



INTRO TO PYTHON FOR DATA SCIENCE

Hello Python!

What you will learn

- Python
- Specifically for Data Science
- Store data
- Manipulate data
- Tools for data analysis

How you will learn

Python

- Guido Van Rossum
- 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

DataCamp Course Outline

Calculations with variables 100xp

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

```
100 * 1.10 ** 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. Up to you to create a new variable to represent `1.10` and then redo the calculations!

Instructions

- Create a variable `factor`, equal to `1.10`.
- Use `savings` and `factor` to calculate the amount of money you end up with after 7 years. Store the result in a new variable, `result`.
- Print out the value of `result`.

[Take Hint \(-30xp\)](#)

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

Submit Answer

IPython Shell

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



IPython Shell

Python Script

- Text Files - .py
- List of Python Commands
- Similar to typing in IPython Shell

Python Script

DataCamp Interface

The screenshot displays the DataCamp interface for a coding exercise. The top navigation bar includes the DataCamp logo, a 'Course Outline' menu, and a green status indicator. The main content area is split into three sections: a problem description on the left, a code editor in the middle, and an IPython shell at the bottom.

Calculations with variables 100xp

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

```
100 * 1.10 ** 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. Up to you to create a new variable to represent `1.10` and then redo the calculations!

Instructions

- Create a variable `factor`, equal to `1.10`.
- Use `savings` and `factor` to calculate the amount of money you end up with after 7 years. Store the result in a new variable, `result`.
- Print out the value of `result`.

[Take Hint \(-30xp\)](#)

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

Script

[Submit Answer](#)

```
IPython Shell
In [1]: |
```

Shell



INTRO TO PYTHON FOR DATA SCIENCE

Let's practice!



INTRO TO PYTHON FOR DATA SCIENCE

Variables and Types



Variable

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

```
In [1]: height = 1.79
```

```
In [2]: weight = 68.7
```

```
In [3]: height
```

```
Out[3]: 1.79
```



Calculate BMI

```
In [1]: height = 1.79
```

```
In [2]: weight = 68.7
```

```
In [3]: height
```

```
Out[3]: 1.79
```

```
In [4]: 68.7 / 1.79 ** 2
```

```
Out[4]: 21.4413
```

```
In [5]: weight / height ** 2
```

```
Out[5]: 21.4413
```

```
In [6]: bmi = weight / height ** 2
```

```
In [7]: bmi
```

```
Out[7]: 21.4413
```

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



Reproducibility

 my_script.py

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

Output:
21.4413

Reproducibility

 my_script.py

```
height = 1.79
weight = 74.2 ←
bmi = weight / height ** 2
print(bmi)
```

Output:
23.1578

Python Types

```
In [8]: type(bmi)
```

```
Out[8]: float
```

```
In [9]: day_of_week = 5
```

```
In [10]: type(day_of_week)
```

```
Out[10]: int
```



Python Types (2)

```
In [11]: x = "body mass index"
```

```
In [12]: y = 'this works too'
```

```
In [13]: type(y)
```

```
Out[13]: str
```

```
In [14]: z = True
```

```
In [15]: type(z)
```

```
Out[15]: bool
```



Python Types (3)

```
In [16]: 2 + 3  
Out[16]: 5
```

Different type = different behavior!

```
In [17]: 'ab' + 'cd'  
Out[17]: 'abcd'
```



INTRO TO PYTHON FOR DATA SCIENCE

Let's practice!