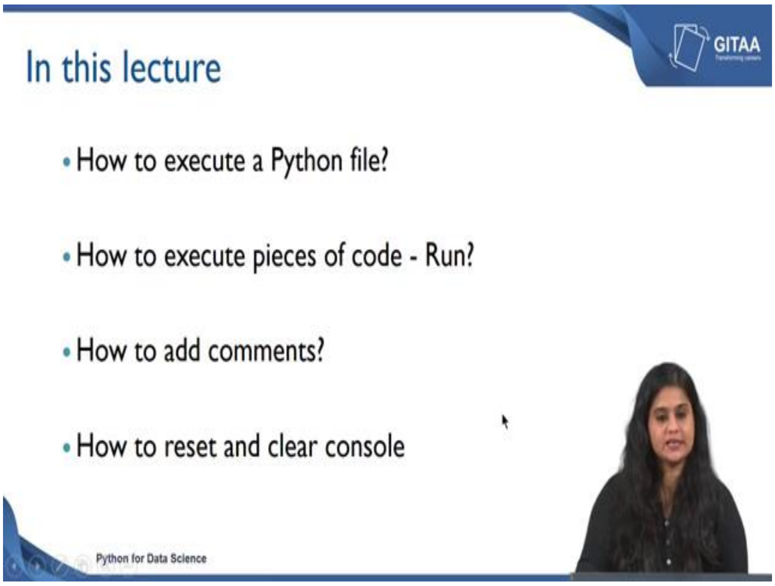
Python for Data Science

Prof. Ragunathan Rengasamy
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture – 04
Introduction to Spyder Part -2

Welcome to the 2nd lecture on Introduction to Spyder.

(Refer Slide Time: 00:17)



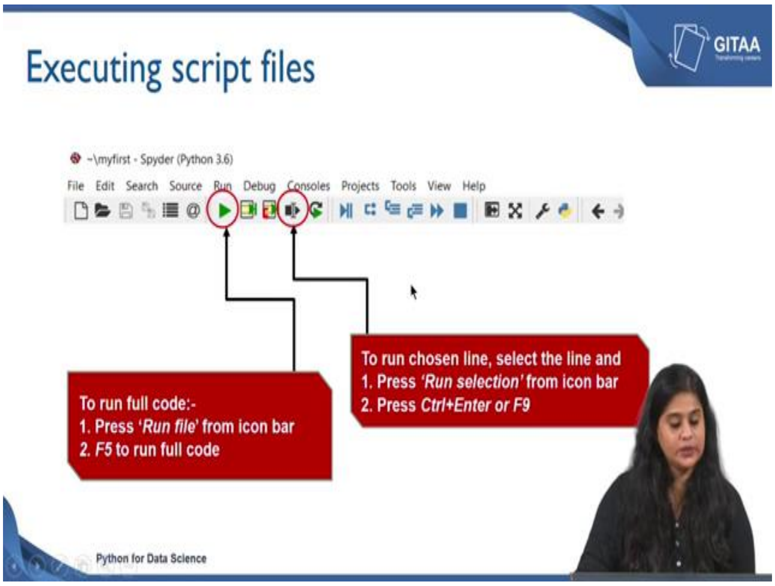
In this lecture

- How to execute a Python file?
- How to execute pieces of code - Run?
- How to add comments?
- How to reset and clear console

GITAA
Python for Data Science

In this lecture, we are going to see how to execute a Python file, how to execute few pieces of code using run and how to add comments and we will also see how to reset the environment and clear the console.

(Refer Slide Time: 00:34)



Executing script files

~\myfirst - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
GITAA
Python for Data Science

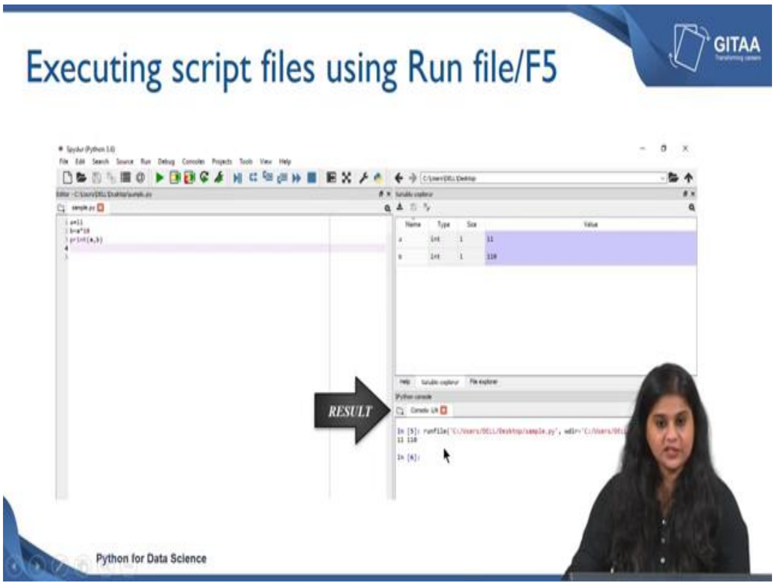
To run full code:-
1. Press '*Run file*' from icon bar
2. *F5* to run full code

To run chosen line, select the line and
1. Press '*Run selection*' from icon bar
2. Press *Ctrl+Enter or F9*

So, let us begin with file execution. On the top in the icon bar you will see a green triangle with its end pointed to right; now this is called the run file option and this will help you run an entire file at once and equivalent shortcut from the keyboard to press F5.

Now, if you want run few section of code or few lines of code, then you can click on the run selection option and this will help you run a chosen line. And equivalent shortcut from the keyboard is to either press F9 or press 'Ctrl+Enter'; after choosing the line. So, let us see how each of these options work and what are the corresponding outputs that they give?

(Refer Slide Time: 01:15)



Executing script files using Run file/F5

```
a=11
b=a*10
print(a,b)
```

Name	Type	Size	Value
a	int	1	11
b	int	1	110

RESULT

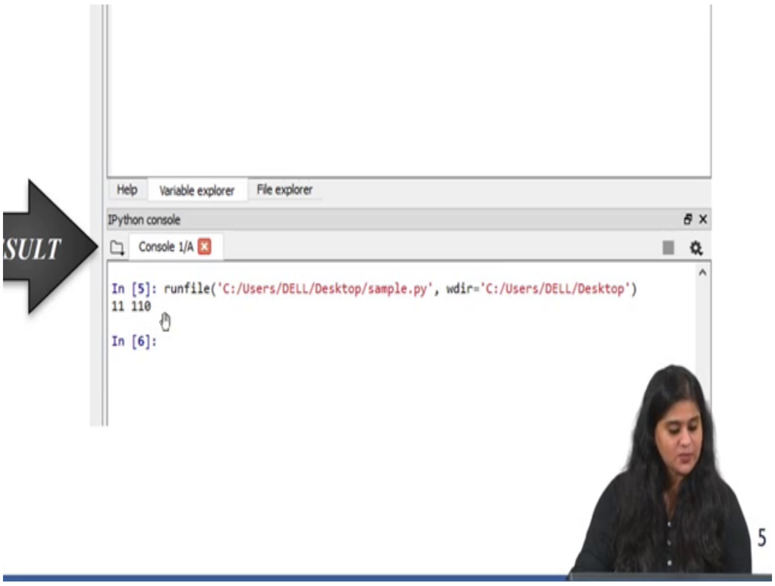
```
In [5]: runfile('C:/Users/DELL/Desktop/sample.py', wdir='C:/Users/DEL
11 110

In [6]:
```

Python for Data Science

So, I am now first starting with the run file option which will run an entire file at once. So, once you have code ready you can click on the green triangle icon and you will see an output here. In this case, I am running the same script file that I used in my earlier lecture.

Now which says a=11, b=8 *10 and print(a,b). So, the output I am likely to be get are the values of a and b here. So, once you run the code; you will see the values being stored in the environment and this you can find under the variable explorer. So, in the variable explorer you can see I have my vales of a and b and they are integer type; the size is 1 because the both have only one value and the corresponding value is displayed. Now, after I run on my console; I have the output here. So, let me just zoom in to show you how does the output look.

(Refer Slide Time: 02:13)



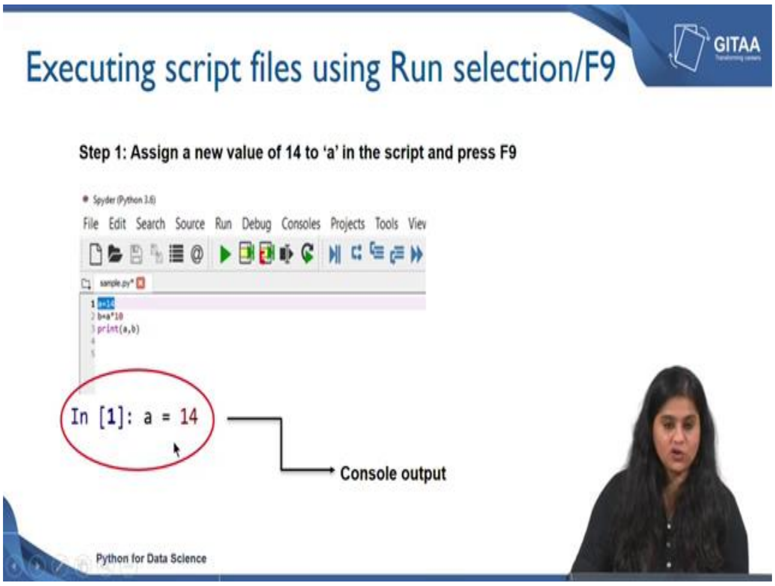
```
In [5]: runfile('C:/Users/DELL/Desktop/sample.py', wdir='C:/Users/DELL/Desktop')
11 110

In [6]:
```

So, this my output I have my value of a and b; a is 11, b is 110. Now, since these work contain within the print statements; I will be getting only these two values as my output. So, another thing to note is that when whenever you click on the run file option, corresponding function is run. So, the run file command is actually the function that will be used to run an entire file at once.

If you are not using the icon or the keyboard shortcut, you can also run your file using this command. The input to the run file function is the name of the file along with its entire directory. Now, you also have another parameter called wdir which means working directory and you can specify in whichever a directory the file is residing.

So, me for it is residing the desktop; so I have given the same directory. So, if do not want to use the icon or the keyboard shortcut, we can also use that run file command. So, in this case since the output is contained within the print statement; you will be able to see only the output a and b.

(Refer Slide Time: 03:23)



Executing script files using Run selection/F9

Step 1: Assign a new value of 14 to 'a' in the script and press F9

Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View
sample.py*

```
a=14
b=a*10
print(a,b)
```

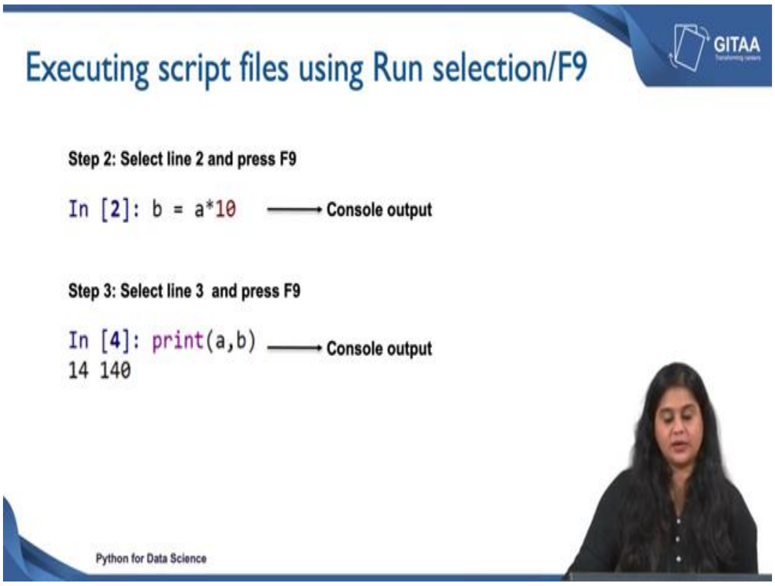
```
In [1]: a = 14
```

Console output

GITAA
Python for Data Science

So, now let us see how to execute few pieces of code using the run selection option or the F9 command from the keyboard. Now, to my earlier code I am assigning a value of 14 to a and I am selecting the line and then I pressing F9. So, I am just going to using the shortcut here; you can also use this icon. Now once you select the line and press F9, you will see a corresponding output being displayed in your console; now this says a = 14. So, whenever you use run selection command or F9; all these lines of code will be displayed in your console. So, whatever I have shown you here is the console output.

(Refer Slide Time: 04:08)



Executing script files using Run selection/F9

Step 2: Select line 2 and press F9

```
In [2]: b = a*10
```

Console output

Step 3: Select line 3 and press F9

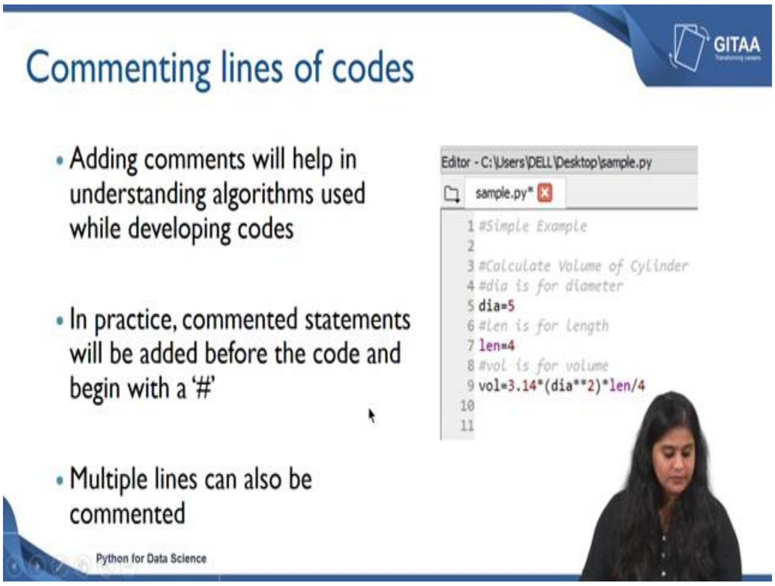
```
In [4]: print(a,b)
14 140
```

Console output

So, now run the second line which is b=a*10 and then press F9, now once you do that you will corresponding see the code in your console. Then you can run the line which says print(a,b) and once you run this line print(a,b); you will see a corresponding output as well. So, if you have noticed in run every time you select a line and run; it the corresponding code is also displayed in the console. But however, if you going to use run file command which runs an entire file at once then all these lines of code is not printed in the console. So, this is also one of the difference between using a run file and run selection command.

So, you can also use run selection to debug. So, if you want to go through each and every line and if you want to find out bags or falls on mistakes that you would have done then you can use the run selection option. So, this is one of the major advantages of using run selection, but again if you have a 1000 line or if you have. So, again if you have a large code then it is going to be impossible for you to use the run selection option.

(Refer Slide Time: 05:20)



Commenting lines of codes

- Adding comments will help in understanding algorithms used while developing codes
- In practice, commented statements will be added before the code and begin with a '#'
- Multiple lines can also be commented

Editor - C:\Users\DELL\Desktop\sample.py

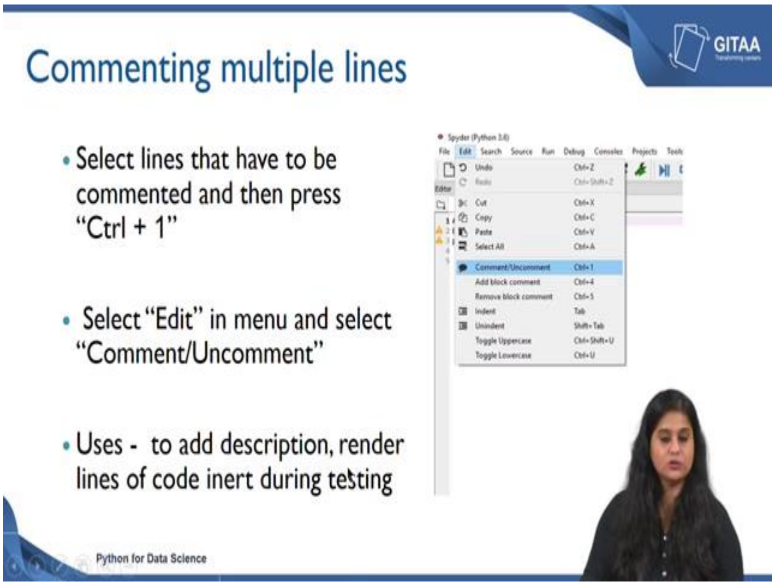
sample.py*

```
#Simple Example

#Calculate Volume of Cylinder
#dia is for diameter
dia=5
#len is for length
len=4
#vol is for volume
vol=3.14*(dia**2)*len/4


```

So, now let us move on to commenting script files. So, adding a comment will aid in the understanding of algorithms that have been used to develop a code. On my right, you can see a snapshot now this is a very trivial example that describes how a volume of cylinder is calculated to comment any line you basically begin with '#'. So, here I have described the title of the task that I am going to do.

Now, I am going to calculate the volume of cylinder; now apart from describing the task or the task objective you can also define what each of the variables mean in your code. So, here I said dia is diameter, len is length vol is volume. Now, this is a very good practice because if you are going to give your code to someone or if you are going to revisited in the future; you might want to know what you have done and why you done it. Now, you can also comment multiple lines instead of just one line.

(Refer Slide Time: 06:13)



Commenting multiple lines

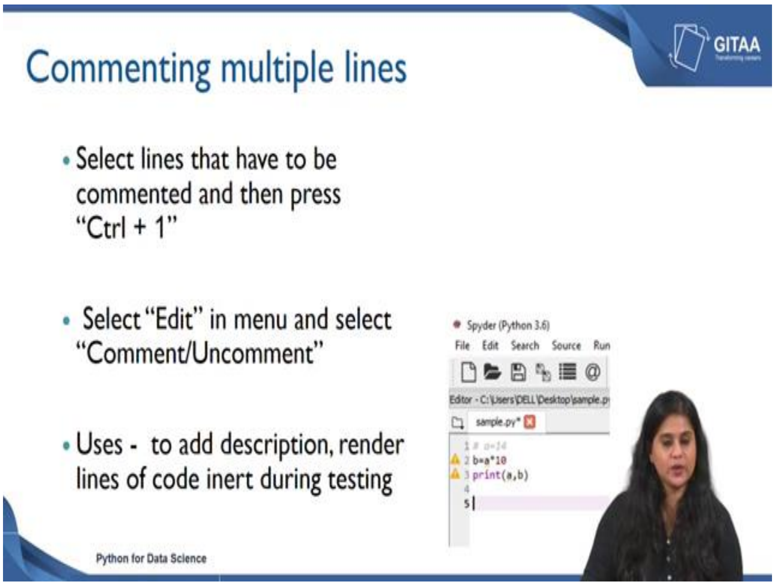
- Select lines that have to be commented and then press "Ctrl + 1"
- Select "Edit" in menu and select "Comment/Uncomment"
- Uses - to add description, render lines of code inert during testing

Python for Data Science

So, to comment to multiple lines select on the lines that have to be commented and then press on "Ctrl+1"; now, this is the keyboard shortcut and alternative ways to go to the edit option in the menu and then select comment or uncomment lines.

You can see the keyboard shortcut also being displayed just adjacent to the comment uncomment option under the edit menu. Like I earlier said you can add description to your code to make it more comprehensible, but apart from just making it incomprehensible if you are in the beginning stage of developing a code where you are trying and testing out of you think then you can also use commenting as a way of making a few lines inert.

(Refer Slide Time: 06:58)



Commenting multiple lines

- Select lines that have to be commented and then press “Ctrl + 1”
- Select “Edit” in menu and select “Comment/Uncomment”
- Uses - to add description, render lines of code inert during testing

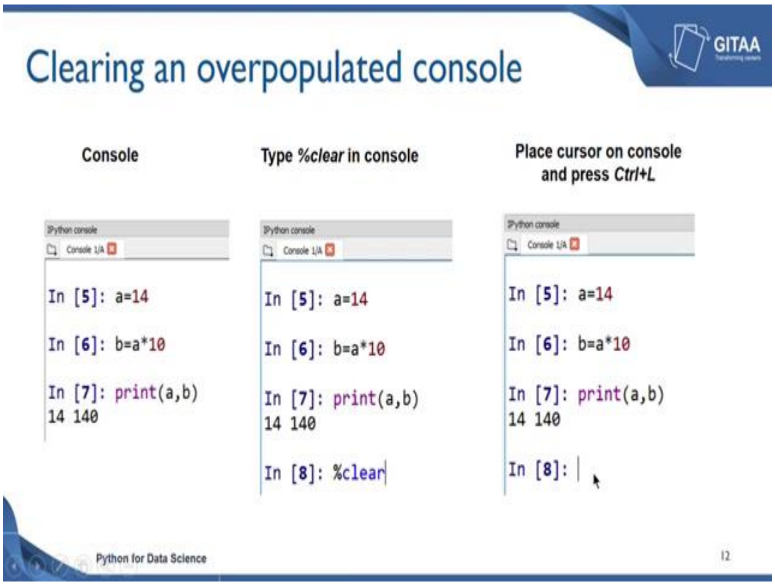
```
# a=14
b=a*10
print(a,b)
```

Python for Data Science

So, what do I mean by this is that. So, let us take the previous example where I say a=14, b=8*10 and print(a,b). Now I am just trying and testing out and seeing what will happen if I just comment a. So, I am basically just making the first line inert and then running the successive lines.

Now, if you are playing with your code and you are in the developing stage then you can also use commenting as a way of keeping lines inert. So, now this is another use of committing. So, till now we seen how to execute an entire file at once and how to execute few lines of course, we have also looked at commenting as a way of adding description to your code. Now let us see how to clear the console and the environment.ment.

(Refer Slide Time: 07:43)



Clearing an overpopulated console

Console

```
In [5]: a=14

In [6]: b=a*10

In [7]: print(a,b)
14 140
```

Type *%clear* in console

```
In [5]: a=14

In [6]: b=a*10

In [7]: print(a,b)
14 140

In [8]: %clear
```

Place cursor on console and press *Ctrl+L*

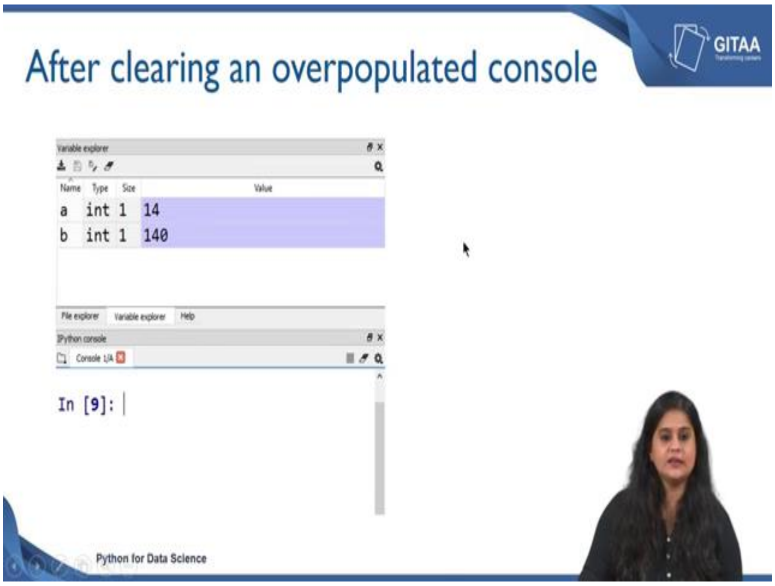
```
In [5]: a=14

In [6]: b=a*10

In [7]: print(a,b)
14 140

In [8]:
```

Python for Data Science

12

If you have an overpopulated console where you have printed multiple lines of codes and multiple outputs; then you might also want to just clear it off and start a fresh. So, let us take this example where I have run the codes; I have the same codes as earlier and I am just running them. So, this is how my console looks right now; now if I want to clear it. So, you can type "%clear" in the console and once you hit enter, your entire console is clear. Now, an alternate ways to use the "Ctrl+l" shortcut from the keyboard that will also work; so this is to clear the console.

(Refer Slide Time: 08:24)



After clearing an overpopulated console

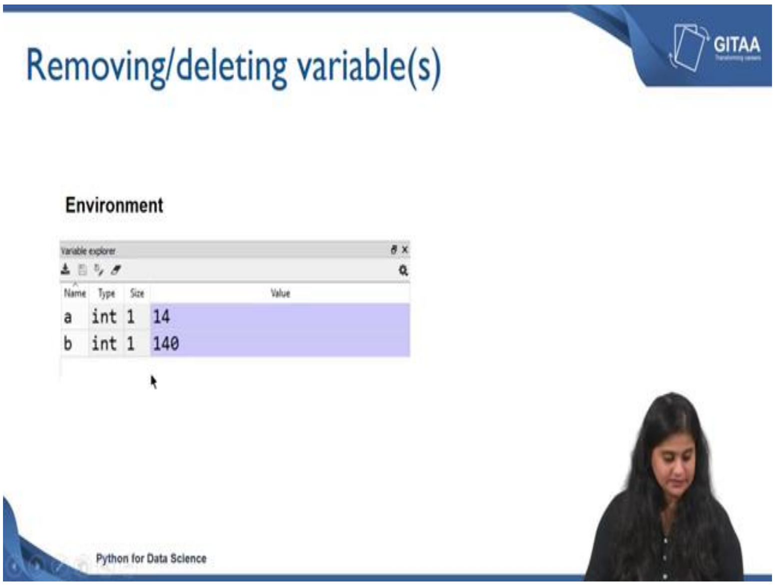
Name	Type	Size	Value
a	int	1	14
b	int	1	140

```
In [9]:
```

Python for Data Science

Now, once you clear the console this is how it looks; I have snapshot here that tells you how does the console look once you clear it. An important point to note here is that; so whenever I clear my console only the output windows cleared the variable explorer still remains intact all the variables are still there; so, clearing a console only means that you are just clearing or flushing out the output window. So, now let us see instead of just clearing the console is there a way to just clear the environment as well.

(Refer Slide Time: 08:57)



Removing/deleting variable(s)

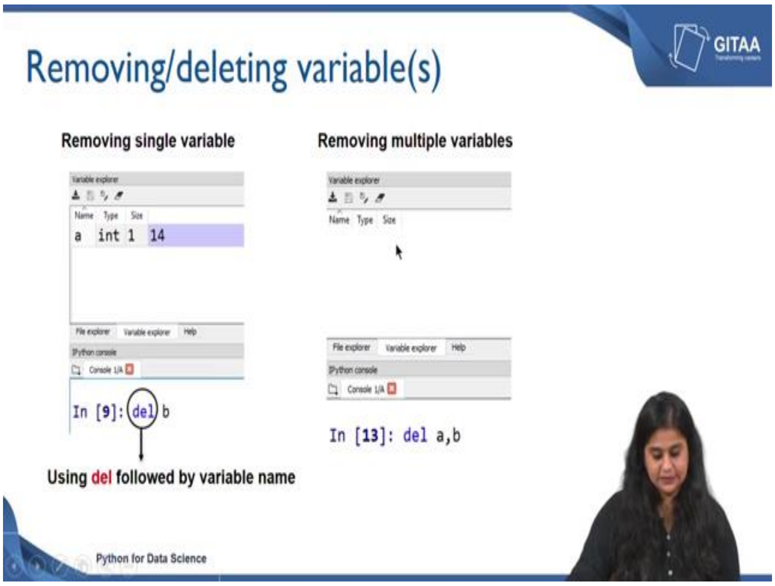
Environment

Name	Type	Size	Value
a	int	1	14
b	int	1	140

Python for Data Science

So, I might also be interested in removing or deleting a few variables from my environment. So to begin with I have two variables in my environment which is a and b and they have a value of 14 and 140 respectively. So, let us see how to remove or delete these variables.

(Refer Slide Time: 09:15)



Removing/deleting variable(s)

Removing single variable

```
In [9]: del b
```

Using del followed by variable name

Removing multiple variables

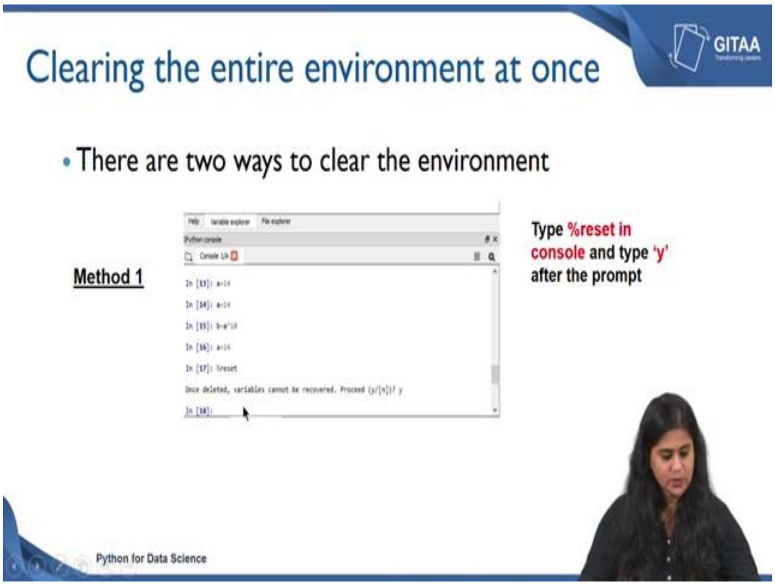
```
In [13]: del a,b
```

Python for Data Science

Now, to remove a single variable; you can just give del space followed by the variable name. Now this you can type in the console; so del stands for delete and it has to be followed by a space and then a variable name. Now, once you hit enter what you will see is that one of the variables in this case b has been removed from the environment. So, here you can see variable b has been removed from the environment. Now, instead of removing a single variable we can also remove multiple variables from the environment. Now still use the same command del, but instead of just giving one variable; you will give two variables and you have to ensure that you separate the variables with the comma.

Now, if you are typing along with me please ensure that b is also present in the environment and then you can type this code. So, once you hit enter you can see that the entire memory has been flushed out and both these values have been deleted.

(Refer Slide Time: 10:18)



Clearing the entire environment at once

- There are two ways to clear the environment

Method 1

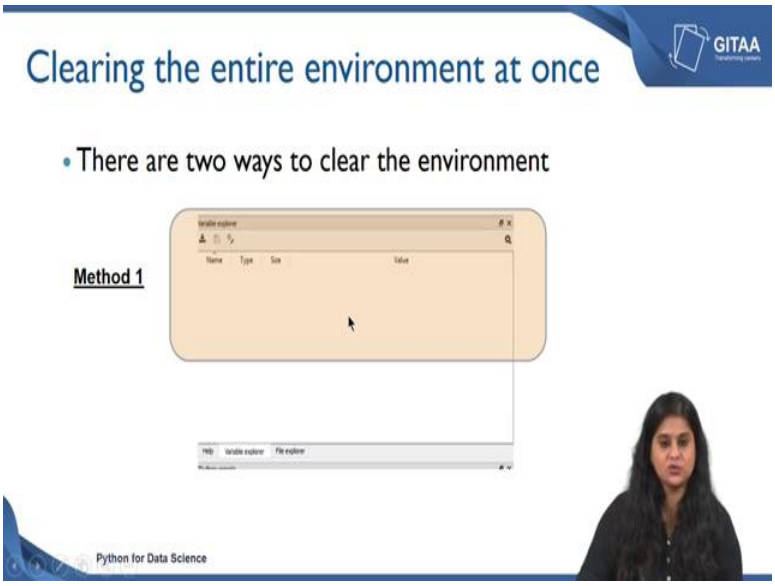
```
In [13]: a=14

In [14]: a=14

In [15]: b=a*10

In [16]: a=14

In [17]: %reset

Once deleted, variables cannot be recovered. Proceed (y/[n])? y

In [18]:
```

Type %reset in console and type 'y' after the prompt

So, instead of dropping variables one by one; you might also be interested in clearing the entire environment at once. So, there are two ways to go about it the first way is to use a "%reset" command in the console.

Now, once you type %reset and hit enter; it will prompt you with the line that reads as once deleted, variables cannot be recovered; proceed yes or no; y stands for yes and n stands for no. Now, this is to ensure that you have not typed percentage reset accidentally and this is just another layer of check to make sure that you do not flush out the important variables in your environment. Now, if you would like to proceed then type 'y'; otherwise you can type 'n' and now once you hit enter, you will see that the entire environment has been cleared out.

(Refer Slide Time: 11:07)



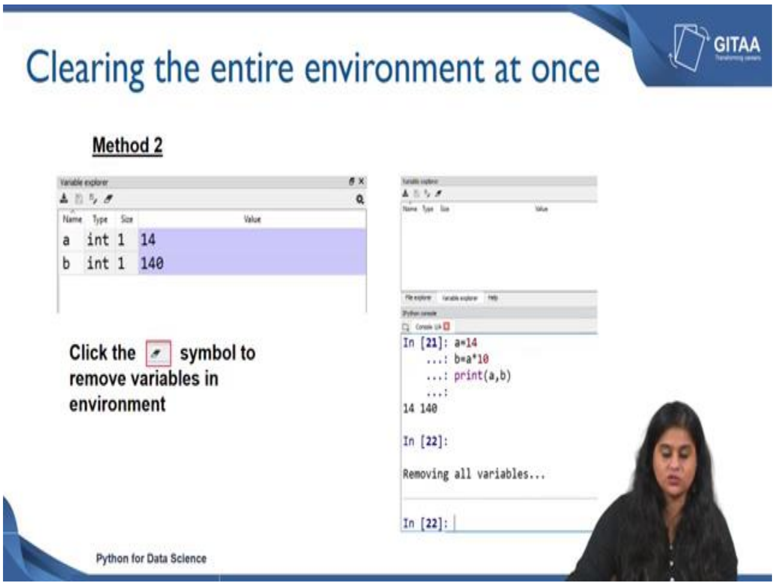
Clearing the entire environment at once

- There are two ways to clear the environment

Method 1

So, this is using a command.

(Refer Slide Time: 11:13)



Clearing the entire environment at once

Method 2

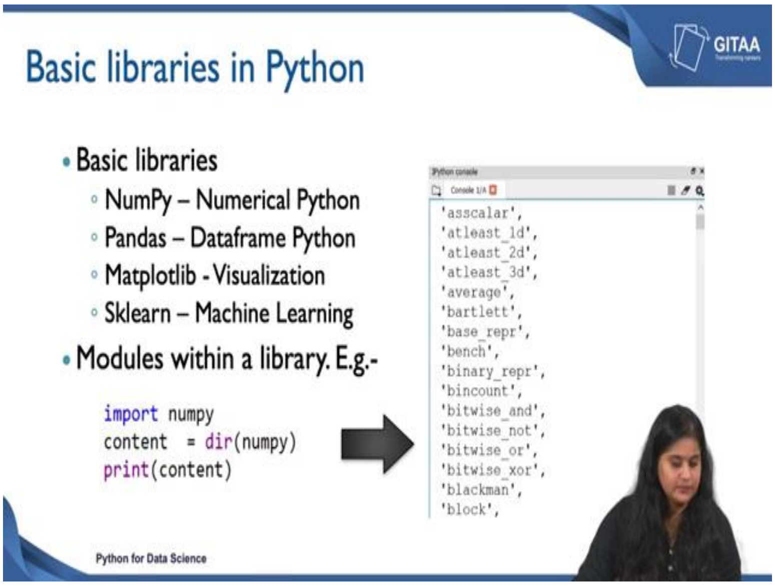
Name	Type	Size	Value
a	int	1	14
b	int	1	140

Click the symbol to remove variables in environment

```
In [21]: a=14
    ...: b=a*10
    ...: print(a,b)
    ...:
14 140

In [22]:

Removing all variables...

In [22]:
```

Now, let us see how to clear the environment using an icon. Now, above the variable explorer there are a couple of icons here; the one on the extreme right looks like an eraser. So, now click on the icon it will prompt you to the dialogue box; if you click all the variables will be removed. So, while it is removing variables it also prompts with the line that says removing all variables. So, till now we have seen how to execute an entire files or few lines of codes and how to comment the code that you have written and how to clear the console and the environment.

(Refer Slide Time: 11:56)



Basic libraries in Python

- Basic libraries
  - NumPy – Numerical Python
  - Pandas – Dataframe Python
  - Matplotlib - Visualization
  - Sklearn – Machine Learning
- Modules within a library. E.g.-

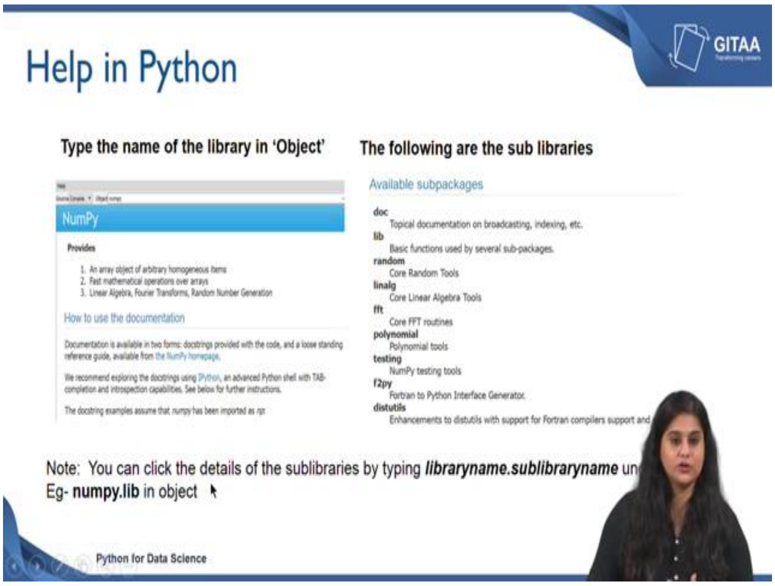
```
import numpy
content  = dir(numpy)
print(content)
```

```
'asscalar',
'atleast_1d',
'atleast_2d',
'atleast_3d',
'average',
'bartlett',
'base_repr',
'bench',
'binary_repr',
'bincount',
'bitwise_and',
'bitwise_not',
'bitwise_or',
'bitwise_xor',
'blackman',
'block',
```

So, now let us take a look at some of the basic libraries in Python. Now, there are four major libraries that get installed at the time of installation of Python. These are NumPy which stands for Numerical Python, Pandas which Paneled Dataframe, Matplotlib which stands for Visualization and Sklearn which is used for machine learning. So, these are four major library that are important to solve a data science problem.

So, these are parent libraries; there also sub libraries contained within these. So, to access the contents of a library, you need to first import the library; in this case I am importing NumPy. Now ‘dir’ represents directory; so this is the directory of the library; in this case it is numpy. Now, I am just saving the entire directory on to a variable name called content and I am just printing the object content. So, once you run these three lines of code on your console all the sub libraries will be printed. So, this is one way to actually access the sub library. So, now this is the little tedious because you actually have to skim through all the sub libraries to know what they consist of and also your console get overpopulated.

(Refer Slide Time: 13:12)



Help in Python

Type the name of the library in 'Object'

NumPy

Provides

1. An array object of arbitrary homogeneous items
2. Fast mathematical operations over arrays
3. Linear Algebra, Fourier Transforms, Random Number Generation

How to use the documentation

Documentation is available in two forms: docstrings provided with the code, and a loose standing reference guide, available from the NumPy homepage.

We recommend exploring the docstrings using IPython, an advanced Python shell with TAB-completion and introspection capabilities. See below for further instructions.

The docstring examples assume that numpy has been imported as np

The following are the sub libraries

Available subpackages

doc
Topical documentation on broadcasting, indexing, etc.
lib
Basic functions used by several sub-packages.
random
Core Random Tools
linalg
Core Linear Algebra Tools
fft
Core FFT routines
polynomial
Polynomial tools
testing
NumPy testing tools
f2py
Fortran to Python Interface Generator.
distutils
Enhancements to distutils with support for Fortran compilers support and

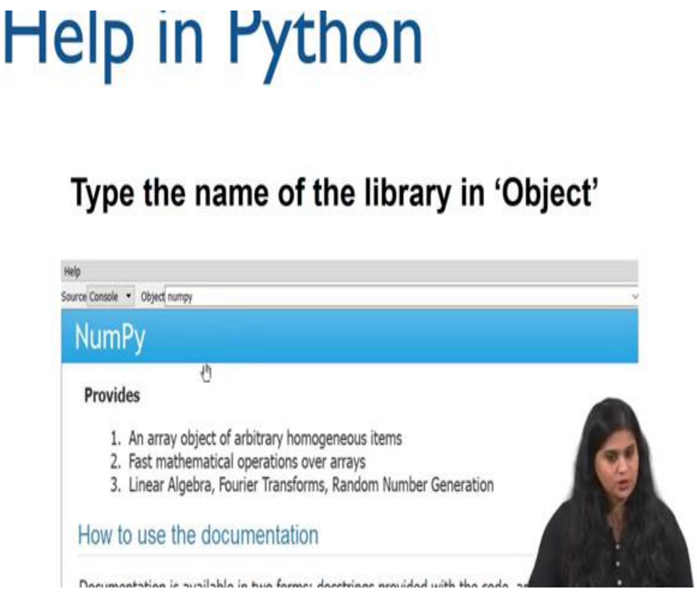
Note: You can click the details of the sublibraries by typing ***libraryname.sublibraryname*** un

Eg- **numpy.lib** in object

Python for Data Science

Now, under the help tab you have a search box that's titled as object; let me just zoom into show you how it looks.

(Refer Slide Time: 13:19)



Help in Python

Type the name of the library in 'Object'

NumPy

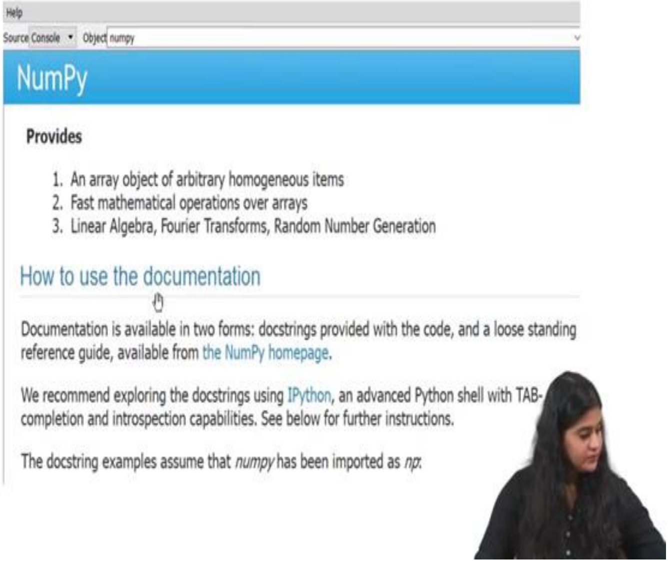
Provides

1. An array object of arbitrary homogeneous items
2. Fast mathematical operations over arrays
3. Linear Algebra, Fourier Transforms, Random Number Generation

How to use the documentation

Now, under object you are going to mention your library name. So, in this case I have mentioned NumPy; the moment you hit enter a documentation pops up.

(Refer Slide Time: 13:30)



NumPy

Provides

1. An array object of arbitrary homogeneous items
2. Fast mathematical operations over arrays
3. Linear Algebra, Fourier Transforms, Random Number Generation

How to use the documentation

Documentation is available in two forms: docstrings provided with the code, and a loose standing reference guide, available from the NumPy homepage.

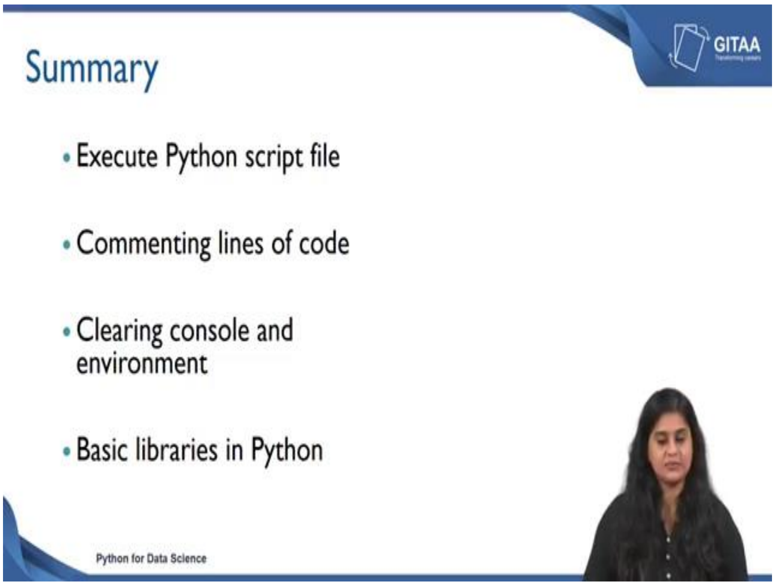
We recommend exploring the docstrings using IPython, an advanced Python shell with TAB-completion and introspection capabilities. See below for further instructions.

The docstring examples assume that *numpy* has been imported as *np*:

Now, the documentation tells you what Python does; what it provides and how to use the documentation. Now, apart from that if you scroll down, you can also see a list of sub libraries that are available under NumPy.

So, the sub libraries available under NumPy are linear algebra, Fourier transform routines, polynomial tools so on and so forth. Now, if you want a specific documentation for each of the sub libraries; so you can type in the library name in the search box, follow it up with dot and then the sub library name. So, let us say if I want to access the sub library lib from numpy; then I am going to write ‘numpy.lib’ under the search box object. So, this is how you get a detailed documentation of all the libraries and sub libraries in Python.

(Refer Slide Time: 14:23)



Summary

- Execute Python script file
- Commenting lines of code
- Clearing console and environment
- Basic libraries in Python

GITAA
Python for Data Science

So, to summarize in this lecture we saw how to execute Python script files, how to comment single lines of codes and multiple lines of codes, how to clear the console and the environment and how to access some of the basic libraries in Python.

Thank you.