



CS 1112: Introduction To Programming

Booleans and Conditionals

Dr. Nada Basit // `basit[at]Virginia[dot]edu`

Friendly Reminders

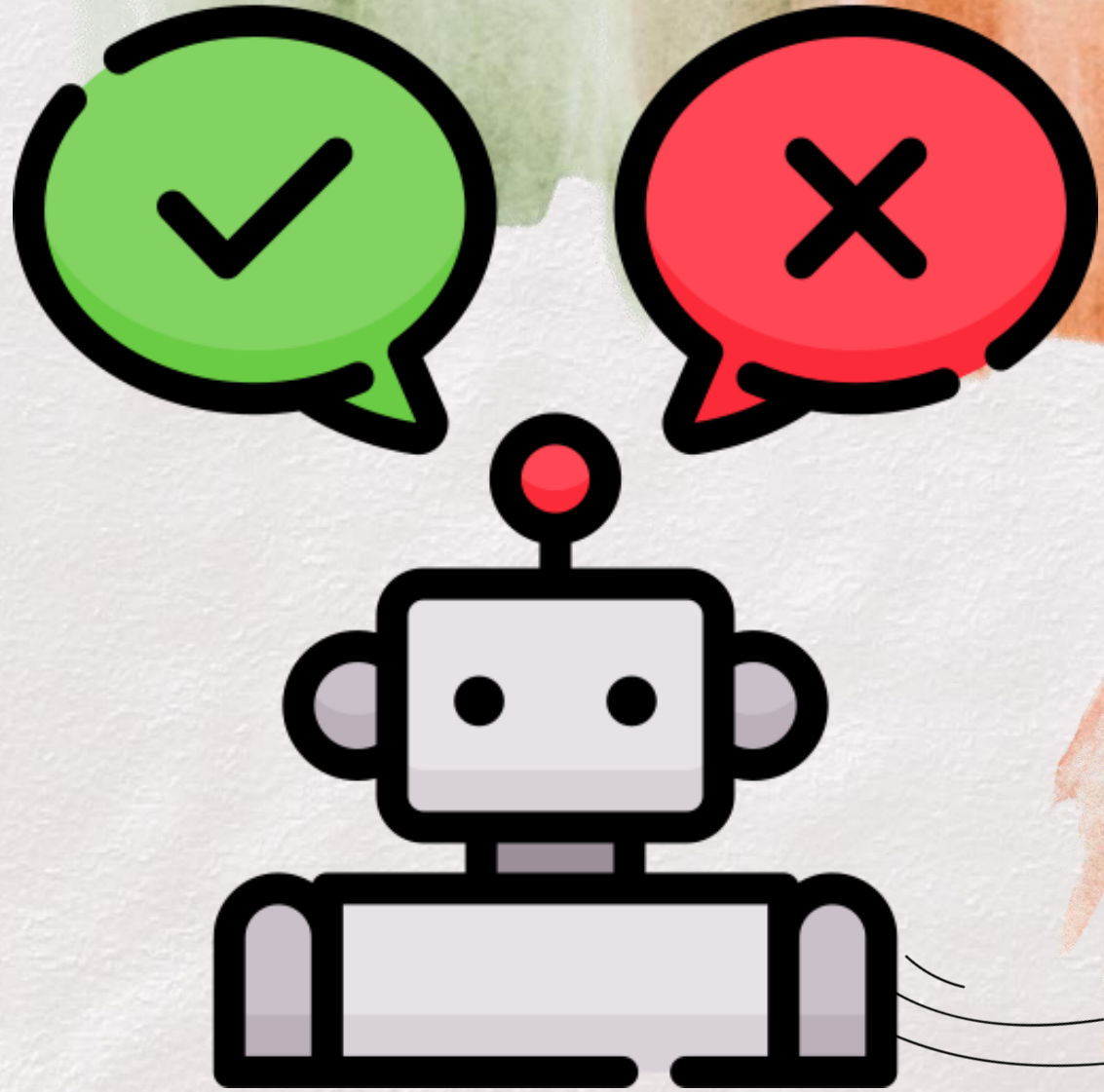
- Your **safety** and **comfort** is important!
 - If you choose to wear a mask you are welcome to do so
 - *We will interpret wearing a mask as being considerate and caring of others in the classroom (not that you are sick), and realize that some may choose to mask to remain distanced*
- Remember to always be **kind, respectful, supportive, compassionate** and **mindful of others!** 😊
- Be an **active** participant in your learning!
You're welcome and **encouraged** to ask questions during class!
- If you feel **unwell**, or think you are, **please stay home**
 - *Contact us! We will work with you!*
 - Get some rest 😊
 - View the recorded lectures – *please allow 24-48 hours to post*



Announcements

- **PA00** graded!
 - You can view your score on Canvas Grades and on Gradescope (*with grader feedback*)
- **Quiz 2** is **due by 11:00pm on Monday (tonight)**!
 - No late quizzes accepted
 - No make-up quizzes allowed
 - If you believe your computer is glitching, it's a good idea to copy down your answers to each of the questions in a word document. In the event something happens, you can send me your solutions (you may also choose to take pictures/ screen-shots for this reason, too)
 - **Take quiz on:** [Sherlock.cs.virginia.edu](https://sherlock.cs.virginia.edu)
- **Programming Assignment 01 (PA01)** is **due by 11:00pm on Wednesday (Feb. 5)**!

Booleans



The Boolean Data Type

- A Boolean type (“**bool**” in Python) is used to denote “**truth**” and only has two possible values:
 - **True**
 - **False**

TYPE	Values	Operators	Inventor
<code>bool</code> <i>Boolean</i>	True, False	not, and, or	George Boole

- The words *True* and *False* are special **keywords** in Python. Do not confuse them for strings!

George Boole: A brief history



- George Boole (Nov. 1815 – Dec. 1864)
- English Mathematician, Philosopher, and Logician
- Most of his career was a professor of mathematics at Queens's College, Cork in Ireland
- Inventor of:
 - Boolean Logic ~ logical theory centered around three simple Boolean Operators:
 - “Or”, “And”, and “Not”
 - Symbolic Logic ~ known as Boolean Algebra
 - Which is a branch in algebra where values are either true or false, usually denoted by 1 and 0
- Boolean Logic is credited with laying the foundations for the Information Age!



GEORGE BOOLE

LL.D., D.C.L., F.R.S.
1815 — 1864

GEORGE BOOLE, FATHER OF MODERN
ALGEBRA, AUTHOR OF "THE LAWS OF THOUGHT"
AND FIRST PROFESSOR OF MATHEMATICS AT
UNIVERSITY COLLEGE, CORK, WAS BORN IN
LINCOLN AND ESTABLISHED AN ACADEMY
IN THIS HOUSE C. 1840,

Plaque on George Boole's House

The Boolean Data Type

- When using the **comparison operators** (`<` `>` `<=` `>=` `==` `!=`), a **bool** value is produced.
 - Note, we use `>=` instead of \geq
 - Given this statement: `if a < b:`
Python will evaluate `a < b` to be either **True** or **False**
- Let's look at the following example:

```
a = 5
b = 10
c = a < b
d = a == b
```

```
print(type(c), c)    # prints: <class 'bool'> True
print(type(d), d)    # prints: <class 'bool'> False
```


Boolean Operators (also called Logical Operators)

- **and**

- Used between 2 Boolean expressions (*binary* operator): `<bool> and <bool>`
- Example: `x < 2 and x > 0`
- **True** when **all** of the things are true

- **or**

- Used between 2 Boolean expressions (*binary* operator): `<bool> or <bool>`
- Example: `x < 2 or x > 0`
- **True** when **any** of the things are true

- **not**

- Precedes 1 Boolean expression (*unary* operator): `not <bool>`
- Example: `not x`
- **True** when **x** is false

Note: you should only use Boolean operators with values of type bool

Boolean Operators

- int and float operations like +, -, *, etc
- Booleans similarly have **3 operations**:
 - **and** **a and b** is **True** if *both* **a** and **b** are **True**, **False** otherwise
 - **or** **a or b** is **True** if *at least one of* **a** or **b** are **True**, **False** otherwise
 - **not** **not a** is *the opposite of* **a**

Truth Tables

A	B	A and B	A	B	A or B	A	not A
True	True	True	True	True	True	True	False
True	False	False	True	False	True	False	True
False	True	False	False	True	True		
False	False	False	False	False	False		

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$
T	T	F	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	F

Practice: A Closer Look at Using Booleans

- `x = 7` *# Anywhere you see 'x', assume the value is 7*
- `a = (x == 7)` *# Evaluate the condition "is x equal to 7", result is True*
- `b = (x != 7)`
- `c = not (x != 7)` *# Evaluate the condition: "what is the opposite of 'is x not equal to 7'", result is not (False), which is True*

Practice: A Closer Look at Using Booleans

- `d = (x < 7)`
- `e = (x <= (3 + 4))`
- `f = (x < 0 or x > 10)`
- `g = (x > 0 or x < 100)`

Practice: A Closer Look at Using Booleans

- `h = (x > 0 or x > 10)`
- `i = (x > 0 and x > 10)`
- `j = (x > 0 and x < 100)`
- `k = (x > 0 and not (x < 100))`

Practice: A Closer Look at Using Booleans

• `l = 8 < x < 10` # x is > 8 and x < 10

• `m = 0 < x < 10` # x is > 0 and x < 10

• `n = (x == 5 or 6)`

• `o = (x == 3 or 7)`

Don't do this, please!

Operands for the logical operators (**and**, **or**, **not**) must be **Booleans**.

This way is better.

• `n = (x == 5 or x == 6)`

• `o = (x == 3 or x == 7)`

Small Boolean Example (1)

```
rain = True
```

```
wind = True
```

```
if rain and not wind:
```

```
    print("Bring an umbrella!")
```

```
elif wind and not rain:
```

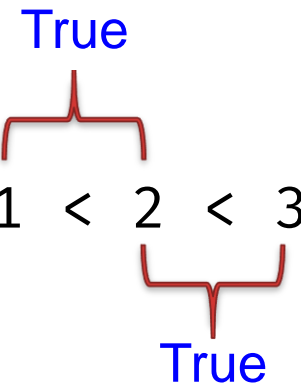
```
    print("Wear a jacket!")
```

Small Boolean Example (2)

```
# trying to turn right
green_light = False
red_light = True
turn_on_red = True
if green_light :
    print("Car may turn")
elif red_light and not turn_on_red :
    print("Car may not turn, no turn on red allowed")
elif red_light and turn_on_red :
    print("Car may turn, turn on red allowed")
```


Comparators Can Be Chained As A Shortcut

- When this happens, the **and** operator is implied

• `1 < 2 < 3` # a shortcut that is equivalent to writing `1 < 2 and 2 < 3`

- `a < b < c < d < e` # this is also legal and acceptable practice

- `1 < 3 > 2 < 400 == 400 < 401` # this is legal in Python, but confusing,
so don't do this! ☺



Quick & Fun Survey Questions

Get to know your peers! ☺

Chocolate or Vanilla or Strawberry?

PYTHON DEMONSTRATION

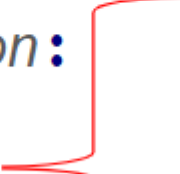
Let's jump on PyCharm!

Boolean examples – `booleans.py`

Conditionals

Conditional Decision Statement

- Boolean operators are used to form **Boolean expressions**
- **Conditional Decision Statement**
 - Defines code that is only **sometimes executed**
 - There are multiple configurations for **if statements**

if <i>boolean expression</i> :		1 of these
statements		
elif <i>boolean expression</i> :		0 or more of these
statements		
elif <i>boolean expression</i> :		0 or more of these
statements		
else:		0 or 1 of these
statements		

Conditional Decision Statement Example

```
age = int(input('Enter your age: '))
```

```
if age < 2:
    print("infant")
elif age < 18:
    print("youth")
elif age < 65:
    print("adult")
else:
    print("senior")
```

1 of these

0 or more of these

0 or 1 of these

Different Versions of Decision Statements

```
if boolean expression:  
    statements
```

```
if boolean expression:  
    statements  
else:  
    statements
```

```
if boolean expression:  
    statements
```

```
elif boolean expression:  
    statements
```

} 1 or more

```
if boolean expression:  
    statements
```

```
elif boolean expression:  
    statements
```

} 1 or more

```
else:  
    statements
```

Remember...

An “**expression**” is a portion of a statement describing a value.

Reminder: Piazza Engagement!

- Quick and friendly reminder to encourage **Piazza engagement**.
- If threshold is met, all students earn some extra credit on the next exam
 - Remember, Piazza engagement is a great way to learn, get answers to questions you may have, and interact with the course material

PYTHON DEMONSTRATION

Let's jump on PyCharm!

Conditional examples - conditionals.py (may finish next class)

Activity on Conditionals

- In **pairs** or groups **up to three** work on the following questions
- **`turtle_conditionals_ica.py`**
- *Use conditionals to determine what your turtle will draw.*

Remember to **check-in** with a TA before leaving class today!

In-Class “lab” Activity!

Notes/Reminders...

Reminder: CS Laptop Loaner Program

- This course requires students to have a **laptop**
- I realize that not everybody might have one (nor necessarily need one for their desired major / path...)
- If you do not have a laptop for any reason... *not to worry!*
- The CS department's Systems staff has a notebook / laptop loaner program and will be able to loan you a notebook / laptop computer for the duration of the semester if you don't have one or if you cannot afford one.
 - Also available if your laptop is broken and under repair, we can arrange for you to receive a loaner laptop for a week or two until your own laptop is fixed

Interested? Link: https://www.cs.virginia.edu/wiki/doku.php?id=cs_laptop_loaner

I am happy to be your sponsor. Please let me know.

Tools: Piazza

- We will use **Piazza** in the following way:
 - Website: <https://piazza.com/> [Linked through **Canvas**]
 - Piazza is a great tool for asking questions about **course content**, **policies**, or getting help on **homework** assignments
 - While you are waiting for an answer, see if there's an answer you can provide to someone else's question. We're all in this together! **CS is a team sport!** 😊
 - TAs will monitor and answer questions throughout the semester
 - Not a means to help you debug your code! (See more below)

It is very important to remember the following:

- **Do not post complete or partial code solutions (for Homework)** on Piazza when seeking answers to your question unless it is in a **PRIVATE** post
- **Do not post complete or partial quiz solutions (code or short-answer)** when seeking answers to your question unless it is in a **PRIVATE** post

Tools: Gradescope

- We will use **Gradescope** in the following way:
 - Website: <https://www.gradescope.com/>
 - Linked through **Canvas**
 - **Homework assignments** will be **submitted**
 - Most programming assignments are autograded
 - Some aspects of programming assignments may be manually graded