Loops and Iteration

A loop allows you to execute the same statements multiple times. Python has two kinds of loop structures: `for` loops, which iterate over the items of a sequence, and `while` loops, which continue to execute as long as a condition is true.

Content Learning Objectives

After completing this activity, students should be able to:

- Explain the syntax and the purpose of a `for` statement.
- Predict how `range()` works given 1, 2, or 3 arguments.
- Identify the three main components of a `while` loop.

Process Skill Goals

During the activity, students should make progress toward:

- Tracing the execution of while/for loops and predict their final output. (Critical Thinking)

Facilitation Notes

Encourage teams to complete Model 1 quickly. Monitor the first page to ensure that each team answers the questions correctly. Report out by having teams swap presenters and compare answers on the second page.

On Model 2, there might be some confusion about the use of the `list()` function. In order to view the actual numbers in the range, we need to convert it to a list. Technically speaking, `range()` is not a function in Python 3, but an immutable sequence type. Be prepared to explain this detail to students if they ask. Make sure teams don't use `list()` in their answers for #14.

If you're running short on time, skip Question #16 or assign as homework. It assumes students have already learned how to convert integers to characters.

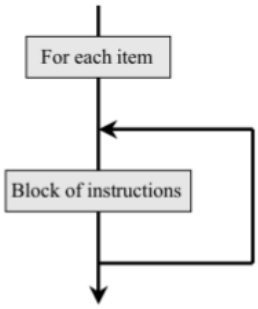
For question #25 on Model 3, it may be easier for students to write the loop itself than to explain its components. If they get stuck, have them write the code on a separate sheet of paper and then use the questions to analyze their solution.

As a wrap-up question, ask the class whether it's possible for a `for` loop to run forever. Discuss the difference between repeating a definite number of times (`for`) vs indefinitely (`while`).

Model 1 `for` Statements

A `for` loop executes the same block of code "for each item in a sequence". Create a new file named `loops.py`, and enter the following code:

```
print("hello")
for x in [2, 7, 1]:
    print("the number is", x)
print("goodbye")
```



Questions (15 min) Start time: _____

1. Run the `loops.py` program. How many times does the indented line of code execute under the `for` loop?

3 times

2. How many times does the line of code NOT indented execute after the `for` loop?

1 time

3. Identify the value of `x` each time the indented line of code is executed.

   a) 1st time: x = 2

   b) 2nd time: x = 7

   c) 3rd time: x = 1

4. Modify the list `[2, 7, 1]` in the following ways, and rerun the program each time. Indicate how many times the `for` loop executes.

   a) non-consecutive numbers: `[5, -7, 0]` 3 times

   b) numbers decreasing in value: `[3, 2, 1, 0]` 4 times

   c) all have the same value: `[4, 4]` 2 times

   d) single value in a list: `[8]` 1 time

5. In general, what determines the number of times that the loop repeats?

The length of the list.

6. What determines the value of the variable `x`? Explain your answer in terms of what is assigned (`x = ...`) each time the loop runs.

The value `x` is selected from the list. Each time the loop runs, the next value from the list is assigned to `x`.

7. Modify the program as follows:

   a) Write a statement that assigns `[0, 1, 2, 3, 4]` to the variable `numbers`.

   ```
   numbers = [0, 1, 2, 3, 4]
   ```

   b) Rewrite the `for x ...` statement to use the variable `numbers` instead.

   ```
   for x in numbers:
   ```

   c) Does the assignment need to come before or after the `for` statement?

   Before

8. Add the following code at the end of your program:

```
for c in "Hi!":
    print(c)
```

   a) What is the output of this `for` statement?

```
H
i
!
```

   b) What determined how many times `print(c)` was called?

   The length of the string

   c) Explain what a `for` statement does with strings.

   It iterates over each character

9. What other data types (besides lists and strings) can a `for` loop handle? Experiment by adding examples to your `loops.py` program. Summarize here what works and what doesn't.

Tuples work similar to lists. Dictionaries give you the keys.
Numbers don't work; you can loop over integers and floats.

Model 2 The `range` Function

The Python `range` function will generate a list of numbers. The `range` function can take up to three numbers as arguments. Fill in the table below by typing the code into a Python Shell:

Python code	Shell output
`range(5)`	range(0, 5)
`list(range(5))`	[0, 1, 2, 3, 4]
`x = range(3)`	
`print(x)`	range(0, 3)
`print(list(x))`	[0, 1, 2]
`list(range(5, 10))`	[5, 6, 7, 8, 9]
`list(range(-3, 4))`	[-3, -2, -1, 0, 1, 2, 3]
`list(range(4, 10, 2))`	[4, 6, 8]
`for i in range(5):` `  print(i)`	prints 0, 1, 2, 3, 4 (on separate lines)

Questions (15 min) Start time: _____

10. Explain the difference in output between the first two lines of code (with and without the `list` function).

The first line of output describes the range as a function. The second line shows the actual range of values as a list.

11. If the argument of the `range` function specifies a single number (x):

a) What will be the first number listed? 0
b) What will be the last number listed? x - 1
c) How many numbers will be in the list? x
d) Use the range function to generate the sequence 0, 1, 2, 3. range(4)

12. If the argument of the `range` function specifies two numbers (x, y):

a) What will be the first number listed? x
b) What will be the last number listed? y - 1
c) How many numbers will be in the list? y - x
d) Use the range function to generate the sequence 1, 2, 3, 4. range(1, 5)

13. If the argument of the `range` function specifies three numbers (x, y, z):

a) What will be the first number listed? x

b) What does the third argument represent? how much to add each time

c) How many numbers will be in the list? $\lceil (y - x)/z \rceil$

d) Use the range function to generate the sequence 1, 3, 5, 7. range(1, 8, 2) or range(1, 9, 2)

14. In your Editor, make a copy of the Model 1 code. Then modify the `for` statement so that the number of times the loop executes is determined by a variable named `times`.

a) How did you change the `for` statement?

```
for i in range(times):  # no need for list() conversion
```

b) How would you cause the loop to print the values 0 to 5?

Add this line before the loop: `times = 6`

15. Consider the two different types of `for` statements used in Model 1 and Model 2.

a) If you wanted to execute a loop 100 times, which type of `for` statement would you choose and why?

for i in range(number), so that you don't have to specify the list.

b) If you wanted to use each item of an existing list inside the loop, which type of `for` statement would you choose and why?

for i in list, since the list exists already and might not be a range.

16. Does the `range` function work with strings? If so, show an example. If not, show how to print the letters A to Z in a loop.

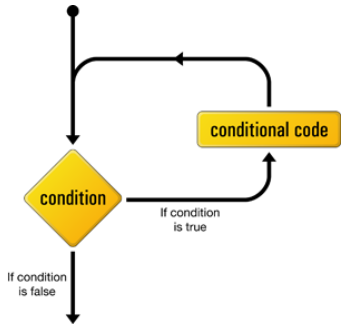
The arguments to range must be integers. You can use the built-in function `chr` to convert integers to their corresponding Unicode characters:

```
for i in range(65, 91):
    print(chr(i))
```

Model 3 `while` Statements

A more general looping structure is the `while` statement. Add the code below to your current `loops.py` program:

```
i = 0
while i < 3:
    print("the number is", i)
    i = i + 1
print("goodbye")
```



Questions (15 min) Start time: _____

17. What must the value of the Boolean expression (after the `while`) be in order for the first print statement to execute? True

18. Circle the statement that changes the variable `i` in the above code. i = i + 1

19. What happens to the value of the variable `i` during the execution of the loop?

It increments by one each time the loop body is executed.

20. Explain why the loop body does not execute again after it outputs "the number is 2".

The variable `i` then becomes 3, which causes the condition `i < 3` to be false.

21. Reverse the order of the statements in the loop body:

```
while i < 3:
    i = i + 1
    print("the number is", i)
```

a) How does the order impact the output displayed by the `print` function?

It prints the numbers 1,2,3 instead of 0,1,2.

b) Does the order impact the total number of lines that are output?

No it does not—either way, there are 3 lines.

22. Identify three different ways to modify the code so that the loop only executes twice.

You can change the first line to `i = 1`. Or you can change the condition to `i < 2`. Or you can change the last line to `i = i + 2`.

23. Describe the three parts of a `while` loop that control the number of times the loop executes.

The first part initializes the variable or condition. The second part tests whether the end has been reached. The third part updates the variable or condition.

24. Comment out the statement `i = i + 1`, and run the module. Then press Ctrl-C (hold down the Ctrl key and press C). Describe the behavior you see, and explain why it happened.

It prints "the number is 0" forever, until you press Ctrl-C. Then it displays "KeyboardInterrupt" as an error message. This all happened because the value of `i` never changed.

When writing a `while` loop, it's helpful to answer a few questions before you start:

- *What needs to be initialized before the loop?*
- *What condition must be true for the loop to repeat?*
- *What will change so that the loop eventually ends?*

25. Consider the function `add(n)` that prompts the user for n numbers and returns the sum of these values. For example, when `add(5)` is called, the user is asked to input five numbers. If the user inputs 3, 1, 5, 2, and 4, the function would return the value 15.

a) Describe the variable that needs to be initialized before the loop begins. i = 0

b) Describe the Boolean expression that must be true for the loop to continue. `i < n`

c) Describe what will need to change so that the loop will eventually end. i = i + 1

d) Now list what else needs to happen inside the body of the loop for you to calculate the sum of the user input.

1) Prompt the user to input a number, and 2) add that number of a running total.

e) Given your previous answer, are there any other values that need to be initialized before the start of the loop?

Yes, the running total should be initialized to zero.