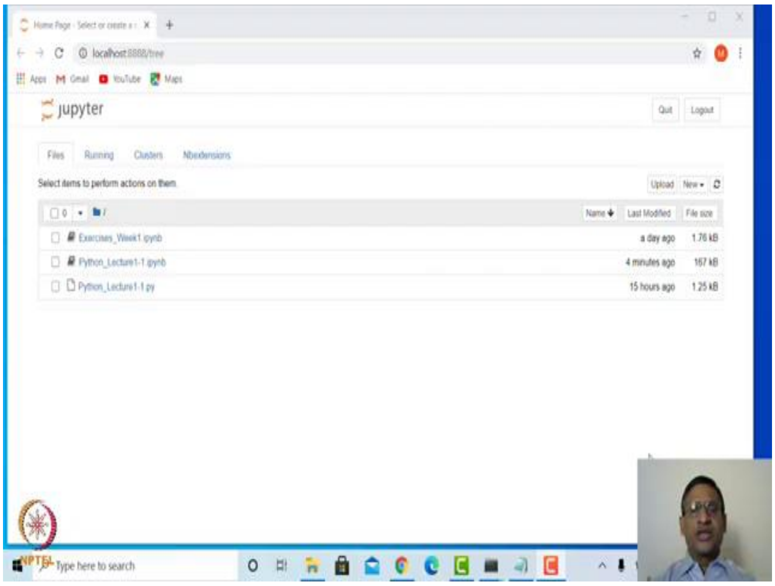
Computational Mathematics with Sagemath
Prof. Ajit Kumar
Department of Mathematics
Institute of Chemical Technology, Mumbai

Lecture – 03
Python as an Advanced Calculator

Hello and welcome to the 2nd lecture on Computational Mathematics with Sagemath. In the previous lecture, we started exploring Python programming language. We used Python to do basic mathematical operations such as addition multiplication division finding quotient remainder and also raising powers. And using these operations we also found out compound interest, simple interest and things like that.

In this lecture, we are going to see how we can make use of Python as an advanced scientific calculator. We will be making use of two built-in modules or libraries namely math module and cmath modules. These modules contains functions such as finding square root, exponential, logarithmic, trigonometrical functions, finding GCD, finding factorial functions and things like that.

(Refer Slide Time: 01:20)

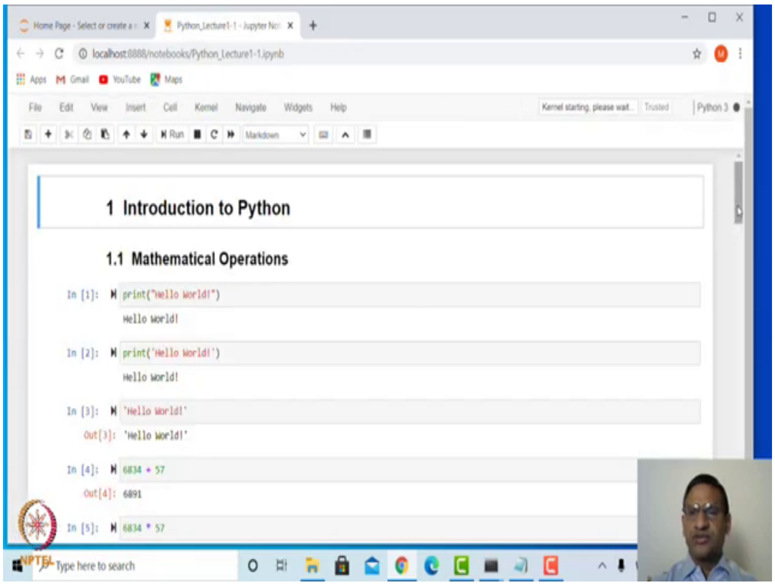


Select items to perform actions on them.

Name	Last Modified	File size
Exercises_Week1.ipynb	a day ago	1.76 kB
Python_Lecture1-1.ipynb	4 minutes ago	167 kB
Python_Lecture1-1.py	15 hours ago	1.25 kB

So, let us get started. We will start Jupyter Notebook by going to Anaconda Prompt and typing Jupyter space Notebook. I have already done that and I have opened this Jupyter Notebook kernel and these are the files in this directory. Let me open this first lecture.

(Refer Slide Time: 01:50)



1 Introduction to Python

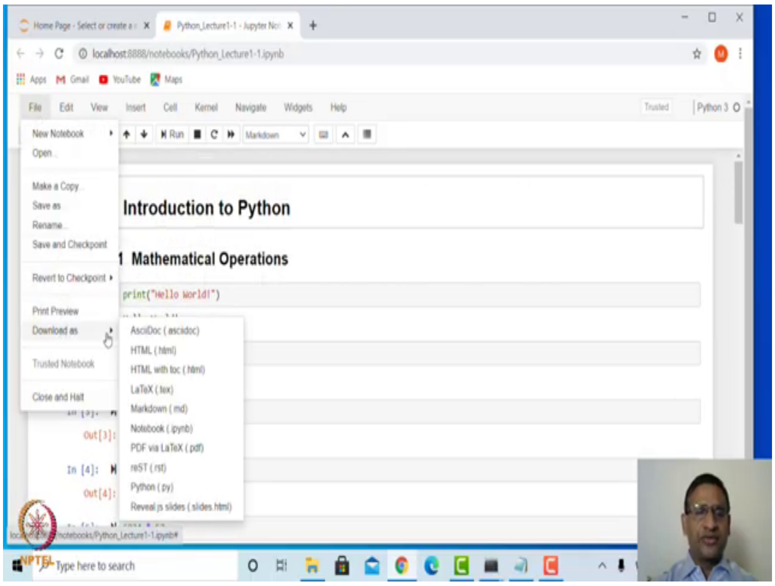
1.1 Mathematical Operations

```
In [1]: print("Hello World!")
        Hello World!

In [2]: print('Hello World!')
        Hello World!

In [3]: 'Hello World!'
Out[3]: 'Hello World!'

In [4]: 6834 + 57
Out[4]: 6891

In [5]: 6834 * 57
```

Python_Lecture1-1 - Jupyter Notebook
localhost:8888/notebooks/Python_Lecture1-1.ipynb
File Edit View Insert Cell Kernel Navigate Widgets Help
Kernel starting, please wait...
Trusted
Python 3
Run
Markdown
Type here to search

So, if I click on   this, it will open the first lecture and you can see these are the things we we did in the last lecture.

(Refer Slide Time: 01:56)



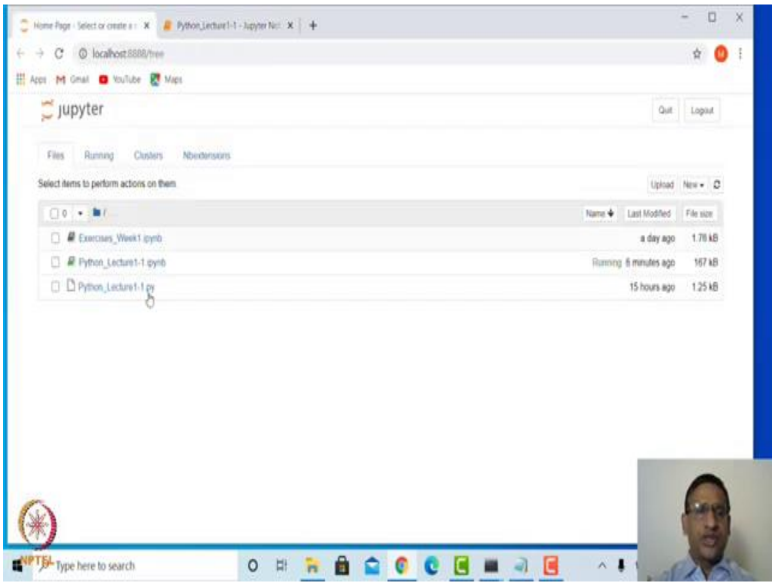
Introduction to Python

Mathematical Operations

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

This file I have saved it in dot py format. So, you can go to download and save as dot py format. I have already saved it.

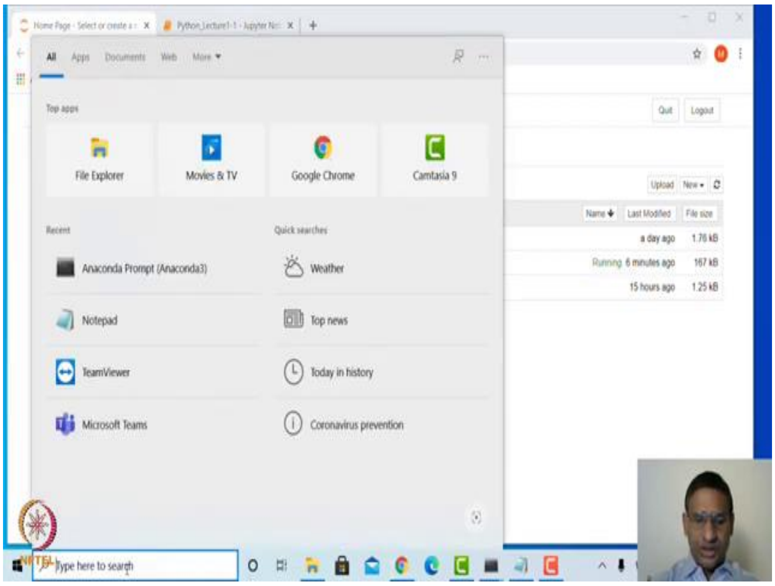
(Refer Slide Time: 02:08)



Name	Last Modified	File size
Exercises_Week1.ipynb	a day ago	1.76 kB
Python_Lecture1-1.ipynb	Running 6 minutes ago	167 kB
Python_Lecture1-1.py	15 hours ago	1.25 kB

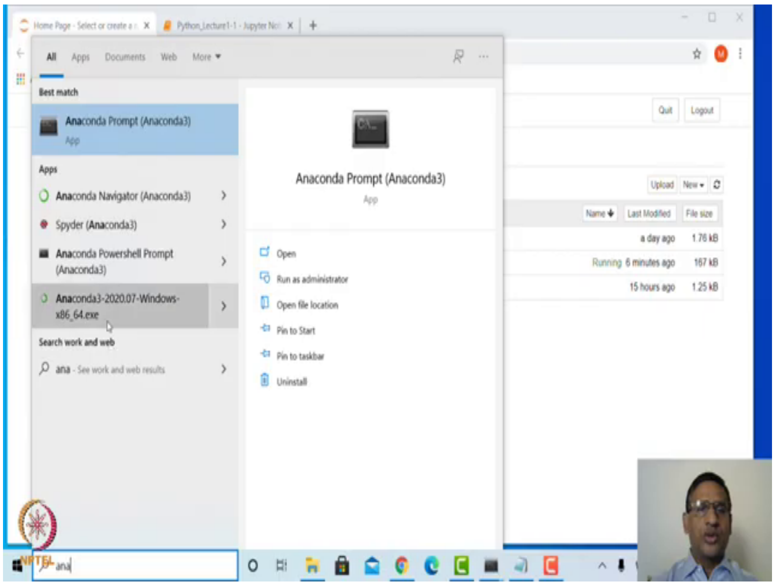
This is what you saw in this directory. So, we have Python underscore Lecture1 hyphen 1 dot py. Now, suppose you want to run this file using Python command line, then what you can do is this.

(Refer Slide Time: 02:23)



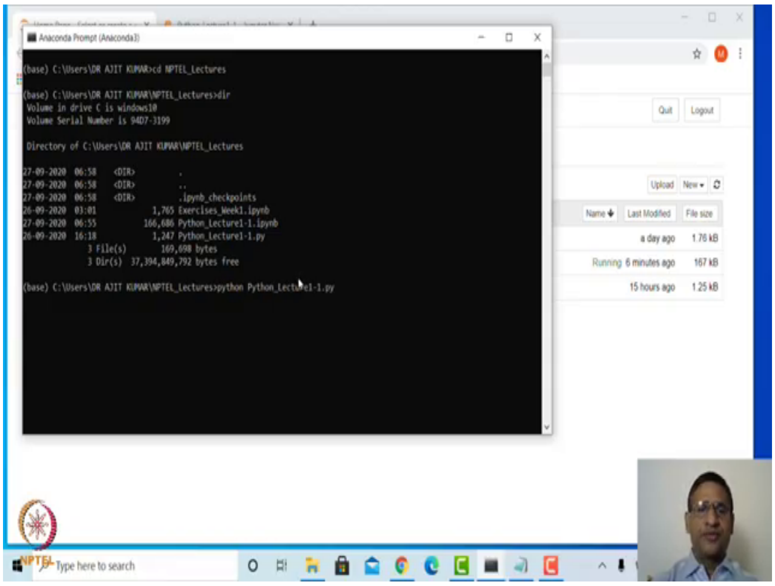
You can go to Anaconda Prompt.

(Refer Slide Time: 02:27)



You can open an Anaconda Prompt.

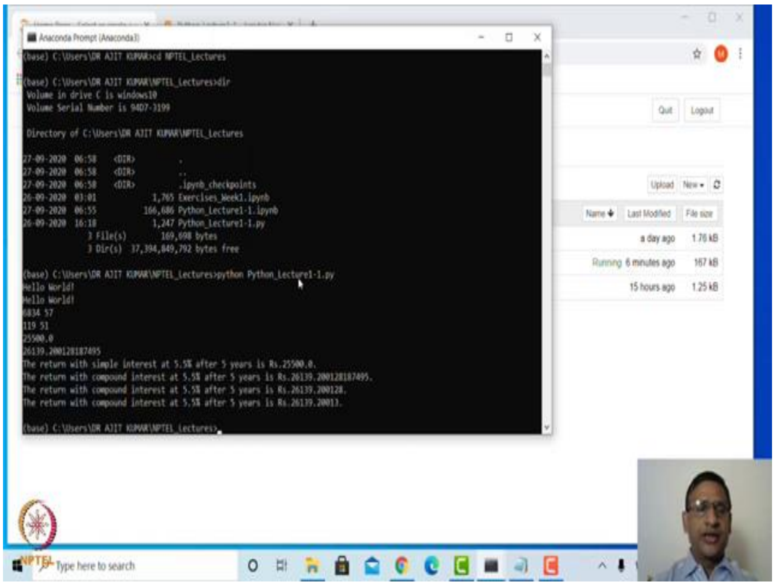
(Refer Slide Time: 02:29)


Anaconda Prompt (Anaconda3)

```
(base) C:\Users\DR AJIT KUMAR>cd NPTEL_Lectures

(base) C:\Users\DR AJIT KUMAR\NPTEL_Lectures>dir
 Volume in drive C is windows10
 Volume Serial Number is 94D7-3199

 Directory of C:\Users\DR AJIT KUMAR\NPTEL_Lectures

27-09-2020  06:58    <DIR>          .
27-09-2020  06:58    <DIR>          ..
27-09-2020  06:58    <DIR>          .ipynb_checkpoints
26-09-2020  03:01             1,765 Exercises_Week1.ipynb
27-09-2020  06:55           166,686 Python_Lecture1-1.ipynb
26-09-2020  16:18             1,247 Python_Lecture1-1.py
               3 File(s)        169,698 bytes
               3 Dir(s)  37,394,849,792 bytes free

(base) C:\Users\DR AJIT KUMAR\NPTEL_Lectures>python Python_Lecture1-1.py
```

and inside this I created a directory, the directory is called NPTEL_Lectures-1 and inside that you have the file Python underscore Lecture hyphen 1 dot py. Now, how do I run this file or execute this file? We can simply say python   space lecture   the name of the file, that is, Python_Lecture1-1 dot py.

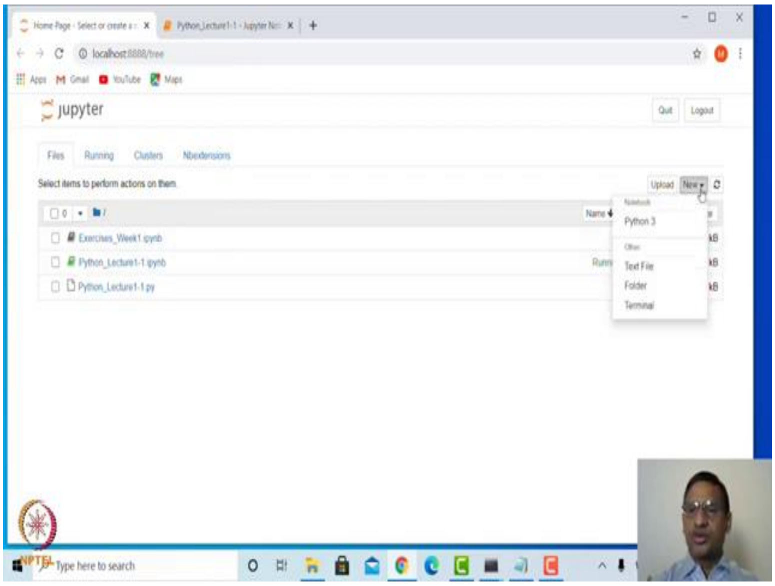
(Refer Slide Time: 03:06)



```
(base) C:\Users\DR AJIT KUMAR>cd NPTEL_Lectures

(base) C:\Users\DR AJIT KUMAR\NPTEL_Lectures>dir
 Volume in drive C is windows10
 Volume Serial Number is 9407-3199

 Directory of C:\Users\DR AJIT KUMAR\NPTEL_Lectures

27-09-2020  06:58    <DIR>          .
27-09-2020  06:58    <DIR>          ..
27-09-2020  06:58    <DIR>          .ipynb_checkpoints
26-09-2020  03:01             1,765 Exercises_Week1.ipynb
27-09-2020  06:55           166,686 Python_Lecture1-1.ipynb
26-09-2020  16:18             1,247 Python_Lecture1-1.py
               3 File(s)        169,698 bytes
               3 Dir(s)  37,394,849,792 bytes free

(base) C:\Users\DR AJIT KUMAR\NPTEL_Lectures>python Python_Lecture1-1.py
Hello World!
Hello World!
6834 57
119 51
25500.0
26139.200128187495
The return with simple interest at 5.5% after 5 years is Rs.25500.0.
The return with compound interest at 5.5% after 5 years is Rs.26139.200128187495.
The return with compound interest at 5.5% after 5 years is Rs.26139.200128.
The return with compound interest at 5.5% after 5 years is Rs.26139.20013.

(base) C:\Users\DR AJIT KUMAR\NPTEL_Lectures>
```

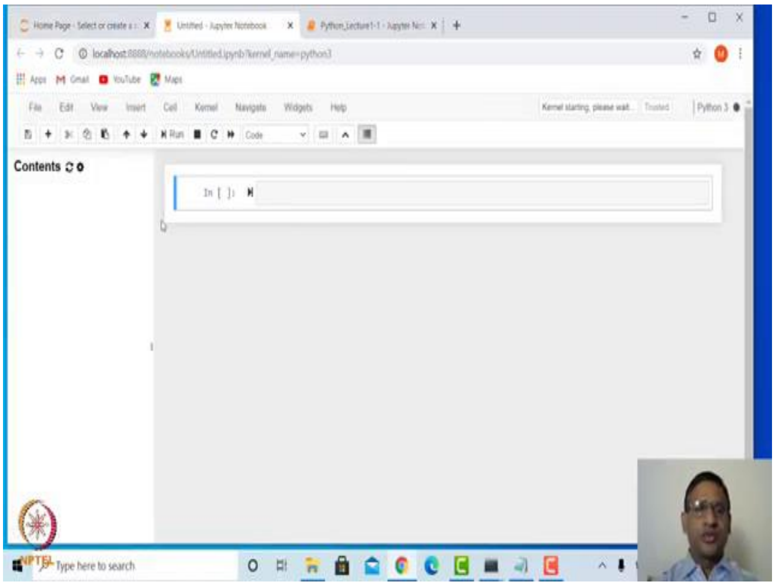
When you hit Enter, it will show all the outputs that we had in   in jupyter notebook.

(Refer Slide Time: 03:24)



Now let us start exploring python as an advanced scientific calculator. So, let us start a new notebook. So, let us go to New, click on Python 3.

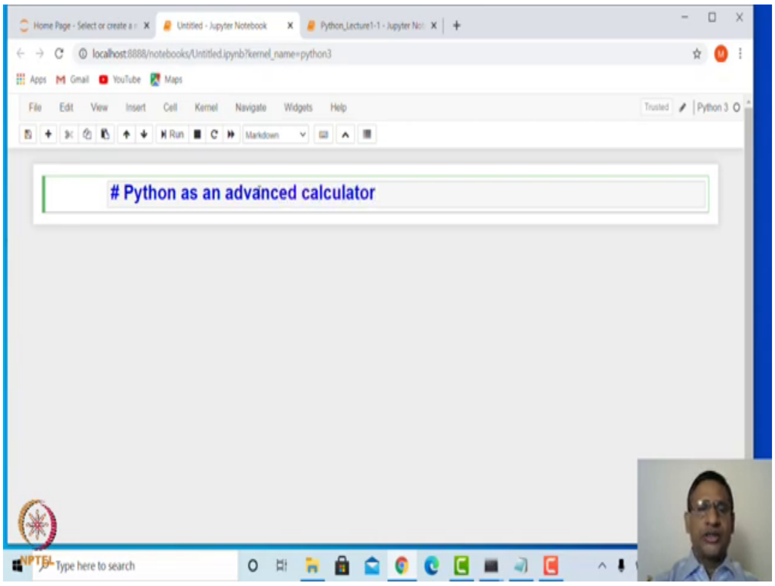
(Refer Slide Time: 03:29)


Home Page - Select or create a
Untitled - Jupyter Notebook
Python_Lecture1-1 - Jupyter No
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
Apps
Gmail
YouTube
Maps
File
Edit
View
Insert
Cell
Kernel
Navigate
Widgets
Help
Kernel starting, please wait...
Trusted
Python 3
Run
Code
Contents
Type here to search

```
In [ ]:
```

This is the notebook. Let me again disable this table of contents.

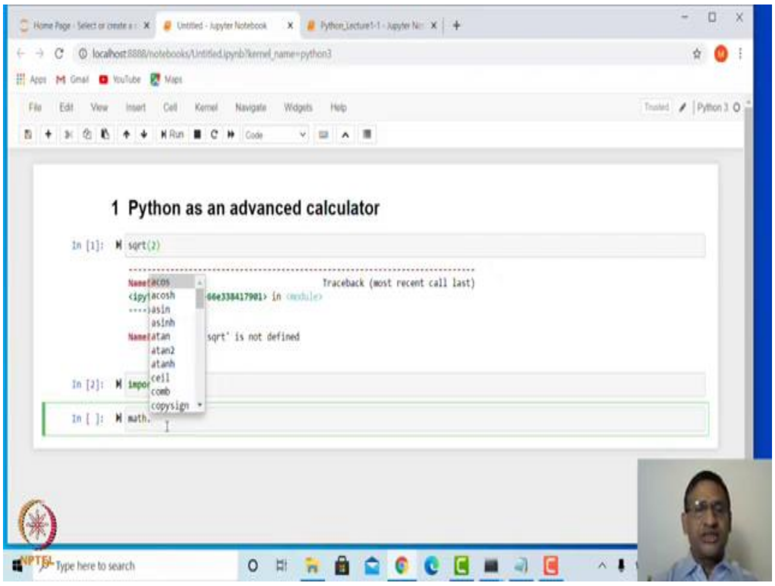
(Refer Slide Time: 03:37)


Home Page - Select or create a
Untitled - Jupyter Notebook
Python_Lecture1-1 - Jupyter No
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
Apps
Gmail
YouTube
Maps
File
Edit
View
Insert
Cell
Kernel
Navigate
Widgets
Help
Trusted
Python 3
Run
Markdown
Type here to search

```
# Python as an advanced calculator
```

Let us give a heading. So, let us go to Markdown and type one hash followed by let us say Python as an advanced calculator and execute this, that is Shift and Enter.

(Refer Slide Time: 04:00)



1 Python as an advanced calculator

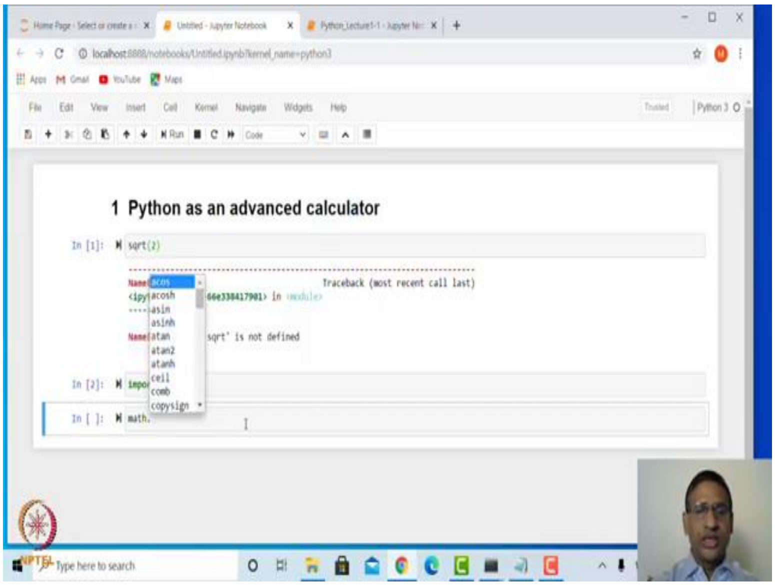
```
In [1]:  sqrt(2)

---------------------------------------------------------------------------
Name acos                               Traceback (most recent call last)
<ipy acosh    66e338417901> in <module>
---- asin
     asinh
Name atan       sqrt' is not defined
     atan2
     atanh
     ceil
In [2]:  impo comb
     copysign
In [ ]:  math.
```

So, now let us see, suppose we want to use a square root function, sqrt of 2, then it will give you an error because, it is not at present in the standard library. So, we need to import these functions from math module. How do I import a module? You can say **import math.**

Now, all the functions, methods which are available in math module can be accessed by typing math dot and then press tab, this is advantage of using editors such as Jupyter Notebook or even Spyder all these functions you can get it by dot tab.

You can go to this down menu. You can see here, there are so many functions available including pi, log10, log2, square root and many other functions.

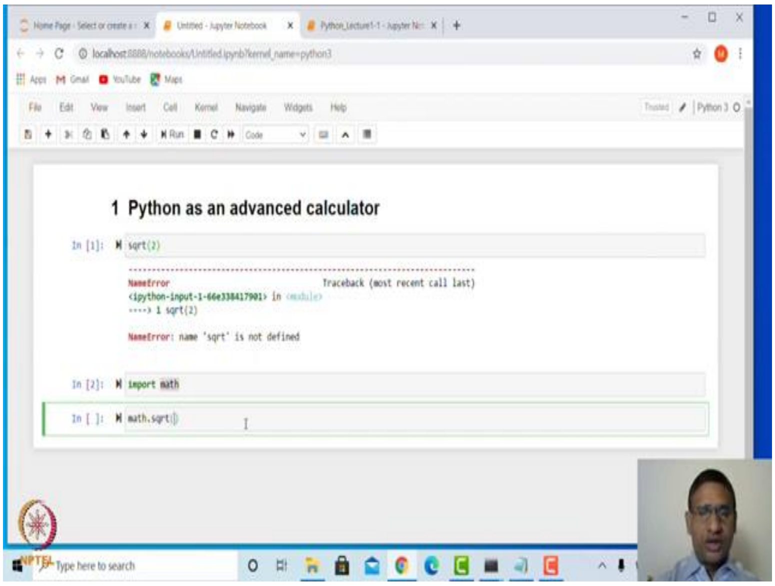
(Refer Slide Time: 05:23)



1 Python as an advanced calculator

```
In [1]:  sqrt(2)

---------------------------------------------------------------------------
Name acos                               Traceback (most recent call last)
<ipy acosh    66e338417901> in <module>
---- asin
     asinh
Name atan       sqrt' is not defined
     atan2
     atanh
     ceil
In [2]:  impo comb
     copysign
In [ ]:  math.
```

(Refer Slide Time: 05:24)

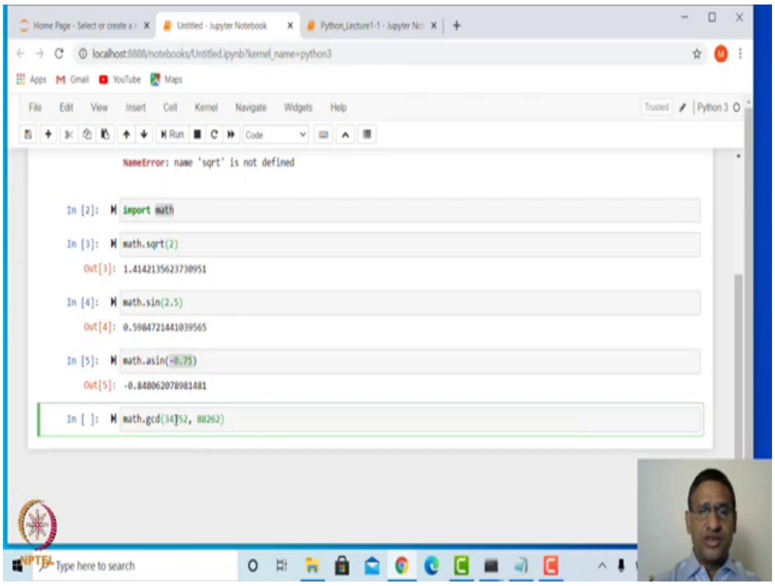


1 Python as an advanced calculator

```
In [1]:  sqrt(2)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-66e338417901> in <module>
----> 1 sqrt(2)

NameError: name 'sqrt' is not defined

In [2]:  import math

In [ ]:  math.sqrt()
```

So, for example, if I want to find square root, I have to say, **math.sqrt(2)**. It will give you the answer.

(Refer Slide Time: 05:31)



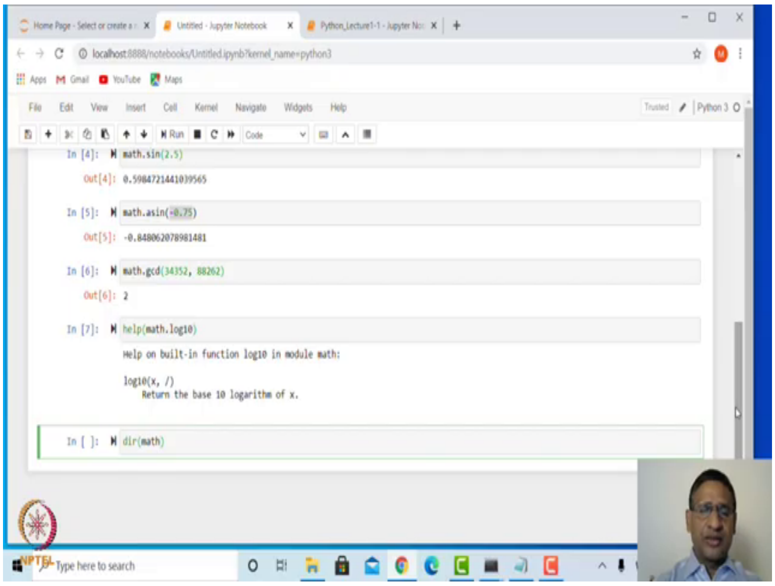
```
NameError: name 'sqrt' is not defined

In [2]:  import math

In [3]:  math.sqrt(2)
Out[3]:  1.4142135623730951

In [4]:  math.sin(2.5)
Out[4]:  0.5984721441039565

In [5]:  math.asin(-0.75)
Out[5]:  -0.848062078981481

In [ ]:  math.gcd(34[illegible]52, 88262)
```

Similarly, if I want to make use of sine function, I have to say **math.sin(2.5). T**his is gives you the answer. 2.5 here is taken in radian. If you want in degree you can convert this into degree and then find out the value.

You can say math dot sin inverse. So, we will say asin that is, arc sin or sin inverse of let us say minus 0.75. Since sin values lies between minus 1 and 1, the argument has to be always between minus 1 and 1.

Similarly if you want to find, let us say, GCD, you can say, math dot gcd of two integers, let us say one integer is this, another integer is this, then we get the gcd of these two integers.

(Refer Slide Time: 06:32)

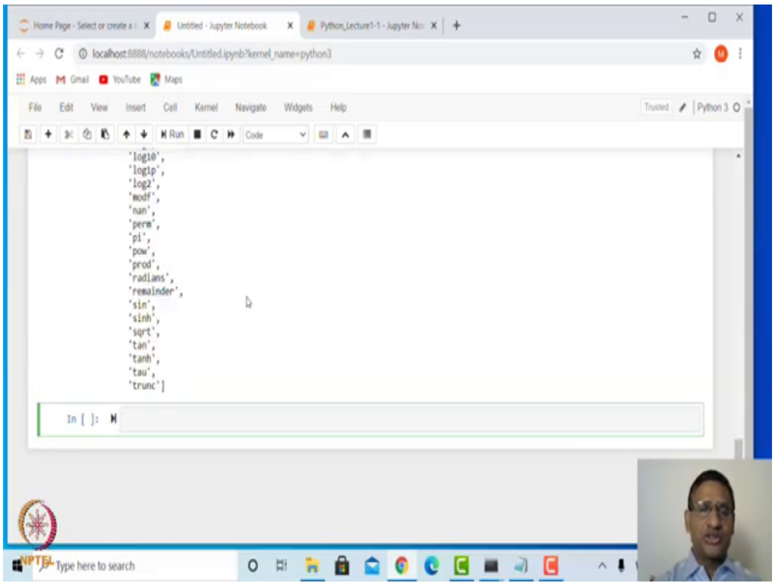


```
In [4]:  math.sin(2.5)
Out[4]:  0.5984721441039565

In [5]:  math.asin(-0.75)
Out[5]:  -0.848062078981481

In [6]:  math.gcd(34352, 88262)
Out[6]:  2

In [7]:  help(math.log10)
         Help on built-in function log10 in module math:

         log10(x, /)
             Return the base 10 logarithm of x.

In [ ]:  dir(math)
```

If you want to   take help on any of these functions, you can say help and I can say, math dot for example, we saw something called log10. So, let us explore this. What does it say? It says that help on built in function log 10 in the module math and how it is used?

Log10 in the bracket you write the value of x and it returns the base 10 of log of x. You can explore all the functions, which are there in math module.You can even list by typing **dir(math)**. You will get the list of all the functions.

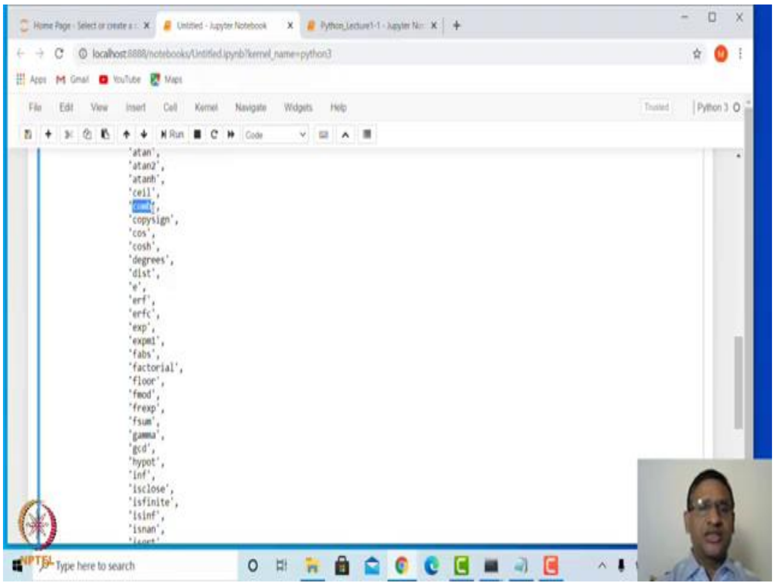
(Refer Slide Time: 07:23)


Untitled - Jupyter Notebook
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
File Edit View Insert Cell Kernel Navigate Widgets Help
Trusted
Python 3
Type here to search

```
 'log10',
 'log1p',
 'log2',
 'modf',
 'nan',
 'perm',
 'pi',
 'pow',
 'prod',
 'radians',
 'remainder',
 'sin',
 'sinh',
 'sqrt',
 'tan',
 'tanh',
 'tau',
 'trunc']

In [ ]:
```

For example, there is something for permutation, **perm** and things like.

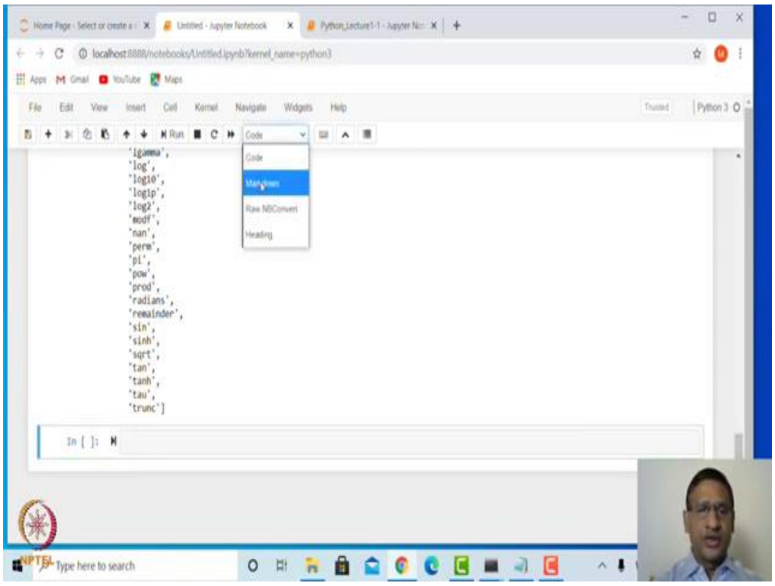
(Refer Slide Time: 07:35)


Untitled - Jupyter Notebook
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
File Edit View Insert Cell Kernel Navigate Widgets Help
Trusted
Python 3
Type here to search

```
 'atan',
 'atan2',
 'atanh',
 'ceil',
 'comb',
 'copysign',
 'cos',
 'cosh',
 'degrees',
 'dist',
 'e',
 'erf',
 'erfc',
 'exp',
 'expm1',
 'fabs',
 'factorial',
 'floor',
 'fmod',
 'frexp',
 'fsum',
 'gamma',
 'gcd',
 'hypot',
 'inf',
 'isclose',
 'isfinite',
 'isinf',
 'isnan',
```

Suppose, we want to explore this function permutation. You will also see something called combination. Thus we have permutation and combination function. Let us say, we want to explore permutation and combination. How do we do?

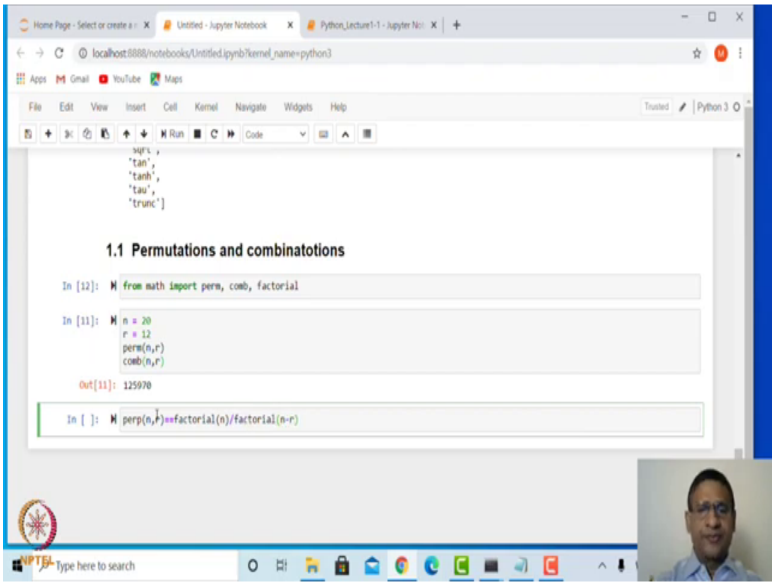
(Refer Slide Time: 07:47)



```
 'lgamma',
 'log',
 'log10',
 'log1p',
 'log2',
 'modf',
 'nan',
 'perm',
 'pi',
 'pow',
 'prod',
 'radians',
 'remainder',
 'sin',
 'sinh',
 'sqrt',
 'tan',
 'tanh',
 'tau',
 'trunc']

In [ ]:
```

First let me create a sub heading called permutation and combination. So, I will get double hash Permutation and combinations. Let me run this.

(Refer Slide Time: 07:53)



```
 'sqrt',
 'tan',
 'tanh',
 'tau',
 'trunc']
```

1.1 Permutations and combinatotions

```
In [12]: from math import perm, comb, factorial

In [11]: n = 20
         r = 12
         perm(n,r)
         comb(n,r)

Out[11]: 125970

In [ ]: perp(n,r)==factorial(n)/factorial(n-r)
```

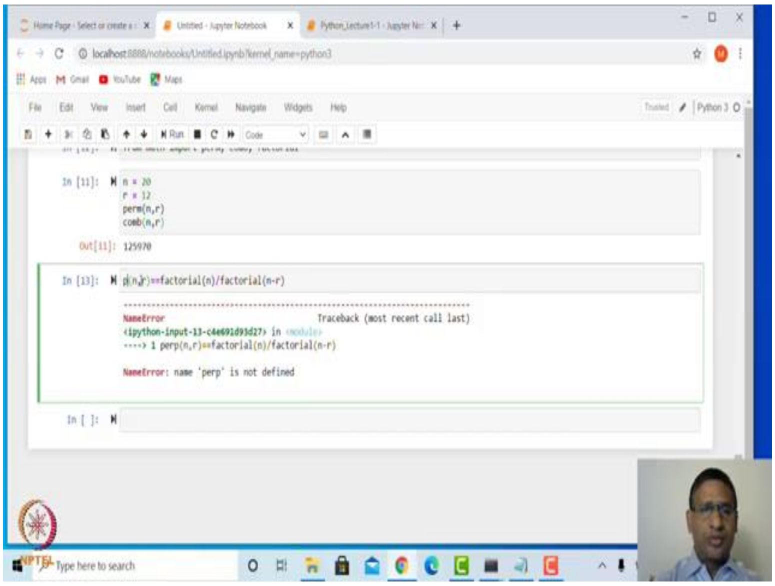
We want to make use of the two functions, namely, perm and comb repeatedly. I do not want always keep on typing math dot perm, math dot comb. There is a way to import these functions alone and how do we that? We can say from math import perm and comma comb and then run this.

Now perm and comb functions are available, we do not have to type math dot perm, we can simply say perm and it will work. So, let us take a number, let us say, n is equal to, for example, 20 and r is equal to, let us say 12. Then we can say **perm(n,r)** and it gives you this value. This is permutation of any 12 objects out of 20 objects. That is choose any 12 objects out of 20 objects and then permute them. This is what you get. Similarly you can find **comb(n,r),** that is choose any r objects out of n objects.

You must have already seen what are the formulas for permutations and combinations. So, combination when, say n C r, this is equal to n factorial, divided by r factorial, divided by n minus r factorial. And permutation n P r, is equal to n factorial divided by n minus r factorial.

Let us try to verify this. How do I verify this? So, let me also go back and also add factorial function. So, factorial function execute this again.

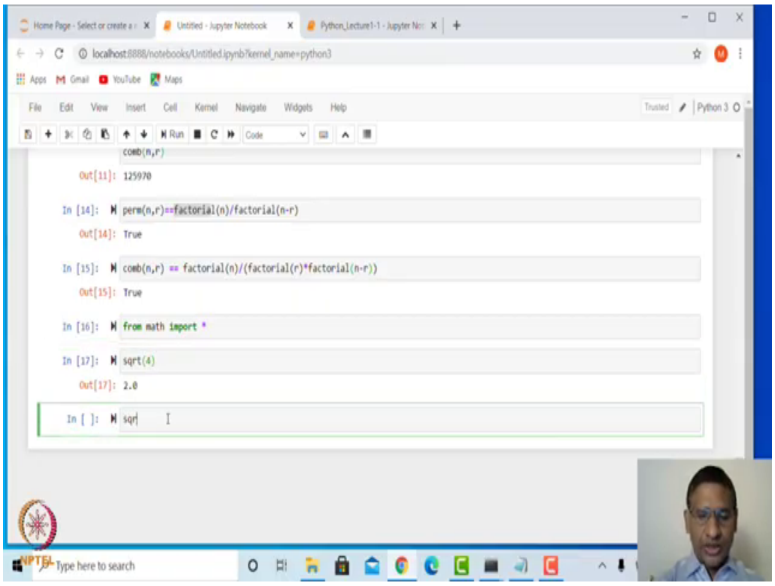
(Refer Slide Time: 10:40)



```
In [12]: from math import perm, comb, factorial

In [11]: n = 20
         r = 12
         perm(n,r)
         comb(n,r)

Out[11]: 125970

In [13]: p(n,r)==factorial(n)/factorial(n-r)

         ---------------------------------------------------------------------------
         NameError                                 Traceback (most recent call last)
         <ipython-input-13-c4e691d93d27> in <module>
         ----> 1 perp(n,r)==factorial(n)/factorial(n-r)

         NameError: name 'perp' is not defined

In [ ]:
```

Then we can say, perm(n,r) double equal to factorial of n divided by factorial of n minus r. The answer is True.

(Refer Slide Time: 10:46)

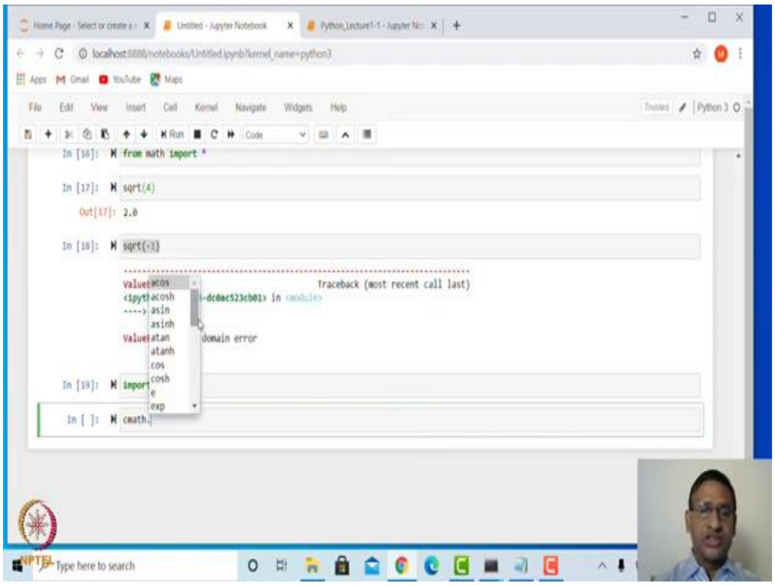


```
    comb(n,r)
Out[11]: 125970

In [14]: perm(n,r)==factorial(n)/factorial(n-r)
Out[14]: True

In [15]: comb(n,r) == factorial(n)/(factorial(r)*factorial(n-r))
Out[15]: True

In [16]: from math import *

In [17]: sqrt(4)
Out[17]: 2.0

In [ ]: sqr
```

Similarly, we can say, combination that is **comb(n,r)**, n choose r, this is equal to, is equal to factorial of n divided by factorial of r, into factorial of n minus r and the answer is True. So, we are able to verify this function.

Now, suppose we want to find square root of some number. We have not imported square root function therefore, we need to say math dot sqrt, but one can also import all the functions which are available in math module. How do we do that? Instead of giving listing all the functions that you have inside math module, what you can say is, **from math import *.** Now, all the functions inside math module will be available, * means everything.

(Refer Slide Time: 12:19)



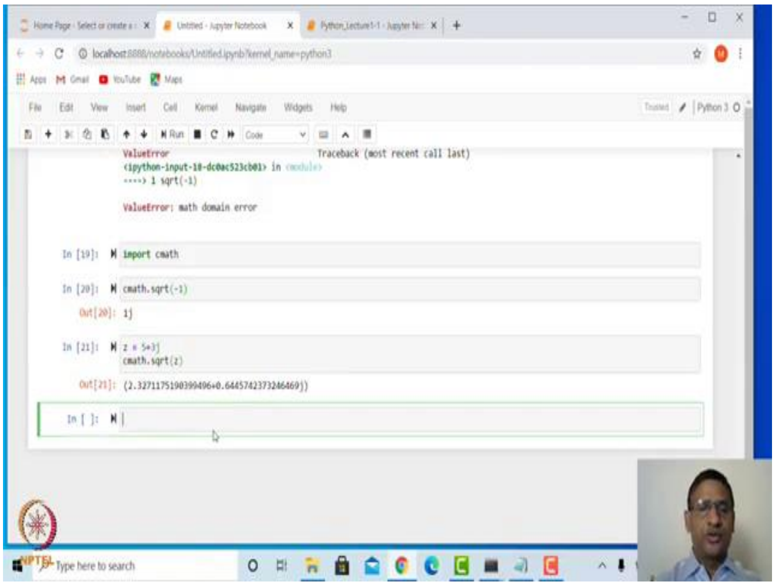
```
In [16]:  from math import *

In [17]:  sqrt(4)
Out[17]:  2.0

In [18]:  sqrt(-1)

          ---------------------------------------------------------------
          Valuel acos                          Traceback (most recent call last)
          <ipyth acosh   -dc0ac523cb01> in <module>
          ----> asin
                 asinh
          Valuel atan    domain error
                 atanh
                 cos
                 cosh
In [19]:  import e
                 exp

In [ ]:   cmath.
```

So, let us see, if I say sqrt of let us say 4, you get the answer. If I say sqrt of negative of 1, this is math domain error, because this sqrt in math module takes only the non-negative argument. If you want to find square root of negative number you have to make use of **cmath** module, that we will look at.

So, how do I how do I import? Let us say, I will say **import cmath.** Inside this, for example, if you look at cmath dot and then press tab, you will see all the functions which are there in math module and some more. So, for example, you can also find square root of minus 1 using this.

Now, let us say cmath dot sqrt of minus 1, you will get the answer. This j stands for I, that is, imaginary number, square root of minus 1.

(Refer Slide Time: 13:08)

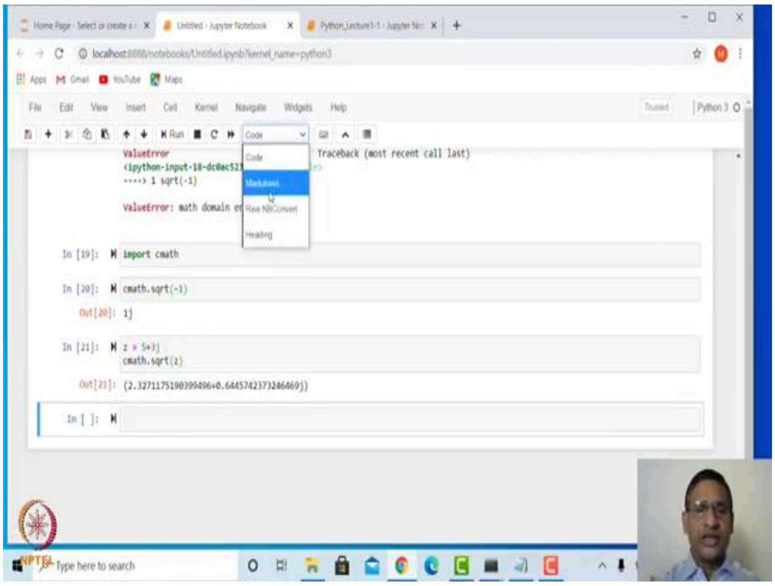


```
ValueError                                Traceback (most recent call last)
<ipython-input-18-dc0ac523cb01> in <module>
----> 1 sqrt(-1)

ValueError: math domain error

In [19]: import cmath

In [20]: cmath.sqrt(-1)
Out[20]: 1j

In [21]: z = 5+3j
         cmath.sqrt(z)
Out[21]: (2.3271175190399496+0.6445742373246469j)

In [ ]:
```

Similarly, suppose I have   a complex number z is equal to let us say 5 plus 3j, this is, 5 plus 3i and we want to find out square root of this, then we can simply say   cmath.sqrt(z), you will get square root of a complex number z.

Similarly, you can find out exponential of a complex number, trigonometrical function of complex number and all the functions which   are there inside cmath module.   Now, let us make use of these modules, math module and cmath module to find, let us say roots of a quadratic.

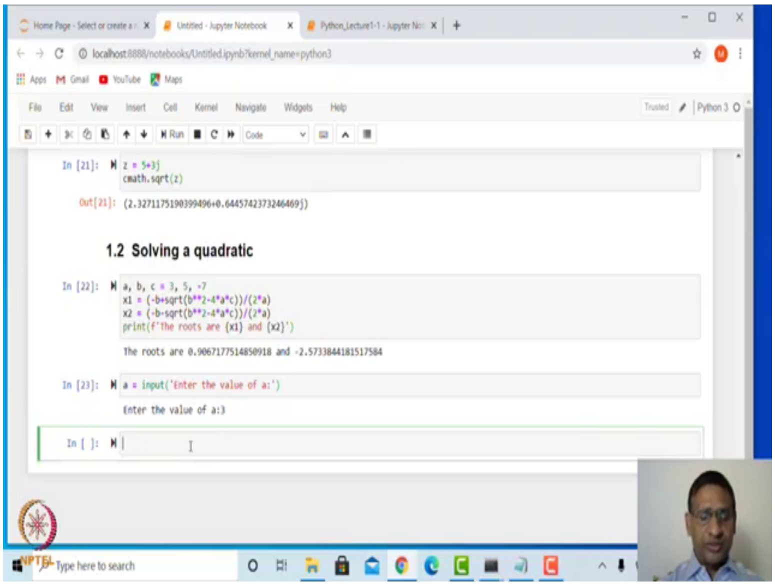
(Refer Slide Time: 14:02)



```
ValueError                                Traceback (most recent call last)
<ipython-input-18-dc0ac523cb01> in <module>
----> 1 sqrt(-1)

ValueError: math domain error

In [19]: import cmath

In [20]: cmath.sqrt(-1)
Out[20]: 1j

In [21]: z = 5+3j
         cmath.sqrt(z)
Out[21]: (2.3271175190399496+0.6445742373246469j)

In [ ]:
```

Let me again create a sub heading, I will say   double hash Solving a quadratic.

(Refer Slide Time: 14:11)



```
In [21]:  z = 5+3j
          cmath.sqrt(z)

Out[21]:  (2.3271175190399496+0.6445742373246469j)
```

1.2 Solving a quadratic

```
In [22]:  a, b, c = 3, 5, -7
          x1 = (-b+sqrt(b**2-4*a*c))/(2*a)
          x2 = (-b-sqrt(b**2-4*a*c))/(2*a)
          print(f'The roots are {x1} and {x2}')

          The roots are 0.9067177514850918 and -2.5733844181517584

In [23]:  a = input('Enter the value of a:')

          Enter the value of a:3

In [ ]:   |
```

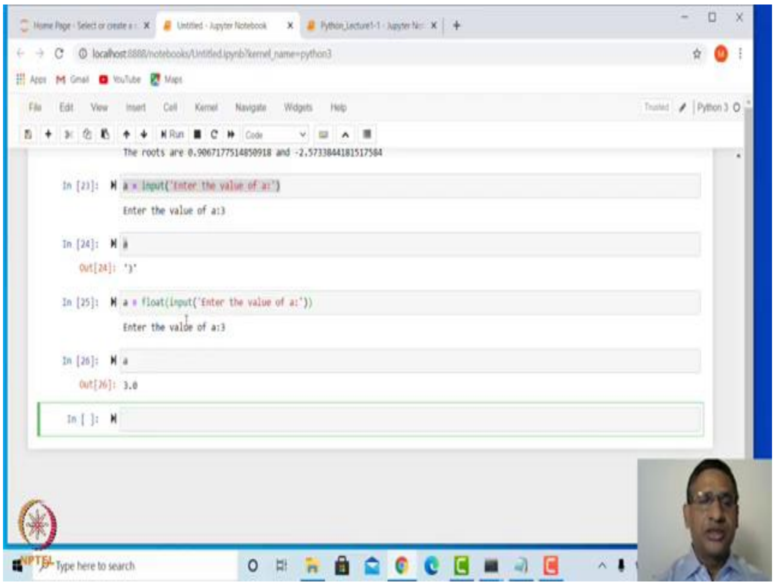
So, let us take a quadratic ax square plus bx plus c for various values of a, b and c. So, one way is, we can define a comma b comma c   is equal to some number, let us say 3 comma 5 comma minus 7. So, we are looking at 3x square plus 5x minus 7 is equal to 0 and we want to find roots of this.

So, let us say the first root, I call as x1, we know the roots of a quadratic ax square plus bx plus c are minus b plus minus square root b square minus 4ac upon 2a. So, let us write x1 as minus b   plus   sqrt square root of b double star 2 minus   4 star a star c, whole thing divided by   2 into a.

Similarly, x2 is   equal to minus b minus this. Let us print what are the roots. Let me again use f string, f inside single quote the roots are are x1 and   x2, close the single quote. So these are the roots of this this quadratic function.

Now, of course, if you want to keep changing this 3, 5, minus 7 etc, the value of a and b and c one can do that. Another way of doing the same thing, is you can actually ask user to input this from the keyboard. How do I do that? I can say a equal to input and you can give some message here. So, let us say the value of a. It will give you an input box and where you can type the value suppose a as 3. Now press enter, the value of a is 3.

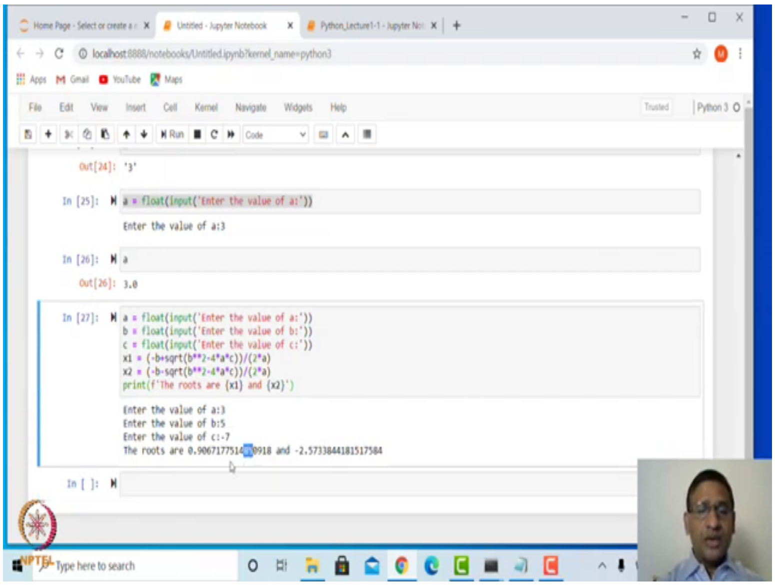
(Refer Slide Time: 17:13)



```
The roots are 0.9067177514850918 and -2.5733844181517584

In [23]:  a = input('Enter the value of a:')

          Enter the value of a:3

In [24]:  a

Out[24]:  '3'

In [25]:  a = float(input('Enter the value of a:'))

          Enter the value of a:3

In [26]:  a

Out[26]:  3.0

In [ ]:
```

Now, let me see what is a? So, a is actually a character, but we do not want character, we want a real number. So, you need to convert this into floating point number or if you want integer, then you can do that. So, how do I do that? Let me let me, copy this fellow and say input and then we will have to say float   and in the bracket you type the value. Now, let us say this is equal to 3. Now if you ask what is a? This is a real number. So, like this you can read a, b, c. Let us let us read all of these in a single cell.

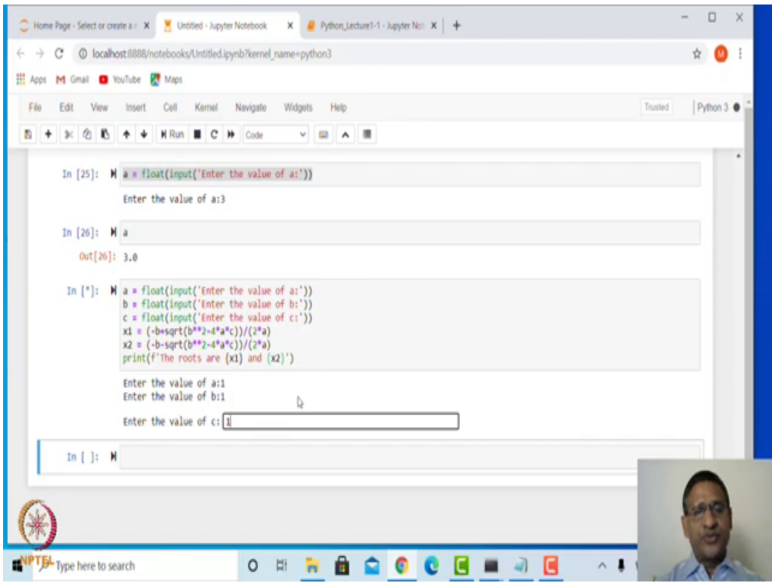
(Refer Slide Time: 18:06)



```
Out[24]:  '3'

In [25]:  a = float(input('Enter the value of a:'))

          Enter the value of a:3

In [26]:  a

Out[26]:  3.0

In [27]:  a = float(input('Enter the value of a:'))
          b = float(input('Enter the value of b:'))
          c = float(input('Enter the value of c:'))
          x1 = (-b+sqrt(b**2-4*a*c))/(2*a)
          x2 = (-b-sqrt(b**2-4*a*c))/(2*a)
          print(f'The roots are {x1} and {x2}')

          Enter the value of a:3
          Enter the value of b:5
          Enter the value of c:-7
          The roots are 0.9067177514850918 and -2.5733844181517584

In [ ]:
```

This is a, this is b and this is c. So, here also it will be a, it will b, and this is c. Once you have read this, then let us find out value of this x1, x2. So, let me paste it here, print and then let me execute this. Let us say value of a is 3, value of b is 5 and value of c is minus 7, then you will get the roots. So, this is how you can read the value of a variable using input.

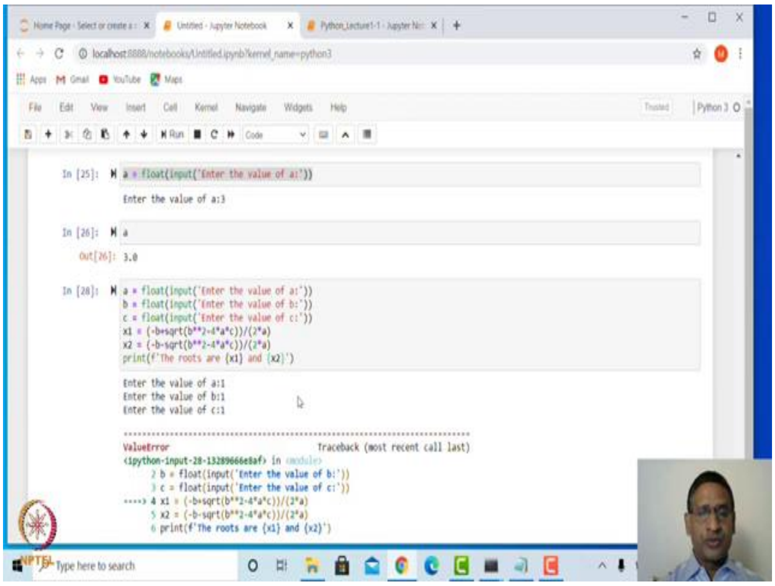
(Refer Slide Time: 18:59)



```
In [25]:  a = float(input('Enter the value of a:'))

          Enter the value of a:3

In [26]:  a

Out[26]:  3.0

In [*]:   a = float(input('Enter the value of a:'))
          b = float(input('Enter the value of b:'))
          c = float(input('Enter the value of c:'))
          x1 = (-b+sqrt(b**2-4*a*c))/(2*a)
          x2 = (-b-sqrt(b**2-4*a*c))/(2*a)
          print(f'The roots are {x1} and {x2}')

          Enter the value of a:1
          Enter the value of b:1

          Enter the value of c: 1

In [ ]:
```

Let us let us go and run this again. Value of a is let us say 1, value of b is also 1, value of c is also 1 and let us see what we get. So, this is x square plus x plus 1 equal to 0 and in this case if you look at the value of b square minus 4ac will be negative. So, square root from math module will not be able to find. So, let us see what you get?

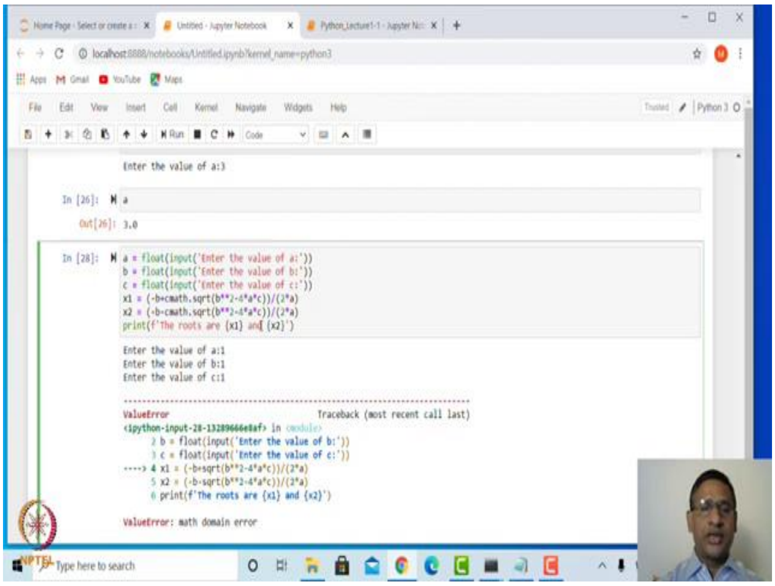
(Refer Slide Time: 19:20)



```
In [25]:  a = float(input('Enter the value of a:'))

          Enter the value of a:3

In [26]:  a

Out[26]:  3.0

In [28]:  a = float(input('Enter the value of a:'))
          b = float(input('Enter the value of b:'))
          c = float(input('Enter the value of c:'))
          x1 = (-b+sqrt(b**2-4*a*c))/(2*a)
          x2 = (-b-sqrt(b**2-4*a*c))/(2*a)
          print(f'The roots are {x1} and {x2}')

          Enter the value of a:1
          Enter the value of b:1
          Enter the value of c:1

          ---------------------------------------------------------------------------
          ValueError                                Traceback (most recent call last)
          <ipython-input-28-13289666e8af> in <module>
                2 b = float(input('Enter the value of b:'))
                3 c = float(input('Enter the value of c:'))
          ----> 4 x1 = (-b+sqrt(b**2-4*a*c))/(2*a)
                5 x2 = (-b-sqrt(b**2-4*a*c))/(2*a)
                6 print(f'The roots are {x1} and {x2}')
```

Again you can see here you get an error and it says the domain error, that domain error because this is finding square root from math module which takes only the non-negative real number.

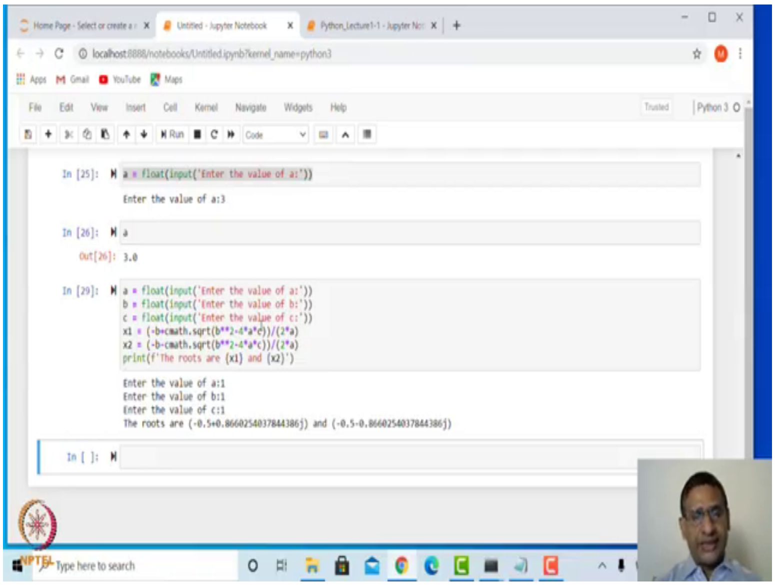
(Refer Slide Time: 19:38)



```
          Enter the value of a:3

In [26]:  a

Out[26]:  3.0

In [28]:  a = float(input('Enter the value of a:'))
          b = float(input('Enter the value of b:'))
          c = float(input('Enter the value of c:'))
          x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
          x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
          print(f'The roots are {x1} and {x2}')

          Enter the value of a:1
          Enter the value of b:1
          Enter the value of c:1

          ---------------------------------------------------------------------------
          ValueError                                Traceback (most recent call last)
          <ipython-input-28-13289666e8af> in <module>
                2 b = float(input('Enter the value of b:'))
                3 c = float(input('Enter the value of c:'))
          ----> 4 x1 = (-b+sqrt(b**2-4*a*c))/(2*a)
                5 x2 = (-b-sqrt(b**2-4*a*c))/(2*a)
                6 print(f'The roots are {x1} and {x2}')

          ValueError: math domain error
```

Instead of this let me let me go back and change this square root from cmath dot sqrt and here also you can say cmath dot square root and now it will be not a problem.

(Refer Slide Time: 19:54)

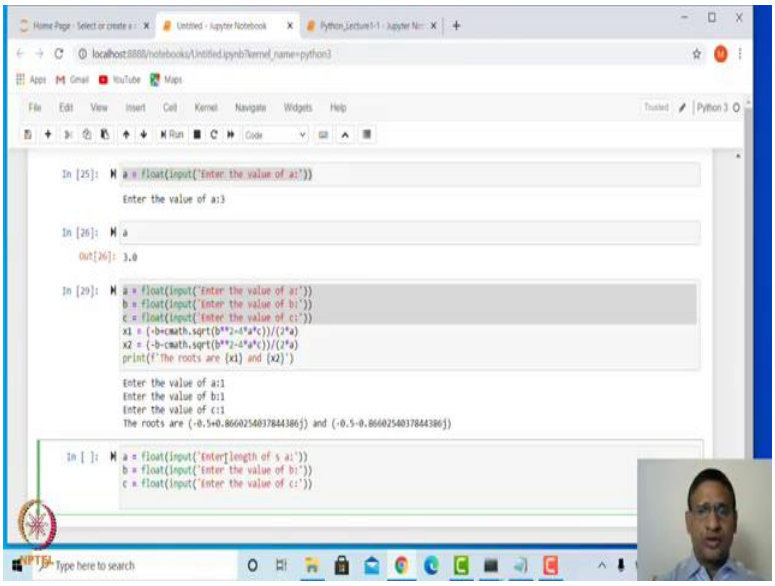


```
In [25]: a = float(input('Enter the value of a:'))

         Enter the value of a:3

In [26]: a

Out[26]: 3.0

In [29]: a = float(input('Enter the value of a:'))
         b = float(input('Enter the value of b:'))
         c = float(input('Enter the value of c:'))
         x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
         x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
         print(f'The roots are {x1} and {x2}')

         Enter the value of a:1
         Enter the value of b:1
         Enter the value of c:1
         The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)

In [ ]:
```

So, 1, 1, 1 and you can see here it will give you the roots, even if the discriminant is negative. Later on we will write our own program to print all the roots of a quadratic. It may involve checking whether discriminant is non-negative or negative and things like that.

Similarly suppose you want to find the area of a triangle using Heron’s formula, let us say if a, b, c are sides of a triangle, and we know that area of this triangle is given by square root of s into s minus a into s minus b into s minus c, where s is the semi perimeter.

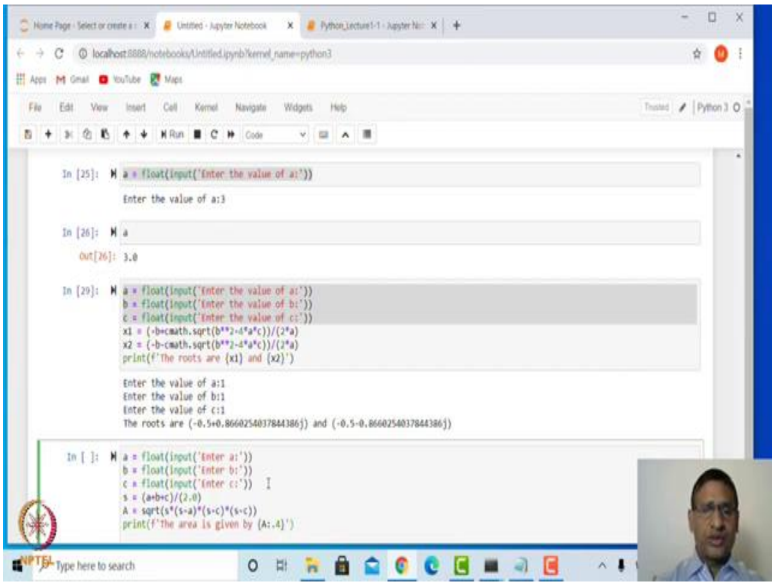
(Refer Slide Time: 20:41)



```
In [25]: a = float(input('Enter the value of a:'))

         Enter the value of a:3

In [26]: a

Out[26]: 3.0

In [29]: a = float(input('Enter the value of a:'))
         b = float(input('Enter the value of b:'))
         c = float(input('Enter the value of c:'))
         x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
         x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
         print(f'The roots are {x1} and {x2}')

         Enter the value of a:1
         Enter the value of b:1
         Enter the value of c:1
         The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)

In [ ]: a = float(input('Enter length of s a:'))
        b = float(input('Enter the value of b:'))
        c = float(input('Enter the value of c:'))
```

So, how we can do that? We can again copy these input a, b, c. So, let me say enter a, the length of a, similarly enter b, enter c.

(Refer Slide Time: 21:02)

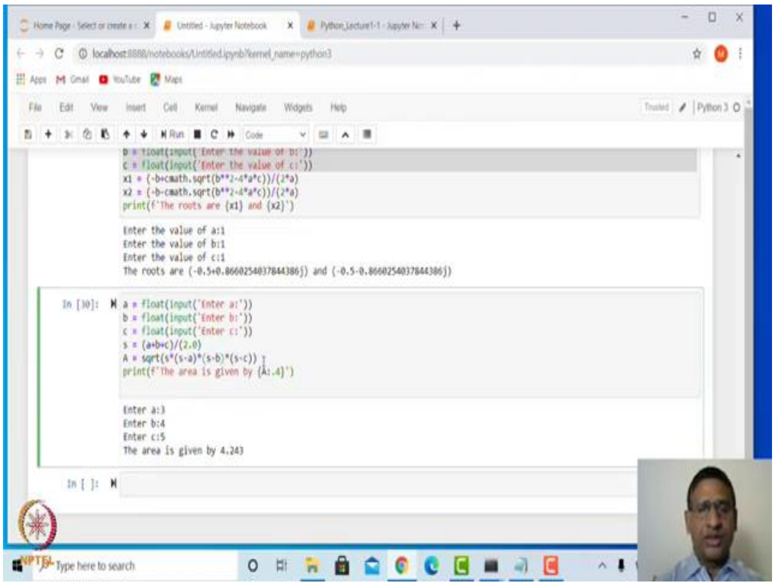


```
In [25]:  a = float(input('Enter the value of a:'))

          Enter the value of a:3

In [26]:  a

Out[26]:  3.0

In [29]:  a = float(input('Enter the value of a:'))
          b = float(input('Enter the value of b:'))
          c = float(input('Enter the value of c:'))
          x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
          x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
          print(f'The roots are {x1} and {x2}')

          Enter the value of a:1
          Enter the value of b:1
          Enter the value of c:1
          The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)

In [ ]:   a = float(input('Enter a:'))
          b = float(input('Enter b:'))
          c = float(input('Enter c:'))
          s = (a+b+c)/(2.0)
          A = sqrt(s*(s-a)*(s-c)*(s-c))
          print(f'The area is given by {A:.4}')
```

After that you calculate the semi perimeter that is, s equals to a plus b plus c, the whole thing divided by 2.0.

The area, the capital A is equal to sqrt s of s minus a into s minus c, the star is important in between, if you do not put star it will give you an error. Let us print that. Ww will say f single quote the area is given by, let us say this is A and colon, let me put 0.4, so, it will print 4 decimal places, close the single quote and that is it.

(Refer Slide Time: 22:17)

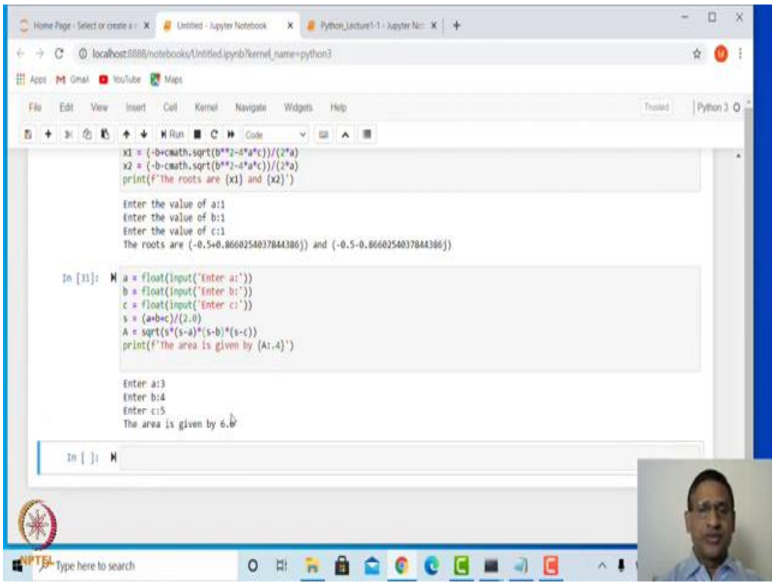


```
b = float(input('Enter the value of b:'))
c = float(input('Enter the value of c:'))
x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
print(f'The roots are {x1} and {x2}')

Enter the value of a:1
Enter the value of b:1
Enter the value of c:1
The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)
```

```
In [30]: a = float(input('Enter a:'))
         b = float(input('Enter b:'))
         c = float(input('Enter c:'))
         s = (a+b+c)/(2.0)
         A = sqrt(s*(s-a)*(s-b)*(s-c))
         print(f'The area is given by {A:.4}')

Enter a:3
Enter b:4
Enter c:5
The area is given by 4.243
```

So, it will ask for so, a is let us say 3, b is 4, and c is 5, and the area is.. I think there is some problem square root s is a plus b plus c by 2. s into s minus a into s minus b into s minus c.

(Refer Slide Time: 22:38)

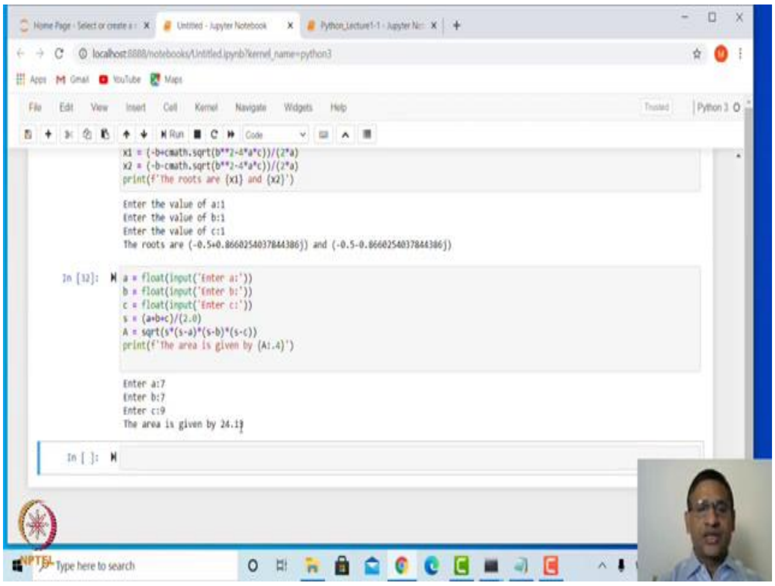


```
x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
print(f'The roots are {x1} and {x2}')

Enter the value of a:1
Enter the value of b:1
Enter the value of c:1
The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)
```

```
In [31]: a = float(input('Enter a:'))
         b = float(input('Enter b:'))
         c = float(input('Enter c:'))
         s = (a+b+c)/(2.0)
         A = sqrt(s*(s-a)*(s-b)*(s-c))
         print(f'The area is given by {A:.4}')

Enter a:3
Enter b:4
Enter c:5
The area is given by 6.0
```

Let us say 3, 4, 5 and the area is this. If I run this again and input let us say 7, let us say 7 and this is let us say 9 and this is the area.

(Refer Slide Time: 22:45)

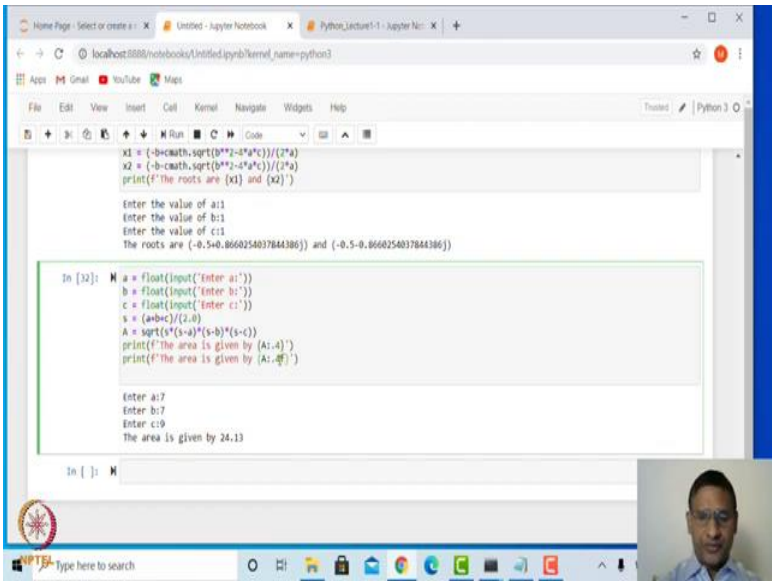


```
x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
print(f'The roots are {x1} and {x2}')

Enter the value of a:1
Enter the value of b:1
Enter the value of c:1
The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)
```

```
In [32]: a = float(input('Enter a:'))
         b = float(input('Enter b:'))
         c = float(input('Enter c:'))
         s = (a+b+c)/(2.0)
         A = sqrt(s*(s-a)*(s-b)*(s-c))
         print(f'The area is given by {A:.4}')

Enter a:7
Enter b:7
Enter c:9
The area is given by 24.13
```

If I run this again and input let us say 7; let us say 7 and this is let us say 9 and this is the area, right. So, this actually when you write 0.4 this gives you the four significant digits.

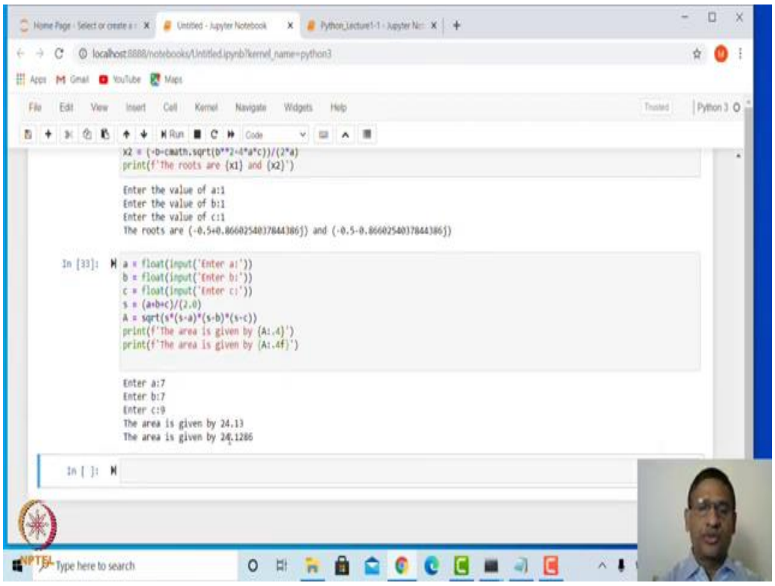
(Refer Slide Time: 23:10)



```
x1 = (-b+cmath.sqrt(b**2-4*a*c))/(2*a)
x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
print(f'The roots are {x1} and {x2}')

Enter the value of a:1
Enter the value of b:1
Enter the value of c:1
The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)
```

```
In [32]: a = float(input('Enter a:'))
         b = float(input('Enter b:'))
         c = float(input('Enter c:'))
         s = (a+b+c)/(2.0)
         A = sqrt(s*(s-a)*(s-b)*(s-c))
         print(f'The area is given by {A:.4}')
         print(f'The area is given by {A:.4f}')

Enter a:7
Enter b:7
Enter c:9
The area is given by 24.13
```

(Refer Slide Time: 23:16)

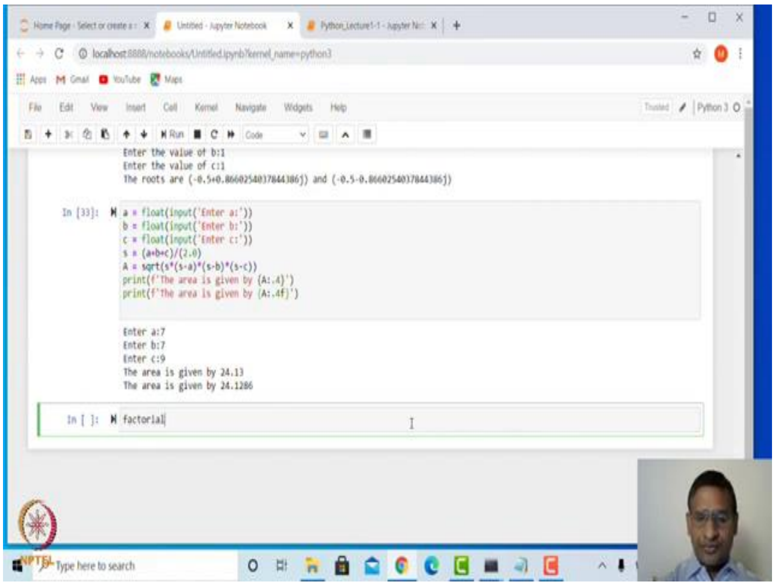


```
x2 = (-b-cmath.sqrt(b**2-4*a*c))/(2*a)
print(f'The roots are {x1} and {x2}')

Enter the value of a:1
Enter the value of b:1
Enter the value of c:1
The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)

In [33]: a = float(input('Enter a:'))
         b = float(input('Enter b:'))
         c = float(input('Enter c:'))
         s = (a+b+c)/(2.0)
         A = sqrt(s*(s-a)*(s-b)*(s-c))
         print(f'The area is given by {A:.4}')
         print(f'The area is given by {A:.4f}')

Enter a:7
Enter b:7
Enter c:9
The area is given by 24.13
The area is given by 24.1286

In [ ]:
```

Now, let us run this again this is let us say 7, 7, 9 you get this four decimal places. This f string formatting provides you all these options. Try to explore all these functions available inside math module and cmath module. Go through one by one and see how we can make use of these functions to do various scientific calculations.

If you have seen the scientific calculator, you must have seen other functionality, that it has and of course, Python will have much better capability than the scientific calculator.

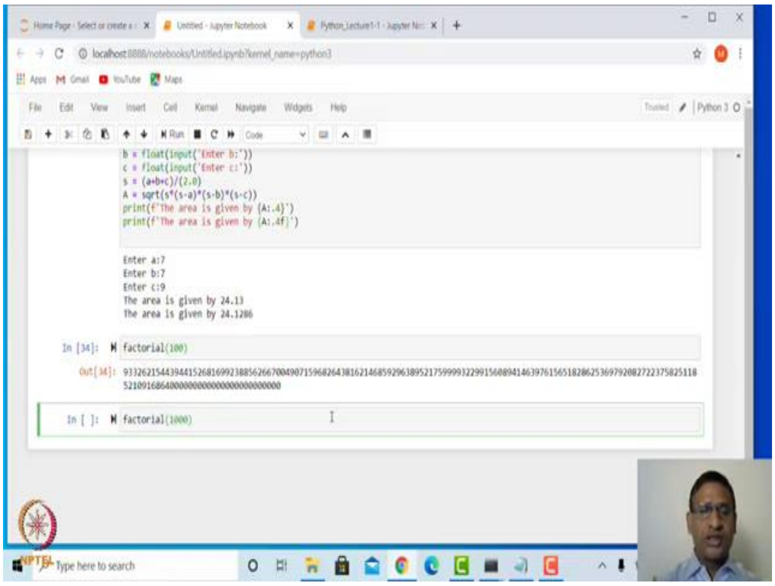
(Refer Slide Time: 23:59)



```
Enter the value of b:1
Enter the value of c:1
The roots are (-0.5+0.8660254037844386j) and (-0.5-0.8660254037844386j)

In [33]: a = float(input('Enter a:'))
         b = float(input('Enter b:'))
         c = float(input('Enter c:'))
         s = (a+b+c)/(2.0)
         A = sqrt(s*(s-a)*(s-b)*(s-c))
         print(f'The area is given by {A:.4}')
         print(f'The area is given by {A:.4f}')

Enter a:7
Enter b:7
Enter c:9
The area is given by 24.13
The area is given by 24.1286

In [ ]: factorial
```

For example, if I say factorial of a large number let us say factorial of 100, it computes.

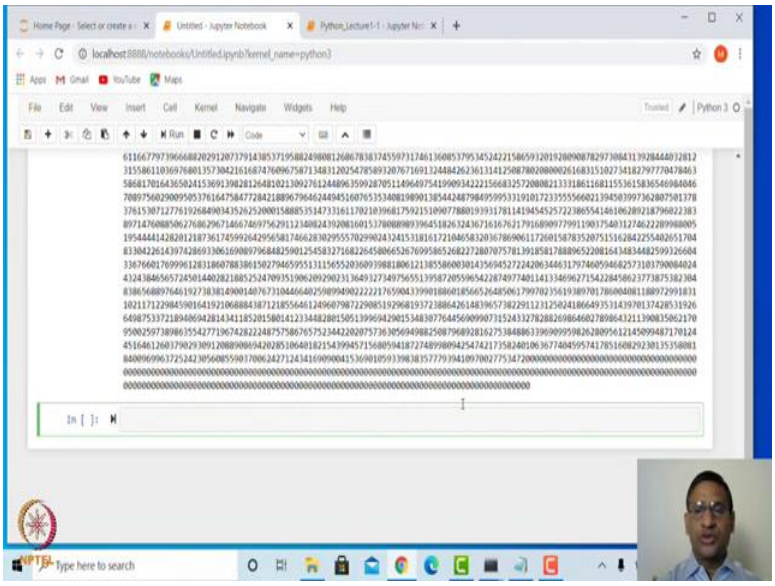
(Refer Slide Time: 24:06)


localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
File Edit View Insert Cell Kernel Navigate Widgets Help
Trusted
Python 3
Run
Code
Type here to search

```
b = float(input('Enter b:'))
c = float(input('Enter c:'))
s = (a+b+c)/(2.0)
A = sqrt(s*(s-a)*(s-b)*(s-c))
print(f'The area is given by {A:.4}')
print(f'The area is given by {A:.4f}')

Enter a:7
Enter b:7
Enter c:9
The area is given by 24.13
The area is given by 24.1286

In [34]: factorial(100)

Out[34]: 93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000000000

In [ ]: factorial(1000)
```

If I say factorial of even let us say 1000, then also it gives you the value, whereas, in calculator you may not be able to find factorial of 1000.

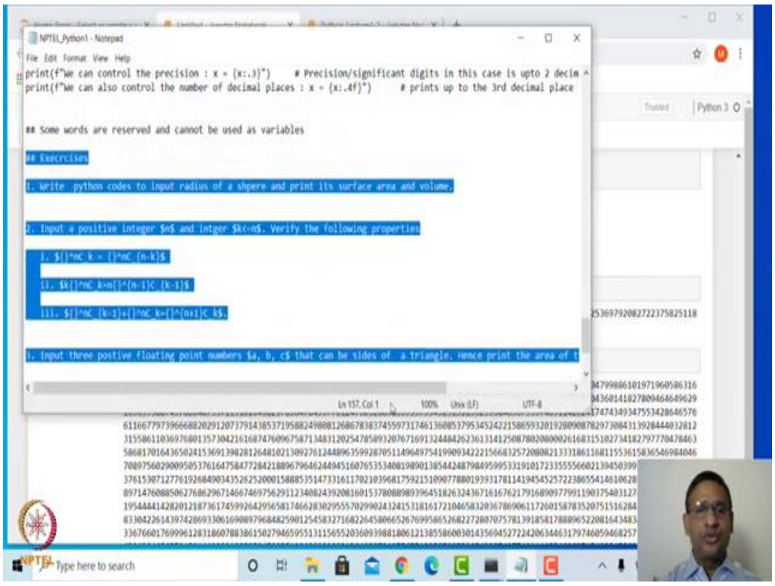
(Refer Slide Time: 24:10)


localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
File Edit View Insert Cell Kernel Navigate Widgets Help
Trusted
Python 3
Run
Code
Type here to search

```
6116677973966688202912073791438537195882498081268678383745597317461360853795345242215865932019280908782973084313928444032812
3155861103697680135730421616874760967587134831202547858932076716913244842623613141250878020800026168315102734182797770478463
5868170164365024153691398281264810213092761244896359928705114964975419909342221566832572080821333186116811553615836546984046
7089756029009505376164758477284218896796462449451607653534081989013854424879849599533191017233555660213945039973628075013378
3761530712776192684903435262520001588853514733161170210396817592151090778801939317811419454525722386554146106289218796022383
8971476088506276862967146674697562911234082439208160153780889893964518263243671616762179168909779911903754031274622289988005
1954444142820121873617459926429565817466283029555702990243241531816172104658320367869061172601587835207515162842255402651704
8330422614397428693306169089796848259012545832716822645806652676995865268227280707578139185817888965220816434834482599326604
3367660176999612831860788386150279465955131115655203609398818061213855860030143569452722420634463179746059468257310379008402 4
4324384656572450144028218852524709351906209290231364932734975655139587205596542287497740114133469627154228458623773875382304
8386568897646192738381490014076731044664025989949022222176590433990188601856652648506179970235619389701786004081188972991831
1021171229845901641921068884387121855646124960798722908519296819372388642614839657382291123125024186649353143970137428531926
6498753372189406942814341185201580141233448280150513996942901534830776445690990731524332782882698646027898643211390835062170
9500259738986355427719674282224875758676575234422020757363056949882508796892816275384886339690995982628095612145099487170124
4516461260379029309120889086942028510640182154399457156805941872748998094254742173582401063677404595741785160829230135358081
8400969963725242305608559037006242712434169090041536901059339838357779394109700277534720000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

In [ ]:
```

You can explore all the functions which are available inside math module and cmath module.

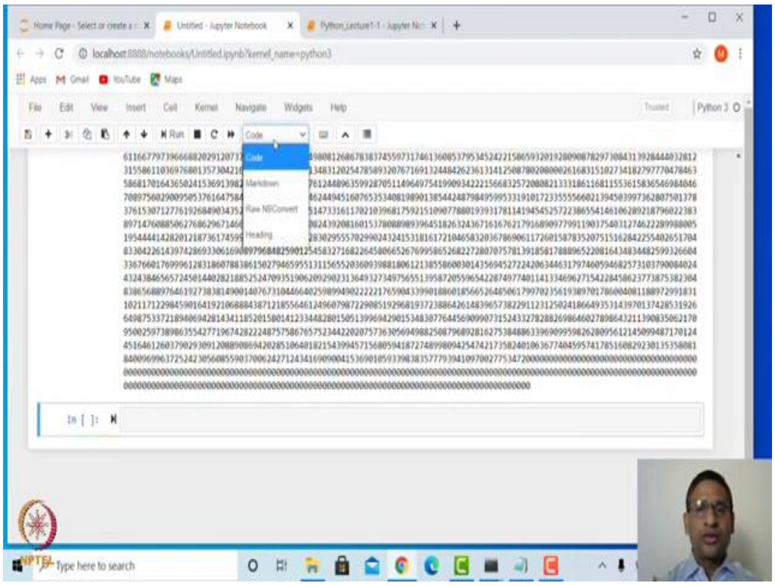
(Refer Slide Time: 24:25)



```
print(f"We can control the precision : x = {x:.3}")      # Precision/significant digits in this case is upto 2 decim
print(f"We can also control the number of decimal places : x = {x:.4f}")      # prints up to the 3rd decimal place


## Some words are reserved and cannot be used as variables

## Execrcises

1. Write  python codes to input radius of a shpere and print its surface area and volume.

2. Input a positive integer $n$ and intger $k<=n$. Verify the following properties

   i. ${}^nC_k = {}^nC_{n-k}$

   ii. $k{}^nC_k=n{}^{n-1}C_{k-1}$

   iii. ${}^nC_{k-1}+{}^nC_k={}^{n+1}C_k$.

3. Input three postive floating point numbers $a, b, c$ that can be sides of  a triangle. Hence print the area of t
```

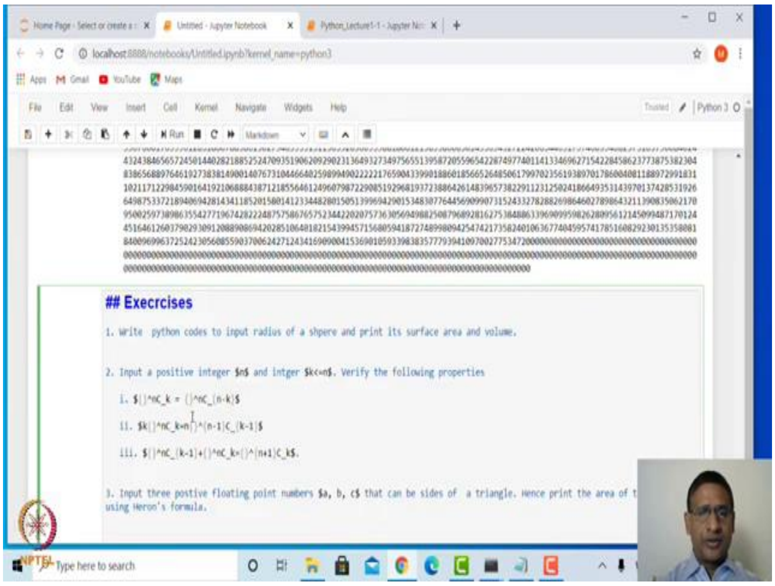
Before I end this lecture, let me let me leave you with some exercises. Let me enter these exercises.

(Refer Slide Time: 24:43)



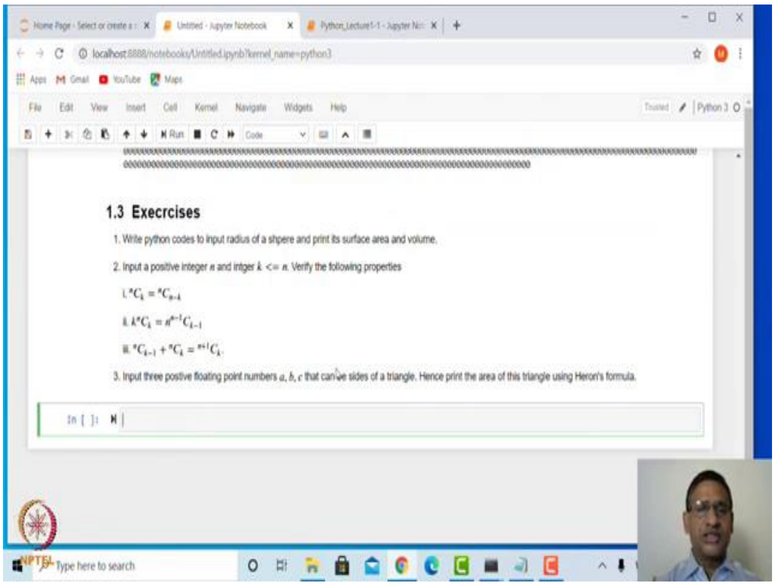
Let me convert this into Markdown.

(Refer Slide Time: 24:46)



```
## Execrcises

1. Write  python codes to input radius of a shpere and print its surface area and volume.

2. Input a positive integer $n$ and intger $k<=n$. Verify the following properties

   i. ${}^nC_k = {}^nC_{n-k}$

   ii. $k{}^nC_k=n{}^{n-1}C_{k-1}$

   iii. ${}^nC_{k-1}+{}^nC_k={}^{n+1}C_k$.

3. Input three postive floating point numbers $a, b, c$ that can be sides of  a triangle. Hence print the area of t
using Heron's formula.
```

(Refer Slide Time: 24:48)



1.3 Execrcises

1. Write python codes to input radius of a shpere and print its surface area and volume.
2. Input a positive integer n and intger $k <= n$. Verify the following properties

   i. ${}^nC_k = {}^nC_{n-k}$

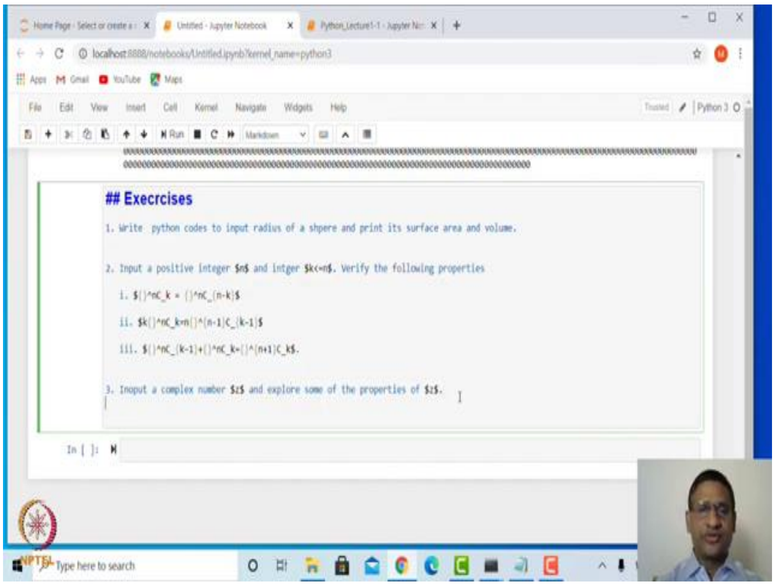
   ii. $k{}^nC_k = n^{n-1}C_{k-1}$

   iii. ${}^nC_{k-1} + {}^nC_k = {}^{n+1}C_k$.
3. Input three postive floating point numbers a, b, c that can be sides of a triangle. Hence print the area of this triangle using Heron's formula.

```
In [ ]:
```

These are the exercises. 1st - write python code to input a radius of a sphere and print it is surface area and volume. Input a positive integer n and integer k less than equal to n and verify the following properties, n C k is equal to n C n minus k and, the other things. The 3rd one is input three positive floating point numbers a, b, c that can be sides of a triangle and hence print the area of this triangle using herons formula. This we have already actually done.

(Refer Slide Time: 25:26)



```
## Execrcises

1. Write  python codes to input radius of a shpere and print its surface area and volume.

2. Input a positive integer $n$ and intger $k<=n$. Verify the following properties

   i. ${}^nC_k = {}^nC_{n-k}$

   ii. $k{}^nC_k=n{}^{n-1}C_{k-1}$

   iii. ${}^nC_{k-1}+{}^nC_k={}^{n+1}C_k$.

3. Inoput a complex number $z$ and explore some of the properties of $z$.
```

```
In [ ]:
```

So let me change this. I will say input a complex number z and verify some of the properties of z. So, input a complex number z, and verify some of the properties of z or instead of verify let me say explore. You have to make use of **cmath** module in this case. These are some simple exercises and you can explore some other computations as well.

Let me end it here. So, thank you very much. In the next lecture, we will look at more on python programming, especially we will look at how to make use of another built-in data structure like creating a list, creating a tuple and dictionary.

Thank you.