Lecture 20: References

CS51P

November 20, 2019

Creating a new objects from old objects

```
lst1 = [0,{1:2}]
lst2 = lst1
lst3 = lst1[:]
lst4 = lst1.copy()

lst1[0]=3
lst1[1][1]=4
```

What are the final values of **lst1**, **`lst2`**, **`lst3`**, and **`lst4`**?

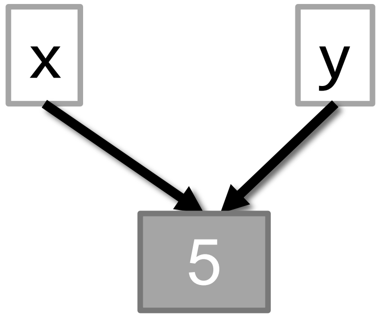
References

Primitive Types

- int
- float
- bool
- string

```
x = 5
y = 5
```

```
>>> x == y
True
>>> x is y
True
```

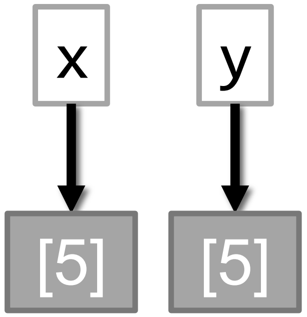


Objects

- tuple
- list
- dictionary
- more to follow…

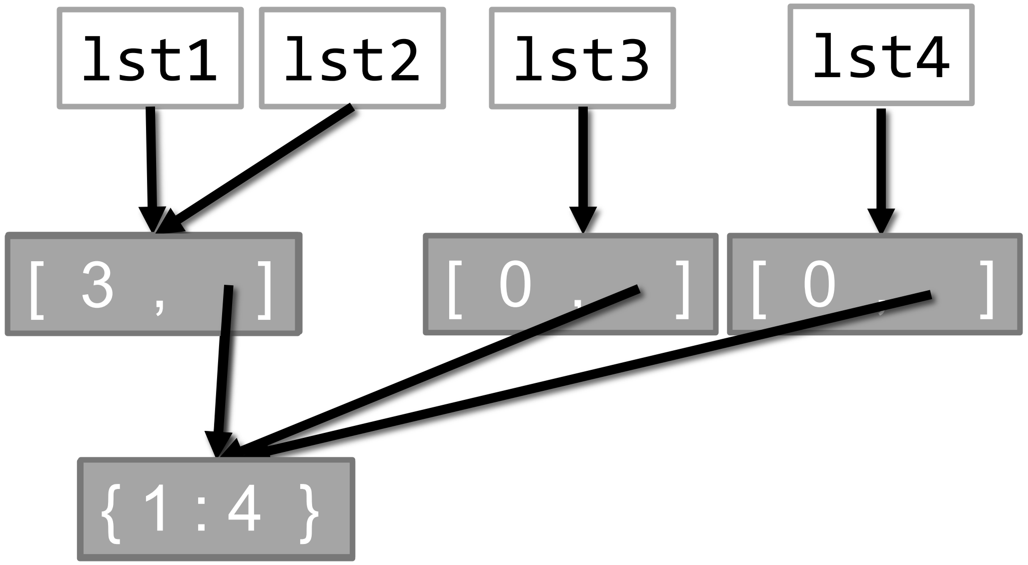
```
x = [5]
y = [5]
```

```
>>> x == y
True
>>> x is y
False
```



References

```
lst1 = [0,1]
lst1 = [0,{1:2}]
lst2 = lst1
lst3 = lst1[:]
lst4 = lst1.copy()
```



This sort of copy is called a **shallow copy**

```
lst1[0] = 3
lst1[1][1] = 4
```

`is` keyword

- you can use the keyword **`is`** to test whether two variables store the same object or different objects

```
>>> 5 is 5
True
>>> lst1 is lst2
True
>>> lst1 is lst3
False
>>> lst1 is lst4
False
```

Exercise

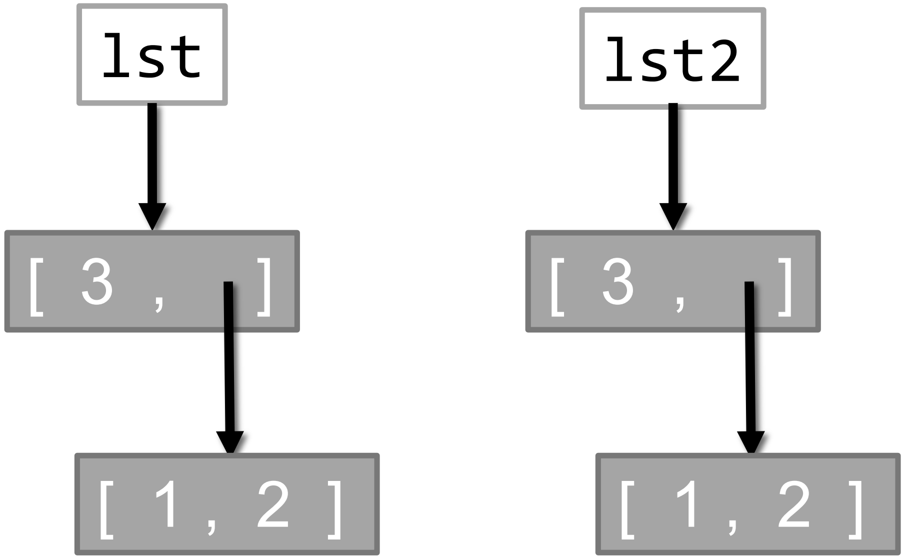
```
lst1 = [[0],1,[[2]]]
lst2 = lst1.copy()
lst2[2] = lst1
lst1[0][0]=3
lst1[1] = 4
print(lst2)
```

What gets printed in the final line of this program?

Deep Copy

- values = [int, float, bool, str]
- if type(item) in values, add item to new copy
- else, add a deep copy of item to new copy

```
lst2 = deep_copy(lst)
```



Examples of Objects

- list
- tuple
- dict
- file
- range
- …
- **Invent your own!**

Handling Errors

```
def example1(filename):
    s = 0

    file = open(filename, "r")
    for i in file:
        s = s + int(i)
    file.close()

    print(s)
```

- what if the file doesn't exist?
- what if it does exist but you don't have access permissions?
- what if the file exists and you can open it for reading, but it doesn't contain integers?

Exceptions

- A flexible mechanism for handling errors

```
try:

ex
```

```
try:

ex

ex
```

```
try:
    # code to execute
except <Error1>:
    # what to do if Error1 occurs
except <Error2>:
    # what to do if Error 2 occurs
else:
    # additional code if no errors
```

```
def exception_v1(filename):
    s = 0

    try:
        file = open(filename, "r")
        for i in file:
            s = s + int(i)
    except IOError:
        print("problem opening file")
    except ValueError:
        print("problem with non-integer")
        file.close()
    else:
        file.close()

    print(s)
```

- Write a function return_int that repeatedly asks the user to enter an integer. Once the user enters an integer the function returns that integer.

- Use exceptions to handle the case where the user does not enter an integer.

Final Project

- Use a real-world dataset to evaluate one or more hypotheses

- Example, given a dataset about AirBnB:
  - Is there any correlation between the price of a listing and the overall satisfaction?
  - Do hosts typically have multiple listings at the same time?
  - How do the prices of a rental change over time?

- Short write-up due Sunday night
  - what are your hypotheses?
  - link to dataset(s) you plan to use

- Meet with instructors during lab next week to discuss

- Due: Friday, December 13 at 5pm