

JamCoders: Week 1

Lecture 3A:

- While loops



While Loops

While Loops

Let's make a word guessing game...

```
secret_word = "bears"
for i in range(5):
    guess = input("Enter a word guess: ")
    if guess.lower() == secret_word:
        print("You got it!")
        break
    else:
        print("That's not right. Try again")
```

While Loops

What if 5 guesses isn't enough?

```
secret_word = "bears"
for i in range(5):
    guess = input("Enter a word guess: ")
    if guess.lower() == secret_word:
        print("You got it!")
        break
    else:
        print("That's not right. Try again")
```

While Loops

What if 5 guesses isn't enough?

```
secret_word = "bears"
for i in range(10):
    guess = input("Enter a word guess: ")
    if guess.lower() == secret_word:
        print("You got it!")
        break
    else:
        print("That's not right. Try again")
```

Make it 10!

While Loops

What if 10 guesses isn't enough?

```
secret_word = "bears"
for i in range(20):
    guess = input("Enter a word guess: ")
    if guess.lower() == secret_word:
        print("You got it!")
        break
    else:
        print("That's not right. Try again")
```

Make it 20!


While Loops

We never know how many guessing will be enough...really, what we want to do is keep repeating until the user guesses correctly.

Or in other words, keep repeating while the answer is incorrect...

While Loops

This expression gets
re-evaluated after
each iteration



```
while expression:  
    execute_code()  
    . . .
```

A while loop continues to repeat indefinitely as long as the condition expression evaluates to `True`.

While Loops

Back to the game...keep taking guesses until the guess is correct

```
secret_word = "bears"
guess = input("Enter a word guess: ")
while guess.lower() != secret_word:
    print("That's not right. Try again..")
    guess = input("Enter a word guess: ")
print("You got it!")
```

While loops let us repeat based on some condition

While Loops

You can also use a while loop with the True condition, called an **infinite loop**, along with Break, to solve this problem.

```
secret_word = "bears"
while True:
    guess = input("Enter a word guess: ")
    if guess.lower() == secret_word:
        print("You got it!")
        break
    else:
        print("That's not right. Try again..")
```

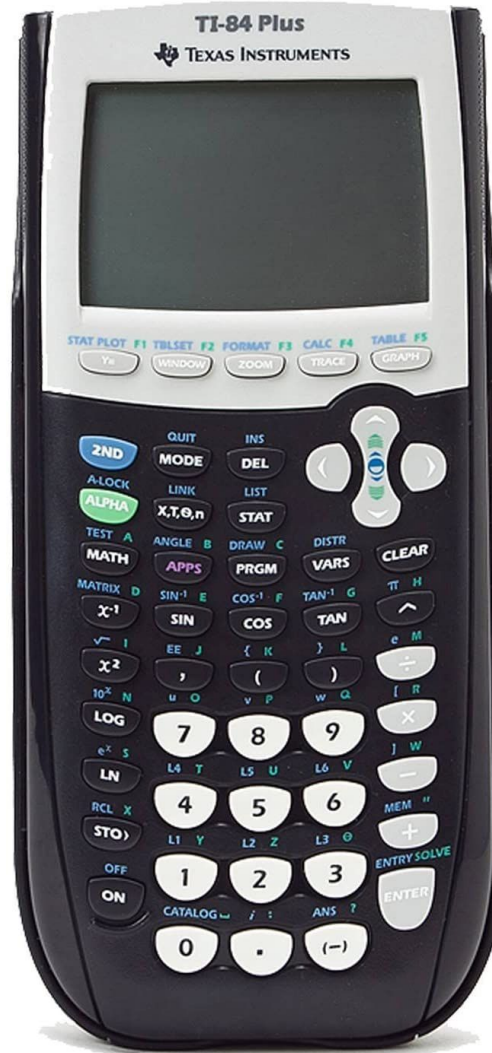
Break and continue work in both while and for loops.

While Loops

Infinite loops are useful for when you want to take input indefinitely with no stopping condition

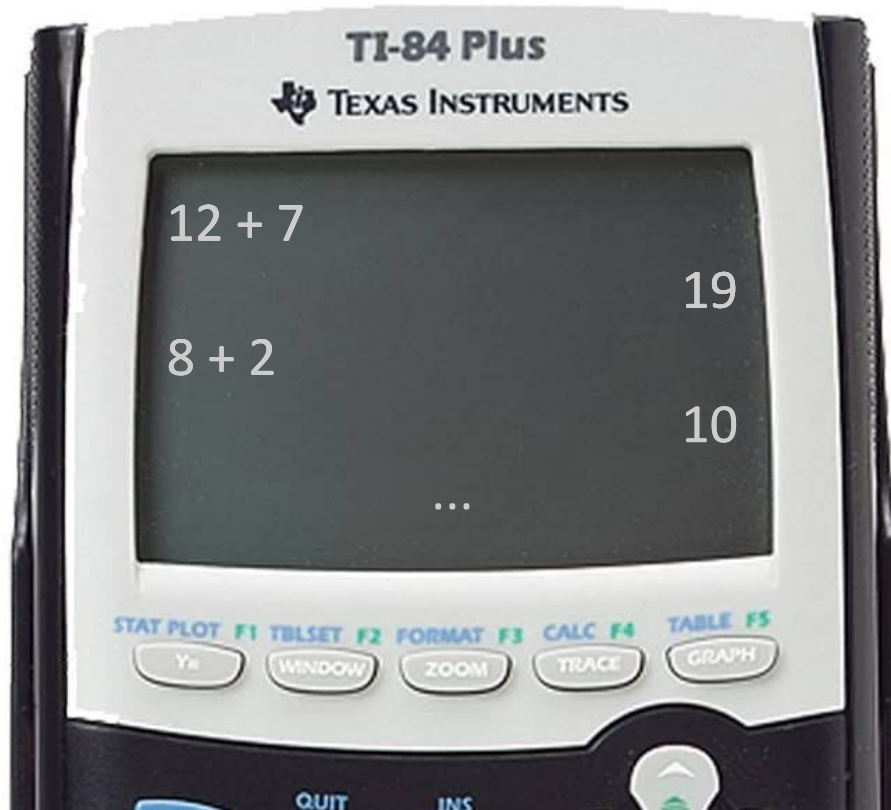
Repetition

Using a calculator



Repetition

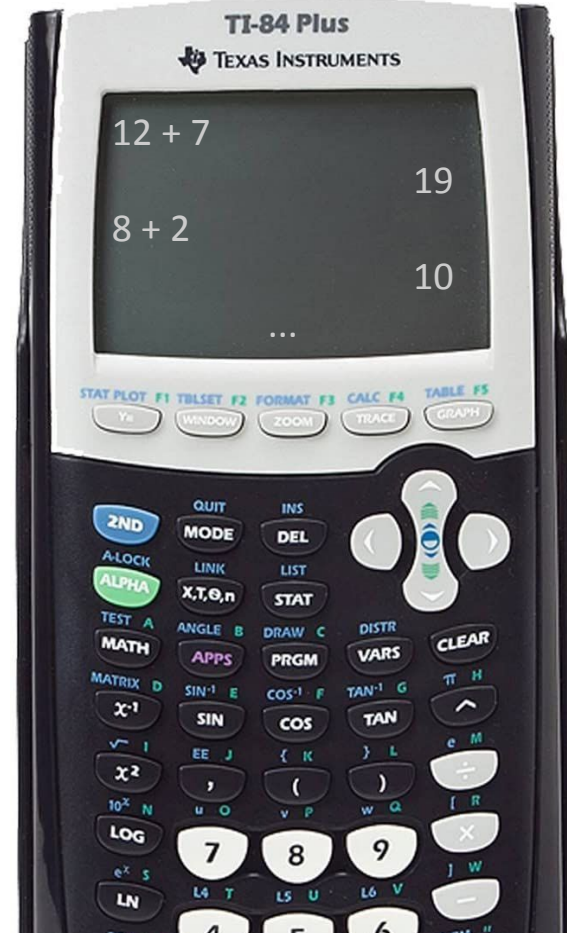
Using a calculator



Repetition

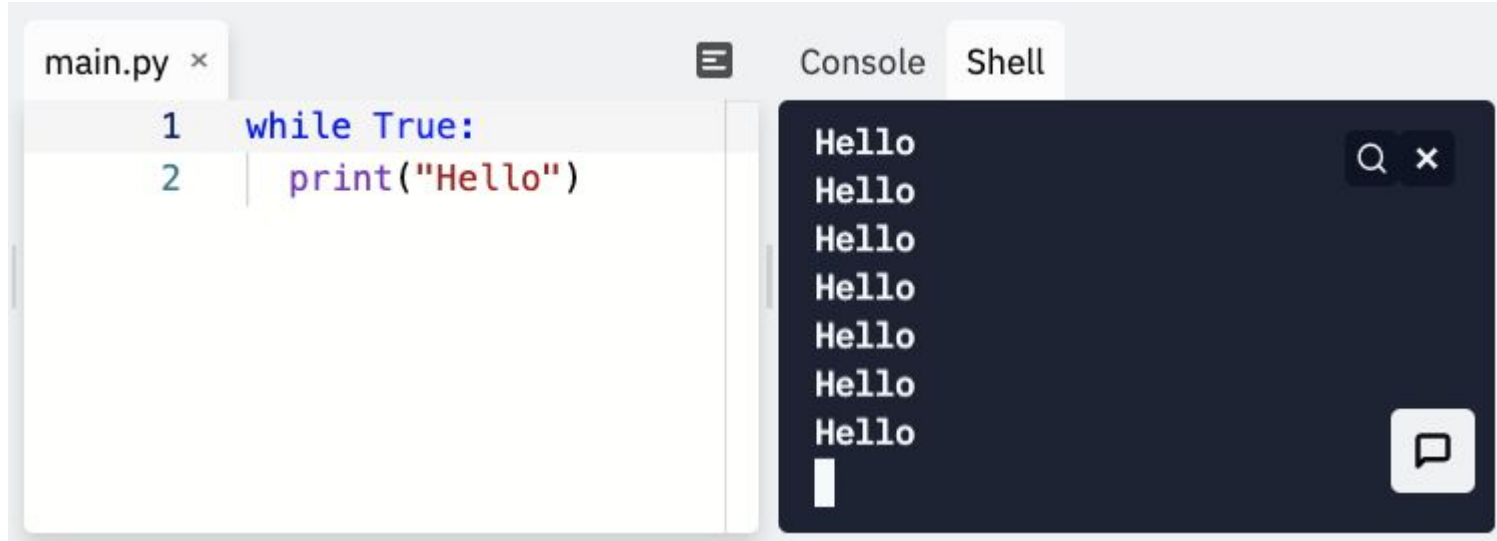
A calculator keeps asking for more input forever.

- Perfect use for a while loop

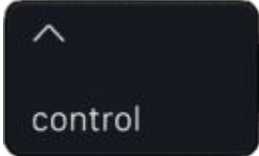



Help, I'm Stuck in an Infinite Loop!

If you're running Python code in the terminal and you get stuck in an infinite loop...

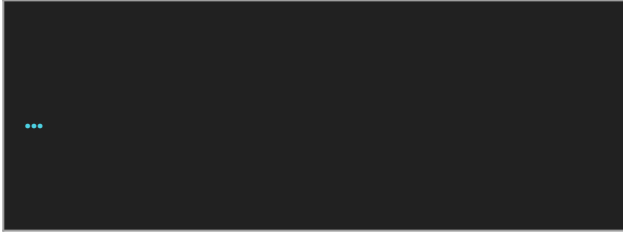


The screenshot shows a code editor with a file named `main.py`. The code contains a `while True:` loop that prints `"Hello"` repeatedly. To the right, a 'Console' window displays the output, showing the word `Hello` printed multiple times, illustrating the infinite loop.

Press  +  on Mac to stop execution (Ctrl + Z on Windows)

For vs While

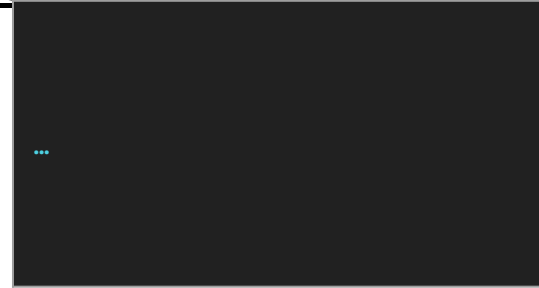
For loops



When you have a particular sequence you know you want to iterate over ahead of time:

- A particular range

While Loops



If you don't know ahead of time how many times to iterate, or over what elements.