

Torrodjae Somerville

CTEC 298-101 Symbolic Computation Using Big Data

Dr. Bemley

Learn Python Tutorial Part 1

10/17/2025

1. “Hello World” tutorial completion screen

The screenshot shows a web browser window for learnpython.org. The URL bar shows 'learnpython.org/en/Hello_World!'. The main content area displays an exercise titled 'Exercise' with the instruction: 'Use the "print" function to print the line "Hello, World!".' Below this, a code editor window titled 'script.py' contains the following code:

```
script.py
1 print("Hello, World!")
2 name="Torrodjae Somerville"
3 print(name)
```

To the right of the code editor is an 'IPython Shell' window showing the output of running the script:

```
<script.py> output:
Hello, World!
File "script.py", line 2
    name=Torrodjae Somerville
          ^
SyntaxError: invalid syntax

<script.py> output:
Hello, World!
Torrodjae Somerville

<script.py> output:
Hello, World!
Torrodjae Somerville
```

A green banner at the bottom of the code editor says 'Great job!'. At the bottom of the page, there is a footer message: 'This site is generously supported by DataCamp. DataCamp offers online interactive Python Tutorials for Data Science. Join over a million other learners and get started learning Python for data science today!'

Full Code: `print("Hello, World!")`

`name = "Torrodjae Somerville"`

`print(name)`

2. “Variables and Types” completion screen

The screenshot shows a web browser window with a blue header bar containing various links like GoToAssist, HR, Microsoft, Sagely, TripLog, Ubiquiti, Wufoo, Zoho Creator, TJ's Technology To..., and scann. Below the header, the word "Exercise" is displayed. A large green button in the center says "Download" with a downward arrow icon, and below it, smaller text says "to use extension (Free)". To the right of this button, the text "Only 2 Steps" is followed by two bullet points: "Click "Download"" and "Add Search Safely on Chrome". Further down, there is a "SEARCH SAFELY" logo. The main content area contains two code editors: "script.py" and "solution.py", and an "IPython Shell" window. The "script.py" editor shows the following code:

```
script.py    solution.py
1 # change this code
2 mystring = "hello"
3 myfloat = 10.0
4 myint = 20
5 name = "Torrodjae Somerville"
6
7 # testing code
8 if mystring == "hello":
9     print("String: %s" % mystring)
10 if isinstance(myfloat, float) and myfloat == 10.0:
11     print("Float: %f" % myfloat)
12 if isinstance(myint, int) and myint == 20:
13     print("Integer: %d" % myint)
14     print(name)
```

The "IPython Shell" window shows the output of running the script:

```
IPython Shell
String: hello
Float: 10.000000
Integer: 20
<script.py> output:
String: hello
Float: 10.000000
Integer: 20
Torrodjae Somerville
<script.py> output:
String: hello
Float: 10.000000
Integer: 20
```

A green banner at the bottom left says "Great job!" with a close button "X". At the bottom right is a "Submit" button with a green background and white text, and a circular progress bar with a green dot.

Full Code:

```
# change this code
mystring = "hello"
myfloat = 10.0
myint = 20
name = "Torrodjae Somerville"

# testing code
if mystring == "hello":
    print("String: %s" % mystring)
if isinstance(myfloat, float) and myfloat == 10.0:
    print("Float: %f" % myfloat)
if isinstance(myint, int) and myint == 20:
    print("Integer: %d" % myint)
    print(name)
```

3. “Lists” tutorial completion screen

Exercise

In this exercise, you will need to add numbers and strings to the correct lists using the "append" list method. You must add the numbers 1,2, and 3 to the "numbers" list, and the words 'hello' and 'world' to the strings variable.

You will also have to fill in the variable second_name with the second name in the names list, using the brackets operator `[]`. Note that the index is zero-based, so if you want to access the second item in the list, its index will be 1.

The screenshot shows a Jupyter Notebook interface. On the left is the code editor pane titled "script.py" containing Python code. On the right is the "IPython Shell" pane showing the execution results.

```

script.py
1 strings = []
2 names = ["John", "Eric", "Jessica"]
3
4 # write your code here
5 second_name = names[1]
6
7 numbers.append(1)
8 numbers.append(2)
9 numbers.append(3)
10
11 strings.append("hello")
12 strings.append("world")
13
14 # this code should write out the filled arrays
15 # and the second name in the names list (Eric).
16 print(numbers)
17 print(strings)

```

IPython Shell

```

File "script.py", line 10, in <module>
    number.append(3)
NameError: name 'number' is not defined

<script.py> output:
[1, 2, 3]
['hello', 'world']
The second name on the names list is Eric

<script.py> output:
[1, 2, 3]
['hello', 'world']
Torrodjae Somerville
The second name on the names list is Eric

```

Great Job!

Solution **Submit**

Full code:

```

numbers = []
strings = []
names = ["John", "Eric", "Jessica"]

# write your code here
second_name = names[1]

numbers.append(1)
numbers.append(2)
numbers.append(3)

strings.append("hello")
strings.append("world")

```

```

# this code should write out the filled arrays and the second name in the names list (Eric).
print(numbers)
print(strings)
print("The second name on the names list is %s" % second_name)
print("Torrodjae")

```

4. “Basic Operators” tutorial completion screen

Exercise

The target of this exercise is to create two lists called `x_list` and `y_list`, which contain 10 instances of the variables `x` and `y`, respectively. You are also required to create a list called `big_list`, which contains the variables `x` and `y`, 10 times each, by concatenating the two lists you have created.

The screenshot shows a Jupyter Notebook interface. On the left, the code editor displays `script.py` with the following content:

```

3
4 # TODO: change this code
5 x_list = [x] * 10
6 y_list = [y] * 10
7 big_list = x_list + y_list
8
9 print("x_list contains %d objects" % len(x_list))
10 print("y_list contains %d objects" % len(y_list))
11 print("big_list contains %d objects" % len
     (big_list))
12
13 # testing code
14 if x_list.count(x) == 10 and y_list.count(y) ==
   10:
15     print("Almost there...")
16 if big_list.count(x) == 10 and big_list.count(y)

```

On the right, the IPython Shell shows the output of running the script:

```

<script.py> output:
x_list contains 10 objects
y_list contains 10 objects
big_list contains 20 objects
Almost there...
Great job Torrodjae!
In [1]: 

```

A green bar at the bottom says "Good work!" with a close button. Below the code editor are two buttons: "Solution" and "Submit".

This site is generously supported by DataCamp. DataCamp offers online interactive Python Tutorials for Data Science. Join over a million other learners and get started learning Python for data science today!

Full Code: `x = object()`

`y = object()`

```

# TODO: change this code
x_list = [x] * 10
y_list = [y] * 10
big_list = x_list + y_list

print("x_list contains %d objects" % len(x_list))
print("y_list contains %d objects" % len(y_list))
print("big_list contains %d objects" % len(big_list))

# testing code
if x_list.count(x) == 10 and y_list.count(y) == 10:
    print("Almost there...")
if big_list.count(x) == 10 and big_list.count(y) == 10:
    print("Great job Torrodjae!")

```

5. “String Formatting” tutorial completion screen

learnpython.org/en/String_Formatting

FAX.PLUS GoToAssist HR Microsoft Sagely TripLog Ubiquiti Wufoo Zoho Creator TJ's Technology To... scannerhowto

.<number of digits>f - Floating point numbers with a fixed amount of digits to the right of the dot.

%x/%X - Integers in hex representation (lowercase/uppercase)

Exercise

You will need to write a format string which prints out the data using the following syntax: Hello John Doe. Your current balance is \$53.44.

script.py solution.py

```
1 data = ("John", "Doe", 53.44)
2 format_string = "Hello %s %. Your current balance
is $%s."
3
4 print(format_string % data)
5 print("Torrodjae Somerville")
```

IPython Shell

```
<script.py> output:
Hello John Doe. Your current balance is $53.44 according
to Torrodjae Somerville.

<script.py> output:
Hello John Doe. Your current balance is $53.44.
Torrodjae Somerville
```

In [1]: |

Great work!

Submit

This site is generously supported by DataCamp. DataCamp offers online interactive Python Tutorials for Data Science. Join over a million other learners and get started learning Python for data science today!

Full Code: data = ("John", "Doe", 53.44)
format_string = "Hello %s %. Your current balance is \$%s."

```
print(format_string % data)
print("Torrodjae Somerville")
```

6. “Basic String Operations” tutorial completion screen

Exercise

Try to fix the code to print out the correct information by changing the string.

The screenshot shows a Jupyter Notebook interface. On the left, there are two code cells: 'script.py' and 'solution.py'. The 'solution.py' cell contains the following code:

```

1 print("Torrodjae Somerville")
2 s = "Strings are awesome!"
3 # Length should be 20
4 print("Length of s = %d" % len(s))
5
6 # First occurrence of "a" should be at index 8
7 print("The first occurrence of the letter a = %d" % s.index("a"))
8
9 # Number of a's should be 2
10 print("a occurs %d times" % s.count("a"))
11
12 # Slicing the string into bits
13 print("The first five characters are '%s'" % s[:5]) # Start to 5
14 print("The next five characters are '%s'" % s[5:10]) # 5 to 10
15 print("The thirteenth character is '%s'" % s[12]) # Just number 12
16 print("The characters with odd index are '%s'" % s[1::2]) #(0-based indexing)
17 print("The last five characters are '%s'" % s[-5:]) # 5th-from-last to end

Great work!

```

On the right, the 'IPython Shell' tab shows the output of running the code:

```

In [1]: <script.py> output:
          Torrodjae Somerville
          Length of s = 20
          The first occurrence of the letter a = 8
          a occurs 2 times
          The first five characters are 'Strin'
          The next five characters are 'gs ar'
          The thirteenth character is 'a'
          The characters with odd index are 'tig r wsm!'
          The last five characters are 'some!'
          String in uppercase: STRINGS ARE AWESOME!
          String in lowercase: strings are awesome!
          String starts with 'Str'. Good!
          String ends with 'ome!'. Good!

```

A green bar at the bottom says 'Great work!' and a 'Submit' button is visible.

This site is generously supported by [DataCamp](#). DataCamp offers online interactive [Python Tutorials](#) for Data Science. Join **over a million** other learners and get started learning Python for data science today!

Full Code:

```

print("Torrodjae Somerville")
s = "Strings are awesome!"
# Length should be 20
print("Length of s = %d" % len(s))

# First occurrence of "a" should be at index 8
print("The first occurrence of the letter a = %d" % s.index("a"))

# Number of a's should be 2
print("a occurs %d times" % s.count("a"))

# Slicing the string into bits
print("The first five characters are '%s'" % s[:5]) # Start to 5
print("The next five characters are '%s'" % s[5:10]) # 5 to 10
print("The thirteenth character is '%s'" % s[12]) # Just number 12
print("The characters with odd index are '%s'" % s[1::2]) #(0-based indexing)
print("The last five characters are '%s'" % s[-5:]) # 5th-from-last to end

# Convert everything to uppercase
print("String in uppercase: %s" % s.upper())

# Convert everything to lowercase
print("String in lowercase: %s" % s.lower())

# Check how a string starts

```

```
if s.startswith("Str"):
    print("String starts with 'Str'. Good!")

# Check how a string ends
if s.endswith("ome!"):
    print("String ends with 'ome!'. Good!")

# Split the string into three separate strings,
# each containing only a word
print("Split the words of the string: %s" % s.split(" "))
```