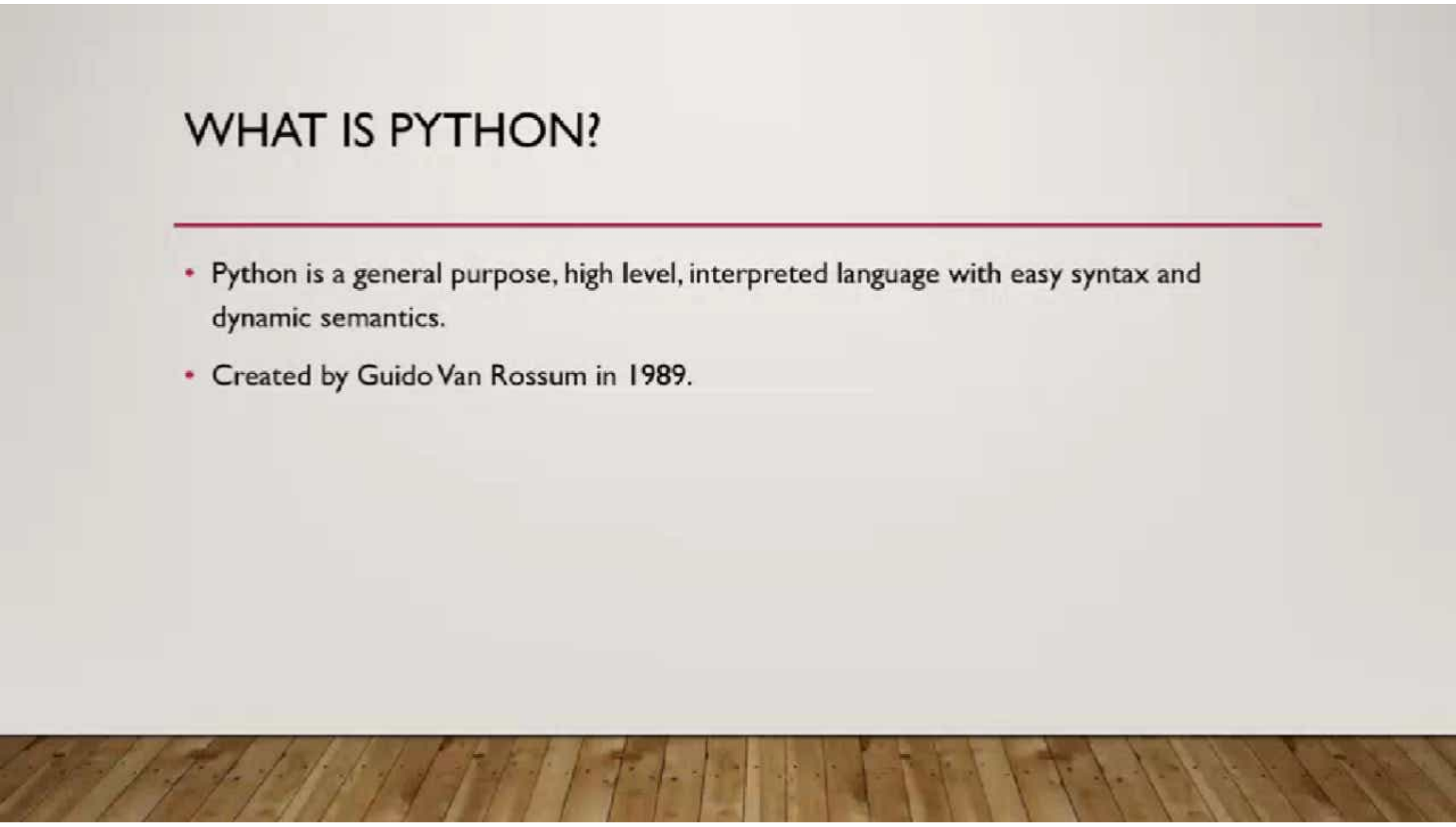
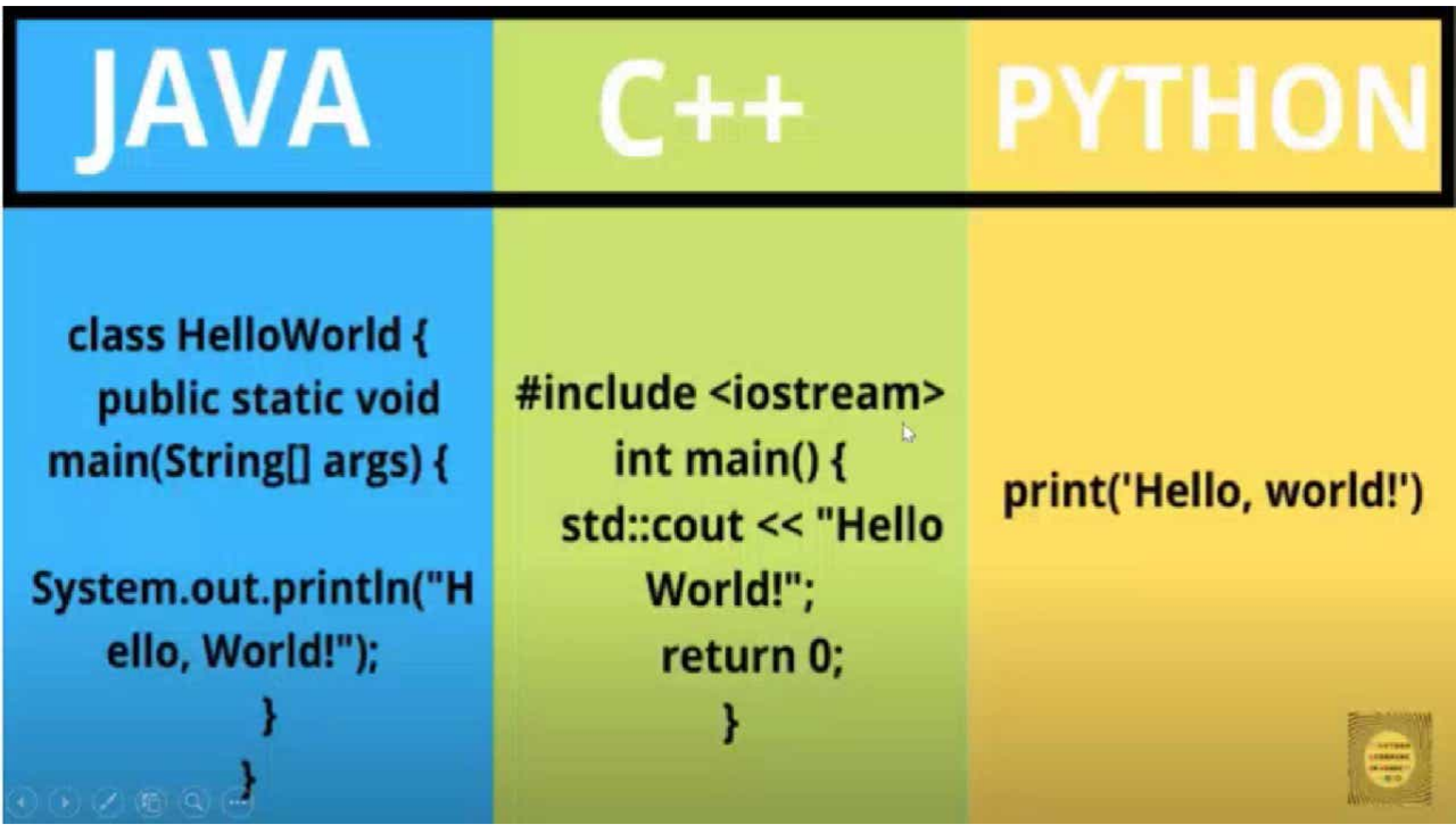
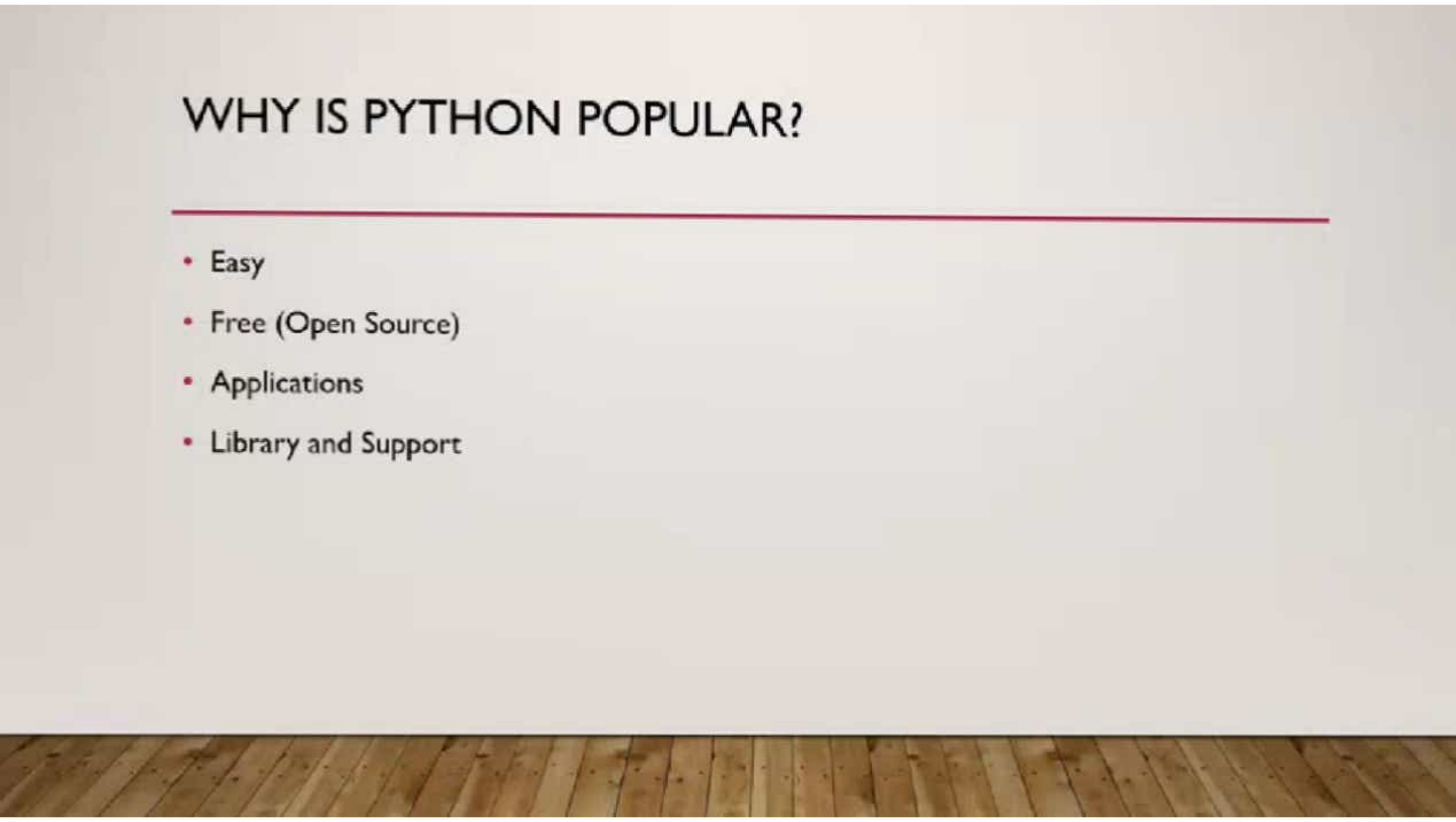


WHAT IS PYTHON?

- Python is a general purpose, high level, interpreted language with easy syntax and dynamic semantics.
- Created by Guido Van Rossum in 1989.



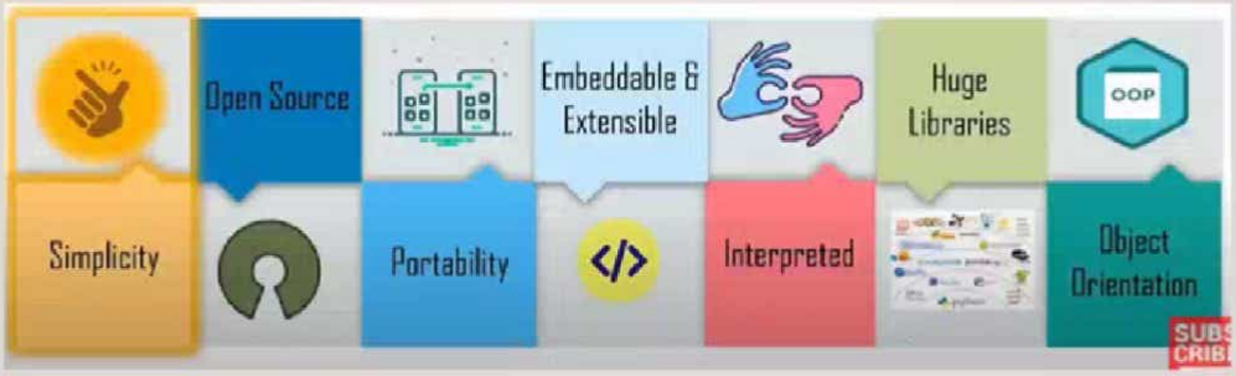
JAVA	C++	PYTHON
class HelloWorld { public static void main(String[] args) { System.out.println("Hello, World!"); } }	#include <iostream> int main() { std::cout << "Hello World!"; return 0; }	print('Hello, world!')



WHY IS PYTHON POPULAR?

- Easy
- Free (Open Source)
- Applications
- Library and Support

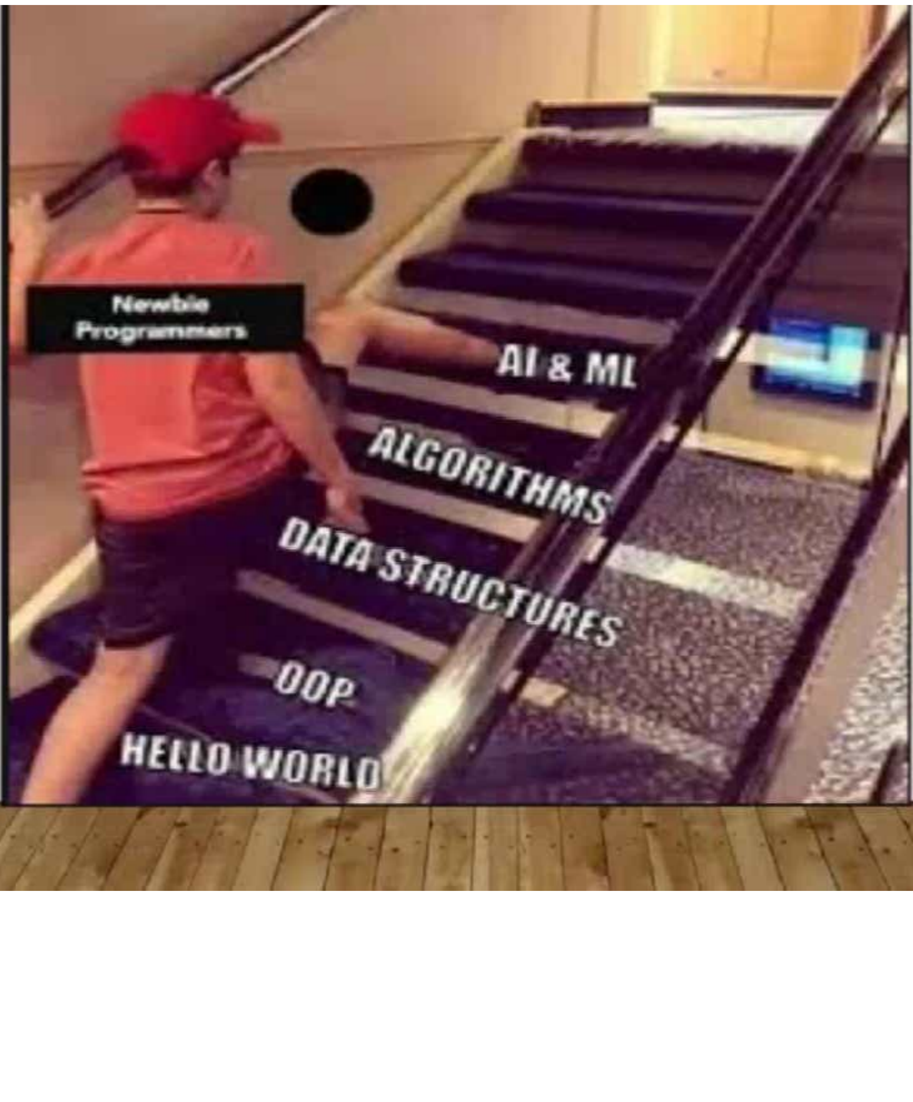
FEATURES OF PYTHON



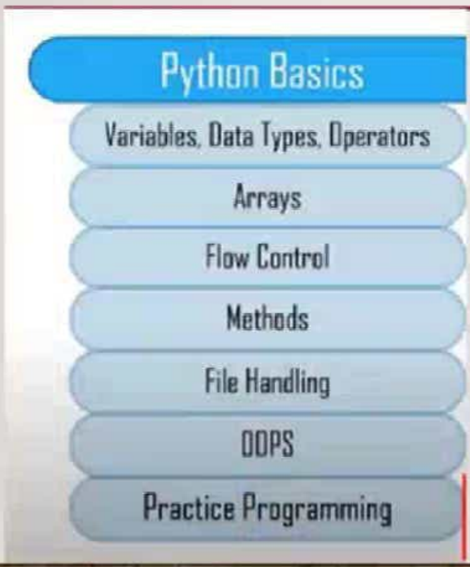
WHERE IS PYTHON USED IN INDUSTRY?

Google Dropbox NETFLIX

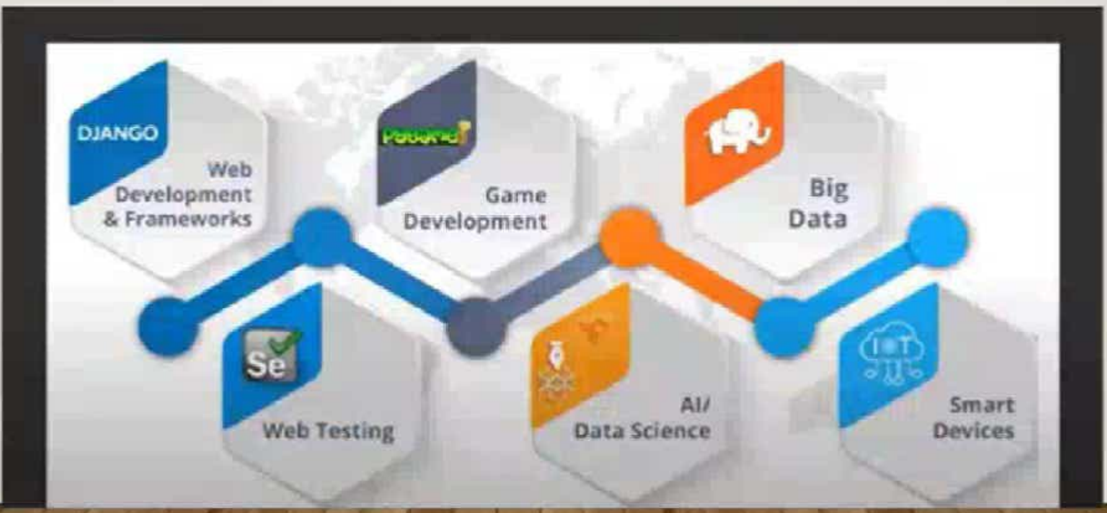
BitTorrent NASA

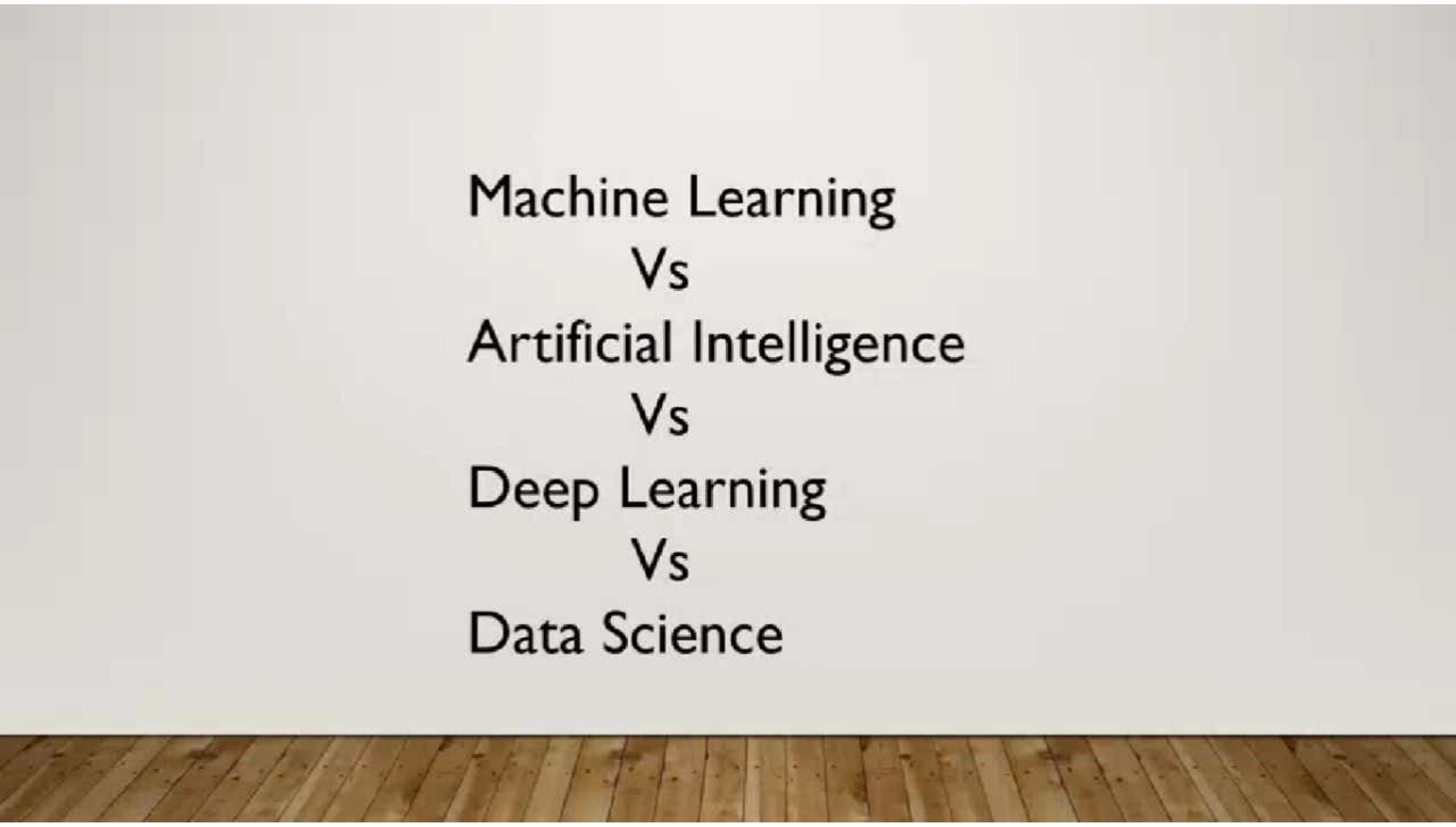

Newbie Programmers
AI & ML
ALGORITHMS
DATA STRUCTURES
OOP
HELLO WORLD

LEARNING PATH



CAREER OPPORTUNITIES





Machine Learning

Vs

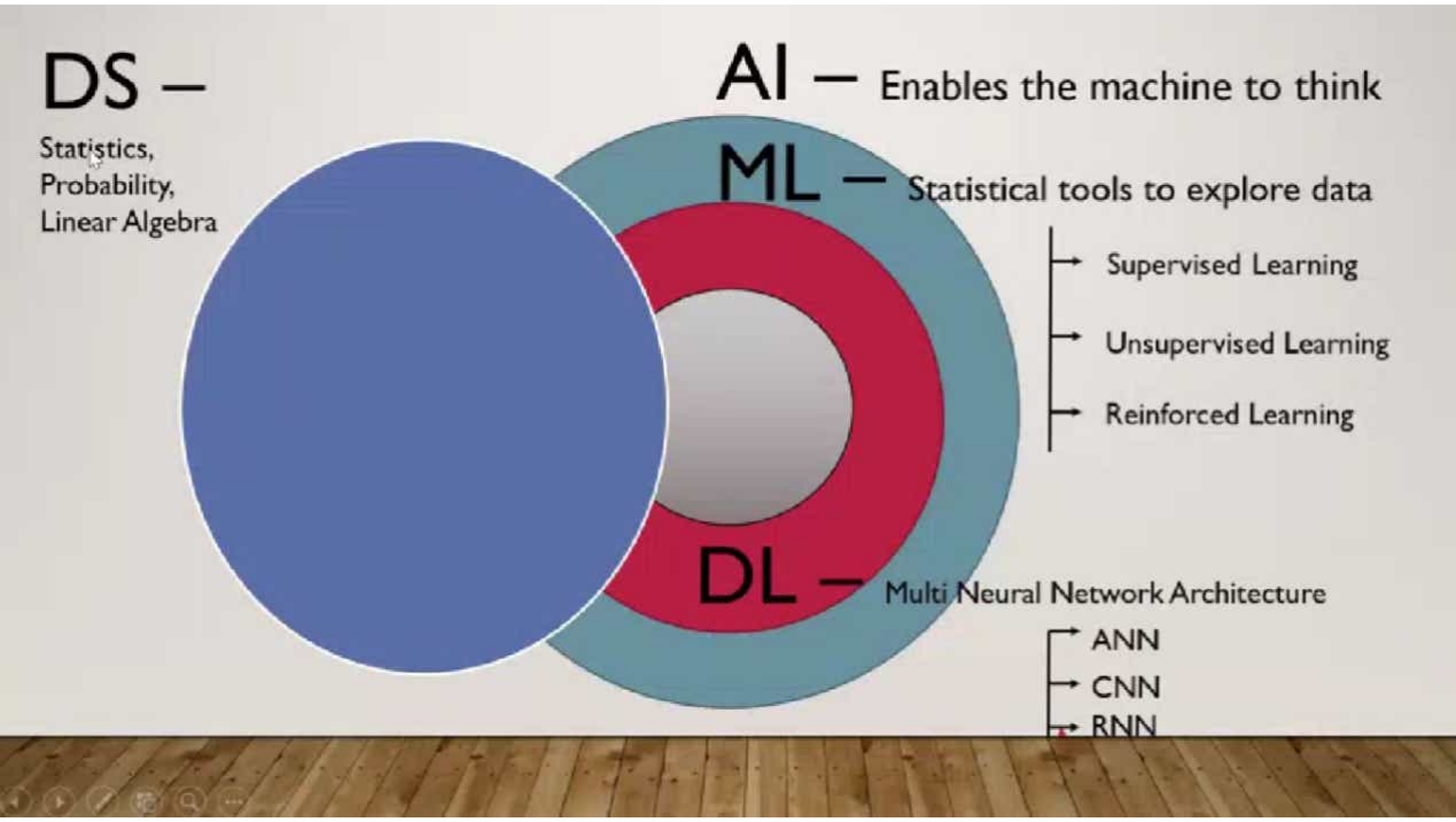
Artificial Intelligence

Vs

Deep Learning

Vs

Data Science



DS –

Statistics,
Probability,
Linear Algebra

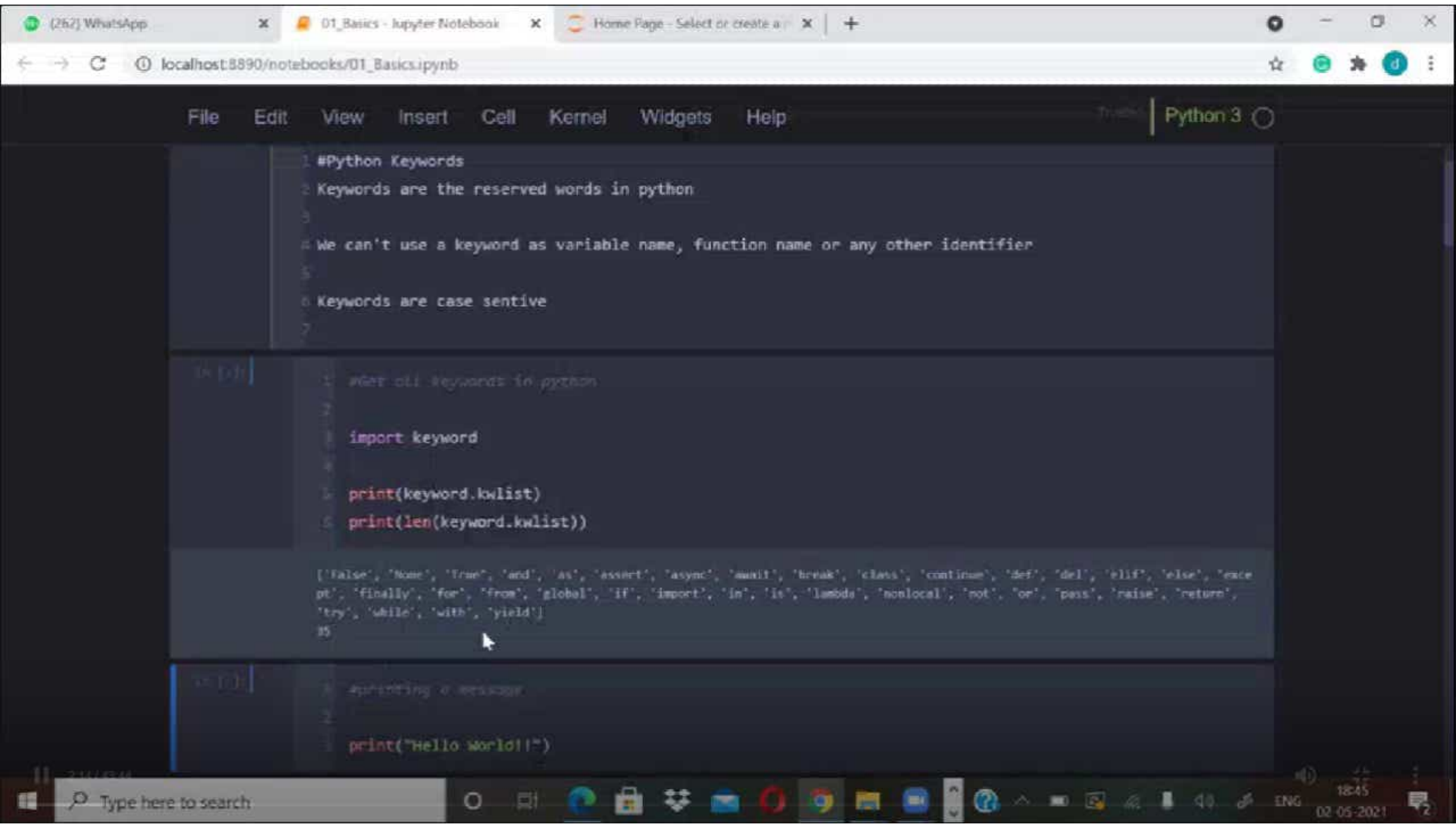
AI – Enables the machine to think

ML – Statistical tools to explore data

- Supervised Learning
- Unsupervised Learning
- Reinforced Learning

DL – Multi Neural Network Architecture

- ANN
- CNN
- RNN

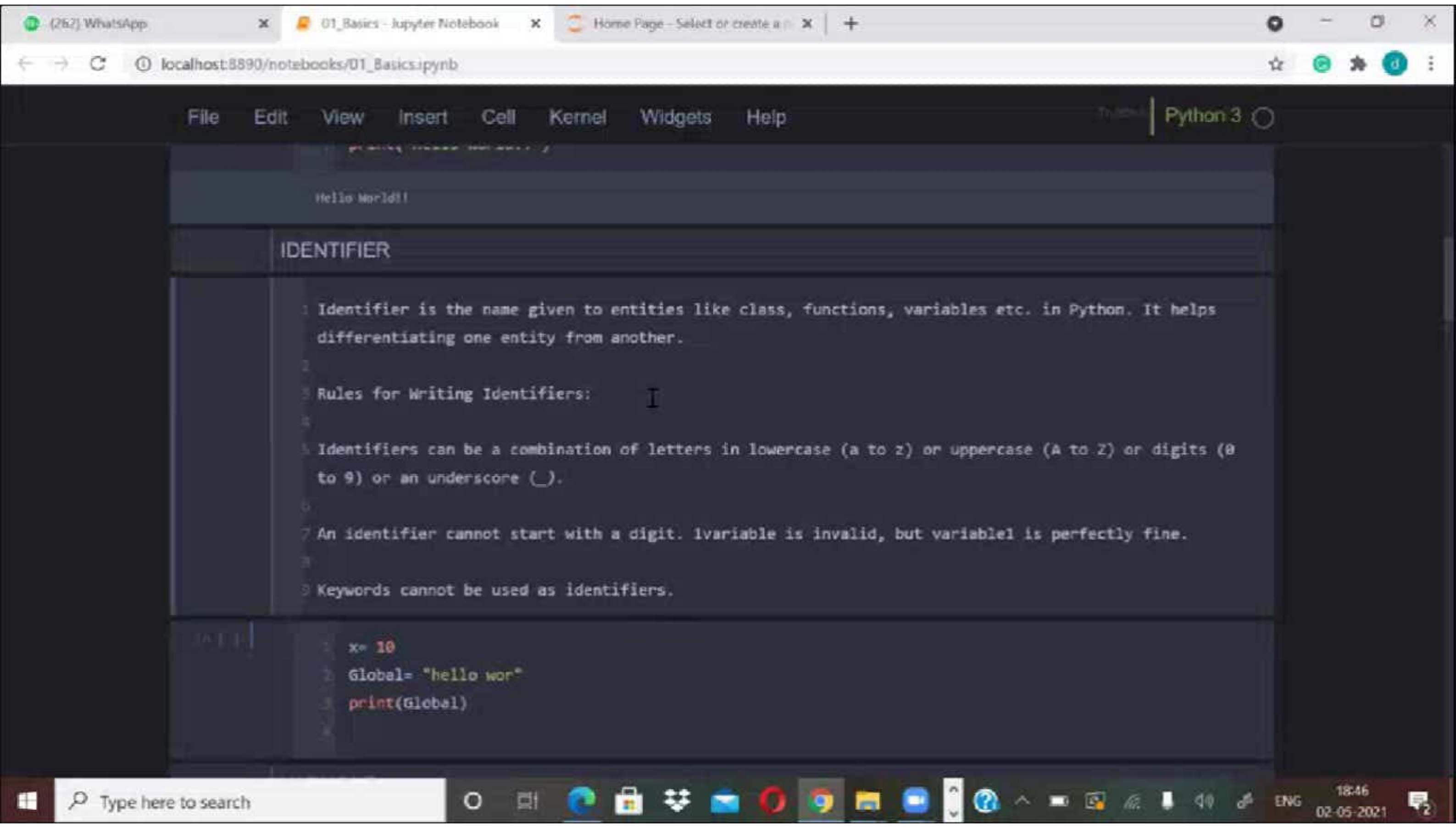

(262) WhatsApp
01_Basics - Jupyter Notebook
Home Page - Select or create a
localhost:8890/notebooks/01_Basics.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3

```
#Python Keywords
Keywords are the reserved words in python

We can't use a keyword as variable name, function name or any other identifier

Keywords are case sentive

```

In [ ]:
```
#Get all keywords in python

import keyword

print(keyword.kwlist)
print(len(keyword.kwlist))
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'exce
pt', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return',
'try', 'while', 'with', 'yield']
35
```

In [ ]:
```
#printing a message

print("Hello World!!")
```


(262) WhatsApp
01_Basics - Jupyter Notebook
Home Page - Select or create a
localhost:8890/notebooks/01_Basics.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
Type here to search
ENG
18:46
02-05-2021

```
Hello World!!
```

IDENTIFIER

Identifier is the name given to entities like class, functions, variables etc. in Python. It helps differentiating one entity from another.

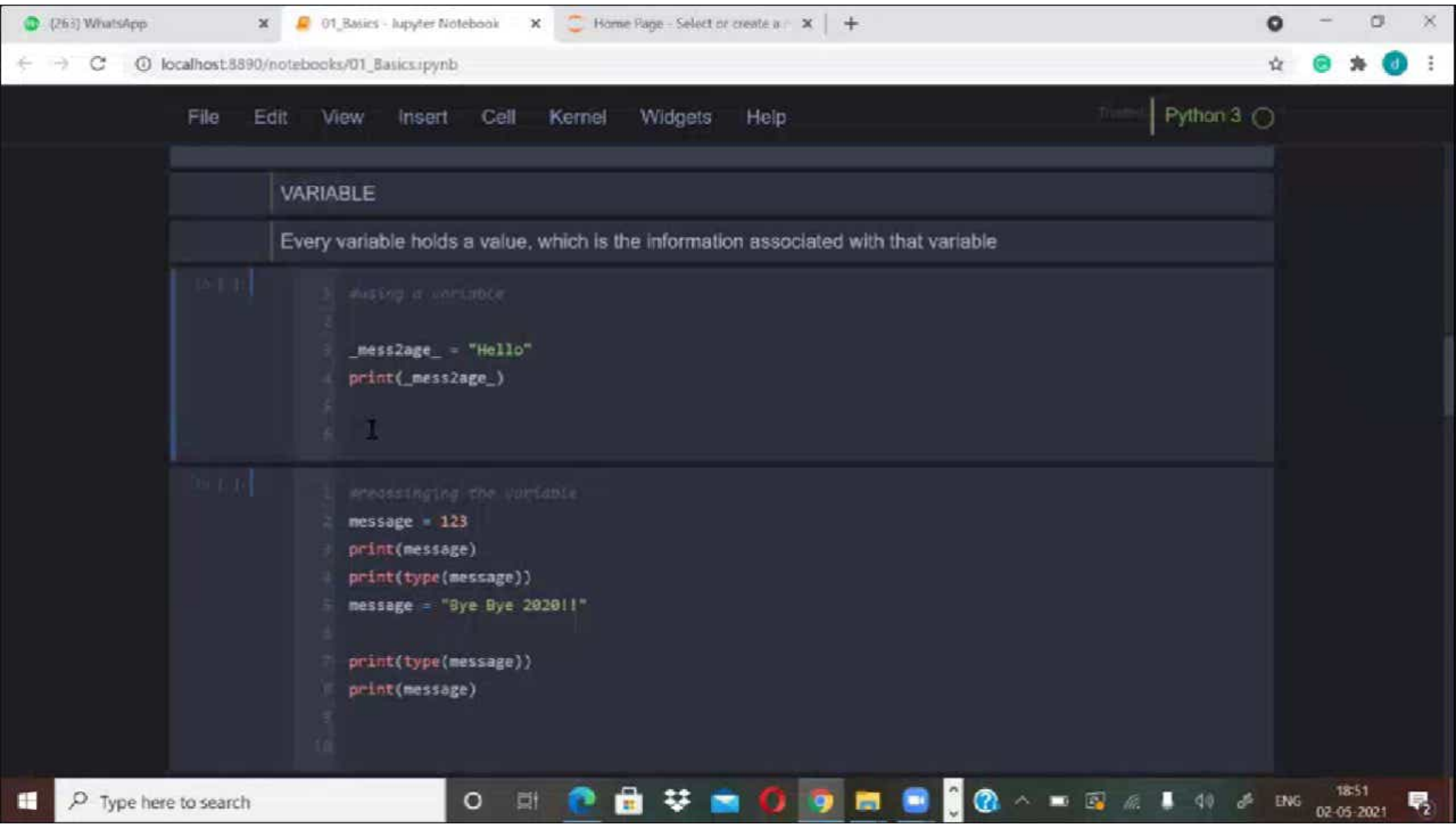
Rules for Writing Identifiers:

Identifiers can be a combination of letters in lowercase (a to z) or uppercase (A to Z) or digits (0 to 9) or an underscore (_).

An identifier cannot start with a digit. 1variable is invalid, but variable1 is perfectly fine.

Keywords cannot be used as identifiers.

```
x= 10
Global= "hello wor"
print(Global)
```

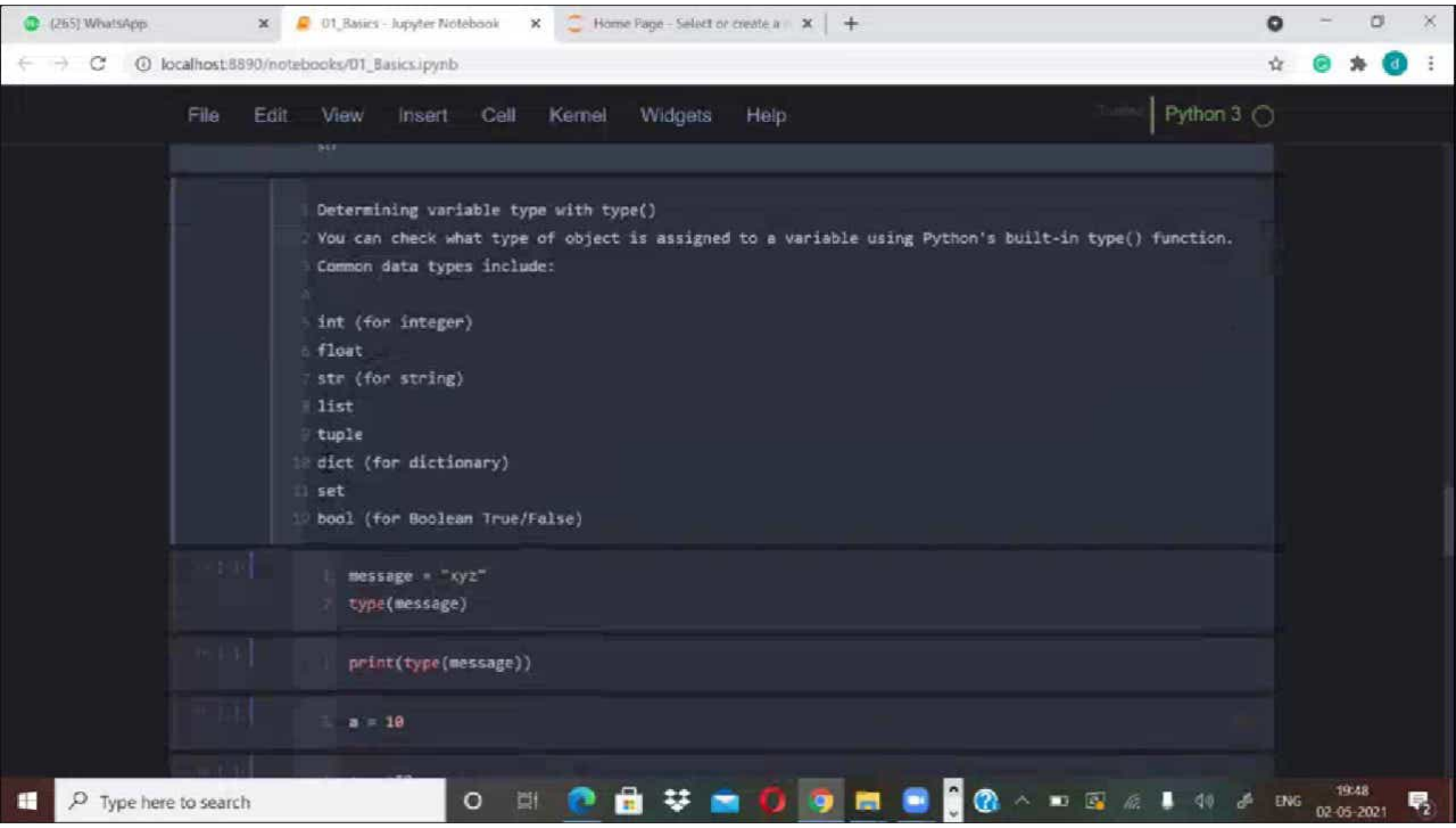

(263) WhatsApp
01_Basics - Jupyter Notebook
Home Page - Select or create a
localhost:8890/notebooks/01_Basics.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Type here to search
ENG
18:51
02-05-2021

VARIABLE

Every variable holds a value, which is the information associated with that variable

```
#using a variable

_mess2age_ = "Hello"
print(_mess2age_)


```

```
#reassinging the variable
message = 123
print(message)
print(type(message))
message = "Bye Bye 2020!!"

print(type(message))
print(message)
```

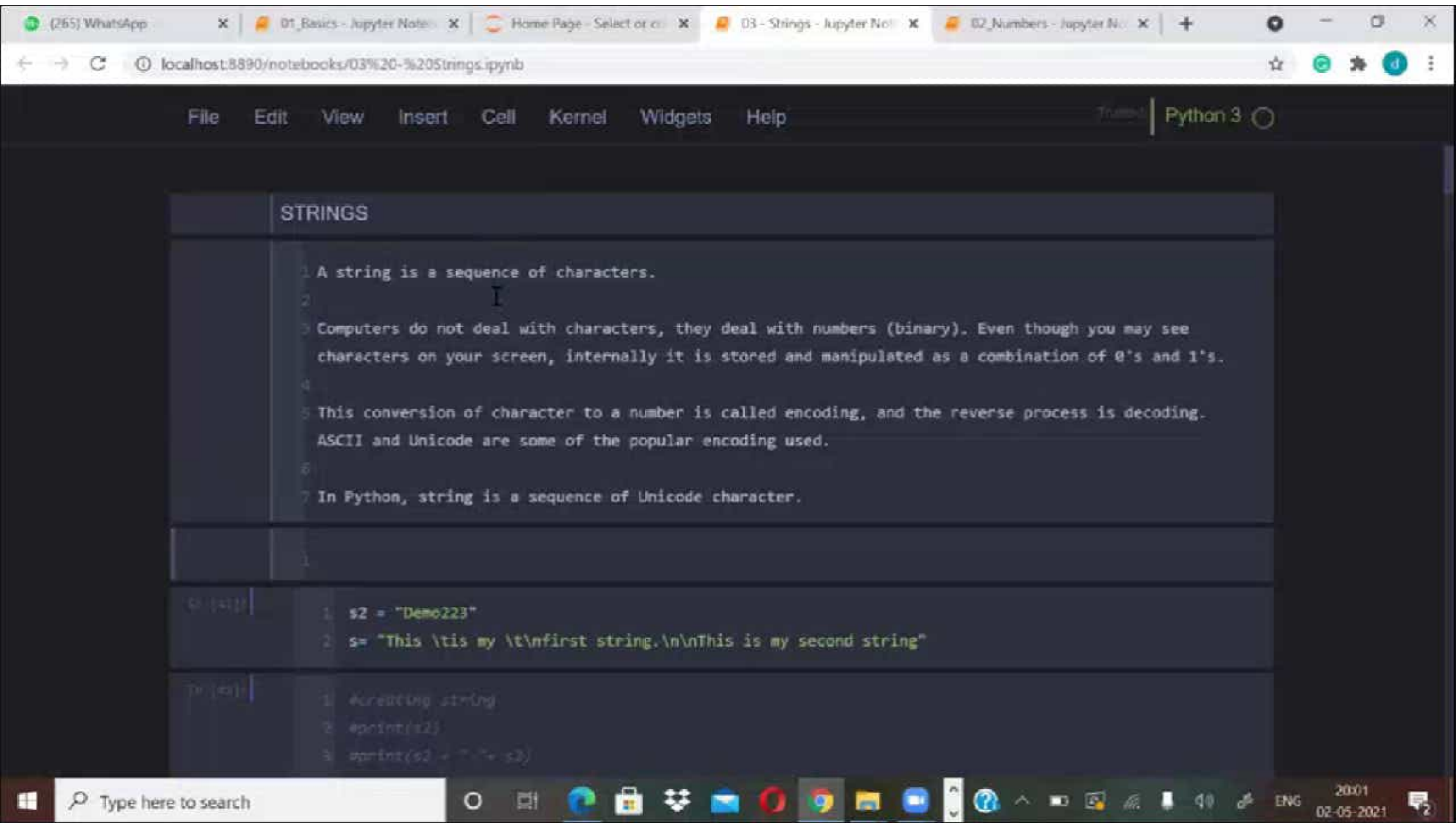

(265) WhatsApp
01_Basics - Jupyter Notebook
Home Page - Select or create a
localhost:8890/notebooks/01_Basics.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Type here to search
ENG
19:48
02-05-2021

```
Determining variable type with type()
You can check what type of object is assigned to a variable using Python's built-in type() function.
Common data types include:

int (for integer)
float
str (for string)
list
tuple
dict (for dictionary)
set
bool (for Boolean True/False)
```

```
message = "xyz"
type(message)
```

```
print(type(message))
```

```
a = 10
```


localhost:8890/notebooks/03%20-%20Strings.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3

STRINGS

A string is a sequence of characters.

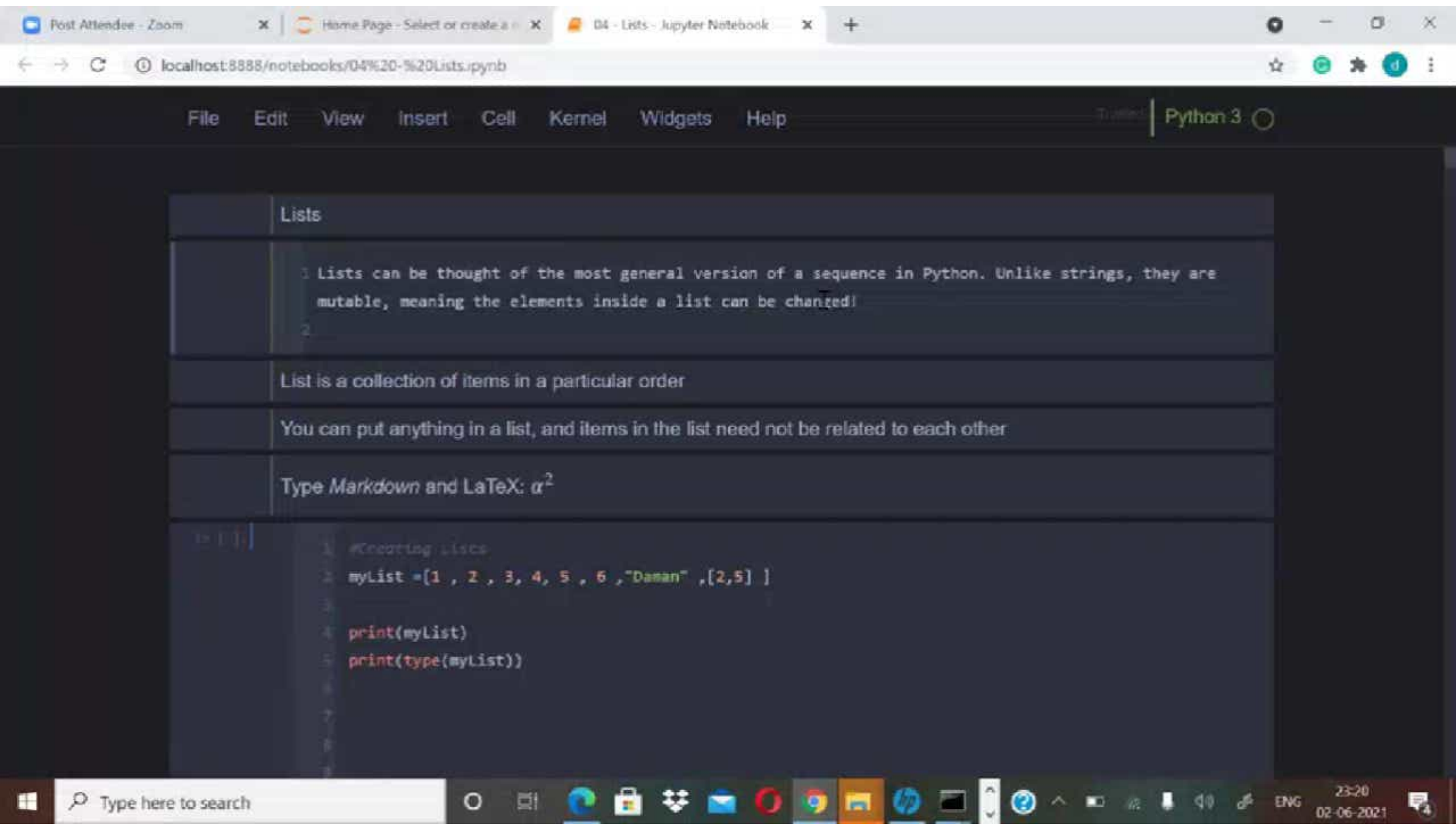
Computers do not deal with characters, they deal with numbers (binary). Even though you may see characters on your screen, internally it is stored and manipulated as a combination of 0's and 1's.

This conversion of character to a number is called encoding, and the reverse process is decoding. ASCII and Unicode are some of the popular encoding used.

In Python, string is a sequence of Unicode character.

```
s2 = "Demo223"
s= "This \tis my \t\nfirst string.\n\nThis is my second string"
```

```
#creating string
#print(s2)
#print(s2 + " "+ s2)
```


Post Attendee - Zoom
Home Page - Select or create a
04 - Lists - Jupyter Notebook
localhost:8888/notebooks/04%20-%20Lists.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Type here to search
ENG
23:20
02-06-2021

Lists

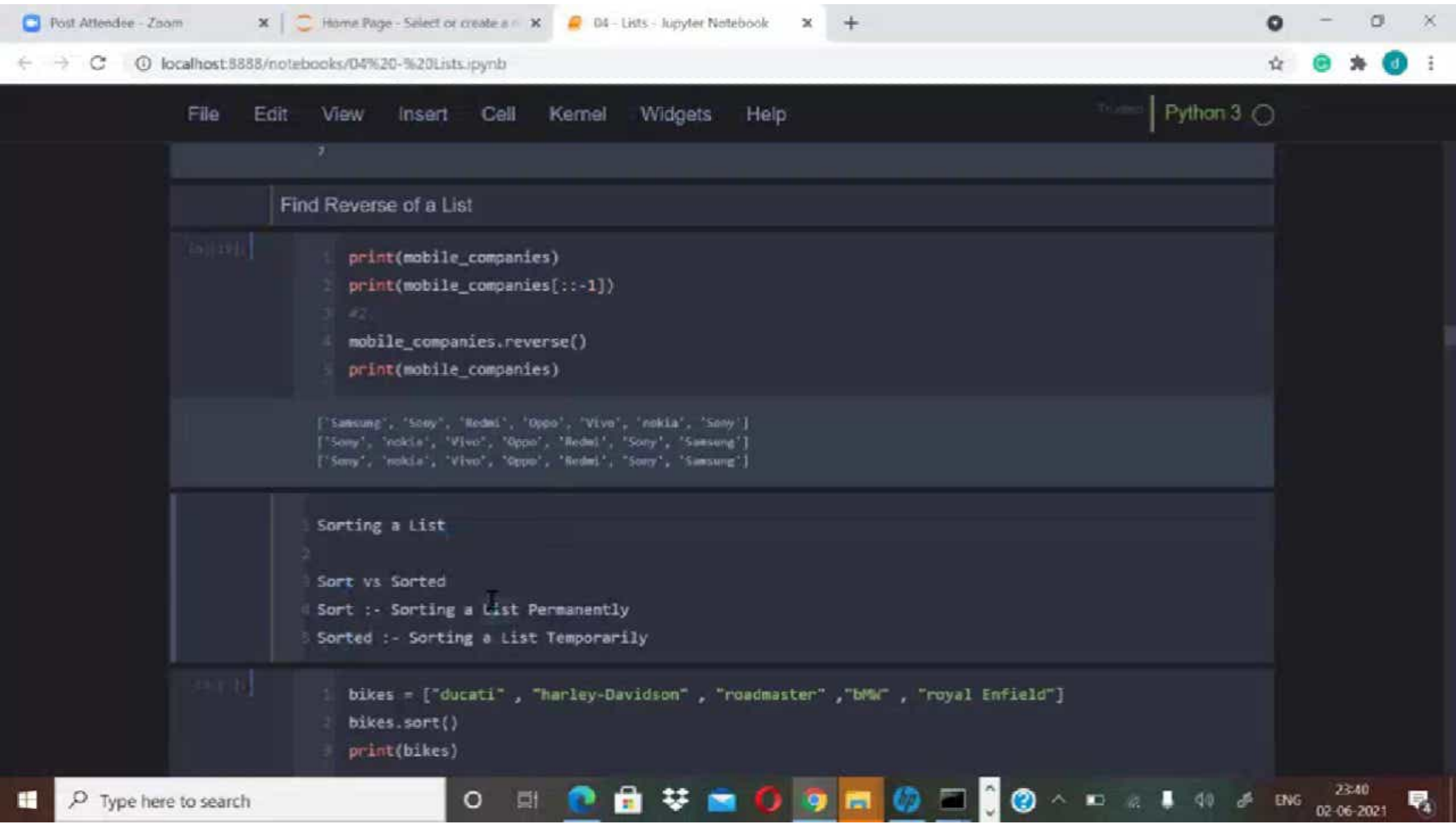
```
Lists can be thought of the most general version of a sequence in Python. Unlike strings, they are mutable, meaning the elements inside a list can be changed!
```

List is a collection of items in a particular order

You can put anything in a list, and items in the list need not be related to each other

Type *Markdown* and LaTeX: α^2

```
#Creating Lists
myList =[1 , 2 , 3, 4, 5 , 6 ,"Daman" ,[2,5] ]

print(myList)
print(type(myList))
```


Post Attendee - Zoom
Home Page - Select or create a n
04 - Lists - Jupyter Notebook
localhost:8888/notebooks/04%20-%20Lists.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Type here to search
ENG
23:40
02-06-2021

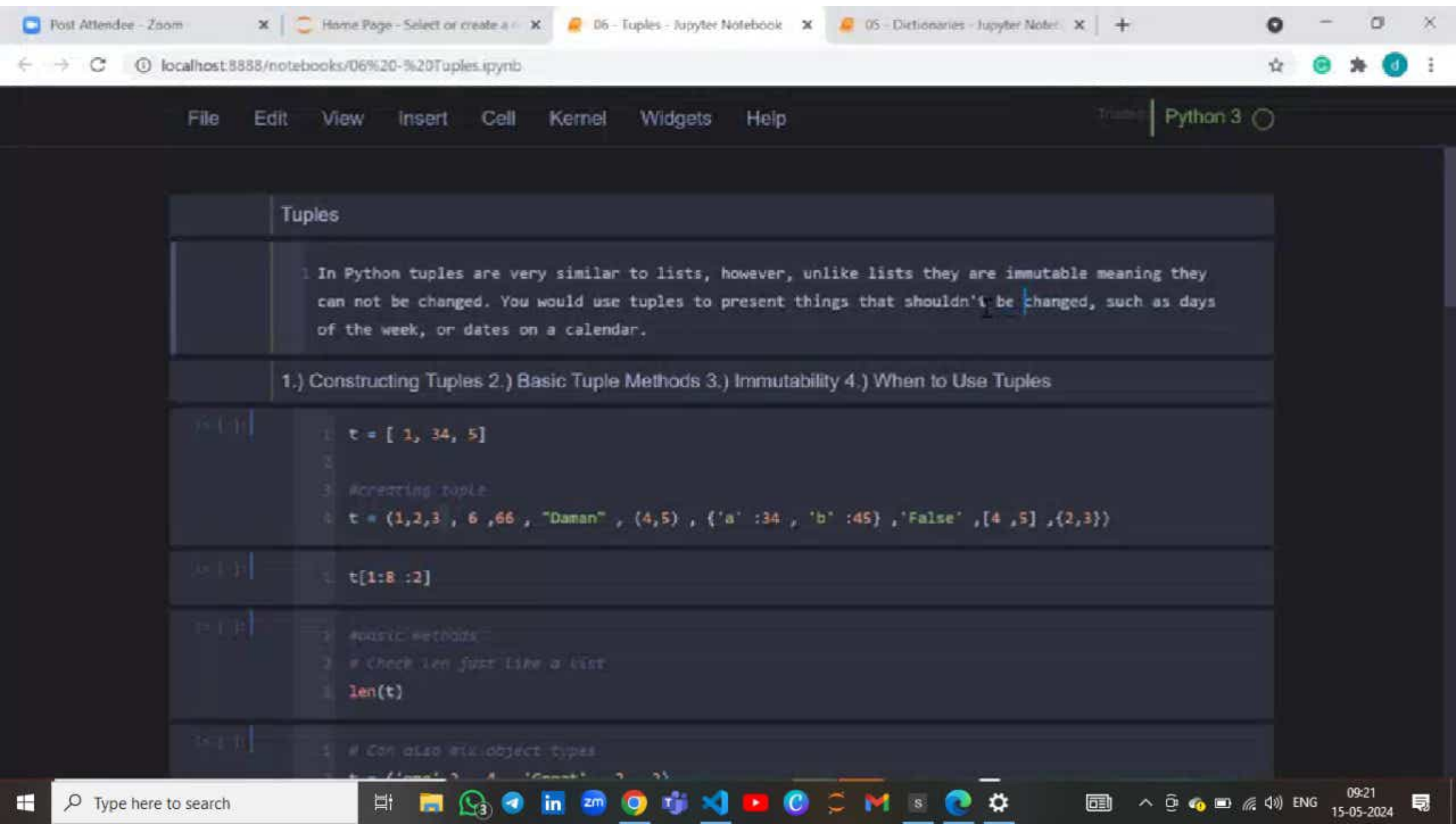
Find Reverse of a List

```python
print(mobile_companies)
print(mobile_companies[::-1])
#2
mobile_companies.reverse()
print(mobile_companies)
```

```
['Samsung', 'Sony', 'Redmi', 'Oppo', 'Vivo', 'nokia', 'Sony']
['Sony', 'nokia', 'Vivo', 'Oppo', 'Redmi', 'Sony', 'Samsung']
['Sony', 'nokia', 'Vivo', 'Oppo', 'Redmi', 'Sony', 'Samsung']
```

```
Sorting a List

Sort vs Sorted
Sort :- Sorting a List Permanently
Sorted :- Sorting a List Temporarily
```

```python
bikes = ["ducati" , "harley-Davidson" , "roadmaster" ,"bMW" , "royal Enfield"]
bikes.sort()
print(bikes)
```



Post Attendee - Zoom
Home Page - Select or create a
06 - Tuples - Jupyter Notebook
05 - Dictionaries - Jupyter Note
localhost:8888/notebooks/06%20-%20Tuples.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Type here to search
ENG
09:21
15-05-2024

Tuples

In Python tuples are very similar to lists, however, unlike lists they are immutable meaning they can not be changed. You would use tuples to present things that shouldn't be changed, such as days of the week, or dates on a calendar.

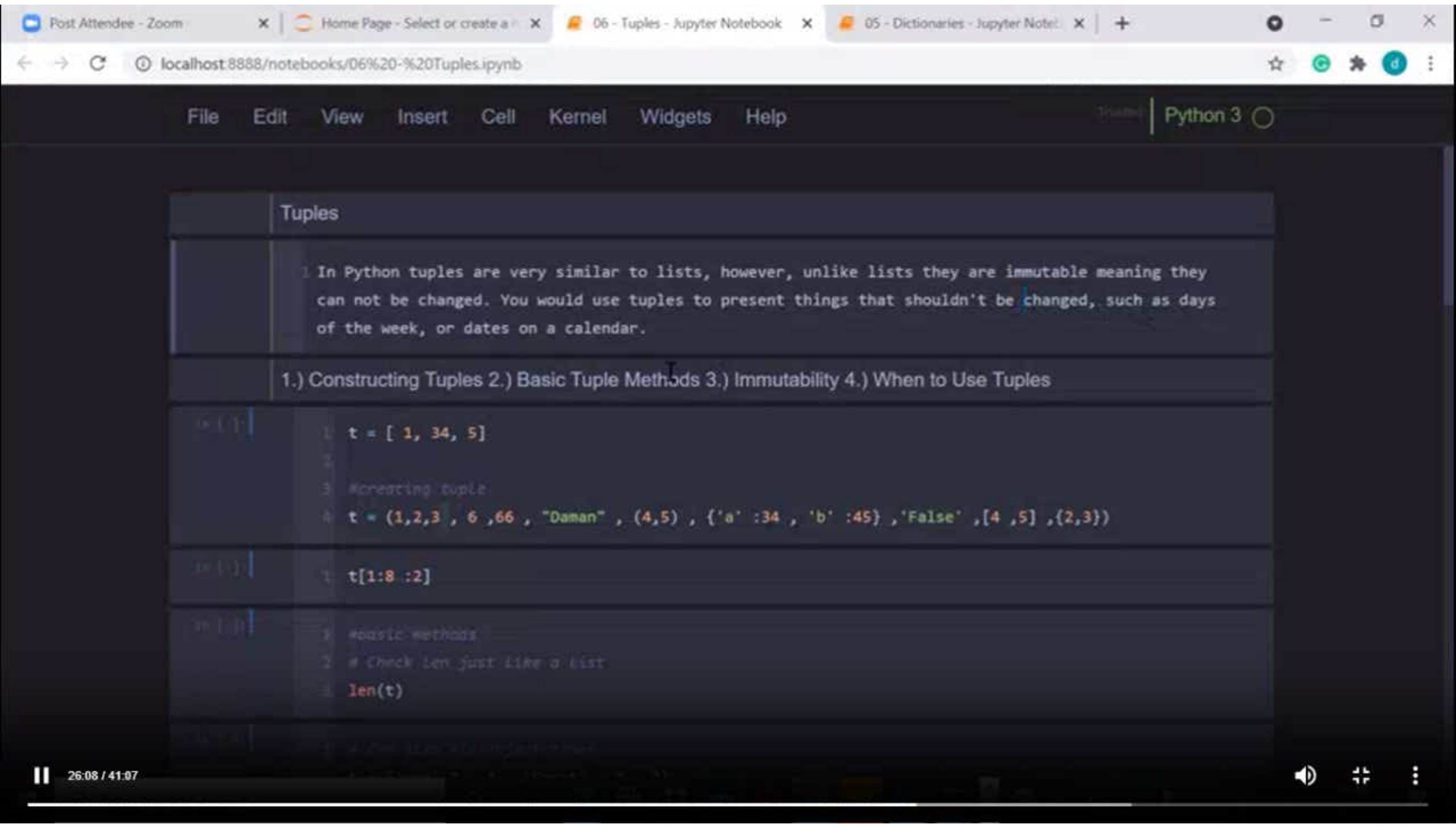
1.) Constructing Tuples 2.) Basic Tuple Methods 3.) Immutability 4.) When to Use Tuples

```
t = [ 1, 34, 5]

#creating tuple
t = (1,2,3 , 6 ,66 , "Daman" , (4,5) , {'a' :34 , 'b' :45} ,'False' ,[4 ,5] ,{2,3})
```

```
t[1:8 :2]
```

```
#basic methods
# Check len just like a list
len(t)
```

```
# Can also mix object types
```


Post Attendee - Zoom
Home Page - Select or create a
06 - Tuples - Jupyter Notebook
05 - Dictionaries - Jupyter Note
localhost:8888/notebooks/06%20-%20Tuples.ipynb
File Edit View Insert Cell Kernel Widgets Help
Python 3
26:08 / 41:07

Tuples

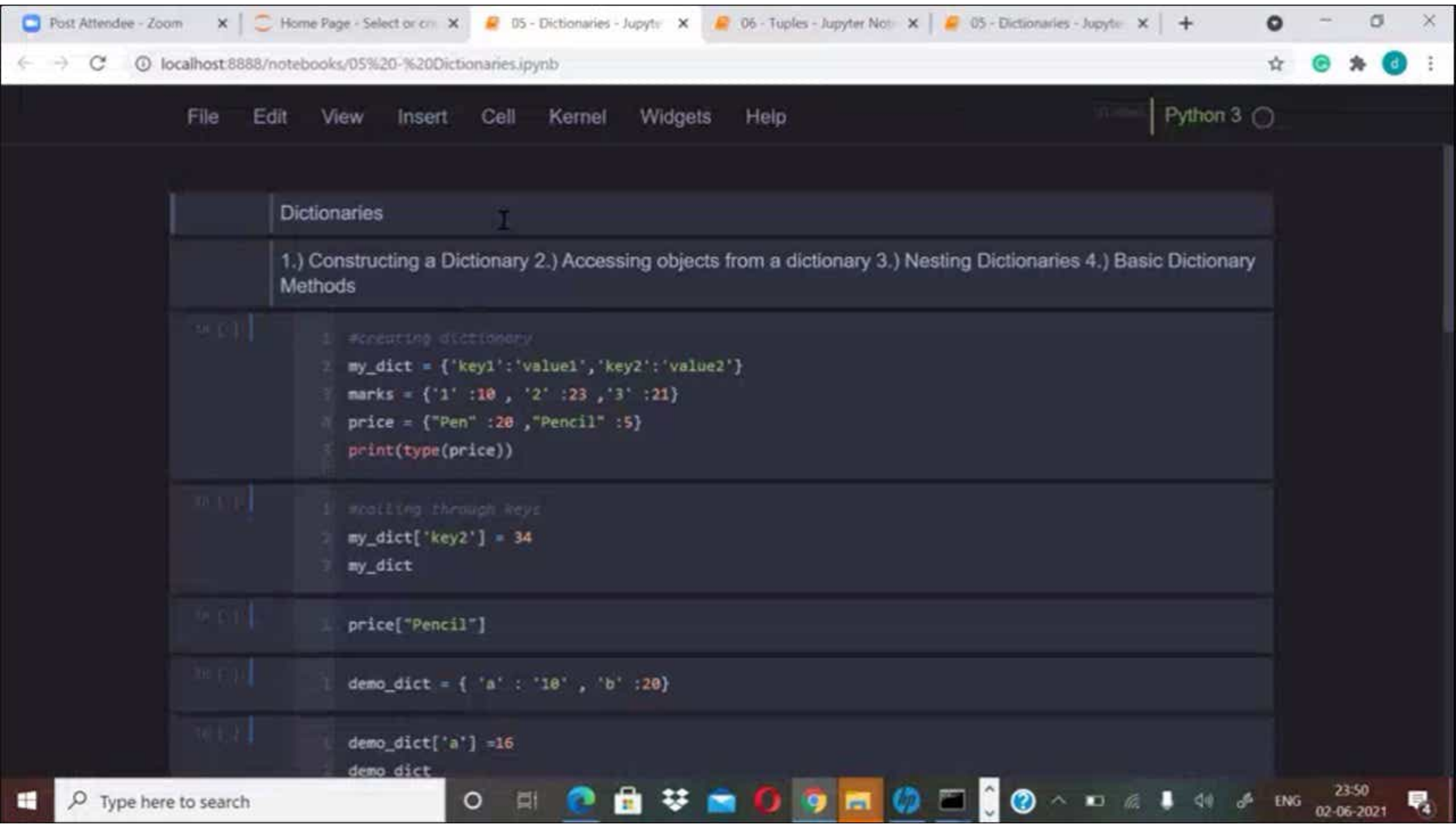
```
In Python tuples are very similar to lists, however, unlike lists they are immutable meaning they
can not be changed. You would use tuples to present things that shouldn't be changed, such as days
of the week, or dates on a calendar.
```

1.) Constructing Tuples 2.) Basic Tuple Methods 3.) Immutability 4.) When to Use Tuples

```
t = [ 1, 34, 5]

#creating tuple
t = (1,2,3 , 6 ,66 , "Daman" , (4,5) , {'a' :34 , 'b' :45} ,'False' ,[4 ,5] ,{2,3})
```

```
t[1:8 :2]
```

```
#basic methods
# Check len just like a list
len(t)
```


Post Attendee - Zoom
Home Page - Select or cr
05 - Dictionaries - Jupyter
06 - Tuples - Jupyter Note
05 - Dictionaries - Jupyter
localhost:8888/notebooks/05%20-%20Dictionaries.ipynb
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Python 3
Type here to search
ENG
23:50
02-06-2021

Dictionaries

1.) Constructing a Dictionary 2.) Accessing objects from a dictionary 3.) Nesting Dictionaries 4.) Basic Dictionary Methods

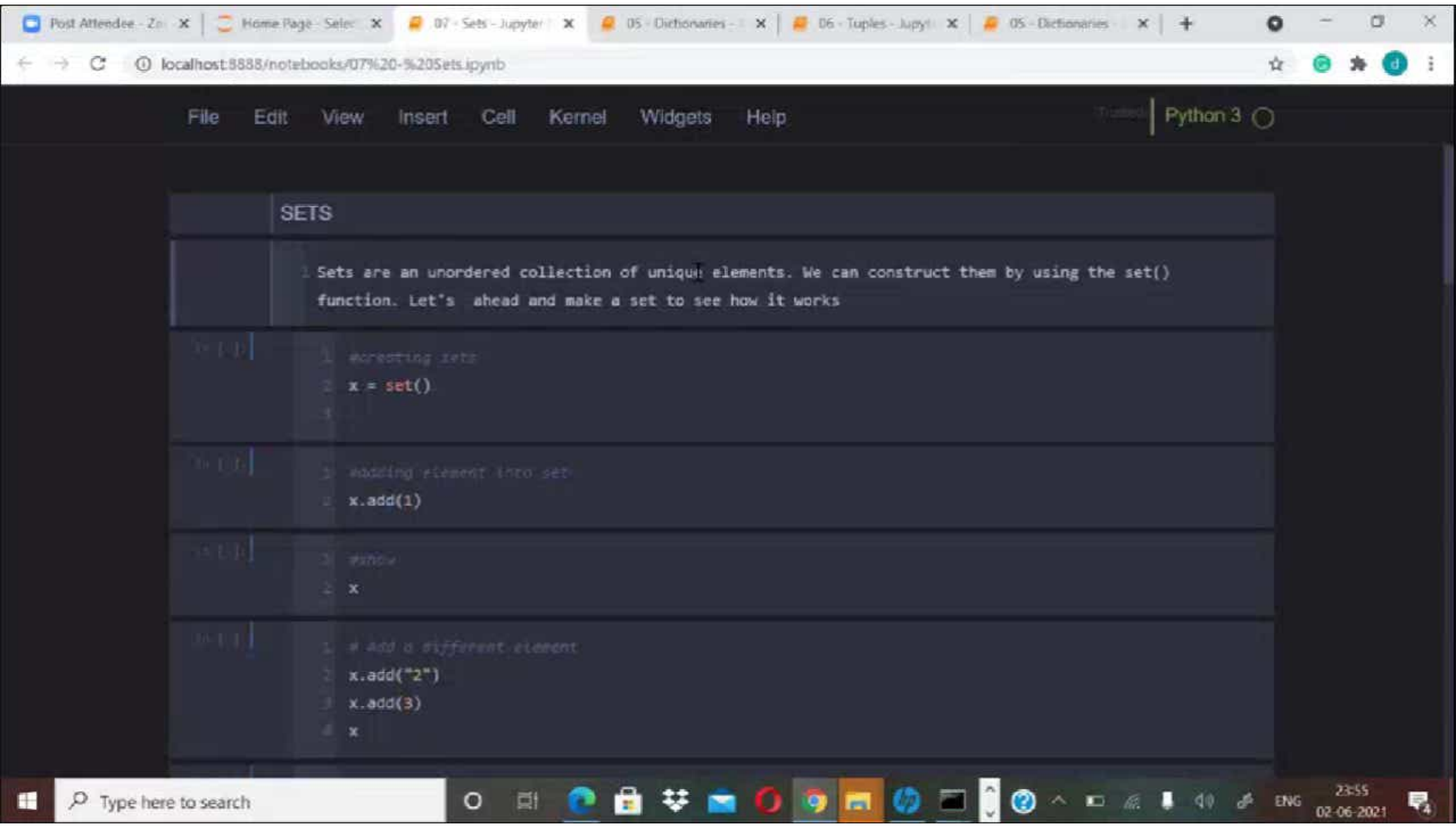
```
#creating dictionary
my_dict = {'key1':'value1','key2':'value2'}
marks = {'1' :10 , '2' :23 ,'3' :21}
price = {"Pen" :20 ,"Pencil" :5}
print(type(price))
```

```
#calling through keys
my_dict['key2'] = 34
my_dict
```

```
price["Pencil"]
```

```
demo_dict = { 'a' : '10' , 'b' :20}
```

```
demo_dict['a'] =16
demo_dict
```



SETS

Sets are an unordered collection of unique elements. We can construct them by using the set() function. Let's ahead and make a set to see how it works

```
#creating sets
x = set()

```

```
#adding element into set
x.add(1)
```

```
#show
x
```

```
# Add a different element
x.add("2")
x.add(3)
x
```