

Programming 0

Course outline

Weeks at a glance

Week	Lecture Topics
1	Environment • Functions and Variables
2	Data Types and Operators • Logic and Decisions •
3	Loops and Lists • Lists as Aggregate Data
4	Functions
5	Packages • Testing
6	Script Architecture and Modules, Coursework release
7	Functions II: Repetition & DRY • Functions III: Argument Styles & Overloading
8	Classes
9	Persistence • Exceptions
10	Case Study: Simulation, Coursework hand-in week
11	Course Summary • Summary and Suggested Next Steps

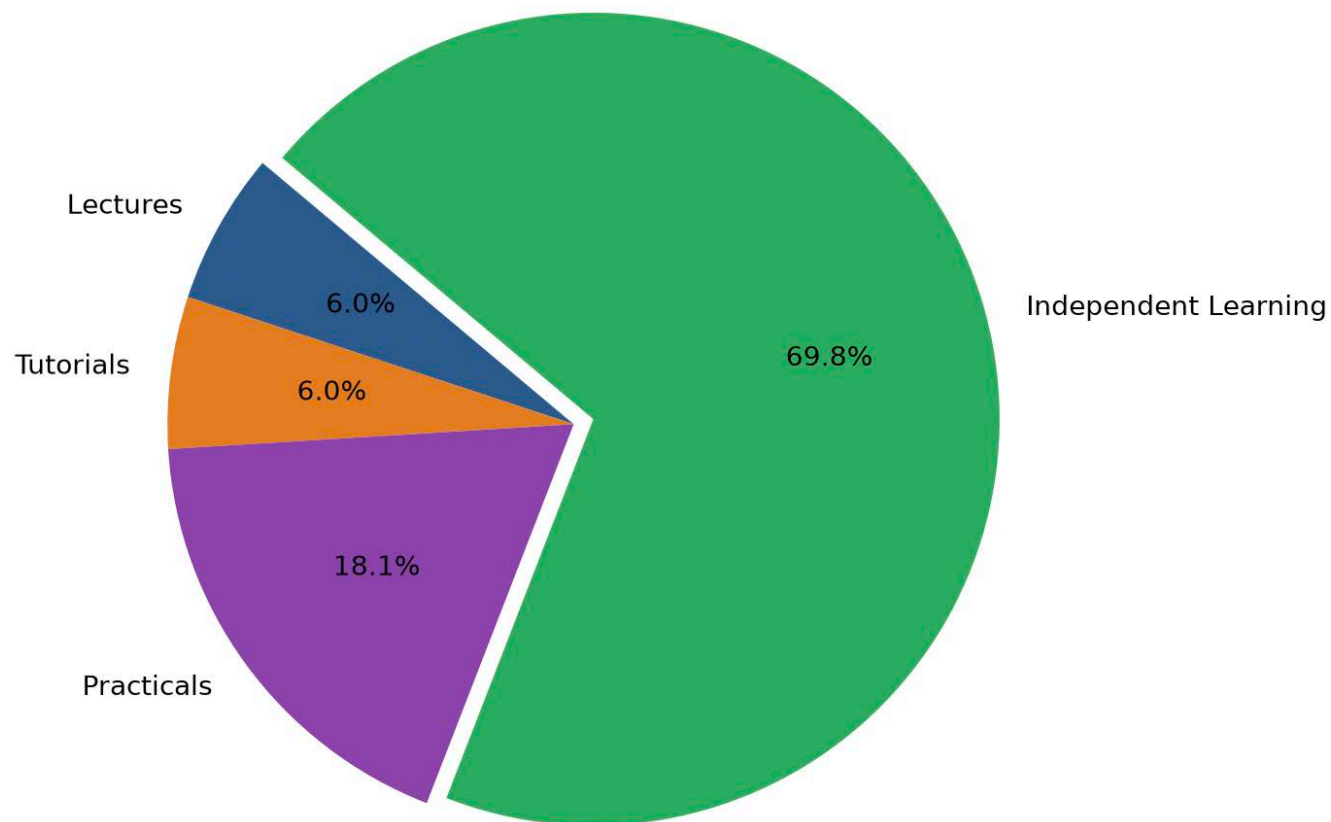
Workload

Hours

- Lectures = 11
- Tutorials = 11
- Practicals = 33
- Independent Learning = 127

My delivery

- Lectures = live (Monday) + recorded
- Tutorials = informal whiteboard sessions
- Practicals = lab-based (workbook & challenges)
- Independent Learning = time you spend coding outside class



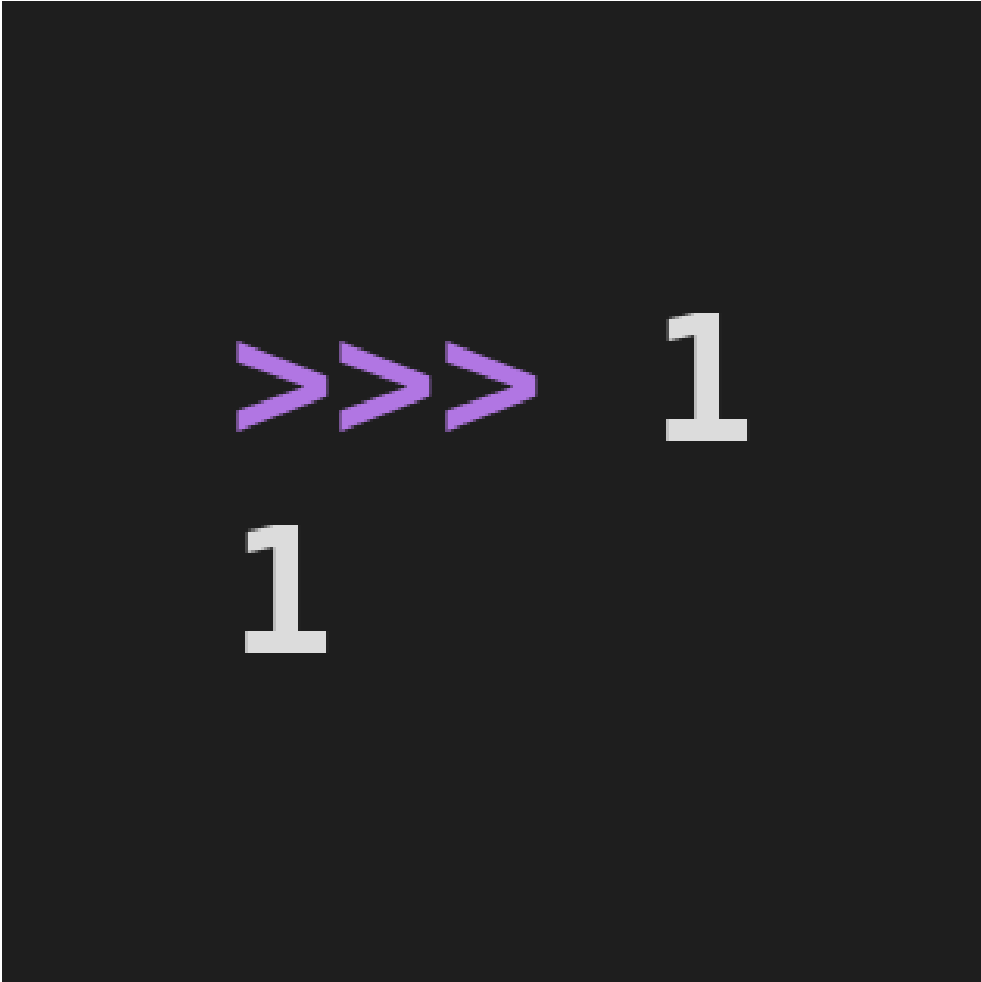
Week 1

■ Environment

- Files, directories, and paths — absolute vs relative
- The terminal: pwd, ls, cd, mkdir, touch, rm
- Naming conventions, and cloud backup before Git

■ Functions and Variables

- Editors, the command line, and the Python interpreter
- Functions, arguments, return values, variables
- Strings, numbers, and defining custom functions



```
>>> 1
1
```

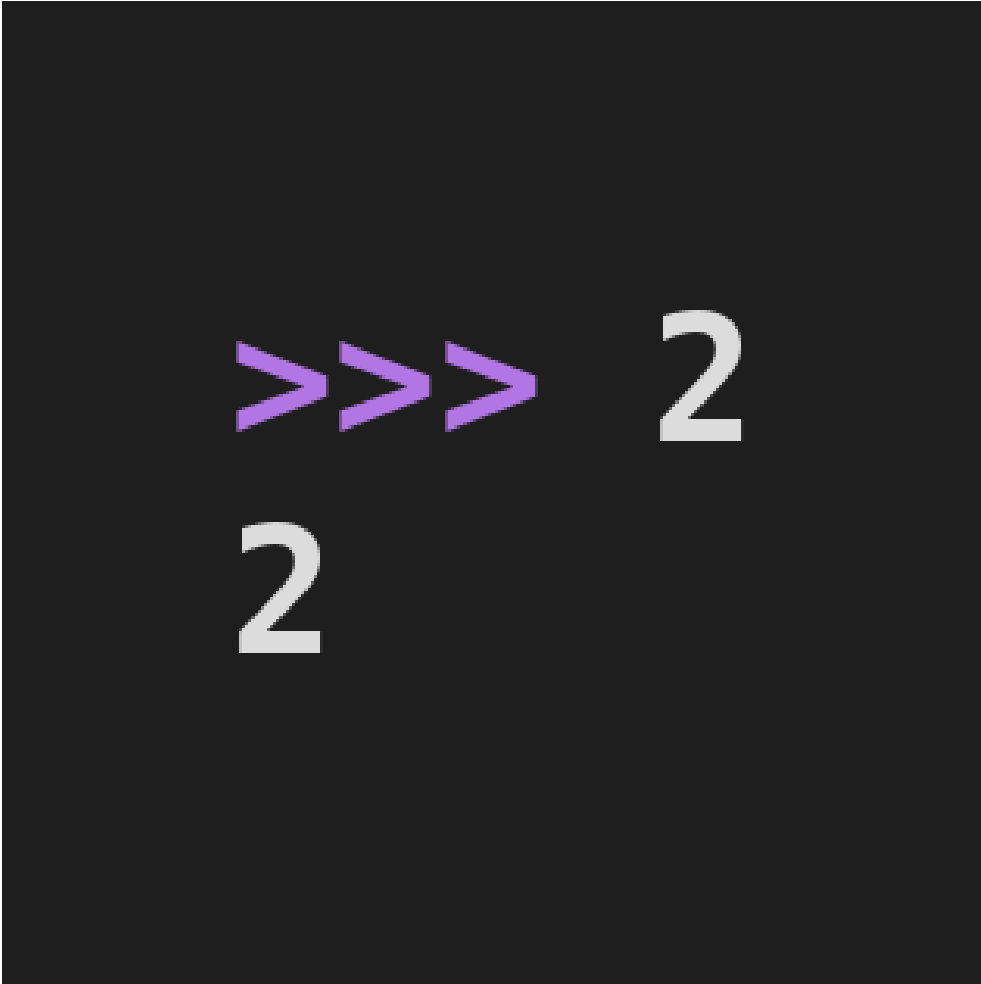
Week 2

■ Data Types and Operators

- Singular vs aggregate data; int and float
- The assignment operator, and basic operators
- Type conversion: int(), float(), str(), bool()

■ Logic and Decisions

- Boolean data from comparisons (<, >, ==, !=)
- Boolean keywords: and, not, or
- Branching with if / elif / else



>>> 2
2

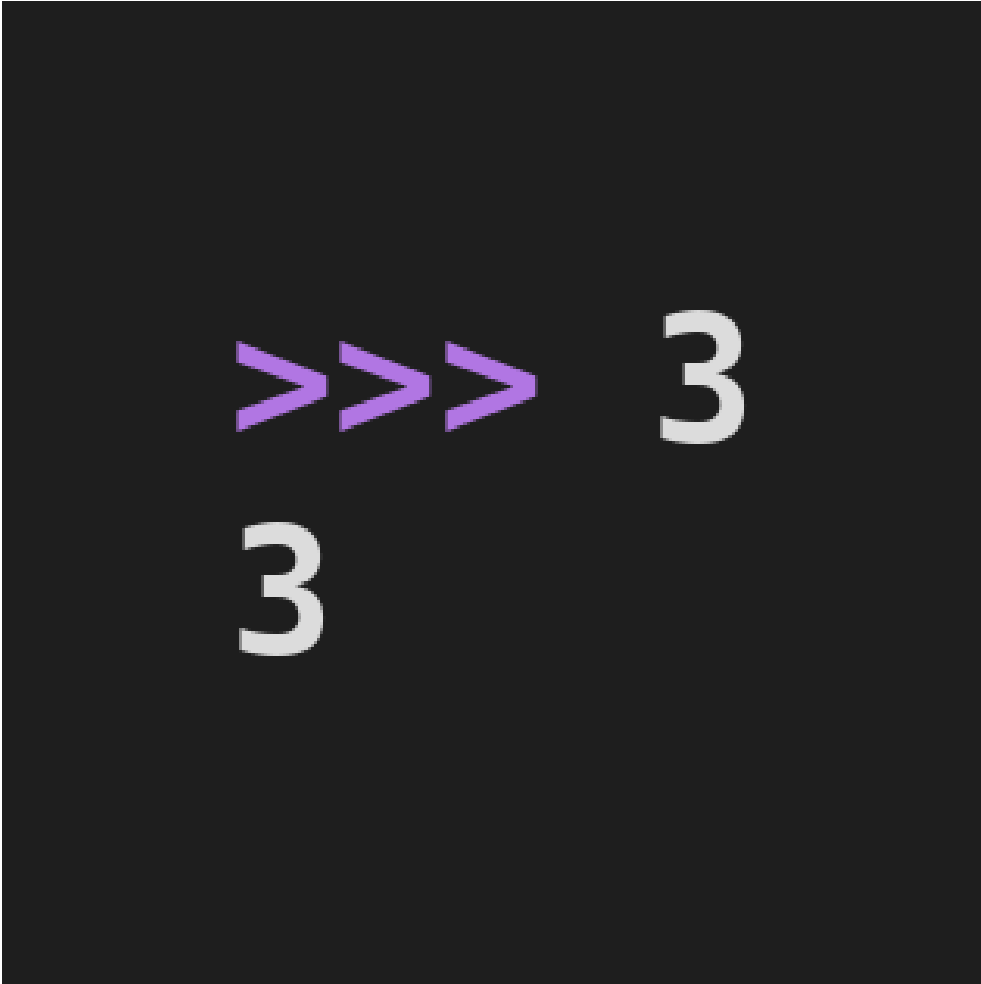
Week 3

■ Loops and Lists

- Why manual repetition doesn't scale
- for loops and range(); while loops
- Input validation with continue / break

■ Lists as Aggregate Data

- Building and modifying lists (the fridge example)
- Parallel lists, lists of lists, 2D coordinates
- Slicing, and common list methods

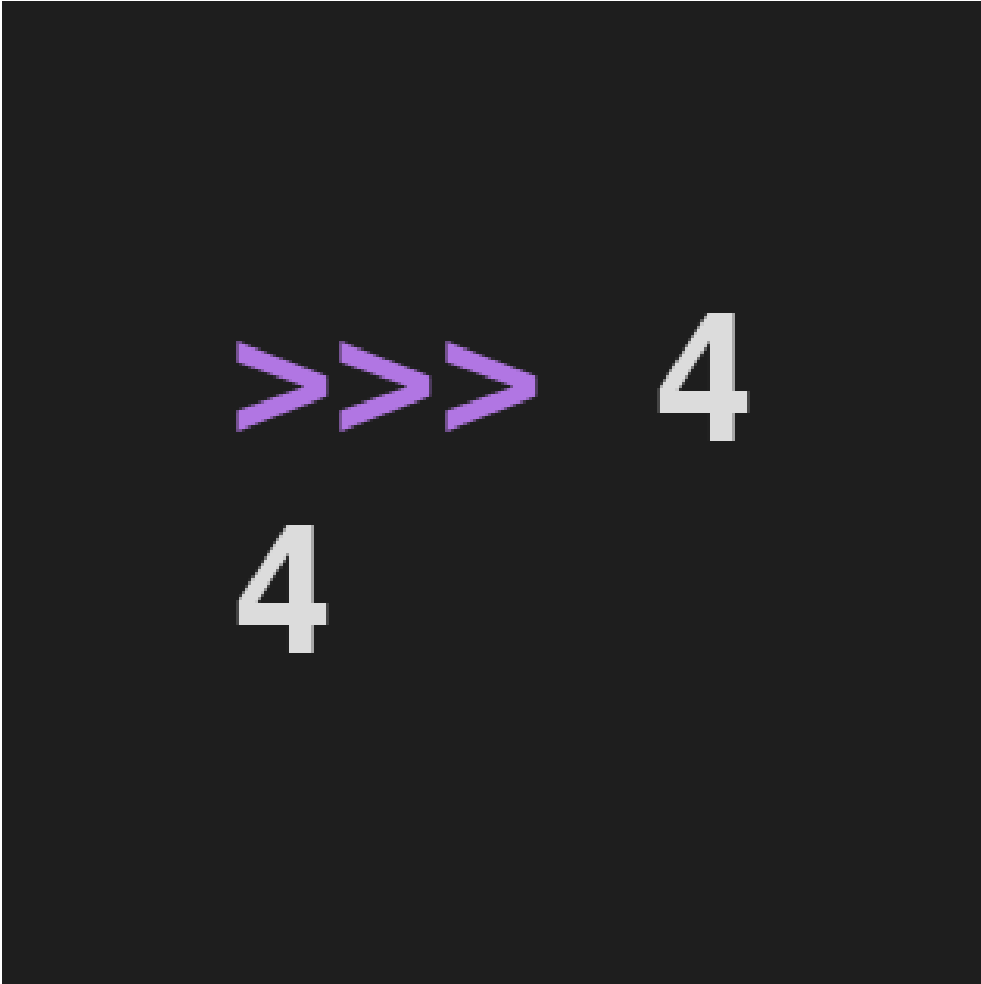


>>> 3
3

Week 4

■ Functions

- Functions as an instance of modular design
- Positional vs keyword arguments, defaults, *args
- Return values, multiple returns, and good practice



>>> 4
4

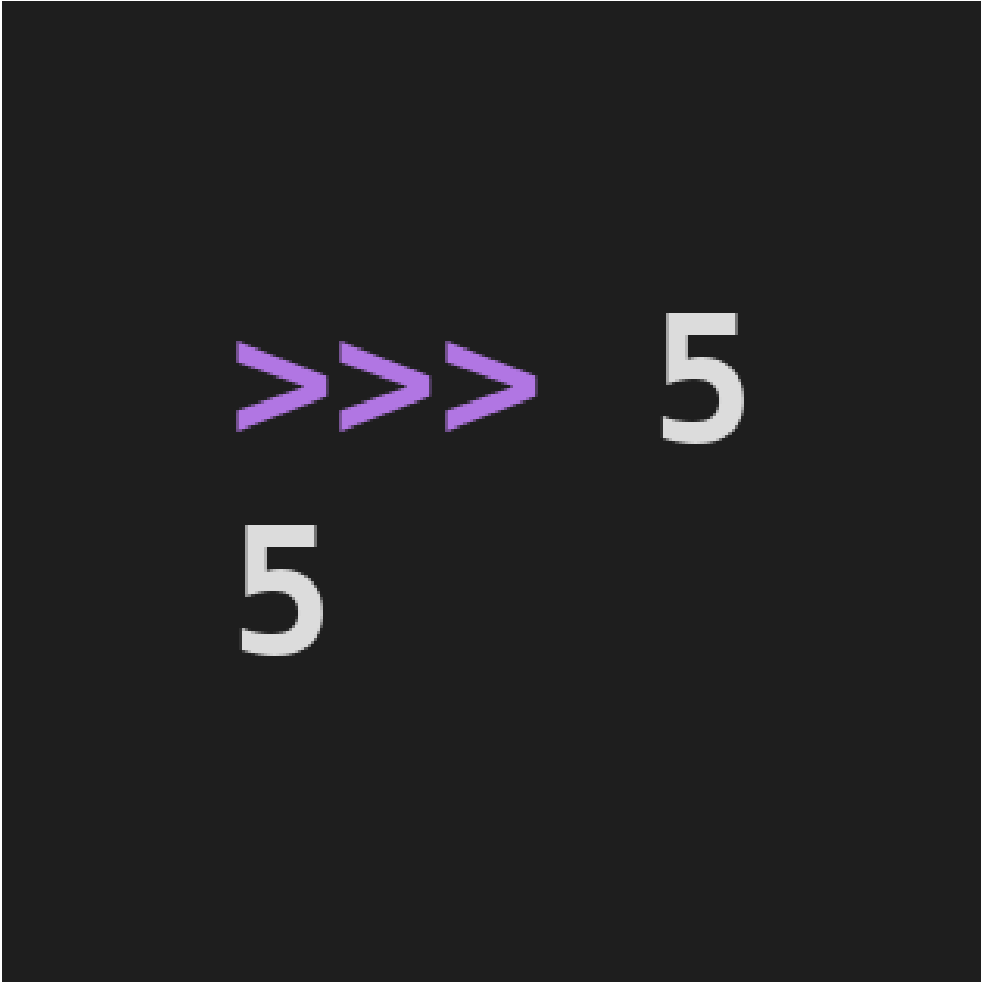
Week 5

■ Packages

- Scripts → modules → packages → libraries
- import, the random module, and namespaces
- pip, PyPI, and choosing the right package

■ Testing

- Expected vs actual behaviour, and assertions
- assert as automated pass/fail checking
- Unit testing with pytest



```
>>> 5
5
```

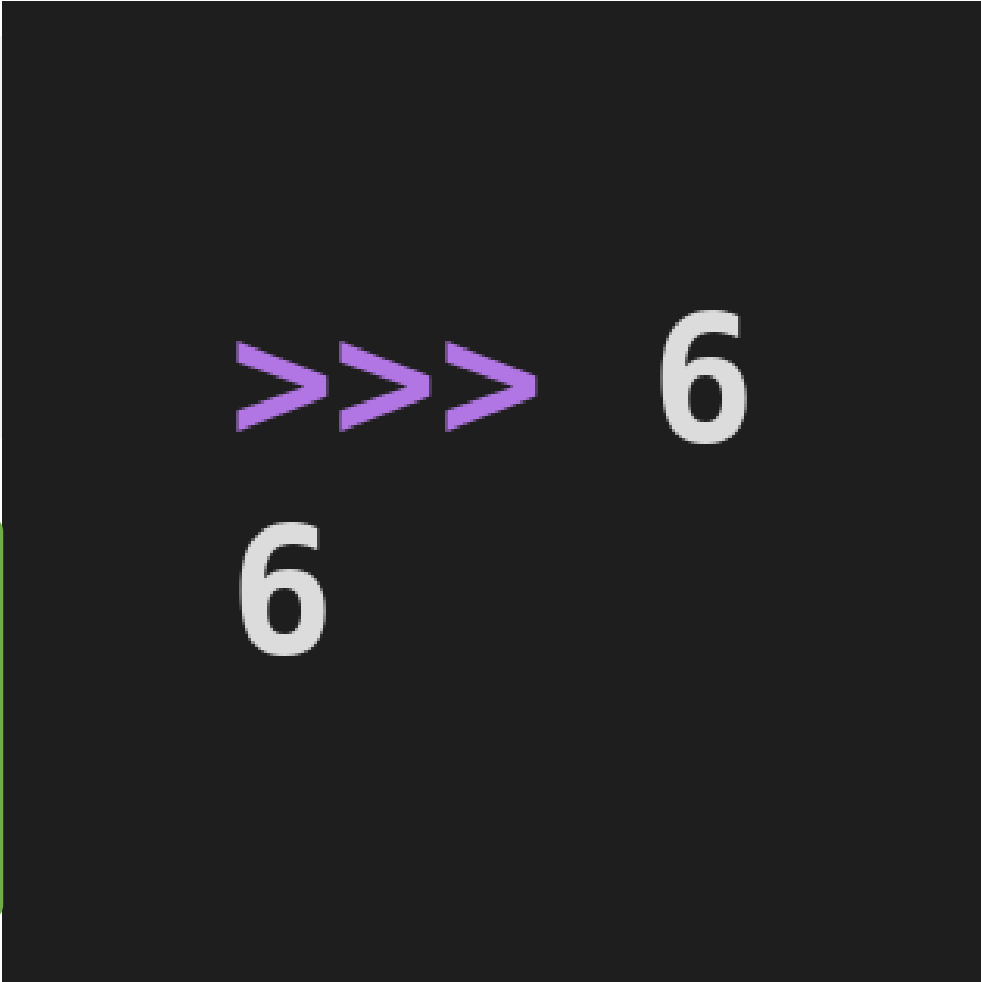
Week 6

■ Script Architecture and Modules

- Structuring larger programs across multiple files
- Organising code as scripts grow beyond one file
- Full lecture content to follow

■ Coursework

- Implementation of user-defined functions
 - Varying difficulty
- Deadline will be end of week 10



>>> 6
6

Week 7

■ Why Functions? DRY!

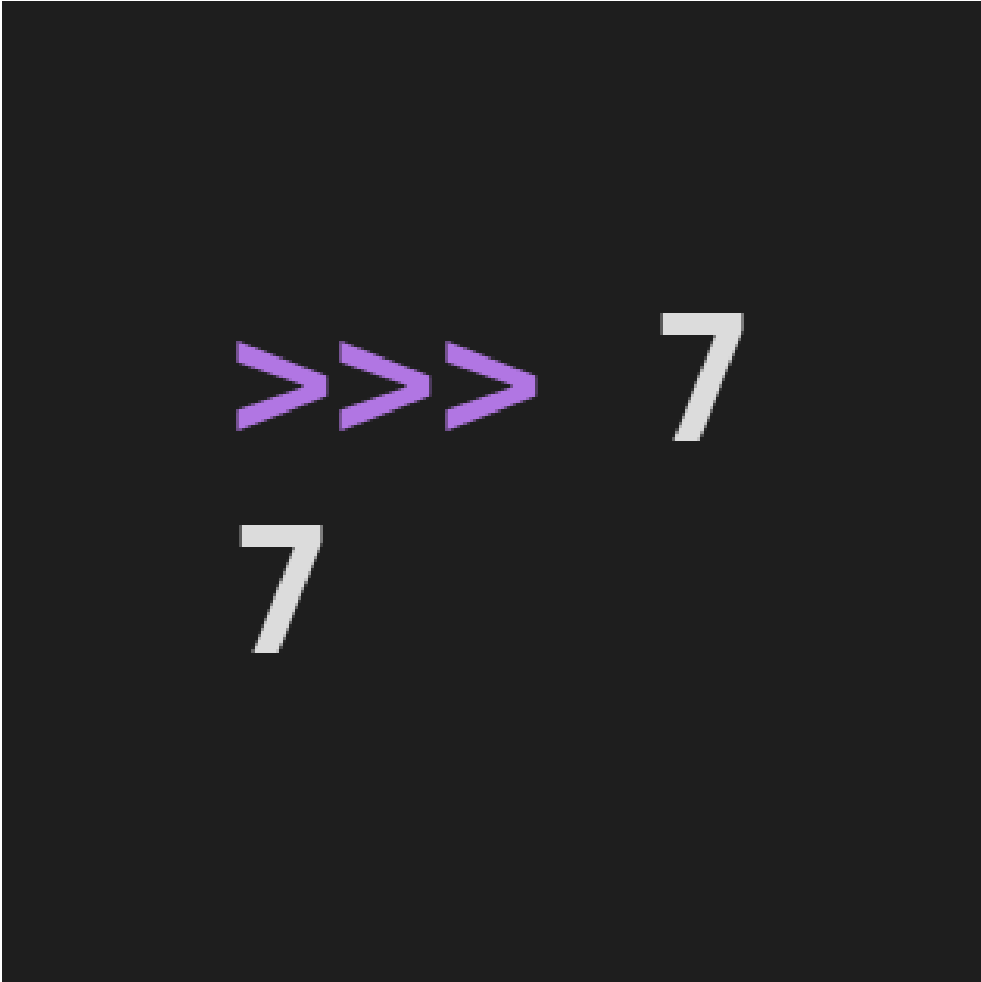
- Concrete thinking vs general thinking
- The DRY principle (Don't Repeat Yourself)
- Euclidean-distance and student-data examples

■ Functions II: Argument Styles and Overloading

- Argument styles and type hints
- Testing recap: expected vs actual behaviour
- Function overloading, Java vs Python

■ Functions II: Multiple Dispatch

- Namespaces, and how Python dispatches calls
- Why same-named functions overwrite each other
- Workarounds, and when overloading is worth it



>>> 7
7

Week 8

■ Classes

- Classes as templates; objects as instances
- self, __init__, and __str__
- Class-level vs instance-level variables



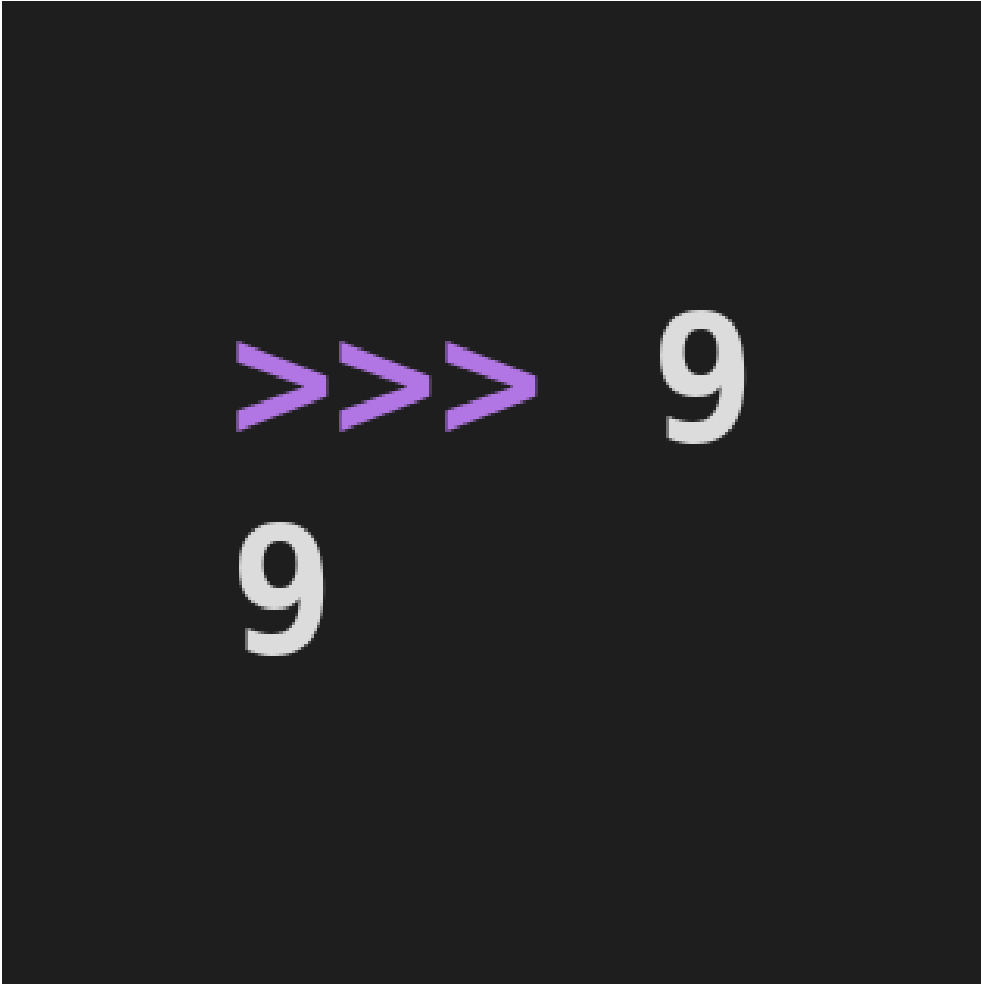
Week 9

■ Persistence

- Revisiting the input/output model
- CSV: reading and writing structured data
- A brief look ahead to databases

■ Command line Args & Exceptions

- `sys.argv`: reading input from the command line
- `try / except`, and `sys.exit()`
- Slicing arguments with `sys.argv[1:]`

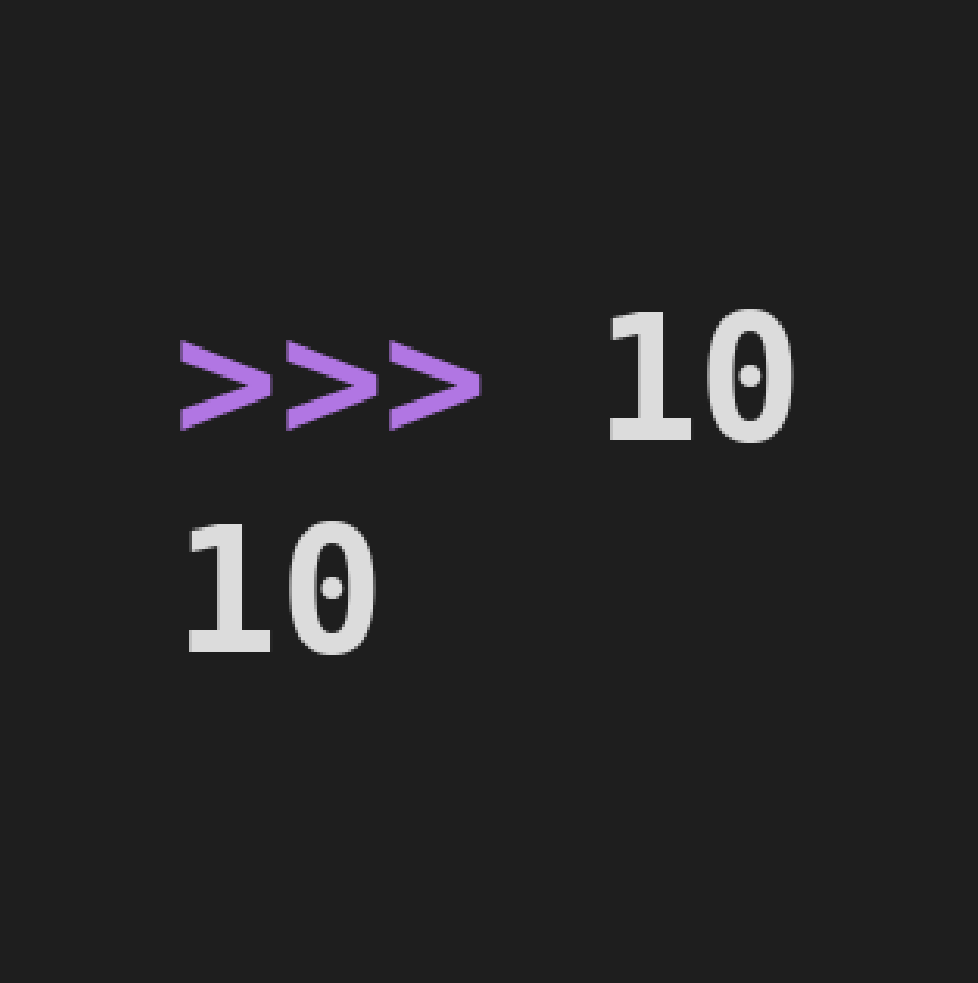


```
>>> 9
9
```

Week 10

■ Case study: simulation

- What simulation is, and why it's useful
- Automata: simple, local rules
- Traffic flow as the running example



>>> 10
10

Week 11

■ Course Summary

- A full week-by-week recap of the course
- A survey of further-learning resources

■ Summary and Suggested Next Steps

- The same recap, extended through the case study
- This course's environment vs others you'll meet
- Using large language models sensibly



>>> 11
11

Thanks!