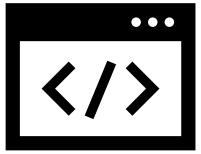
Imperial College London



BIOE50010 – Programming 2

Computer Lab 1

Binghuan Li | *Department of Chemical Engineering*

binghuan.li19@imperial.ac.uk

October 12, 2023

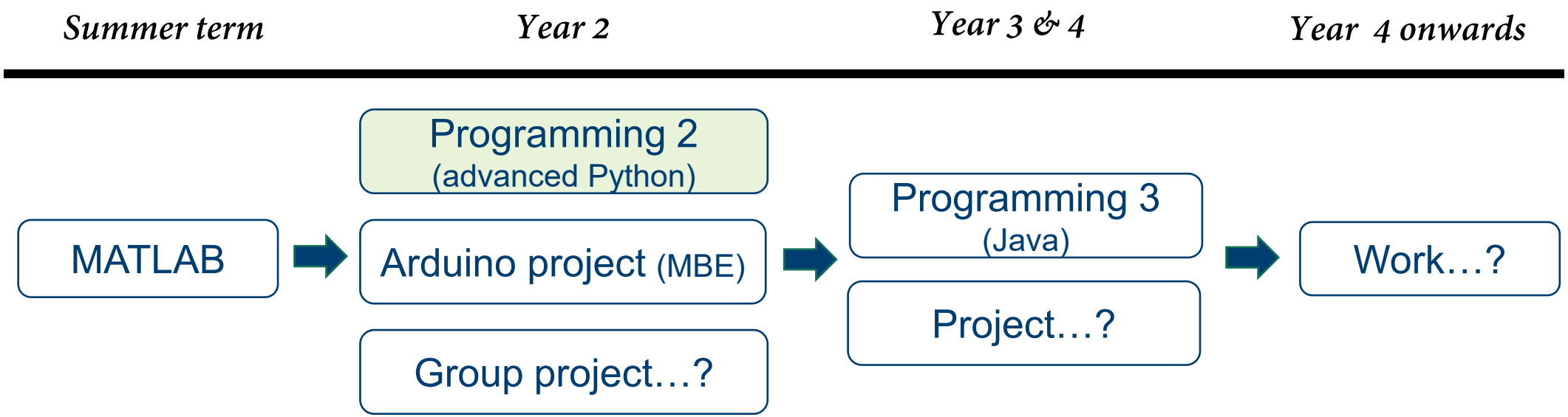
Imperial College London

Top (do not) tips

From where we stopped last term…

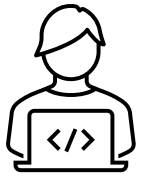
Imperial College London

- An indictive timeline:



- *Thoughts? Expectations? Confidence?*

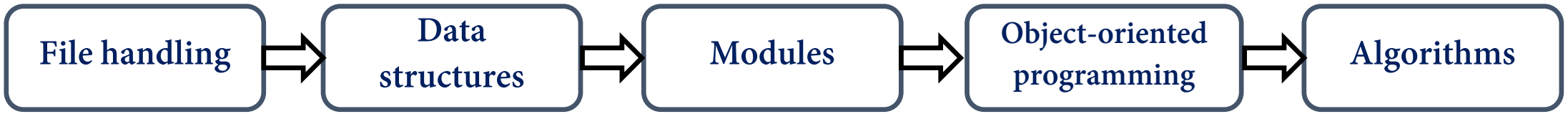
> *"There are only two kinds of languages: the ones people complain about and the ones nobody uses."*
>
> *- Bjarne Stroustrup*



About this session

Imperial College London

- *Weekly computer lab (2h×10) + lecture (2h ×9) + code clinics (1h ×8)*



- *Assessed by 2 timed assignments – will be communicated by the leader*

Q - What do you have?

- Access to your materials via Blackboard
- Lectures and recommended textbooks
- Online materials on Python
- us ☺

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

- Data types: `int`, `str`, `list`, `dict`, …
- Control flows: `if…elif…else`, `while`, `for`
- Functions and scopes

General tips to survive & thrive?

1) Syntax, syntax, syntax – that is everything
2) Don't rush – but please follow up…!
3) "why doesn't my code work?" won't help you solve the issue.
4) Use *Stack Overflow/chatGPT* wisely – how do you tell if a certain piece of code is good or not?
5) Working code is the best code.

"Premature optimization is the root of all evil."

✓ Correct syntax

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

✗ Syntax error: indentation

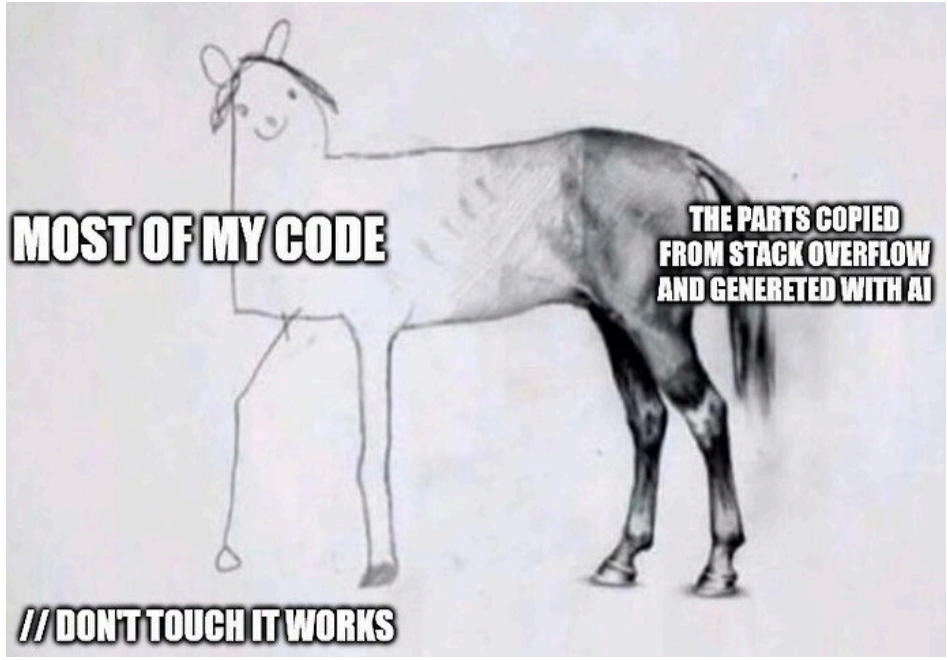
```
for i in range (0,10):
print(i)
```

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

General tips to survive & thrive?

1) Syntax, syntax, syntax – that is everything
2) Don't rush – but please follow up…!
3) "why doesn't my code work?" won't help you solve the issue.
4) Use *Stack Overflow/chatGPT* wisely – how do you tell if a certain piece of code is good or not?
5) Working code is the best code.

"Premature optimization is the root of all evil."



Modular Programming

Imperial College London

Modules (functions) are put together to make up the 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

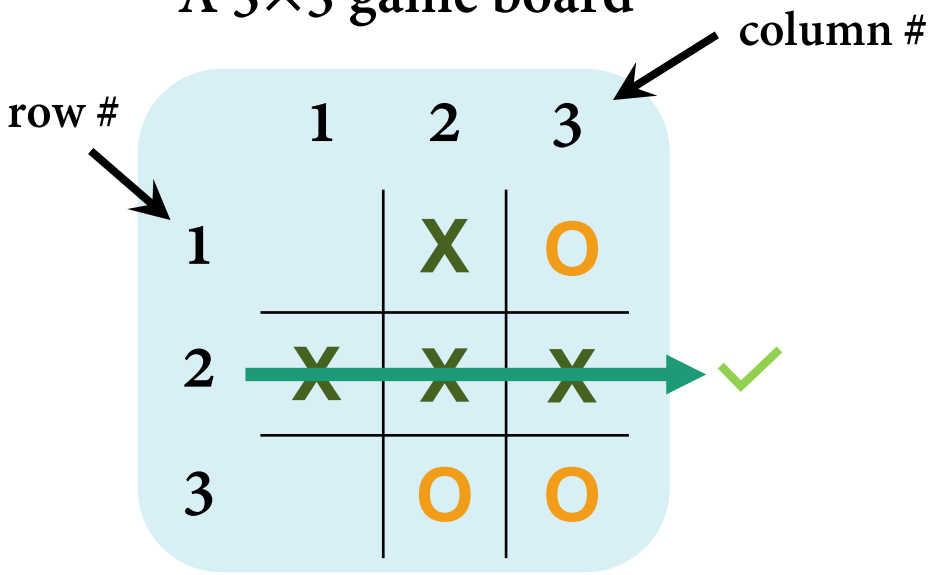
Function definition to use the `pythagoras` function

Trigger the `main` function to execute

Revision Task - Tic Tac Toe

Imperial College London

A 3×3 game board



2 players: X and O
play in turn

One possible walkthrough:

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 !

Imperial College London

Revision Task - **Tic Tac Toe**

- A ***tie*** occurs when the board is full and neither player has won. For example,

	1	2	3
1	O	X	O
2	X	O	X
3	X	O	X

It is a tie!

Imperial College London

Your task today

Code a Python tic tac toe game in the *modular* programming fashion.

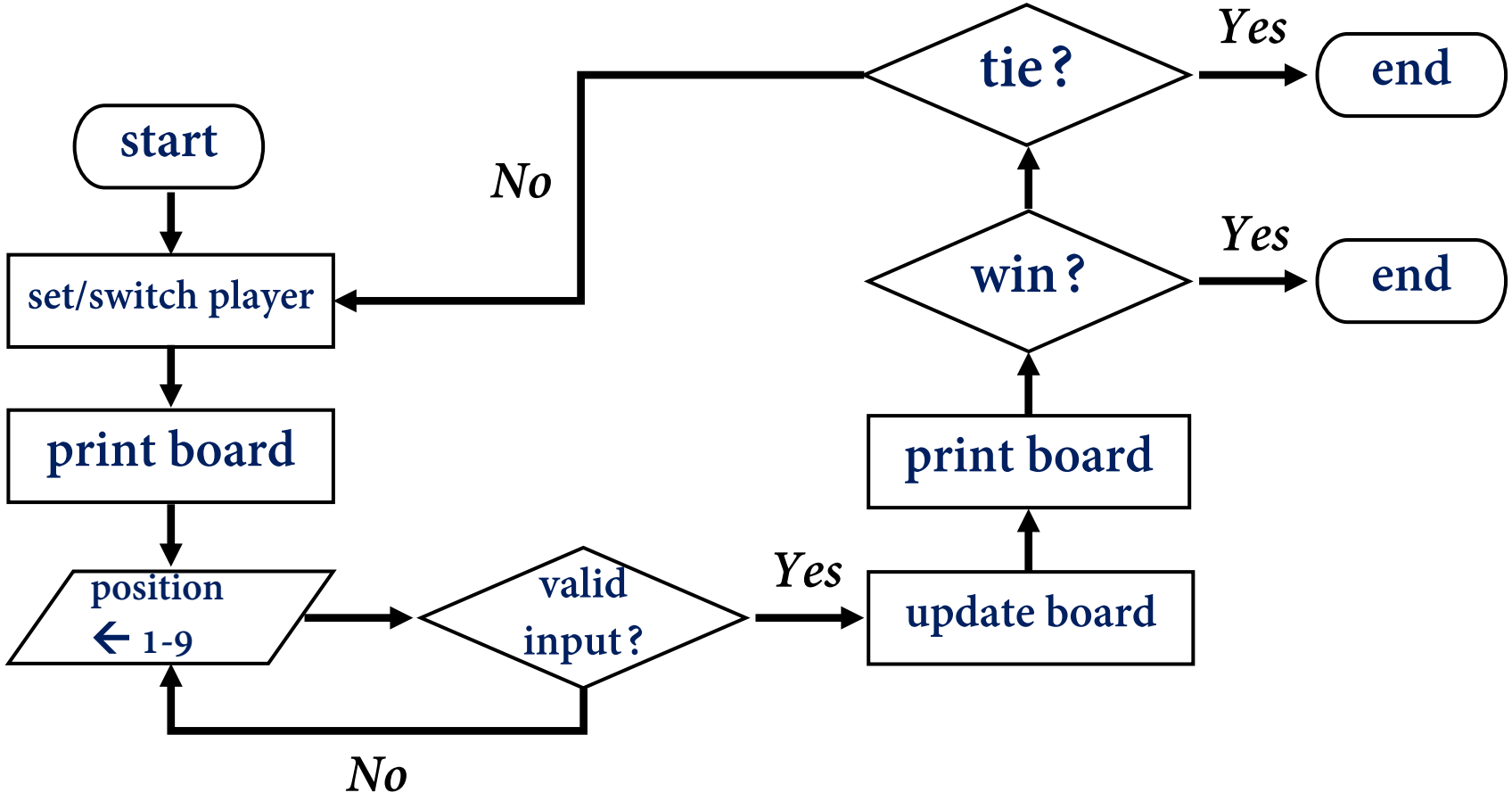
1. Consult the sample output
2. What functions do you need? Possible things you may reuse a lot –
   - Print a 3×3 board
   - Update the cells in the 3×3 board
   - Check if the termination condition reached: X/O win the game? Tie?
3. Will the game continue after the current round or game is over?
4. Error/exception handling: check user input – accept or reject?

Imperial College London

Questions?

That's it for now.

You can now proceed to the Lab 1 exercise.


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