



LEARNINGFUZE

Intro to Data Science

**#1 rated Coding and Data Science Program in all of Orange County,
Los Angeles, and the Inland Empire.**

Prep Course Introduction - Python



Why Python?

Python is a scripting language; that can be used to create projects from front-end to back-end and machine learning.

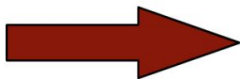
Python is open source language.

Python has an great community that you can learn from and contribute to.

Python has a great machine learning library that is very helpful in delivering data science projects



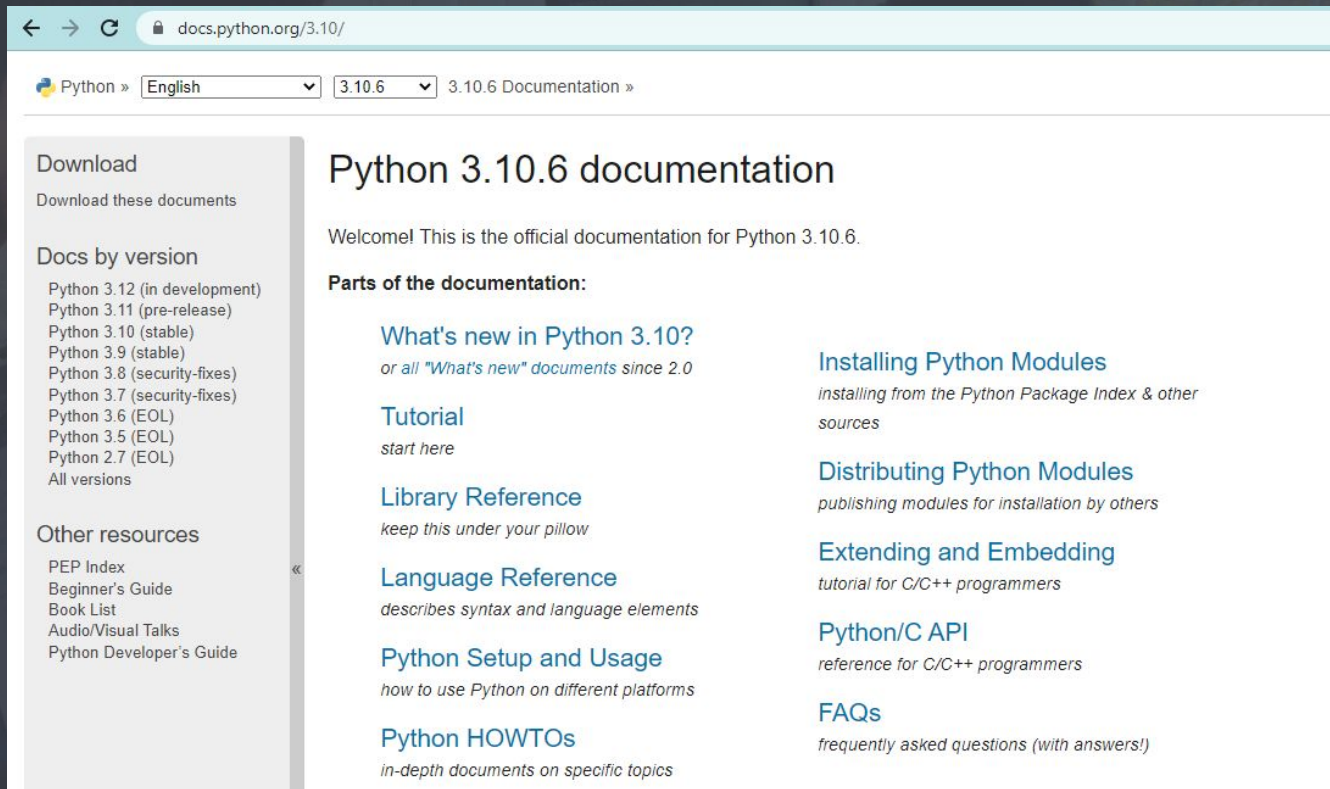
Python 2



Python 3



<https://docs.python.org/3.10/>



The screenshot shows the official Python 3.10.6 documentation page. The browser address bar displays 'docs.python.org/3.10/'. The page has a light blue header with navigation links for 'Python', language selection (currently 'English'), version selection (currently '3.10.6'), and a link to '3.10.6 Documentation'. The main content area is titled 'Python 3.10.6 documentation' and includes a welcome message. A sidebar on the left contains links for downloading documents, viewing documentation by version (from Python 3.12 down to 2.7), and other resources like the PEP Index and Beginner's Guide. The main content area lists several key sections: 'What's new in Python 3.10?', 'Tutorial', 'Library Reference', 'Language Reference', 'Python Setup and Usage', and 'Python HOWTOs'. On the right side, there are additional links for 'Installing Python Modules', 'Distributing Python Modules', 'Extending and Embedding', 'Python/C API', and 'FAQs'.

← → ↻ docs.python.org/3.10/

Python » English 3.10.6 3.10.6 Documentation »

Download

Download these documents

Docs by version

- Python 3.12 (in development)
- Python 3.11 (pre-release)
- Python 3.10 (stable)
- Python 3.9 (stable)
- Python 3.8 (security-fixes)
- Python 3.7 (security-fixes)
- Python 3.6 (EOL)
- Python 3.5 (EOL)
- Python 2.7 (EOL)
- All versions

Other resources

- PEP Index
- Beginner's Guide
- Book List
- Audio/Visual Talks
- Python Developer's Guide

Python 3.10.6 documentation

Welcome! This is the official documentation for Python 3.10.6.

Parts of the documentation:

- [What's new in Python 3.10?](#)
or all "What's new" documents since 2.0
- [Tutorial](#)
start here
- [Library Reference](#)
keep this under your pillow
- [Language Reference](#)
describes syntax and language elements
- [Python Setup and Usage](#)
how to use Python on different platforms
- [Python HOWTOs](#)
in-depth documents on specific topics

- [Installing Python Modules](#)
installing from the Python Package Index & other sources
- [Distributing Python Modules](#)
publishing modules for installation by others
- [Extending and Embedding](#)
tutorial for C/C++ programmers
- [Python/C API](#)
reference for C/C++ programmers
- [FAQs](#)
frequently asked questions (with answers!)



<https://docs.python.org/3.10/>

Read up through chapter 4 of the official
Python tutorial:

<https://docs.python.org/3.10/tutorial/index.html>



<https://docs.python.org/3.9/>

Read chapter 5 of the official Python tutorial:

<https://docs.python.org/3/tutorial/datastructures.html>



Python Control Flow

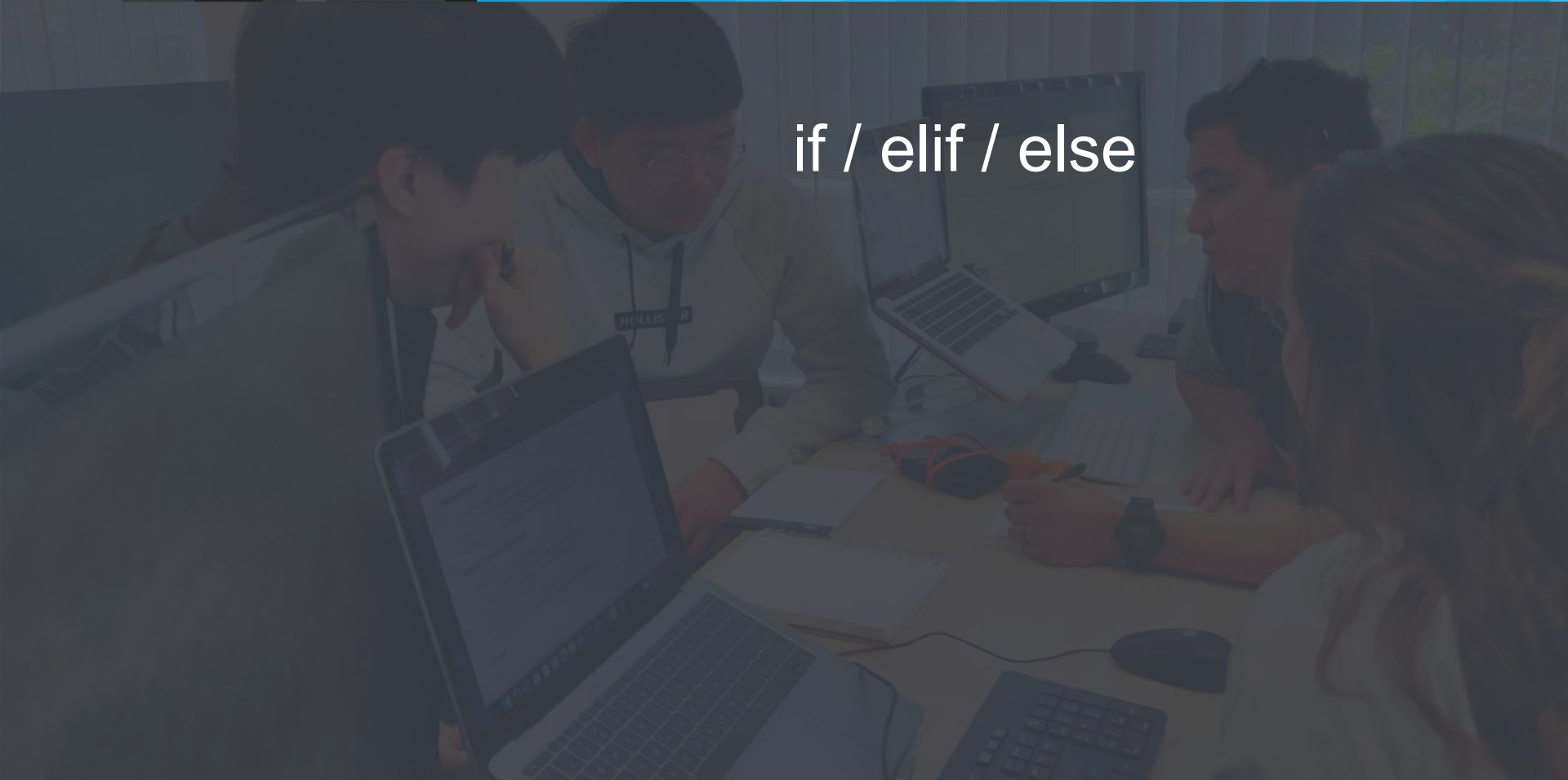
Python executes code in a top-down order. A program can take different decisions based on different situations, using control flow statements.

1. If Statements
2. Try & Except Statements
3. For Loop Statements
4. While Statements
5. Break & Continue Statements



Conditionals

if / elif / else





Ask for the age of the person and then:

- If the person is less than 2 years old, print a message that the person is a baby.
- If the person is at least 2 years old but less than 4, print a message that the person is a toddler.
- If the person is at least 4 years old but less than 13, print a message that the person is a kid.
- If the person is at least 13 years old but less than 20, print a message that the person is a teenager.
- If the person is at least 20 years old but less than 65, print a message that the person is an adult.



Conditionals

Exercise

Given an integer, perform the following conditional actions:

- if integer is odd, print Weird
- if integer is even and in the inclusive range, print Not Weird
- if integer is even and greater than , print Weird

integer range:

$1 \leq \text{integer} \leq 100$



Exercise :

Write code that prompts to input two numbers.

Add them together and print the result.



Try / except

try / except

Exercise :

Build on the previous exercise to catch the error if either input values are not a number, and print a friendly error message.



Python Data Structures

Python data structures

1. Lists
2. Tuples
3. Dictionary
4. Sets



List

- A list is a mutable data structure which holds a sequence of items in a specific order.
- Square brackets [] are used for a list.
- Add, remove, changed or search items in the list.
- The items in the list can be of any type, including other data structures. Numbers, strings, lists, tuples, dictionaries, and sets can all be included in a list.
- Duplicated items are allowed.



Exercise :

```
city = ["Vancouver", "Irvine", "Houston", "Toronto", "Newport"]
```

1. Add a city to the list (LA)
2. Use `sorted()` to print your list in alphabetical order without modifying the actual list and show that your list is still in its original order by printing it.
3. Use `sorted()` to print your list in reverse alphabetical order without changing the order of the original list.
4. Use `reverse()` to change the order of your list. Print the list to show that its order has changed.
5. Use `reverse()` to change the order of your list again.
6. Use `sort()` to change your list so it's stored in alphabetical order



for loop

Exercise :

```
cars = ['Tesla','Ferrari','McLaren']
```

Think of three electric car manufacturers, store the names of these manufacturers in a list called cars, and then use a for loop to print out the name of each manufacturer.

- Modify your program to print a model from each manufacturer
- Add a line at the end of your program stating which one is an electric car manufacturer



Exercise :

1. Ask for an input of a electric car manufacture (Audi)
2. Remove the first car manufacturer from the list
3. Remove Audi from the list by name
4. Print each manufacturer along with its location within list



Tuple

- A tuple is an immutable data structure similar to a list, but with less functionality.
- Parenthesis () are used for a tuple.
- A tuple can be represented by a series of items
- A single element tuple has a trailing , (4,)



Exercise :

```
buffet = ('Chicken','Beef','Shrimp','Fries','Salad')
```

1. Use a for loop to print each food the restaurant offers.
2. Try to modify one of the items, and make sure that Python rejects the change.
3. The restaurant changes its menu, replacing two of the items with different foods. Add a block of code that rewrites the tuple, and then use a for loop to print each of the items on the revised menu.



Looping

Combines the elements of two lists if they are not equal:

$x = [1, 2, 3]$

$y = [3, 1, 4]$



Looping

while Loop





Exercise :

Write a loop that prompts the user to enter a series of pizza toppings until they enter a 'quit' value. As they enter each topping, print a message saying you'll add that topping to their pizza.

- Use a break statement to exit the loop when the user enters a 'quit' value.
- Maximum of 4 toppings allowed



Dictionary

- A dictionary is a way to store key / value pairs. It is a collection which is unordered, changeable and indexed.
- Dictionaries are commonly used to store multiple details about a piece of data.
- Dictionary notation uses curly brackets, the key-value pairs are separated by a colon, and the pairs are separated by commas.



Dictionary

Dictionary

```
{'fname': 'Zia', 'lname': 'Khan', 'city': 'Irvine',}
```



Exercise :

Use a dictionary to store information about a person you know. Store their first name, last name, and the city in which they live. You should have keys such as `first_name`, `last_name`, and `city`. Print each piece of information stored in your dictionary.

- Loop through all the Keys in a Dictionary
- Add a cell number to the dictionary



Exercise :

Make three new dictionaries representing different people, and store all three dictionaries in a list called people. Loop through your list of people. As you loop through the list, print everything you know about each person.



Exercise :

Write a loop that prompts the user to enter a series of pizza toppings until they enter a 'quit' value. As they enter each topping, print a message saying you'll add that topping to their pizza.

- Use a break statement to exit the loop when the user enters a 'quit' value.
- Maximum of 4 toppings allowed



- Sets are unordered collections of simple objects and are used when the existence of an object in a collection is more important than the order or how many times it occurs.
- Using sets you can test for membership,
 - Intersection,
 - Union
 - Difference
- Sets do not allow duplicates



Set:

A set is an unordered collection with no duplicate elements.

Exercise:

```
basket = ['apple', 'orange', 'apple', 'pear', 'orange', 'banana']
```



Python statements

Assignment

print

if/elif/else

for/else

while/else

pass

```
a, b = 'good', 'bad'
```

```
print('Python')
```

```
if "python" in text:  
    print(text)
```

```
for x in mylist:  
    print(x)
```

```
while X > Y:  
    print('hello')
```

```
while True:  
    pass
```



Python statements

break

continue

def

return

yield

global

```
while True:
```

```
    if exittest(): break
```

```
while True:
```

```
    if skiptest(): continue
```

```
def f(a, b, c=1, *d):  
    print(a+b+c+d[0])
```

```
def f(a, b, c=1, *d):  
    return a+b+c+d[0]
```

```
def gen(n):  
    for i in n: yield i*2
```

```
def function():  
    global x
```



Python statements

import

from

class

try/except

With

del

```
import sys
```

```
from sys import stdin
```

```
class Subclass(Superclass):
```

```
    x = []
```

```
    def method(self): pass
```

```
try:
```

```
    action()
```

```
except:
```

```
    print('action error')
```

```
with open('data') as myfile:
```

```
    process(myfile)
```

```
del variable
```