

# Hello Python!

INTRODUCTION TO PYTHON



**Hugo Bowne-Anderson**  
Data Scientist at DataCamp

# How you will learn

The screenshot shows a DataCamp exercise interface for "Calculations with variables".

**Exercise Overview:** Course Outline, script.py, 500 XP.

**Instructions:**

- Create a variable `growth_multiplier`, equal to `1.1`.
- Create a variable `result`, equal to the amount of money you saved after `7` years.
- Print out the value of `result`.

**Code Editor (script.py):**

```
1 # Create a variable savings
2 savings = 100
3
4 # Create a variable growth_multiplier
5 growth_multiplier = 1.1
6
7 # Calculate result
8 result = savings |
9
10 # Print out result
11
```

**Run Code** and **Submit Answer** buttons are visible.

**IPython Shell:**

```
In [1]: |
```

# Python



- General purpose: build anything
- Open source! Free!
- Python packages, also for data science
  - Many applications and fields
- Version 3.x - <https://www.python.org/downloads/>

# IPython Shell

## Execute Python commands

The screenshot shows a DataCamp exercise interface for "Calculations with variables".

**Exercise Overview:** The exercise is titled "Calculations with variables". It asks the user to calculate the amount of money saved after 7 years of investing \$100 at a 10% annual growth rate.

**Code Editor:** The code editor contains the following Python script:

```
script.py
1 # Create a variable savings
2 savings = 100
3
4 # Create a variable growth_multiplier
5
6
7 # Calculate result
8
9
10 # Print out result
11
```

**Instructions:** The instructions list the following tasks:

- Create a variable `growth_multiplier`, equal to `1.1`.
- Create a variable, `result`, equal to the amount of money you saved after `7` years.
- Print out the value of `result`.

**IPython Shell:** The IPython shell window is highlighted with a red border. It displays the prompt "In [1]: |".

**UI Elements:** The interface includes a "Course Outline" tab, a "Run Code" button, and a "Submit Answer" button.

# IPython Shell

The screenshot shows a DataCamp exercise interface for "Calculations with variables".

**Exercise Overview:** Course Outline, script.py, 500 XP.

**Instructions:**

- Create a variable `growth_multiplier`, equal to `1.1`.
- Create a variable, `result`, equal to the amount of money you saved after `7` years.
- Print out the value of `result`.

**Code Completion:** A code editor window titled "script.py" contains the following code:

```
1 # Create a variable savings
2 savings = 100
3
4 # Create a variable growth_multiplier
5
6
7 # Calculate result
8
9
10 # Print out result
11
```

**IPython Shell:** An IPython shell window titled "IPython Shell" shows the command "In [1]: |".

# Python Script

- Text files - .py
- List of Python commands
- Similar to typing in IPython Shell

The screenshot shows a DataCamp exercise interface for a Python script named `script.py`. The code editor contains the following script:

```
1 # Create a variable savings
2 savings = 100
3
4 # Create a variable growth_multiplier
5 growth_multiplier = 1.1
6
7 # Calculate result
8 |
9
10 # Print out result
11
```

The line `8 |` is highlighted with a red box, indicating where the user should type the next line of code. Below the code editor are three buttons: `Run Code` and `Submit Answer`. To the left of the code editor is a sidebar with the title **Calculations with variables** and instructions about calculating money after 7 years. At the bottom left is a list of tasks with **100 XP** available.

**Instructions**

- Create a variable `growth_multiplier`, equal to `1.1`.
- Create a variable, `result`, equal to the amount of money you saved after `7` years.
- Print out the value of `result`.

**IPython Shell**

In [1]: |

# Python Script

The screenshot shows a DataCamp Python script exercise interface. On the left, there's a sidebar with the DataCamp logo, a 'Course Outline' button, and a 'script.py' file tab. The main area has a title 'Calculations with variables' and instructions about calculating money after 7 years. It shows a code snippet: `100 * 1.1 ** 7`. Below it, text explains using variables instead of hard-coded values. A list of tasks for the exercise includes creating variables for growth multiplier and result, and printing the result. There's also a 'Take Hint (-30 XP)' button. On the right, there's an 'IPython Shell' window showing 'In [1]:' and a 'Run Code' button.

Calculations with variables

Remember how you calculated the money you ended up with after 7 years of investing \$100? You did something like this:

```
100 * 1.1 ** 7
```

Instead of calculating with the actual values, you can use variables instead. The `savings` variable you've created in the previous exercise represents the \$100 you started with. It's up to you to create a new variable to represent `1.1` and then redo the calculations!

Instructions 100 XP

- Create a variable `growth_multiplier`, equal to `1.1`.
- Create a variable, `result`, equal to the amount of money you saved after `7` years.
- Print out the value of `result`.

Take Hint (-30 XP)

IPython Shell

In [1]:

Run Code

Submit Answer

# Python Script

The screenshot shows a DataCamp exercise interface for a Python script. The top navigation bar includes the DataCamp logo, a 'Course Outline' tab, and several icons. The main area is titled 'script.py'. On the left, there's an 'Exercise' sidebar with the title 'Calculations with variables'. It contains instructions about calculating money after 7 years of investing \$100, showing the code `100 * 1.1 ** 7`, and a note about using variables instead. Below this are 'Instructions' (100 XP) and a list of tasks:

- Create a variable `growth_multiplier`, equal to `1.1`.
- Create a variable, `result`, equal to the amount of money you saved after `7` years.
- Print out the value of `result`.

At the bottom of the sidebar is a 'Take Hint (-30 XP)' button. On the right, there's an 'IPython Shell' window showing 'In [1]: |' and three buttons: 'Run Code', 'Submit Answer', and a refresh icon.

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

# DataCamp Interface

The screenshot shows the DataCamp Python exercise interface. At the top, there's a navigation bar with a DataCamp logo, course outline, and other course-related icons. Below it, the main area is divided into several sections:

- Exercise:** A title "Calculations with variables" and a note about calculating money after 7 years of investing \$100.
- Code Editor:** A code file named "script.py" containing the following Python code:

```
1 # Create a variable savings
2 savings = 100
3
4 # Create a variable growth_multiplier
5
6
7 # Calculate result
8
9
10 # Print out result
11 |
```
- Instructions:** A list of tasks:
  - Create a variable `growth_multiplier`, equal to `1.1`.
  - Create a variable, `result`, equal to the amount of money you saved after `7` years.
  - Print out the value of `result`.
- IPython Shell:** A panel labeled "In [1]:" which is currently empty.

At the bottom right of the main area, there are "Run Code" and "Submit Answer" buttons.

# Let's practice!

INTRODUCTION TO PYTHON

# Variables and Types

INTRODUCTION TO PYTHON



**Hugo Bowne-Anderson**  
Data Scientist at DataCamp

# Variable

- Specific, case-sensitive name
- Call up value through variable name
- 1.79 m - 68.7 kg

```
height = 1.79
```

```
weight = 68.7
```

```
height
```

```
1.79
```

# Calculate BMI

```
height = 1.79  
weight = 68.7  
height
```

```
1.79
```

$$\text{BMI} = \frac{\text{weight}}{\text{height}^2}$$

```
68.7 / 1.79 ** 2
```

```
21.4413
```

```
weight / height ** 2
```

```
21.4413
```

```
bmi = weight / height ** 2  
bmi
```

```
21.4413
```

# Reproducibility

```
height = 1.79  
weight = 68.7  
bmi = weight / height ** 2  
print(bmi)
```

21.4413

# Reproducibility

```
height = 1.79  
weight = 74.2 # <--  
bmi = weight / height ** 2  
print(bmi)
```

23.1578

# Python Types

```
type(bmi)
```

```
float
```

```
day_of_week = 5  
type(day_of_week)
```

```
int
```

# Python Types (2)

```
x = "body mass index"  
y = 'this works too'  
  
type(y)
```

```
str
```

```
z = True  
type(z)
```

```
bool
```

# Python Types (3)

```
2 + 3
```

```
5
```

```
'ab' + 'cd'
```

```
'abcd'
```

- Different type = different behavior!

# Let's practice!

INTRODUCTION TO PYTHON

