Python Decision Logic

Alexander Kalinowski

April 16, 2025

Boolean Logic

Boolean Logic

1. The use of TRUE or FALSE statements in programming
2. Typically represented as a binary decision: TRUE (1) and FALSE (0)
3. A critical piece of digital systems, mathematical set theory, and statistics

Boolean Operators

- AND: if one statement is TRUE and a second statement is also TRUE
- OR: if one statement is TRUE or a second statement is TRUE
- NOT: if one statement is not TRUE

Comparing Statements

- AND: **if** one statement is TRUE and a second statement is also TRUE
- OR: **if** one statement is TRUE or a second statement is TRUE
- NOT: **if** one statement is not TRUE

Conditional Decisions in Python

- The statements in the prior slides are formally known as **conditions**
- A condition is something that we want to check the TRUE/FALSE value of
- Example: if a variable is greater than a fixed number: $(x > 4)$
- Example: if one variable is less than another: $(x < y)$
- Example if two variables are equal to one another: $(x == y)$

If-else logic

- Python provides built-in operators for dealing with conditional decisions
- The **if** statement: checks if a condition is true or not
- The **if-else** statement: if a condition is true, perform an action; else, perform another action
- **Nested** statements: the layering of if-else statements to express more complex logic

Simple Examples

- Let's begin with a placeholder variable x
- We wish to check that x is greater than 10
- If the condition is met $(x > 10)$, we will ask the program to print "Test passed, $x > 10$"

Else Logic

- What should we do if the condition we are checking is false?
- We can handle these cases using **else** statements
- We can also check other conditions using the **elif** statement
- We'll see some of these in following code examples

An Important IF Statement

- When using Python as a scripting language, we need functionality to run the script
- The construct here utilizes the IF conditional
- At the end of a script, we call

```
if __name__ == ''__main__'':
```

- This will indicate to the interpreter to execute the logic defined in the script

Code Examples

- simple_check.py
- age_check.py
- credit_check.py
- credit_simple.py

References