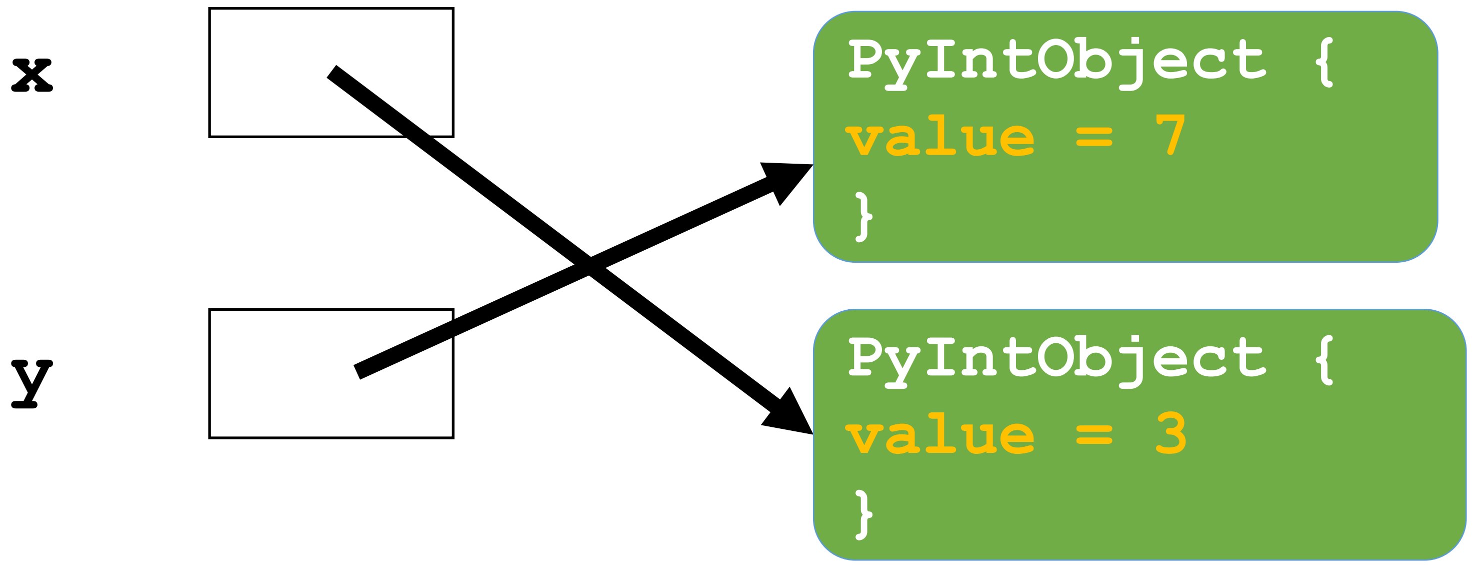
python

```
>>> x = 7
>>> y = x
>>> x = 3
>>> print(x,", ",y)
```



python

```
>>> x = 7
>>> y = x
>>> x = 3
>>> print(x,", ",y)
3, 7
```



Loops in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Author a python program which uses a loop

C

```
#include "stdio.h"
int main() {
   int i = 0;
   for(i=0; i < 10; i++) {
      printf("%d\n",i);
   }
}
```

python

```
for i in range(0,10):
        print(i)
```

Python uses indentation rather than brackets.

C

```
#include "stdio.h"
int main() {
  int i = 0;
  for(i=0; i < 10; i++) {
    printf("%d\n",i);
  }
}
```

python

```
for i in range(0,10):
     print(i)
```

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
for i in range(0,10,2):
     print(i)
```

What do you think this will print?

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
for i in range(0,10,2):
     print(i)
```

```
0
2
4
6
8
```

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
for i in _____________:
     print(i)
```

```
range(start, stop[, step])
```

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

```
2
5
8
11
```

python

```
for i in range(2,12,3) :
     print(i)
```

```
2
5
8
11
```

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
for i in range(2,12,3) :
    print(i)
```

```
2
5
8
11
```

Understanding how to use range well can be incredibly useful!

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
i = 2
while i < 12:
      print(i)
      i+=3
```

```
2
5
8
11
```

Python for Data Science

Loop Quiz - Explanation

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

python

```
i = 0
while i < 10:
     print(i)
     i+=1
```

How many times is the statement i<10 evaluated?

python

```
i = 0
while $\boxed{i < 10:}$
      print(i)
      i+=1
```

```
0 < 10   True
```

python

```
i = 0
while i < 10:
        print(i)
        i+=1
```

```
0 < 10   True
1 < 10   True
```

python

```
i = 0
while i < 10:
      print(i)
      i+=1
```

```
0 < 10   True
1 < 10   True
2 < 10   True
3 < 10   True
4 < 10   True
5 < 10   True
```

python

```
i = 0
while i < 10:
     print(i)
     i+=1
```

```
0 < 10  True      6 < 10  True
1 < 10  True      7 < 10  True
2 < 10  True      8 < 10  True
3 < 10  True      9 < 10  True
4 < 10  True
5 < 10  True
```

python

```
i = 0
while i < 10:
        print(i)
        i+=1
```

```
0 < 10   True       6 < 10   True
1 < 10   True       7 < 10   True
2 < 10   True       8 < 10   True
3 < 10   True       9 < 10   True
4 < 10   True       10 < 10 False
5 < 10   True
```

Conditions in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Author a python program which uses conditionals

python

```
for i in range(0,10,2):
     print(i)
```

```
0
2
4
6
8
```

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
for i in range(0,10):
    ________________
        print(i)
```

```
0
2
4
6
8
```

What should I put here to print just the even values?

range(**start**, **stop**[, **step**])

Returns values between **start** and **stop**, increasing by the value of **step** (defaults to 1).

python

```
for i in range(0,10):
         __________________
              print(i)
```

```
0
2
4
6
8
```

What should I put here to print just the even values?

% (modulo)

x % y produces the remainder from x / y.

For example, 22%3 is 1 because 22 / 3 is 21 R1

python

```
for i in range(0,10):
      if i % 2 == 0:
            print(i)
```

```
0
2
4
6
8
```

What should I put here to print just the even values?

% (modulo)

x % y produces the remainder from x / y.

For example, 22%3 is 1 because 22 / 3 is 21 R1

python

```
for i in range(0,10):
     if i % 2 == 0:
          print(i)
```

```
0
2
4
6
8
```

% (modulo)

x % y produces the remainder from x / y.

For example, 22%3 is 1 because 22 / 3 is 21 R1

python

```
for i in range(0,5):
     if i % 2 == 0:
          print(i)
     # fill in missing
          # code
```

% (modulo)

x % y produces the remainder from x / y.

For example, 22%3 is 1 because 22 / 3 is 21 R1

```
0
11
2
13
4
```

What do I need to change to print out the values on the left (hint, for odds, it is print 10+i)?

python

```
for i in range(0,5):
     if i % 2 == 0:
          print(i)
     else:
          print(i+10)
```

% (modulo)

x % y produces the remainder from x / y.

For example, 22%3 is 1 because 22 / 3 is 21 R1

```
0
11
2
13
4
```

python

```
for i in range(0,5):
     if i % 3 == 0:
          print(i)
     elif i % 3 == 1:
          print(i+10)
     else:
          print(i-10)
```

```
0
11
-8
3
14
```

% (modulo)

x % y produces the remainder from x / y.

For example, 22%3 is 1 because 22 / 3 is 21 R1

Functions in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Create a function in python with inputs and outputs
- Explain the implications of passing an object reference by value

C

```
int my_abs( int val) {
   if(val < 0) {
      return 0-val;
   }
   return val;
}
```

python

```
def my_abs(val):
     if val < 0:
          return 0-val
     return val
```

python

```
def my_abs(val):
    if val < 0:
        return 0-val
    return val

print(my_abs(-7))
```

```
7
```

python

```
def my_abs(val):
      if val < 0:
            return 0-val
      return val

print(my_abs(“Hi”))
```

```
Traceback (most recent call last):
  File “funct.py”, line 6, in <module>
    print(my_abs(“Hi”))
  File “funct.py”, line 2, in my_abs
    if val < 0:
TypeError: unorderable types: str() < int()
```

python

```
def print_abs(val):
      if val < 0:
            print(0-val)
      else:
            print(val)

x = print_abs(-2.7)
print(x)
```

What do you think this will do when we run it?

python

```
def print_abs(val):
      if val < 0:
            print(0-val)
      else:
            print(val)

x = print_abs(-2.7)
print(x)
```

```
2.7
None
```

python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

What do you think this will do when we run it?

python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

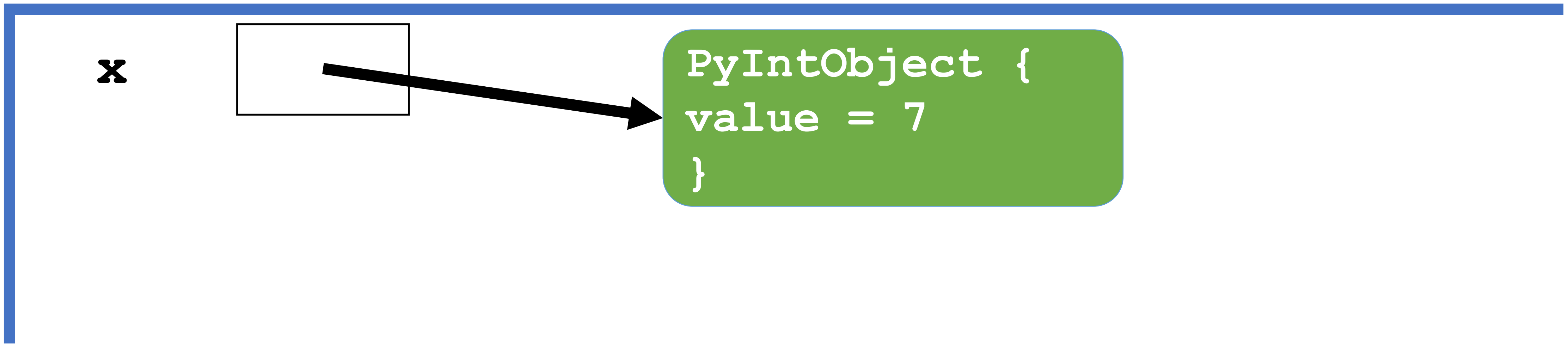
```
7
```

python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

```
7
```

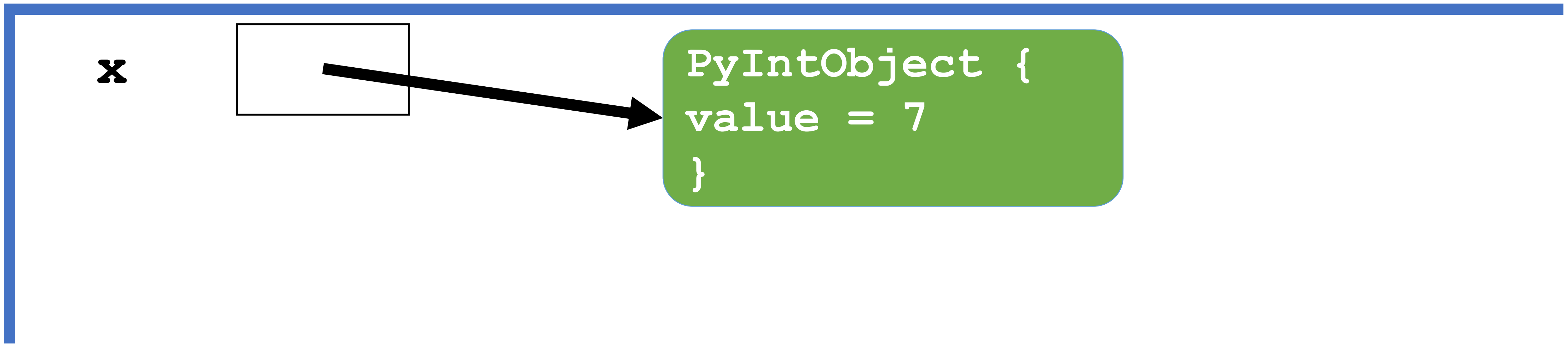


python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

```
7
```

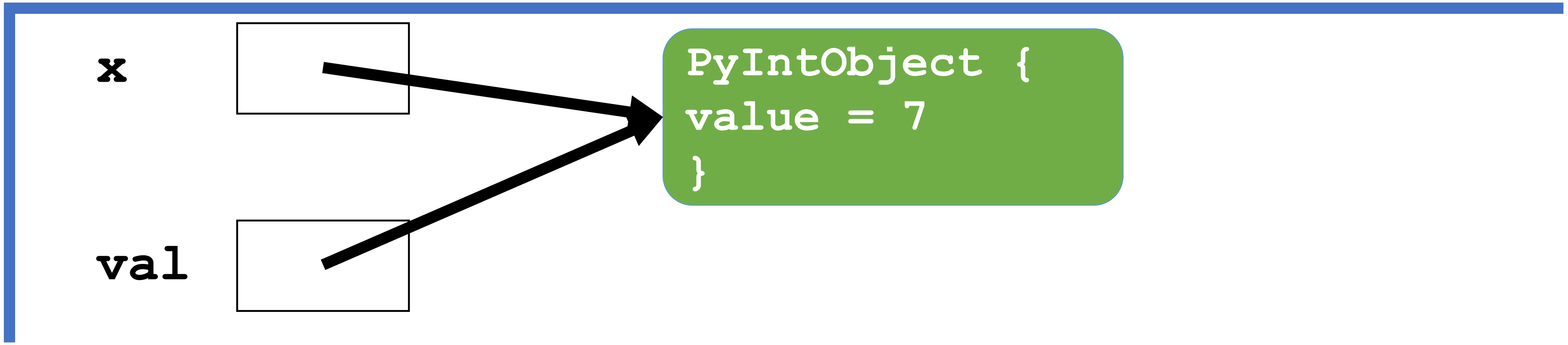


python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

```
7
```

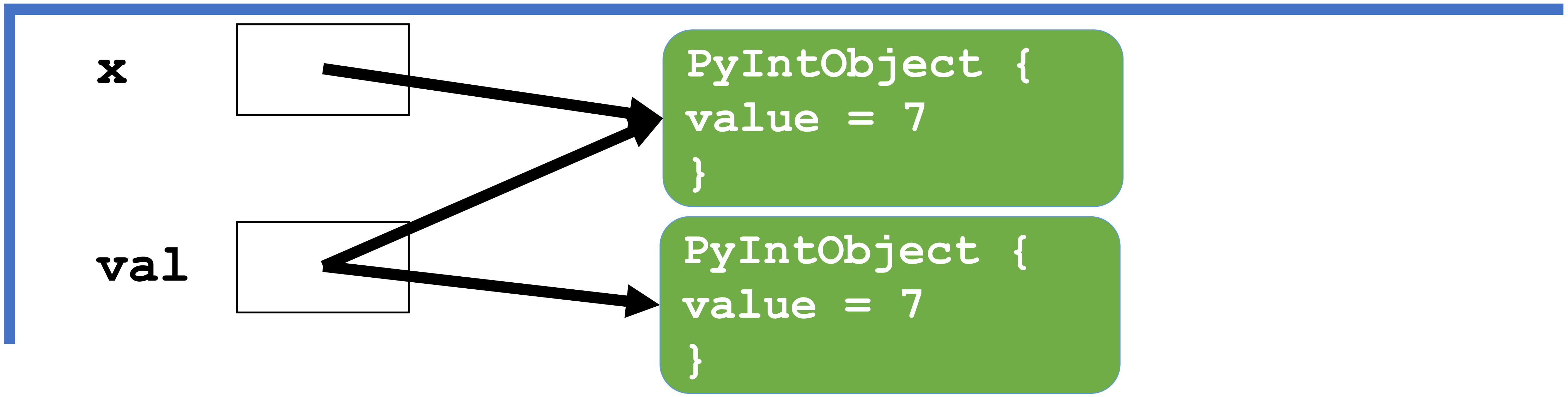


python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

```
7
```

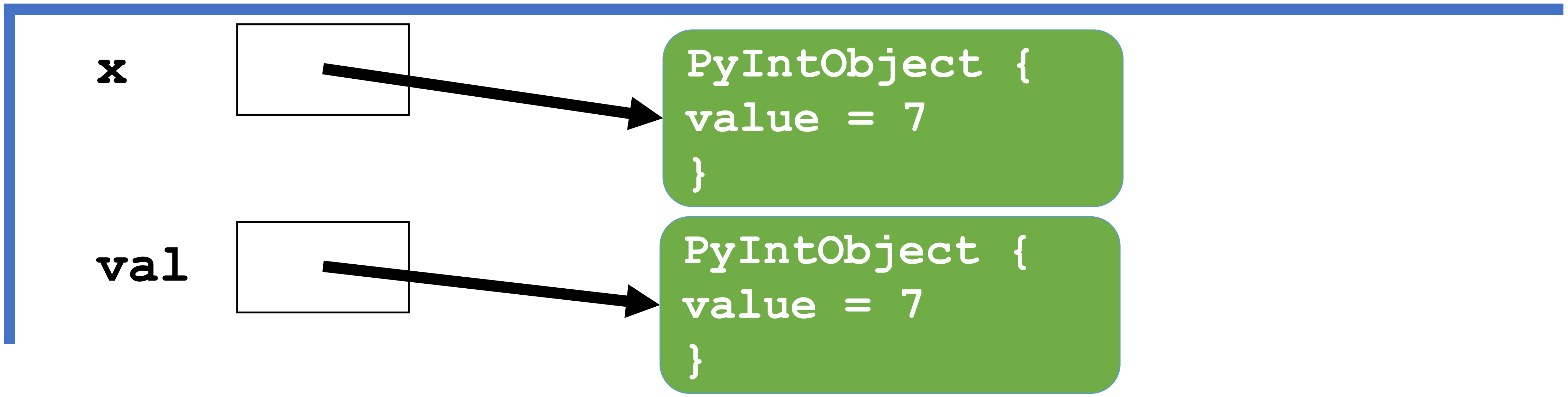


python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

```
7
```

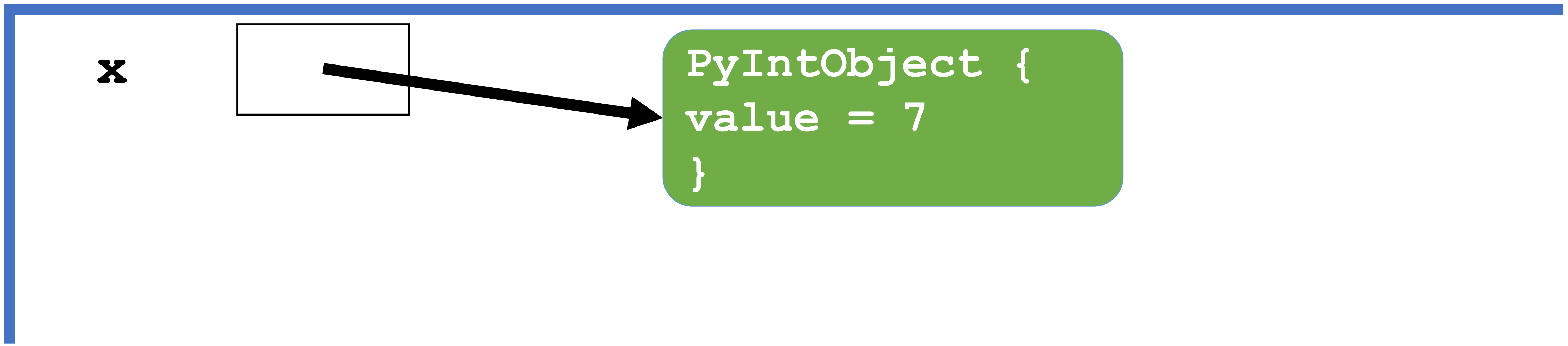


python

```
def inc_val(val):
     val = val+1


x = 7
inc_val(x)
print(x)
```

7



Function Quiz Explanation

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

Function 1

```
def my_abs(val):
    if val < 0:
        return 0-val
    return val
```

Function 2

```
def my_abs(val):
    if val < 0:
        print 0-val
    else:
        print val
```

Which function returns the absolute value of "val"?

A. **Function 1**
B. **Function 2**
C. **Both**
D. **Neither**

Function 1

```
def my_abs(val):
    if val < 0:
        return 0-val
    return val
```

Function 2

```
def my_abs(val):
    if val < 0:
        print 0-val
    else:
        print val
```

Which function returns the absolute value of "val"?

[✓ A. Function 1]
B. Function 2
C. Both
D. Neither

Function Quiz Explanation

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

```
def swap(val1, val2):
      tmp = val1
      val1 = val2
      val2 = tmp

x = 6
y = 3
swap(x, y)
print(x,”, “,y)
```

What is printed?

A. **6, 3**
B. **3, 6**
C. **3, 3**
D. **6, 6**

```
def swap(val1, val2):
      tmp = val1
      val1 = val2
      val2 = tmp

x = 6
y = 3
swap(x, y)
print(x,”, “,y)
```

What is printed?

- [✓ **A. 6, 3**]
- **B. 3, 6**
- **C. 3, 3**
- **D. 6, 6**

Scope in Python

Dr. Ilkay Altintas and Dr. Leo Porter

Twitter: #UCSDpython4DS

By the end of this video, you should be able to:

- Apply scoping rules to understand the lifetime of a variable
- Create a global variable

python

```
def my_abs(val):
      if val < 0:
            return 0-val
      return val

print(val)
```

python

```
def my_abs(val):
      if val < 0:
            return 0-val
      return val

print(val)
```

```
Traceback (most recent call last):
  File "scope1.py", line 6, in <module>
    print(val)
NameError: name 'val' is not defined
```

python

```
val = 0
def my_abs(val):
        if val < 0:
                return 0-val
        return val

print(val)
```

Beware, generally bad practice

python

```
my_val = 0
def my_abs(val):
        if val < 0:
                return 0-val
        return val

print(my_val)
```

"my" denotes global