RECORDING IN-PROGRESS

Day 3: Lists

Wednesday 7/14/21

RECORDING IN-PROGRESS

Today’s agenda

1. Daily check-in
2. Recap from yesterday
3. Working with Lists
   a. Creating lists
   b. Adding to lists
   c. Removing from lists
   d. Finding items in lists
4. Live coding

Daily check-in

Check in

- Canvas announcements
  - Zoom will now have a waiting room
  - You don’t need a Tufts.edu account anymore for our Zoom room
  - If you’re having problems receiving emails from your Tufts account, Angelie is posting a solution to Canvas today
- Lab 2
  - Questions first?
  - Go over solution - volunteer?
  - Make sure you’re submitting in a timely manner
- Questions?

RECAP

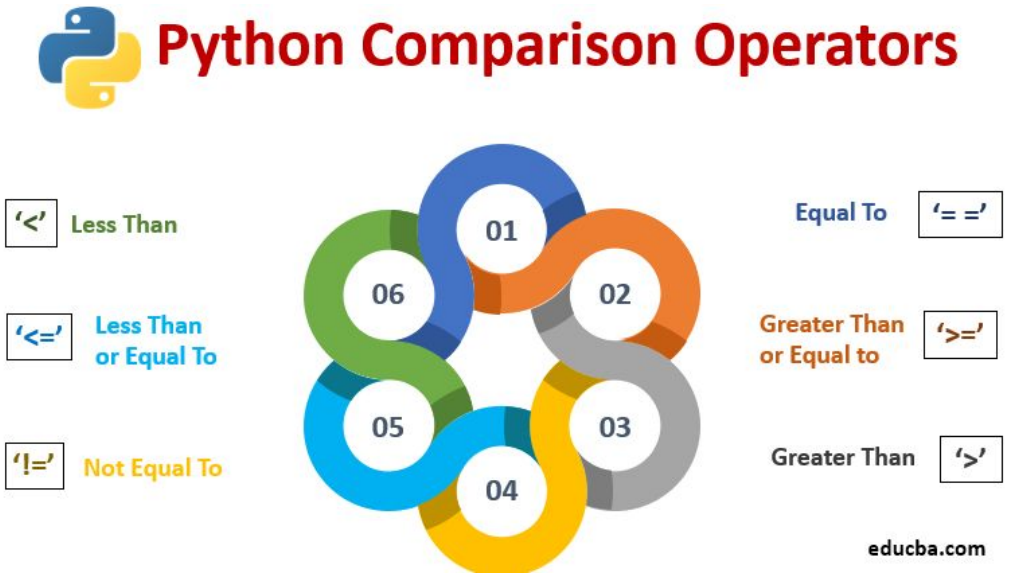
Booleans

Name	Data Type	Description
Integer	int	a number that can be written without fractions: 22 10 0 -300
Boolean	bool	logical value that indicates **True** or **False**
Floating-point	float	a number that has a decimal component: 3.14 2.73 10.0
String	str	sequence of characters: "hello world" "hey88340" '2018'

Bools are a type (like int, float, str) that can only be True or False

Comparison operators

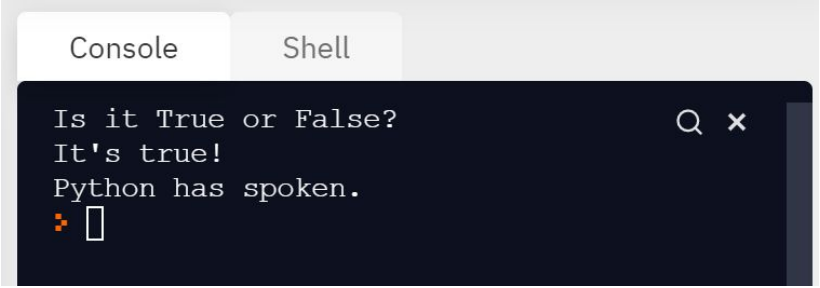
There are 6 Python comparison operators, which we use to compare multiple values using a logical statement, like 5 > 2. The statement will evaluate to either True or False.



Conditionals

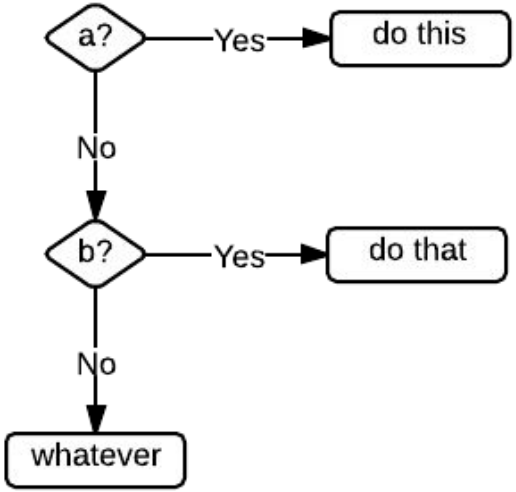
We can use if-statements to control the flow of our program. When an if-statement passes, or evaluates to True, the indented code belonging to that if-statement will be executed. However, if an if-statement does not pass, or it evaluates to False, then the code will not get executed.

```
11  print("Is it True or False?")
12
13  if (True):
14      print("It's true!")
15
16  if (False):
17      print("It's false!")
18
19  print("Python has spoken.")
```



Conditionals with if, elif, else

```
if a:
    do this
elif b:
    do that
else:
    whatever
```



Python 3.4.1: dogs litter number.py - //vmware-host/Shared Fo

File Edit Format Run Options Windows Help

```
litter = int(input("How many puppies were born?"))

if litter <= 5:
    print("good size")
elif litter == 6:
    print("just right")
elif litter == 7:
    print("large litter")
else:
    print("goodness me")
```

Every if-statement will get evaluated in Python. However, if you use if/elif/else, only *one* of the if, elif, else statements will get evaluated.

New topic: Lists

So far in our code, we've seen how we can create variables and assign them a single value, like an int, str, float, bool, etc.:

X = True

Y = 5.14

message = "Hello World!"

How would we represent values on a shopping list?

item1 = “Eggs”

item2 = “Milk”

item3 = “Butter”

item4 = “Cheese”

...

..

.

shoppingList = [“Eggs”, “Milk”, “Butter”, “Cheese”, ...]

Lists

Unlike previous Python types, which represented a single **int**, **float**, **str**, **bool**, etc...

```
player1 = “Mario”
player2 = “Luigi”
X = 3.14
```

Lists - Python type that lets us stores **multiple** pieces of data (a collection of values)

```
players = [“Mario”, “Luigi”, “Peach”]
```

Creating a List in Python

(1) Name the list. Use camelCase if it's more than one word

```
players = [“Mario”, “Luigi”, “Peach”]
```

(2) The '=' symbol assigns the data on the right into the list, like any other variables

(3) All the items in a list are surrounded by square brackets with a comma between each one

Each item is assigned a number in the list, starting at 0

```
players = [“Mario”, “Luigi”, “Peach”]
```

0 1 2

Each item is assigned a number in the list

```
players = [“Mario”, “Luigi”, “Peach”, “Daisy”, “Yoshi”]
```

0 1 2 3 4

Facts about lists:

- Lists can be assigned to variables
- You can have as many items as you want in a List
- The items in the List are ordered
- The first item in a List starts at the number (aka index) 0

Outputs from a List

Output an item from a List

```
players = [“Mario”, “Luigi”, “Peach”]

print(players[2])
```

(1) Use the print statement

(2) Put the name of the list inside the normal brackets

(3) Put the **index** (number) of the item you want to output **in square brackets**

Output an item from a List

```
>>> players = [“Mario, “Luigi”, “Peach”, “Daisy”, “Yoshi”]
>>> print(players[0])
?
>>> print(players[1])
?
>>> print(players[4] + “ and ” + players[0])
?
```

Output an item from a List

```
>>> players = [“Mario, “Luigi”, “Peach”, “Daisy”, “Yoshi”]
                 0        1        2        3        4
>>> print(players[0])

Mario

>>> print(players[1])

Luigi

>>> print(players[4] + “ and ” + players[0])

Yoshi and Mario
```

Change & edit items in a List

Update an item from a List

```
players = [“Mario”, “Luigi”, “Peach”]

players[0] = “Bowser”
```

(1) The item in the list to be replaced

(2) The ‘=’ symbol used for assignment

(3) The new data to go into the list

Output an item from a List

```
>>> players = [“Mario, “Luigi”, “Peach”]
>>> players[0] = “Bowser”
>>> print(players)
[“Bowser, “Luigi”, “Peach”]
>>> players[1] = players[2]
>>> print(players)
[“Bowser, “Peach”, “Peach”]
```

Add & remove items from a List

Add an item to a list using append(x)

```
players = [“Mario”, “Luigi”, “Peach”]

players.append(“Bowser”)
```

(1) Adding a new item to the list ‘players’

(2) .append(...) adds whatever’s inside the parentheses to the ‘players’ list

(3) The new item to get added into the ‘players’ list

Add an item to a list using append(x)

```
>>> players = [“Mario”, “Luigi”, “Peach”]
>>> players.append(“Bowser”)
>>> print(players)
[“Mario”, “Luigi”, “Peach”, “Bowser”]
```

The list now contains the string ‘Bowser’. Lists can contain any **type** of item, as in: ints, floats, booleans, etc.

Find an item in a List

Finding an item in a list

```
players = [“Mario”, “Luigi”, “Peach”]
```

- **Is “Bowser” in my ‘players’ list?**
- **Is “Mario” in my ‘players’ list?**
- **Is “Daisy” in my ‘players’ list?**

Find an item in a list, then…

This is a conditional statement. "**If** this **item** is in my list, **then** do something."

```
if <item> in <list>:
    <Run this code>
```

If the **item** is in the list, the code will run

Find an item in a list, then…

```
players = [“Mario”, “Luigi”, “Peach”, “Daisy”, “Yoshi”]
players.remove(“Luigi”)
if “Luigi” in players:
    print(“Luigi in the house!”)
else:
    print(“Luigi is missing!”)
# What will get printed?
```

Find an item in a list, then…

```
players = [“Mario”, “Luigi”, “Peach”, “Daisy”, “Yoshi”, “Luigi”]
players.remove(“Luigi”)
print(players)
[“Mario”, “Peach”, “Daisy”, “Yoshi”, “Luigi”]
```

Live coding in Replit