

m01_workshop1_guided

September 22, 2025

1 M01 Python Engineering - Workshop 1

© 2025 Decoded Limited. All rights reserved. Website: <https://decoded.com>

Each challenge includes a stretch option for those who are already confident or looking for an extra challenge. These stretch tasks are completely optional, so don't worry if you don't get to them. The most important thing is to focus on the core challenge, as that's where the key learning happens. If you do feel ready to go further, give the stretch a try!

At the very end of the workshop resource, there is also a Super Stretch Challenge (Data Analyser). If you are an experienced Python programmer, feel free to work on this in parallel if you finish early.

1.1 Challenge: Write and run Python code

Goal: Build confidence using different ways to write and run Python code.

Use the Terminal / REPL:

- Go to your unique <https://code-lab...> link and enter your password.
- Open the terminal and launch the Python REPL using `python`.
- Run a couple of simple commands like `print()` and basic arithmetic.

Basic Arithmetic Operations in Python:

Operation

Syntax

Addition

`a + b`

Subtraction

`a - b`

Multiplication

`a * b`

Division

`a / b`

Floor Division

a // b

Modulus (Remainder)

a % b

Exponentiation

a ** b

Write and run a Python script in a code editor:

- In your IDE, create a new .py file (e.g. `my_script.py`).
- Write a simple `print()` statement and run it using `my_script.py`.

Try Jupyter Notebook:

- Go to your unique `https://nb-lab...` link and enter your password.
- Launch a new Jupyter Notebook.
- Create a Markdown cell that includes a heading.
- Run a couple of simple commands like `print()` and basic arithmetic.

Stretch: Create a Python script that asks for the user's name using `input()`, then greets them.

1.2 Challenge: Refactor code to meet PEP 8 standards

Goal: Begin developing habits for writing cleaner, more professional Python code by applying PEP 8 guidelines.

- Fix the following Python code so that it conforms to [PEP 8](#).
- Run the code and verify the output.

```
def add( x,y ):  
    return x+y  
result=add(5 ,3)  
print( "Result is",result )
```

Stretch: Once your fixed version works, try extending it:

- Modify the function to calculate both the sum and the difference.
- Rename the function to reflect what it now does.
- Include a simple docstring that explains what the function does.
- Return both results as a tuple.
- Unpack the tuple into two variables and print both values using an f-string.

[]:

1.3 Challenge: Explore Data Types

Goal: Apply knowledge from demonstration.

1. Create one variable of each of the following types: `int`, `float`, `bool`, `str`, `tuple`, `list`, `set`, and `dict`.
2. Print each variable and use the `type()` function to check its type.

Stretch:

- Add 10 to your int
- Multiply your float by 2
- Reverse the bool using `not`
- Convert your str to uppercase using `upper()`
- Add a new item to your list using `append()`
- Access the first item in your tuple
- Add a new number to your set using `add()`
- Add a new key-value pair to your dict

```
[ ]: # Example starter
temperature = 21.5
print(temperature, type(temperature))
```

21.5 <class 'float'>

1.4 Challenge: Control Flow in Python

Goal: Apply knowledge from demonstration.

1. Write an `if` statement that checks a person's age and prints:
 - "Child" if under 12
 - "Teenager" if 12–17
 - "Adult" otherwise
2. Use a `for` loop to print each number from 1 to 5
3. Use a `while` loop to count down from 3 to 1 and then print "Go!"

Stretch: Write a for loop that goes from 1 to 20 and for each number:

- Print "Fizz" if the number is divisible by 3
- Print "Buzz" if the number is divisible by 5
- Print "FizzBuzz" if the number is divisible by both 3 and 5
- Otherwise, just print the number

```
[ ]: # 1. Age check
age = 14
# your if/elif/else here
```

```
[ ]: # 2. For loop
# your for loop here
```

```
[ ]: # 3. While loop
# your while loop here
```

1.5 Super Stretch Challenge: Data Analyser

Goal: Apply your Python skills to analyse some numbers.

Write a program that takes a list of numbers (hardcoded or from user input) and:

- Prints the largest, smallest and average.
- Removes duplicates (hint: convert the list to a `set`).
- Stores the results in a dictionary with keys like "`max`", "`min`", "`average`".
- Print the dictionary using f-strings in a readable way.

[]:

END