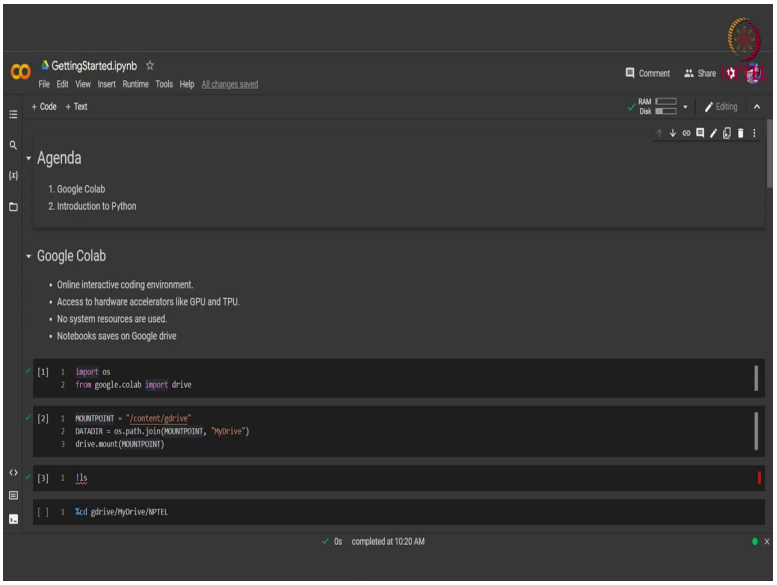
Social Network Analysis
Prof. Shivani Kumar
Department of Computer Science and Engineering
Indraprastha Institute of Information Technology, Delhi

Lecture - 04
Introduction to Python/Colab

Hello all, I am Shivani and I am going to take you through some coding exercises and lectures for this course Social Network Analysis. So, today we will be beginning with an introduction to Python with the help of Google Colab.

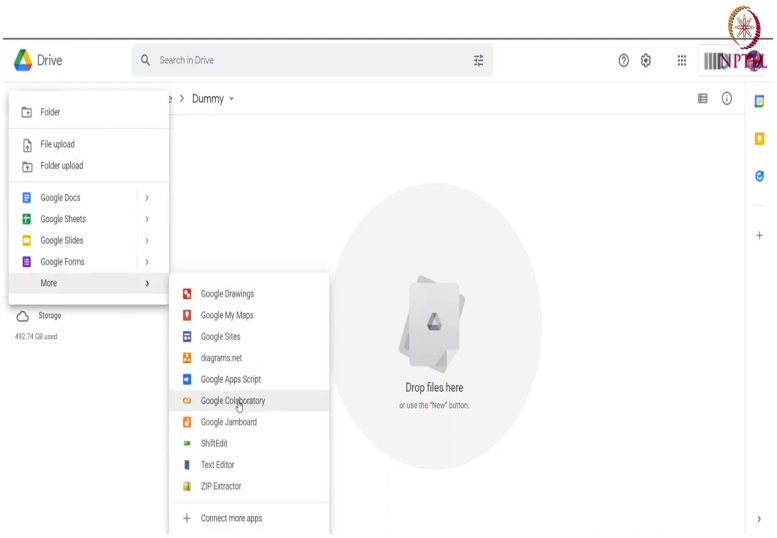
(Refer Slide Time: 00:40)



To begin with we must understand what Google Colab is. So, basically Google Colab is an online interactive platform which allows us to do Python programming on the go.

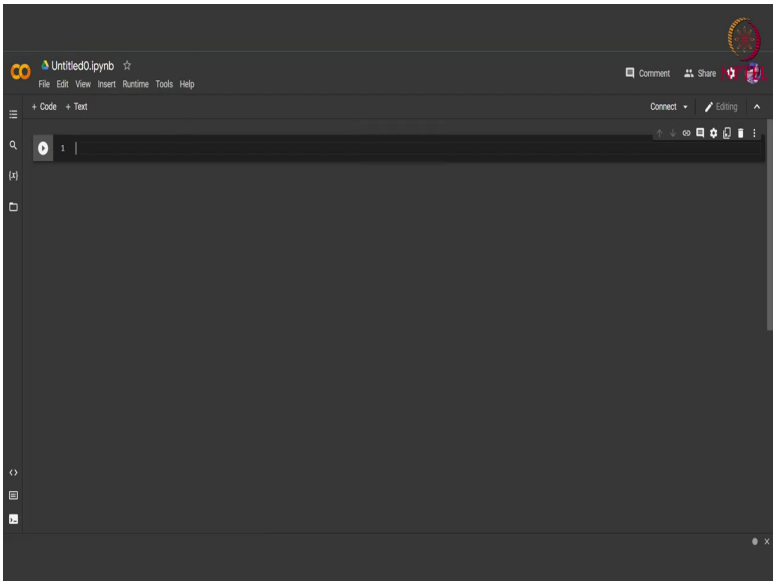
Google Colab provides us with some hardware accelerators like GPUs and TPUs along with cloud CPUs. Essentially if you are working on Google Colab; that means, you are using 0 percent of your own PCs resources except for the internet of course. So, before we get started with the coding in Google Colab, we must know how to create a notebook here, to do that you need two things; first an internet connection and another thing is a Google Drive account.

(Refer Slide Time: 01:36)



If you have a Google Drive account simply go to the folder where you want the a notebook to be and press new. Then if you go to the option More, you will see the Google Colaboratory option here, simply click here and you will be taken to a new brand new notebook.

(Refer Slide Time: 01:56)

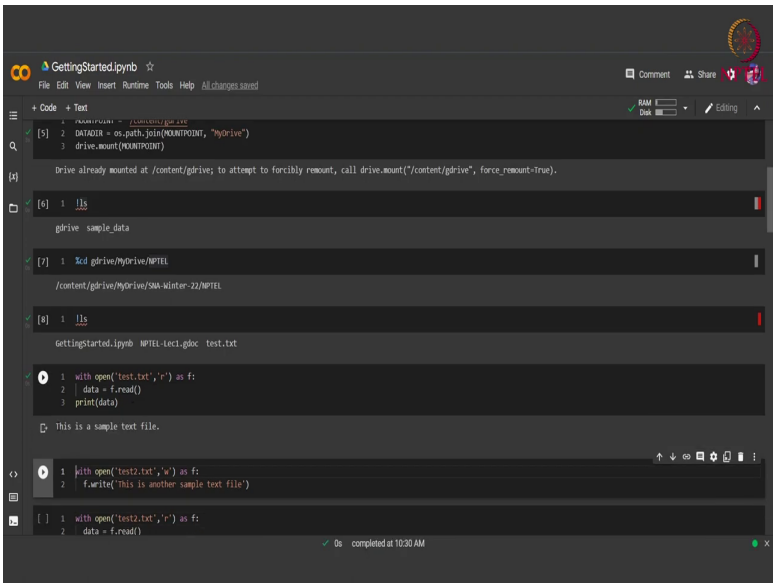


This will be an empty notebook, in which you can write any code that you want. We already have created a notebook here and we will be using this for today's session.

Since we are working on Google Drive and we may want to access files that are already present in this folder, to do that we must connect our notebook to the drive since we are getting assigned a new Cloud CPU each time we create the notebook. So, to mount the Google Drive to our Google Colab notebook we must use this mount function this function is available in this Python library called google dot colab.

We import this library and then we simply call this drive dot mount function giving it as input, the path that we want to mount. So, here our path is simply a drive and we also append this my drive path to the main path.

(Refer Slide Time: 03:09)



We run this cell and this cell will generally ask us for a permission for which we just select accept and yes and then we are shown this message that the drive is mounted.

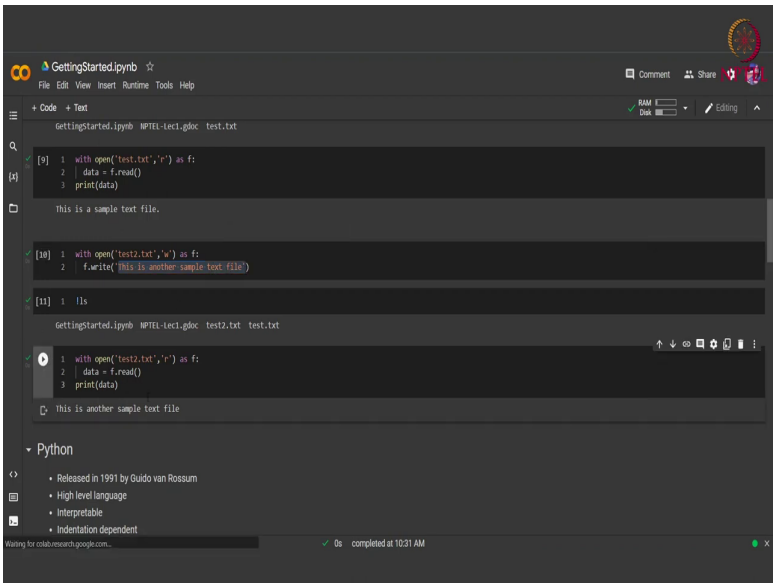
Now, since this is a Python notebook, we cannot use a terminal directly here, but we can use shell commands by using this exclamation marks symbol. We simply run this shell command and we can see what all files and folders we have on the path that is mounted to our drive. Except for exclamation mark we can also use this percentage sign in order to run shell scripts. So, we simply move to the required folder where we have our data and we again do ls here and we see that we have 3 files in this particular folder.

Now, what we can do is, now that we have our folder mounted with the notebook, we can also create and read information from this folder. So, suppose we have this text file already

present in our folder as can be seen from this ls command, that we have this text test dot txt already present in our folder. We can read this by calling this function, with open name of the file in what mode we want to open it.

So, here we are opening it in the reading mode, then the file descriptor and then we simply call the read function from the descriptor in order to read the contents of the file. After we have the contents in this variable called data, we simply print this variable using the print function. Let us run this cell and we can see that we are shown the content of the file that is this is a sample text file.

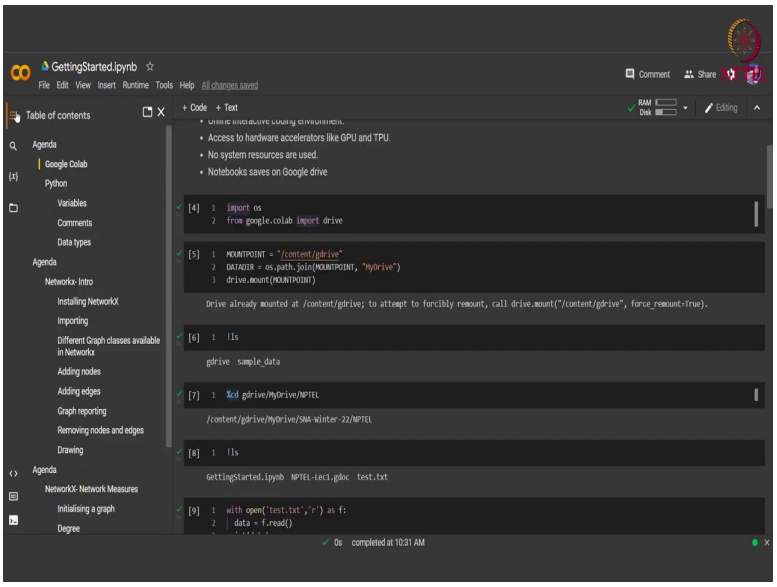
(Refer Slide Time: 05:26)



Now, instead of just reading we can also write to the directory that is mounted with our notebook we can simply call the a text file, but with the option of writing. So, we write with open, the name of the file the option write and the file descriptor and then we call the write function with the contents that we want to write in the file. We run this cell and now we will open this file in the read mode to see whether we have the contents now in or not.

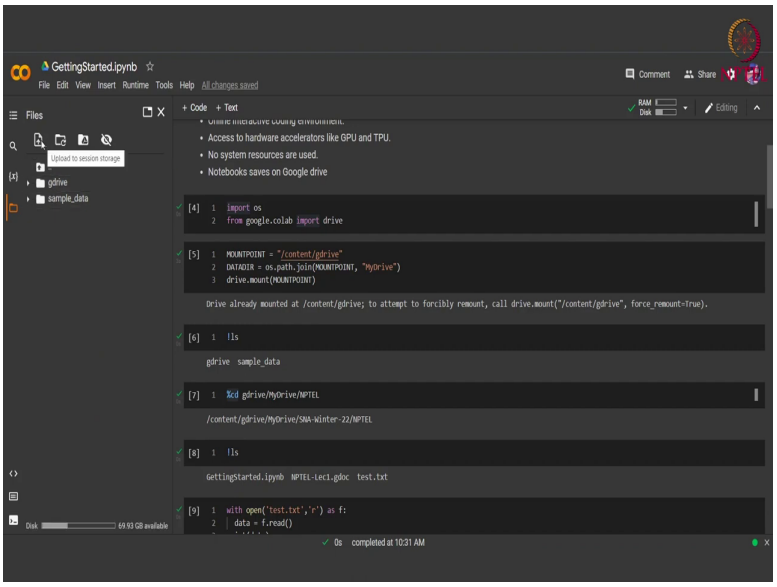
But before that let us first see whether our file is actually present in this folder or not. We see that now we also have this test2 dot txt present in our folder. We read this using the same command as we read the test dot txt, we read this and see that whatever content that we that we have written inside the file that is being read here. Instead of mounting our file using this these functions, Google also provides us with a ui that is that can help us in mounting.

(Refer Slide Time: 06:52)



So, here if we click at the left menu, we can see the table of contents. If we go down we have this folder option.

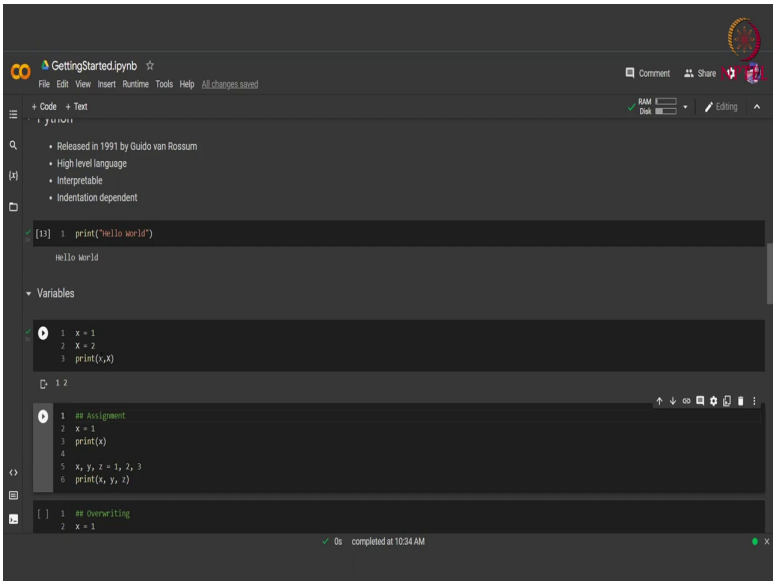
(Refer Slide Time: 07:03)



We click here and we can see that since this is already mounted, we can see the contents of the drive. We can refresh this contents or we can mount the drive again. We can also directly upload any file from here. Now, that we are familiar with Google Colab we would encourage you to explore Google Colab more on your own, since we only covered a very small amount

of functionality that it provides. But this is enough for us to get started with Python. Let us move on to Python.

(Refer Slide Time: 07:50)



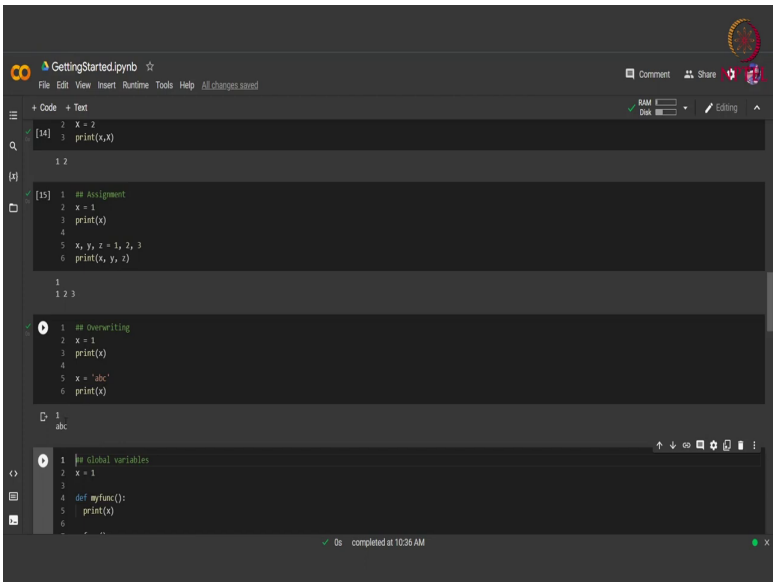
So, Python is basically a high level interpretable and a general purpose programming language. So, by general purpose I mean that it can be used for various purposes like web development software development or server side scripting. We; so let us first start with our “Hello World” function of Python.

So, we can simply call this print function and give the input as “Hello World” and we are given the “Hello World” as the output. Now Python it like any other programming languages it has variables, data types, comments and such things. So, let us first begin with the variables that are involved in Python. So, like as I said like any other programming language Python also has variables with a similar naming convention. For example, a variable name cannot start with a digit, it has to start with an alphabet or an underscore.

Also, Python variables are case sensitive that is small x and capital X will represent two different variables. So, this can be seen here if we have small x equal to 1 and capital X equal to 2 and we print x and capital X together, we will get the output as 1 and 2 because the variables are case sensitive. Now as seen in the above code snippet, what we did was assignment that is we assigned some value to the variable x.

So, here also we are assigning the value of integer 1 and then we are simply printing the variable x. Apart from just this single assignment we can also assign different variables with different values by simply doing this comma separation. So, we have 3 variables, x, y and z separated by comma and they are assigned with three different values 1, 2 and 3 also separated by comma.

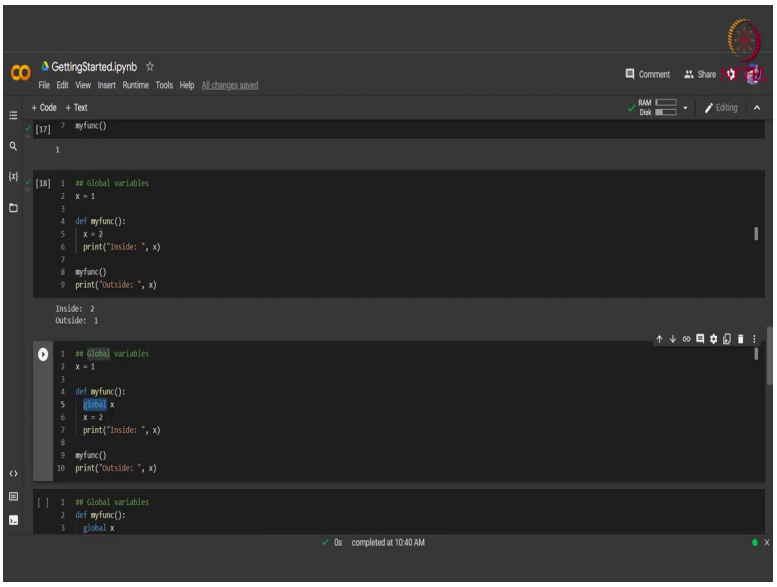
(Refer Slide Time: 10:21)



We print these and we see that the first print statement gives us 1 the value of x and the next print statement gives us the value of x, y and z that is 1, 2 and 3. A very interesting aspect of Python variables is that a variable is not bounded to any data type that is if we have already assigned a variable the type integer we can further go down in the program and can assign the same variable a string value.

For example, here we can see that first we have initialized x with the value of an integer 1, we print this x and then we assign the same variable x with the value a b c that is a string, then we print then we print x again let us run this cell. And we see that first we are give, we are outputted the value one which is an integer, but again when we print the same variable with the string value we are given the output as string and no error is shown here.

(Refer Slide Time: 11:46)



Moving on, we might also want some global variables to be present in our program, global variables are variables that have the scope throughout the program, that is we can access them anywhere and probably modify them also, we will see about that. So, first we start with x as the global variable this is or this is by default a global variable, because it is defined outside of any function. So, we provide it with the value 1 and then we define a function called myfunc and simply print this value of x inside the function.

Then we call this function. Let us see what happens when we run this cell. We run this cell and we see that this print statement inside the function which is outputting the value of x, actually gives us the value that is press that was assigned to this x globally, that is we are outputted the value 1. Now, this happens when we do not modify the value of x anywhere inside the function.

But what happens when we have another x inside the function and we modify that value of x. So, here in this code snippet you can see that we have a global variable x with the value 1 here. And in this function called myfunc what we do is we simply assign the value 2 to the same variable x, but is it the same variable. So, in this scenario this is not the same variable, when we are declaring this or this assignment that is present inside the function it is creating another local variable x that has a scope only inside the function. That is if we have modified the value or we have assigned the value 2 to this x and print this x inside the function and again print an x outside the function, these 2 values will be different.

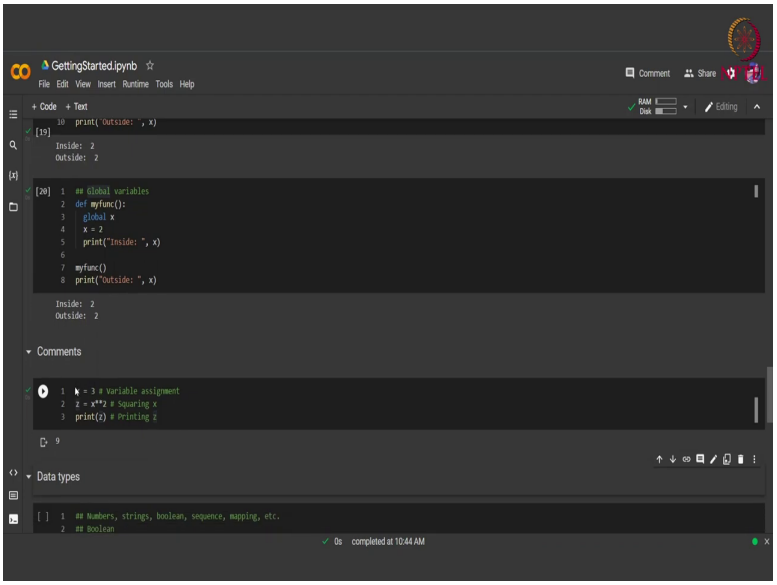
Because inside the function we will be referring to the local version of x, but outside the function since, the scope of the local x has finished we will be referring to the global x that was already present in our in our program. So, let us run this cell to verify our claim, we run this and we see that inside the function the value of x is coming out to be 2 as expected, since we already assigned it the value 2 inside the function.

So, this is a local variable as can be seen by the value of x that is coming outside the function. So, outside the function the value still remains 1, because the global x had the value 1, but it can also happen that we want to modify the global value inside the function. In order to do that what we do is we use the keyword global. We simply define x normally outside the functions as a global variable, this is by default a global variable since it is defined outside of any function.

But inside the function we need to declare that the x that we are using inside the function is actually a global variable to do that we use the keyword global. So, here we write global x and then modify the value of x, that is we are already telling the interpreter that the x that you are going to deal with is a global variable. We assigned the value 2 to this x and then again print x inside as well as outside.

Now, according to our hypothesis the value of x inside should be 2, but outside also it should be 2. Since, we already have defined that the x that we are modifying is the global x.

(Refer Slide Time: 16:34)



We run this cell and we see that as expected we are getting the value of x as to both inside and outside the functions. Then again even if we do not initialize this x outside of any function we can initialize a new global variable inside the function using the global keyword.

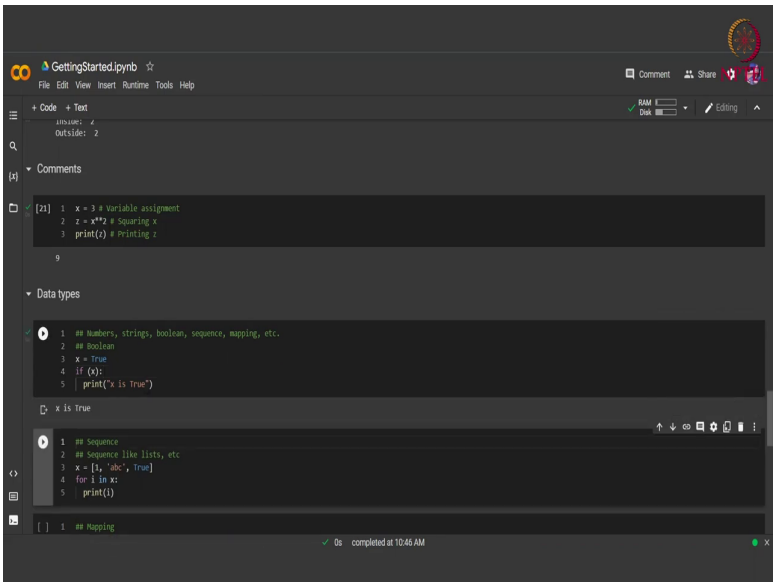
So, we just mentioned here, that we are defining a global variable x assign some value to that variable. And now again if we call this x outside this function we will still get x as a valid variable and with the value 2. Even though we have not defined this variable outside of this function, but inside we have defined it using the global keyword. We run this cell and we see that there are no errors and we are getting the value of x inside as 2 as well as outside as 2.

Now, like any other programming language Python also has comments. So, these comments they are ignored by the interpreter, but they are mostly used to improve the readability of the code so that sharing of code becomes easy. After all we are using Google Colab which allows us to share the code. So, we simply write comments in Python using the hash sign, as we already saw in the previous code snippets, I have mentioned the functionality of each cell using the comments here.

So, the number of hashes do not matter, just after the first hash whatever follows is considered as a comment. So, in this cell we can see that this that we have assigned a comment with each line to describe the functionality of each line separately. So, for example, here the first line is simply a variable assignment, the second line squares the value of x. So, for squaring we use this double asterisk operator then finally, we print the value of this square which we capture in another variable called z.

We run this and we see that whatever is written other than the comment is actually handled by the interpreter and the part that follows after hash that is the comment part is just simply ignored and is useful for the programmers to understand the code.

(Refer Slide Time: 19:30)

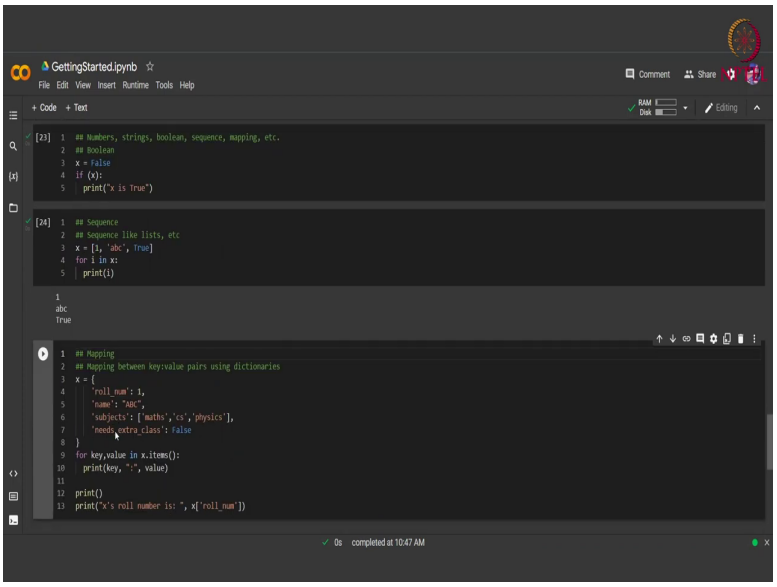


Moving on Python also supports various data types. So, in our previous cell we already saw the integer and the string data types. We saw that we can assign the same variable with different data types and there we saw integer and string.

Apart from integer and strings Python also supports Boolean, sequence, mapping type data types and a lot more. So, let us first see how Boolean is handled in Python. So, Boolean variable is simply a variable that can have either of the two values that is true or false. So, here we assign the variable x the value true. So, here our x becomes a Boolean variable, we can use such variable in a conditional statement like if.

So, we check if x; that means, if x is true then the following code statement will be executed. For example, here if x is true then the code will print that x is true and we have already assigned the value of x as true.

(Refer Slide Time: 20:55)



So, when we run this code this will be printed that x is true. We can also try with the value false and we will see that nothing will be printed because this conditional statement will come as false.

Now, apart from Boolean variables a very interesting data type of Python is a sequential data type. So, sequential data type can have lists sets or tuples in it and other things also. Let us focus on lists here. So, lists basically are like arrays, but in Python a very interesting thing to observe is that in a list we can have different elements of different data types. As you can see here we have a list x where we have three different data types in a single list.

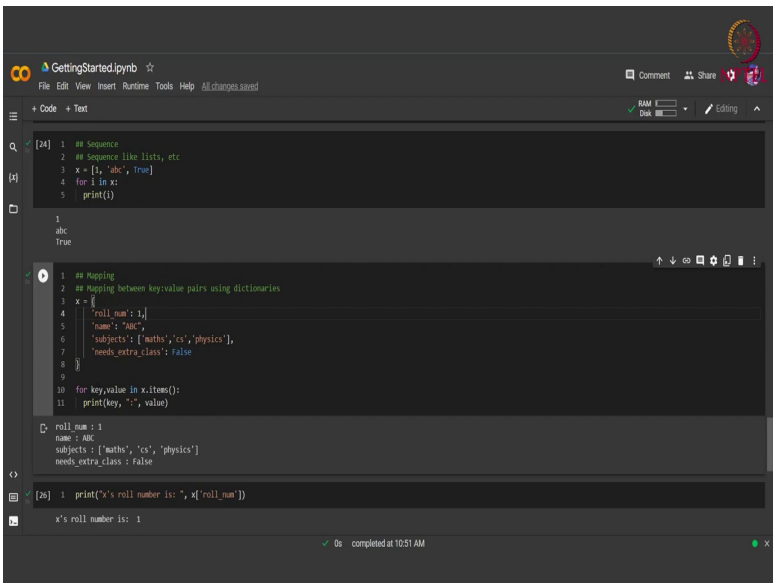
This is unlike an array which has the same data type. So, here list contains an integer, a string and a Boolean value. We can iterate over this list just like we iterate over an array by using a for loop we simply write for i in x that is. So, I will take on the value of one element at a time sequentially and we are simply printing this I, let us run this code and we see that each of the element of the list is printed in a sequential manner.

Moving on from sequences, we have the mapping data type. Now, mapping can be in a key value format. In Python we use the data type dictionaries to perform the mapping. So, dictionaries are basically a data type that has a key and then a value associated with each key. For example, here you can see that we are defining x as a dictionary and to define a dictionary we use these curly brackets and inside these brackets we have defined our keys followed by a colon followed by the value for the corresponding key.

So, for example, here x might contain information of a student. So, it has keys such as roll number, name subjects and or Boolean variable called needs extra class. Now, here we can see that each key can have different data type, data type values. For example, here roll number is an integer whereas, name is a string subjects is actually a sequential data type of three strings, then we have a Boolean type called needs extra class.

Now, in order to iterate over this dictionary, we simply write x dot items which gives us a tuple of key value pairs, in this tuple to access each key value we write for key comma value in x dot items. So, the variable key will take on the values of each key sequentially. Whereas, the variable value will take on the corresponding value, we then print this key and value pairs separated by a colon.

(Refer Slide Time: 24:56)



So, let us first print this and then we will see how we can access a particular key from the dictionary. So, when we iterate over all the items, we see that each key and value is being printed as we wanted separated by a colon and followed by the value. It is simply printed like this, but we can also access each key in a separate manner. So, to do that we simply write the name of the dictionary followed by the name of the key that we want to access.

So, here if we want to access the key roll number, we write x and in square brackets the string roll number that is the key name. So, here we write x's roll number is followed by this format. We run this and we see that x's roll number is whatever the value is here the value is

1. So, now, we it is important to note that a dictionary can have the same name of a key only once.

For example, we cannot have 2 roll numbers inside the same dictionary. So, that is each key must be unique inside the dictionary, although the value of the key can be changed over time. So, today we saw about how to get started with Colab and how to get started with Python with by learning some basic variables and basic data types we encourage you to go home and dwell deeper into this Python's world, because there is a plethora of information to explore there.