Objectives

- Indefinite Loops

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
```

Short-circuit Evaluation

- Don't necessarily need to evaluate all expressions in a compound expression
- A **and** B
  - If A is `False`, compound expression is `False`
- A **or** B
  - If A is `True`, compound expression is `True`
- No need to evaluate B
  - Put more important/limiting expression first
  - Example:
    ```
    if count != 0 and sum/count > 10:
            do something
    ```

INDEFINITE LOOPS

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

while condition :
    statement1
    statement2
    …
    statementn

keyword

body of while loop

loop stops when condition is False

- Like a looped **if** statement
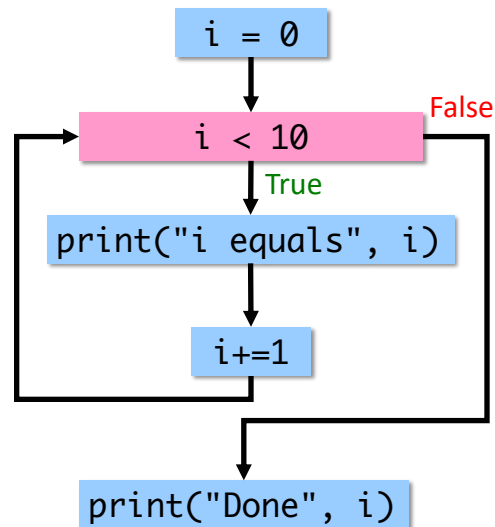 - ➢ Execute statements **only** when condition is true

---

While Loop

```
i = 0
while i < 10 :
    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?

i = 0

i < 10   False
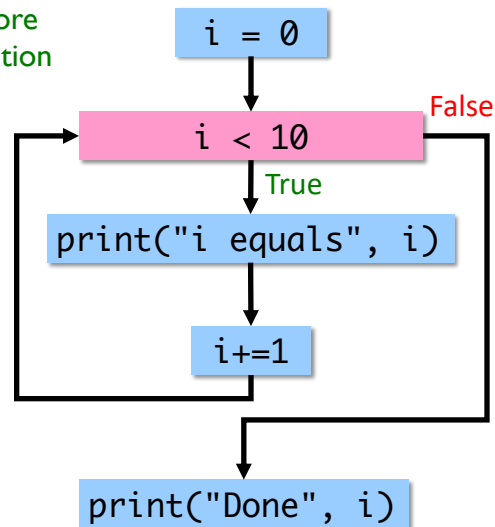
True

print("i equals", i)

i+=1

print("Done", i)

4

While Loop

```
i = 0
while i < 10 :
    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?

```
i = 0
```

```
i < 10
```
False

True

```
print("i equals", i)
```

```
i+=1
```

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

While vs. For Loops

- Any **for** loop can be translated into a **while** loop
- But **NOT** vice versa

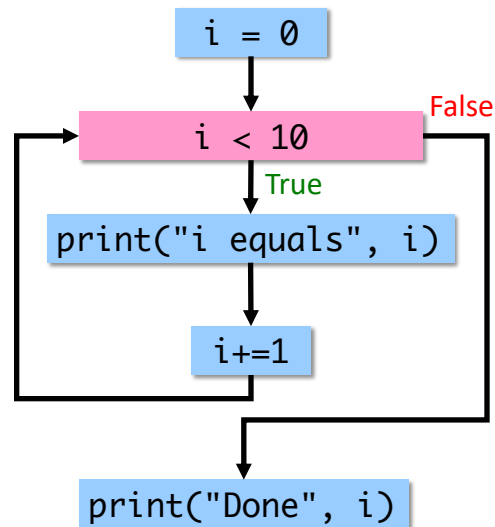➤ **while** loops are more **powerful** than **for** loops

Convert to a **for** loop

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

```
i = 0
while i < 10 :
    print("i equals", i)
    i+=1
print("Done", i)
```

i = 0

↓

i < 10   False

True ↓

print("i equals", i)

↓

i+=1

↓

print("Done", i)

Comparing `while` and `for`

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

```
i = 0
while i < 10 :
    print("i equals", i)
    i+=1
print("Done", i)
```

```
for i in range(10):
    print("i equals", i)

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

What Will This Loop Do?

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

Infinite Loop

- Condition will never be False so keeps executing

```python
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 ("")
- 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!
```

Design Pattern: Sentinel Loop

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

```
value = get input
while value != sentinel :
    process value
    value = get input
```

`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.")
```

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.")
```

Says when to stop

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

Transform Therapist

- Write using a break instead…
  - (on your own)

Summary: While vs. For Loops

- Any `for` loop can be translated into a `while` loop
- But **not** vice versa

- `while` loops are more **powerful** than `for` loops
  - Give an example of a `while` loop that can't be converted to a `for`

Flipping Coins

- Problem: How many flips does it take to get 3 consecutive heads?
  - How can we simulate flipping a coin?

game.py
consecutiveHeads.py

Looking Ahead

- Lab 5 due Friday
- Broader Issue: Self-driving Cars