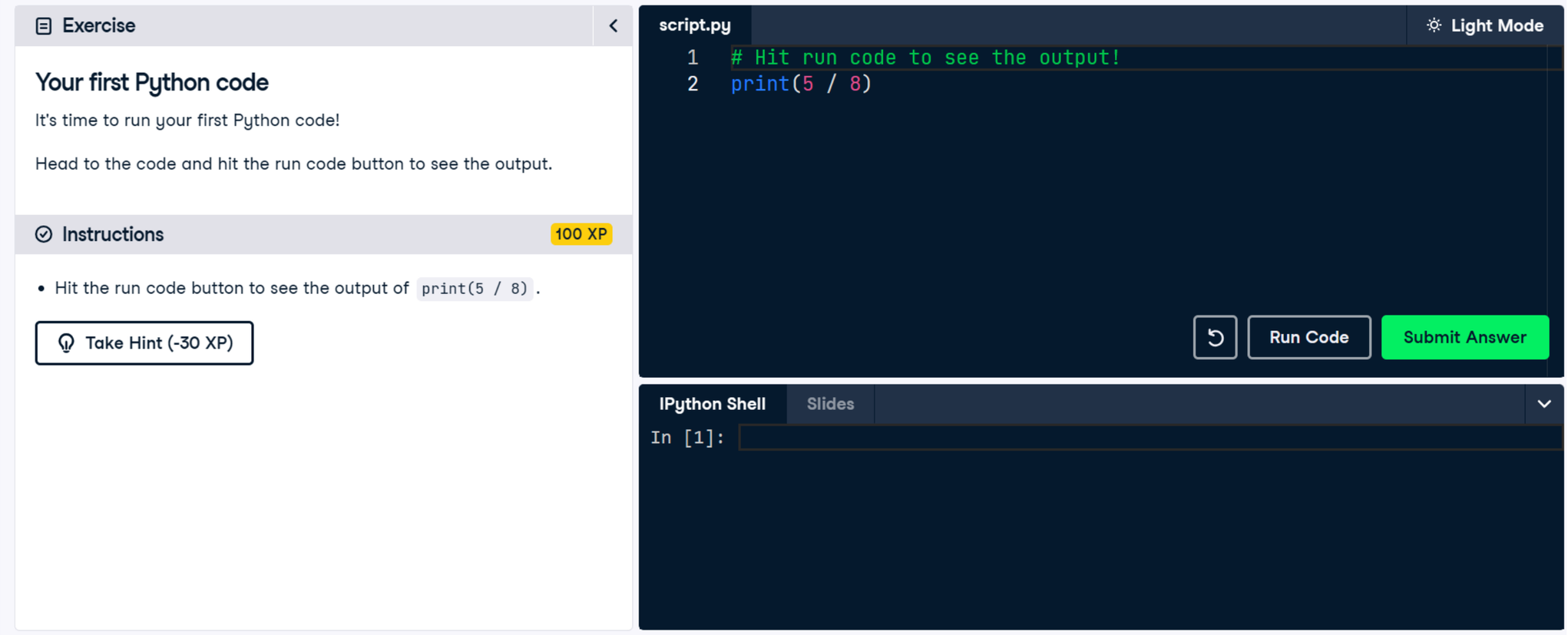
Hello Python!

INTRODUCTION TO PYTHON

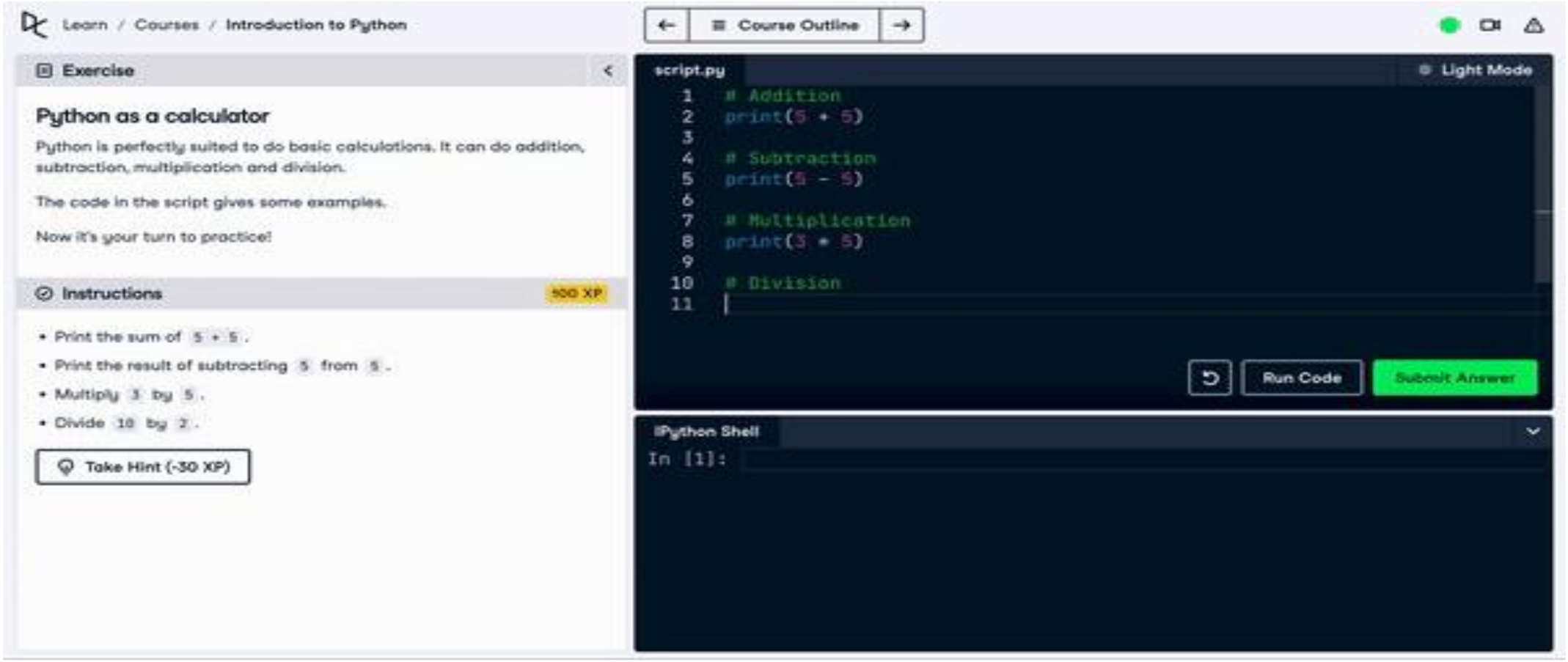
Hugo Bowne-Anderson
Data S

datacamp

Exercise


Exercise
Your first Python code
It's time to run your first Python code!
Head to the code and hit the run code button to see the output.
Instructions
100 XP
Hit the run code button to see the output of print(5 / 8).
Take Hint (-30 XP)
script.py
Light Mode
1 # Hit run code to see the output!
2 print(5 / 8)
Run Code
Submit Answer
IPython Shell
Slides
In [1]:

How you will learn

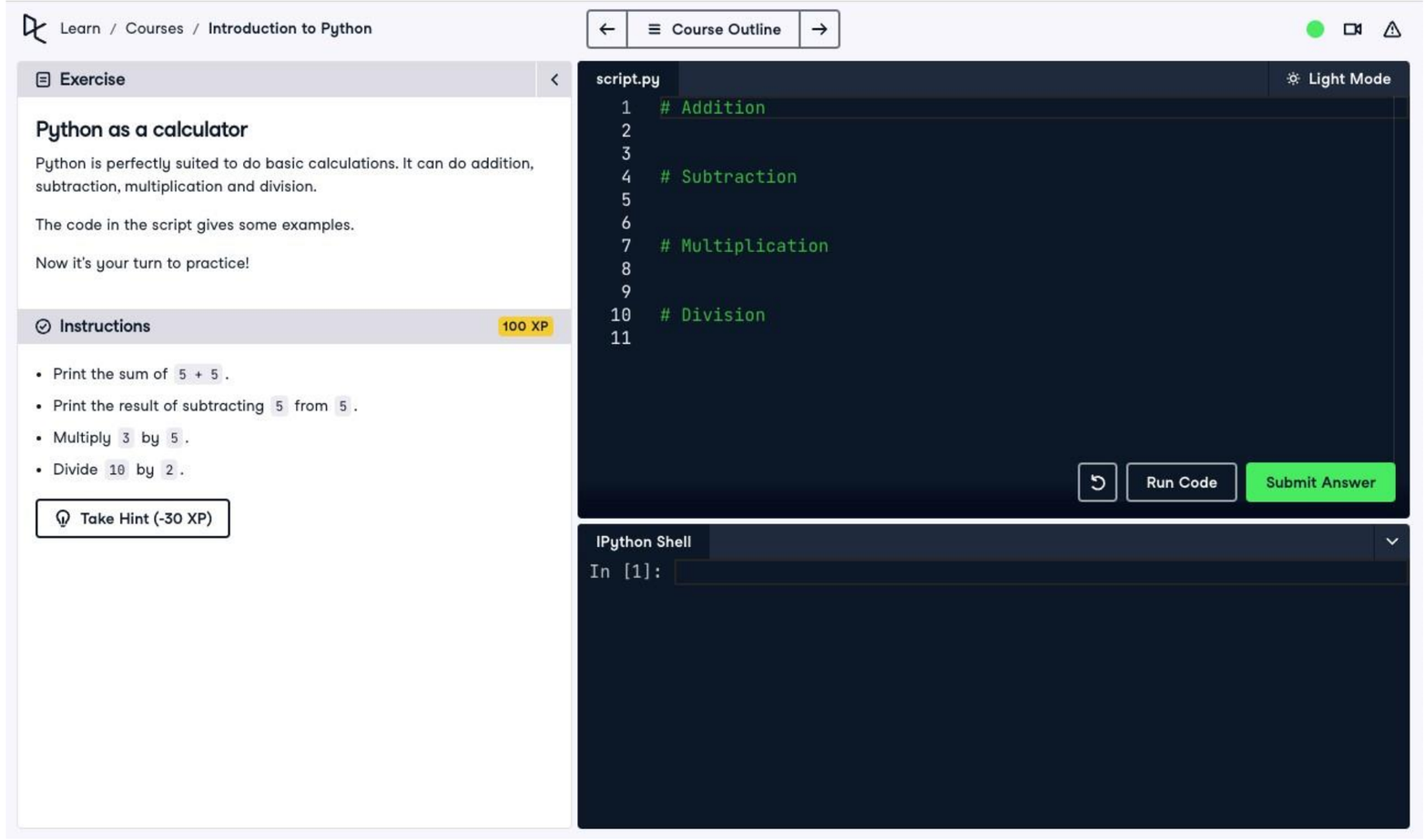


Python

- General purpose: build anything
- Open source! Free!
- Python packages, also for data science
  - Many applications and fields

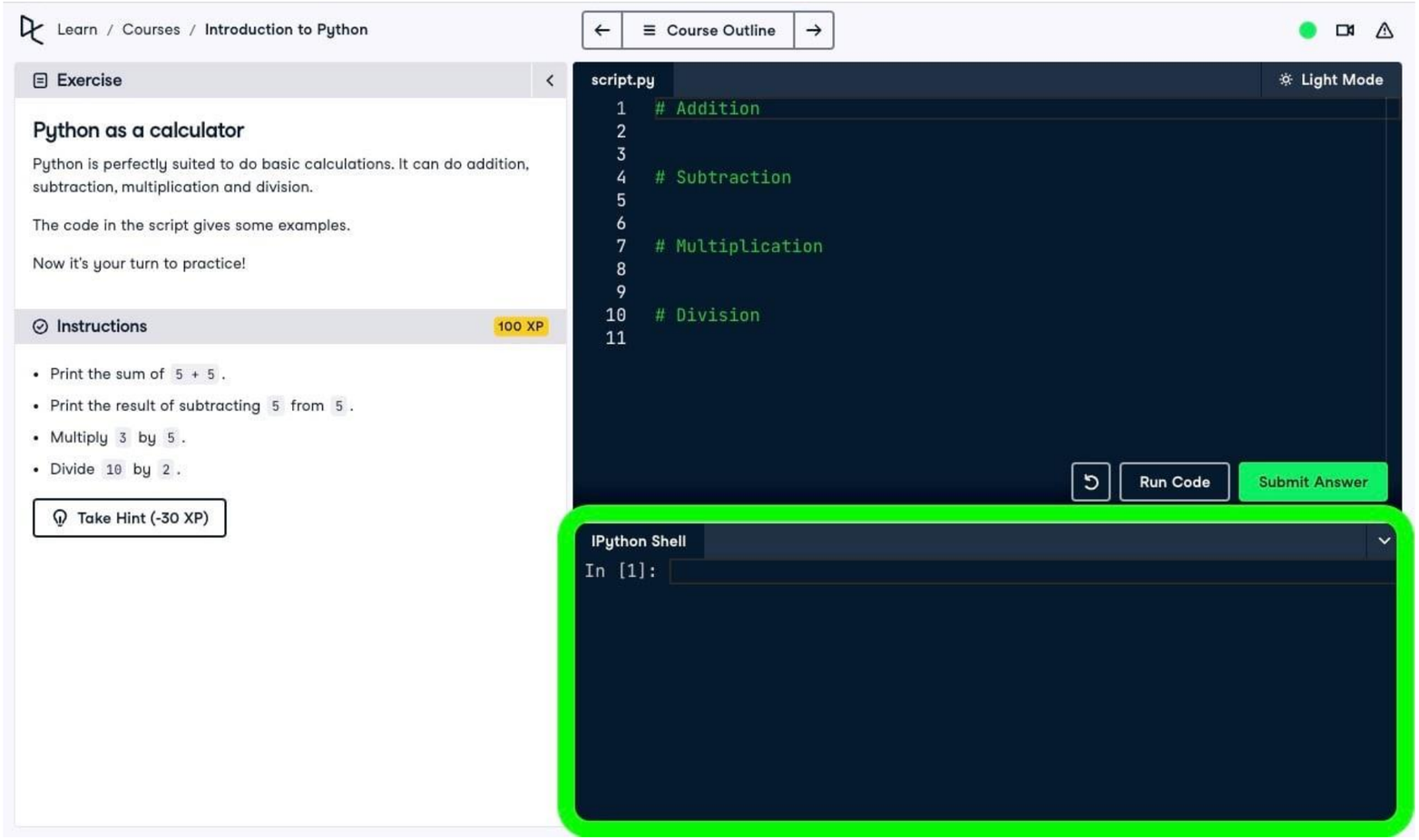
IPython Shell

Execute Python commands

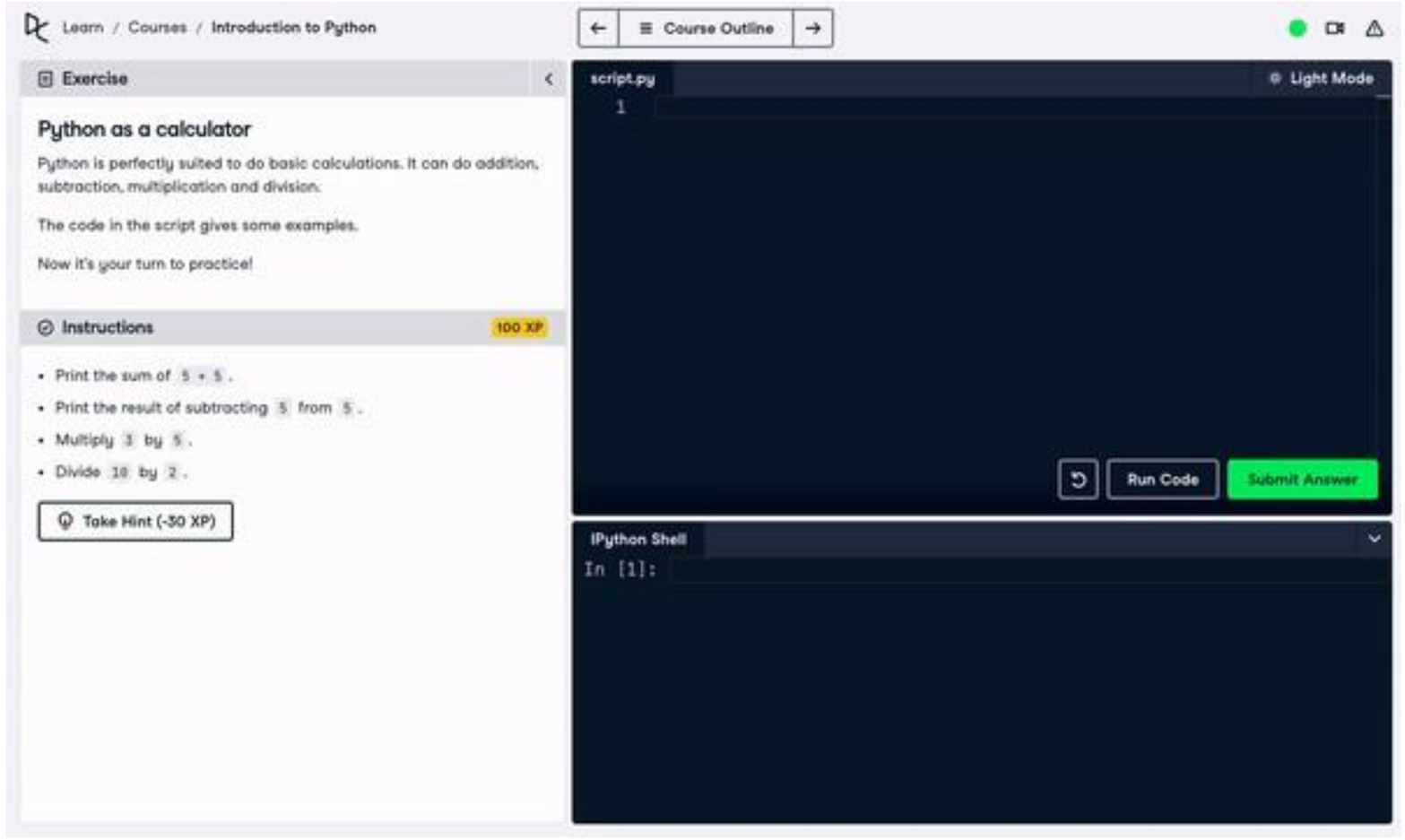


Execute Python commands

IPython Shell

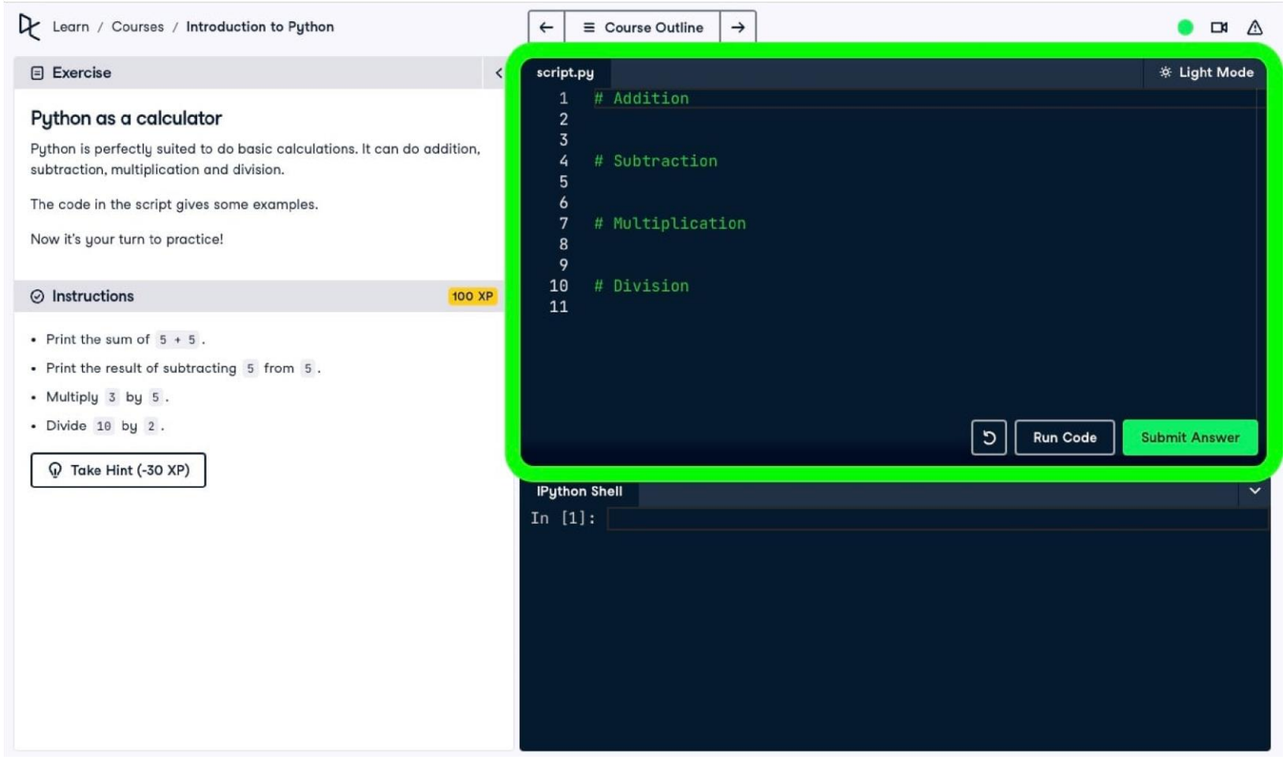


IPython Shell

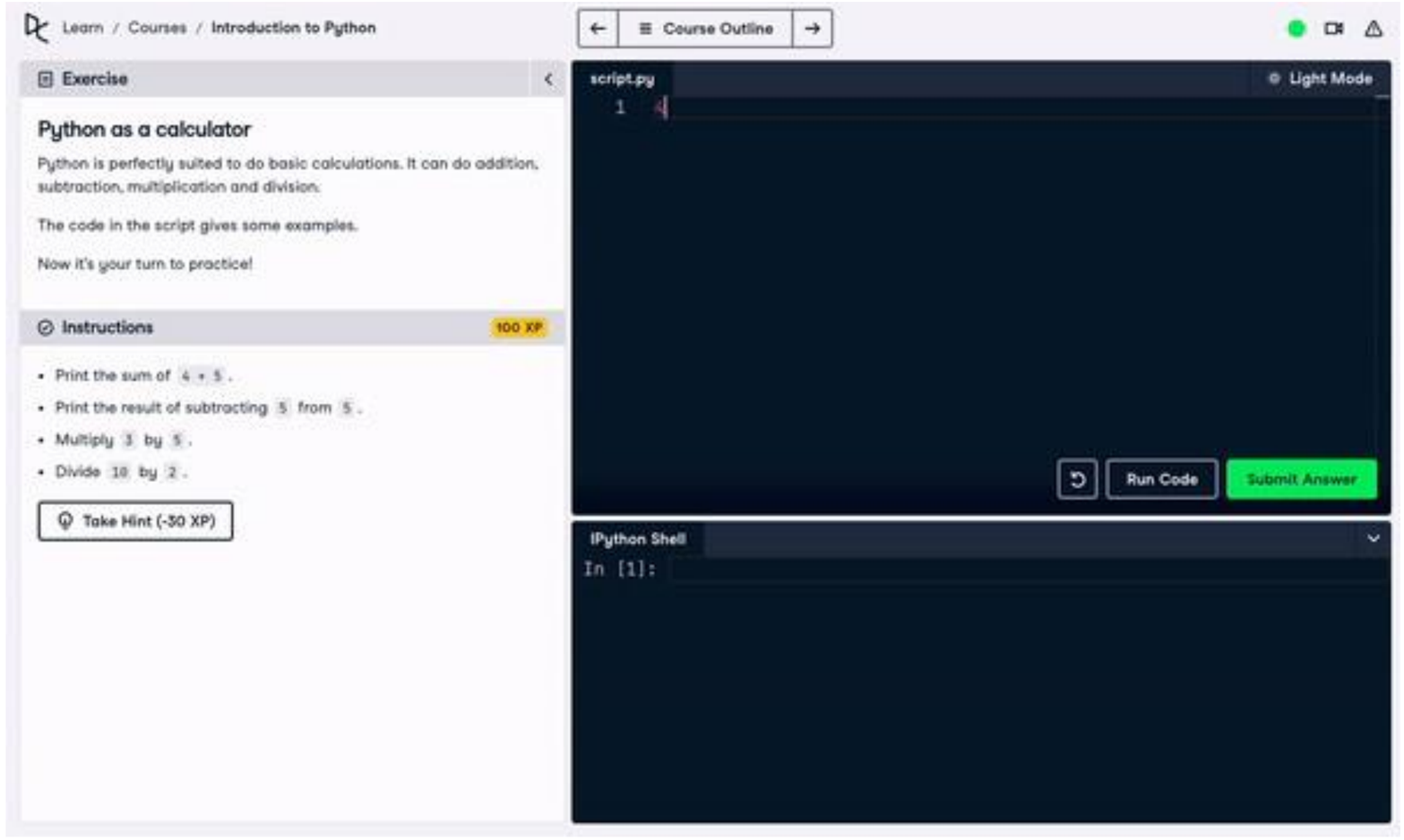


Python Script

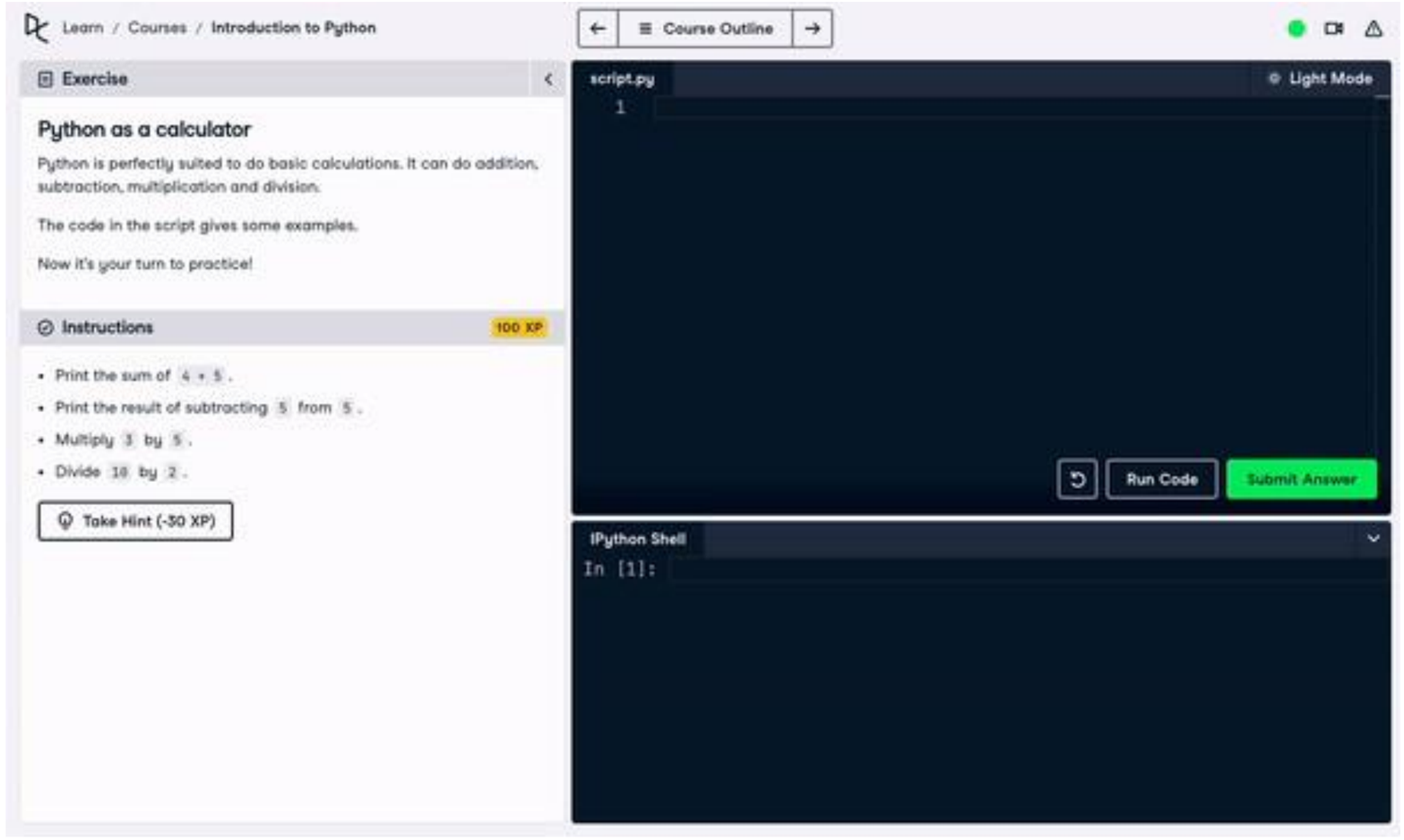
- Text files - `.py`
- List of Python commands
- Similar to typing in IPython Shell



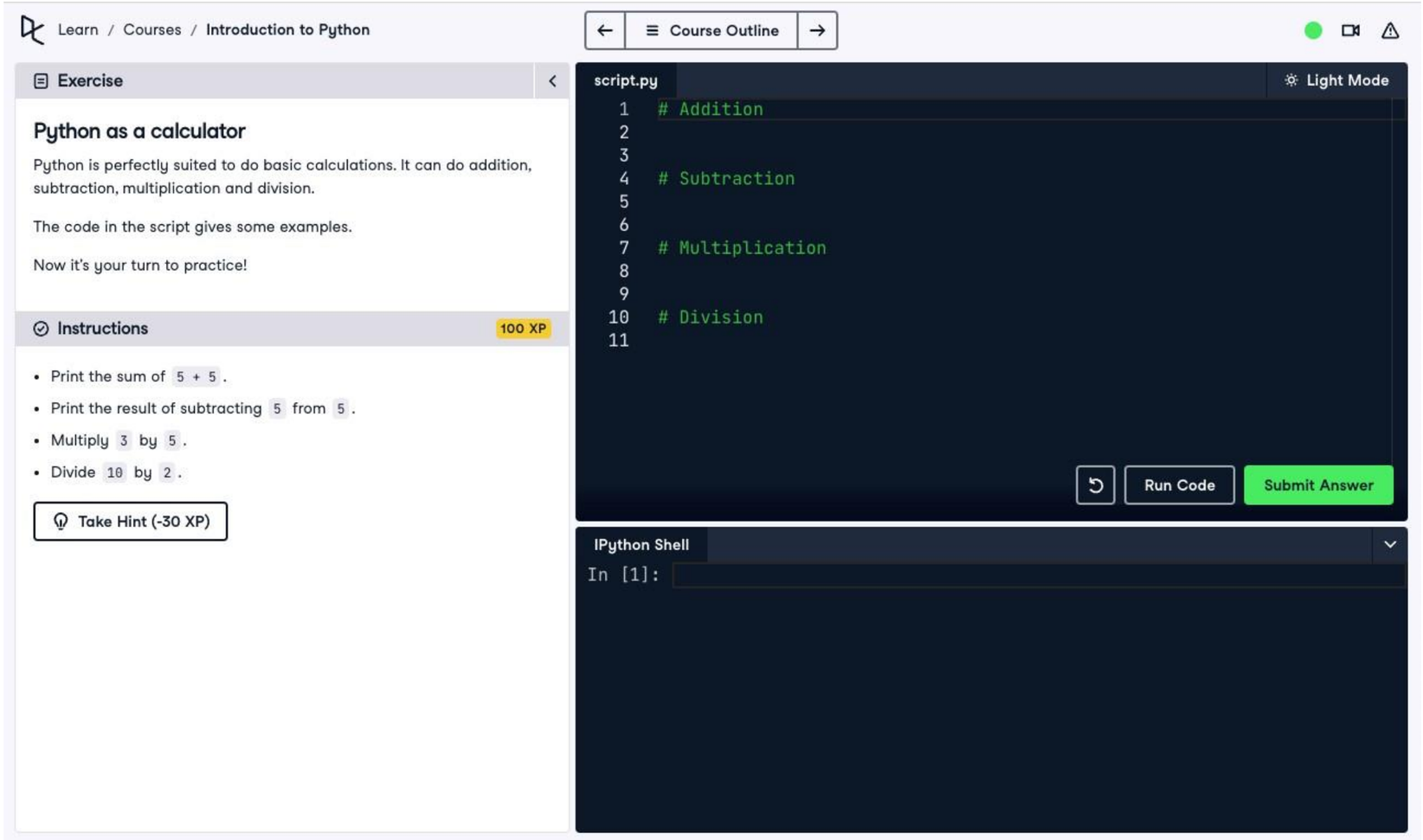
Python Script



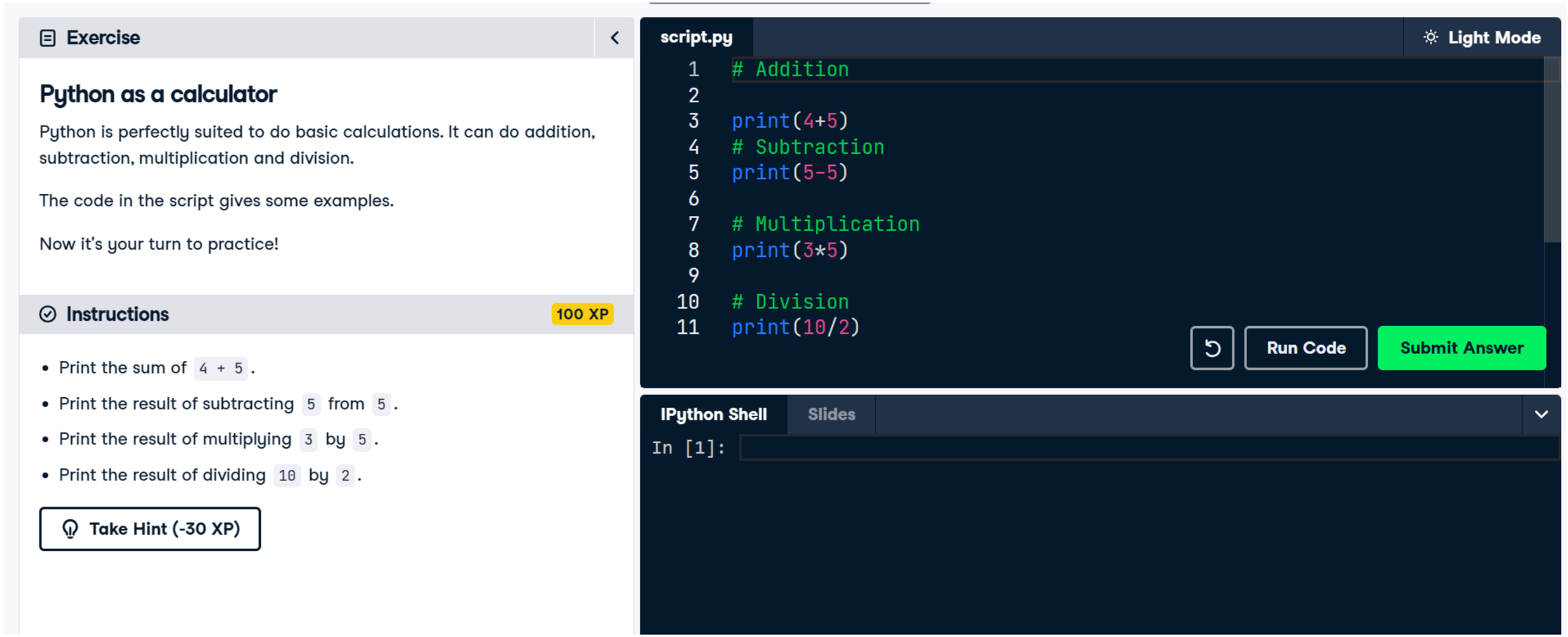
Python Script



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


Learn / Courses / Introduction to Python
Course Outline
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 5 + 5.
Print the result of subtracting 5 from 5.
Multiply 3 by 5.
Divide 10 by 2.
Take Hint (-30 XP)
script.py
Light Mode
1 # Addition
2
3
4 # Subtraction
5
6
7 # Multiplication
8
9
10 # Division
11
Run Code
Submit Answer
IPython Shell
In [1]:

Exercise



datacamp

Let's practice!

INTRODUCTION TO PYTHON

Variables and Types

INTRODUCTION TO PYTHON

Hugo Bowne-Anderson
Data Scientist at DataCamp

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
```

$$BMI = \frac{weight}{height_2}$$

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

```
21.4413
```

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

```
21.4413
```

```
21.4413
```

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

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
```

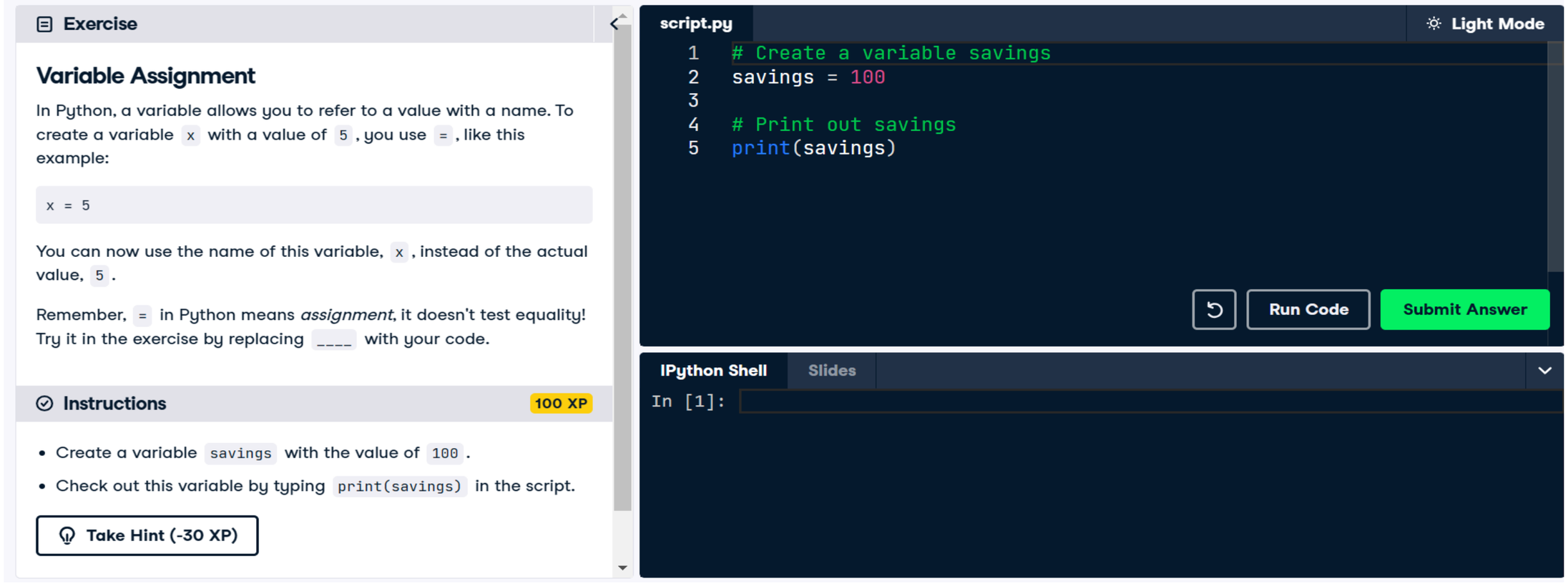
```
'ab' + 'cd'
```

```
'abcd'
```

- Different type = different behavior!

Let's practice!

INTRODUCTION TO PYTHON



datacamp

Exercise

Calculations with variables

You've now created a savings variable, so let's start saving!

Instead of calculating with the actual values, you can use variables instead.

How much money would you have saved four months from now, if you saved $10 each month?

Instructions

100 XP

- Create a variable `monthly_savings`, equal to `10` and `num_months`, equal to `4`.
- Multiply `monthly_savings` by `num_months` and assign it to `new_savings`.
- Print the value of `new_savings`.

Take Hint (-30 XP)

script.py

Light Mode

```
1  # Create the variables monthly_savings and num_months
2
3
4  monthly_savings = 10
5  num_months = 4
6  # Multiply monthly_savings and num_months
7  new_savings = monthly_savings *num_months
8
9  # Print new_savings
10 print(new_savings)
```

Run Code

Submit Answer

IPython Shell

Slides

```
In [1]:
```

datacamp

Exercise

Other variable types

In the previous exercise, you worked with the integer Python data type:

- `int`, or integer: a number without a fractional part. `savings`, with the value `100`, is an example of an integer.

Next to numerical data types, there are three other very common data types:

- `float`, or floating point: a number that has both an integer and fractional part, separated by a point. `1.1`, is an example of a float.
- `str`, or string: a type to represent text. You can use single or double quotes to build a string.
- `bool`, or boolean: a type to represent logical values. It can only be `True` or `False` (the capitalization is important!).

Instructions

100 XP

- Create a new float `half` with the value `0.5`

script.py

Light Mode

```
1  # Create a variable half
2
3  half = 0.5
4  # Create a variable intro
5  intro = "Hello! How are you?"
6
7  # Create a variable is_good
8  is_good = True
```

Run Code | Submit Answer

IPython Shell | Slides

```
In [1]:
```

datacamp

Exercise

Next to numerical data types, there are three other very common data types:

- `float`, or floating point: a number that has both an integer and fractional part, separated by a point. `1.1`, is an example of a float.
- `str`, or string: a type to represent text. You can use single or double quotes to build a string.
- `bool`, or boolean: a type to represent logical values. It can only be `True` or `False` (the capitalization is important!).

Instructions

100 XP

- Create a new float, `half`, with the value `0.5`.
- Create a new string, `intro`, with the value `"Hello! How are you?"`.
- Create a new boolean, `is_good`, with the value `True`.

Take Hint (-30 XP)

script.py

Light Mode

```
1  # Create a variable half
2
3  half = 0.5
4  # Create a variable intro
5  intro = "Hello! How are you?"
6
7  # Create a variable is_good
8  is_good = True
```

Run Code | Submit Answer

IPython Shell | Slides

```
In [1]:
```

datacamp

Exercise

Operations with other types

Variables come in different types in Python. You can see the type of a variable by using `type()`. For example, to see type of `a`, execute: `type(a)`.

Different types behave differently in Python. When you sum two strings, for example, you'll get different behavior than when you sum two integers or two booleans.

Time for you to test this out.

Instructions 1/2

50 XP

- Add `savings` and `new_savings` and assign it to `total_savings`.
  Use `type()` to print the resulting type of `total_savings`.

- Calculate the sum of `intro` and `intro` and assign the result to `doubleintro`.
  Print out `doubleintro`. Did you expect this?

script.py

Light Mode

```
savings = 100
new_savings = 40

# Calculate total_savings using savings and new_savings
total_savings = savings + new_savings
print(total_savings)

# Print the type of total_savings
print(type (total_savings))
```

Run Code

Submit Answer

IPython Shell

Slides

```
In [1]:
```