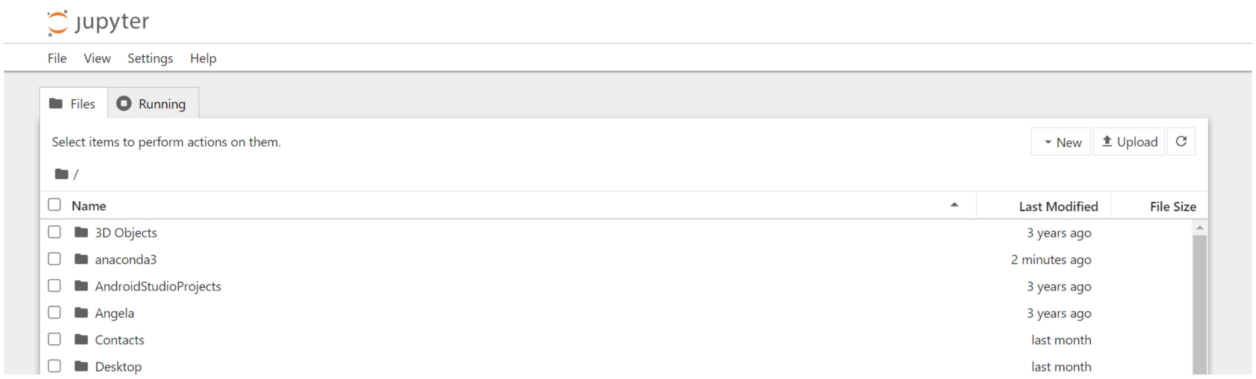
Data Analyst Bootcamp

Python

Install Anaconda in ur laptop

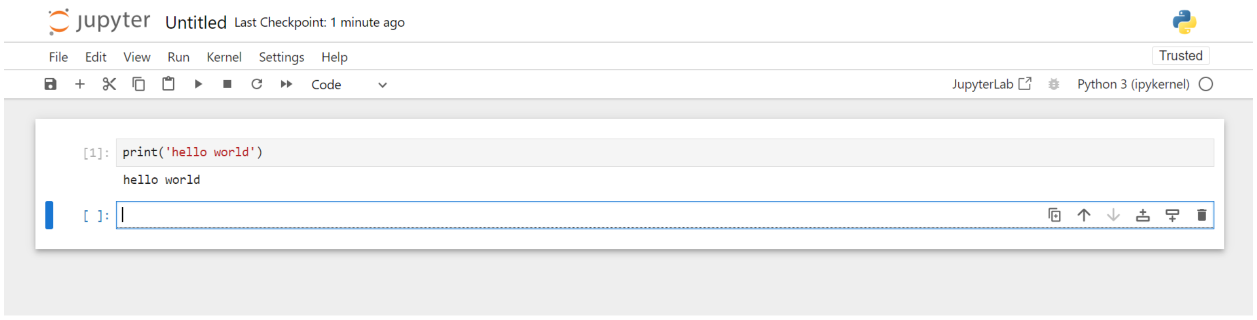


Select items to perform actions on them.

/

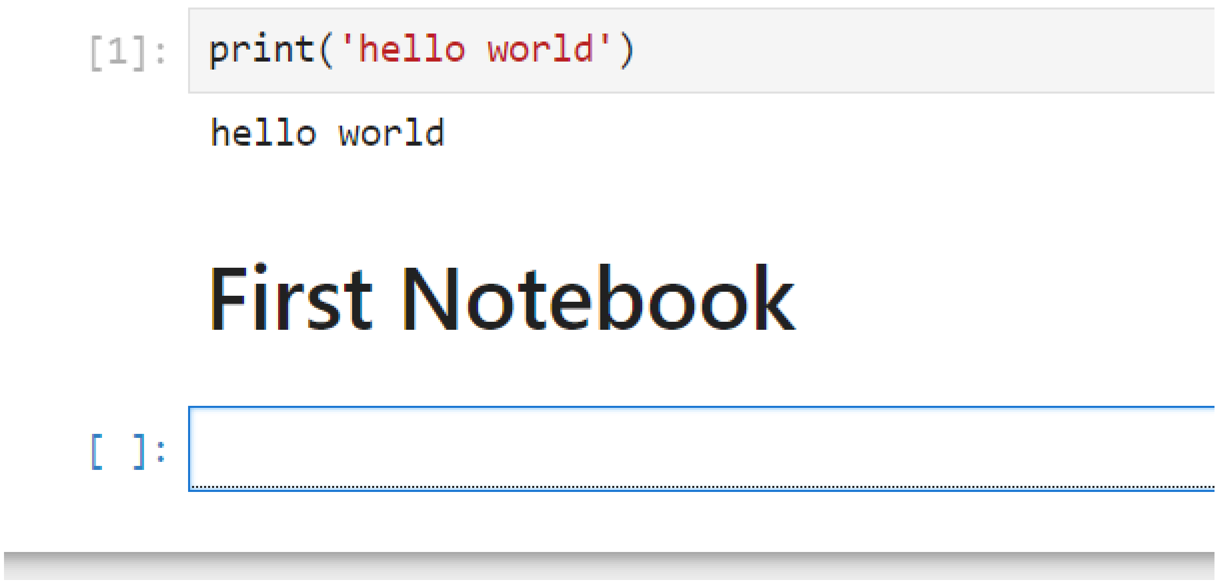
Name	Last Modified	File Size
3D Objects	3 years ago	
anaconda3	2 minutes ago	
AndroidStudioProjects	3 years ago	
Angela	3 years ago	
Contacts	last month	
Desktop	last month	

Click on new → notebook


jupyter
Untitled
Last Checkpoint: 1 minute ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

```
[1]: print('hello world')
     hello world
[ ]:
```

write on cell and click shift+enter to run the cell

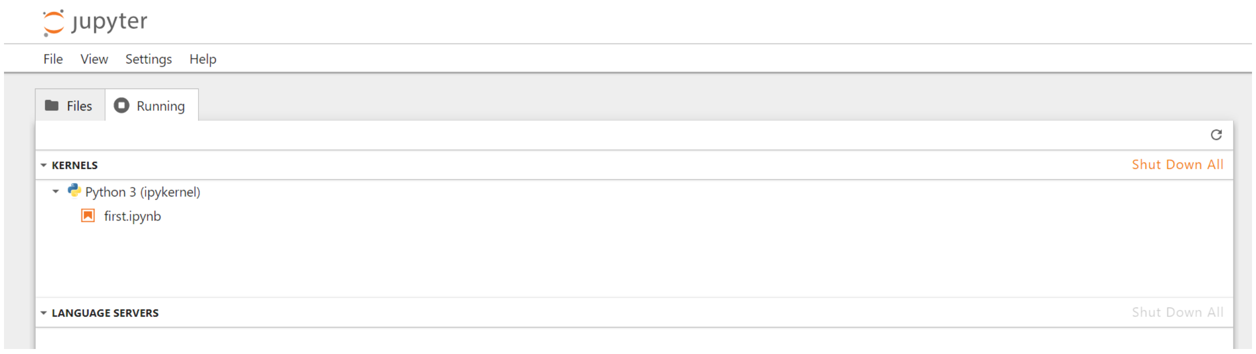


```
[1]: print('hello world')
hello world
```

First Notebook

```
[ ]:
```

first notebook because we clicked markdown instead of code on top



we can shutdown the program here

Variables

```
Variables in Python
x = 22

print(x)
22
type(x)
int
```

```
y = 'Mint Chocolate Chip'

print(y)
Mint Chocolate Chip
type(y)
str
# Case Sensitivity

y = 'Chocolate'

Y = 'Mint Chocolate Chip'

print(Y)
Mint Chocolate Chip
#Assign multiple values to multiple variables

x,y,z = 'Chocolate', 'Vanilla', 'Rocky Road'

print(x)
print(y)
print(z)
Chocolate
Vanilla
Rocky Road
#Assign one value to multiple variables

x = y = z = 'Root Beer Float'

print(x)
print(y)
print(z)
Root Beer Float
Root Beer Float
Root Beer Float
ice_cream = ['Chocolate', 'Vanilla', 'Rocky Road']
```

```
x,y,z = ice_cream

print(x)
print(y)
print(z)
Chocolate
Vanilla
Rocky Road
# Camel Case

#Test Variable Case

testVariableCase = 'Vanilla Swirl'
# Pascal Case

#Test Variable Case

TestVariableCase = 'Vanilla Swirl'
# Snake Case

#Test Variable Case

test_variable_case = 'Vanilla Swirl'
#Good ways to write variables

testvar = 'Vanilla Swirl'
test_var = 'Vanilla Swirl'
_test_var = 'Vanilla Swirl'
testVar = 'Vanilla Swirl'
TestVar = 'Vanilla Swirl'
testVar2 = 'Vanilla Swirl'
#Bad ways to write variables

2testVar = 'Vanilla Swirl'
test-Var2 = 'Vanilla Swirl'
test Var2 = 'Vanilla Swirl'
```

```
test,Var2 = 'Vanilla Swirl'
x = 'Ice Cream is my favorite' + 2

print(x)
--------------------------------------------------------------------------
TypeError                                 Traceback (most recen
Input In [19], in <cell line: 1>()
----> 1 x = 'Ice Cream is my favorite' + 2
      3 print(x)

TypeError: can only concatenate str (not "int") to str
y = 3 + 2

print(y)
5
x = 'Ice Cream'
y = ' is'
z = ' my favorite.'

print(x+y+z)
Ice Cream is my favorite.
x = 1
y = 2
z = 3

print(x+y+z)
6
x = 'Ice Cream'
y = 2



print(x,y)
Ice Cream 2
```

List and Tuple are almost similar and Tuple is immutable which is not chanagable

set has unqiue elements

Data Types

```
Data Types
type(-12 + 100)
int
type(12 + 10.25)
float
type(12 + 3j)
complex
#Boolean

type(1 > 5)
bool
1 == 1
True
# Strings

'Single Quote'
'Single Quote'
"Double Quote"
'Double Quote'
multiline = """
The ice cream vanquished
my longing for sweets,
upon this diet I look away,
it no longer exists on this day.

"""
print(multiline)
The ice cream vanquished
my longing for sweets,
upon this diet I look away,
it no longer exists on this day.
```

```
"""
I've always wanted to eat a gallon of "ice cream."
"""
type(multiline)
str
a = 'Hello World!'
print(a[2:5])
llo
a*3
'Hello World!Hello World!Hello World!'
a + a
'Hello World!Hello World!'
# list

[1,2,3]
[1, 2, 3]
['Cookie Dough','Strawberry','Chocolate']
['Cookie Dough', 'Strawberry', 'Chocolate']
['Vanilla', 3, ['Scoops','Spoon'],True]
['Vanilla', 3, ['Scoops', 'Spoon'], True]
ice_cream = ['Cookie Dough','Strawberry','Chocolate']

ice_cream.append('Salted Caramel')

ice_cream
['Cookie Dough', 'Strawberry', 'Chocolate', 'Salted Caramel']
ice_cream[0] = 'Butter Pecan'

ice_cream
['Butter Pecan', 'Strawberry', 'Chocolate', 'Salted Caramel']
nest_list = ['Vanilla', 3, ['Scoops','Spoon'],True]
nest_list[2][1]
'Spoon'
#tuple
```

```
tuple_scoops = (1,2,3,2,1)
type(tuple_scoops)
tuple
tuple_scoops[0]
1
tuple_scoops.append(3)
---------------------------------------------------------------
AttributeError                            Traceback (most recen
Input In [46], in <cell line: 1>()
----> 1 tuple_scoops.append(3)

AttributeError: 'tuple' object has no attribute 'append'
# sets



daily_pints = {1,2,3}
type(daily_pints)
set
print(daily_pints)
{1, 2, 3}
daily_pints_log = {1,2,31,2,3,4,1,2,5,6,3,2}

print(daily_pints_log)
{1, 2, 3, 4, 5, 6, 31}
wifes_daily_pints_log = {1,3,5,7,3,24,5,7,3,2,0}
print(daily_pints_log | wifes_daily_pints_log)
{0, 1, 2, 3, 4, 5, 6, 7, 24, 31}
print(daily_pints_log & wifes_daily_pints_log)
{1, 2, 3, 5}
print(wifes_daily_pints_log - daily_pints_log )
{0, 24, 7}
print(wifes_daily_pints_log ^ daily_pints_log )
{0, 4, 6, 7, 24, 31}
# dictionaries
# Key/Value Pair
```

```
dict_cream = {'name': 'Alex Freberg', 'weekly intake': 5, 'favor
type(dict_cream)
dict
print(dict_cream)
{'name': 'Alex Freberg', 'weekly intake': 5, 'favorite ice crear
dict_cream.values()
dict_values(['Alex Freberg', 5, ['MCC', 'Chocolate']])
dict_cream.keys()
dict_keys(['name', 'weekly intake', 'favorite ice creams'])
dict_cream.items()
dict_items([('name', 'Alex Freberg'), ('weekly intake', 5), ('fa
dict_cream['name']
'Alex Freberg'
dict_cream['name'] = 'Christine Freberg'

print(dict_cream)
{'name': 'Christine Freberg', 'weekly intake': 5, 'favorite ice
dict_cream.update({'name': 'Christine Freberg', 'weekly intake'

print(dict_cream)
{'name': 'Christine Freberg', 'weekly intake': 10, 'favorite ice
del dict_cream['weight']

print(dict_cream)
{'name': 'Christine Freberg', 'weekly intake': 10, 'favorite ice
```

Comparison, Logical, and Membership Operators in Python

```
Comparison Operators
10 == 50
False
10 != 50
True
'Vanilla' == 'Vanilla'
True
```

```
x = 'Vanilla'
y = 'Chocolate'

x != y
True
10 <= 10
True
50 >= 10
True

Logical Operators
(70 > 50) and (50 > 10)
True
(10 > 50) or (50 > 10)
True
('Vanilla' > 'Chocolate') and (50 > 10)
True
not(50 > 10)
False

Membership Operators
ice_cream = 'I love Chocolate ice cream'

'love' in 'I love Chocolate ice cream'
True
scoops = [1,2,3,4,5]
wanted_scoops = 8

wanted_scoops in scoops
False
```

Operator	Name
==	Equal
!=	Not equal
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

Operator	Description
and	Returns True if both statements are true
or	Returns True if one of the statements is true
not	Reverse the result, returns False if the result is true

Operator	Description
in	Returns True if a sequence with the specified value is present in the object
not in	Returns True if a sequence with the specified value is not present in the object

If Else Statements in Python

```
If - Elif - Else Statements
if 25 > 10:
    print('It worked!')
It worked!
if 25 > 10:
    print('It worked!')
else:
    print('It did not work...')
It worked!
if (25 < 10) or (1 < 3):
    print('It worked!')
elif 25 < 20:
    print('elif worked!')
elif 25 < 21:
    print('elif 2 worked!')
elif 25 < 40:
    print('elif 3 worked!')
elif 25 < 50:
    print('elif 4 worked!')
else:
    print('It did not work...')
It worked!
print('It worked!') if 10>30 else print('It did not work...')
It did not work...
if (25 < 10) or (1 < 3):
    print('It worked!')
    if 10 > 5:
        print('This nested if statement worked!')
elif 25 < 20:
    print('elif worked!')
elif 25 < 21:
    print('elif 2 worked!')
elif 25 < 40:
    print('elif 3 worked!')
elif 25 < 50:
    print('elif 4 worked!')
else:
    print('It did not work...')
It worked!
This nested if statement worked!
```

For Loops in Python

```
For Loops
integers = [1,2,3,4,5]
for number in integers:
    print(number)
1
2
3
4
5
for number in integers:
    print('yep!')
yep!
yep!
yep!
yep!
yep!
integers = [1,2,3,4,5]
for Jelly in integers:
    print(Jelly + Jelly)
2
4
6
8
10
ice_cream_dict = {'name': 'Alex Freberg', 'weekly intake': 5, '
for cream in ice_cream_dict.values():
    print(cream)
Alex Freberg
5
['MCC', 'Chocolate']
for key, value in ice_cream_dict.items():
    print(key, '->',value)
name -> Alex Freberg
weekly intake -> 5
favorite ice creams -> ['MCC', 'Chocolate']
Nested For Loops
flavors = ['Vanilla', 'Chocolate', 'Cookie Dough']
toppings = ['Hot Fudge', 'Oreos', 'Marshmallows']
for one in flavors:
    for two in toppings:
        print(one, 'topped with', two)
Vanilla topped with Hot Fudge
Vanilla topped with Oreos
Vanilla topped with Marshmallows
Chocolate topped with Hot Fudge
Chocolate topped with Oreos
Chocolate topped with Marshmallows
Cookie Dough topped with Hot Fudge
Cookie Dough topped with Oreos
Cookie Dough topped with Marshmallows
```

While Loops in Python

```
While Loops
number = 0

while number < 5:
    print(number)
    number = number + 1
0
1
2
```

```
3
4
number = 0

while number < 5:
    print(number)
    if number == 3:
        break
    number = number + 1
0
1
2
3
number = 0

while number < 5:
    print(number)
    if number == 3:
        break
    number = number + 1
else:
    print('No longer < 5')
0
1
2
3
number = 0

while number < 5:
    number = number + 1
    if number == 3:
        continue
    print(number)
else:
    print('No longer < 5')
1
```

```
2
4
5
No longer < 5
```

Functions in Python

```
Functions
def first_func():
    print('We did it!')
first_func()
We did it!
def number_squared(number):
    print(number**2)
number_squared(5)
25
def number_squared_cust(number,power):
    print(number**power)
number_squared_cust(5,3)
125
args_tuple = (5,6,1,2,8)

def number_args(*number):
    print(number[0]*number[1])
number_args(*args_tuple)
30
def number_squared_cust(number,power):
    print(number**power)
number_squared_cust(power = 5,number = 3)
243
def number_kwarg(**number):
    print('My number is: ' + number['integer'] + 'My other numbe
number_kwarg(integer = '2309', integer2 = '349')
My number is: 2309My other number: 349
```

Converting Data Types

```
num_int = 7

type(num_int)
int
num_str = '7'

type(num_str)
str
nun_str_conv = int(num_str)

type(nun_str_conv)
int
num_sum = num_int + nun_str_conv

print(num_sum)
14
type(num_sum)
int
list_type = [1,2,3]

type(list_type)
list
type(tuple(list_type))
tuple
list_type = [1,2,3,3,2,1,2,3,2,1]
type(set(list_type))
set
dict_type = {'name': 'Alex','age': 28, 'hair': 'N/A'}

type(dict_type)
dict
dict_type.items()
dict_items([('name', 'Alex'), ('age', 28), ('hair', 'N/A')])
dict_type.values()
dict_values(['Alex', 28, 'N/A'])
```

```
dict_type.keys()
dict_keys(['name', 'age', 'hair'])
type(list(dict_type.keys()))
list
type(list(dict_type.values()))
list
long_str = "I like to party"

set(long_str)
{' ', 'I', 'a', 'e', 'i', 'k', 'l', 'o', 'p', 'r', 't', 'y'}
```

BMI Calculator

```
name = input("Enter you name: ")

weight = int(input("Enter your weight in pounds: "))

height = int(input("Enter your height in inches: "))

BMI = (weight * 703) / (height * height)

print(BMI)

if BMI>0:
    if(BMI<18.5):
        print(name +", you are underweight.")
    elif (BMI<=24.9):
        print(name +", you are normal weight.")
    elif (BMI<29.9):
        print(name +", you are overweight. You need to exercise
    elif (BMI<34.9):
        print(name +", you are obese.")
    elif (BMI<39.9):
        print(name +", you are severely obese.")
    else:
        print(name +", you are morbidly obese.")
```

```
else:
    print("Enter valid input")
```

```
Enter you name: Alex
Enter your weight in pounds: 170
Enter your height in inches: 69
25.101869355177485
Alex, you are overweight. You need to exercise more and stop
sitting and writing so many python tutorials
```

Building an Automated File Sorter in File Explorer using Python

shutil is called "shell utilities" because it allows you to perform various file-related tasks that are similar to the commands that you might use in a Unix shell, such as copying, moving, renaming, and deleting files and directories.

```
import os, shutil

path = r"C:/Users/User/Documents/FileSorter/"

file_name=os.listdir(path)

folder_names=['csv files','image files','pdf files']
for loop in range(0,3):
    if not os.path.exists(path+ folder_names[loop]):
         os.makedirs((path+ folder_names[loop]))

for file in file_name:
    if ".csv" in file and not os.path.exists(path+ "csv files/"
        shutil.move(path + file, path + "csv files/"+ file)
    if ".pdf" in file and not os.path.exists(path+ "pdf files/"
        shutil.move(path + file, path + "pdf files/"+ file)
    if ".jpg" in file and not os.path.exists(path+ "image files/
        shutil.move(path + file, path + "image files/"+ file)
```