LECTURE ONE.

Data Science Course with Python

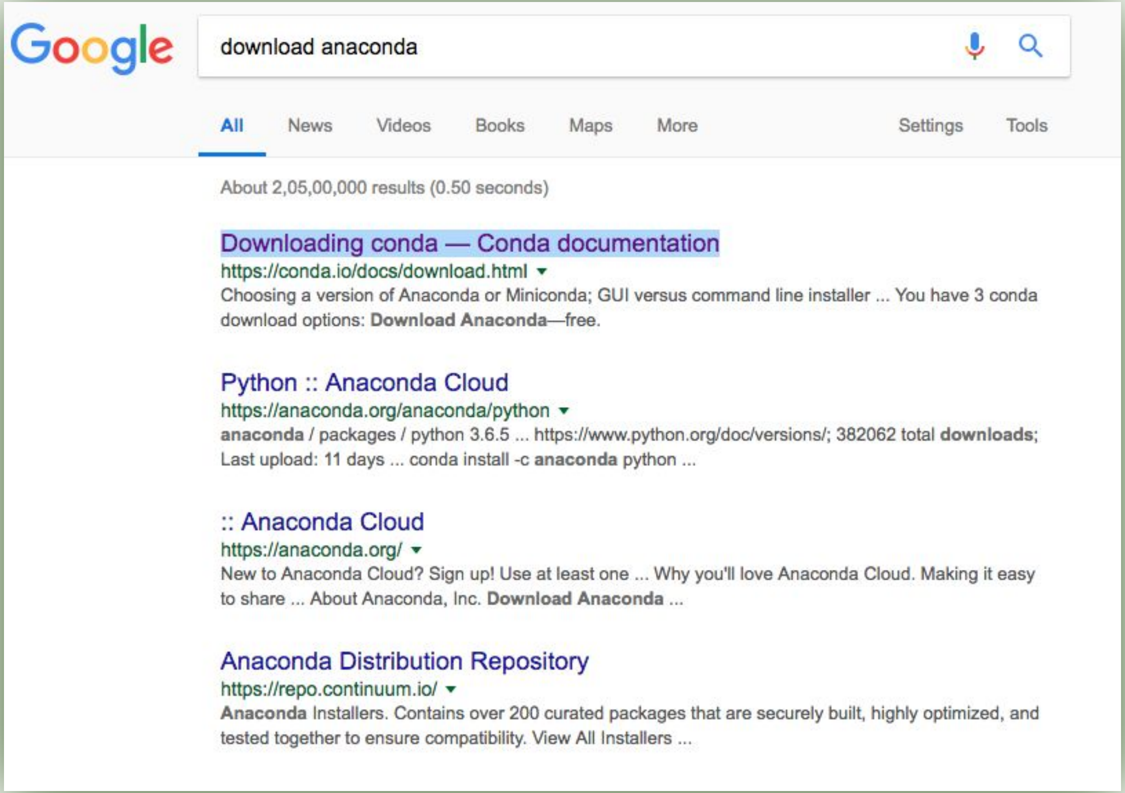
- Jupyter Notebook Setup & Installation
- Walkthrough of Jupyter Environment
- Basics of Python Programming

Jupyter Installation

Installing Python Anaconda/Jupyter

- Download Link : https://www.anaconda.com/download/
- Select Operating System : (OS X/Windows/Linux)
- Install : Anaconda Navigator
- My Applications Tab → Launch Jupyter 4.1.0

Step1:



About 2,05,00,000 results (0.50 seconds)

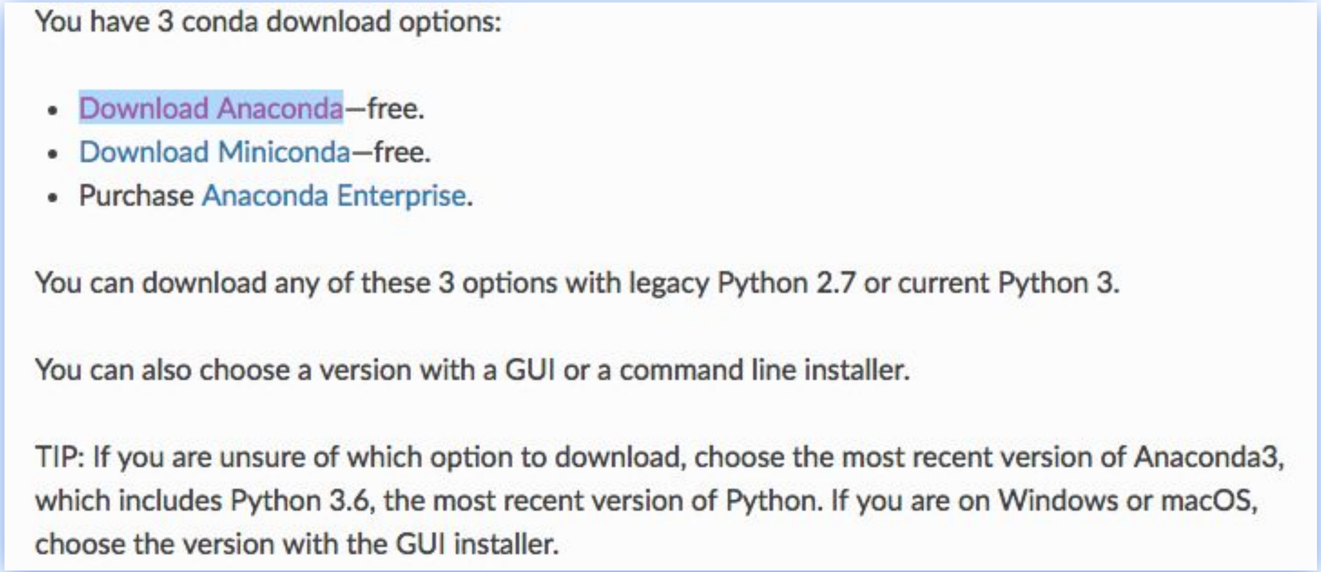
Downloading conda — Conda documentation
https://conda.io/docs/download.html
Choosing a version of Anaconda or Miniconda; GUI versus command line installer ... You have 3 conda download options: **Download Anaconda**—free.

Python :: Anaconda Cloud
https://anaconda.org/anaconda/python
anaconda / packages / python 3.6.5 ... https://www.python.org/doc/versions/; 382062 total **downloads**; Last upload: 11 days ... conda install -c **anaconda** python ...

:: Anaconda Cloud
https://anaconda.org/
New to Anaconda Cloud? Sign up! Use at least one ... Why you'll love Anaconda Cloud. Making it easy to share ... About Anaconda, Inc. **Download Anaconda** ...

Anaconda Distribution Repository
https://repo.continuum.io/
Anaconda Installers. Contains over 200 curated packages that are securely built, highly optimized, and tested together to ensure compatibility. View All Installers ...

Step 2:


You have 3 conda download options:

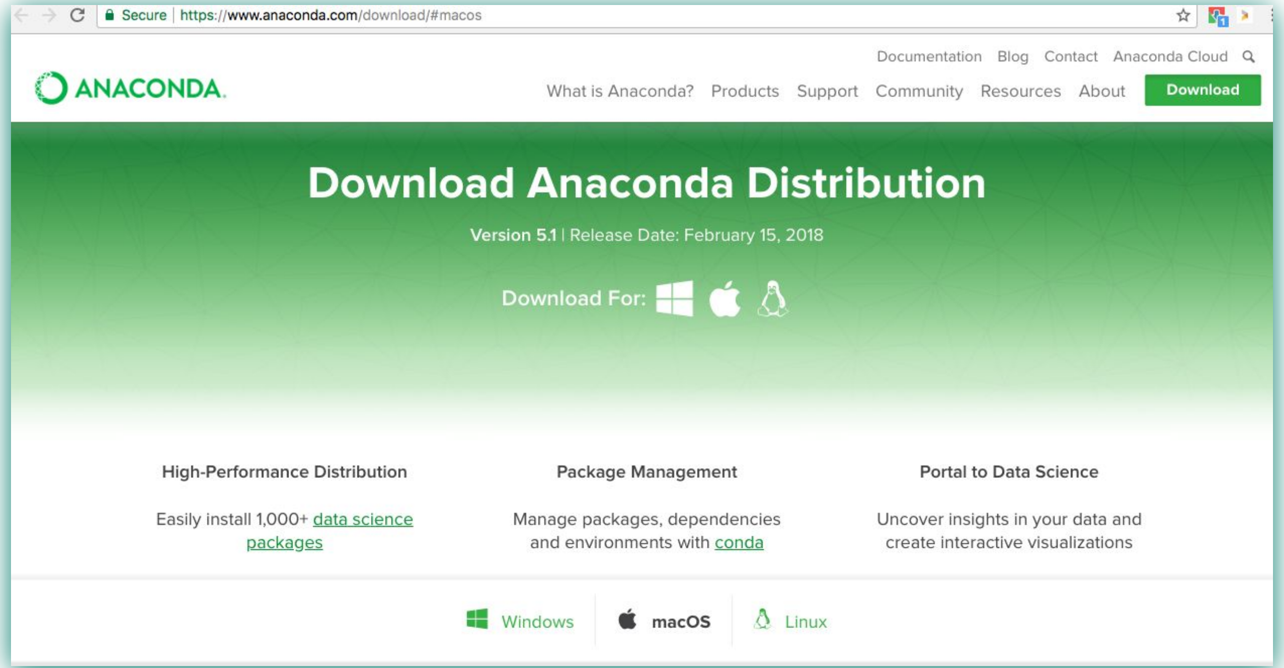
- Download Anaconda—free.
- Download Miniconda—free.
- Purchase Anaconda Enterprise.

You can download any of these 3 options with legacy Python 2.7 or current Python 3.

You can also choose a version with a GUI or a command line installer.

TIP: If you are unsure of which option to download, choose the most recent version of Anaconda3, which includes Python 3.6, the most recent version of Python. If you are on Windows or macOS, choose the version with the GUI installer.

Step 3 :



Secure | https://www.anaconda.com/download/#macos

ANACONDA.

Documentation Blog Contact Anaconda Cloud

What is Anaconda? Products Support Community Resources About Download

Download Anaconda Distribution

Version 5.1 | Release Date: February 15, 2018

Download For:

High-Performance Distribution

Easily install 1,000+ data science packages

Package Management

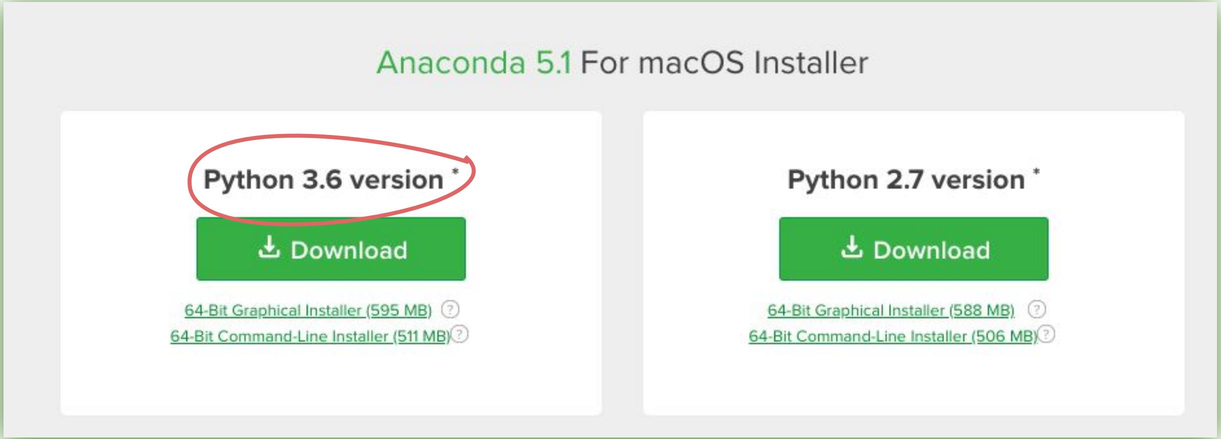
Manage packages, dependencies and environments with conda

Portal to Data Science

Uncover insights in your data and create interactive visualizations

Windows macOS Linux

Step 4:


Anaconda 5.1 For macOS Installer

Python 3.6 version *

64-Bit Graphical Installer (595 MB)

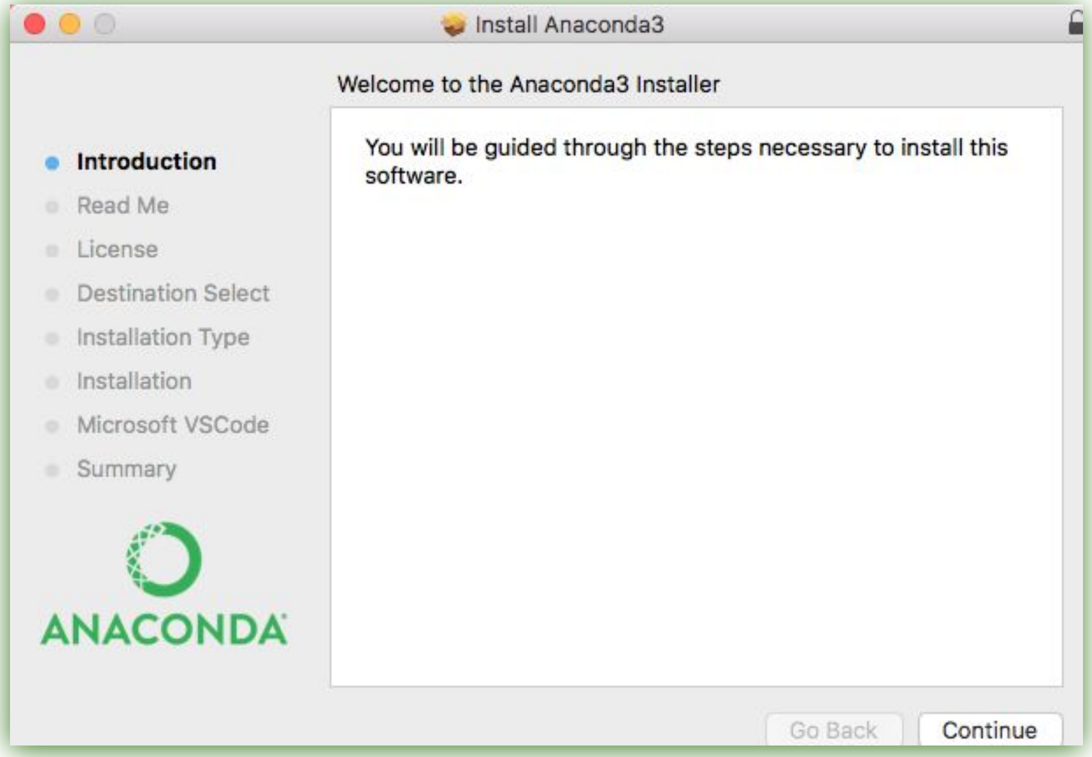
64-Bit Command-Line Installer (511 MB)

Python 2.7 version *

64-Bit Graphical Installer (588 MB)

64-Bit Command-Line Installer (506 MB)

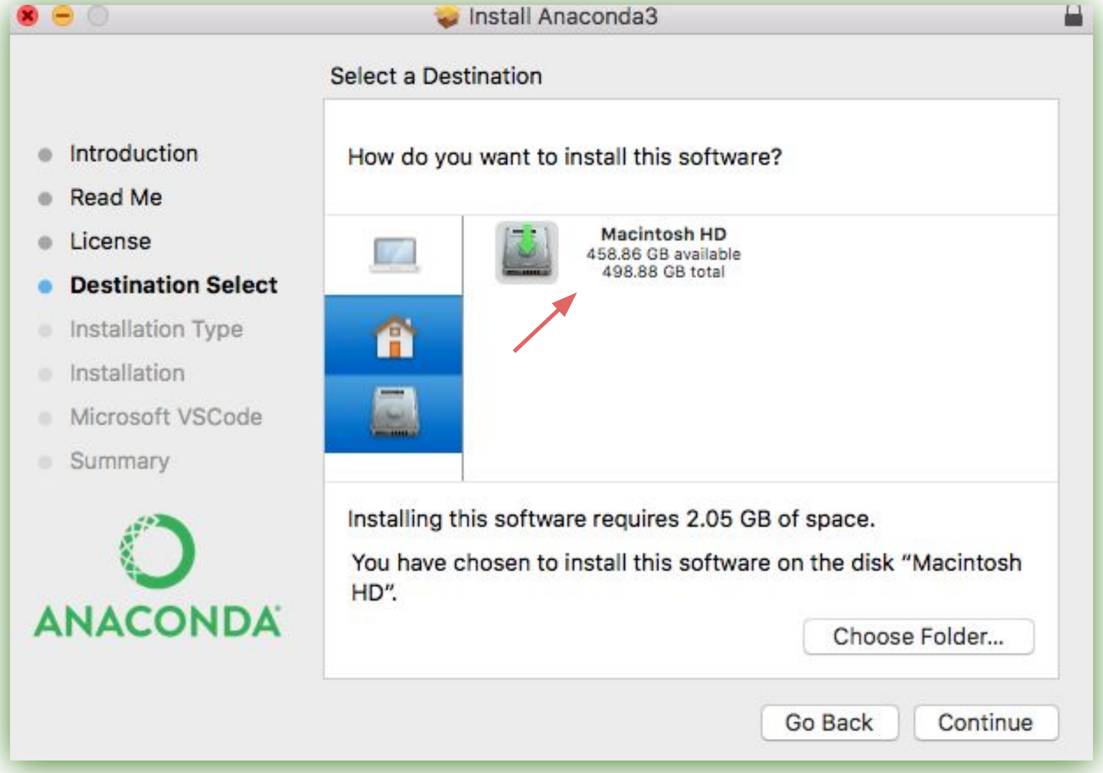
Step 5:



Welcome to the Anaconda3 Installer

You will be guided through the steps necessary to install this software.

Step 6:



Select a Destination

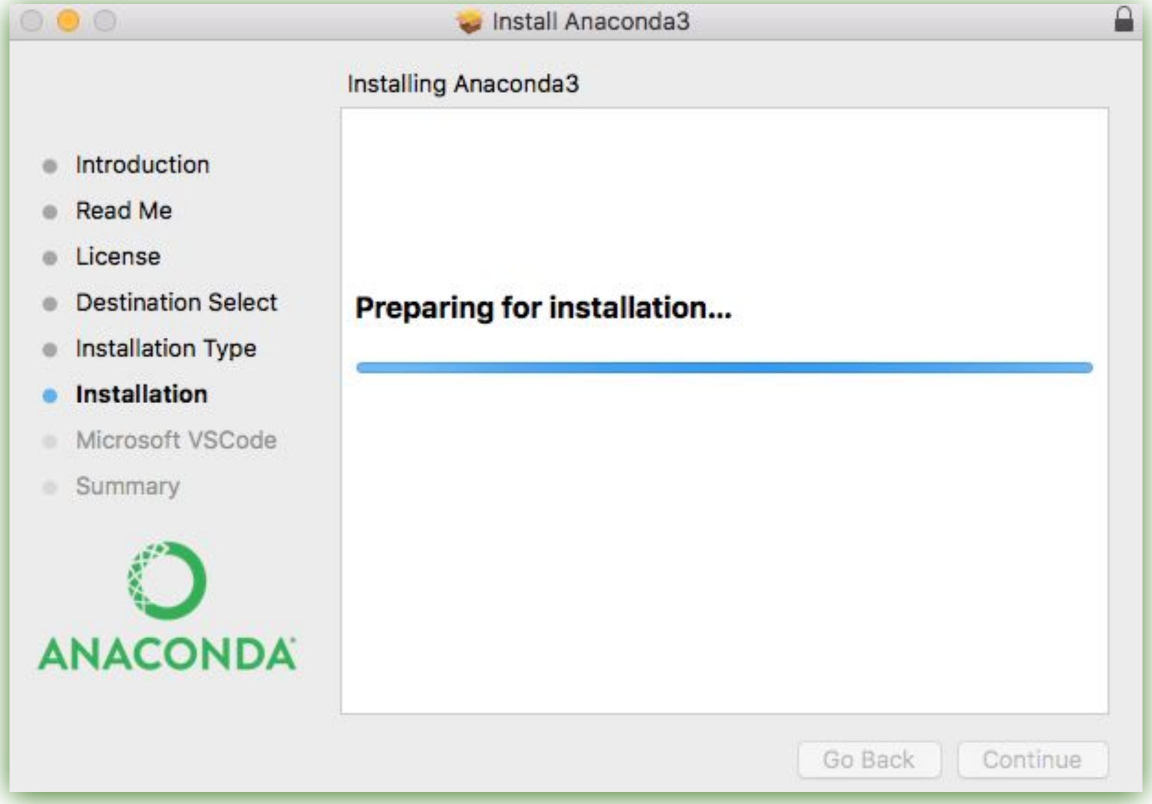
How do you want to install this software?

Macintosh HD
458.86 GB available
498.88 GB total

Installing this software requires 2.05 GB of space.

You have chosen to install this software on the disk "Macintosh HD".

Step 7:

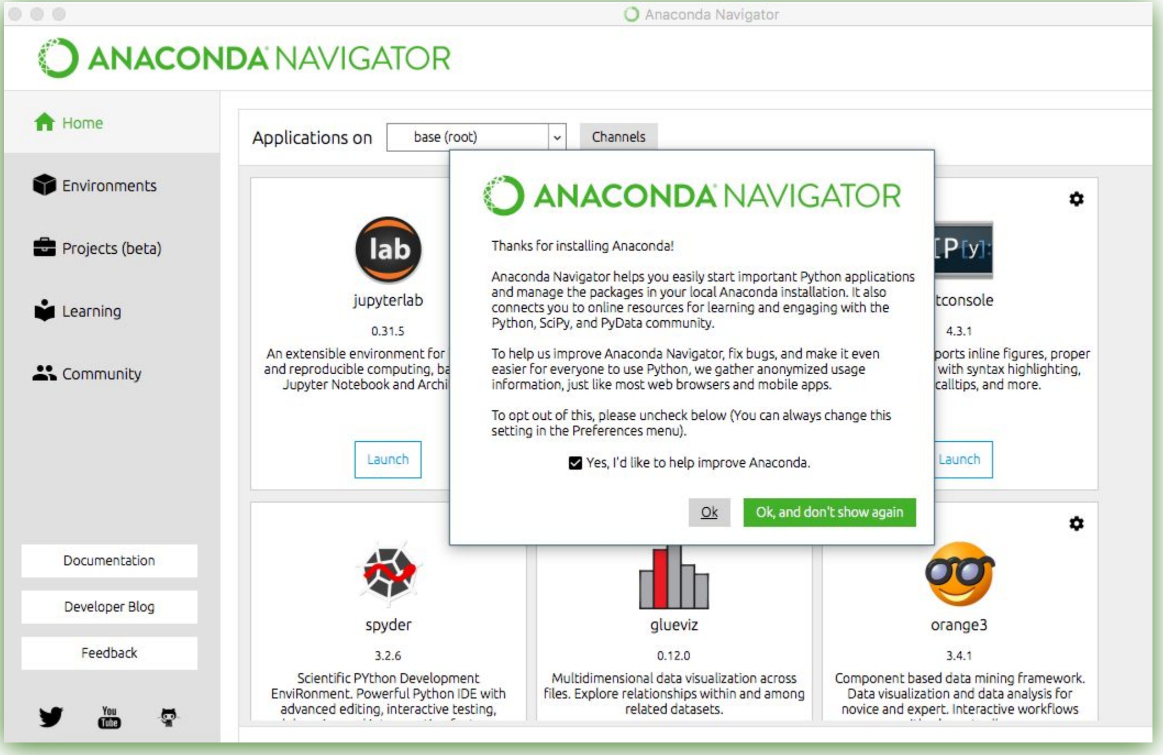

Install Anaconda3

Installing Anaconda3

- Introduction
- Read Me
- License
- Destination Select
- Installation Type
- **Installation**
- Microsoft VSCode
- Summary

Preparing for installation...

Step 8:



An extensible environment for and reproducible computing, ba Jupyter Notebook and Archi

ports inline figures, proper with syntax highlighting, calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing,

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows

ANACONDA NAVIGATOR

Thanks for installing Anaconda!

Anaconda Navigator helps you easily start important Python applications and manage the packages in your local Anaconda installation. It also connects you to online resources for learning and engaging with the Python, SciPy, and PyData community.

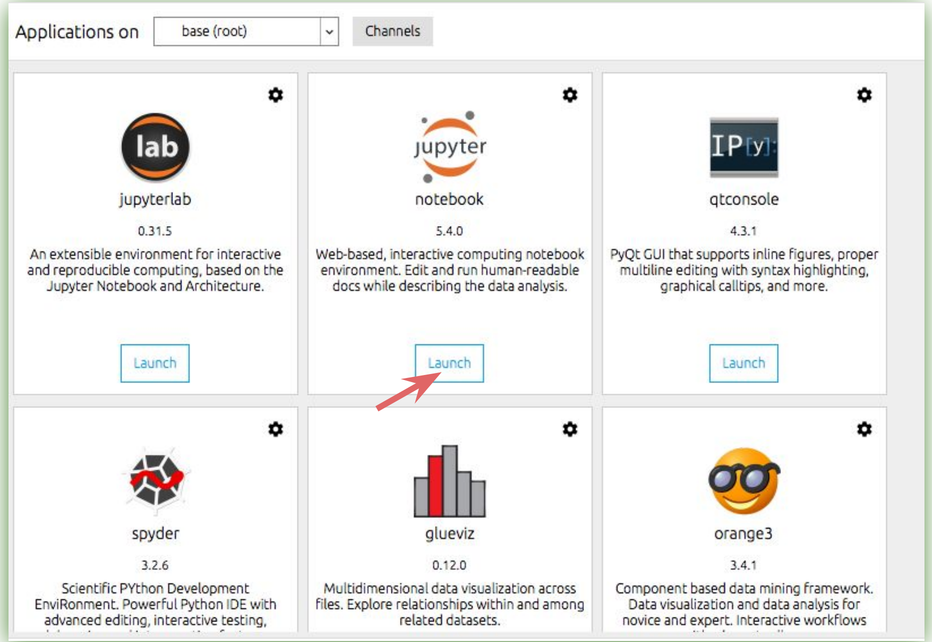
To help us improve Anaconda Navigator, fix bugs, and make it even easier for everyone to use Python, we gather anonymized usage information, just like most web browsers and mobile apps.

To opt out of this, please uncheck below (You can always change this setting in the Preferences menu).

☑ Yes, I'd like to help improve Anaconda.

Ok | Ok, and don't show again

Step 9:



Applications on base (root) Channels

jupyterlab
0.31.5
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.
Launch

notebook
5.4.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.
Launch

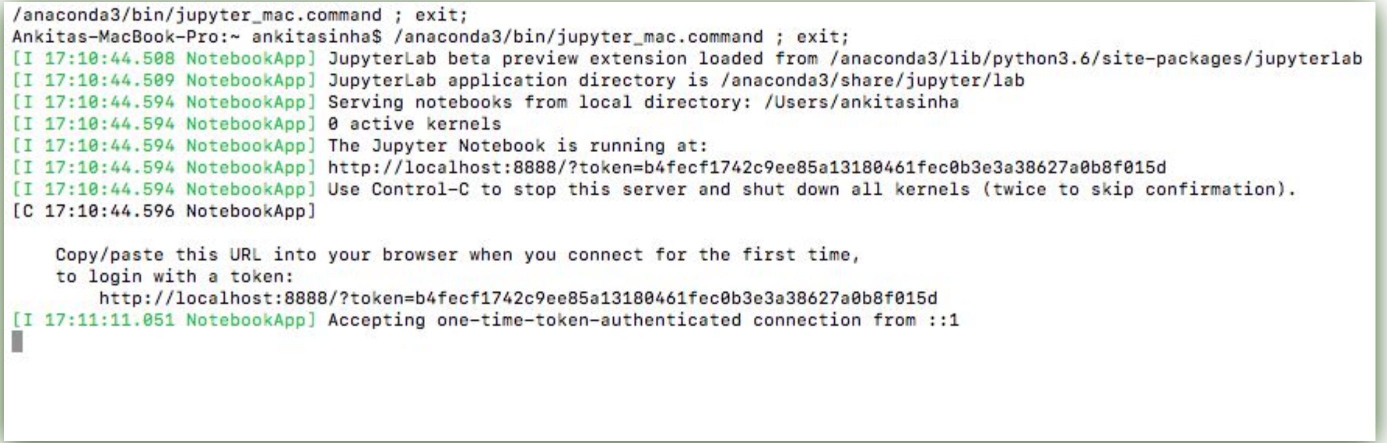
qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

spyder
3.2.6
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing,

glueviz
0.12.0
Multidimensional data visualization across files. Explore relationships within and among related datasets.

orange3
3.4.1
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows

Step 10:

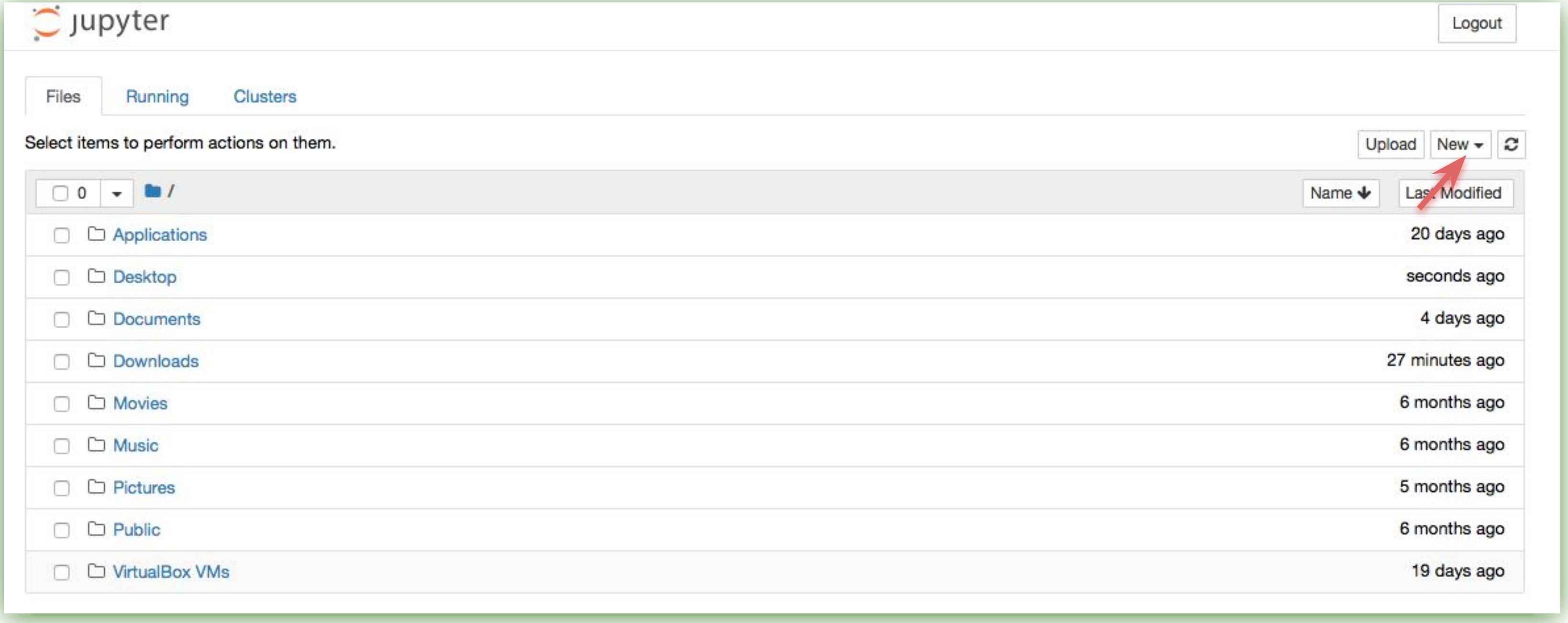


```
/anaconda3/bin/jupyter_mac.command ; exit;
Ankitas-MacBook-Pro:~ ankitasinha$ /anaconda3/bin/jupyter_mac.command ; exit;
[I 17:10:44.508 NotebookApp] JupyterLab beta preview extension loaded from /anaconda3/lib/python3.6/site-packages/jupyterlab
[I 17:10:44.509 NotebookApp] JupyterLab application directory is /anaconda3/share/jupyter/lab
[I 17:10:44.594 NotebookApp] Serving notebooks from local directory: /Users/ankitasinha
[I 17:10:44.594 NotebookApp] 0 active kernels
[I 17:10:44.594 NotebookApp] The Jupyter Notebook is running at:
[I 17:10:44.594 NotebookApp] http://localhost:8888/?token=b4fecf1742c9ee85a13180461fec0b3e3a38627a0b8f015d
[I 17:10:44.594 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 17:10:44.596 NotebookApp]

    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://localhost:8888/?token=b4fecf1742c9ee85a13180461fec0b3e3a38627a0b8f015d
[I 17:11:11.051 NotebookApp] Accepting one-time-token-authenticated connection from ::1
```

Note : By default the first notebook server starts on your localhost on port 8888, and later notebook servers search for ports near that one.

Step 11:

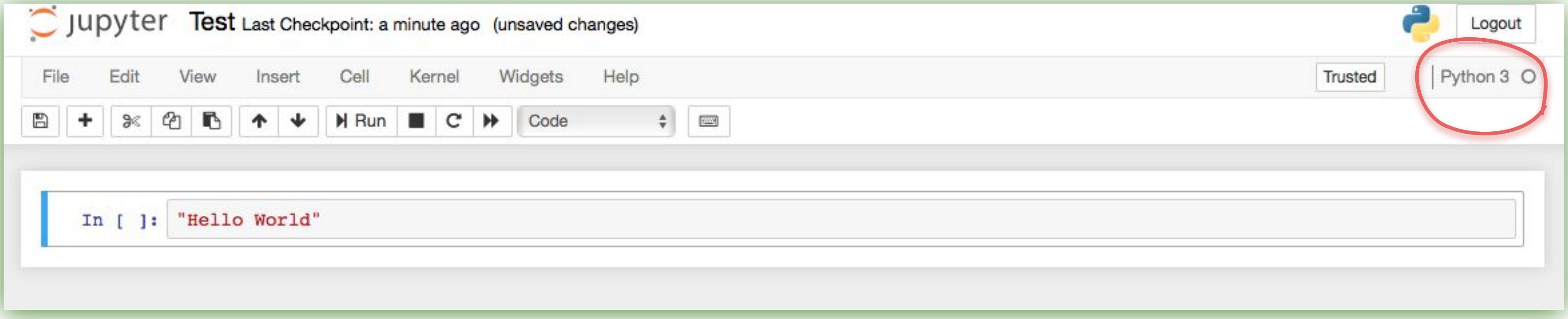


Select items to perform actions on them.

Applications	20 days ago
Desktop	seconds ago
Documents	4 days ago
Downloads	27 minutes ago
Movies	6 months ago
Music	6 months ago
Pictures	5 months ago
Public	6 months ago
VirtualBox VMs	19 days ago

Note : Dashboard serves as a homepage for the notebook. Its main purpose is to display the notebooks and files in the current directory. Select New to open a new notebook. Choose Python 3 kernel.

Step 12:

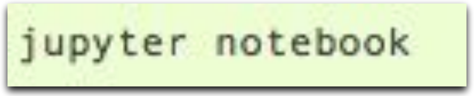


```
In [ ]: "Hello World"
```

Accessing Jupyter Notebooks

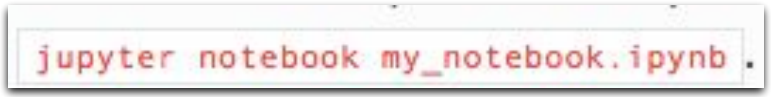
Bash Commands

Terminal/ Command Line



```
jupyter notebook
```

File Access

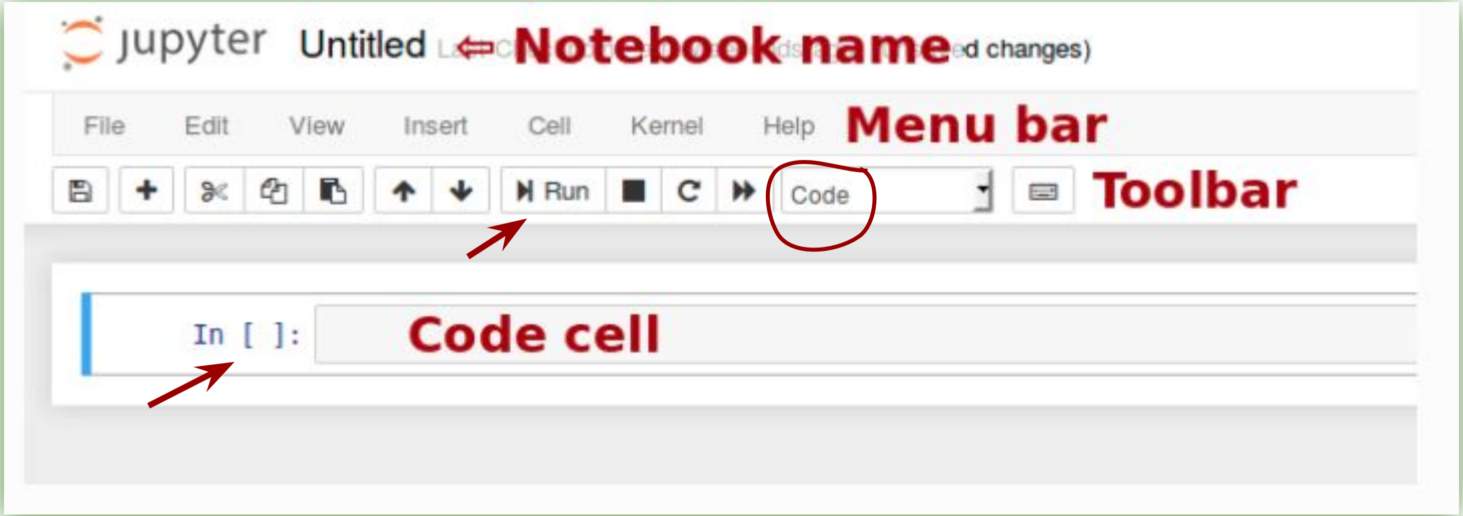


```
jupyter notebook my_notebook.ipynb
```

Note: New notebook is opened in a new tab and is located in the current directory.

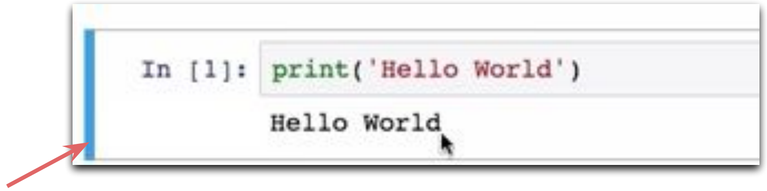
Understanding Jupyter Notebooks

Jupyter User Interface



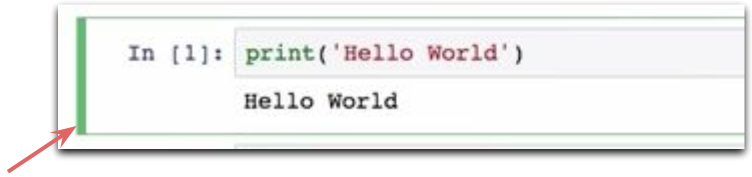
Types of Keyboard Input Modes

1. Command Mode



```
In [1]: print('Hello World')
        Hello World
```

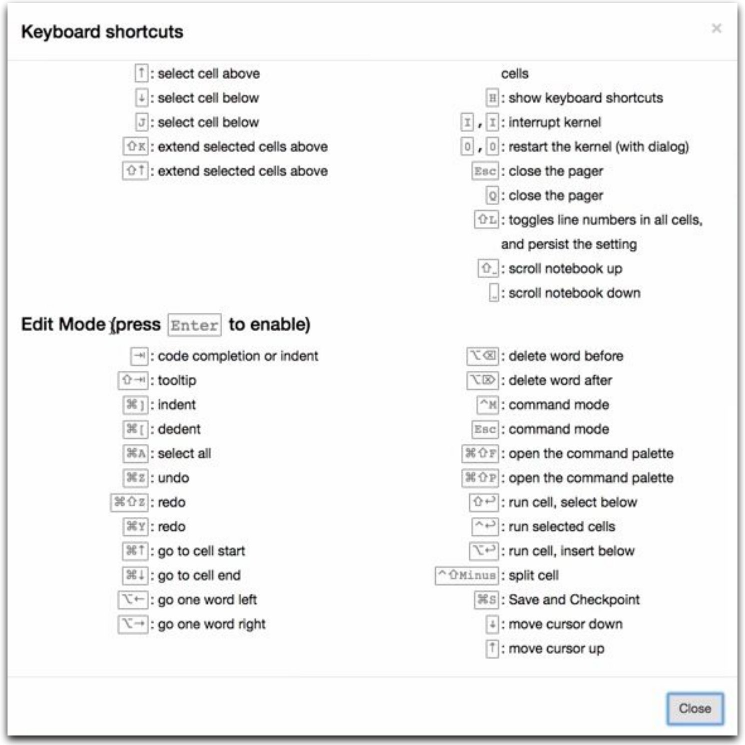
2. Edit Mode



```
In [1]: print('Hello World')
        Hello World
```

Recommended Keyboard Shortcuts

1. **Basic navigation:** enter, shift-enter, esc, up/k, down/j
2. **Saving the notebook:** s
3. **Change Cell types:** y, m, 1-6, t
4. **Cell creation:** a, b, alt-enter
5. **Cell editing:** x, c, v, d, z
6. **Kernel operations:** i, 00
7. **Documentation support :** shift-tab



Keyboard shortcuts

↑ : select cell above
↓ : select cell below
J : select cell below
⇧K : extend selected cells above
⇧↑ : extend selected cells above

cells
H : show keyboard shortcuts
I , I : interrupt kernel
0 , 0 : restart the kernel (with dialog)
Esc : close the pager
Q : close the pager
⇧L : toggles line numbers in all cells, and persist the setting
⇧␣ : scroll notebook up
␣ : scroll notebook down

Edit Mode (press Enter to enable)

⇥ : code completion or indent
⇧⇥ : tooltip
⌘] : indent
⌘[: dedent
⌘A : select all
⌘Z : undo
⌘⇧Z : redo
⌘Y : redo
⌘↑ : go to cell start
⌘↓ : go to cell end
⌥← : go one word left
⌥→ : go one word right

⌥⌫ : delete word before
⌥⌦ : delete word after
^M : command mode
Esc : command mode
⌘⇧F : open the command palette
⌘⇧P : open the command palette
⇧↩ : run cell, select below
^↩ : run selected cells
⌥↩ : run cell, insert below
^⇧Minus : split cell
⌘S : Save and Checkpoint
↓ : move cursor down
↑ : move cursor up

Close

Types of Cells

1. Code Cell (Default)
2. Markdown Cells (explanatory text)
3. Raw Cells

Code Cells

1. Contains Python code
2. Interpreted directly from a cell.
3. Interpreter processes the code snippet in the cell
4. Code cells display code and output.

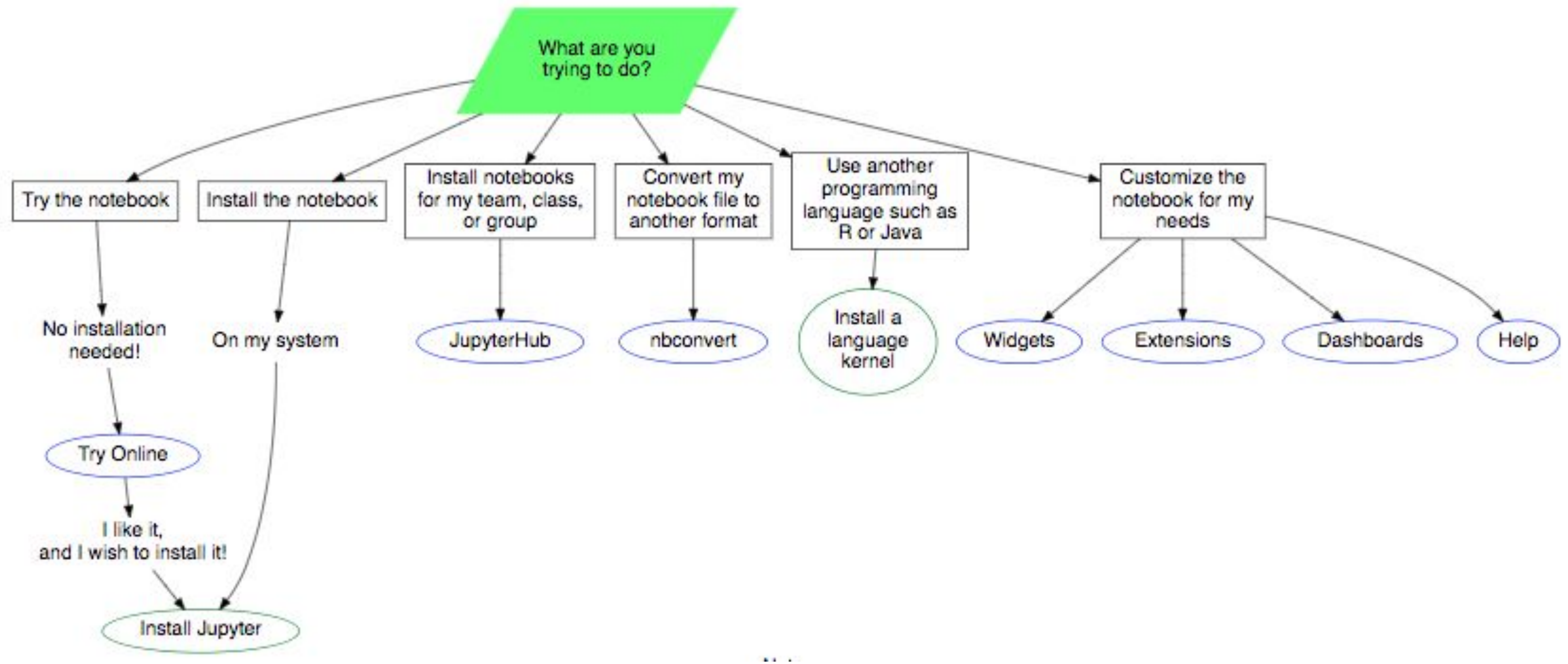
Markdown Cells

1. Display text in web page format
2. Can be shared on Github or as a blogpost.
3. Contains Explanatory Text
4. Formats the way cells are displayed
5. Markdown cells only display output.

Raw Cells

1. Not evaluated by the notebook
2. Can be of different formats e.g Latex/HTML etc.
3. Stored as Metadata

Components of Jupyter


What are you trying to do?
Try the notebook
Install the notebook
Install notebooks for my team, class, or group
Convert my notebook file to another format
Use another programming language such as R or Java
Customize the notebook for my needs
No installation needed!
On my system
JupyterHub
nbconvert
Install a language kernel
Widgets
Extensions
Dashboards
Help
Try Online
I like it, and I wish to install it!
Install Jupyter

Python Programming Basics

First Program

```
>>>print(“Hello World!”)
Hello World!
```

Numerical Calculations

```
>>> 2+2
4
>>>5+4-3
6
>>>2*(3+5)
16
>>>10/2
5.0
>>>(-7+2)*(-4)
20
```

- Dividing By 0 produces zero division error

Floats

```
>>> 7.85500
7.855
>>>3/4
0.75
>>>2.0+5.3
7.3
>>>6*7.0                            //Implicit conversion
42.0
```

Exponentiation

```
>>> 2**5
32
>>>9**(½)
3.0
```

Floor Division and Modulo

```
>>> 20//6                    //Evaluates Quotient
3
>>> 42.0%5                   //Evaluates Remainder
2.0
```

Strings

```
>>> print("Python is Fun")
Python is Fun
>>> print('A billion cells')
A billion cells
```

Escape Characters

```
>>> print('Meet Peter\'s friend. He\'s the head boy of School.')
Meet Peter's friend. He's the head boy of School.
>>>print("The Teacher asked, \"Where is everyone?\" to the empty classroom.")
The Teacher asked, "Where is everyone?" to the empty classroom.
>>> print('Good Morning. \n Welcome to Trip Planner.')
Good Morning.
Welcome to Trip Planner.
```

Input and Output

```
>>>print(“Welcome\n Aboard!!”)
Welcome
 Aboard!!

>>>print(‘print(“print”)’)
print(“print”)
```

Input and Output

```
>>>input('Enter Your Name:')
Enter Your Name: Sam
Sam
```

Concatenation

```
>>>"Tea"+'break'
Teabreak
>>>print("Tea"+", "+"Break")
Tea, Break
>>>"2"+"2"
22
>>>1+'2'+3+'4'
TypeError: unsupported operand type(s) for +:
'int' and 'str'
```

String Operations

```
>>>print("Monday"*3)
MondayMondayMonday

>>>4 * '5'
5555

>>>print(3*'7')
777
```

- Strings cannot be multiplied with string or floats. It will generate a TypeError

Type Conversion

```
>>>"2"+"3"
'23'
>>>int("2")+int("3")
5
```

- Integer Type Conversion : int()
- Float Type Conversion : float()
- String Type Conversion: str()

Converting User input

```
>>> float(input(“Enter a number: “))+
float(input(“Enter another number: ”))

Enter a number: 25
Enter another number: 75
100.0
```

Variables

```
>>> x=7
>>> print(x)
7
>>> print(x+3)
10
>>>print(x)
7
```

Reassigning Variables

```
>>> x=125.25
>>> print(x)
125.25
>>> x="This is a string"
>>>print(x+'!!')
This is a string!!
```

- Variables do not have a specific type

Naming Restrictions

```
//Characters Allowed
Letters*
Numbers
Underscores

Table_furniture                // Valid
12Table                        // Invalid
Table Chair                    // Invalid
```

- ❖ Letters are Case Sensitive in python

Deleting Variables

```
>>> Fruit = “Apple”
>>>Fruit
Apple
>>>Snacks
NameError
>>>del Fruit
>>>Fruit
```

- Referencing unassigned variables lead to NameError

In-Place Operators

```
>>>x=2
>>>print(x)
2
>>>x+=3
>>>print(x)
5
>>>y=”Air”
>>>y+=”Travel”
>>>print(y)
AirTravel
```

```
>>>x*=5
??
>>>x-=2
??
>>>x/=2
??
>>>x%=2
??
```

Booleans & comparison

Boolean Values

Comparison operators

Equals Operator	==
Not Equals	!=
Less than	<
Greater than	>
Less than equal	<=
Greater than equal	>=

Conditional Statements

```
if expression:
    statements
```

- ❖ Python uses indentations instead of curly braces to delimit blocks of code. No use of semi-colon after statements.

Conditional Statements

```
if exp_1:
    Statements
elif exp_2:
    Statements
Else:
    Statements
```

❖ elif is equivalent to else if elseif elsif in other programming languages.

Boolean Logic

and **or** **not**

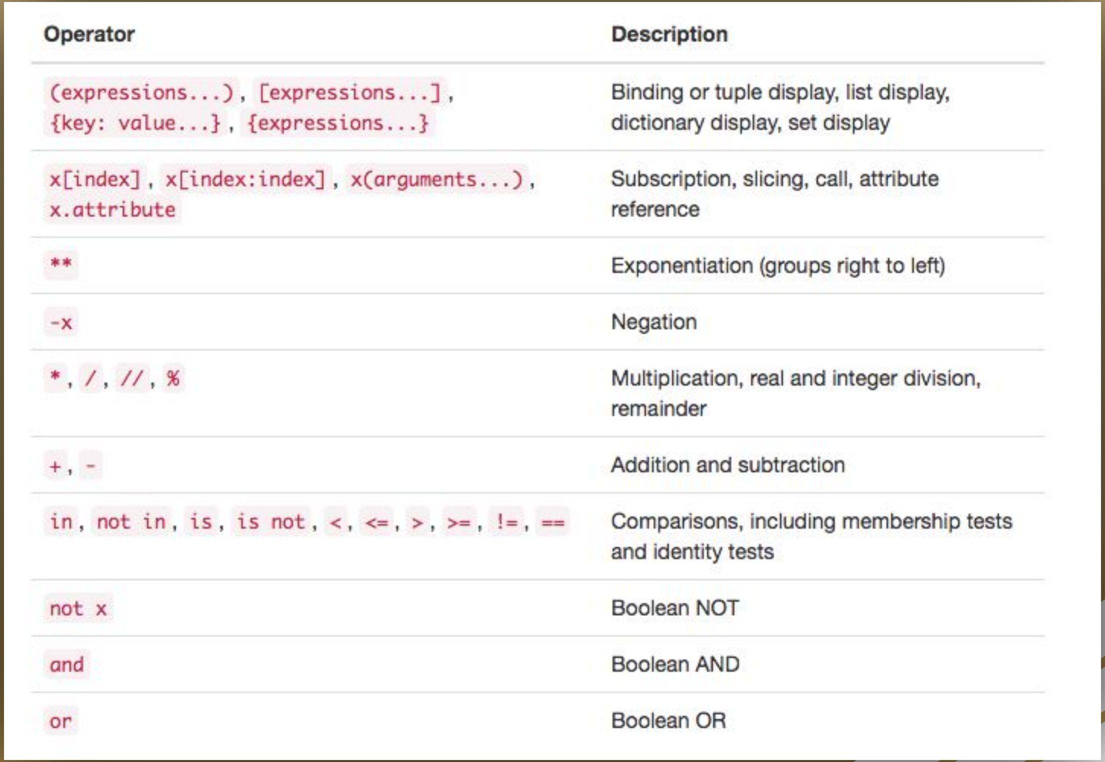
```
>>>1==1 and 2==2
True
>>>1!=1 or 2==2
True
>>>not 1==1
False
>>>not 1>7
True
```

Operator Precedence

PEMDAS

P	⇒	Parenthesis	(highest priority)
E	⇒	Exponentials	
M	⇒	Multiplication	
D	⇒	Division	
A	⇒	Addition	
S	⇒	Subtraction	(lowest priority)

Operator Precedence



Operator	Description
`(expressions...)`, `[expressions...]`, `{key: value...}`, `{expressions...}`	Binding or tuple display, list display, dictionary display, set display
`x[index]`, `x[index:index]`, `x(arguments...)`, `x.attribute`	Subscription, slicing, call, attribute reference
`**`	Exponentiation (groups right to left)
`-x`	Negation
`*`, `/`, `//`, `%`	Multiplication, real and integer division, remainder
`+`, `-`	Addition and subtraction
`in`, `not in`, `is`, `is not`, `<`, `<=`, `>`, `>=`, `!=`, `==`	Comparisons, including membership tests and identity tests
`not x`	Boolean NOT
`and`	Boolean AND
`or`	Boolean OR

Quiz 1

What is the output of this code?

```
>>>Fruit = 5
>>>Basket = 3
>>>del Fruit
Basket = 2
Fruit=10
print(Fruit*Basket)
```

Output:

20

Quiz 2

Which of these are valid variable name?

```
A.   A variable name
B.   A_VARIABLE_NAME
C.   A-variable
D.   A_123
```

Output:

B

D

Quiz 3

Find the output of this code?

```
>>>  foo = “7”
>>>  foo= foo + “0”
>>>  Bar = int(foo) +8
>>>  print(float(Bar))
```

Output:

```
78.0
```

Quiz 4

Find the output of this code?

```
>>> word = input('Enter a word:')
Enter a word: Time

>>> print(word + ' table')
```

Output:

Time table

Quiz 5

Find the output of this code?

```
>>>  x = 5
>>> y = x+3
>>> y = int(str(y)+”2”)
>>>  print(y)
```

Output:

```
82
```

Quiz 6

Fill in the blanks to declare a variable, multiply 5 to it and print its value?

```
>>>  x = 5
>>> x * = 5
>>>  print(x)
```

Quiz 7

Fill in the blanks to declare a variable, add 5 to it and print its value?

```
>>>  x = 5
>>> x + = 5
>>>  print(x)
```

Quiz 8

What is the output of this code?

```
>>>  x = 3
>>> num = 17
>>>  print(num%x)
```

Output:

```
2
```

Quiz 9

What is the output of this code?

```
>>>  7!=8
```

Output:

```
True
```

Quiz 10

What is the output of this code?

```
>>> 7 > 7.0
```

Output:

```
False
```

Quiz 11

What is the output of this code?

```
>>> 8.7 <=8.70
```

Output:

```
True
```

Quiz 12

What part of an *if* statement should be indented?

1. The first line
2. All of it
3. The Statements within it

Output:

Option Three

Quiz 13

What is the output of this code?

```
>>> Books = 7
>>> if Books > 5:
>>>     print(“five”)
>>> if Books > 8:
>>>     print(“eight”)
```

Output:

five

Quiz 14

What is the output of this code?

```
>>> num = 7
>>> if num > 3:
>>>     print("3")
>>>     if num < 5:
>>>         print("5")
>>>         if num ==7:
>>>             print("7")
```

Output:

```
3
```

Quiz 15

What is the output of this code?

```
>>> if 1+1==2:
>>>     if 2*2==8:
>>>         print(“if”)
>>>     else:
>>>         print(“else”)
```

Output:

else

Quiz 16

Fill in the blanks to compare the variables and print output.

```
>>> x = 10
>>> y = 20
>>> __ x>y__
>>>     print("if statement")
>>> ___
>>>     print ("else statement")
```

Quiz 17

What is the output of this code?

```
>>> if (1==1) and (2+2>3):
>>>     print(“true”)
>>> else:
>>>     print(“false”)
```

Output:

```
true
```

Quiz 18

Fill in the blanks to print Factory Sale

```
>>> purchase=500
>>> credits = 100
>>> if purchase>599____credits>99:
>>>     _____("Factory Sale")
```

Output:

- > or
- > print

Quiz 19

What is the output?

```
>>> if not True:
>>>     print(“1”)
>>> elif not(1+1==3):
>>>     print(“2”)
>>> else:
        print(“3”)
```

Output:

2

Quiz 20

What is the result of this code?

```
>>> if 1+1*3==6:
>>>     print(“Yes”)
>>> else:
>>>     print(“No”)
```

Output:

```
No
```

Quiz 21

What is the output of this code?

```
>>> x=4
>>> y=2
>>> if not 1+1==y or x==4 and 7==8:
>>>     print("Yes")
>>> elif x>y:
>>>     print ("No")
```

Output:

No

Helpful Tips

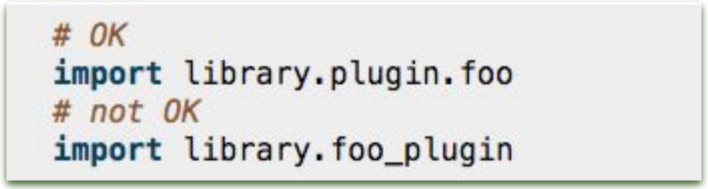
★ While naming a file never include special characters like spaces, hyphens, periods or slashes.

`my.spam.py` *// Error Prone*

Python expects to find a spam.py file in a folder named my which is not the case here.

★ Similarly, Hyphens can be mistaken for subtract operator.

★ Try to keep module names short so there is no need to separate words.



```
# OK
import library.plugin.foo
# not OK
import library.foo_plugin
```

Summary

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

★ Ideal for Rapid Application Development

★ Also used as a scripting language or glue language to connect existing components together.

★ Reduces maintenance costs by Enhancing readability

Assignment

What are the advantages and limitations of using Python?

Further Reading

1. *https://www.educba.com/benefits-and-limitations-of-using-python/*

2. *https://www.educba.com/java-vs-python/*

Next….

1. *Hands on Lab on Jupyter Notebook*
2. *Loops*
3. *Data Structures in python : List, Range, Dictionary*
4. *Functions and Modules*
5. *Files*
6. *Exceptions*

End of Lecture Two