Writing Your First Python Code

Estimated time needed: **25** minutes

Objectives

After completing this lab you will be able to:

- Write basic code in Python
- Work with various types of data in Python
- Convert the data from one type to another
- Use expressions and variables to perform operations

When learning a new programming language, it is customary to start with an "hello world" example. As simple as it is, this one line of code will ensure that we know how to print a string in output and how to execute code within cells in a notebook.

```
# Try your first Python output

print('Hello, Python!')
```

After executing the cell above, you should see that Python prints Hello, Python!. Congratulations on running your first Python code!

```
# Check the Python Version

import sys
print(sys.version)

3.11.2 (main, May  3 2023, 04:00:05) [Clang 17.0.0
(https://github.com/llvm/llvm-project df82394e7a2d06506718cafa347b



# Practice on writing comments

print('Hello, Python!') # This line prints a string
# print('Hi')

# Print string as error message

frint("Hello, Python!")
```

The error message tells you: where the error occurred (more useful in large notebook cells or scripts), and what kind of error it was (NameError) Here, Python attempted to run the function frint, but could not determine what frint is since it's not a built-in function and it has not been previously defined by us either.

```
# Try to see built-in error message

print("Hello, Python!)
```

Python is what is called an interpreted language. Compiled languages examine your entire program at compile time, and are able to warn you about a whole class of errors prior to execution. In contrast, Python interprets your script line by line as it executes it. Python will stop executing the entire program when it encounters an error (unless the error is expected and handled by the programmer, a more advanced subject that we'll cover later on in this course).

Try to run the code in the cell below and see what happens:

```
# Print string and error to see the running order

print("This will be printed")
frint("This will cause an error")
print("This will NOT be printed")

# Write your code below. Don't forget to press Shift+Enter to execute
the cell

# Write your code below. Don't forget to press Shift+Enter to execute
the cell
```

```
# Integer

11

# Float

2.14

# String

"Hello, Python 101!"

# Type of 12

type(12)

# Type of 2.14

type(2.14)

# Type of "Hello, Python 101!"

type("Hello, Python 101!")
```

In the code cell below, use the type() function to check the object type of 12.0.

```
# Write your code below. Don't forget to press Shift+Enter to execute
the cell
```

We can verify this is the case by using, you guessed it, the type() function:

```
# Print the type of -1

type(-1)

# Print the type of 4

type(4)

# Print the type of 0

type(0)
```

Once again, can test some examples with the type() function:

```
# Print the type of 1.0

type(1.0) # Notice that 1 is an int, and 1.0 is a float

# Print the type of 0.5

type(0.5)

# Print the type of 0.56

type(0.56)

# System settings about float type

sys.float_info

# Verify that this is an integer

type(2)

# Convert 2 to a float

float(2)

# Convert integer 2 to a float and check its type

type(float(2))

float
```

```
# Casting 1.1 to integer will result in loss of information

int(1.1)

1

# Convert a string into an integer

int('1')

1

# Convert a string into an integer with error

int('1 or 2 people')

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call
last)
Cell In[2], line 3
      1 # Convert a string into an integer with error
----> 3 int('1 or 2 people')

ValueError: invalid literal for int() with base 10: '1 or 2 people'

# Convert the string "1.2" into a float

float('1.2')

# Convert an integer to a string

str(1)

# Convert a float to a string

str(1.2)

'1.2'

# Value true

True

# Value false

False

False

# Type of True

type(True)
```

```
bool

# Type of False

type(False)

# Convert True to int

int(True)

1

# Convert 1 to boolean

bool(1)

True

# Convert 0 to boolean

bool(0)

# Convert True to float

float(True)

1.0

# Write your code below. Don't forget to press Shift+Enter to execute
the cell

# Write your code below. Don't forget to press Shift+Enter to execute
the cell

# Addition operation expression

43 + 60 + 16 + 41

# Subtraction operation expression

50 - 60

# Multiplication operation expression

5 * 5
```

We can also perform division with the forward slash:

```
# Division operation expression

25 / 5
```

```
# Division operation expression

25 / 6
```

As seen in the quiz above, we can use the double slash for integer division, where the result is rounded down to the nearest integer:

```
# Integer division operation expression

25 // 5

# Integer division operation expression

25 // 6
```

Let's write an expression that calculates how many hours there are in 160 minutes:

```
# Write your code below. Don't forget to press Shift+Enter to execute
the cell
```

Python follows well accepted mathematical conventions when evaluating mathematical expressions. In the following example, Python adds 30 to the result of the multiplication (i.e., 120).

```
# Mathematical expression

30 + 2 * 60
```

And just like mathematics, expressions enclosed in parentheses have priority. So the following multiplies 32 by 60.

```
# Mathematical expression

(30 + 2) * 60

# Store value into variable

x = 43 + 60 + 16 + 41

# Print out the value in variable

x

# Use another variable to store the result of the operation between
variable and value

y = x / 60
y
```

```
# Overwrite variable with new value

x = x / 60
x

# Name the variables meaningfully

total_min = 43 + 42 + 57 # Total length of albums in minutes
total_min

# Name the variables meaningfully

total_hours = total_min / 60 # Total length of albums in hours
total_hours

# Complicate expression

total_hours = (43 + 42 + 57) / 60  # Total hours in a single 
expression
total_hours

# Write your code below. Don't forget to press Shift+Enter to execute 
the cell

# Write your code below. Don't forget to press Shift+Enter to execute 
the cell

# Write your code below. Don't forget to press Shift+Enter to execute 
the cell
```

Congratulations, you have completed your first lesson and hands-on lab in Python.

Author

Joseph Santarcangelo

Other contributors

Mavis Zhou

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2022-01-10	2.1	Malika	Removed the readme for GitShare
2020-08-26	2.0	Lavanya	Moved lab to course repo in GitLab

Date (YYYY-MM-DD)	Version	Changed By	Change Description