THINKFUL

Intro to Data Science: Python Fundamentals

Welcome to Thinkful!

We teach tech skills that lead to fulfilling, high-paying careers.

Our students learn **in-demand** industry tools through **100% online programs** as they work toward a **job-ready portfolio** with the help of an **expert mentor**.

Let's get started.

Outline

- ➔ What is Data Science?
- ➔ Programming & Python
- ➔ Data Types
- ➔ Variables
- ➔ Data Structures
- ➔ Functions

Enter Data Scientists

- Data scientists take the information collected by an organization and turn it into something valuable.
- Some specific steps in that process:
  - Data Wrangling
  - Analytics
  - Predictions

What is Programming?

A program, in simplest terms, is a set of instructions for a computer to carry out.

In Data Science, you may program:

- ➔ To collect data from the web
- ➔ Reformat or clean up your data
- ➔ Display an interesting visualization

What is Python?

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.

History of Python:

- ➔ Created by Guido Van Rossum
- ➔ Released in 1994
- ➔ Was name after Monty Python's Flying Circus

Data and Data Types

Individual Data

- ➔ Strings: “Data Science”
- ➔ Integers: 12
- ➔ Floats: 12.3459
- ➔ Boolean: True or False
- ➔ None: Undefined

Collections of Data

- ➔ Lists
- ➔ Dictionaries
- ➔ Sets
- ➔ Tuples

Variables

Rules of Variables:

- ➔ Variable names must start with a letter or underscore
- ➔ The remainder of your variable may include letters, numbers, and underscores
- ➔ Variable name cannot be a reserved word

```
# Assigning variables

first_name = 'Jane'
lastName = "Smith"
age = 34
gpa = 3.22
passingClass = True
courses = ['geology', 'math', 'English']
```

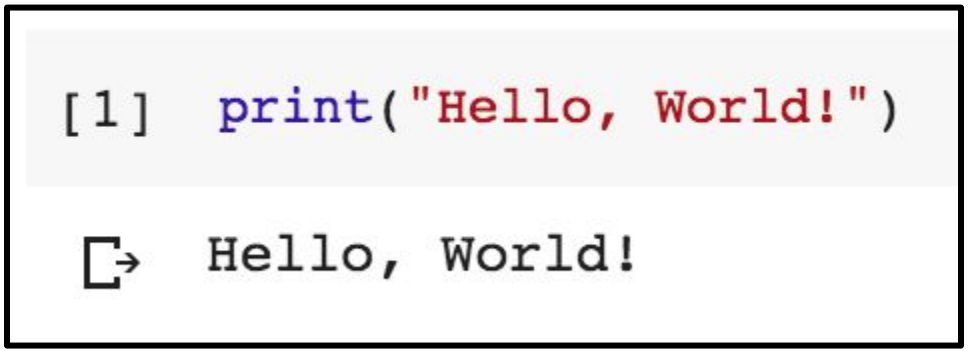
Let's Start Coding

- ➔ We'll be using a Google-hosted Python notebook called Colaboratory to demonstrate this work

Starter Code:
bit.ly/Python-Colab

- ➔ Click **File**
- ➔ Select **Save a Copy in Drive**
- ➔ This is your personal version of the notebook--let's get started!

First Line of Code



Second Line Of Code

```
[2] import this

The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Variable Basics

```
[3]  my_name = 'Monty'
     meaning_of_life = 42

     print(my_name, meaning_of_life)
```

```
Monty 42
```

Working With Strings

```
[6]  # We can "concatenate" strings or "add" them together
     full_name = my_name + ' ' + 'Python'
     print(full_name)
```

```
Monty Python
```

```
[7]  # We can make a string all lower case or upper case
     print(my_name.lower(), my_name.upper())
```

```
monty MONTY
```

```
[8]  # We can split strings on certain characters
     full_name.split(' ')
```

```
['Monty', 'Python']
```

```
[9]  # Or replace specific characters
     full_name.replace('o', '?')
```

```
'M?nty Pyth?n'
```

Working With Numbers

```
[11] # We can also work with exponents and remainders

     print('x to the power of y:', x**y)
     print('Remainder of x divided by y:', x%y)
```

```
x to the power of y: 78125
Remainder of x divided by y: 5
```

```
[10] # We can do some basic arithmetic
     x = 5
     y = 7

     print('Sum:', x+y)
     print('Difference:', x-y)
     print('Product:', x*y)
     print('Quotient:', x/y)
```

```
Sum: 12
Difference: -2
Product: 35
Quotient: 0.7142857142857143
```

```
[12] # Compound operators
     # update variable state

     original = 5
     original += 3
     print(original)
     original *= 2
     print(original)
```

```
8
16
```

Creating A List

```
[13] pizza_toppings = ['pepperoni', 'pineapple', 'extra cheese']
     print(pizza_toppings)

     ['pepperoni', 'pineapple', 'extra cheese']
```

Working With Lists

```
[15] # Getting the first element in a list
     pizza_toppings[0]
```

```
'pepperoni'
```

```
[16] # Upgrading 'pineapple' to 'mushrooms'
     pizza_toppings[1] = 'mushrooms'
     pizza_toppings
```

```
['pepperoni', 'mushrooms', 'extra cheese']
```

```
[17] # Looking at multiple items at a time
     pizza_toppings[0:2]
```

```
['pepperoni', 'mushrooms']
```

```
[18] # One cool little trick with slicing
     pizza_toppings[::-1]
```

```
['extra cheese', 'mushrooms', 'pepperoni']
```

List Challenge

Time to Practice Those List Skills:

- ➔ Make a new list called `stoplight_colors`
- ➔ Fill it with the following strings:
  - ◆ “Green”
  - ◆ “Yellow”
  - ◆ “Broken”
- ➔ Using indexing, update “Broken” to “Red”
- ➔ Print the updated list to confirm the change

Creating A Dictionary

```
[19]  groceries = {
            'apples': 4,
            'bananas': 6,
            'cookies': 3
      }
```

Working With Dictionaries

```
[21] # Checking a single value
     groceries['apples']
```

```
4
```

```
[22] # Updating a value
     groceries['apples'] += 4
     groceries
```

```
{'apples': 8, 'bananas': 6, 'cookies': 3}
```

```
[23] # Adding a new pair
     groceries['carrots'] = 0
     groceries
```

```
{'apples': 8, 'bananas': 6, 'carrots': 0, 'cookies': 3}
```

```
[24] # Checking for membership
     print('apples' in groceries)
     print('watermelons' in groceries)
```

```
True
False
```

Dictionary Challenge

Time to Practice Those Dictionary Skills:

- ➔ Take our `groceries` dictionary
- ➔ Add a new key:value pair of your choice
- ➔ Delete “apples” from the dictionary entirely (you may need to use Google!)
- ➔ Eat 2 cookies (subtract 2 cookies from the inventory)
- ➔ Print out the dictionary to confirm the changes

Function Foundations

```
[25] # Defining our function
     def hello(name):
       print("Hello " + name)

     # "Calling" or executing our function
     hello(name = 'Monty')
```

```
Hello Monty
```

Using Functions

```
[29] def far_to_cel(fahrenheit):
       celsius = int((fahrenheit - 32) / 1.8)
       return celsius

[30] # Printing resulting values

     print(far_to_cel(50))
     new_temp = 85
     print(far_to_cel(new_temp))

     10
     29
```

Using Functions
Pt. II

```
[31] # Saving the values that have been returned as new variables

     winter_temp = far_to_cel(30)
     summer_temp = far_to_cel(90)

     diff_summer_winter = summer_temp - winter_temp
     print(diff_summer_winter)
```

```
33
```

Function Challenge

Time to Practice Those Function Skills:

- ➔ Define a new function called `circumference`
- ➔ It should take one argument (radius)
- ➔ Given the radius provided, perform the necessary calculation and store it internally in a variable called `cir`
  - ◆ For reference, that equation is included in the notebook
- ➔ Return the resulting value
- ➔ Call (execute) your function with a new radius!

Common Questions

You might also be wondering

- ❏ What are the outcomes of your students for this field?
- ❏ How do I show my work to a potential employer?
- ❏ Is this course entirely online?
- ❏ What should I do from here?

Take the First Step to A New Career

Anyone who's driven to change their future and achieve a high-earning career is able to enter the world's next workforce. We'll be by your side as you build the skills you need, with personal mentorship and an active, online community of students and educators.

Expand your career opportunities by breaking into tech. Chat with an admissions rep and we'll help you find the perfect fit.

Schedule a Call