

Mathematics

Lecture 10

MAT 112
Math by Python

Prepared by
Dr. Israa Lewaa

Lecture 10

Lecture Objectives

Study how to download python

Learn how to run different codes with Python for factorization

Learn how to run different codes with Python for graphing functions

Learn how to run different codes with Python for solving limits

Learn how to run different codes with Python for differential

Learn how to run different codes with Python for Integration

- 1- Download Python (**Anaconda**) – Use **Jupyter** Notebook.
- 2- Python for factorization
- 3- Python for graphing functions
- 4- Python for solving limits
- 5- Python for differential
- 6- Python for Integration

1- Go to the technical engineer to set up the program called Anaconda.

2- Use Jupyter **ONLINE**.

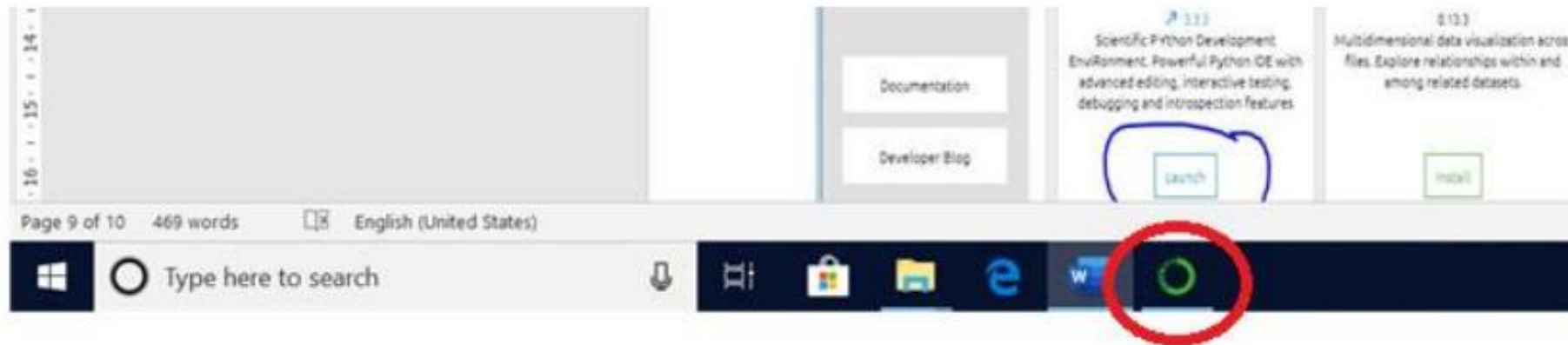
OR

☐ Solve all the problems manually.

Lecture 10

How to open the program ?

You will see that the Anaconda Navigator icon appears on the bottom toolbar. Click on the icon to see the contents of Navigator.

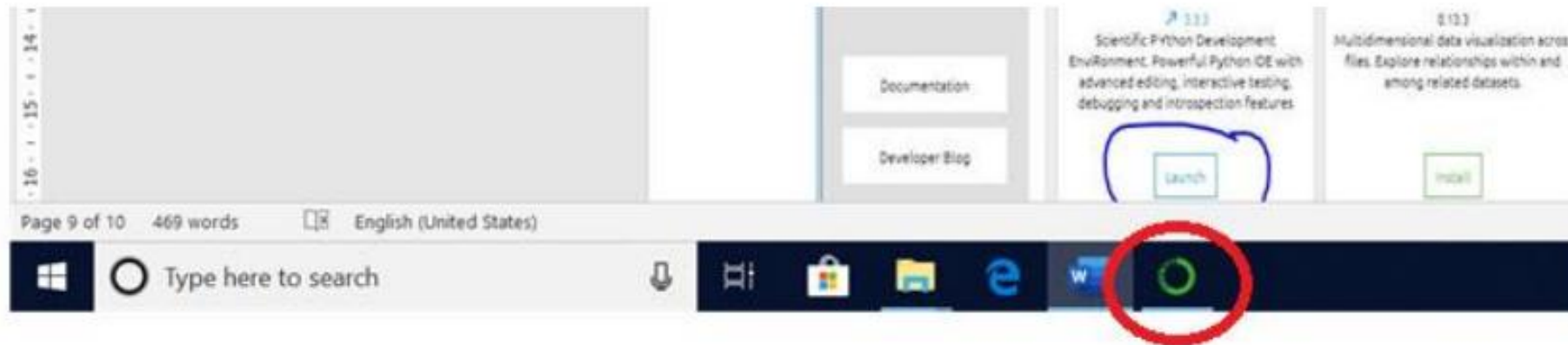


You will see that following page appears showing different options available which you can use.

Lecture 10

How to open the program ?

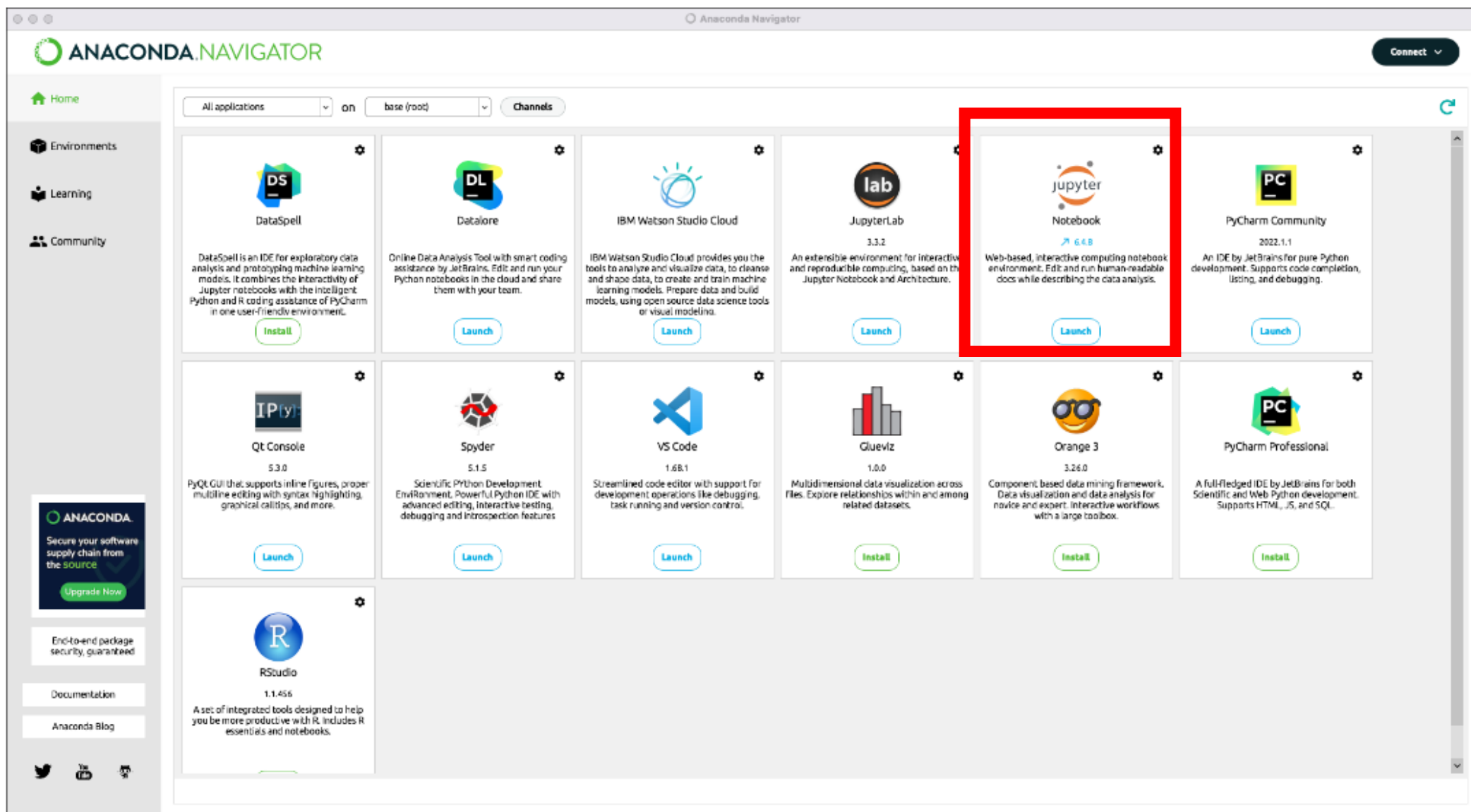
You will see that the Anaconda Navigator icon appears on the bottom toolbar. Click on the icon to see the contents of Navigator.



You will see that following page appears showing different options available which you can use.

Lecture 10

How to open the program ?



Lecture 10

Jupyter Interface



Quit

Logout

Files Running Clusters Nbextensions

Select items to perform actions on them.

Upload

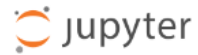
New ▾



☐ 0 ▾	📁 /	Name ▾	Last Modified	File size
☐	📁 ansel		3 months ago	
☐	📁 Contacts		4 months ago	
☐	📁 Creative Cloud Files		16 hours ago	
☐	📁 Desktop		17 hours ago	
☐	📁 Documents		17 hours ago	
☐	📁 Downloads		2 minutes ago	
☐	📁 Favorites		4 months ago	
☐	📁 Links		4 months ago	
☐	📁 Microsoft		4 months ago	
☐	📁 Music		4 months ago	

Lecture 10

Open new file in Jupyter



Quit

Logout

Files Running Clusters Nbextensions

Select items to perform actions on them.

Upload New ↕

☐ 0		Name ↓	Last Modified	File size
☐	📁 /			
☐	📁 anseI		3 months ago	
☐	📁 Contacts		4 months ago	
☐	📁 Creative Cloud Files		16 hours ago	
☐	📁 Desktop		17 hours ago	
☐	📁 Documents		17 hours ago	
☐	📁 Downloads		2 minutes ago	
☐	📁 Favorites		4 months ago	
☐	📁 Links		4 months ago	
☐	📁 Microsoft		4 months ago	

Lecture 10

Open new file in Jupyter

The screenshot displays the JupyterLab web interface. At the top, there's a header with the 'jupyter' logo and 'Quit' and 'Logout' buttons. Below the header, there are tabs for 'Files', 'Running', 'Clusters', and 'Nbextensions'. The 'Files' tab is active, showing a file browser. A red box highlights the 'New' button in the top right corner of the file browser. A dropdown menu is open, showing options: 'Python 3 (ipykernel)', 'Text File', 'Folder', and 'Terminal'. The 'Python 3 (ipykernel)' option is highlighted. A tooltip message says 'Create a new notebook with Python 3 (ipykernel)'. The file browser shows a list of files and folders, including 'ansel', 'Contacts', 'Creative Cloud Files', 'Desktop', 'Documents', 'Downloads', 'Favorites', 'Links', 'Microsoft', and 'Music'.

jupyter

Quit Logout

Files Running Clusters Nbextensions

Select items to perform actions on them.

0 /

Python 3 (ipykernel)

Text File

Folder

Terminal

17 hours ago

7 minutes ago

4 months ago

4 months ago

4 months ago

4 months ago

Create a new notebook with Python 3 (ipykernel)

Lecture 10

New Notebook

jupyter calculus by Python Last Checkpoint: 2 minutes ago (autosaved)



Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted

Python 3 (ipykernel)



In []:

Lecture 10

New Notebook

jupyter calculus by Python Last Checkpoint: 2 minutes ago (autosaved)



Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted

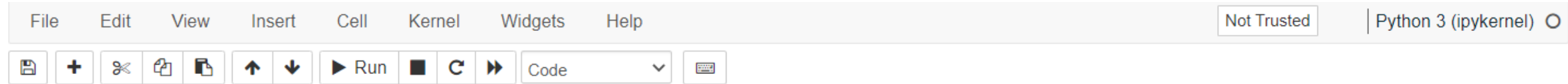
Python 3 (ipykernel)



In []:

Lecture 10

Sympy Package



```
In [1]: from sympy import init_session
init_session()
```

IPython console for SymPy 1.10.1 (Python 3.9.12-64-bit) (ground types: python)

These commands were executed:

```
>>> from sympy import *
>>> x, y, z, t = symbols('x y z t')
>>> k, m, n = symbols('k m n', integer=True)
>>> f, g, h = symbols('f g h', cls=Function)
>>> init_printing()
```

Documentation can be found at <https://docs.sympy.org/1.10.1/>

Lecture 10

Sympy Package

```
In [2]: import sympy  
dir(sympy)
```

```
Out[2]: ['Abs',  
        'AccumBounds',  
        'Add',  
        'Adjoint',  
        'AlgebraicField',  
        'AlgebraicNumber',  
        'And',  
        'AppliedPredicate',  
        'Array',  
        'AssumptionsContext',  
        'Atom',  
        'AtomicExpr',  
        'BasePolynomialError',  
        'Basic',  
        'BlockDiagMatrix',  
        'BlockMatrix',  
        'CC',  
        'CRootOf',  
        'Catalan',
```

Solving Equations and Working With Math Functions

```
In [3]: f = x**2 - 9
```

```
Out[3]:  $x^2 - 9$ 
```

```
In [ ]: f
```

```
In [4]: solve(f,x)
```

```
Out[4]: [-3, 3]
```

Factorization

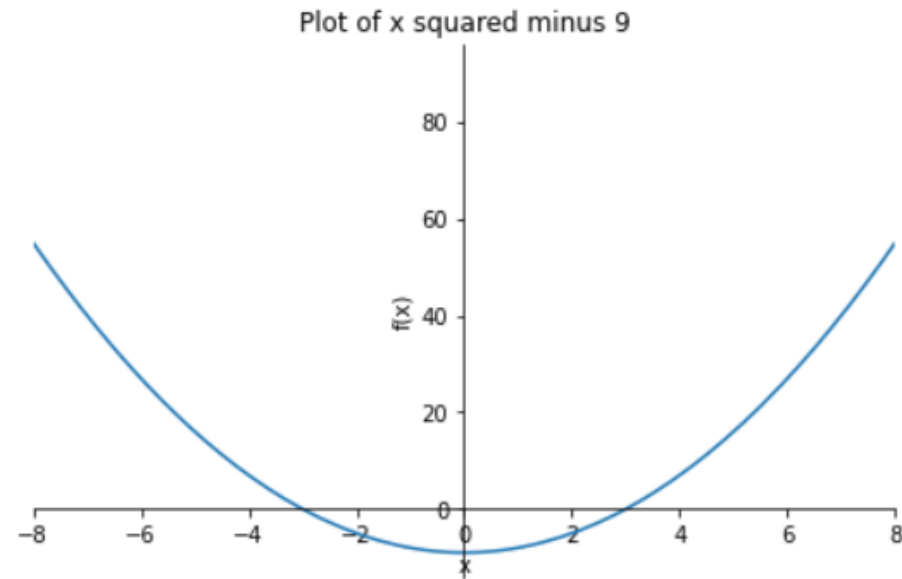
In [5]: `factor(f)`

Out[5]: $(x - 3)(x + 3)$

Graph function

```
In [6]: %matplotlib inline  
p = plot(f, xlim=(-8, 8), show=False,  
        title="Plot of x squared minus 9")
```

```
In [7]: p.show()
```



Lecture 10

Writing Root

```
In [8]: g = sqrt(x)/(1-x**2)
```

```
In [9]: g
```

```
Out[9]: 
$$\frac{\sqrt{x}}{1-x^2}$$

```

Limits

```
In [10]: k = (x**2 - 25)/(x**2 + 2 * x - 15)
```

```
In [11]: k
```

```
Out[11]: 
$$\frac{x^2 - 25}{x^2 + 2x - 15}$$

```

```
In [12]: limit(k,x,-5)
```

```
Out[12]: 
$$\frac{5}{4}$$

```

Differential

```
In [13]: M= 4*x**3 + sin(x) - 19
```

```
In [14]: diff(M)
```

```
Out[14]: 12x2 + cos(x)
```

Integration

```
In [7]: V = 5 * x**3
```

```
In [8]: integrate(V)
```

```
Out[8]:  $\frac{5x^4}{4}$ 
```

Integration

```
In [6]: N = 12 * x**2 + cos(x)
```

```
In [16]: integrate(N)
```

```
Out[16]:  $4x^3 + \sin(x)$ 
```

1- Download Python

2- Python for factorization

factor(f)

3- Python for graphing functions

plot(f)

4- Python for solving limits

limit(f,x,3)

5- Python for differential

diff(f)

6- Python for Integration

integrate(f)

1 Math is not about memorizing ... Math is about understanding

Every rule in differentiation or integration has a geometric and practical meaning. Understand **why** we differentiate and **why** we integrate. Plot the function before solving — you will understand **50% of the problem** before you even start.

2 Solve problems every day, even just one

Consistency matters more than quantity.
One problem a day is better than 30 problems the night before the exam.
Remember:
Math = Practice × Practice

3 Use Python to Understand, Not Just to Solve

Make Python your learning assistant.

Use simple libraries like:

numpy, matplotlib, sympy.

Visualize functions, derivatives, and integrals — seeing the graph will help you understand much faster than reading the formula.

Example:

```
import sympy as sp
x = sp.symbols('x')
f = x**3 - 5*x
sp.diff(f, x), sp.integrate(f, x)
```

4 Create a One-Page Summary After Each Lecture

Keep it short and useful.

Include:

Key formulas

Important examples

Common mistakes

Two practice questions

Your summary sheet will save you **half the study time** at the end of the semester.

5 Never Hesitate to Ask Questions

Most students get stuck in Chapter 2 or 3 simply because they don't ask.

There is **no such thing as a silly question** — the real mistake is continuing without understanding.
