IMPERIAL

BIOE50010 – Programming 2

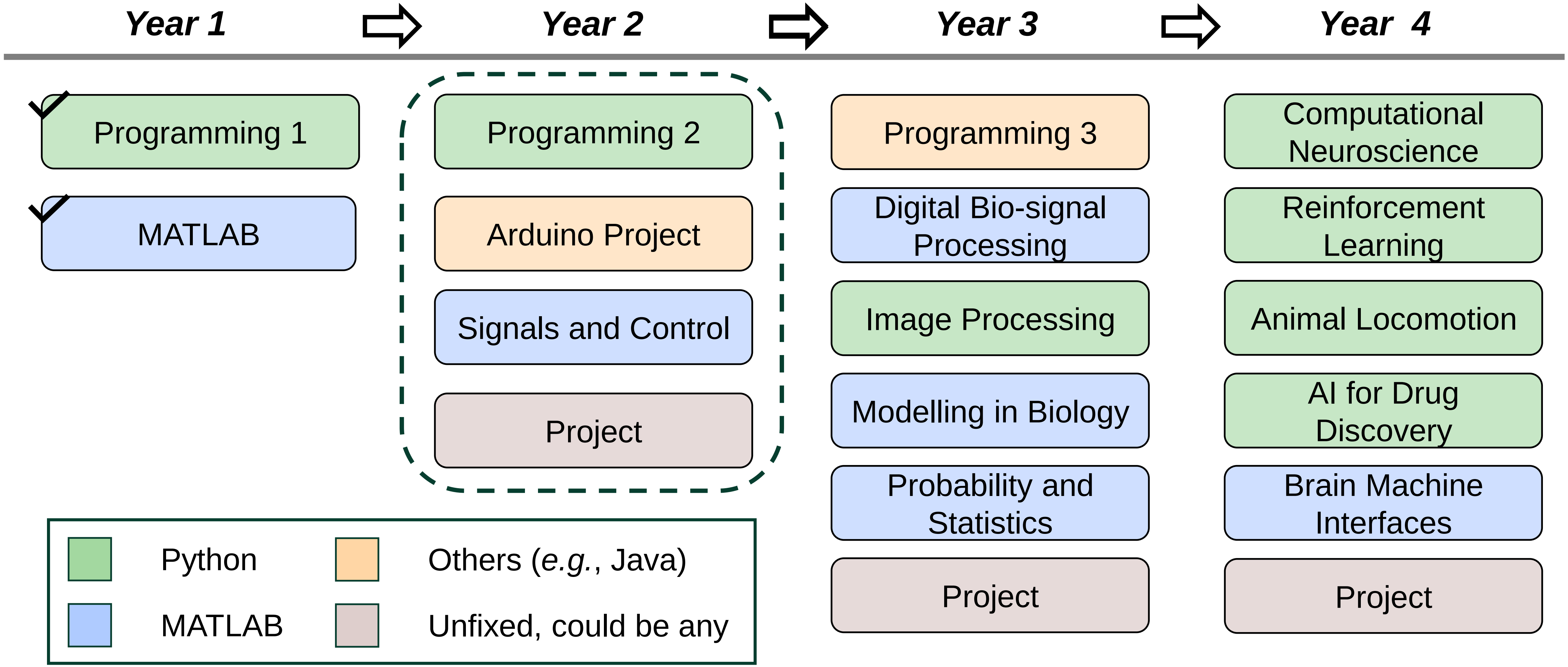
Computer Lab 1

Binghuan Li Department of Chemical Engineering

Maria Portela Department of Bioengineering

10 October, 2024

An indictive timeline



Year 1 ⇨	Year 2 ⇨	Year 3 ⇨	Year 4
✓ 🟩 Programming 1	🟩 Programming 2	🟧 Programming 3	🟩 Computational Neuroscience
✓ 🟦 MATLAB	🟧 Arduino Project	🟦 Digital Bio-signal Processing	🟩 Reinforcement Learning
	🟦 Signals and Control	🟩 Image Processing	🟩 Animal Locomotion
	🟫 Project	🟦 Modelling in Biology	🟩 AI for Drug Discovery
		🟦 Probability and Statistics	🟦 Brain Machine Interfaces
		🟫 Project	🟫 Project

🟩	Python	🟧	Others (*e.g.*, Java)
🟦	MATLAB	🟫	Unfixed, could be any

* Information retrieved from the Module Descriptor 2024-25. Indictive only.

The Programming 2 Course

Assessment Modes

- 1 timed assignment (50%) + 1 programming exam (50%) *
- To be soon communicated by the module leader directly to you.

Where to Seek Help?

- **Module leader:** Dr James Choi
- Questions are encouraged to post on **Ed Discussion**
- General programming advices are welcomed.
- AI tools should be used with great discretion – be sure you understand everything.

* Information retrieved from the Module Descriptor 2024-25. Indictive only.

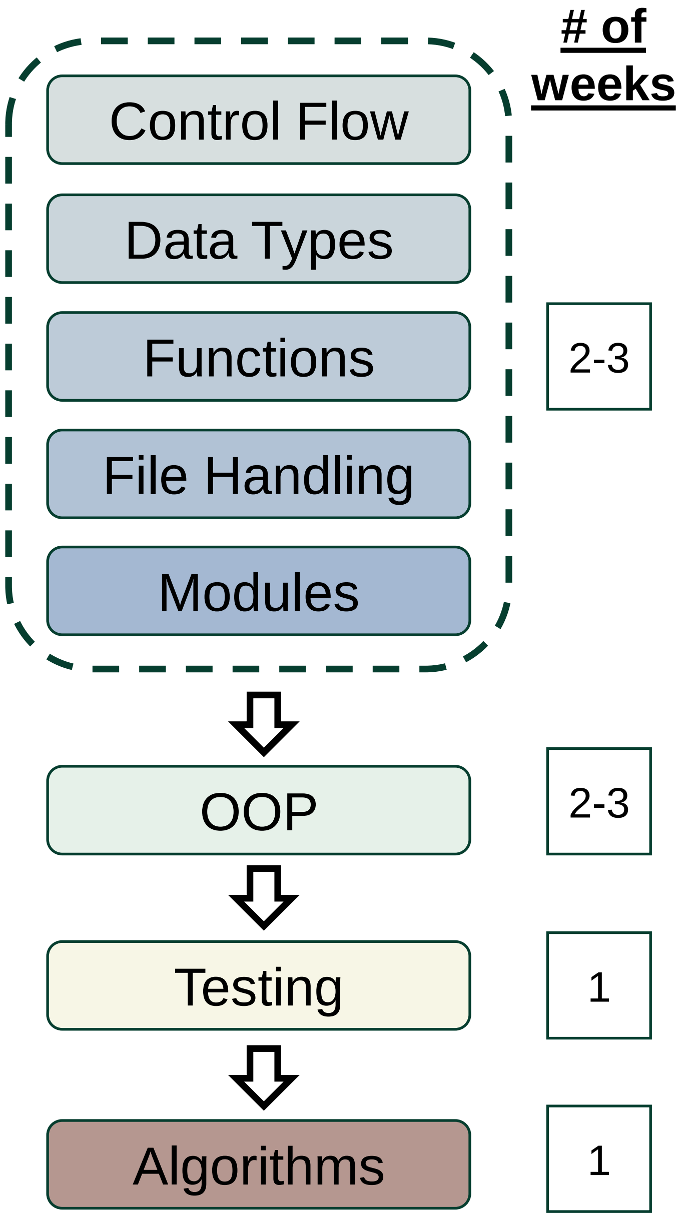
Rough Structure & Rationale

Lectures (2 hours × 9 weeks)

- Introduction to general coding concepts with **definitions** and **live coding examples**.
- **Aim:** to enhance your understanding of core concepts and techniques.

Labs (2 hours × 10 weeks)

- **Exercises** to apply the concepts from lectures delivered in self-learning and peer learning fashion.
- **Aim**: apply coding concepts in a practical setting.



Will It Be Tough?

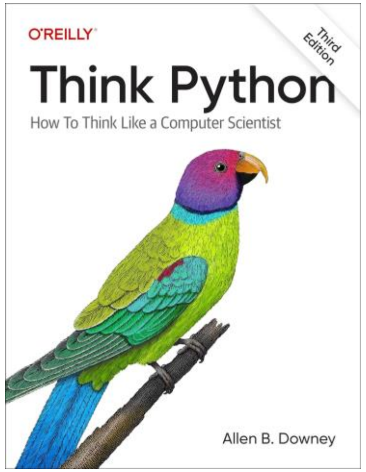
“There are only two kinds of languages: the ones people complain about and the ones nobody uses.”

- Bjarne Stroustrup

Prerequisite & References

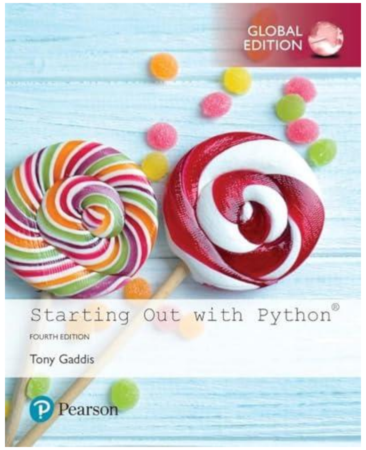
Q - What do we expect you have known from Programming 1?

- **Data types:** `int`, `str`, `list`, `dict`, …
- **Operations:** arithmetic (`+`, `-`), comparison (`==`), logical (`True`, `False`)
- **Control flows:** `if…elif…else` condition, `while` condition, `for` loop
- **Functions and scopes:** definition a function, pass and return data to/from function



Think Python 2e,
by A. Downey

“official” textbook, rigorous, comprehensive, but informative like a dictionary



Starting Out with Python,
by T. Gaddis

“unofficial” textbook, very friendly to Python freshers with intuitive explanations, but can be shallow for advanced coders

Expectations for Labs

- Use this time as it works better for you
  - You don’t need to work on the exercises in advance
  - You may ask questions about previous labs
- Try it yourself before looking at the solutions – you will not **learn** coding if you don’t **do** coding
- Ask questions (to your peers and to us)
  - Explain what you expected your code to output and what happened instead
  - Naming your functions and variables sensibly and organising your code logically will make it easier for us to help you

Tips?

- **Syntax, syntax, syntax**
- Don’t rush – but please keep up! Log your progress
- “Why doesn’t my code work?” will not help you (and others) debug efficiently
- Use **Stack Overflow** / **ChatGPT** / **Co-pilot** wisely – how do you tell if a certain piece of code is good enough or not?
- <u>Working code is the best code.</u>

✓ **Correct syntax**

```
for i in range (0,10):
␣␣␣␣print(i)
```

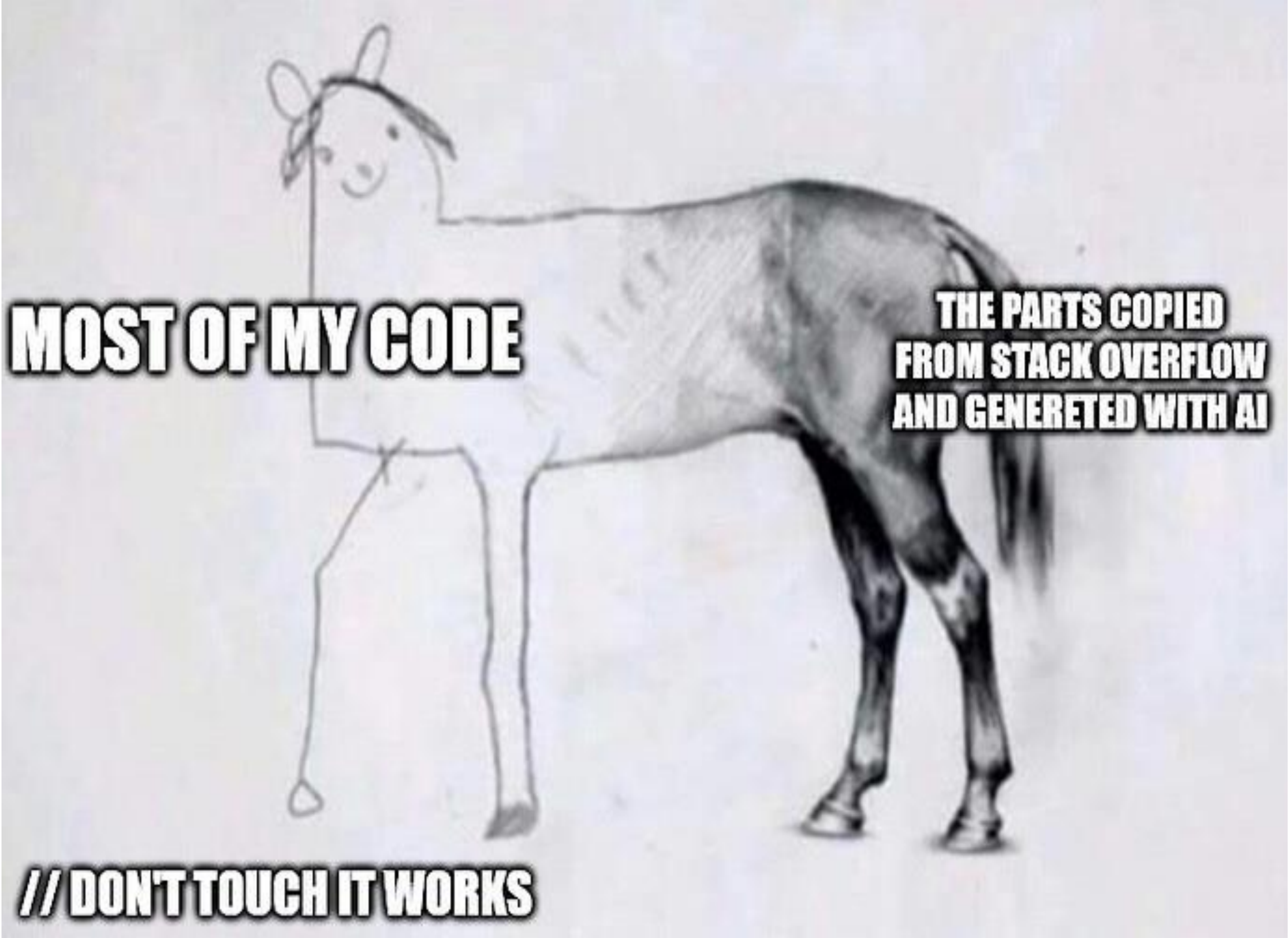
✗ **Erroneous syntax: indentation**

- not indented

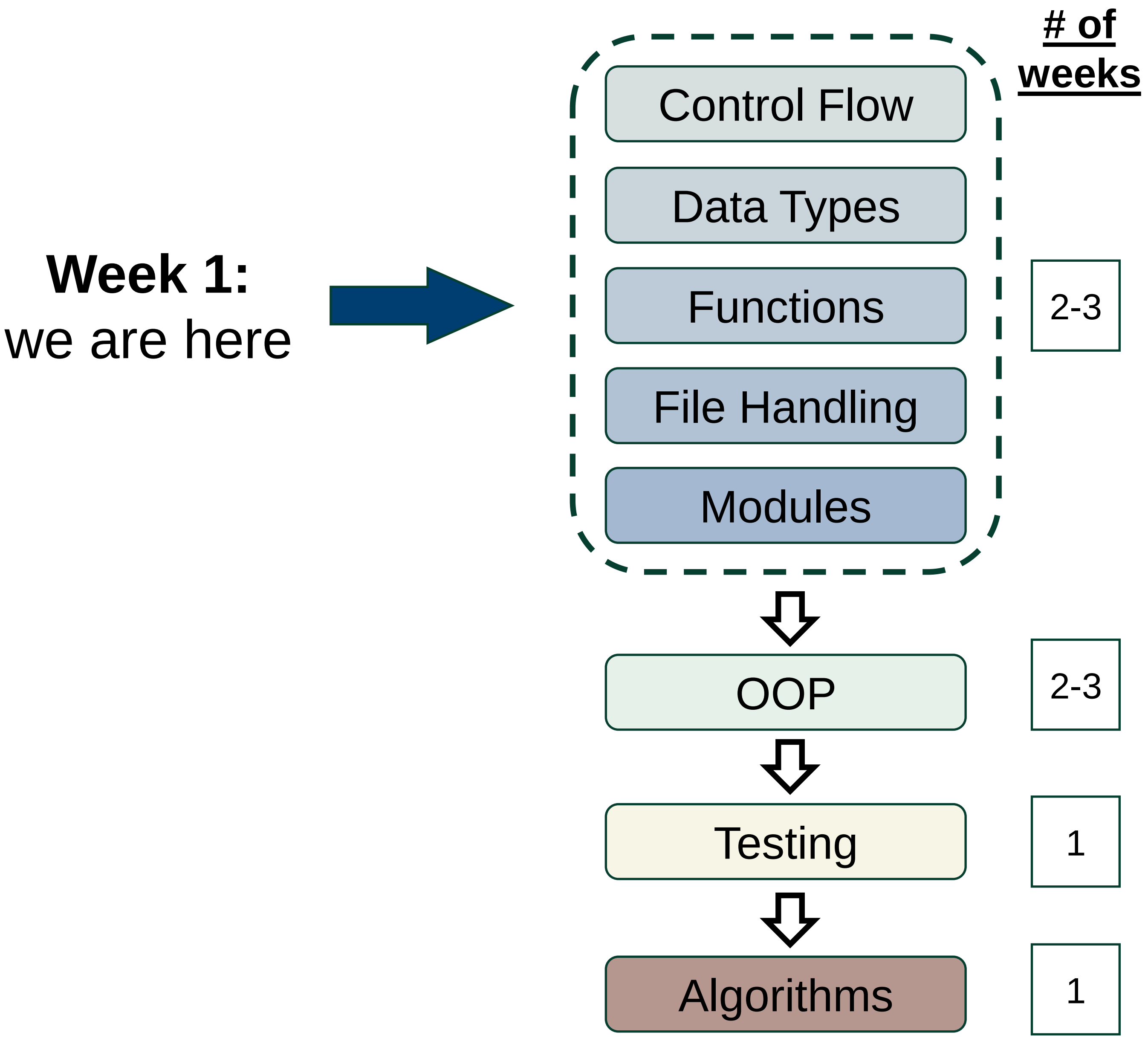
```
for i in range (0,10):
print(i)
```

- Inconsistent indentation

```
for i in range (0,10):
␣␣␣␣print(“i equals to”)
␣␣␣␣␣␣␣␣print(i)
```


MOST OF MY CODE
THE PARTS COPIED FROM STACK OVERFLOW AND GENERETED WITH AI
// DON'T TOUCH IT WORKS

Progress Check



Week 1:
we are here

# of weeks

Control Flow

Data Types

Functions

File Handling

Modules

2-3

OOP

2-3

Testing

1

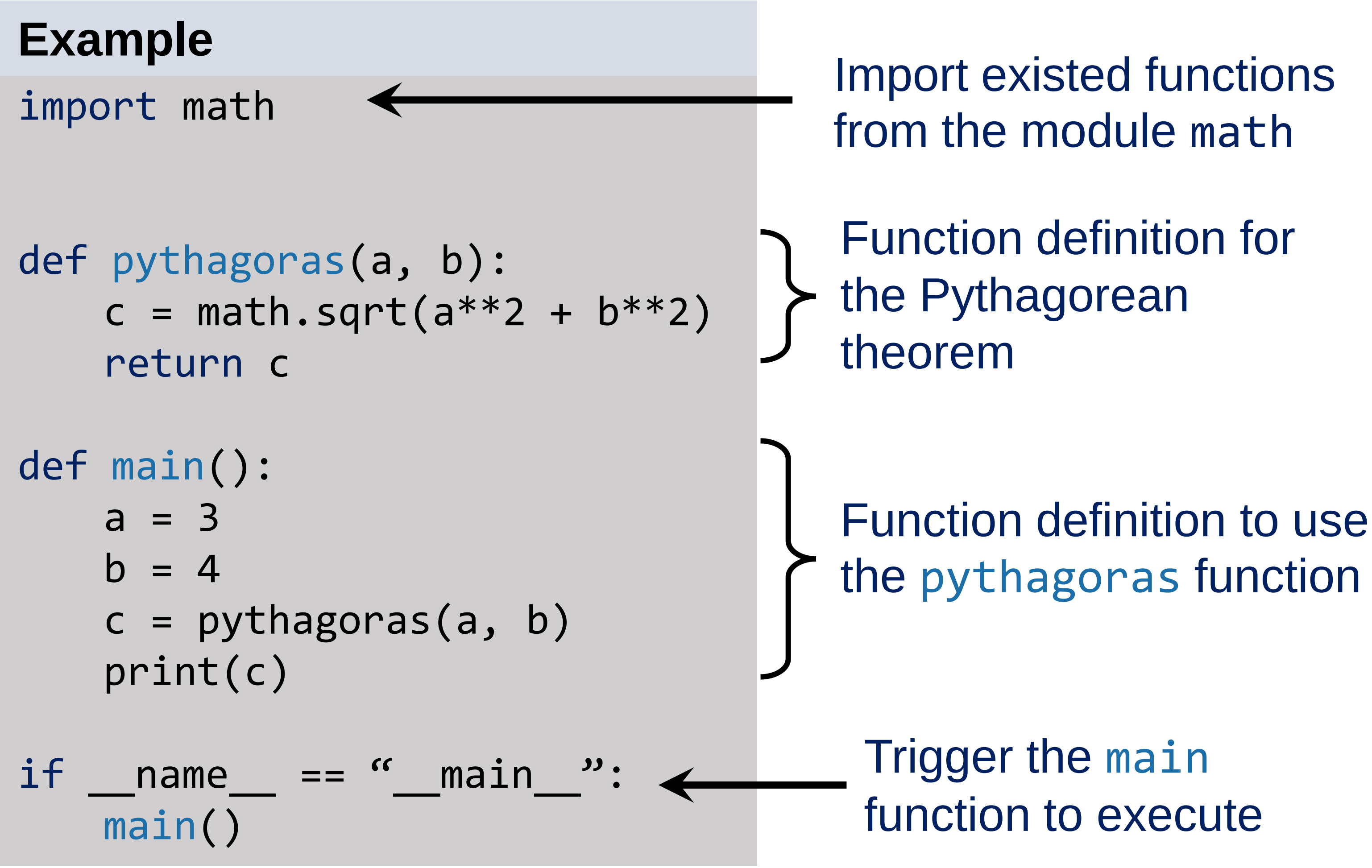
Algorithms

1

Modular Programming

Modules (functions) are put together to make up one executable program.

- Functions are separately defined → reusable
- Functions are triggered serially in a main script



Example

```
import math


def pythagoras(a, b):
    c = math.sqrt(a**2 + b**2)
    return c

def main():
    a = 3
    b = 4
    c = pythagoras(a, b)
    print(c)

if __name__ == “__main__”:
    main()
```

Import existed functions from the module `math`

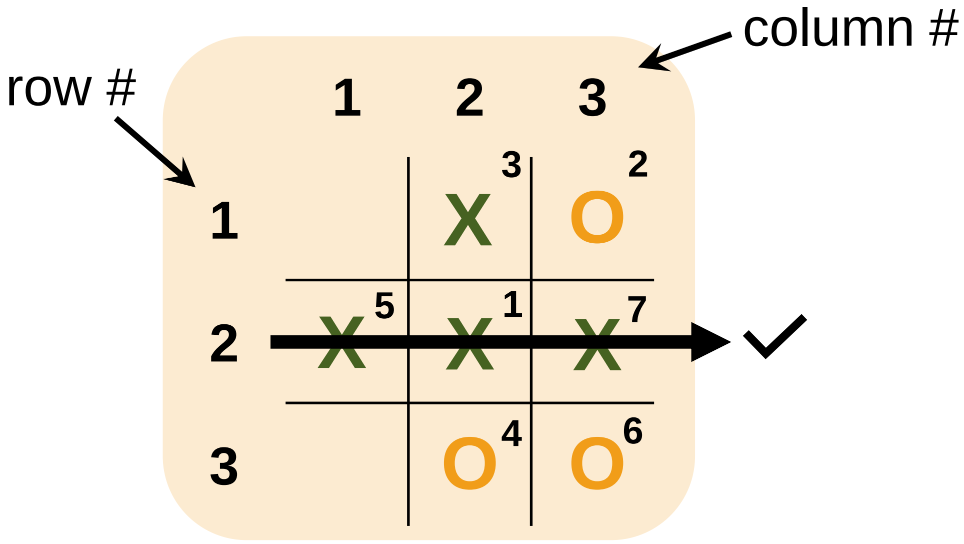
Function definition for the Pythagorean theorem

Function definition to use the `pythagoras` function

Trigger the `main` function to execute

Tic Tac Toe

A 3×3 game board



row #

column #

	1	2	3
1		X^{3}	O^{2}
2	X^{5}	X^{1}	X^{7}
3		O^{4}	O^{6}

✓

2 players:
X and O play in turn

step 1	player **X**	row **2**	col **2**
step 2	player **O**	row **1**	col **3**
step 3	player **X**	row **1**	col **2**
step 4	player **O**	row **3**	col **2**
step 5	player **X**	row **2**	col **1**
step 6	player **O**	row **3**	col **3**
step 7	player **X**	row **2**	col **3**

game over, player **X** win!

Note: A ***tie*** occurs when the board is full and neither player has won.

Your task today

Write a Python programme to realise the game Tic Tac Toe. **Modularise** your programme (using functions), and your code should consider the following aspects:

1. How many steps do you need? Draw yourself a flowchart on a piece of paper, it may include…
   - **Format** a 3×3 board;
   - **Switch**/set a player, **take** the move;
   - **Update** the cells in the 3×3 board;
   - **Check** if the termination condition reached: X/O win the game? Tie?
2. If you code this flow using functions, how many functions you may need? (i.e., how many functions are reusable?)
3. Error/exception handling: check user input – accept or reject?

Questions?

That’s it for now.

You can now proceed to the Lab 1 exercise.

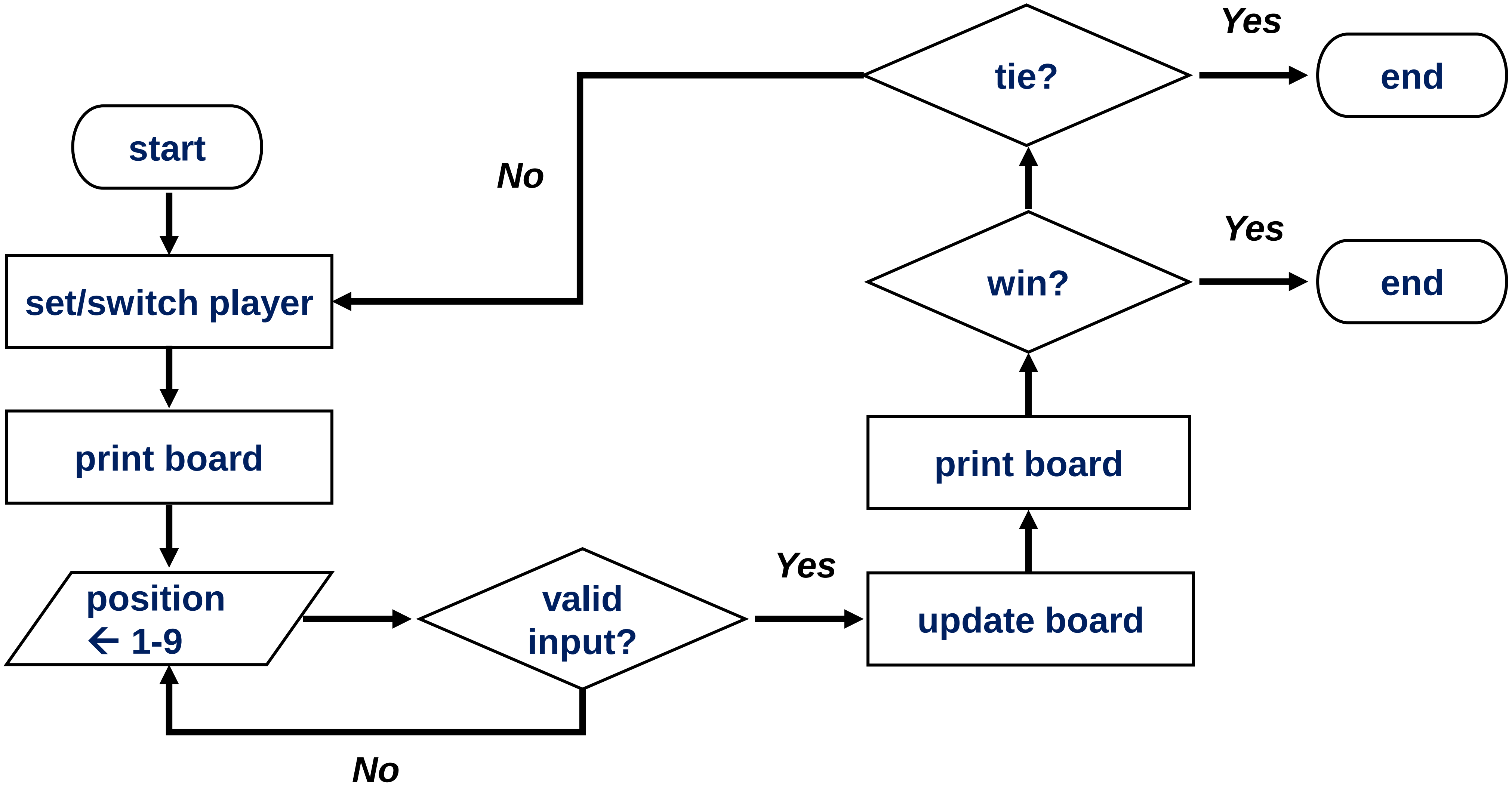
Need More Help?

Quick refresh of Python basics

- **Programming 1, Labs 1 & 2 scripts** by C. Rowlands, via Blackboard.
- Intro to computer science - Python on Khan Academy.
- (Unofficial but nice) cheat sheets here, here, or here.
- Not sure about a certain function? Try to use **`help()`** function for an answer!

Need coding examples?

- **Loops and Conditions**: pp.152 program 3-6 (grader.py), pp.188 program 4-2 (temperature.py), pp.214 program 4-17 (test_score_averages.py)
- **Functions and Modularisation:** pp.288 program 5-28 (geometry.py)


start
set/switch player
print board
position
← 1-9
valid
input?
No
Yes
update board
print board
win?
Yes
end
tie?
Yes
end
No