



# CoGrammar

## Lecture 3: Iteration



**SKILLS  
FOR LIFE**

**SKILLS BOOTCAMPS**



Department  
for Education

## Lecture Housekeeping

---

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.  
**(FBV: Mutual Respect.)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Open Classes.  
You can submit these questions here: [Open Class Questions](#)

## Lecture Housekeeping cont.

---

- For all **non-academic questions**, please submit a query:  
[www.hyperiondev.com/support](https://www.hyperiondev.com/support)
- Report a **safeguarding** incident:  
[www.hyperiondev.com/safeguardreporting](https://www.hyperiondev.com/safeguardreporting)
- We would love your **feedback** on lectures: [Feedback on Lectures](#)

# Lecture Objectives

- **Getting acquainted with Iteration in Python.**
- **Understanding what programming “loops” are and how we can use them to save us time coding.**

# What are Loops?

- ★ **Loops are used when we need to repeat a certain block of code multiple times.**
- ★ **There are two types of loops that will be introduced:**
  - **while loops**
  - **for loops**

# While Loops

- ★ While loops are used in situations when we are not sure how many times we need to repeat the code block.
- ★ Therefore, we can use a while loop to execute a certain condition. While our condition is True, the code within the loop will terminate the moment our condition becomes False.

# While Loop Example

```
kittens = 0
question = input("Has a kitten attempted world domination? (y/n) : ")
while question == "y":
    kittens = kittens + 1
    print(str(kittens) + " attempted world domination")
    question = input("Add another kitten? (y/n) : ")
```

# Infinite Loops

- ★ There may be some cases where we would need the loop to keep looping for as long as the program is running.
- ★ This would be referred to as an infinite loop.
- ★ Example:

```
while True:  
    print("I am an infinite loop")  
    print("And no one can stop me!")
```



# Breaking the Loop

- ★ At some point, we would need to break out of our infinite loop. In order to achieve that, we can use the `break` statement to exit the loop.
- ★ Example:

```
while True:
    question = input("Do you wish to stop me? (y/n) : ")
    if question == "y":
        print("As you wish")
        break
```

# Continuing the Loop


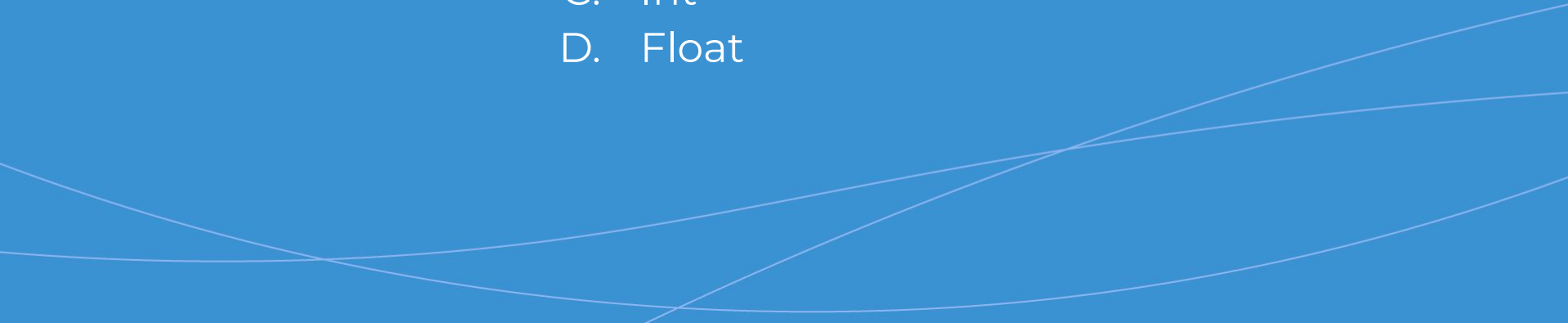
- ★ The `continue` statement is used to skip any and all lines of code within a loop for the current iteration only.
- ★ The loop will not terminate, but will continue with the next iteration.
- ★ The loop will not break.

# Continue Example

```
while True:
    print("I am a loop!")
    question = input("Would you like me to continue? (y/n) : ")
    if question == "y":
        print("Back to the beginning!")
        continue
    else:
        print("I shall cease")
        break
```




# What data type do “while loops” need in order to run?

- 
- A. String
  - B. Bool
  - C. Int
  - D. Float
- 

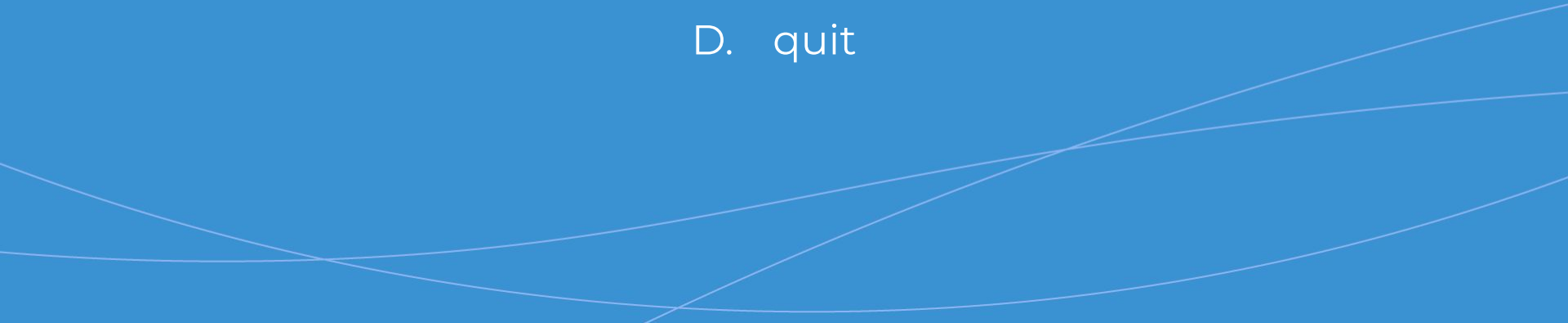


# What will happen if we never set the bool in our “while loop” to false?

- A. The loop will end after 10 loops
  - B. We will get a syntax error
  - C. The loop will only run 1 time
  - D. The loop will continue infinitely
- 



# Which keyword is used to end / exit any loop?

- A. break
  - B. continue
  - C. exit
  - D. quit
- 

# Let's Breathe

**Let's take a small break before moving on to the next topic.**



# For Loops

- ★ For loops are used when we need code to run a specified number of times.
- ★ Think of it making the task of creating ten print statements much easier.

# No need to do this

```
print('')  
print('')  
print('')  
print('')  
print('')  
print('')  
print('')
```



# For Loop Syntax

- ★ **Iterable\_object** : a list of numbers, a string of characters, a range etc.
- ★ **Item** : temporary variable used inside the for loop to reference the current position of our iterator.

```
for item in iterable_object:  
    # Logic goes here
```

# For Loop Example

- ★ The below loop will iterate over the string “coffee”.
- ★ This entails the temporary variable letter being continuously updated with each character found in “coffee”.

```
string = "coffee"  
for letter in string:  
    print(letter)
```

# For Loop Example

As letter will iterate over every instance of our string, we get the output of “coffee” spelled out on separate lines.

```
string = "coffee"  
for letter in string:  
    print(letter)
```

```
c  
o  
f  
f  
e  
e
```

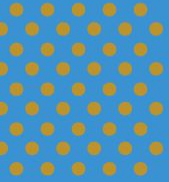

# For Loops & Range

- ★ With for loops, we can also get a range of integers from a starting value to an ending value
- ★ Note that the output here will be values from 1 to 9

```
for num in range(1,10):  
    # Take note that the ending value 10  
    # is exclusive.  
    # Similar to string slicing.  
    print(num)
```

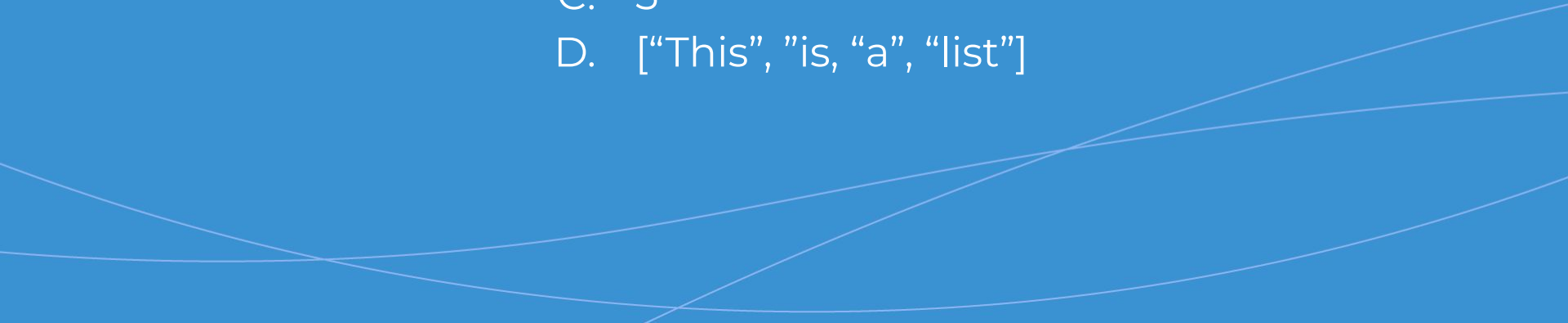


# What are “for loops” used for?

- 
- A. Only counting to numbers
  - B. The same as while loops
  - C. To repeat code block a preset number of times
  - D. To create infinite loops
- 

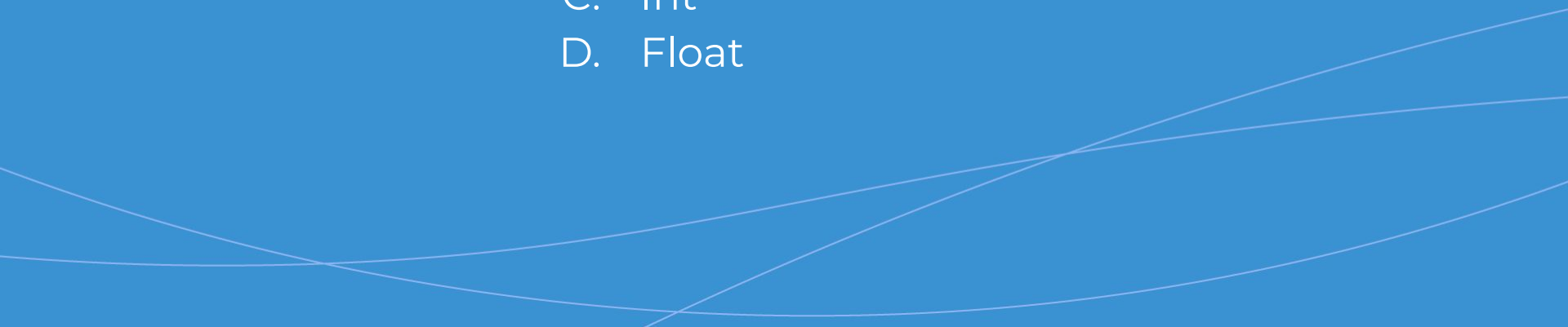


# Which value below will not work with a “for loop”?

- A. “Hello”
  - B. range(5)
  - C. 5
  - D. [“This”, ”is, “a”, “list”]
- 



# Which data type is not well suited for “for loops”?

- A. Bool
  - B. String
  - C. Int
  - D. Float
- 

# CoGrammar

## Q & A SECTION

**Please use this time to ask  
any questions relating to the  
topic, should you have any.**





# CoGrammar

**Thank you for joining!**