**Table of Contents**

[Jupyter Notebook 2](#__RefHeading___Toc126_3575276354)

[Jupyter Notebook and JupyterLab < 3.0 2](#__RefHeading___Toc255_4228277728)

[JupyterLab >= 3, Jupyter Notebook Classic, and RetroLab 2](#__RefHeading___Toc257_4228277728)

[Python Collections (Arrays) 4](#__RefHeading___Toc130_3575276354)

[Set 5](#__RefHeading___Toc132_3575276354)

[Example 5](#__RefHeading___Toc134_3575276354)

[Dictionary 5](#__RefHeading___Toc136_3575276354)

[Example 5](#__RefHeading___Toc138_3575276354)

# Jupyter Notebook

jupyter notebook --notebook-dir=”Z:\Documents\SoftwareEngineering\Python\General Assembly”

### Jupyter Notebook and JupyterLab < 3.0

For old Jupyter Notebook interface installed with notebook package and run as jupyter notebook (see the next section for the identical interface installed with nbclassic and run with jupyter nbclassic, and for JupyterLab):

1. Open cmd (or Anaconda Prompt) and run jupyter notebook --generate-config.
2. This writes a file to C:\Users\username\.jupyter\jupyter\_notebook\_config.py.

C:\Users\akari\.jupyter

1. Browse to the file location and open it in an Editor
2. Search for the following line in the file: #c.NotebookApp.notebook\_dir = ''
3. Replace by c.NotebookApp.notebook\_dir = '/the/path/to/home/folder/'

Make sure you use forward slashes in your path and use /home/user/ instead of ~/ for your home directory, backslashes could be used if placed in double quotes even if folder name contains spaces as such : "D:\yourUserName\Any Folder\More Folders\"

1. Remove the # at the beginning of the line to allow the line to execute

### JupyterLab >= 3, Jupyter Notebook Classic, and RetroLab

For recent nbclassic and JupyterLab >= 3 use c.ServerApp.root\_dir instead of c.NotebookApp.notebook\_dir (and jupyter server --generate-config instead of jupyter notebook --generate-config).

For help type ‘help’ i.e. for definition of ‘print’ function type:

**help(print)**

**?print**

Getting help from Jupyter and Python to remember what stuff does:

* shift+tab in the brackets of a function or argument will bring up the docstring, or short instructions on how to use it
* help(function\_or\_method) will print out the same docstring
* dir(variable) will remind you what methods are available (for a variable, depending on its type or data structure)

Other tricks

* tab whilst typing will try to autocomplete things
* command+click or alt+click will enable **multi-line editing** - which is so amazingly cool and a must have feature for a text editor once you know it exists.

%pip install geocode

## Python Collections (Arrays)

There are four collection data types in the Python programming language:

**List is a collection which is ordered and changeable. Allows duplicate members. Lists are used to store multiple items in a single variable.**

**Create a List:**

thislist = ["apple", "banana", "cherry"]  
print(thislist)

* List items are ordered, changeable, and allow duplicate values.
* List items are indexed, the first item has index [0], the second item has index [1] etc.
* Ordered: When we say that lists are ordered, it means that the items have a defined order, and that order will not change.
* If you add new items to a list, the new items will be placed at the end of the list.
* Note: There are some list methods that will change the order, but in general: the order of the items will not change.
* Changeable: The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.
* Allow Duplicates: Since lists are indexed, lists can have items with the same value:

Tuples

[**Tuple**](https://www.w3schools.com/python/python_tuples.asp) is a collection which is ordered and unchangeable. Allows duplicate members.

Tuples are used to store multiple items in a single variable.

A tuple is a collection which is ordered and unchangeable. Tuples are written with round brackets.

**Create a Tuple:**

thistuple = ("apple", "banana", "cherry")

print(thistuple)

uple Items

* Tuple items are ordered, unchangeable, and allow duplicate values.
* Tuple items are indexed, the first item has index [0], the second item has index [1] etc.
* Ordered: When we say that tuples are ordered, it means that the items have a defined order, and that order will not change.
* Unchangeable: Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.
* Allow Duplicates: Since tuples are indexed, they can have items with the same value:

## Set

[**Set**](https://www.w3schools.com/python/python_sets.asp) is a collection which is unordered, unchangeable\*, and unindexed. No duplicate members.

Sets are used to store multiple items in a single variable. A set is a collection which is unordered, unchangeable\*, and unindexed.

\* Note: Set items are unchangeable, but you can remove items and add new items.

Sets are written with curly brackets.

### Example

**Create a Set:**

thisset = {"apple", "banana", "cherry"}  
print(thisset)

* Set Items: Set items are unordered, unchangeable, and do not allow duplicate values.
* Unordered: Unordered means that the items in a set do not have a defined order.
* Set items can appear in a different order every time you use them, and cannot be referred to by index or key.
* Unchangeable: Set items are unchangeable, meaning that we cannot change the items after the set has been created.
* Once a set is created, you cannot change its items, but you can remove items and add new items.
* Duplicates Not Allowed: Sets cannot have two items with the same value.

No duplicate members.

## Dictionary

[**Dictionary**](https://www.w3schools.com/python/python_dictionaries.asp) is a collection which is ordered\*\* and changeable.

Dictionaries are used to store data values in key:value pairs. A dictionary is a collection which is ordered\*, changeable and do not allow duplicates.

Dictionaries are written with curly brackets, and have keys and values:

### Example

Create and print a dictionary:

thisdict = {  
"brand": "Ford",  
"model": "Mustang",  
"year": 1964  
}  
print(thisdict)

* Dictionary Items: Dictionary items are ordered, changeable, and does not allow duplicates. Dictionary items are presented in key:value pairs, and can be referred to by using the key name.
* Ordered or Unordered? :As of Python version 3.7, dictionaries are ordered. In Python 3.6 and earlier, dictionaries are unordered. When we say that dictionaries are ordered, it means that the items have a defined order, and that order will not change. Unordered means that the items does not have a defined order, you cannot refer to an item by using an index.
* Changeable: Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created.
* Duplicates Not Allowed: Dictionaries cannot have two items with the same key: