[220 / 319] Operators

Meena Syamkumar
Andy Kuemmel
Cole Nelson

Readings:
Chapter 1 of Think Python,
Chapter 2 of Python for Everybody

Additional readings:
Computer terminology

Learning Objectives

- Run Python code using:
  - Command line
  - Jupyter Notebook

Evaluate:

- numeric expressions containing mathematical operators (e.g., “+” and “-“)
- string expressions containing string operators and escape characters

Differentiate:

- behavior of the /, //, and % operators

Recognize examples of different Python data types:

- int, float, str, bool

Evaluate:

- expressions containing comparison operators (e.g., “==” and “>”)
- Boolean expressions containing the operators “and”, “or”, “not”
- mixed expressions using the correct order of operations (precedence)

Today's Outline

Software
- Interpreters
- Notebooks

Demos

Operator Precedence

Demos

Boolean Logic

Demos

What you need to write/run code

An interpreter

- Python 3 (not 2!)
- some extra packages (comes with anaconda installation)
- runs Python code

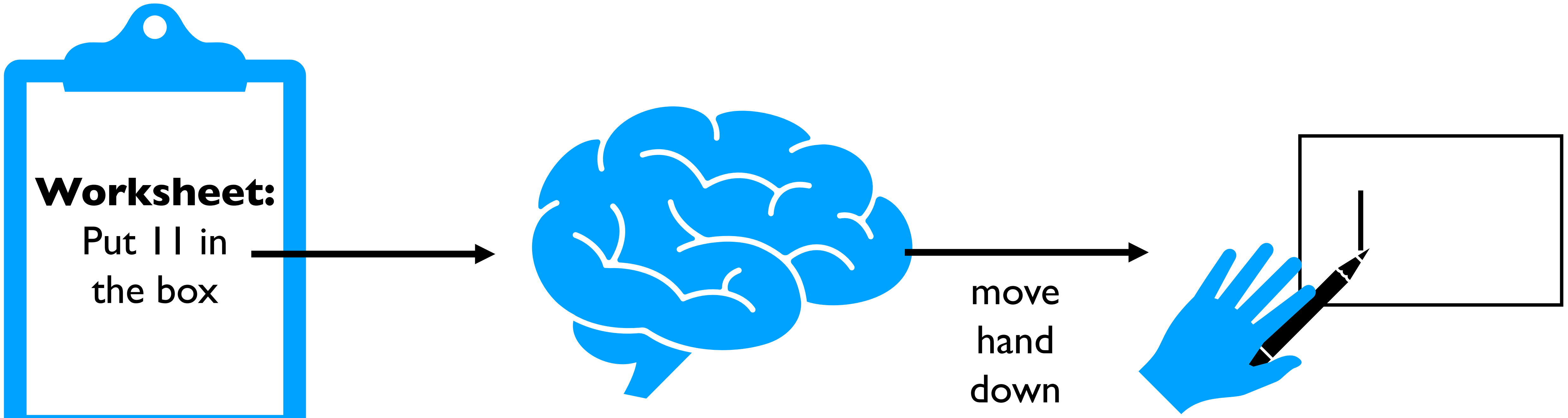
Jupyter Notebooks

- comes with anaconda installation
- acts like both interpreter and editor (type and save Python code)

Interpreter

A program that runs a program

- Translates something the human likes (nice Python code) to something the machine likes (ONEs and ZEROs)

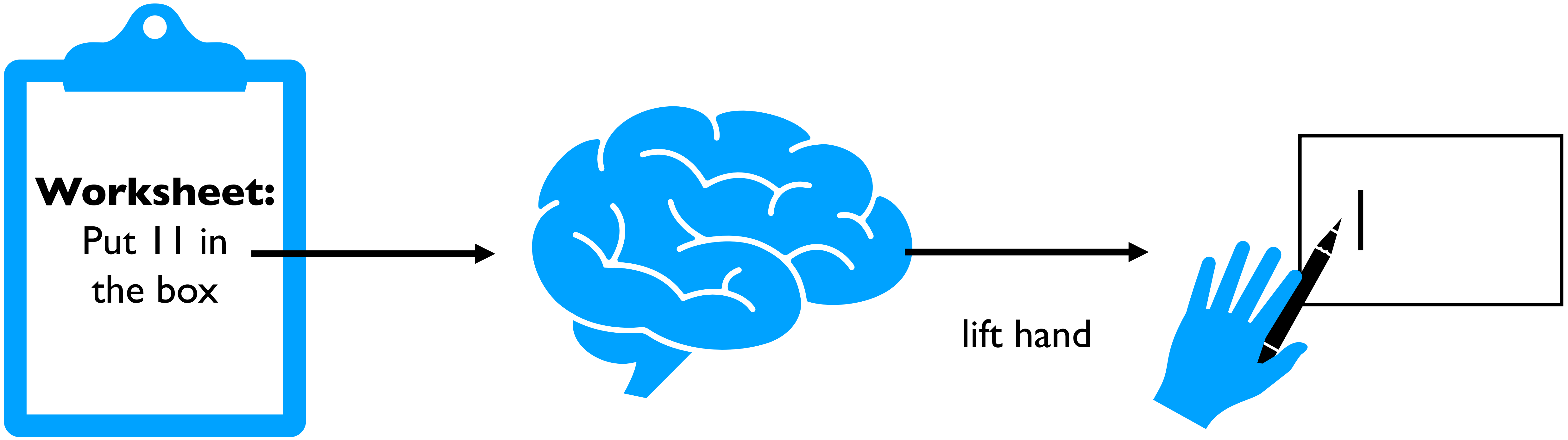


You were an interpreter when you did the pseudocode worksheets!

Interpreter

A program that runs a program

- Translates something the human likes (nice Python code) to something the machine likes (ONEs and ZEROs)

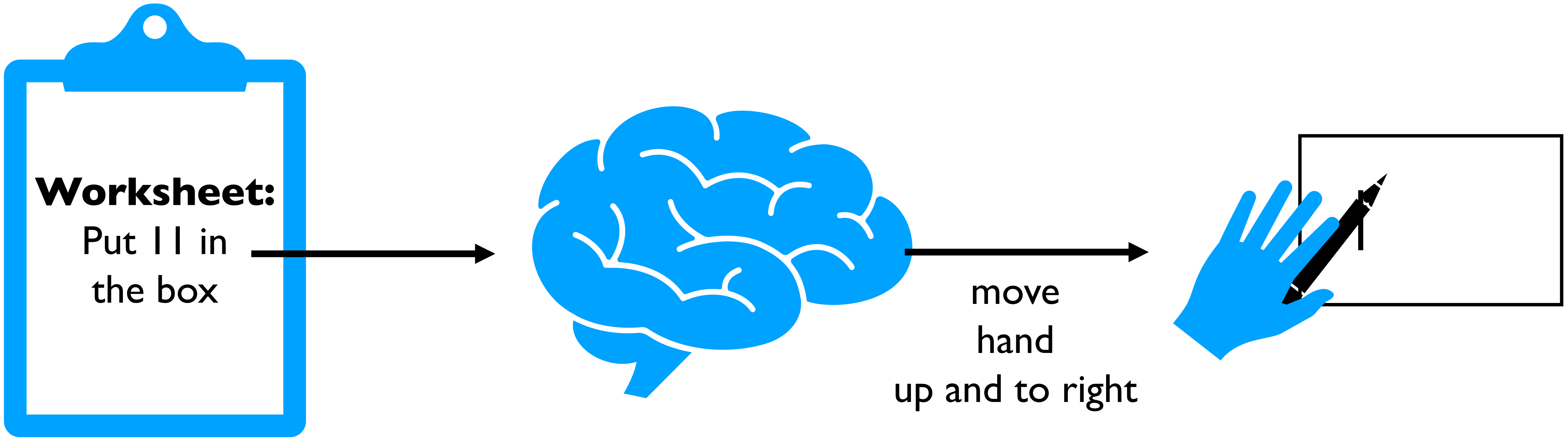


You were an interpreter when you did the pseudocode worksheets!

Interpreter

A program that runs a program

- Translates something the human likes (nice Python code) to something the machine likes (ONEs and ZEROs)

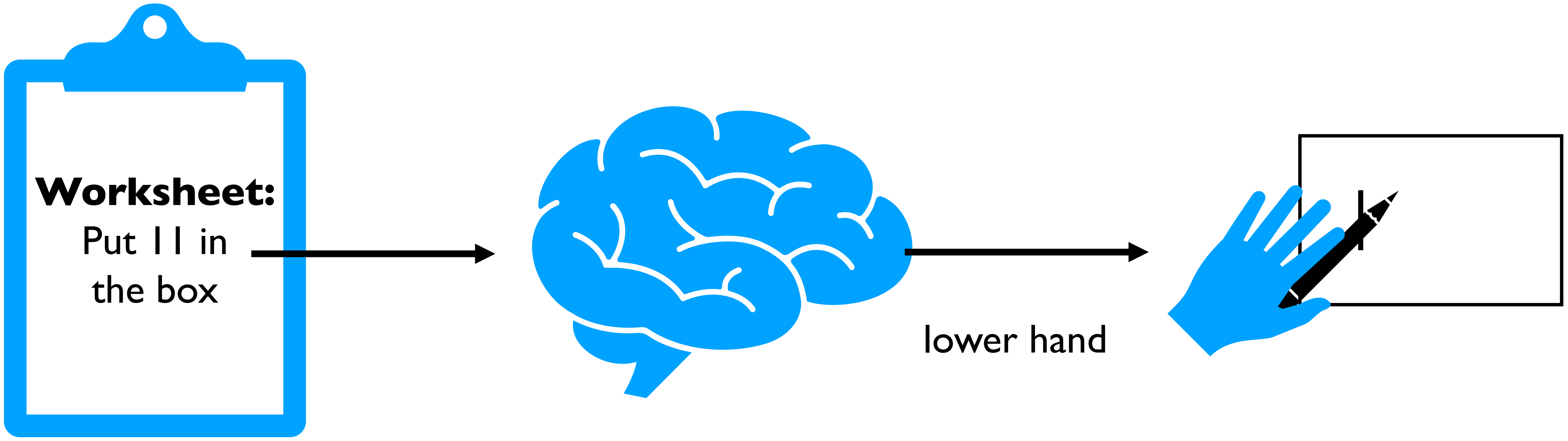


You were an interpreter when you did the pseudocode worksheets!

Interpreter

A program that runs a program

- Translates something the human likes (nice Python code) to something the machine likes (ONEs and ZEROs)

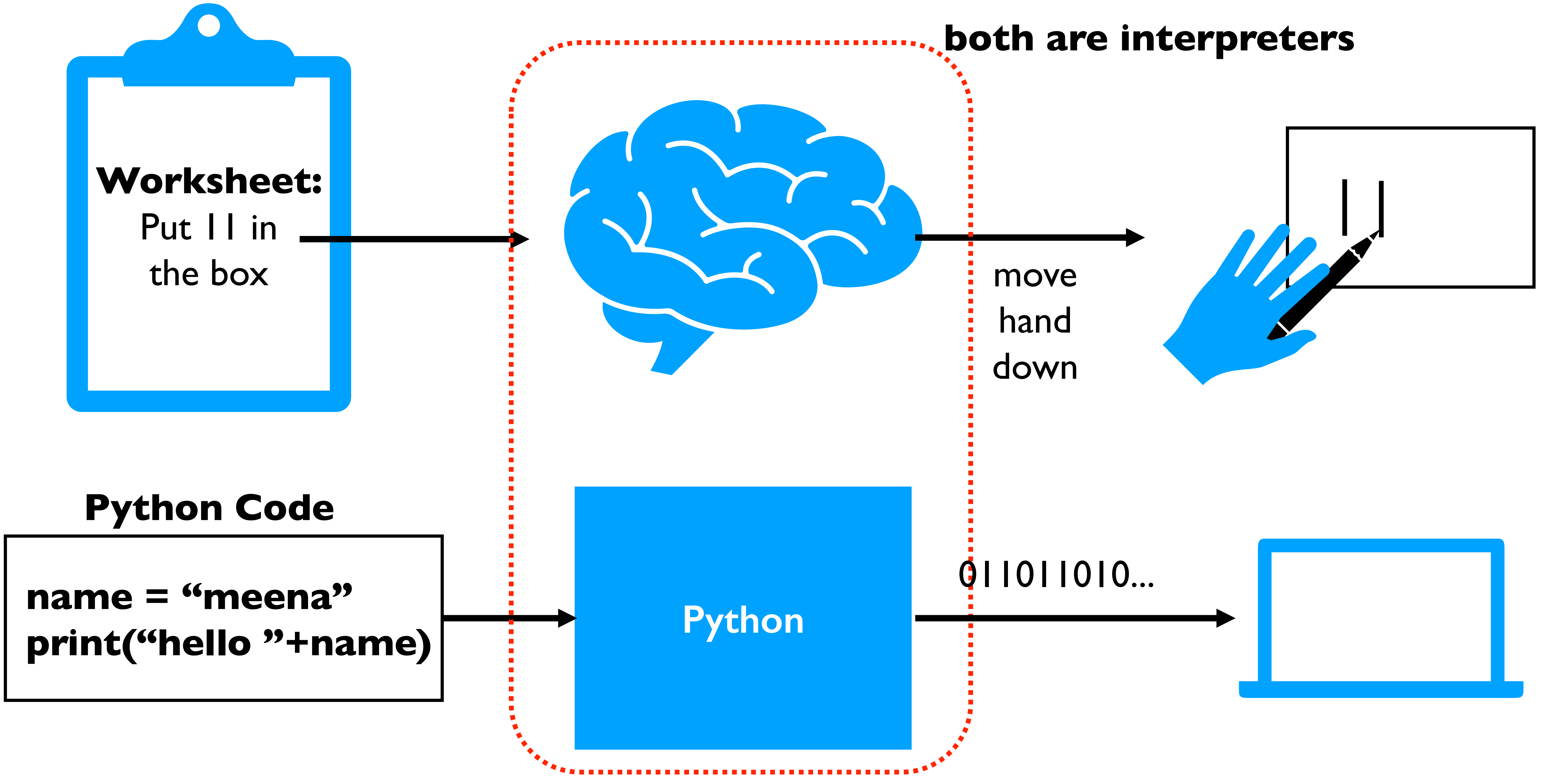


You were an interpreter when you did the pseudocode worksheets!

Interpreter

A program that runs a program

- Translates something the human likes (nice Python code) to something the machine likes (ONEs and ZEROs)

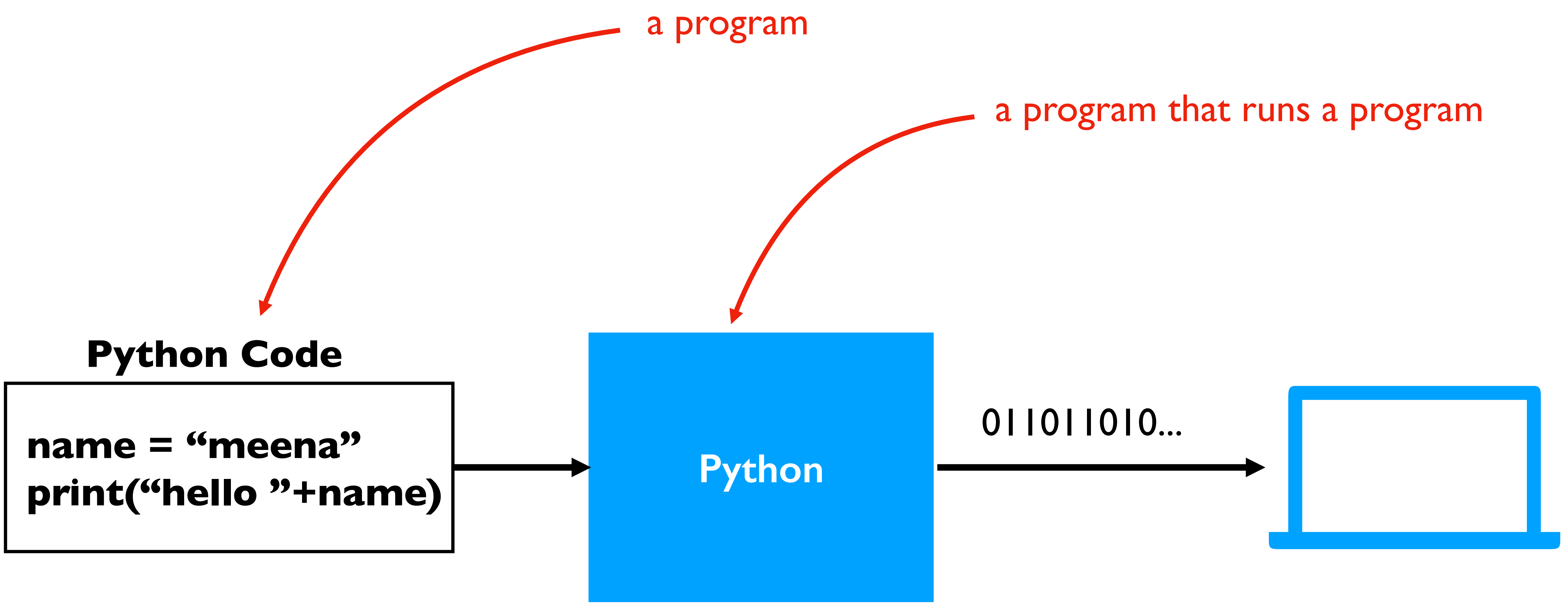


```
name = “meena”
print(“hello ”+name)
```

Interpreter

A program that runs a program

- Translates something the human likes (nice Python code) to something the machine likes (ONEs and ZEROs)

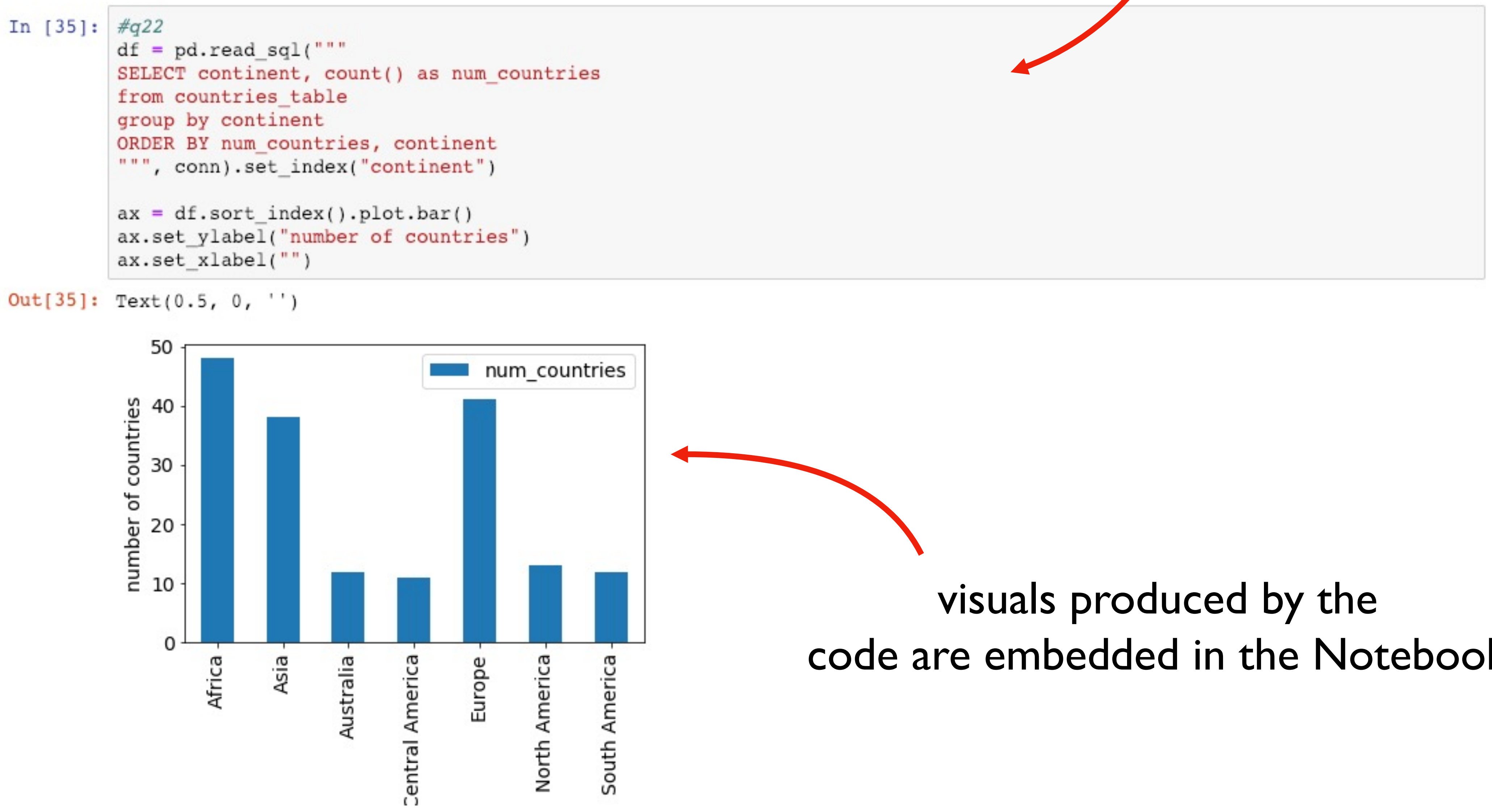


```
name = “meena”
print(“hello ”+name)
```

Jupyter Notebooks

notebooks breakup code into
"cells" containing Python code

...



```
In [35]: #q22
         df = pd.read_sql("""
         SELECT continent, count() as num_countries
         from countries_table
         group by continent
         ORDER BY num_countries, continent
         """, conn).set_index("continent")

         ax = df.sort_index().plot.bar()
         ax.set_ylabel("number of countries")
         ax.set_xlabel("")
Out[35]: Text(0.5, 0, '')
```

visuals produced by the
code are embedded in the Notebook

A Notebook is a file that contains code and other things
(e.g., documentation, images, tables, etc.)

.ipynb (Interactive Python Notebook) files are not easy to open in a regular text editor

3 ways we'll run Python

1. interactive mode

Quick syntax check

```
ty-mac:~$ python
Python 3.9.7 (default, Sep 16 2021, 16:59:28)
[Clang 10.0.0 ] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 1 + 1
2
```

triple arrows mean Python code runs as you type it

2. script mode

Run auto-grader tests

the interpreter program is named "python"; run it

```
ty-mac:~$ python test.py
```

the name of the file containing your code (called a "script") is passed as an argument to the python program

3. notebook "mode"

```
ty-mac:~$ jupyter notebook
```

open Jupyter in a web browser

we'll do most work in notebooks this semester

Today's Outline

Software

- Interpreters
- Notebooks

Demos

Operator Precedence

Demos

Boolean Logic

Demos

Today's Outline

Software
- Interpreters
- Notebooks

Demos

Operator Precedence

Demos

Boolean Logic

Demos

Order of Simplification

Python works by simplifying, applying one operator at a time

```
3 * 3 + 2 * 2 + 16 ** (1/2)
3 * 3 + 2 * 2 + 16 ** (0.5)
3 * 3 + 2 * 2 + 4
9 + 2 * 2 + 4
9 + 4 + 4
13 + 4
17
```

Rules

- First work within parentheses
- Do higher precedence first
- Break ties left to right

Operator Precendence

	What is it?	Python Operator	
Mathematical	exponents	**	**simplify first**
	signs	+x, -x	
	multiply/divide	*, /, //, %	
	add/subtract	+, -	
	comparison	==, !=, <, <=, >, >=	
Logic	boolean stuff	not	
	...	and	**simplify last***
	...	or	

these are the ones you should be learning at this point in the semester (there are a few more not covered now)

* one exception is an optimization known as "short circuiting"

Today's Outline

Software

- Interpreters
- Notebooks

Demos

Operator Precedence

Demos

Boolean Logic

Demos

Today's Outline

Software
- Interpreters
- Notebooks

Demos

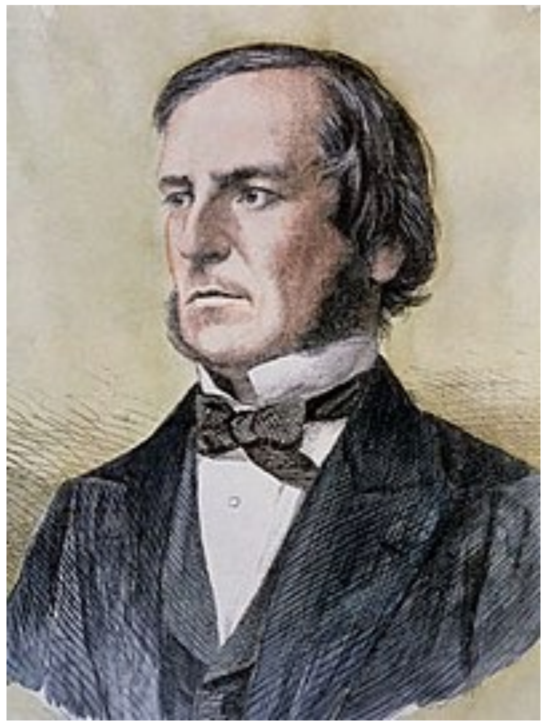
Operator Precedence

Demos

Boolean Logic

Demos

Boolean Logic



The logic of truth:

- Named after George Boole
- Two values: True and False
- Three operators: **and**, **or**, and **not**

AND

	False	**True**
False	False	False
True	False	**True**

OR

	False	**True**
False	False	**True**
True	**True**	**True**

NOT

False	**True**
True	False

FALSE!

It's a Saturday AND we're attending CS 220 lecture

AND

	False	True
False	False	False
True	False	**True**

OR

	False	True
False	False	**True**
True	**True**	**True**

NOT

False	True
True	False

TRUE!

Project 1 is due on Wednesday
OR I'll eat my hat

AND

	False	True
False	False	False
True	False	**True**

OR

	False	True
False	False	**True**
True	**True**	**True**

NOT

False	True
True	False

Control Flow: Remember that conditionals and loops *sometimes* do something. We'll use bool logic a LOT to control when we do/don't.

AND

	False	**True**
False	False	False
True	False	**True**

OR

	False	**True**
False	False	**True**
True	**True**	**True**

NOT

False	**True**
True	False

Today's Outline

Software
- Interpreters
- Notebooks

Demos

Operator Precedence

Demos

Boolean Logic

Demos