GENERAL ASSEMBLY

Python Control Flow

Patrick D. Smith

Lead Instructor, General Assembly DSI

LEARNING OBJECTIVES

- Explain and utilize if, else, and elif statements
- Explain and utilize for loops
- Construct complex conditional loops

Opening (7-ish Minutes)

Control Flow in Python

As efficient data scientists (and programmers), we often want our models and programs to mimic real-world situations.

Control Flow in Python

As efficient data scientists (and programmers), we often want our models and programs to mimic real-world situations.

Think about any common situation - what do I want for breakfast? What should I do this weekend? In everyday life, we're constantly making decisions about what we'd like to do.

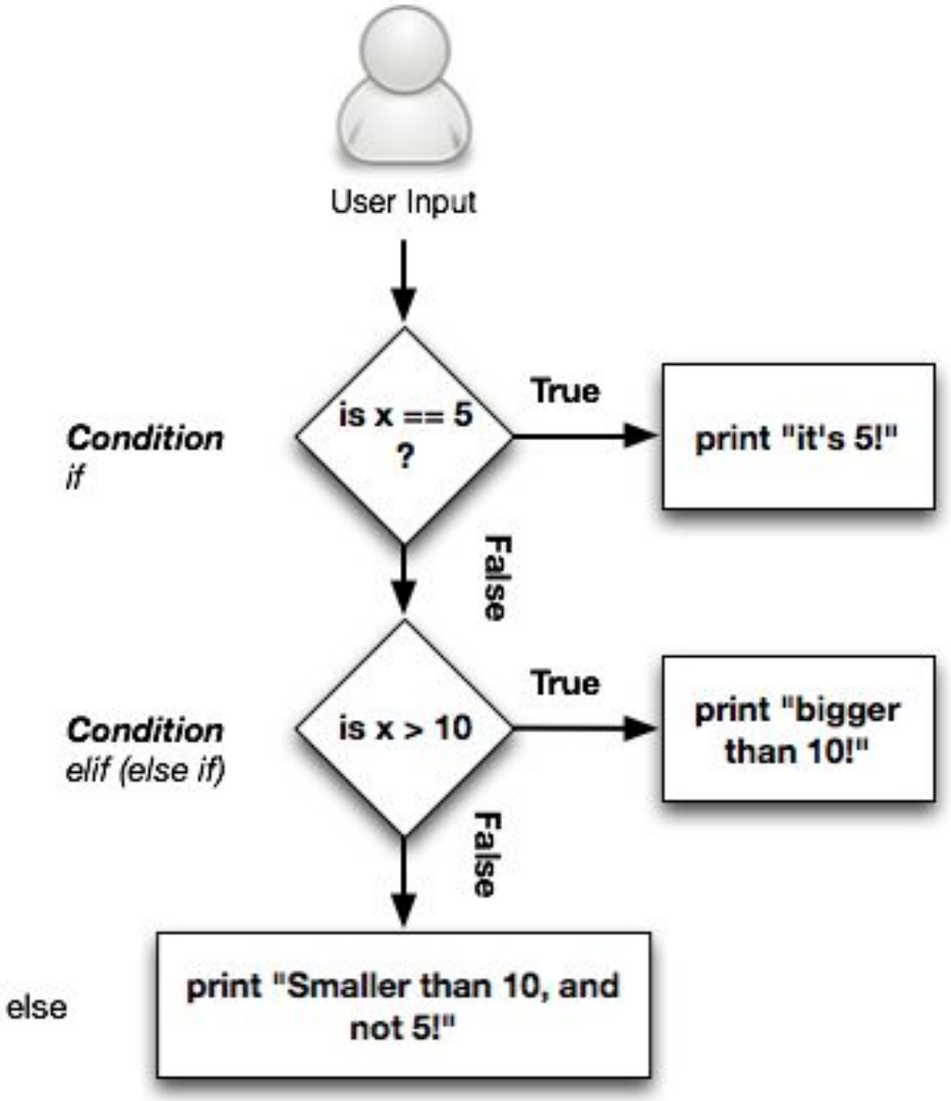
Control Flow in Python

As efficient data scientists (and programmers), we often want our models and programs to mimic real-world situations.

Think about any common situation - what do I want for breakfast? What should I do this weekend? In everyday life, we're constantly making decisions about what we'd like to do.

To simulate this in a pythonic manner, we use **control flow**, which helps our programs make judgements just like we do.

Control Flow in Python



Intro: *If* statements and more (15 Minutes)

Control Flow in Python

- The if statement is used to check a condition: *if* the condition is true, we then run a block of following statements (we call this the ***if block -*** creative right?)
- If the condition isn't true, we use *else* to process another block of statements
- Keep in mind - the *else* block is optional!

Let’s take a look at an example of some if statements:

Control Flow in Python

Pro Tips:

- You can combine the *if* and *else* statements with **elif**
- *If* Inception: You can have an if statement inside of an if statement - we call these ***nested if statements***

Let's take a look at an example of elif and a nested if statement:

Loops (20 Minutes)

Control Flow in Python

Now, we're going to look at two of the most fundamental building blocks of python programming (or just programming in general)

The for loop

And

The while loop

The For Statement

- The ***for*** statement is used to iterate over the elements of a sequence.
- It's used when you have a piece of code which you want to repeat n number of times.
- You can use any object (such as strings, arrays, lists, tuples, dict and so on) in a for loop in Python.

Let’s take a look at an example:

The While Statement

- The while loop tells the computer to do something as long as a condition is met.
- A while loop consists of a block of code and a condition. The condition is evaluated, and if the condition is true, the code within the block is executed.
- This repeats until the condition becomes false.

Let’s take a look at an example:

The Break Statement

- The ***break*** statement is used to stop the execution of a loop even if the loop has not finished
- It's important to note that if you "break" out of a for or while loop, the else statement will not be executed.

Let’s take a look at an example:

The Continue Statement

- The ***continue*** statement is used to tell Python to skip the rest of the statements in the current loop and continue to the next iteration of the loop
- For instance; if a number is out of the bounds of what you're looking for - you can use a continue statement to skip over that number.

Let’s take a look at an example:

Guided Practice: Python Control Flow (20 Min)

Independent Practice: Python Control Flow (20 Min)

Conclusion