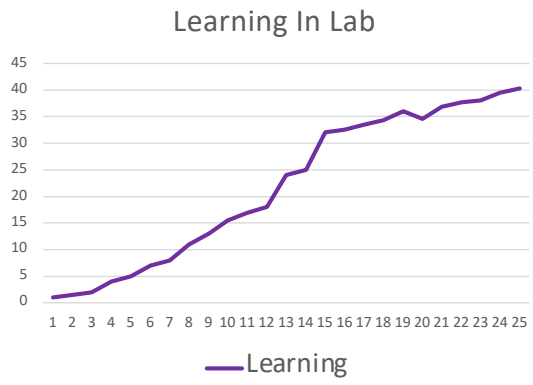
Objectives

- Indefinite Loops

1

Your Learning Journey

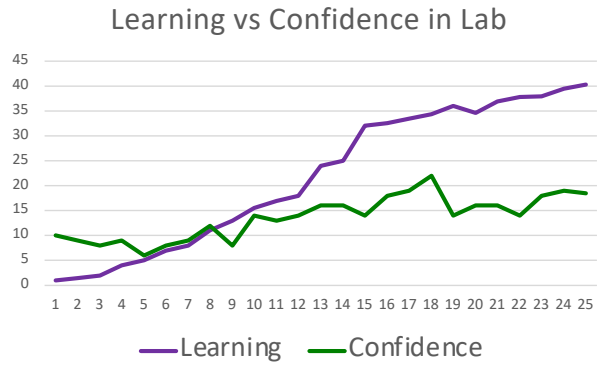
- You're learning a lot
  - Struggle is part of the learning



2

Your Learning Journey

- But struggle affects your confidence
  - Confidence != Learning

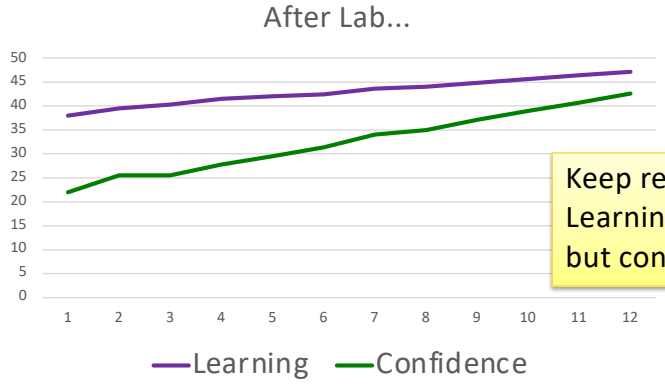


Lab ends and … ???

3

Your Learning Journey

- But struggle affects your confidence
  - Confidence != Learning



Keep reviewing, practicing
Learning may not increase as much, but confidence should

4

Review

- We are judging a science fair.
- There is different criteria for winning a first place ribbon, depending on what grade the student is in.
- Given the variables `scienceScore` and `grade`
  - Write a condition that will evaluate to True if (and only if) the student's score is above the first place threshold of 60 points and the student's grade is 8.
    - Otherwise, the condition should evaluate to False

Review

- We are judging a science fair.
- There is different criteria for winning a first place ribbon, depending on what grade the student is in.
- Given the variables `scienceScore` and `grade`
  - Write a condition that will evaluate to `True` if (and only if) the student's score is above the first place threshold of 60 points and the student's grade is 8.
    - Otherwise, the condition should evaluate to False

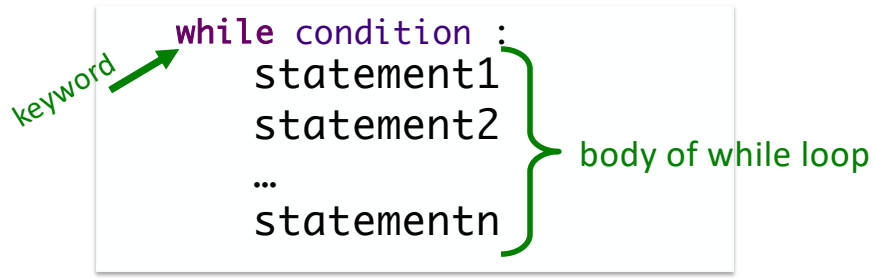
```
scienceScore > 60 and grade == 8
```

INDEFINITE LOOPS

Definite vs Indefinite Loops

- `for` loops are ***definite*** loops
  - Execute a *fixed* number of times
- ***Indefinite*** loops: keep iterating until certain conditions are met
  - Depending on condition, no guarantee in advance of how many times the loop body will be executed

While Loop Syntax



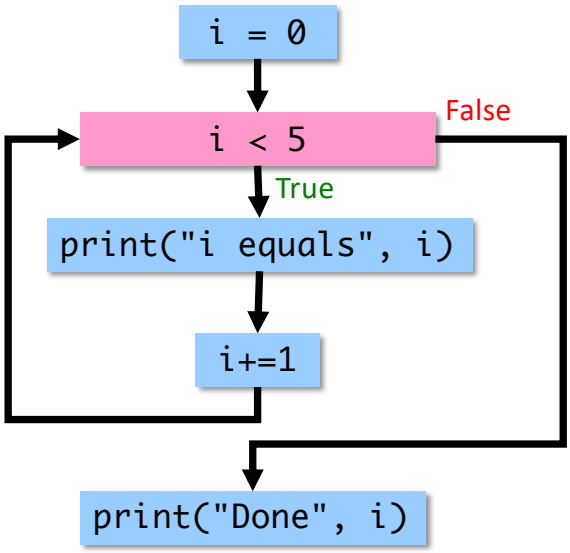
- Like a *looped* `if` statement
  - Execute statements **only** when condition is true
  - Stop executing when condition is false

While Loop

```
i = 0
while i < 5 :
    print("i equals", i)
    i+=1
print("Done", i)
```

While Loop

```
i = 0
while i < 5 :
    print("i equals", i)
    i+=1
print("Done", i)
```



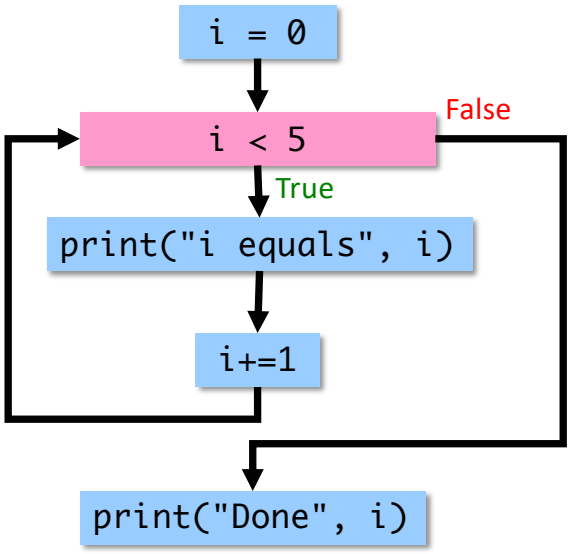
Questions:

- How many times will `i` get printed out?
- How many times is the condition evaluated?
- What is the value of `i` after the loop?

While Loop

```
i = 0
while i < 5 :
    print("i equals", i)
    i+=1
print("Done", i)
```

Initialize i before using in condition



Questions:

- How many times will `i` get printed out?
- How many times is the condition evaluated?
- What is the value of `i` after the loop?

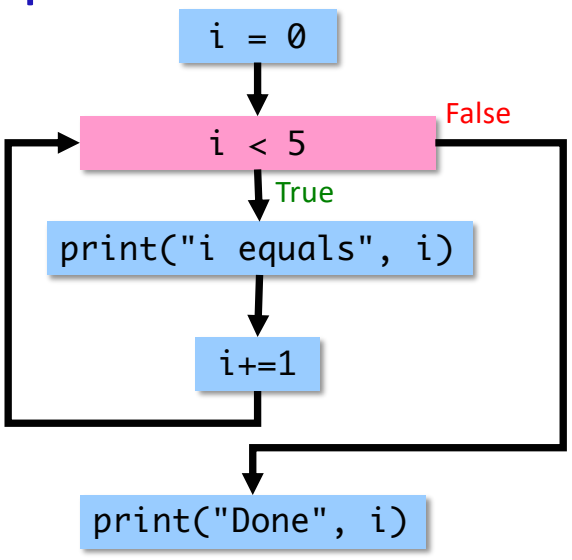
While vs. For Loops

- ***Any*** `for` loop can be translated into a `while` loop
- But **NOT** vice versa
  - Only ***some*** `while` loops can be converted into `for` loops

- `while` loops are more ***powerful*** than `for` loops

Convert to a `for` loop

```
i = 0
while i < 5 :
    print("i equals", i)
    i+=1
print("Done", i)
```



We ***can*** convert this `while` loop into a `for` loop because it executes a ***fixed*** number of times.

Comparing while and for

- What are the main differences between these loops?
- What are the advantages and disadvantages of each?

```
i = 0
while i < 5 :
    print("i equals", i)
    i+=1
print("Done", i)
```

```
for i in range(5):
    print("i equals", i)

print("Done", i+1)
```


What Will This Loop Do?

```
count = 1
while count > 0:
    print(count)
    count += 1
```

Infinite Loop

- Condition will never be `False` so keeps executing

```
count = 1
while count > 0:
    print(count)
    count += 1
```

- To stop an executing program in Linux use
  - Control-C

Infinite Loop Discussion

- Is there ever a time that an infinite loop is wanted?
  - Yes! For example in web servers, we have something like

```
while True:
      listenForRequest()
      handleRequest()
```

- Can a computer automatically detect infinite loops?
  - No, that is an ***undecidable*** problem
  - Best to **prevent** infinite loops (more later)
    - Benefit of `for` loops: *definite* loops

A Very Simple Therapist

- Whenever a user tells the computer/program what they think, the program asks, "How does that make you feel?"
- Ends when user enters nothing ("")

A Very Simple Therapist

- Whenever a user tells the computer/program what they think, the program asks, "How does that make you feel?"
- Ends when user enters nothing ("")
- Partial example output:

```
Tell me what is bothering you.
There is too much going on in my life.
How does that make you feel?
I feel like I am out of control and can't juggle it all.
How does that make you feel?
Really stressed and tired.
How does that make you feel?

Thank you!  Come again!
```

therapist.py

Design Pattern: Sentinel Loop

- Sentinel: when to stop
  - "guard" to the loop

```
value = set value
while value != sentinel :
    process value
    value = set value
```

- "Keep going until you see the sentinel"
- Options for "set value"
  - From a simple assignment, user input, calling a function, reading from a file, …

`while` Loops: Sentinel Loop

```
# condition says when loop will continue
x=eval(input("Enter number:"))
while x % 2 != 0 :
  print("Error!")
  x = eval(input("Enter number: "))
print(x, "is an even number.")
```

Example of a `while` loop that cannot be transformed into a `for` loop
(Why not?)

`while` Loops: comparing use of `break`

```
# condition says when loop
# will continue
x=eval(input("Enter number:"))
while x % 2 != 0 :
  print("Error!")
  x = eval(input("Enter number: "))
print(x, "is an even number.")
```

Loop condition says when to keep going

```
# have to look inside loop to
# know when it stops
while True :
  x = eval(input("Enter number:"))
  if x % 2 == 0 :
    break        "breaks" out of a loop
  print("Error!")
print(x, "is an even number.")
```

Internal condition says when to stop

Using break statements:
Best when loop has to execute at least once.

Summary: While vs. For Loops

- ***Any*** `for` loop can be translated into a `while` loop
- But **NOT** vice versa
  - Only ***some*** `while` loops can be converted into `for` loops

- `while` loops are more ***powerful*** than `for` loops

Flipping Coins

- Problem: How many flips does it take to get 3 consecutive heads?
- Given: In the *game* module:
  - Constants: HEADS and TAILS
  - Function:
    - How can we simulate flipping a coin?

```
def flipCoin():
    """
    Simulates flipping a non-biased coin.
    returns either HEADS or TAILS.
    """
```

game.py

Flipping Coins

- Problem: How many flips does it take to get 3 consecutive heads?
- Given: In the *game* module:
  - Constants: HEADS and TAILS
  - Function: `flipCoin()`
- Challenge: Solve in two ways
  - without a break
  - with a break

game.py
consecutiveHeads.py

Looking Ahead

- Lab 5 due Friday
- Broader Issue: Autonomous Vehicles