PALs CSC-152

Section 02, Professor Kim, M-W-F

Tony Stone

November 12, 2025

Lesson III

1 Boolean Logic

1.1 Boolean?

Boolean refers to a binary variable with the possible values: "True" or "False." Boolean values open the door to the world of Boolean Algebra (You will learn more about this in later courses). We don't need to worry about rigorous boolean algebra right now, but we can discuss the boolean operations available to us. Let $T =$ "True" and $F =$ "False":

- **AND** T when both inputs are T

$$T \text{ AND } F = F$$

$$T \text{ AND } T = F$$

$$F \text{ AND } F = F$$

- **OR** T when at least one input is T

$$T \text{ AND } F = T$$

$$T \text{ AND } T = T$$

$$F \text{ AND } F = F$$

- **NOT** the opposite value of the input

$$\text{NOT } F = T$$

$$\text{NOT } T = F$$

Boolean algebra however is a very fascinating topic. There is so much more than just these operations, with very interesting outcomes and uses. Though we won't worry about it in the meantime, please keep it in your mind and look forward to it!

1.2 The Boolean Data Type

Naturally, in computer programming, the **Boolean data type** is a data type that represents these very values. Much like the int data type holds a $0, 1, ..., 9$, the boolean variable represents the state of "true" or "false."

1.3 Python's bool

Python has the boolean data type build in. The keyword that Python uses is **bool**. bools, and pythons implemented boolean logic allow us to utilize the above boolean operations in our programming. Lets see the same operations shown above, but as Python implementations!

- **AND**

```python
# 'AND' operator is 'and'
print(True and False) # Output: False
print(True and True) # Output: True
print(False and False) # Output: False
```

- **OR**

```python
# 'OR' operator is 'or'
print(True or False) # Output: True
print(True or True) # Output: True
print(False or False) # Output: False
```

- **NOT**

```python
# 'NOT' operator is 'not'
print(not False) # Output: True
print(not True) # Output: False
```

These examples provide a great foundation for operations using bools, but this hardly scratches the surface. Often, we would like to use bools as a way to deal with certain conditions. But more on that later! Lets build up some ideas about other ways bools can be used:

- **Inequalities**

```python
# value comparison using <, >, <=, =>
anum = 5
bnum = 6
print(anum > bnum) # Output: False
print(anum < bnum) # Output: True
```

- **Typecasting**

```python
# typecasting bool
print(bool(1)) # Output: True (Values larger than 1 are true)
astr = ""
bstr = "any string"
alist = []
blist = ["at least one item"]
print(bool(astr)) # Output: False
print(bool(bstr)) # Output: True
```

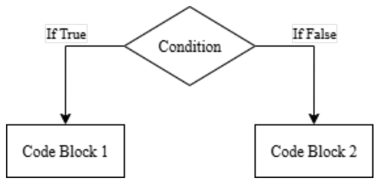

Figure 1: Branching

```python
    print(bool(alist)) # Output: False
    print(bool(blist)) # Output: True
```

- **NOT**

```python
    # In case you need the opposite:
    print(not astr) # Output: True
    print(not bstr) # Output: False
    print(not alist) # Output: True
    print(not blist) # Output: False
    # Note, you don't need to explicitly
    # typecaste with 'bool' when using not
```

2 Branching

2.1 Decision structures

Say we would like to program something that will make a series of predefined decisions based on the input. We need a decision structure! Or a piece of code implemented in our program that will handle different things. We will stay rooted in python implementations. But do not worry, you will face the nuances of conditional statements later on in you CS journeys.

This is known as **branching**. Think of different possibilities of what gets run in the code as branches. See Figure 1. the word 'if' is very important, and is going to be crucial when it comes to implementing decisions in our code.

2.2 if, elif, and else

We use if when we want something to happen **if** something is True. Sometimes we would like this as a standalone check, maybe we want to change the state of a variable *if* something happens. We can do this with the keyword **if** and a piece of boolean logic:

```python
age = int(input("How old are you"))
if age < 13:
    print(":D")
```

In this case, we prompt a user for their age. *If* they are 12 or younger, then we will print them a smiley face! But what about those 13 and up? We can greet them using the **else** keyword. What happens, is that if the **if statement** doesn't execute, the **else statement** will.

```python
age = int(input("How old are you"))
if age < 13:
    print(":D")
else:
    print("hello")
```

Our original smiley face might be a little too kiddish for a 12 year old, we can add the **elif** keyword to extend our decision structure. elif is exactly what it sounds like; 'else-if' it is an else statement that uses an if statement!

```python
age = int(input("How old are you"))
if age < 7:
    print(":D")
elif age < 13:
    print(":)")
else:
    print("hello")
```

3 Key Terms

- **Branching**: The extent of different outputs in a program.
- **Boolean**: A variable with possible values "True" or "False"
- **bool**: Python's boolean data type
- **AND, OR, NOT**: See above definitions
- **Decision Structure**: Code that provides different outputs based on inputs
- ___ **Statement**: By convention, a piece of code containing a keyword is described this way, e.g., "if Statement" or "else Statement"

4 Availability

- In Lecture:
  - Wednesday: 10:00am - 11:00am
- PAL Sessions:
  - Monday: 2:00pm - 3:00pm
  - Wednesday: 9:00am - 10:00am
  - Wednesday: 2:00pm - 3:00pm

References

[1] Tony Gaddis. *Starting Out with Python*. Pearson, 5th edition, 2021.

[2] GeeksforGeeks. Python Tutorial — Learn Python Programming Language - GeeksforGeeks — geeksforgeeks.org. https://www.geeksforgeeks.org/python/python-programming-language-tutorial/. [Accessed 29-10-2025].

[3] Bradley N. Miller and David L. Ranum. Problem solving with algorithms and data structures using python. https://runestone.academy/ns/books/published/pythonds/index.html, 2014. Interactive online edition — accessed: 2024.

[4] Python Software Foundation. *Python: A dynamic, open source programming language*, 2025. Version 3.x documentation.

[5] W3Schools. W3Schools.com — w3schools.com. https://www.w3schools.com/python/default.asp. [Accessed 29-10-2025].