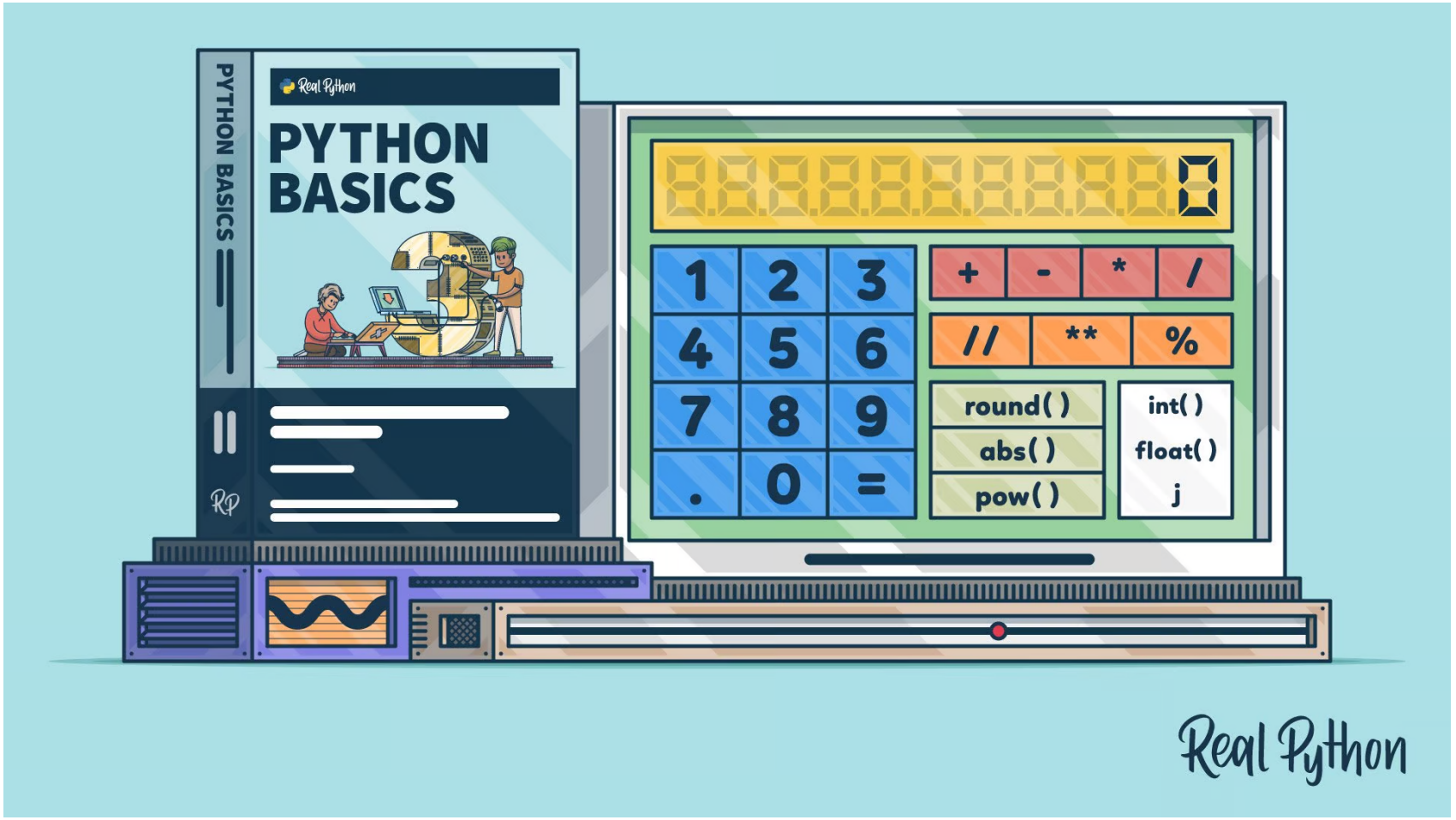
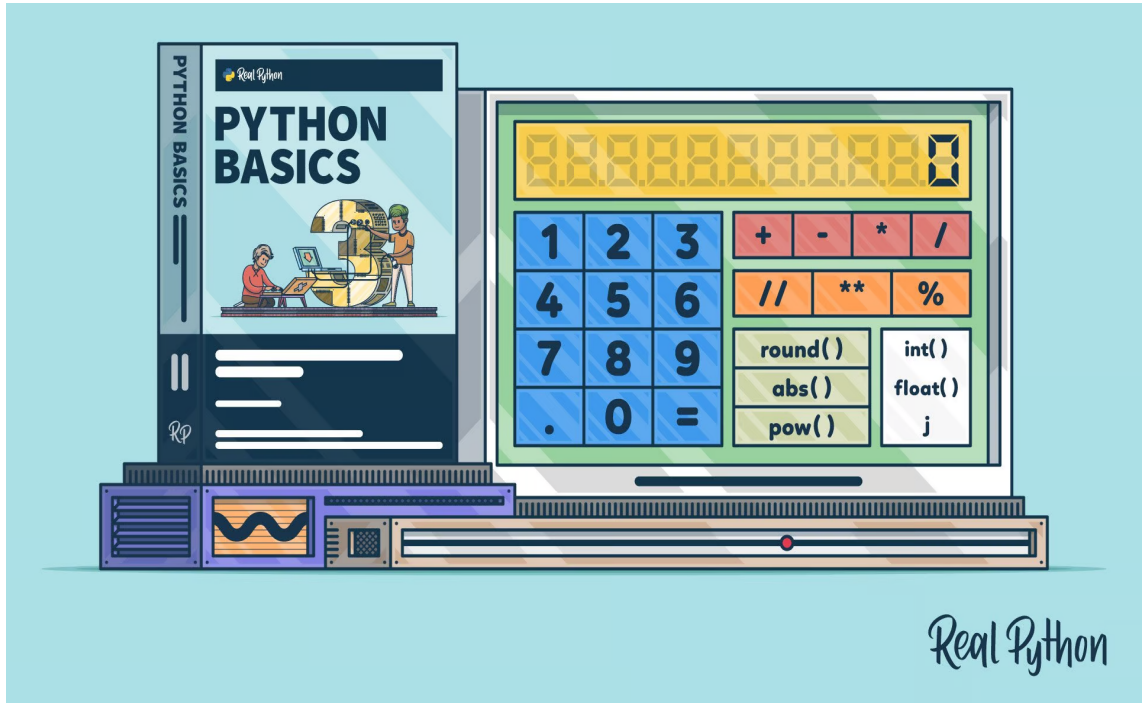
Python Basics Exercises: Numbers and Math



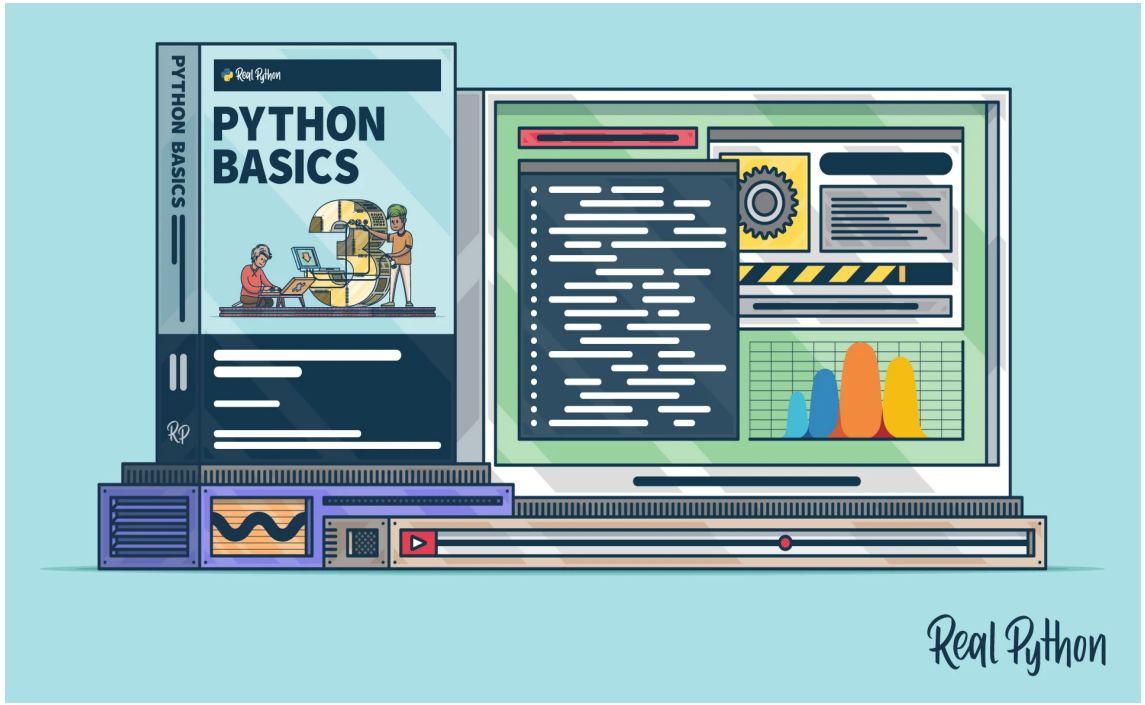
Recommended Course

Watch this video course first if you haven't already:

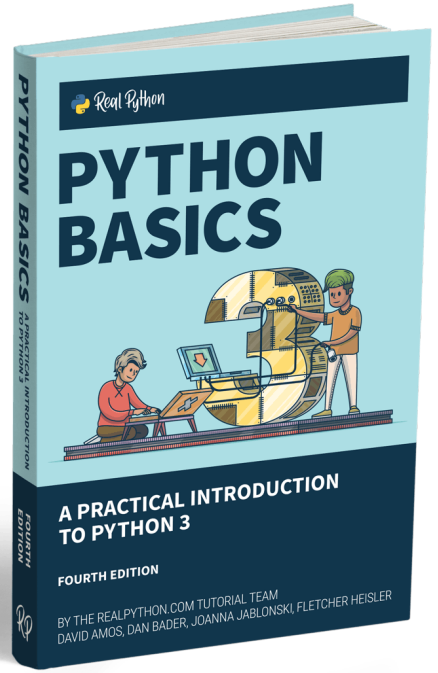


Python Basics: Numbers and Math

Python Basics



Learning Path (Video Course Series)

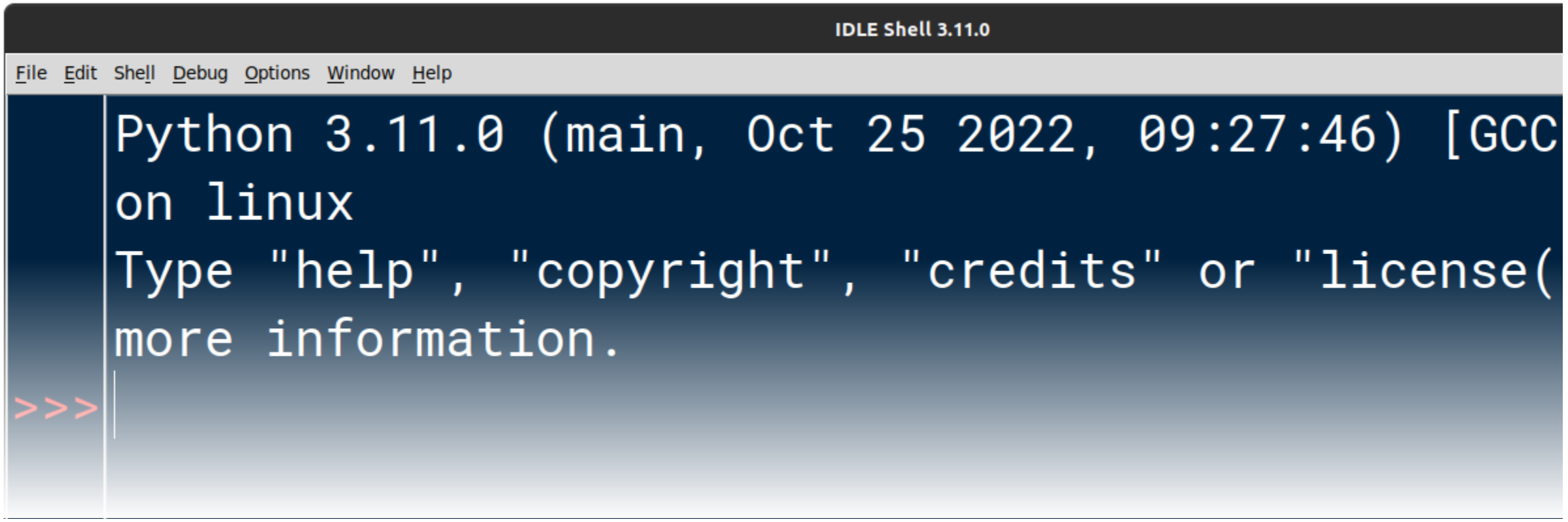


Printed Book (Paperback)

Prerequisites

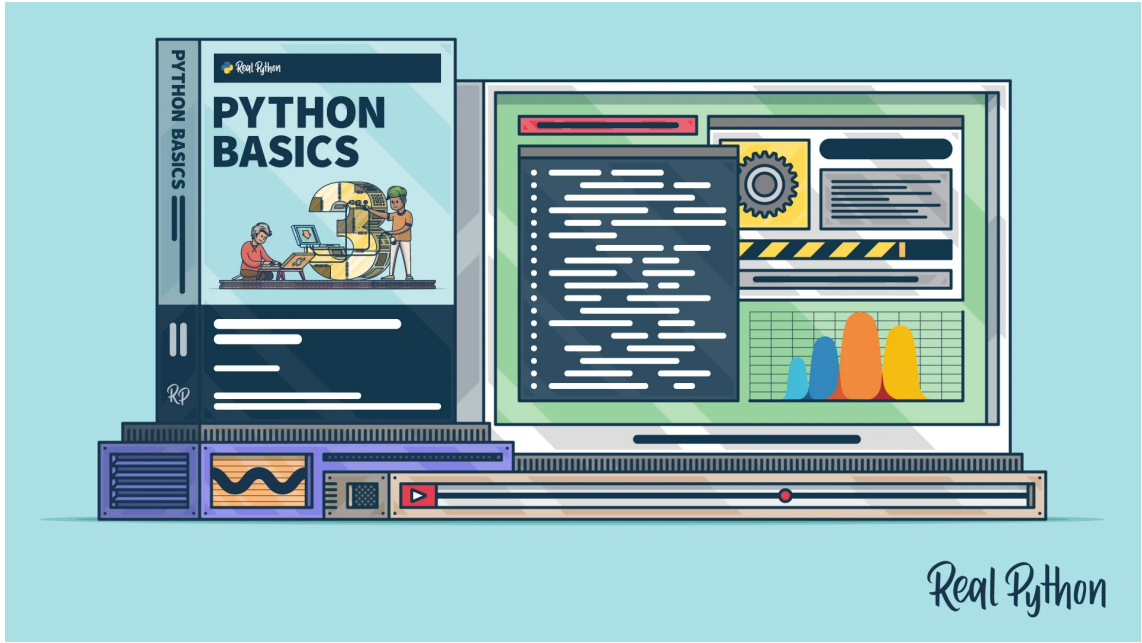
What you'll need:

IDLE: **I**ntegrated **D**evelopment and **L**earning **E**nvironment

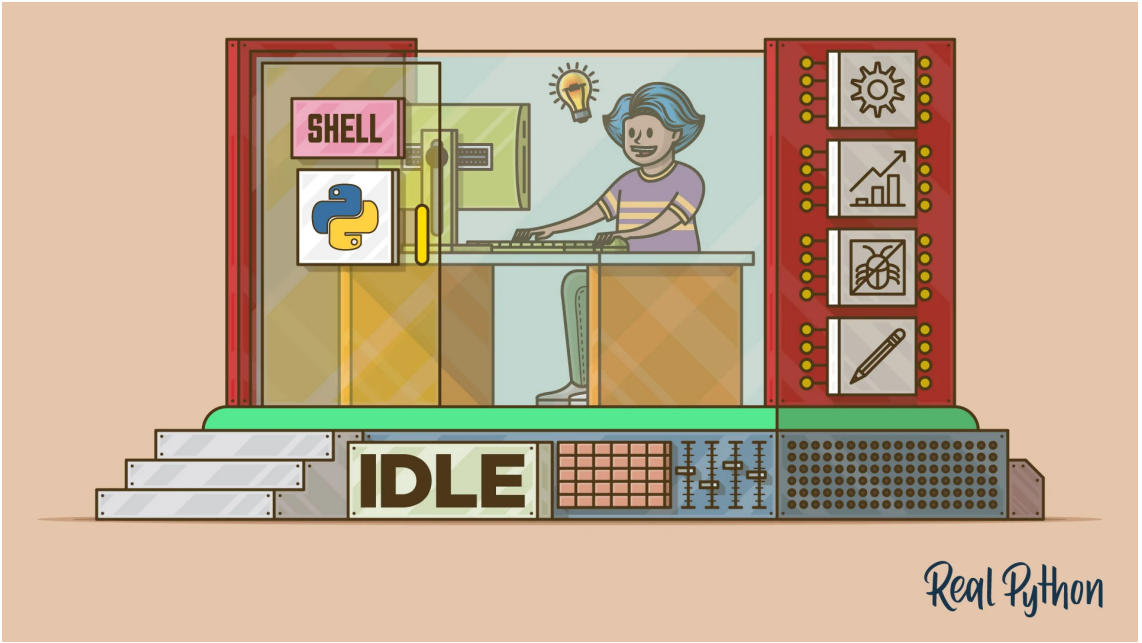


Prerequisites

What you'll need:



Python Basics: Setting Up Python



Starting With Python IDLE

Problem-Solving Process

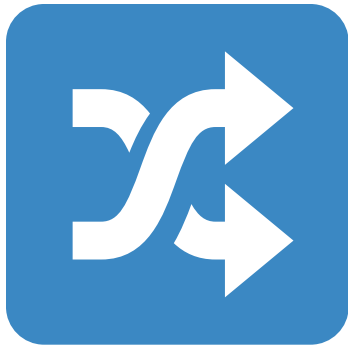
The steps for each exercise:

- 💡 **Understand**
- 📝 **Plan**
- 💻 **Implement**
- ✅ **Verify**

Problem-Solving Process

The steps for each exercise:

- 💡 Understand
- 📝 Plan
- 💻 Implement
- ✅ Verify

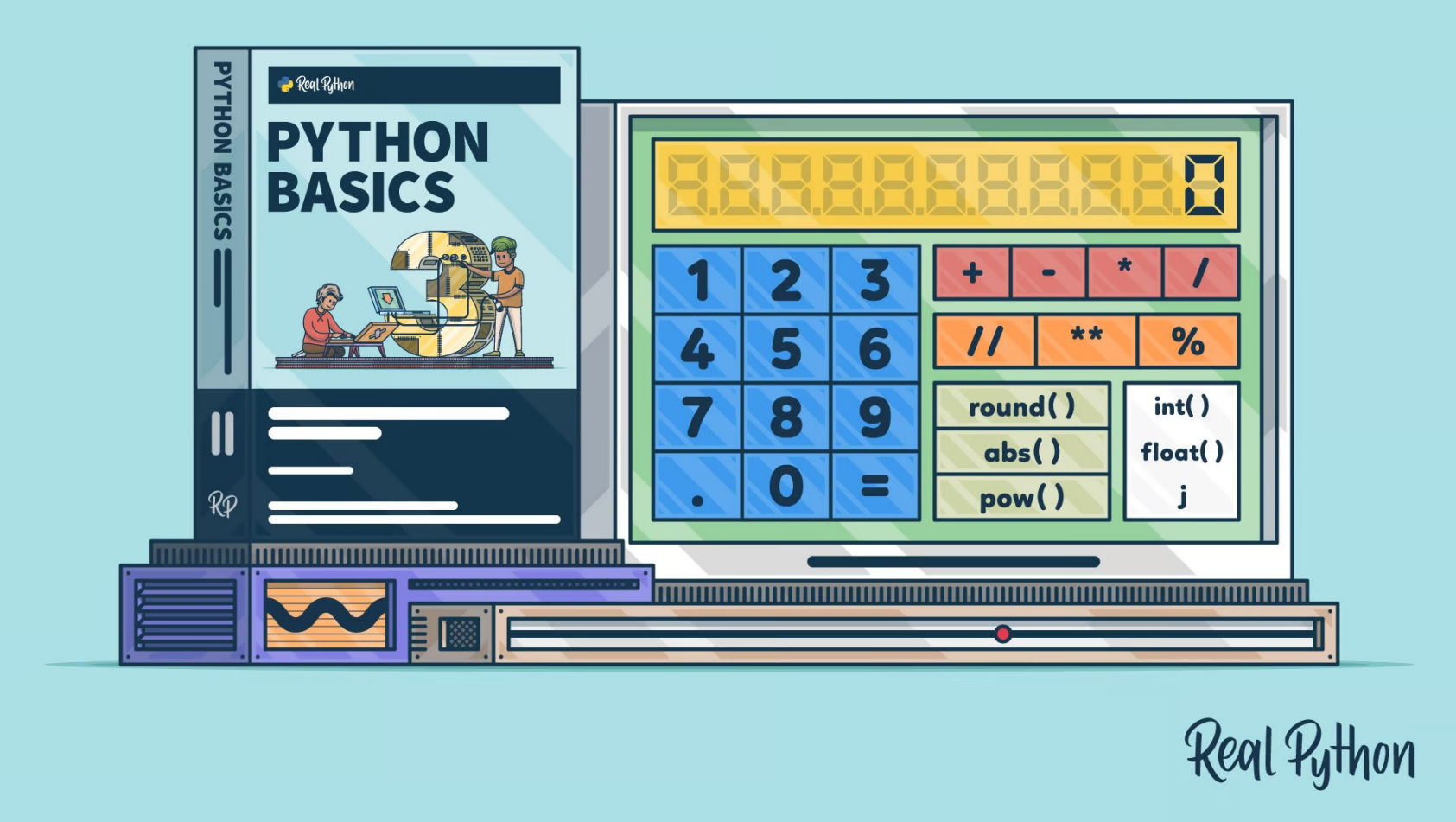


Problem-Solving Process

The steps for each exercise:

- 💡 Understand
- 📝 Plan
- 💻 Implement
- ✅ Verify

Good Luck!


PYTHON BASICS
Real Python
PYTHON BASICS
RP
0
1 2 3
4 5 6
7 8 9
. 0 =
+ - * /
// ** %
round( )
abs( )
pow( )
int( )
float( )
j
Real Python

Creating Numbers - Exercise 1

Define Integer Literals

📋 Instructions

Write a program that creates two variables, `num1` and `num2`. Both `num1` and `num2` should be assigned the integer literal `25000000`, one written with underscores and one without. Print `num1` and `num2` on two separate lines.

Creating Numbers - Exercise 1

Define Integer Literals

✨ Solution

```
>>> num1 = 25_000_000
>>> num2 = 25000000
>>> print(num1)
25000000
>>> print(num2)
25000000
```

Creating Numbers - Exercise 2

Define Floating-Point Literals

📋 Instructions

Write a program that assigns the floating-point literal `175000.0` to the variable `num` using E notation and then prints `num` in the interactive window.

Creating Numbers - Exercise 2

Define Floating-Point Literals

✨ Solution

```
>>> num = 1.75e5
>>> print(num)
175000.0
```

Creating Numbers - Exercise 3

Find the Maximum Number

📋 Instructions

In IDLE's interactive window, try to find the smallest exponent `N` for which `2e<N>`, where `<N>` is replaced with your number, returns `inf`.

Creating Numbers - Exercise 3

Find the Maximum Number

✨ Solution

Your solution may vary!

```
>>> 2e308
inf
```

Formatting Numbers - Exercise 1

Limit Decimal Places

📋 Instructions

Print the result of the calculation `3 ** .125` as a fixed-point number with three decimal places.

Formatting Numbers - Exercise 1

Limit Decimal Places

✨ Solution

```
>>> print(f"{3 ** .125:.3f}")
1.147

>>> print(format(3 ** .125, ".3f"))
1.147

>>> print("{:.3f}".format(3 ** .125))
1.147
```

Formatting Numbers - Exercise 2

Format Currency

📋 Instructions

Print the number `150000` as currency with the thousands grouped with commas. Currency should be displayed with two decimal places.

Formatting Numbers - Exercise 2

Format Currency

✨ Solution

```
>>> print(f"${150000:,.2f}")
$150,000.00
```

Formatting Numbers - Exercise 3

Show a Percentage

📋 Instructions

Print the result of `2 / 10` as a percentage with no decimal places. The output should look like `20%`.

Formatting Numbers - Exercise 3

Show a Percentage

✨ Solution

```
>>> print(f"{2 / 10:.0%}")
20%
```

Getting Numbers From the User - Exercise 1

Calculate the Exponent

📋 **Instructions** (part 1 out of 4)

Write a program called `exponent.py` that receives two numbers from the user and displays the first number raised to the power of the second number.

Getting Numbers From the User - Exercise 1

Calculate the Exponent

📋 **Instructions** (part 2 out of 4)

Here's sample run of what the program should look like, including example input from the user:

```
Enter a base: 1.2
Enter an exponent: 3
1.2 to the power of 3 = 1.7279999999999998
```

Getting Numbers From the User - Exercise 1

Calculate the Exponent

📋 **Instructions** (part 3 out of 4)

Keep the following in mind:

1. Before you can do anything with the user's input, you'll have to assign both calls to `input()` to new variables.
2. `input()` returns a string, so you'll need to convert the user's input into numbers to do arithmetic.
   (…)

Getting Numbers From the User - Exercise 1

Calculate the Exponent

📋 **Instructions** (part 4 out of 4)

Keep the following in mind:

(…)

3. You can use an f-string to print the result.
4. You can assume that the user will enter actual numbers as input.

Getting Numbers From the User - Exercise 1

Calculate the Exponent

✨ Solution

```
# exponent.py

base = float(input("Enter a base: "))
exponent = float(input("Enter an exponent: "))
result = base ** exponent
print(f"{base} to the power of {exponent} = {result}")
```

Using Math Functions - Exercise 1

Round Numbers

📋 Instructions

Write a program that asks the user to input a number and then displays that number rounded to two decimal places. When run, your program should look like this:

```
Enter a number: 5.432
5.432 rounded to 2 decimal places is 5.43
```

Using Math Functions - Exercise 1

Round Numbers

✨ Solution

```
num = float(input("Enter a number: "))
print(f"{num} rounded to 2 decimal places is {round(num, 2)}")
```

Using Math Functions - Exercise 2

Calculate the Absolute Value

📋 Instructions

Write a program that asks the user to input a number and then displays the absolute value of that number. When run, your program should look like this:

```
Enter a number: -10
The absolute value of -10 is 10.0
```

Using Math Functions - Exercise 2

Calculate the Absolute Value

✨ Solution

```
num = float(input("Enter a number: "))
print(f"The absolute value of {num} is {abs(num)}")
```

Using Math Functions - Exercise 3

Check the Numeric Type

📋 **Instructions** (part 1 out of 3)

Write a program that asks the user to input two numbers by using `input()` twice, then displays whether the difference between those two numbers is an integer.

Using Math Functions - Exercise 3

Check the Numeric Type

📋 **Instructions** (part 2 out of 3)

When run, your program should look like this:

```
Enter a number: 1.5
Enter another number: .5
The difference between 1.5 and .5 is an integer? True!
```

Using Math Functions - Exercise 3

Check the Numeric Type

📋 **Instructions** (part 3 out of 3)

If the user inputs two numbers whose difference is not integral, then the output should look like this:

```
Enter a number: 1.5
Enter another number: 1.0
The difference between 1.5 and 1.0 is an integer? False!
```

Using Math Functions - Exercise 3

Check the Numeric Type

✨ Solution

```
num1 = float(input("Enter a number: "))
num2 = float(input("Enter another number: "))
diff = num1 - num2
print(
    f"The difference between {num1} and {num2} is an integer? "
    f"{diff.is_integer()}!"
)
```

Congratulations 🎉

Summary

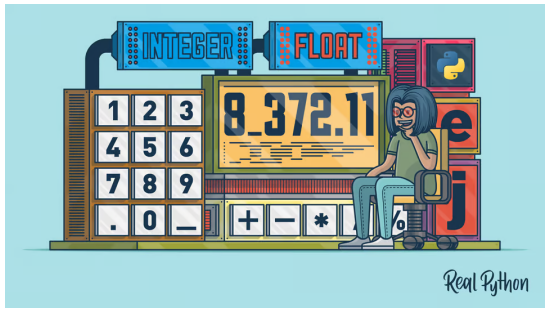
In this course, you practiced using:

- Integer and floating-point literals
- Format specification mini-language
- Numbers supplied by the user
- Math functions and number methods

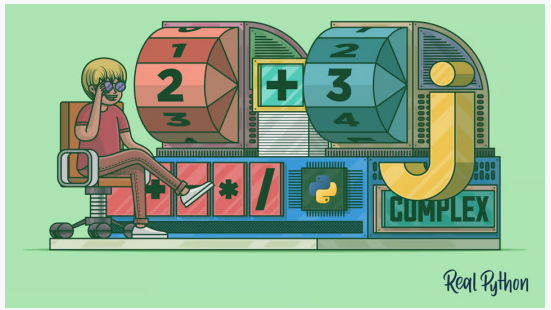
Tips

- 👓 Read the instructions carefully
- 🔪 Break exercises into smaller tasks
- 📝 Draft your idea using pen and paper
- 🐾 Take baby steps when implementing code
- 📑 Give your variables descriptive names
- 💬 Use code comments when necessary
- ✅ Test each step repeatedly to get feedback

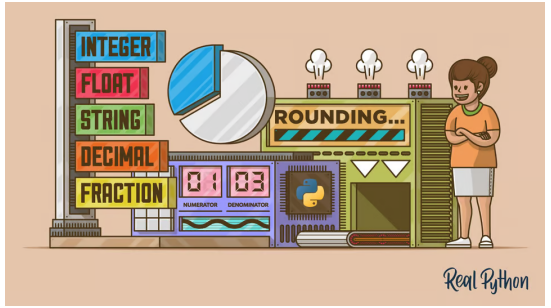
Additional Resources (1/4)



Numbers in Python

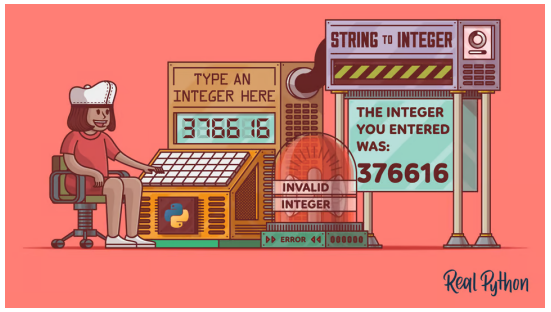


Simplify Complex Numbers With Python

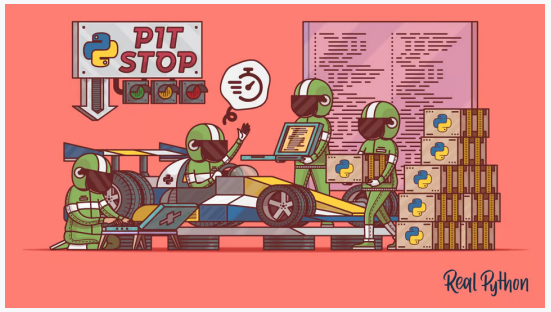


Representing Rational Numbers With Python Fractions

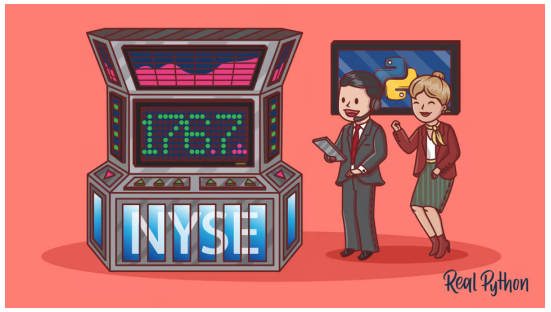
Additional Resources (2/4)



How to Read Python Input as Integers

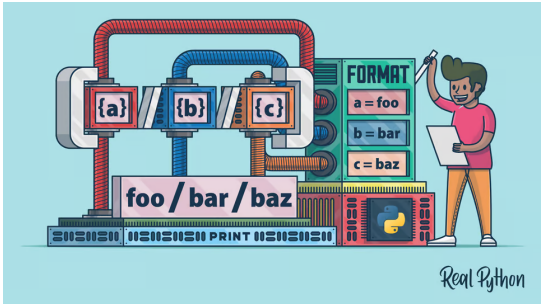


How to Convert a Python String to int

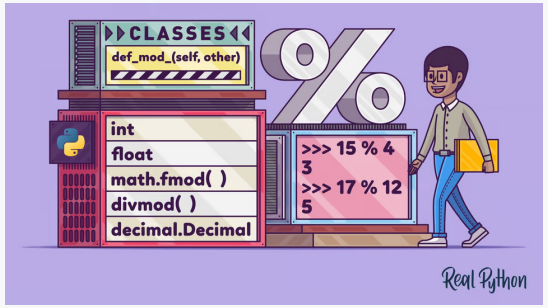


How to Round Numbers in Python

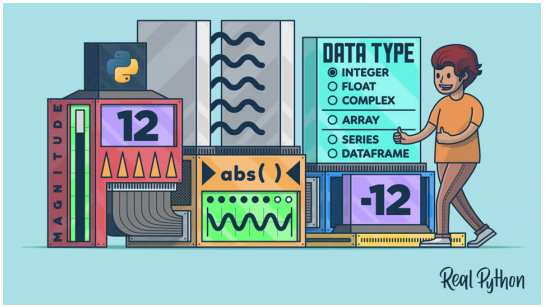
Additional Resources (3/4)



A Guide to the Newer Python String Format Techniques

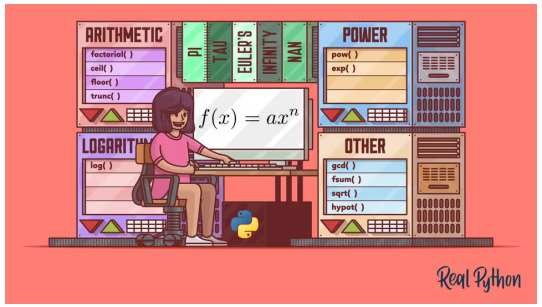


Python Modulo in Practice: How to Use the % Operator



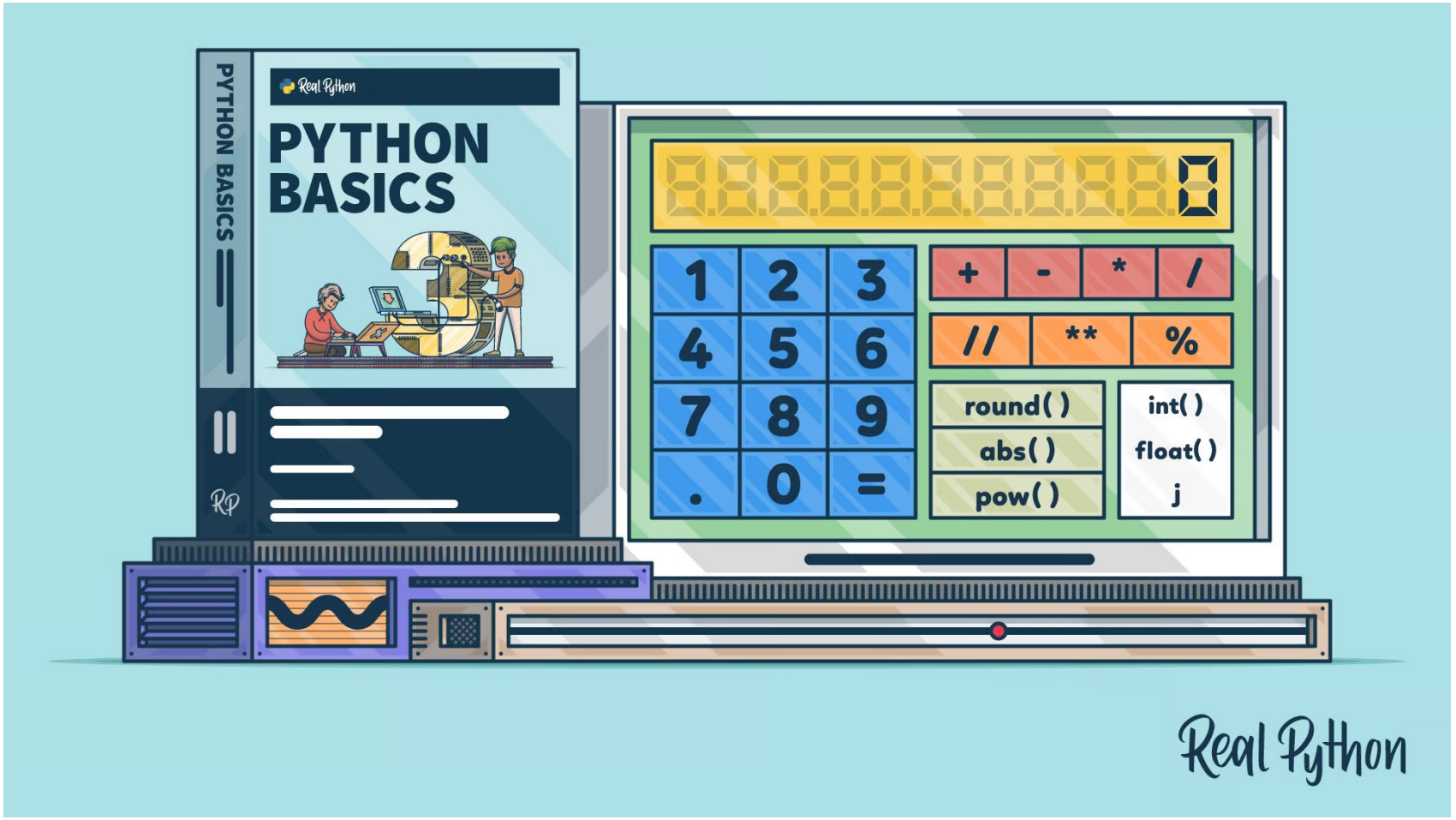
How to Find an Absolute Value in Python

Additional Resources (4/4)



The Python math Module: Everything You Need to Know

Thank You!


PYTHON BASICS
Real Python
PYTHON BASICS
RP
1 2 3
4 5 6
7 8 9
. 0 =
+ - * /
// ** %
round( )
abs( )
pow( )
int( )
float( )
j
Real Python