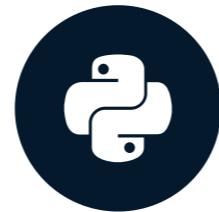


# What is Python?

INTRODUCTION TO PYTHON FOR DEVELOPERS



George Boorman

Curriculum Manager, DataCamp

# What we will learn

- *No prior knowledge required*
- What Python is and why it is popular for programming
- How to execute code using the DataCamp UI
- Code comments and data types
- Performing calculations
- Creating and manipulating variables
- Building conditional workflows
- For and while loops

# What is Python?



- Open-source: free to use
- Packages: use other's code to avoid starting from scratch
- Syntax resembles natural language

# The swiss army knife of programming languages

## Use-cases

- Task automation
- Web apps
- Artificial Intelligence (AI)
  - Web scraping
  - Content generation/summary
  - Image recognition



<sup>1</sup> <https://unsplash.com/@dmjdenise>

# Python in Big Tech



NETFLIX

<sup>1</sup> <https://instagram-engineering.com/what-powers-instagram-hundreds-of-instances-dozens-of-technologies-adf2e22da2ad>

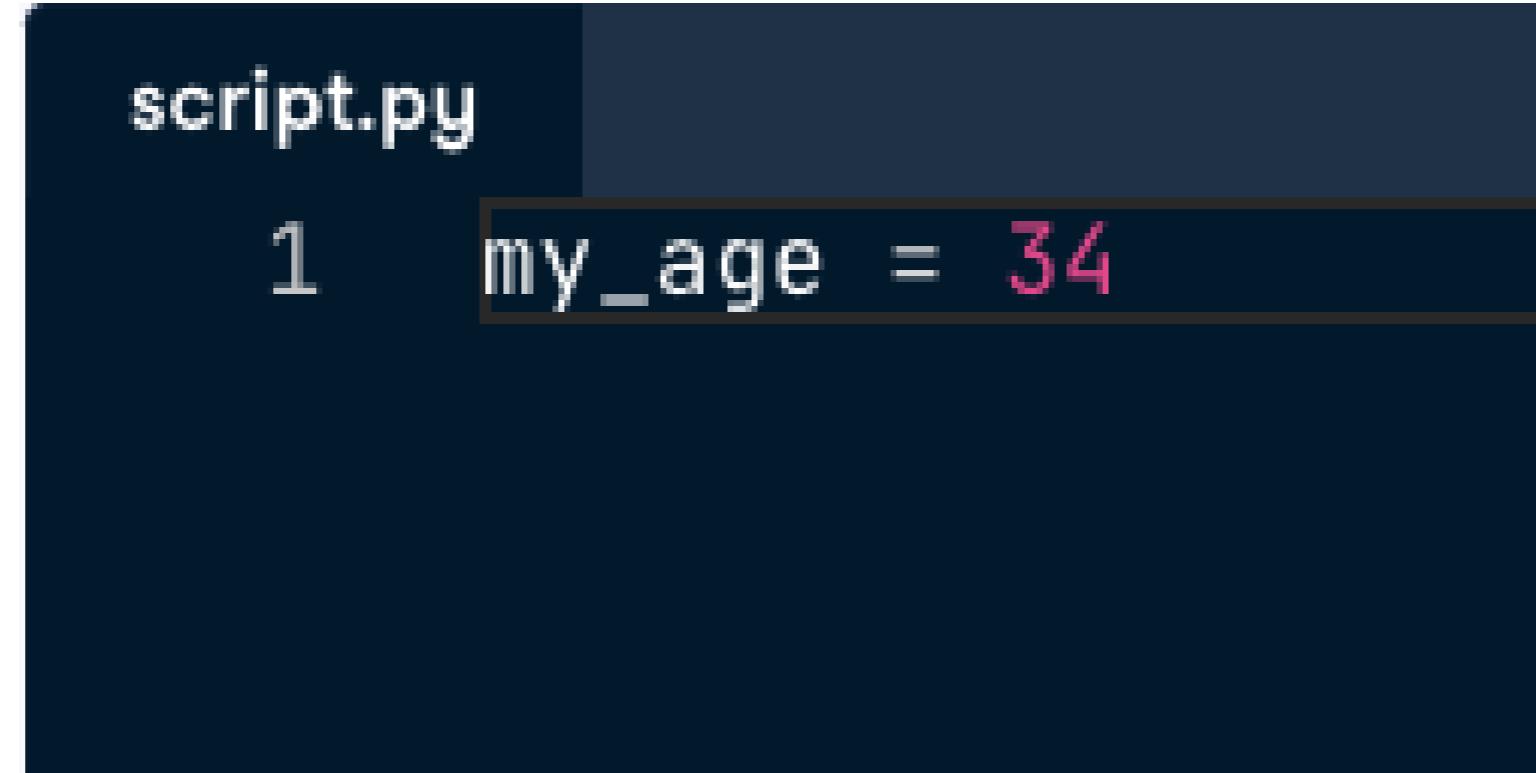
# Not always fit-for-purpose



- Difficult to cut down with a swiss army knife
- Easier with an axe or chainsaw
- Not suitable:
  - Mobile apps - `Swift` or `Kotlin`
  - Hardware programming - `C`

<sup>1</sup> <https://unsplash.com/@theoneofussocialclub>

# Speed of development



A screenshot of a code editor window titled "script.py". The code editor shows a single line of Python code: "my\_age = 34". The line "my\_age = 34" is highlighted with a pink-to-white gradient background.

```
script.py
1 my_age = 34
```

- Uses natural language
- Easier to create and understand code

# Speed of development



- Python is slow(er) than some other languages
- But Python is easier to use
  - so it can be quicker for development

# **Let's practice!**

**INTRODUCTION TO PYTHON FOR DEVELOPERS**

# How to run Python code

INTRODUCTION TO PYTHON FOR DEVELOPERS



George Boorman

Curriculum Manager, DataCamp

# The DataCamp interface

The screenshot shows the DataCamp interface for a Python exercise. On the left, there's a sidebar with a 'Exercise' tab and a 'Python as a calculator' section containing text and code examples. Below it is an 'Instructions' section with a yellow '100 XP' badge and a list of tasks. A 'Take Hint (-30 XP)' button is also present. The main area has a dark theme with a code editor titled 'script.py' containing basic arithmetic operations. To the right of the code editor is a 'Python Shell' window with an input field labeled 'In [1]:'. At the bottom right are buttons for 'Run Code' and 'Submit Answer'.

Exercise

## Python as a calculator

Python is perfectly suited to do basic calculations. It can do addition, subtraction, multiplication and division.

The code in the script gives some examples...

Now it's your turn to practice!

Instructions 100 XP

- Print the sum of `4 + 5`.
- Print the result of subtracting `5` from `5`.
- Multiply `5` by `5`.
- Divide `10` by `2`.

Take Hint (-30 XP)

script.py

```
1 # Addition
2 print(4 + 5)
3
4 # Subtraction
5 print
6 # print
7 # Multiplication
8
9
10 # Division
11
```

Python Shell

In [1]:

Run Code

Submit Answer

# IPython Shell

The screenshot shows a Python exercise interface. On the left, there's an 'Exercise' panel titled 'Python as a calculator'. It contains text about Python being suited for basic calculations and examples of addition, subtraction, multiplication, and division. Below this is a list of tasks:

- Print the sum of `5 + 5`.
- Print the result of subtracting `5` from `5`.
- Multiply `3` by `5`.
- Divide `10` by `2`.

A button labeled 'Take Hint (-30 XP)' is present. On the right, the main area shows a code editor with a file named 'script.py' containing the following code:

```
script.py
1 # Addition
2
3
4 # Subtraction
5
6
7 # Multiplication
8
9
10 # Division
11
```

Below the code editor is an 'IPython Shell' interface with a command history entry: 'In [1]:'. A green box highlights this shell area. At the bottom right of the main interface are buttons for 'Run Code' and 'Submit Answer'.

# Python Script

The screenshot shows a Python script exercise interface. At the top left, there's a navigation bar with icons for 'Learn', 'Courses', and 'Introduction to Python'. Below it, a sidebar on the left contains sections for 'Exercise' (with a 'Python as a calculator' heading), 'Instructions' (listing tasks like 'Print the sum of 5 + 5.'), and a 'Take Hint (-30 XP)' button. The main area is a code editor titled 'script.py' containing the following code:

```
script.py
1 # Addition
2
3
4 # Subtraction
5
6
7 # Multiplication
8
9
10 # Division
11
```

The code editor has a green border and includes buttons for 'Run Code' and 'Submit Answer' at the bottom right. Below the code editor is an 'IPython Shell' section with the text 'In [1]:'.

# Code comments

```
# This is a code comment
```

- Code is read by more than it is written
- Code is shared with others
- Comments help people (including us) to understand our code

# Performing calculations

**Instructions** | 100 XP

- Print the sum of `5 + 5`.
- Print the result of subtracting `5` from `5`.
- Multiply `3` by `5`.
- Divide `10` by `2`.

**Take Hint (-30 XP)**

IPython Shell

In [1]:

Run Code | Submit Answer

# Performing calculations

The screenshot shows a learning platform interface for an 'Introduction to Python' course. On the left, there's a sidebar with navigation links like 'Learn', 'Courses', and 'Introduction to Python'. Below that is a section titled 'Exercise' with the sub-section 'Python as a calculator'. It contains text about Python being suited for basic calculations and some examples. A list of tasks under 'Instructions' includes:

- Print the sum of `4 + 5`.
- Print the result of subtracting `5` from `5`.
- Multiply `3` by `5`.
- Divide `10` by `2`.

A 'Take Hint (-50 XP)' button is also present. The main workspace is a dark-themed code editor with tabs for 'script.py' and 'IPython Shell'. In the code editor, the file 'script.py' contains the number `1`. Below the code editor are buttons for 'Run Code' and 'Submit Answer'. The IPython Shell window shows the prompt `In [1]:`.

- Use `print()` to generate output from script

# More calculations

Exercise

Python as a calculator

Python is perfectly suited to do basic calculations. It can do addition, subtraction, multiplication and division.

The code in the script gives some examples.

Now it's your turn to practice!

Instructions

- Print the sum of `4 + 5`.
- Print the result of subtracting `5` from `5`.
- Multiply `5` by `5`.
- Divide `10` by `2`.

Take Hint (-30 XP)

script.py

```
1 # Addition
2 print(4 + 5)
3
4 # Subtraction
5 print
6 # print
7 # Multiplication
8
9
10 # Division
11
```

Local

Run Code

Submit Answer

Python Shell

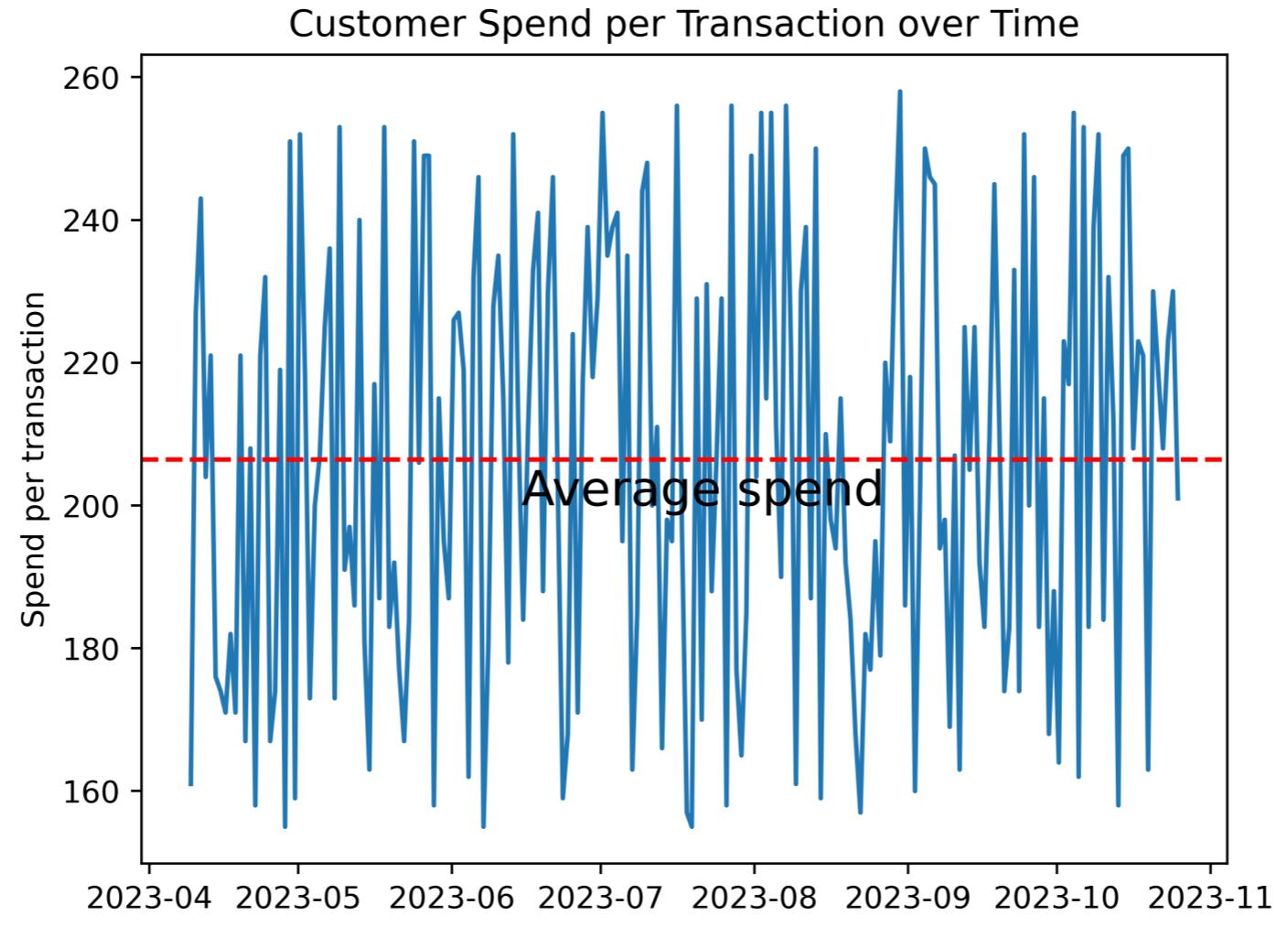
In [1]:

# Calculations cheat sheet

Syntax	Action	Example	Output
*	Multiply	4 * 10	40
+	Addition	7 + 9	16
-	Subtract	23 - 4	19
/	Division	27 / 3	9
**	Power	3 ** 2	9
%	Modulo	7 % 4	3

# Numbers in programming

- Age verification
- Average value over a time period

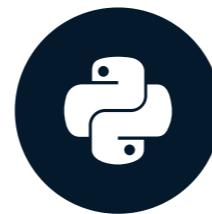


# **Let's practice!**

**INTRODUCTION TO PYTHON FOR DEVELOPERS**

# Variables and data types

INTRODUCTION TO PYTHON FOR DEVELOPERS



George Boorman

Curriculum Manager, DataCamp

# Variables

```
# Create the customer_age variable with a value of 25  
customer_age = 25
```

- `variable_name` : Name of the variable (case-sensitive)
- `=` : For assigning a value
- `value` : The value to assign to the variable

# Why use variables?

- Avoid having to retype the same information
- Can update variables if conditions change

```
# Original value of customer_age  
customer_age = 25  
# Update customer_age to 26  
customer_age = 26
```

# Integers

```
# This is an integer  
180
```

```
# This is also an integer  
customer_id = 180
```

- No need to tell Python what data type the variable is!

```
type(customer_id)
```

```
<class 'int'>
```

# FLOATS

```
# Decimal values  
customer_average_spend = 55.28
```

```
# Check the data type  
type(customer_average_spend)
```

```
<class 'float'>
```

# Strings

```
# Single quotes  
customer_name = 'George Boorman'  
  
# Double quotes also works  
customer_name = "George Boorman"  
  
# Check the data type  
type(customer_name)
```

```
<class 'str'>
```

# Booleans

```
# Define active and inactive customer variables  
active_customer = True  
inactive_customer = False  
  
# Check active_customer data type  
type(active_customer)
```

```
<class 'bool'>
```

- Case sensitive: `true` and `false` won't work

# Variable naming

```
# Define total_spend  
total_spend = 3150.96
```

```
# Try to define using a space  
total spend = 3150.96
```

SyntaxError: invalid syntax

```
# Try to name a variable starting with a number  
2023_spend = 3150.96
```

SyntaxError: invalid decimal literal

# Case conventions

```
# Snake case  
total_spend = 3150.96
```

```
# CamelCase  
TotalSpend = 3150.96
```

- Both are acceptable - use personal preference

# Calculations with variables

```
# Define num_transactions  
num_transactions = 57  
  
# Calculate total_spend by multiplying two variables  
total_spend = num_transactions * customer_average_spend  
  
# Print two variables  
print(customer_name, total_spend)
```

George Boorman 3150.96

# **Let's practice!**

**INTRODUCTION TO PYTHON FOR DEVELOPERS**