Day 4: Loops

Thursday 7/15/21

Daily check-in

RECORDING IN-PROGRESS

Agenda

- Daily check-in
- Recap from yesterday
- Loops
  - While loops
  - For loops
- Examples
- Live coding

Check in

- Questions? Thoughts?
- Announcements
  - Angelie is posted a solution on Canvas for people who aren't receiving emails to their Tufts inbox
  - Reminder of how to find CAs during their office hours
    - In their Sococo office between 5-6pm EST and 7:30-8:30am EST
- Lab 3
  - Questions first?
  - Go over solution
    - Ashley will review the short answer questions
    - Volunteer for part 2?
- Questions?

RECAP

We learned about a new type called **Lists**

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

Lists can contain any type of element inside of them, including ints, bools, floats, strings, etc.

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

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
```

Adding items to a List using append() and insert()

```
>>> players = [“Mario, “Luigi”, “Peach”, “Daisy”, “Yoshi”]
>>> players.append(“Wario”)
>>> players
[“Mario, “Luigi”, “Peach”, “Daisy”, “Yoshi”, “Wario”]
>>> players.insert(0, “Waluigi”)
>>> players
[“Waluigi”, “Mario, “Luigi”, “Peach”, “Daisy”, “Yoshi”, “Wario”]
```

Removing items from a list using pop() and remove()

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

Remove an item from a list, then check if it’s in the list still

```
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?
```

New topic: Loops

What is this “while” statement doing from yesterday’s lab?

```
173   userIsShopping = True
174   shoppingList = []
175
176   while userIsShopping:
177       print("Shopping cart menu: \n" +
178             "(1) Print your shopping cart\n" +
179             "(2) Add an item to your shopping cart\n" +
180             "(3) Remove an item from your shopping cart\n" +
181             "(4) Quit\n" +
182             "(5) Print a specific item from your shopping cart")
```

Loops

Loops let us continue to do something in our code, repeatedly, until we tell the program to stop / quit.

```
173  userIsShopping = True
174  shoppingList = []
175
176  while userIsShopping:
177    print("Shopping cart menu: \n" +
178          "(1) Print your shopping cart\n" +
179          "(2) Add an item to your shopping cart\n" +
180          "(3) Remove an item from your shopping cart\n" +
181          "(4) Quit\n" +
182          "(5) Print a specific item from your shopping cart")
```

```
210  elif userInput == 4:
211    userIsShopping = False
```

Loops

While loops - Loops over some code until a given condition evaluates to false

For loops - Loops through a collection of data (like lists) until all data has been looped over

Iteration

```
while <boolean condition>:
    Do this
```

Iteration

Starts the iteration statement.

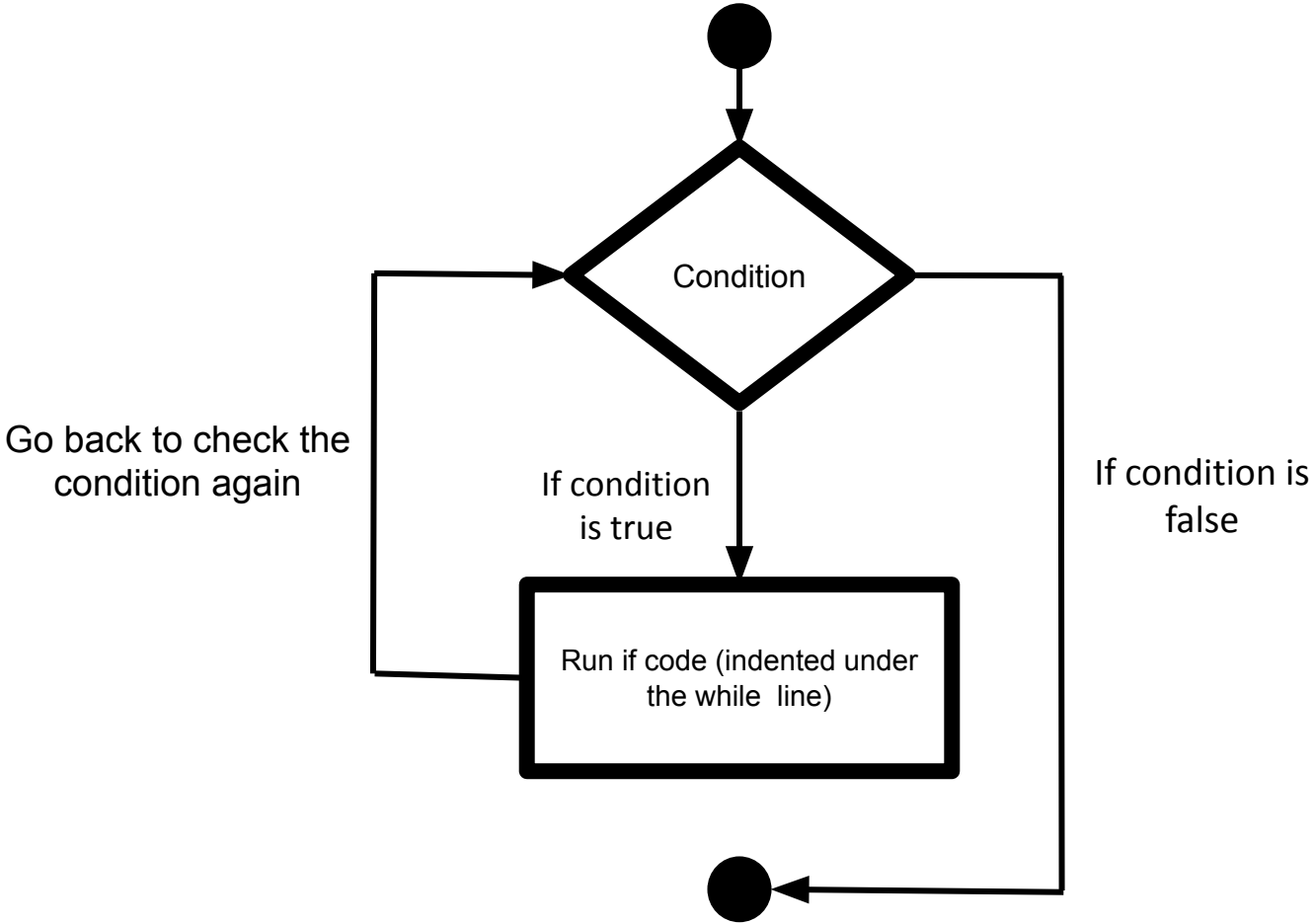
':' after the condition, just like if-statements

```
while <boolean condition>:
    Do this
```

This instruction is repeated while the condition is TRUE.

Indented - use the tab key. Your code won't work if you don't indent inside the iteration statement.

Iteration With While - Flowchart



Iteration with while - coding tips

```
answer = input("What is the capital of France?")

while answer != "Paris":
    print("Incorrect! Try again.")
    answer = input("What is the capital of France?")

print("Correct!")
```

Iteration with while - coding tips

0. Not part of the loop (comes before ‘while’). Runs before the loop starts.

2. Write the Boolean condition (usually what you DON’T want the user to input).

```
answer = input("What is the capital of France?")

while answer != "Paris":
    print("Incorrect! Try again.")
    answer = input("What is the capital of France?")

print("Correct!")
```

1. Start with while.

3. Code that is repeated while the condition is **true.**

Not part of the loop (not indented). Runs after the loop ends.

Getting stuck in while loops

```
counter = 1

while counter < 5:
   print(“Hello”)

print (“The loop has ended”)
# Has it?
```

Getting unstuck from while loops

```
counter = 1

while counter < 5:
    print(“Hello”)
    counter = counter + 1

print (“The loop has ended”)
# How do we know?
```

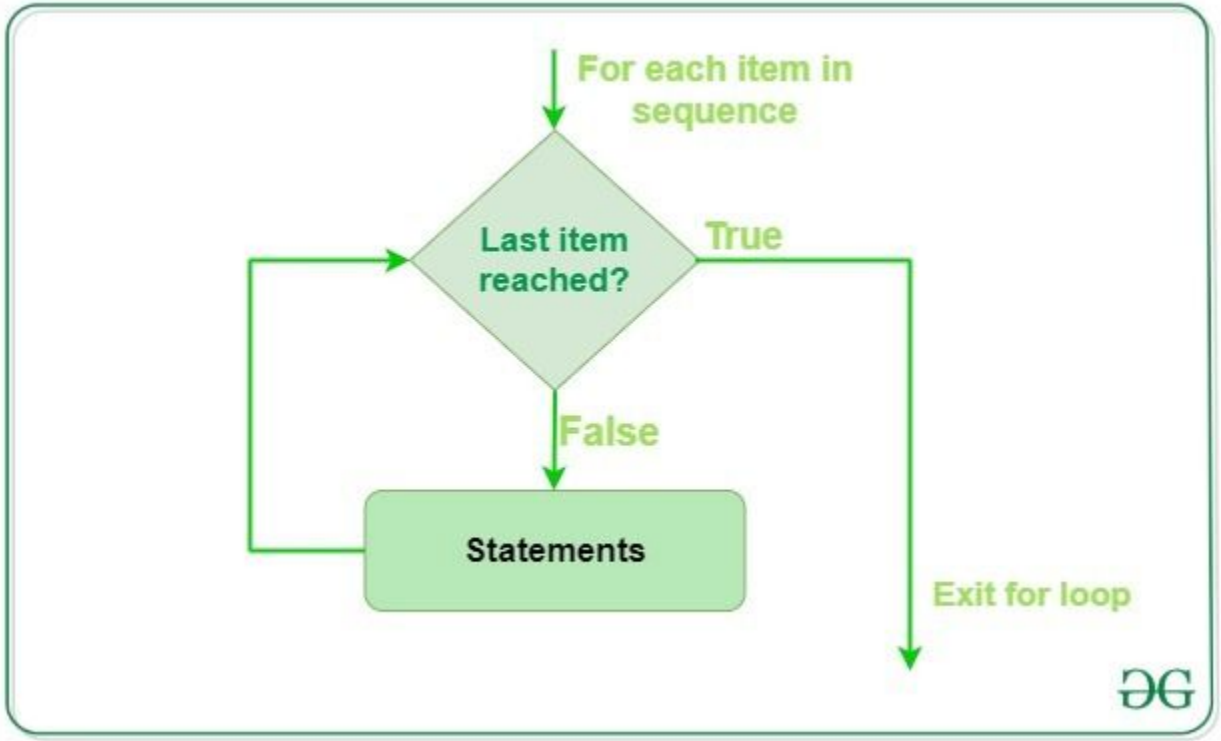
For loops

For Loops

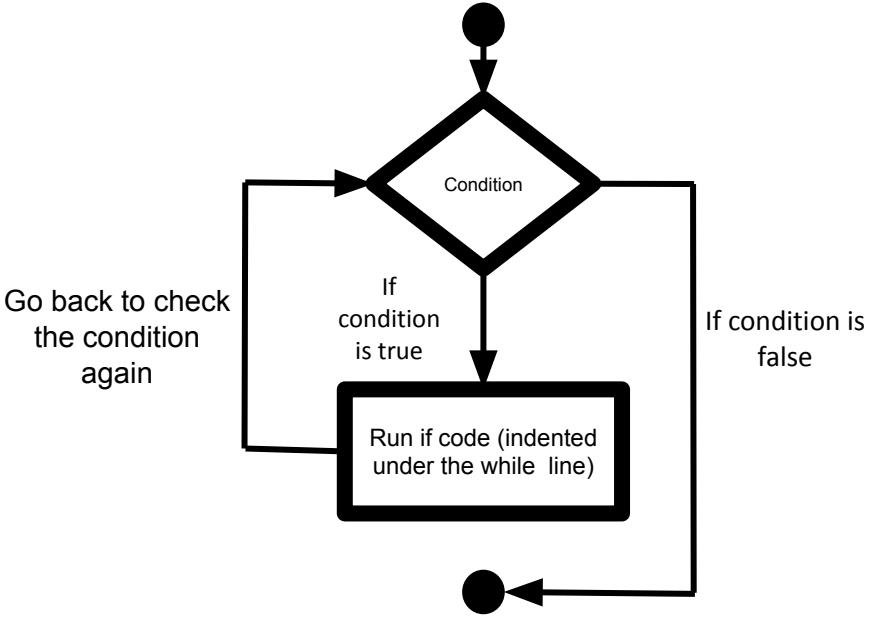
- **While** loops are great for repeating a block of code until the condition written in the while loop is no longer true
- **For** loops, on the other hand, are good for “sequential travel” or “sequential traversal”. It will let you repeat the code until all of the elements have been traversed

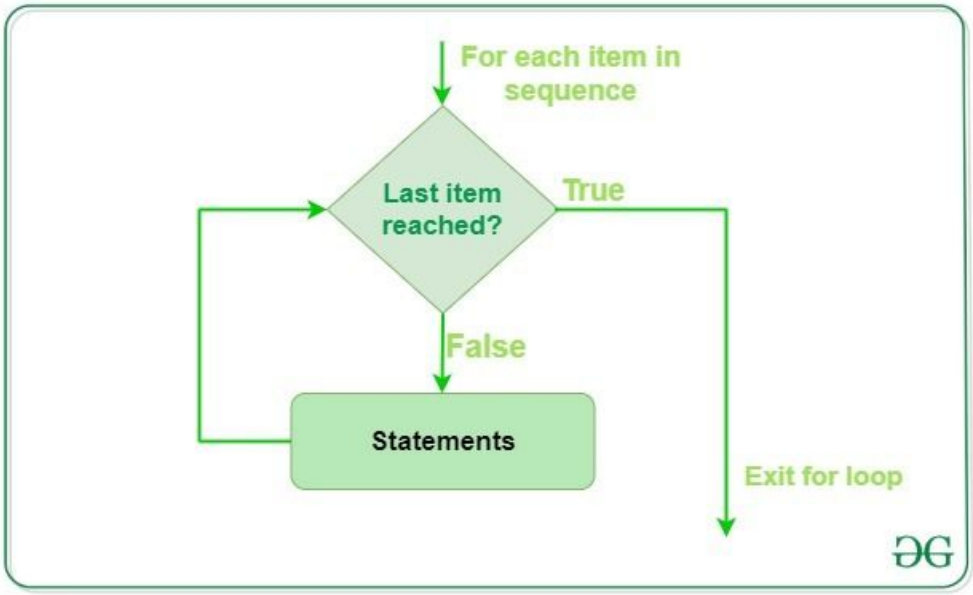
For loops - Loops through a collection of data (like lists) until all data has been looped over

For loops


For each item in sequence
Last item reached?
True
False
Statements
Exit for loop

For loops


Condition
If condition is true
Go back to check the condition again
If condition is false
Run if code (indented under the while line)


For each item in sequence
Last item reached?
True
False
Statements
Exit for loop

For each item in my list, do this

<variable> can be anything you want, it's a new variable you're declaring for the **for** loop

```
for <variable> in <collection>:
    <Run this code>
```

For every item in the list (or every variable in the collection) this code will run

<collection> is usually a List, but it can also be a range of numbers (similar to if you wanted to loop over a list of ints)

Using a **for** loop with a list

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

for p in players:

    print(“My favorite character is: “ + p)


What would the output be?
```

Using a **for** loop with a list

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

for playerName in players:

    #print(“My favorite character is: ” + playerName)

    print(f“My favorite character is: {playerName}”)
```

What would the output be?

```
My favorite character is Mario

My favorite character is Luigi

My favorite character is Peach

My favorite character is Yoshi
```

Live coding in Replit