


Hello Python!

INTRODUCTION TO PYTHON



Hugo Bowne-Anderson
Data Scientist at DataCamp

How you will learn





datacamp

←

Course Outline

→

Daily XP 0



Exercise

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!

Instructions100 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)

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
12
13
```

↺

Run Code

Submit Answer

IPython Shell

Slides

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



Exercise

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)

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 * growth_multiplier ** 7
9
10 # Print out result
11 print(result)
12
13
```

↺

Run Code

Submit Answer


IPython Shell

Slides

In [1]:

IPython Shell

Execute Python commands





 datacamp

←

≡ Course Outline

→

Daily XP 100



Exercise

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)

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 * growth_multiplier ** 7
9
10 # Print out result
11 print(result)
12
13
```

↺

Run Code


Submit Answer

IPython Shell

Slides

In [1]:

IPython Shell





 datacamp

←

Course Outline

→

Daily XP 100



Exercise

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!

Instructions100 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)

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
```

↺

Run Code

Submit Answer

IPython Shell

Slides

In [1]:

Python Script

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

The screenshot shows the DataCamp interface for an exercise titled "Calculations with variables". The exercise instructions are on the left, and the code editor and IPython Shell are on the right.

Exercise: 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 \(-30 XP\)](#)

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 * growth_multiplier ** 7
9
10 # Print out result
11 print(result)
12
13
```

IPython Shell

In [1]:

Python Script

datacamp

Exercise

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!

Instructions100 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)

script.py

1

Run CodeSubmit Answer

Python Shell

Slides

In [1]:

Python Script

The screenshot shows the DataCamp web interface. On the left, the 'Exercise' panel is titled 'Calculations with variables'. It contains a paragraph: 'Remember how you calculated the money you ended up with after 7 years of investing \$100? You did something like this:' followed by a code block: `100 * 1.1 ** 7`. Below this, it says: '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!'. Under the 'Instructions' section, there are three bullet points: 'Create a variable `growth_multiplier` equal to `1.1`.', 'Create a variable, `result`, equal to the amount of money you saved after `7` years.', and 'Print out the value of `result`.'. A 'Take Hint (-30 XP)' button is at the bottom of the instructions. The main area on the right is a dark-themed code editor with a file named 'script.py' containing a single line '1'. Below the editor are three buttons: a circular arrow icon, 'Run Code', and 'Submit Answer'. At the bottom of the interface is an 'IPython Shell' panel with 'In [1]:' and a text input field. The top navigation bar includes the DataCamp logo, a 'Course Outline' button, and a 'Daily XP 100' indicator.

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

DataCamp Interface

datacamp

Exercise

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!

Instructions100 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)

← Course Outline →

Daily XP 100

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
```

↺

Run Code

Submit Answer

IPython Shell

Slides

▼

In [1]:

datacamp

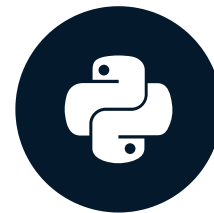
INTRODUCTION TO PYTHON

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