# Python For Everyone


## Mathematics Department Karachi University

**Engineer Syed Umaid Ahmed**
***BE (EE), ME (Mechatronics) Continued***
***NED University of Engineering & Technology***

- **Python is the fourth most popular programming language in the world currently and this extraordinary growth is only set to increase as observed by Stack Overflow Trends**

- **Google, Facebook, Mozilla, Quora, etc. use Python Web Framework**

- **Another well-known factor for the rapid growth of Python is its use in Data Science**

- **Machine Learning, Robotics, Computer Vision & Artificial Intelligence**

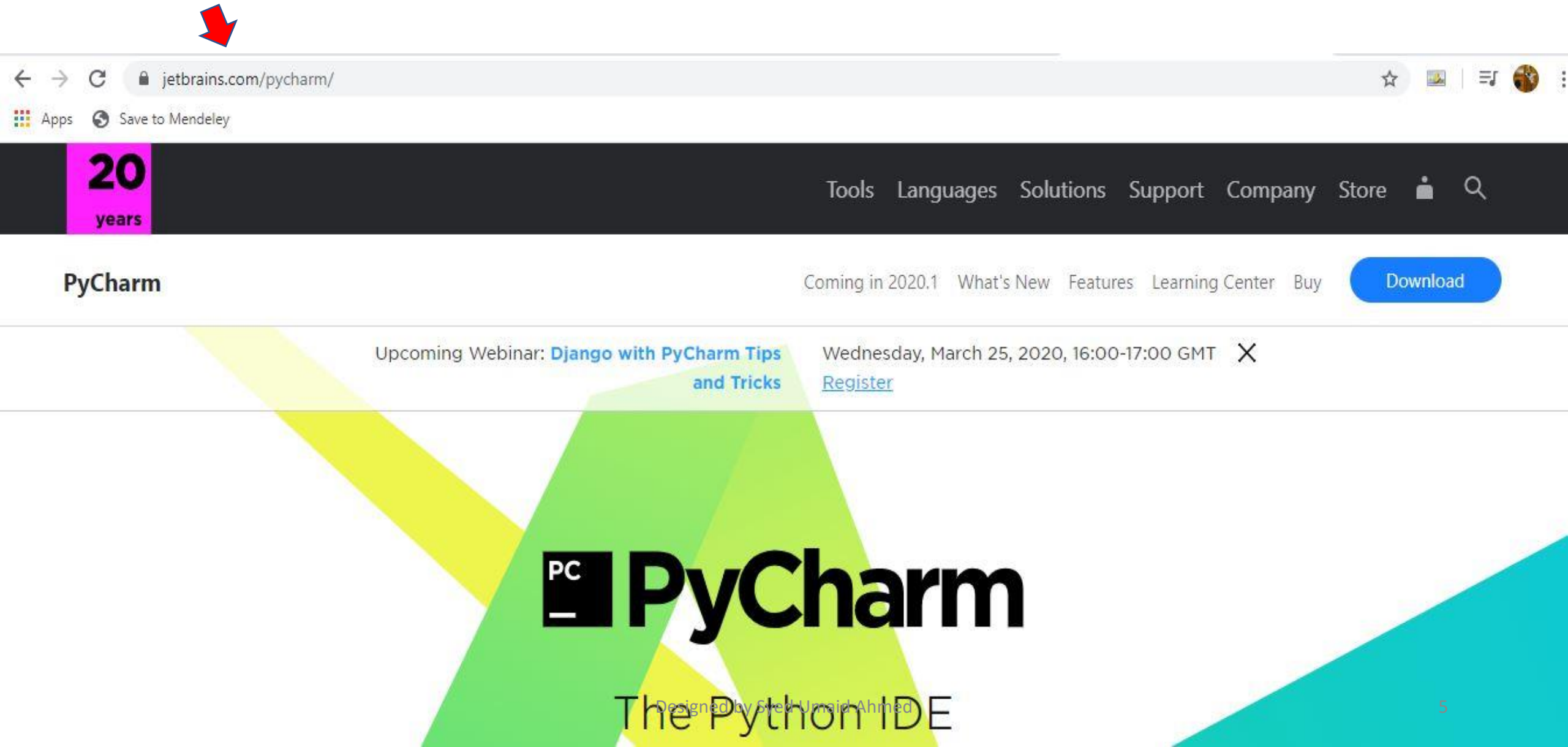
- **Python is Easy To Use**
- **Python has a Supportive Community**
- **Python has multiple Libraries and Frameworks**
- **Python has Corporate Support**
- **Python is used in Big Data and Machine Learning**
- **Python is used in Web Development**
- **Python is used in Robotics and Academics**
- **All the new microcontrollers and processors are python compatible**

# Install Python IDE in your Computer

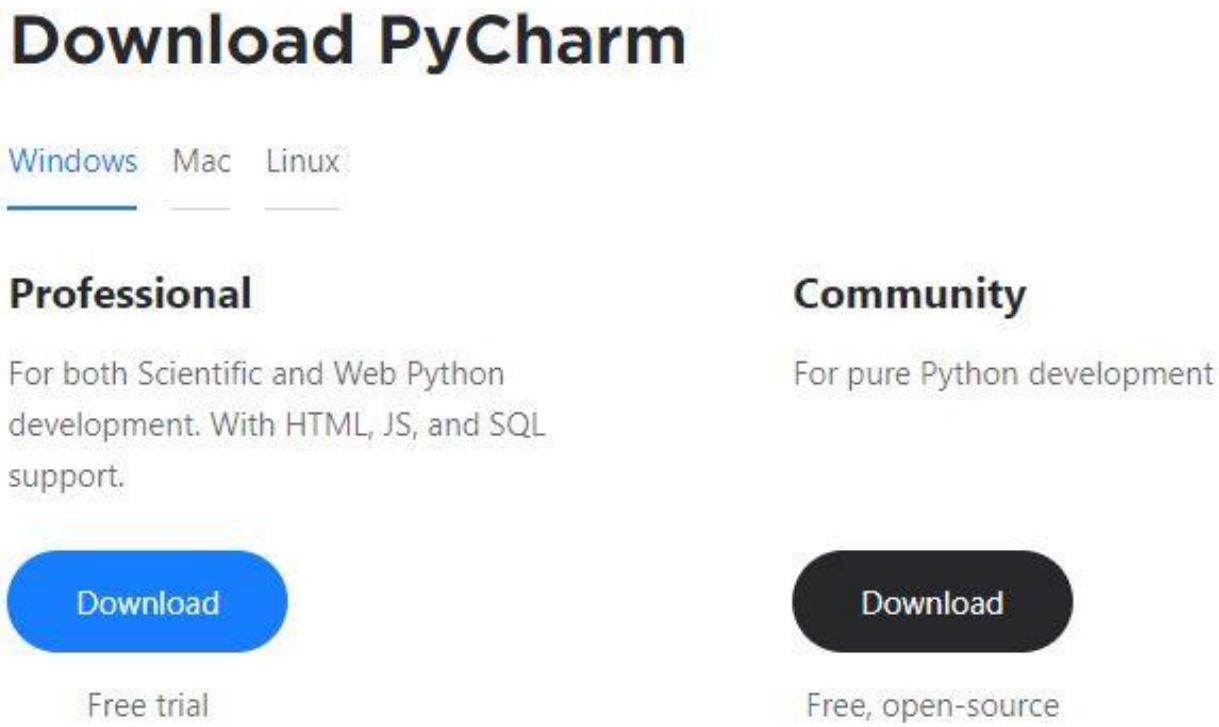
**Go to Link First**

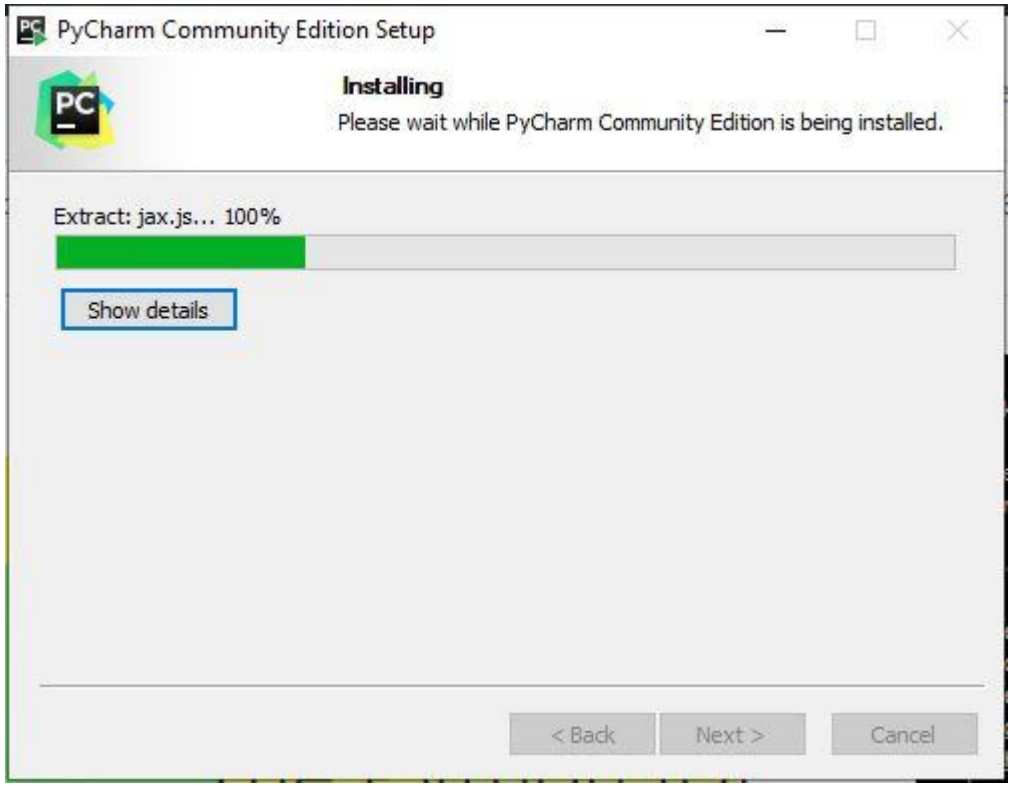

**Download Version 3.7.4 & please check mark "ADD VARIABLE TO PATH"**

**Go to Link Secondly:** ***jetbrains.com/pycharm/***

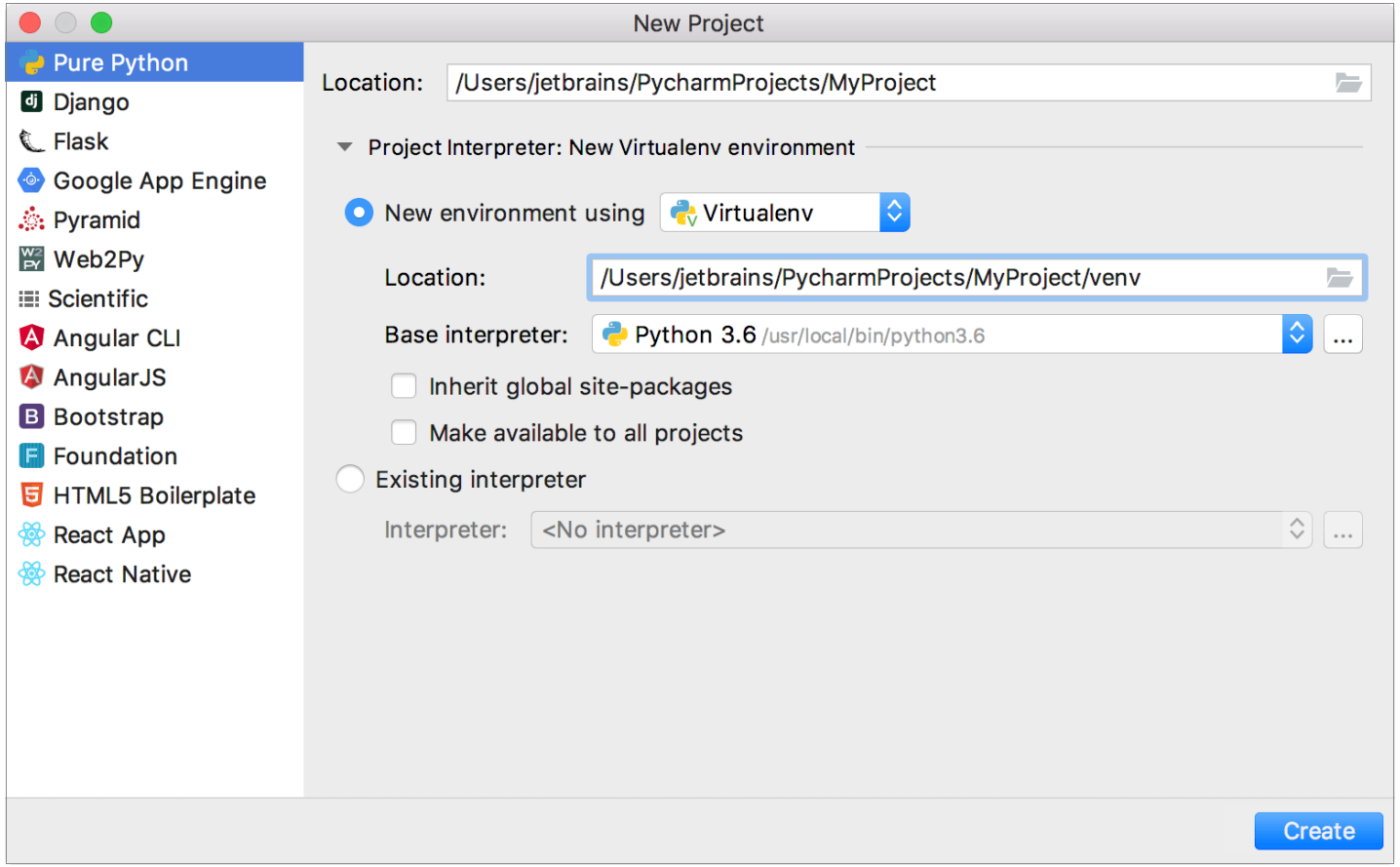


**Go to Link Thirdly:** **Install Community Edition of "*pycharm*"**



# Check this link to start working and basic Understanding of PyCharm Environment

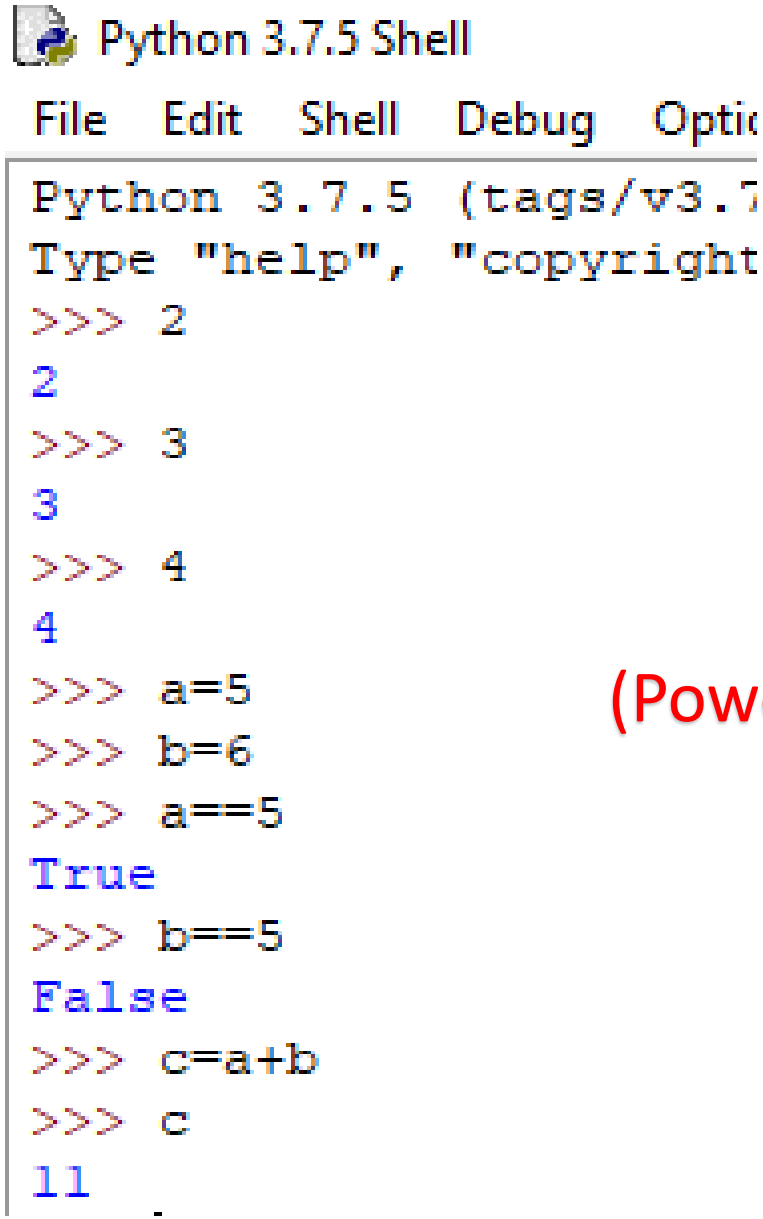



***Detailed Explanation: (Must go to this link to start)***

https://www.jetbrains.com/help/pycharm/creating-and-running-your-first-python-project.html

# Variables and Operators

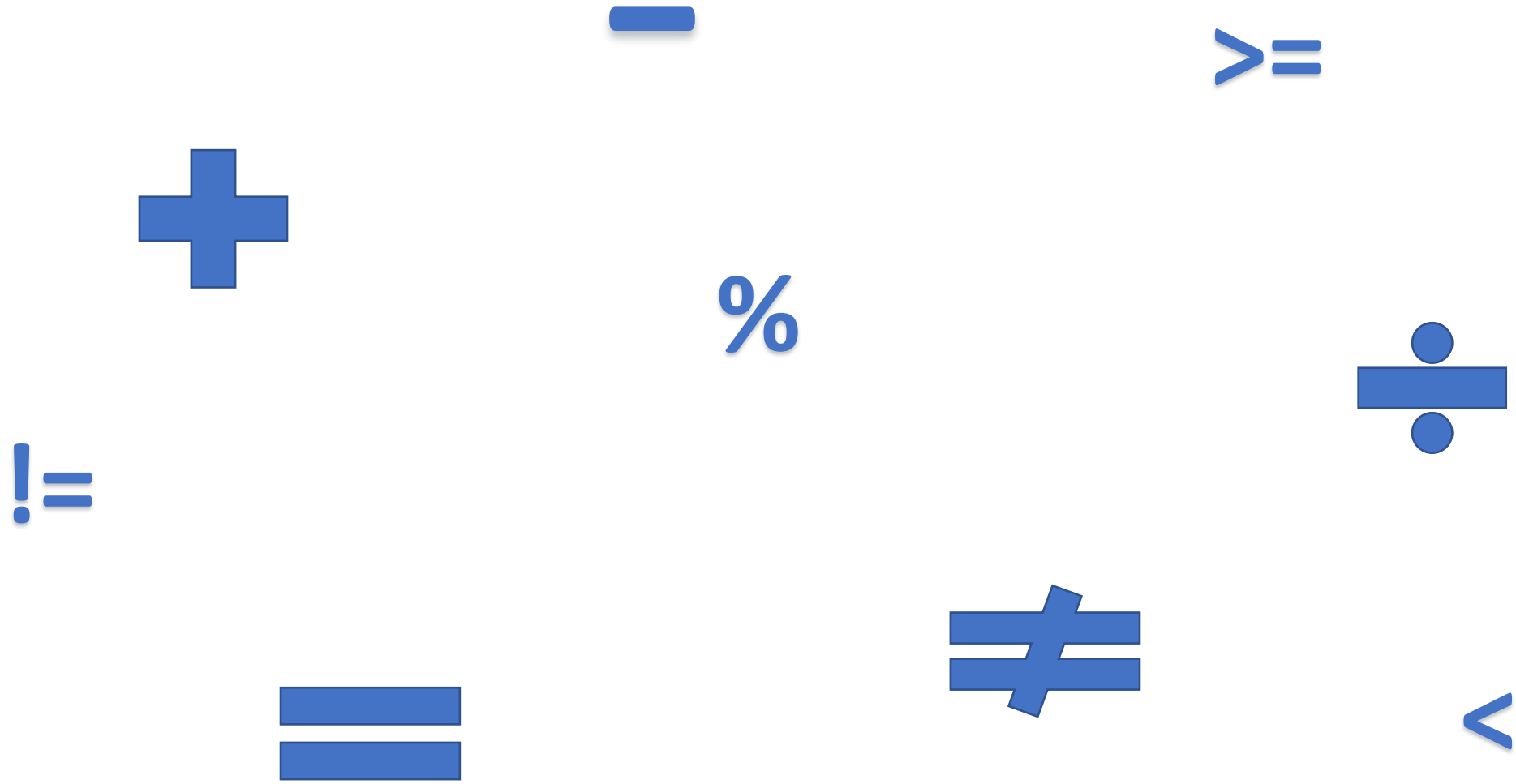
- Variables
- Checking
- Assigning
- True & False
- Addition
- Division
- Subtraction
- Check '=='
- Assign '='




2**2 = 4

(Power with Double Star)

# Perform Basic Arithmetic Operations (Try at Home)



# Arithmetic Operators (+, - , / , % …)

| Operator | Name | Example |
|---|---|---|
| + | Addition | x + y |
| - | Subtraction | x - y |
| * | Multiplication | x * y |
| / | Division | x / y |
| % | Modulus | x % y |
| ** | Exponentiation | x ** y |
| // | Floor division | x // y |

# Conditional Statements (if, else, elif)

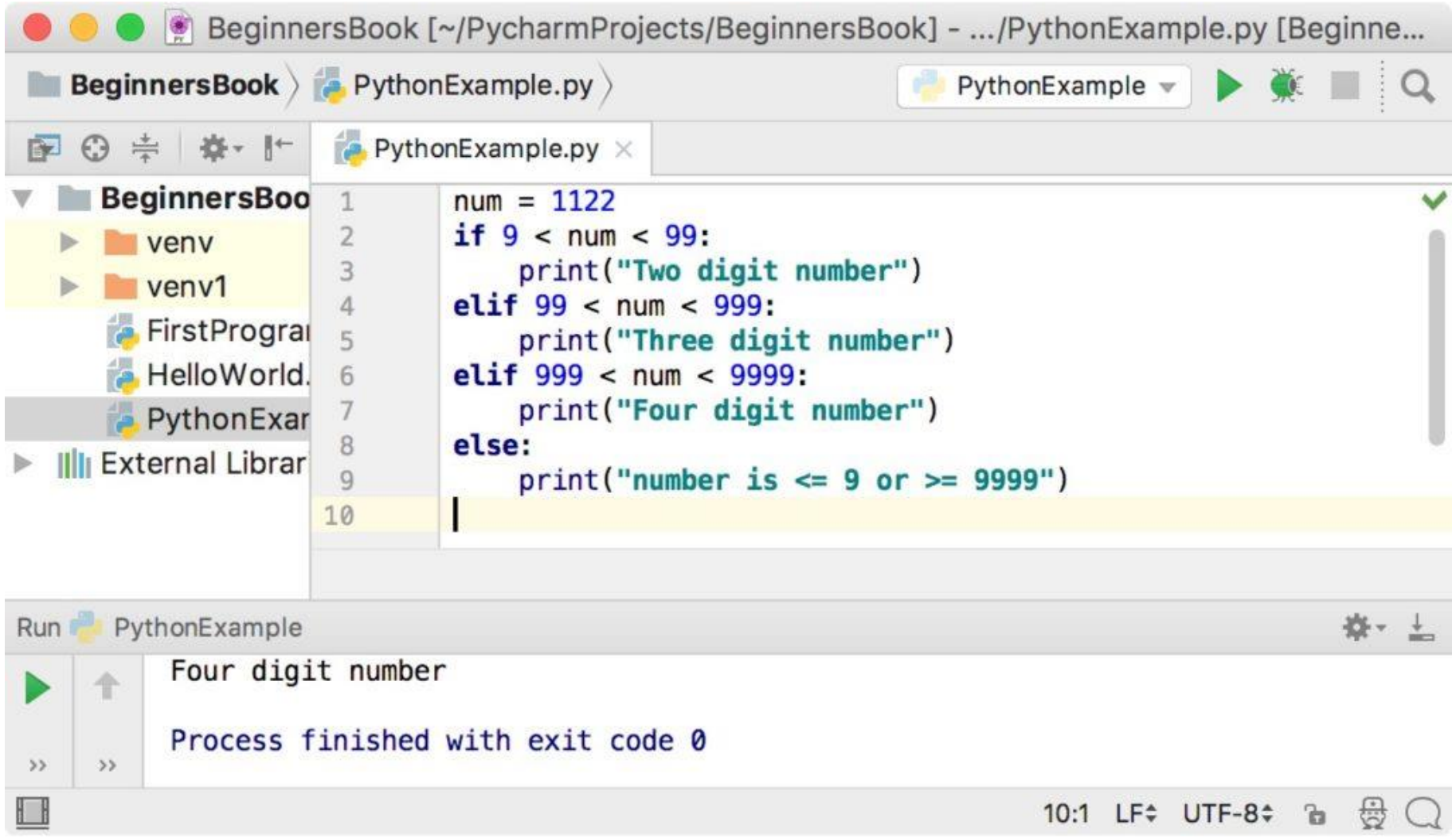
- **while loop**
- **If and else conditions**
- **Using format command**
- **Purpose of {0} inside print**

Chapter_2.py

```
while True:
    num = int(input("Enter a number: "))
    if (num % 2) == 0:
        print("{0} is Even number".format(num))
    else:
        print("{0} is Odd number".format(num))
```

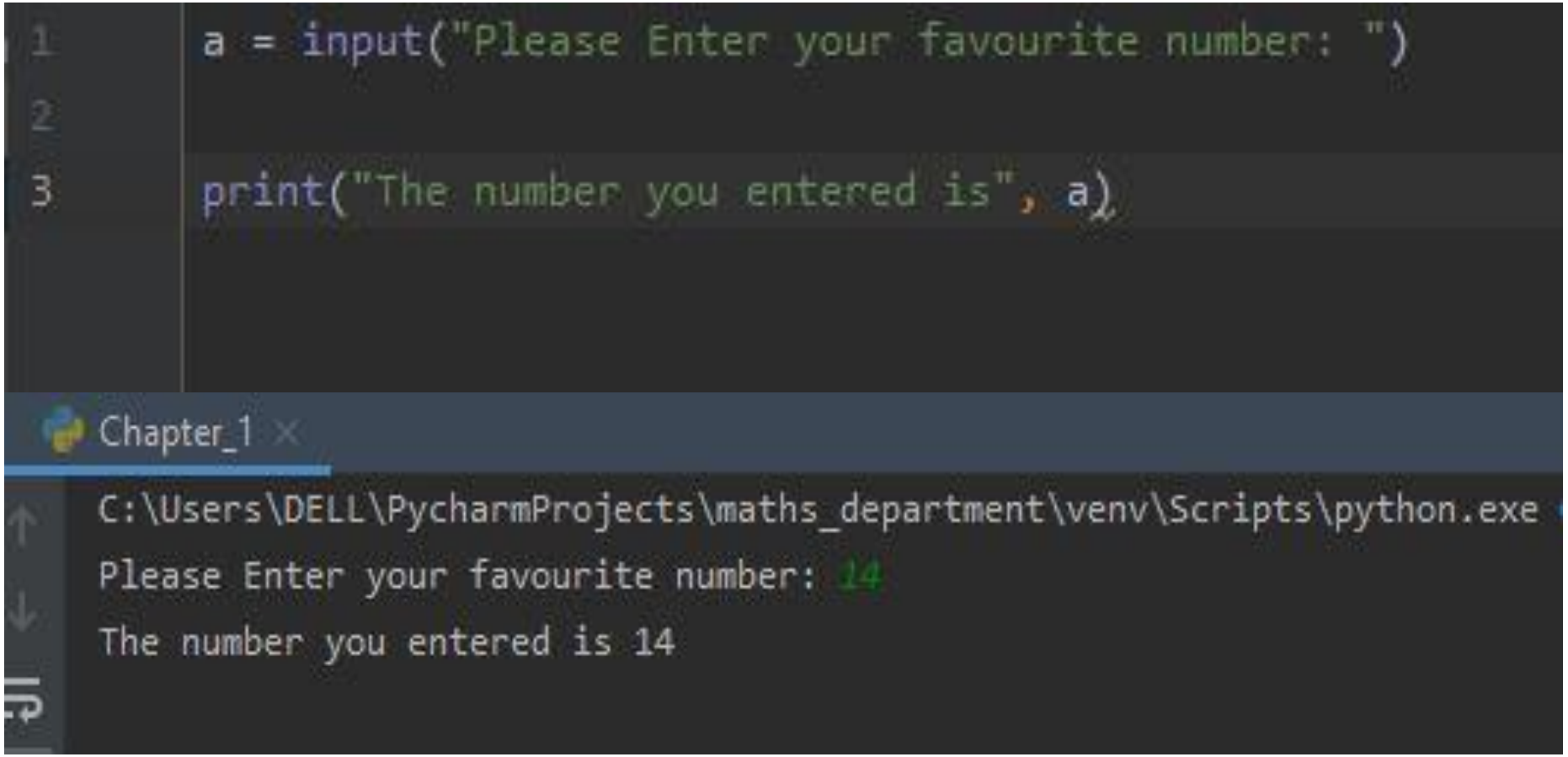
```
C:\Users\DELL\PycharmProjects\maths_department\venv\Scripts\python.exe
Enter a number: 0
0 is Even number
Enter a number: 21
21 is Odd number
Enter a number: 22
22 is Even number
```

# Conditional Statements: if, else and elif



# Taking the Simplest Input

*Share the example of a real program*




*The variable written without “ ” always gives the value.=, i.e. a*

# Some examples operators in Python

```
x = 5

print(x > 3 and x < 10)

# returns True because 5 is greater than 3 AND 5 is less than 10
```

```
x = 5

print(x > 3 or x < 4)

#returns True as an answer as or is for checking (IF ONE of BOTH is TRUE)
```

```
x = 5

print(not(x > 3 and x < 10))

# returns False because not is used to reverse the result
```

*For exploring every possible operator, Check this:*
https://www.w3schools.com/python/python_operators.asp

# *The real life problem and its solution through logical operators*

**This is an Fuel Prediction engine developed on Raspberry Pi Microcomputers. It is continuously measuring the level of liquid. The problem is with the "Alarm System"**
**We want to glow the LED and make the buzzer ring when the fuel goes on Two levels. Tank full limit is 310 milliliters**

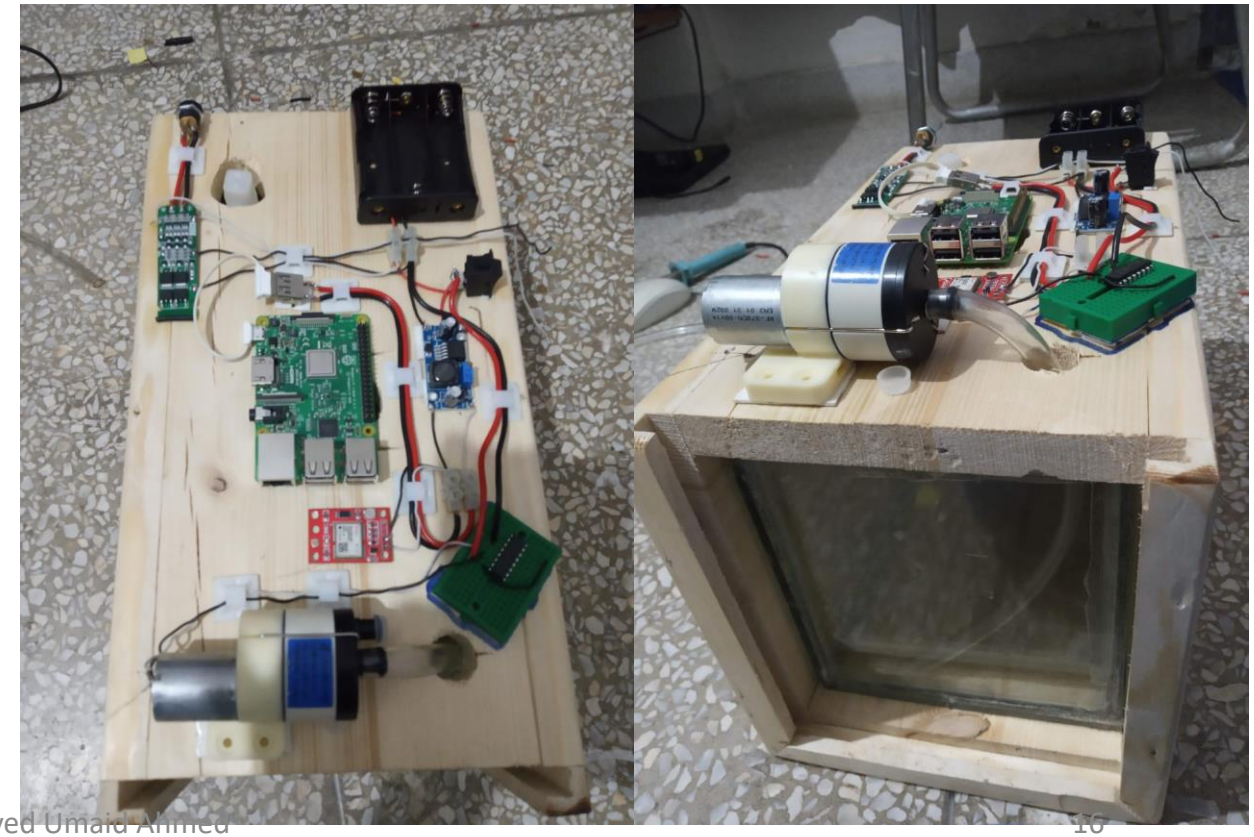
**Level 1: Overfilling (Please Stop)**
**Level 2: Less fuel (Engine is getting empty)**

## Buzzer/LED Indication ?

**If fuel >300 ml or fuel <10 ml :**

**ALARM !!!**

**"OR" operator check for both**
**Ring alarm if anyone is true**



# Our Results after using the ‘OR’ operator (Play Video)

Successful !

# Loops in Python Programming

***For Loop go to (n-1) = 10 – 1 = 9***

***When using "str" , python treats it as string, when you multiply string, it prints it "NO OF TIMES"***

Chapter_3.py

```
1   for i in range(10):
2       print(str(i) * i)
3
```

```
C:\Users\DELL\PycharmProjects\maths_department\venv\Scripts\python.exe

1
22
333
4444
55555
666666
7777777
88888888
999999999
```

Chapter_3.py

```
1   n = int(input("Input a number: "))
2
3   # use for loop to iterate 10 times
4   for i in range(1, 11):
5       print(n, 'x', i, '=', n * i)
```

```
C:\Users\DELL\PycharmProjects\maths_department\venv\Scripts\python.exe
Input a number: 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

# Loops in the List (Some Concepts)

List = [ 0, 1, 2, 3, 4, 5]

| 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|

List[0] = 0

List[1] = 1

List[2] = 2

List[3] = 3

List[4] = 4

List[5] = 5

List[0:] = [0,1,2,3,4,5]

List[:] = [0,1,2,3,4,5]

List[2:4] = [2, 3]

List[1:3] = [1, 2]

List[:4] = [0, 1, 2, 3]

# Loops in Lists (Arrays)

```
In [2]: numbers = [1, 5, 12, 91, 102]

In [3]: for i in numbers:
            print(i * i)

1
25
144
8281
10404
```

**Numbers Total in list= 5**

**On location i[0] is one**
**1*1 = 1**

**On location i[1] is five**
**5*5 =25**

**On location i[2] is twelve**
**12*12 = 144**

**Etc…**
**91*91= 8281**

```
1  # Loops in lists
2  languages = ["C", "C++", "Perl", "Python"]
3  for x in languages:
4      print(x)
```

```
C:\Users\DELL\PycharmProjects\maths_department\
C
C++
Perl
Python
```

# Nested Loops (Square Pattern)

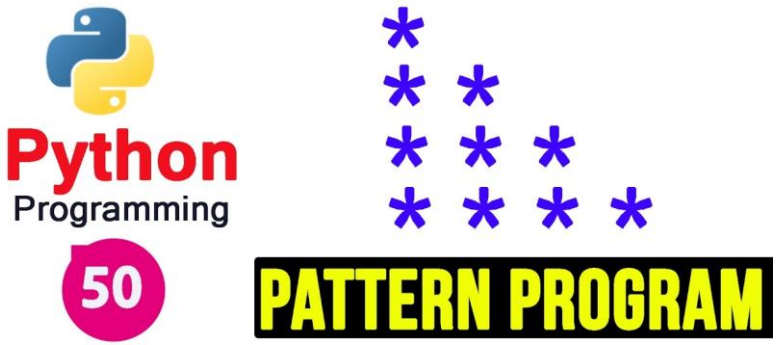
squarenum.py - /Users/suresh/Documents/Python Programs/squarenum.py (3.7.0)

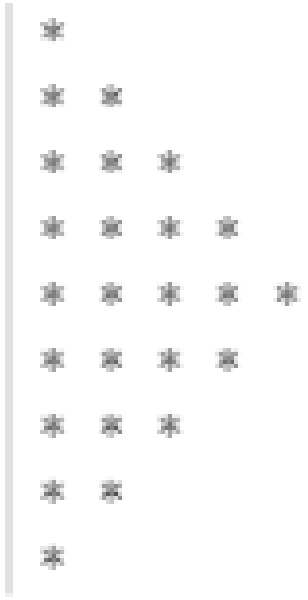
```
# Python Program to Print Square Number Pattern

side = int(input("Please Enter any Side of a Square  : "))

print("Square Number Pattern")

for i in range(side):                                    ©tutorialgateway.org
    for i in range(side):
        print('1', end = '  ')
    print()
```

Python 3.7.0 Shell

```
======= RESTART: /Users/suresh/Documents/Python Programs/squarenum.py =======
Please Enter any Side of a Square  : 6
Square Number Pattern
1  1  1  1  1  1
1  1  1  1  1  1
1  1  1  1  1  1
1  1  1  1  1  1
1  1  1  1  1  1
1  1  1  1  1  1
>>>
```

# Python Pattern Printing (Understanding Loops)



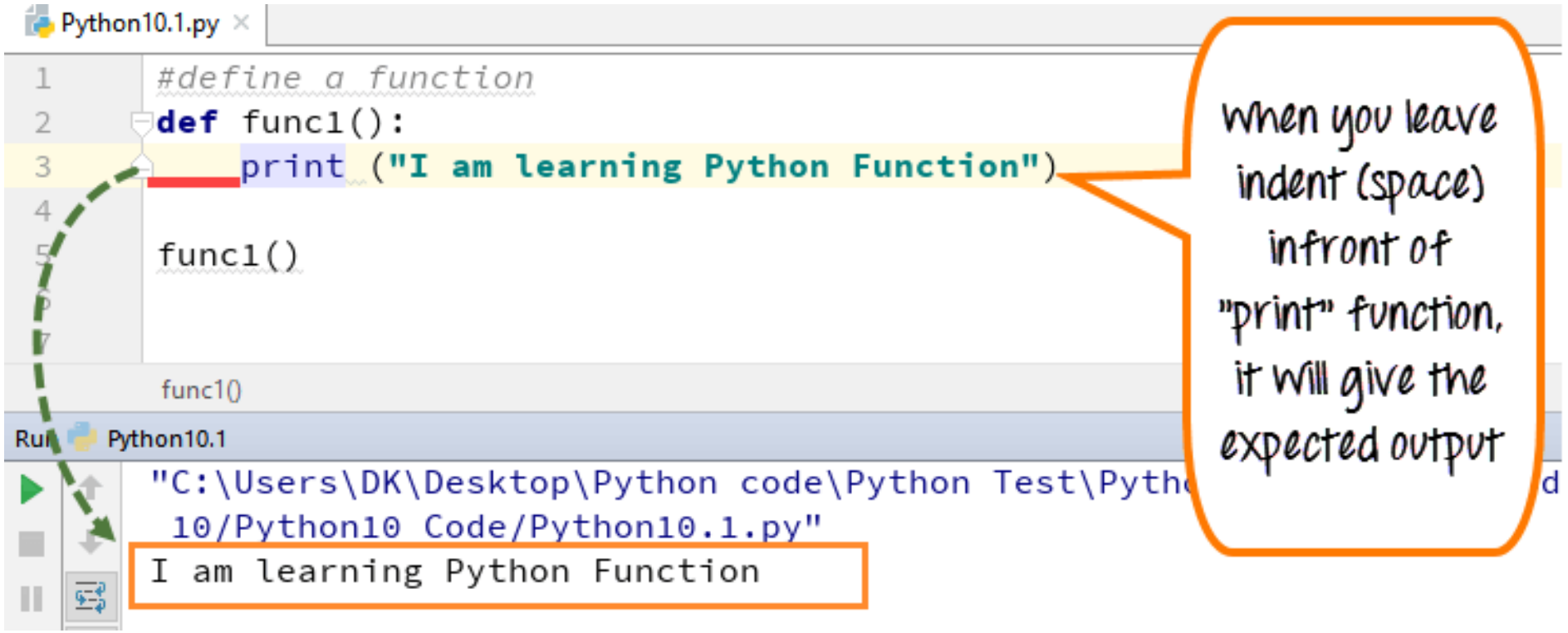
6. Write a Python program to construct the following pattern, using a nested loop number (two levels of loop statements).

*Expected Output:*

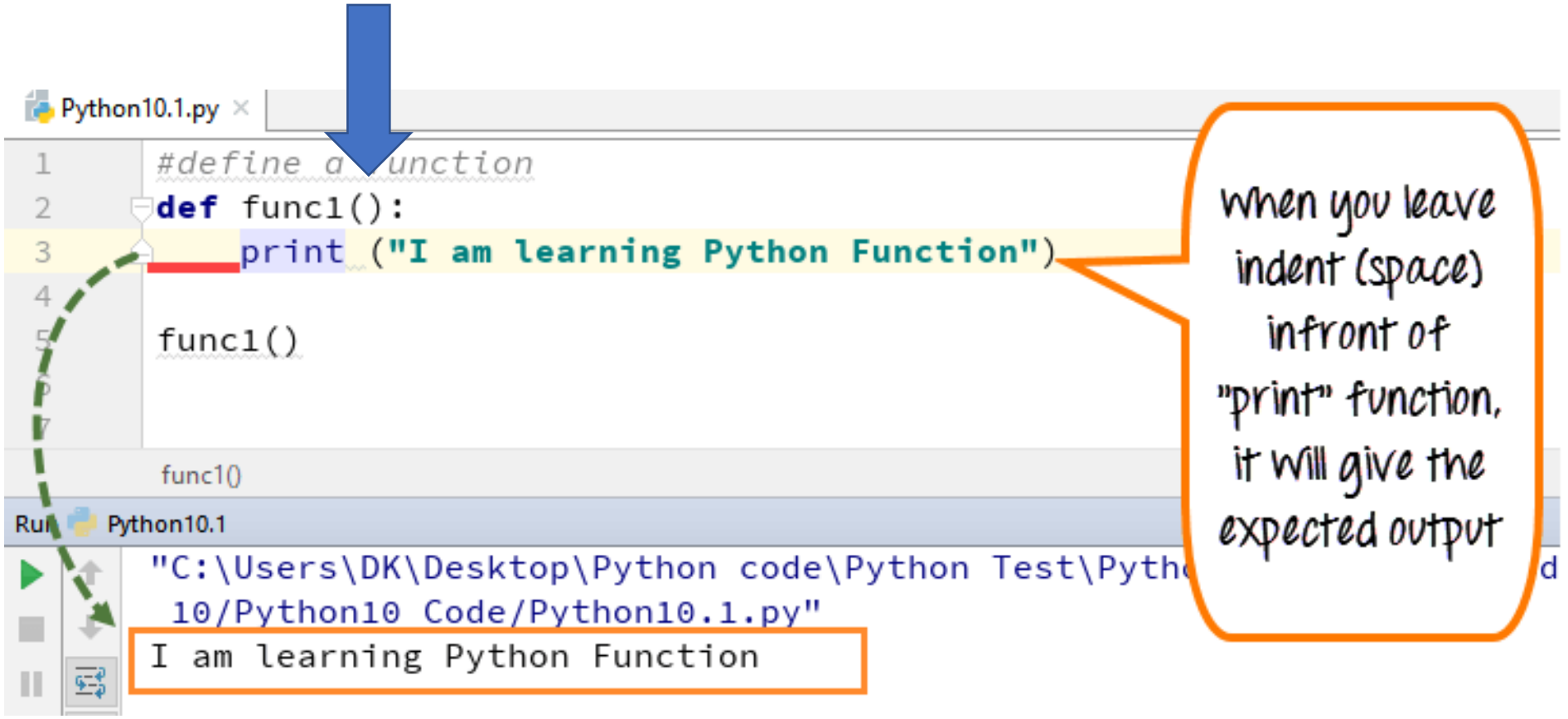
```
1
22
333
4444
55555
666666
7777777
88888888
999999999
```

Hint: to print a string without new line use: print('temp2', end='')

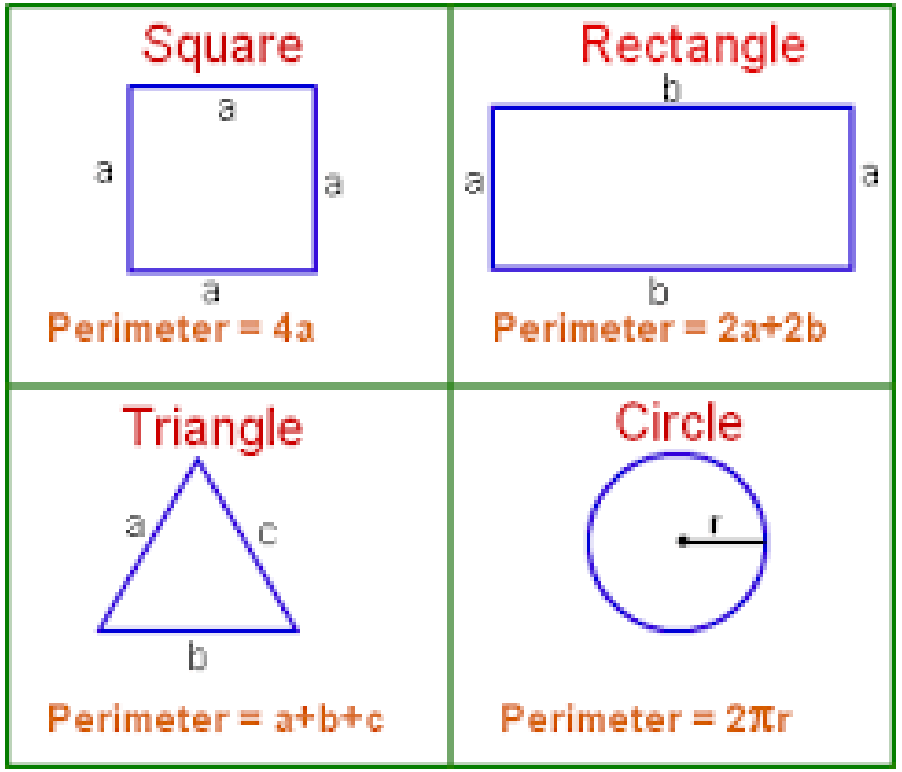
# Getting Started with Functions in Python



# Function with Arguments



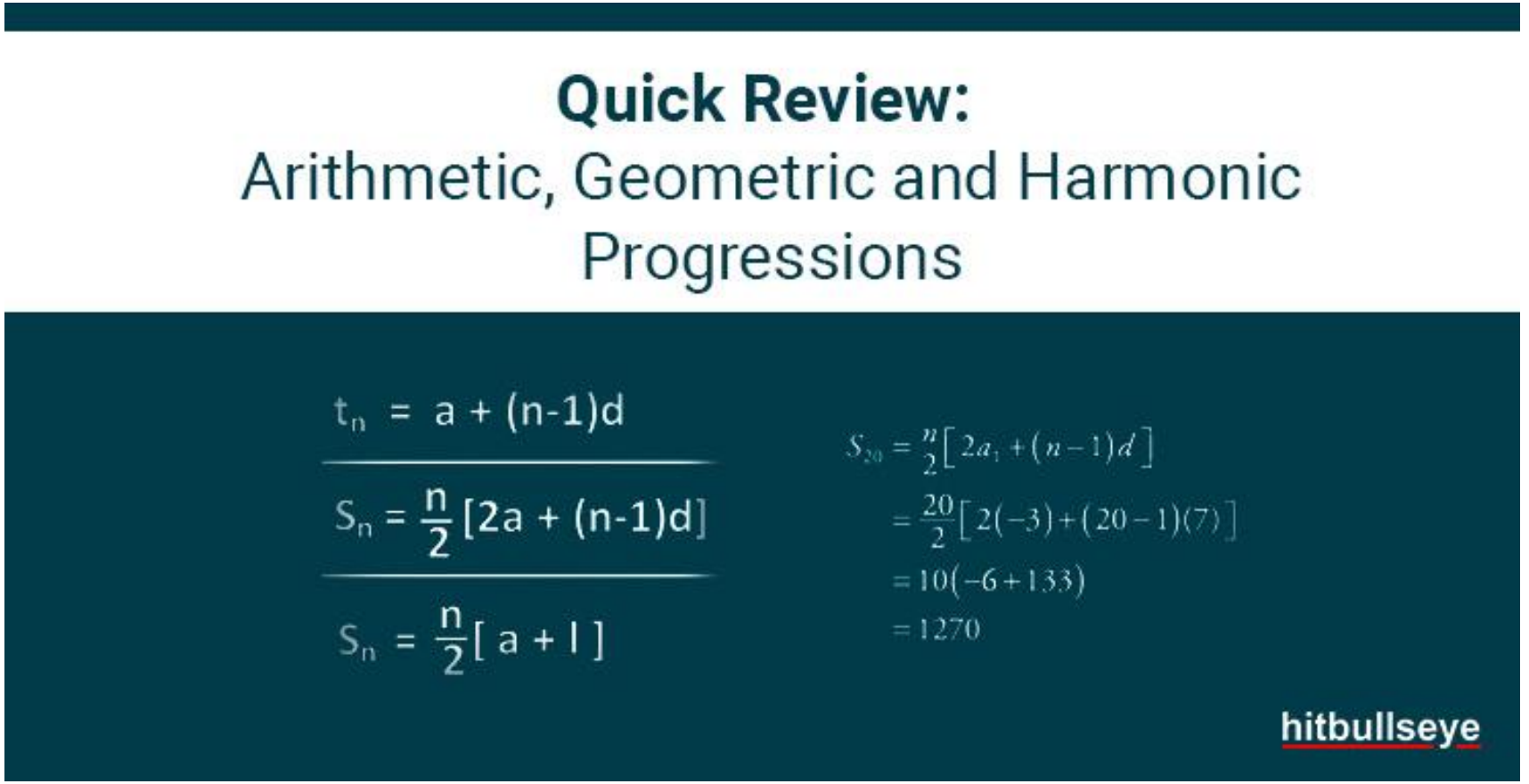
# Calculate the Area and Perimeter Through a Function



```
def AreaPerimeter (height, width):
    height = int(height)
    width = int(width)
    area = height * width
    perimeter = (2 * height) + (2 * width)
    print ("The area is: " , area)
    print ("The perimeter is: " , perimeter)
    return



while True:
    h = input ("Enter height: ")
    w = input ("Enter width: ")
    AreaPerimeter (h, w)
```

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\DELL\Desktop\Mathematics Department\Slide_4_Function_Perimeter.py
Enter height: 12
Enter width: 5
The area is:  60
The perimeter is:  34
Enter height:
```

# Finding the Nth Term of an Arithmetic Progression



```
def AO(x,y,z):
    first_term=int(x)
    difference=int(y)
    Number=int(z)



    Answer = first_term+(Number-1)*difference
    print("The term is: ", Answer )
    return

while True:
    a = input("Enter first term: ")
    b = input("Enter difference: ")
    c = input("Enter nth number of Term: ")
    AO(a,b,c)
```

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51)
4)] on win32
Type "copyright", "credits" or "license()" for more inf
>>>
 RESTART: C:\Users\DELL\Desktop\Mathematics Department\
py
Enter first term: 2
Enter difference: 2
Enter nth number of Term: 7
The term is:  14
Enter first term:
```

# Design Task for Function Values

$$f(x) = \sqrt{x+3} - x + 1$$

```
import math
def f(x):
    return math.sqrt(x + 3) - x + 1

for x in [0,1,math.sqrt(2),math.sqrt(2)-1]:
    print("f({:.3f}) = {:.3f}".format(x,f(x)))
```

```
>>>
 RESTART: C:\Users\DELL\Desktop\Mathematics Department'
for task_and_values.py
f(0.000) = 2.732
f(1.000) = 2.000
f(1.414) = 1.687
f(0.414) = 2.434
>>>
```

# Project of the Distance Calculator (Function with Four Arguments)

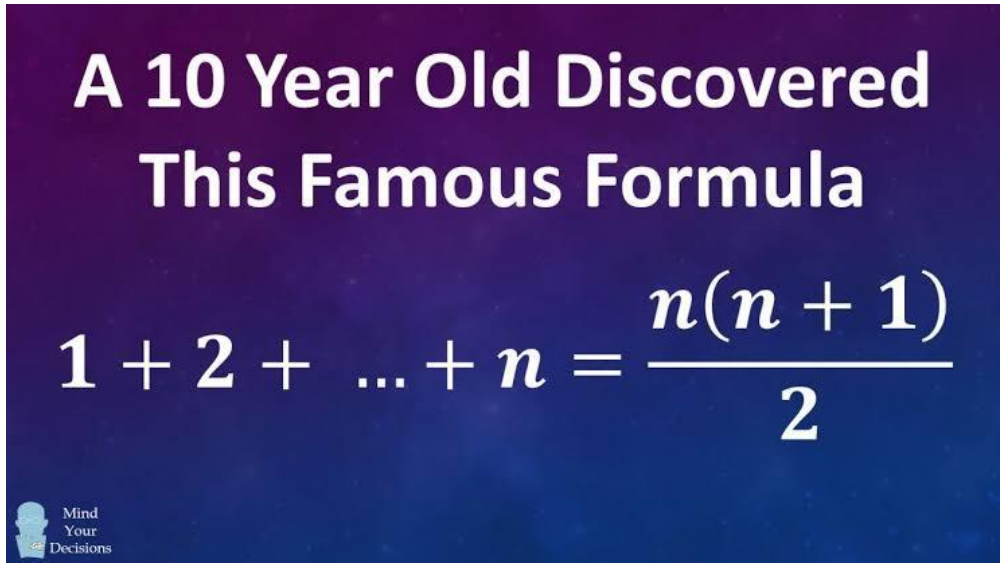

$$a = \sin^2(\Delta lat/2) + \cos(lat1) \times \cos(lat2) \times \sin^2(\Delta long/2)$$

$$c = 2 * \sin^{-1}(\sqrt{a})$$

$$d = R \times c$$

d = distance in kilometers

```
from math import radians, cos, sin, asin, sqrt
from time import sleep
import geocoder

g = geocoder.ip('me')

a=(g.lng)
b=(g.lat)
c=(g.lng)
d=(g.lat)

print(a)
print(b)

sleep(2)

print(c)
print(d)



def haversine(lon1, lat1, lon2, lat2):
    lon1 = radians(lon1)
    lat1 = radians(lat1)
    lon2 = radians(lon2)
    lat2 = radians(lat2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1


    a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2
    c = distance = 2 * asin(sqrt(a))
    r = 6371
    z= c * r
    print(z)


def image_file():
    haversine(a,b,c,d)
```




Total sum of simple series.py - C:\Users\Umaid\Desktop\Mathematics Department\Total sum of simple series.py (3.7.5)

File Edit Format Run Options Window Help

```
#1,2,3,4,5,6,7,8,9,.....


N=10
expected_sum = N * (N + 1) / 2
print(expected_sum)
```

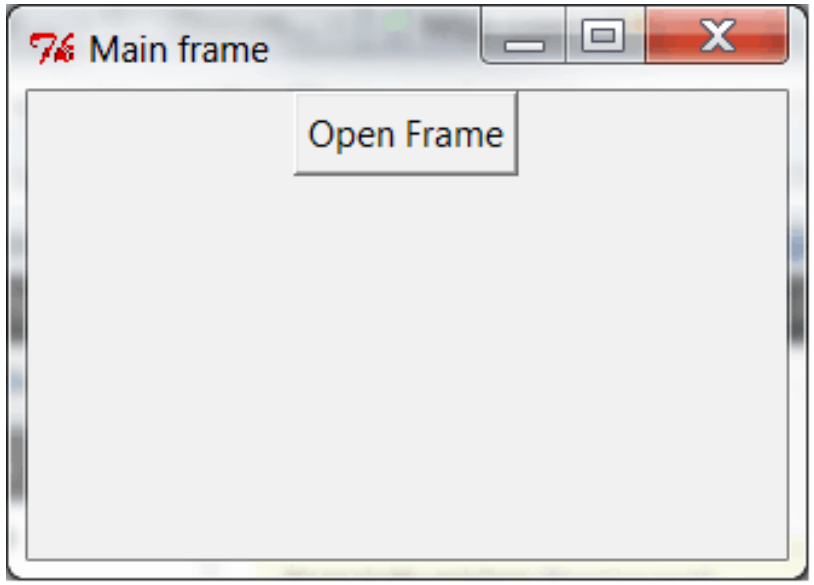
# Getting Started with GUI Development

**Tkinter in Python GUI Programming is standard Python GUI library.**

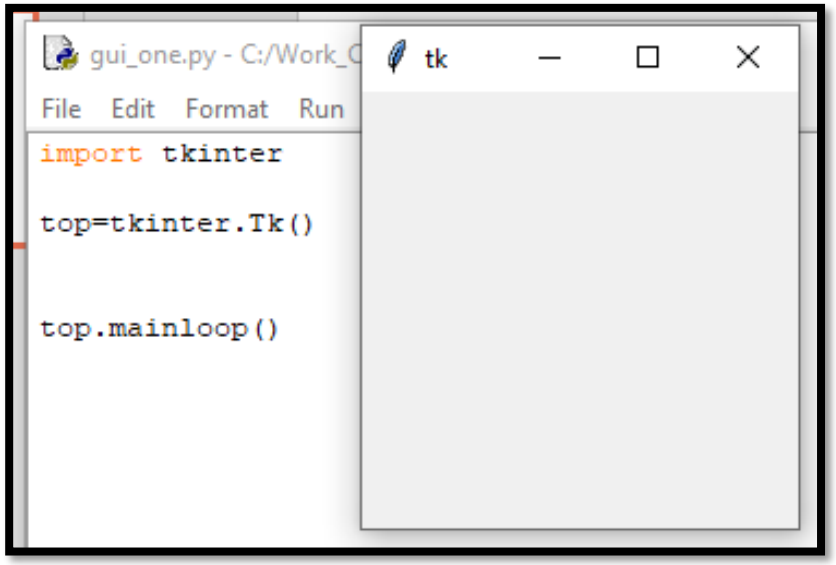
**It gives us an object-oriented interface to the Tk GUI toolkit.**

**So, let's start Python graphics, let's create a simple GUI application:**

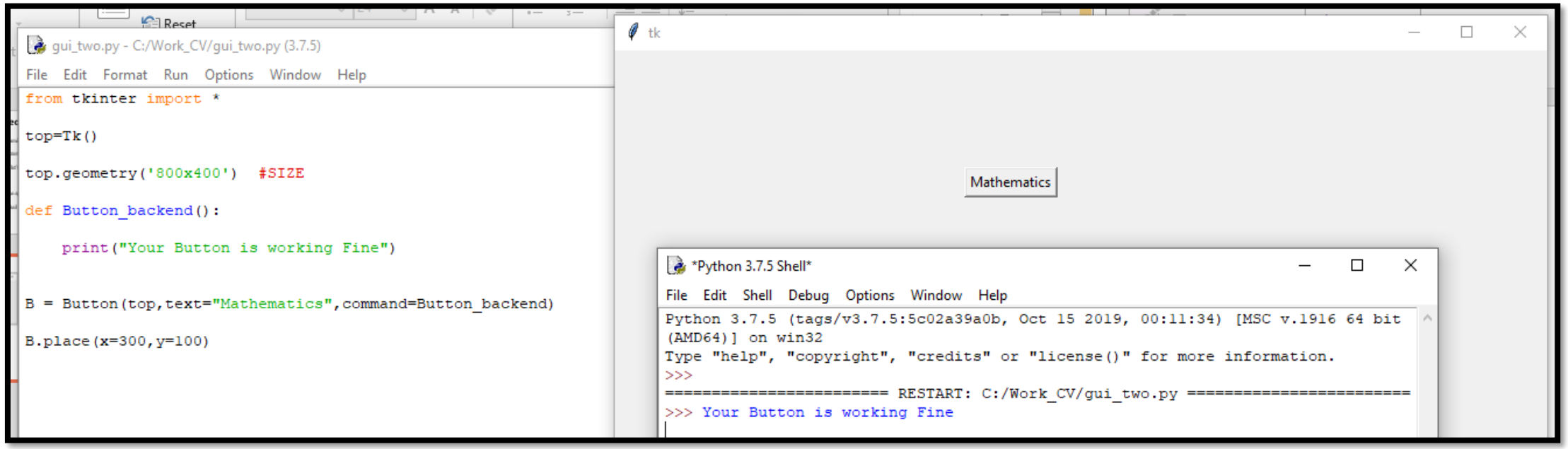
- **Import the module Tkinter.**
- **Create the main window for the application.**
- **Add a widget or more.**
- **Enter the main event loop to deal with an event when it is triggered.**



# Example One: (Generating Window)




# Example Two (Adding Button):

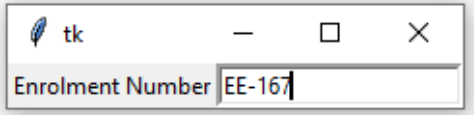


# Example Three (Adding Entries):

gui_three.py - C:/Work_CV/gui_three.py (3.7.5)

File Edit Format Run Options Window Help

```
from tkinter import *

top=Tk()

#LABELING WIDGET

L1=Label(top,text="Enrolment Number")

L1.pack(side=LEFT)


#ENTRY WIDGET

E1=Entry(top,bd=3)

E1.pack(side=RIGHT)
```

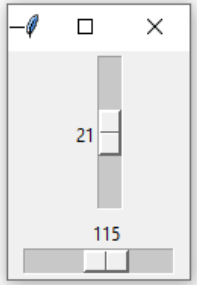



# Example Four (Scale) Horizontal & Vertical:

gui_four.py - C:/Work_CV/gui_four.py (3.7.5)

File Edit Format Run Options Window Help

```
from tkinter import *
master = Tk()
w = Scale(master, from_=0, to=42)
w.pack()


w = Scale(master, from_=0, to=200, orient=HORIZONTAL)
w.pack()
mainloop()
```



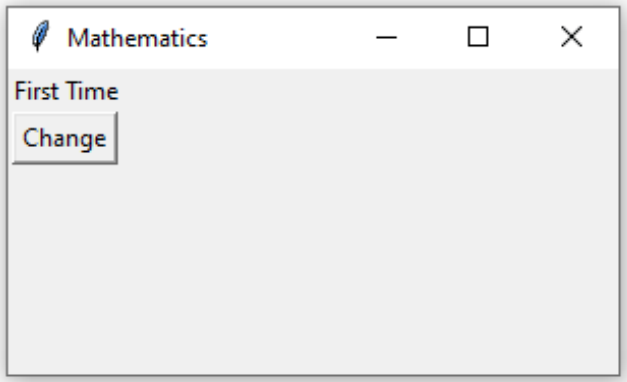
# Example Five StringVar( ):

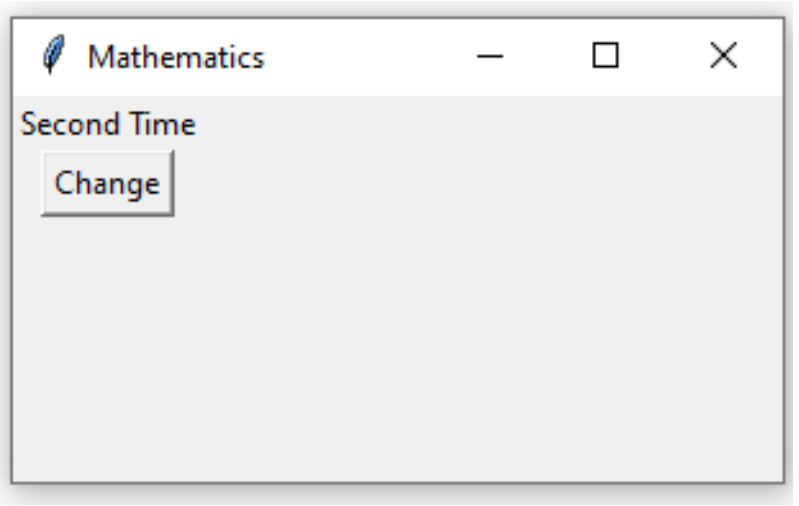
## Changing Text/Labels By Pressing the Button

gui_five.py - C:\Work_CV\gui_five.py (3.7.5)

File Edit Format Run Options Window Help

```
import tkinter as tk

def change():
    my_string_var.set('Second Time')

root = tk.Tk()
root.title('Mathematics')
root.geometry('300x150')
my_string_var = tk.StringVar()
my_string_var.set('First Time')
tk.Label(root, textvariable=my_string_var).grid()
tk.Button(root, text='Change', command=change).grid(row=1)
root.mainloop()
```



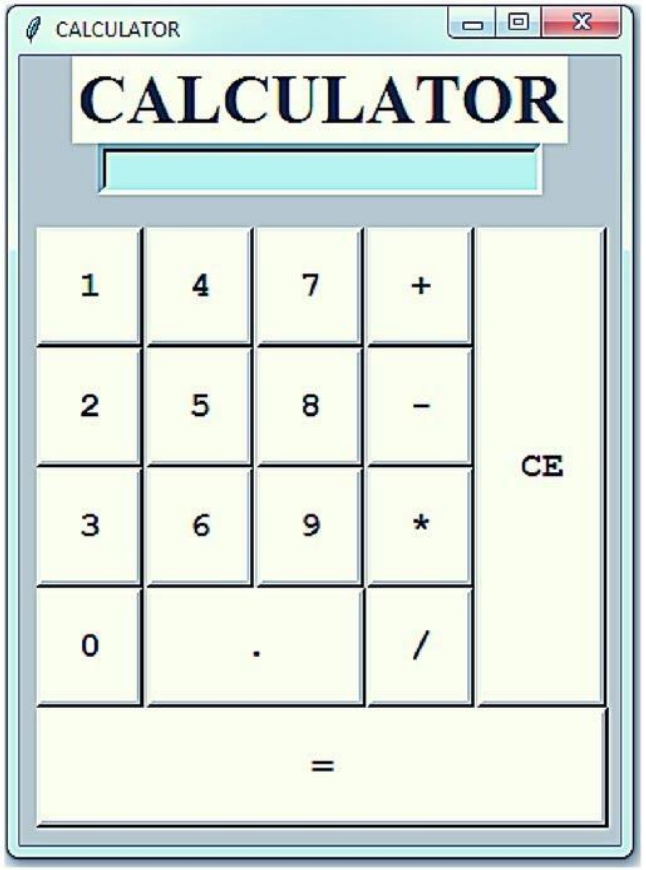



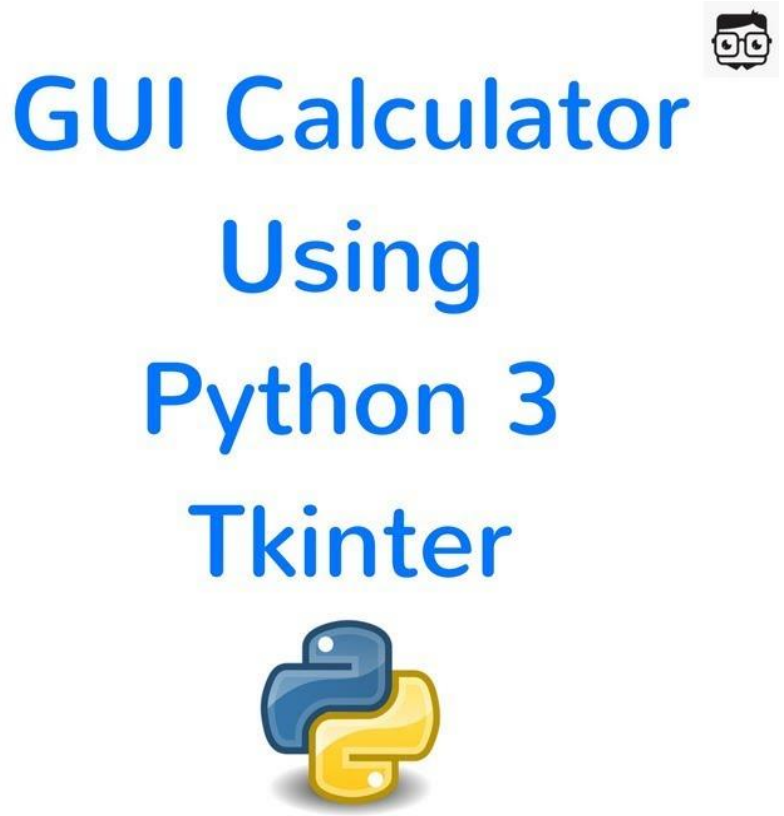

After Pressing Button

# Making Your Own Calculator using Python Tkinter

**Assignment:**

**Making a Calculator Based on Python Programming With Tkinter**



# Getting Started with Plotting : Matplotlib

```
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3]
# corresponding y axis values
y = [2,4,1]

# plotting the points
plt.plot(x, y)

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')

# giving a title to my graph
plt.title('My first graph!')

# function to show the plot
plt.show()
```

Graphs

Analyze Visualize

Plots

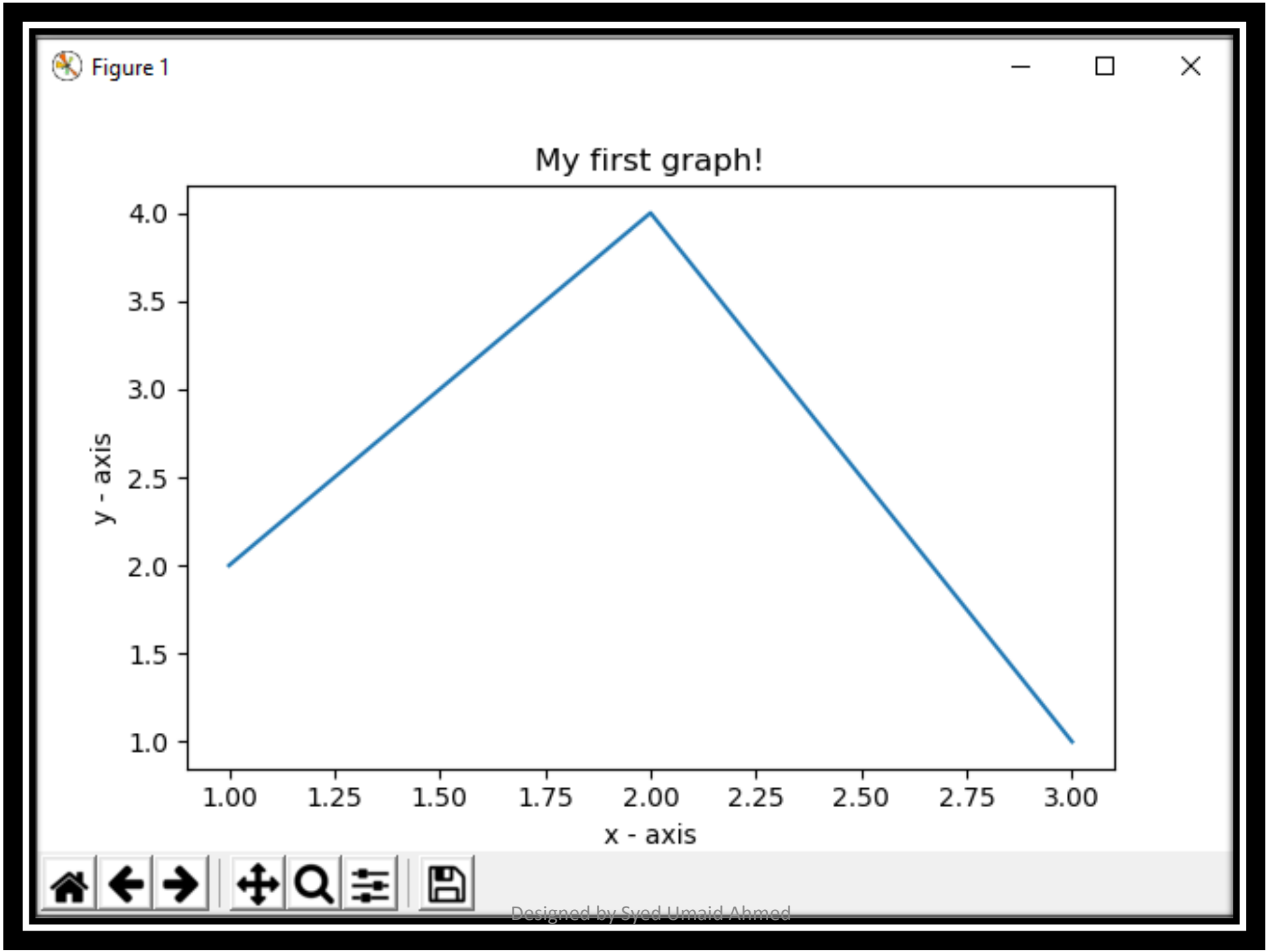


Figure 1
My first graph!
4.0
3.5
3.0
2.5
2.0
1.5
1.0
y - axis
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
x - axis

# Example # 2

plot_one.py - C:\Users\DELL\Desktop\Week 6_Plotting Lecture\plot_one.py (3.7.0)

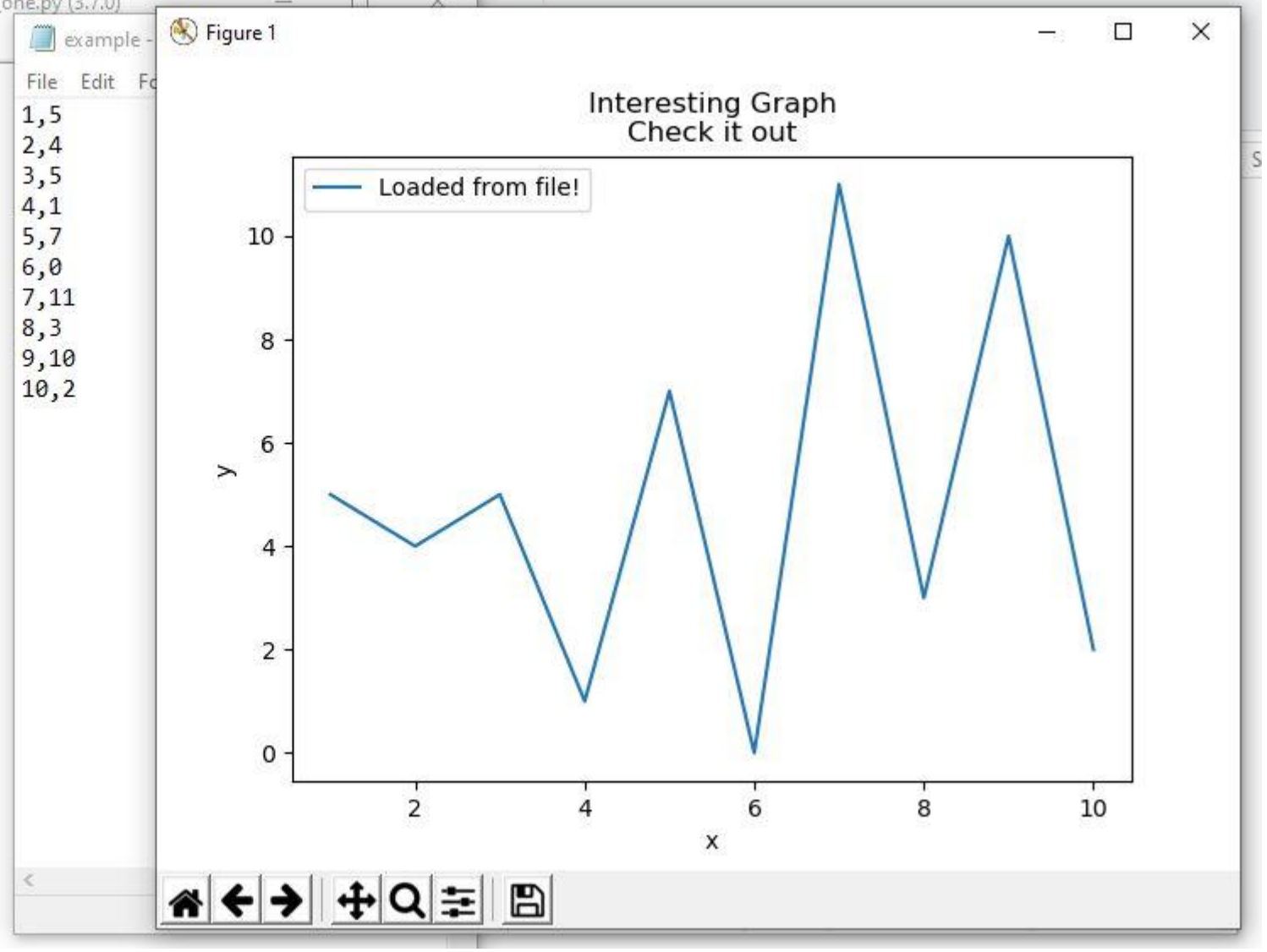
File Edit Format Run Options Window Help

```
import matplotlib.pyplot as plt
import csv

x = []
y = []

with open('example.txt','r') as csvfile:
    plots = csv.reader(csvfile, delimiter=',')
    for row in plots:
        x.append(int(row[0]))
        y.append(int(row[1]))

plt.plot(x,y, label='Loaded from file!')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Interesting Graph\nCheck it out')
plt.legend()
plt.show()
```

example -

File Edit Fo

```
1,5
2,4
3,5
4,1
5,7
6,0
7,11
8,3
9,10
10,2
```



```
import matplotlib.pyplot as plt
x1 = [1,2,3]
y1 = [2,4,1]

plt.plot(x1, y1, label = "line 1")

x2 = [1,2,3]
y2 = [4,1,3]

plt.plot(x2, y2, label = "line 2")
   # naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')
# giving a title to my graph
plt.title('Two lines on same graph!')

# show a legend on the plot
plt.legend()

# function to show the plot
plt.show()
```

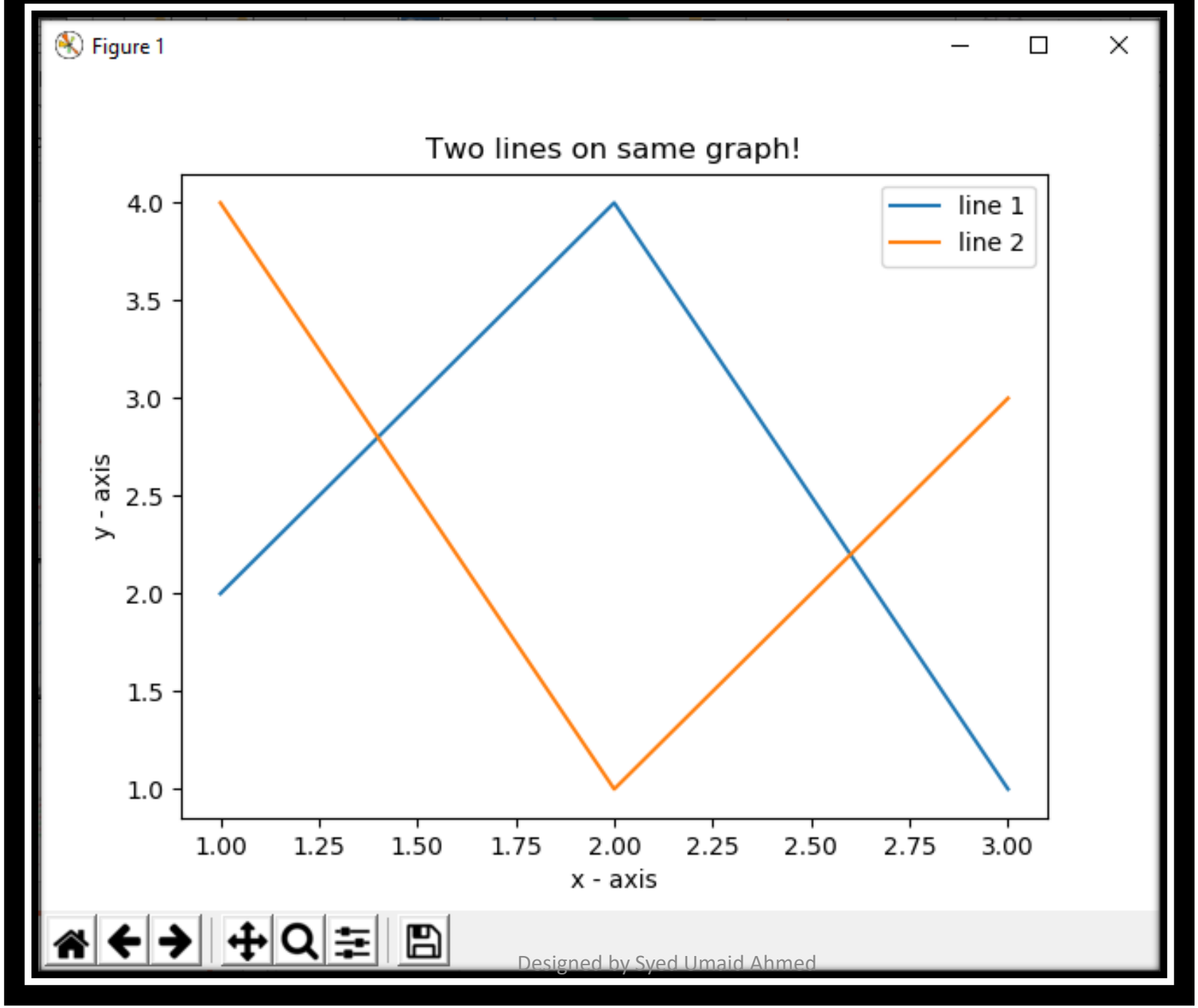


Figure 1
Two lines on same graph!
line 1
line 2
y - axis
x - axis
4.0
3.5
3.0
2.5
2.0
1.5
1.0
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00

```
import matplotlib.pyplot as plt

# x axis values
x = [1,2,3,4,5,6]
# corresponding y axis values
y = [2,4,1,5,2,6]

# plotting the points
plt.plot(x, y, color='green', linestyle='dashed', linewidth = 3,
         marker='o', markerfacecolor='blue', markersize=12)

# setting x and y axis range
plt.ylim(1,8)
plt.xlim(1,8)

# naming the x axis
plt.xlabel('x - axis')
# naming the y axis
plt.ylabel('y - axis')

# giving a title to my graph
plt.title('Some cool customizations!')

# function to show the plot
plt.show()
```

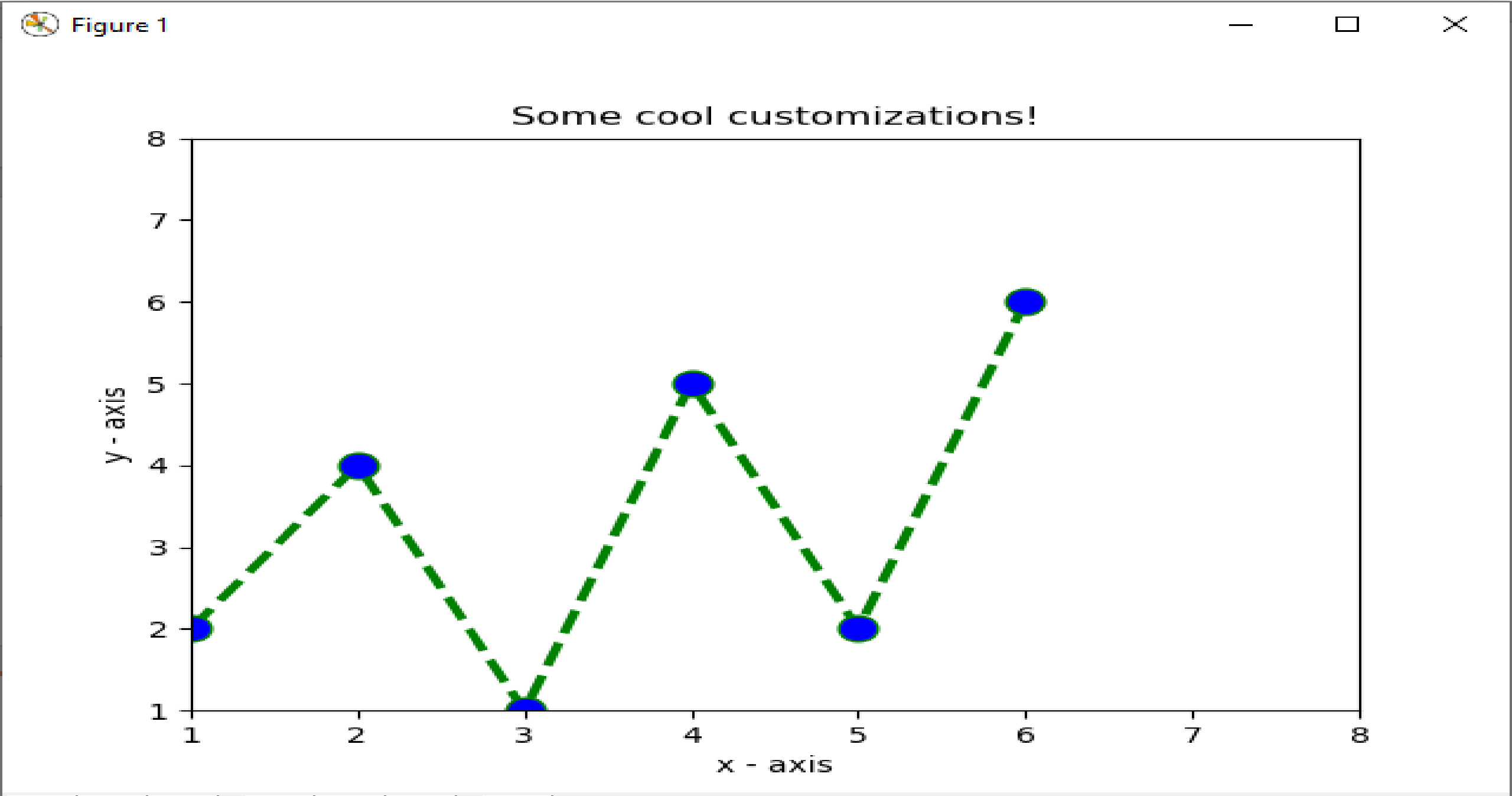


Figure 1
Some cool customizations!
y - axis
x - axis
1
2
3
4
5
6
7
8

```
import matplotlib.pyplot as plt

year = [1960, 1970, 1980, 1990, 2000, 2010]
pop_pakistan = [44.91, 58.09, 78.07, 107.7, 138.5, 170.6]
pop_india = [449.48, 553.57, 696.783, 870.133, 1000.4, 1309.1]
plt.plot(year, pop_pakistan, color='g')
plt.plot(year, pop_india, color='orange')
plt.xlabel('Countries')
plt.ylabel('Population in million')
plt.title('Pakistan India Population till 2010')
plt.show()
```

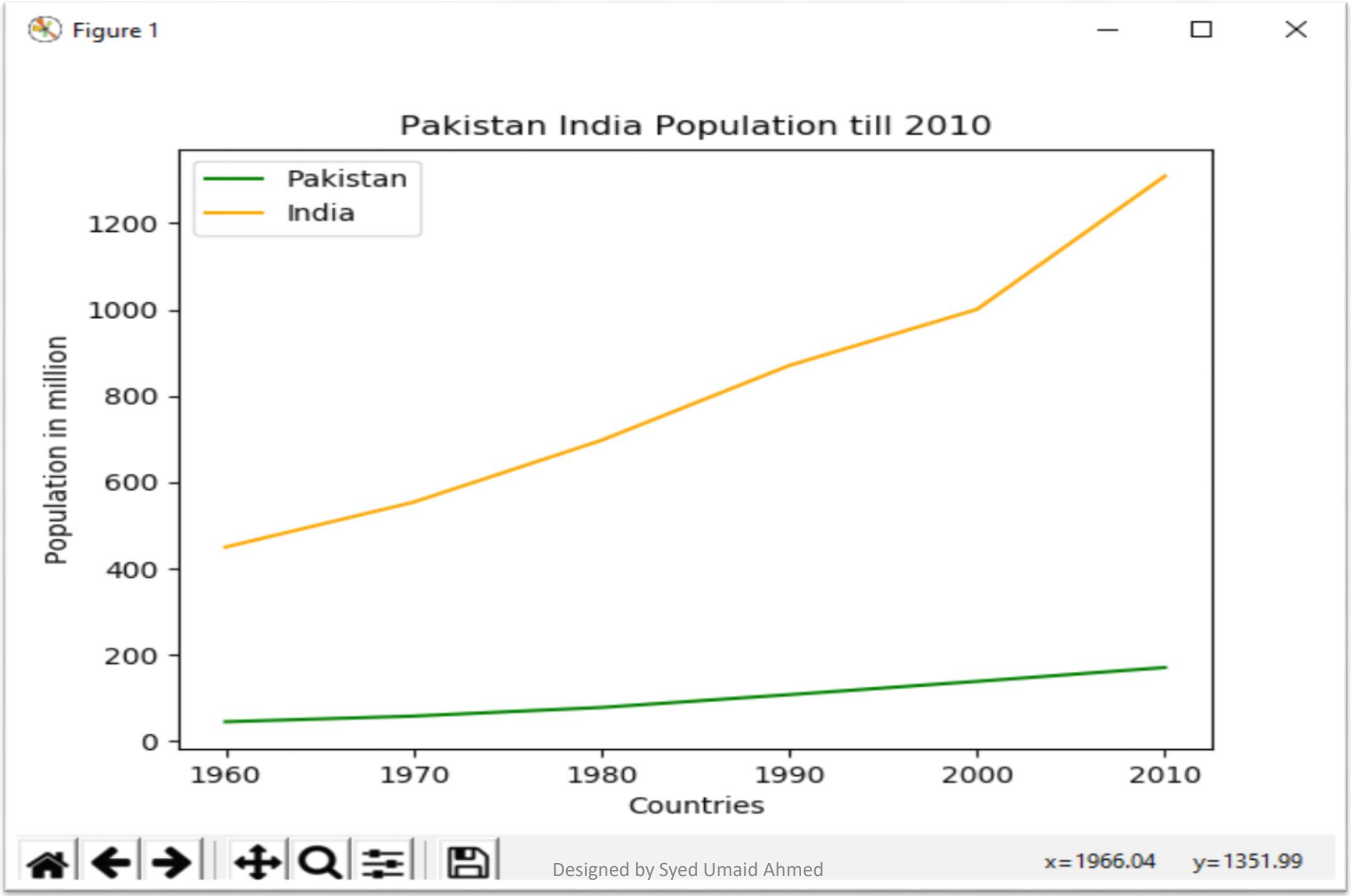


Figure 1
Pakistan India Population till 2010
Pakistan
India
Population in million
0
200
400
600
800
1000
1200
1960
1970
1980
1990
2000
2010
Countries
x=1966.04
y=1351.99

**NumPy** is a Python package. It stands for 'Numerical Python'.

It is a library consisting of multidimensional array objects and a collection of routines for processing of array.

Linear Algebra, Fourier Transform Shape Manipulation

```
import matplotlib.pylab as plt
import numpy as np

x = np.linspace(np.pi, -np.pi, 10)

plt.plot(x, np.sin(x))

plt.xlabel('Angle [rad]')

plt.ylabel('sin(x)')

plt.axis('tight')


plt.show()
```

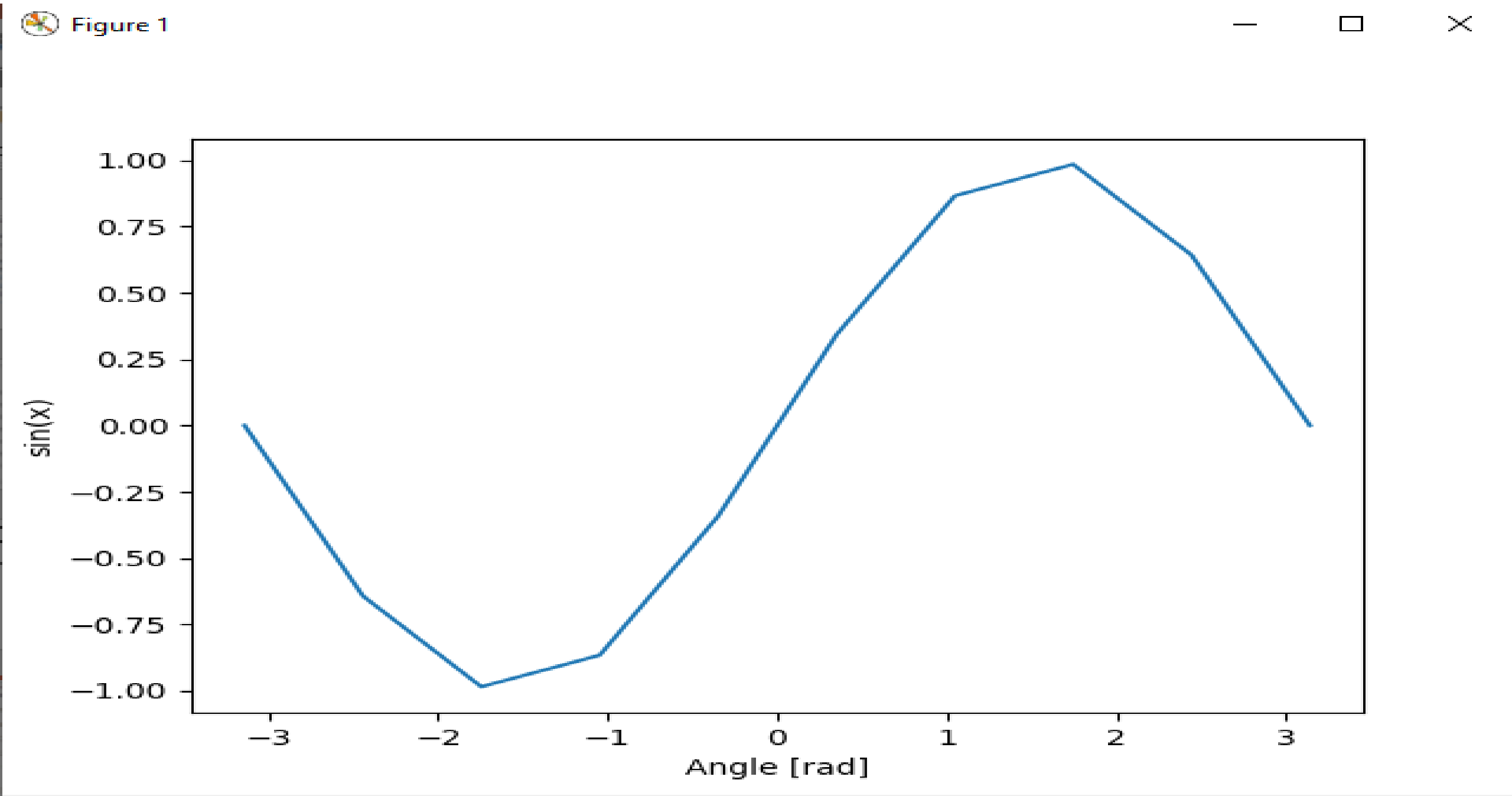


Figure 1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
sin(x)
−3
−2
−1
0
1
2
3
Angle [rad]

```
import matplotlib.pyplot as plt


goods_price = [93.95, 112.12, 104.89, 165.98, 120.78]
currency_change = [39.01, 50.29, 57.08, 69,98, 94,38]
year = [2016, 2017, 2018, 2019, 2020]


plt.plot(year, goods_price)


plt.show()
```

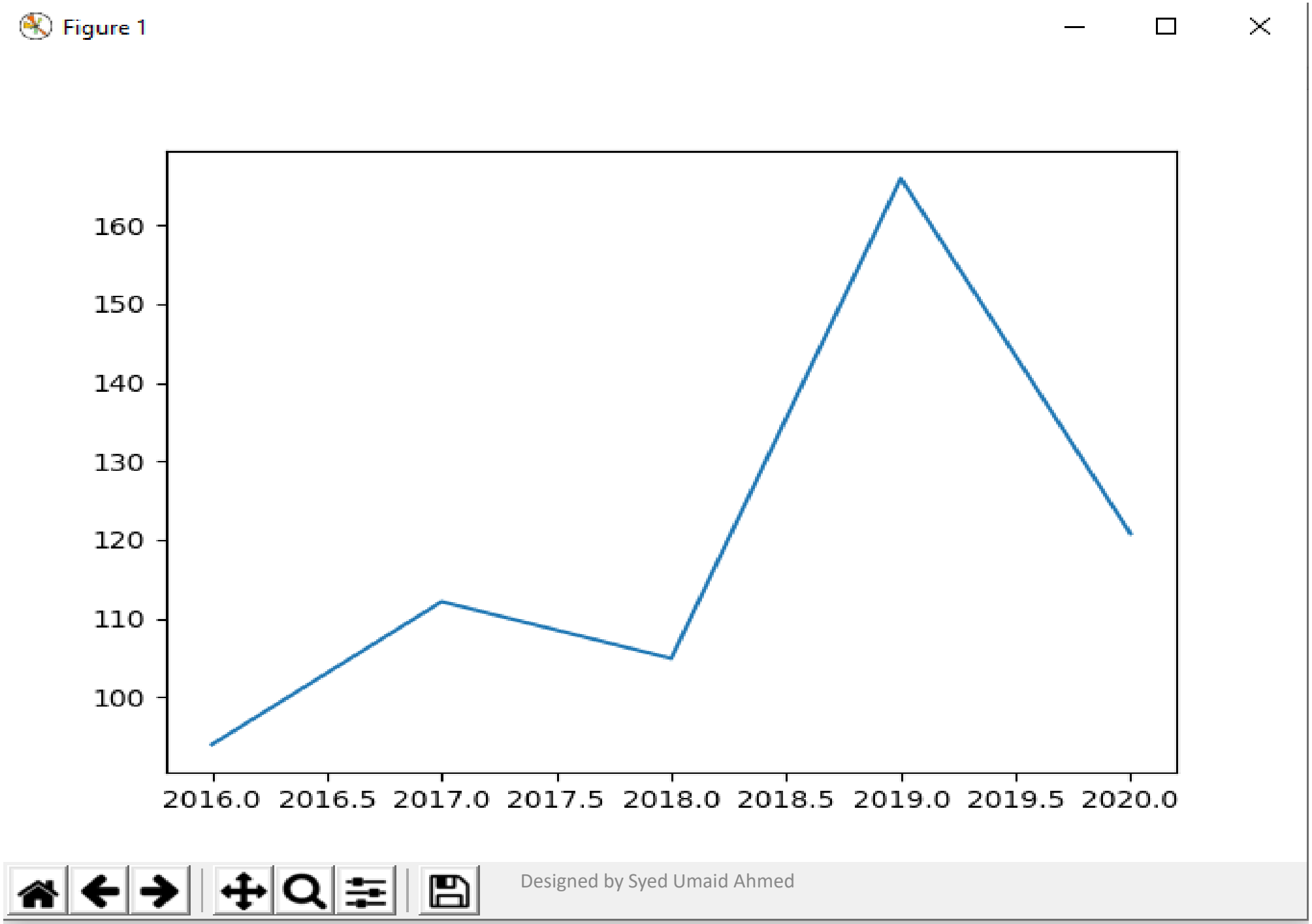


Figure 1
160
150
140
130
120
110
100
2016.0
2016.5
2017.0
2017.5
2018.0
2018.5
2019.0
2019.5
2020.0

# Can we see millions of data points from our eyes ?

```
import numpy as np
import matplotlib.pyplot as plt

in_array = np.linspace(-(2*np.pi), 2*np.pi, 40)
out_array = np.cos(in_array)

print("in_array : ", in_array)
print("\nout_array : ", out_array)



plt.plot(in_array, out_array, color = 'red', marker = "o")
plt.title("numpy.cos()")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```

```
>>>
 RESTART: C:\Users\DELL\Desktop\Mathematics Department\Plotting the Responses\Plot_3_data_points.py
in_array :  [-6.28318531 -5.96097068 -5.63875604 -5.31654141 -4.99432678 -4.67211215
 -4.34989752 -4.02768289 -3.70546826 -3.38325363 -3.061039   -2.73882436
 -2.41660973 -2.0943951  -1.77218047 -1.44996584 -1.12775121 -0.80553658
 -0.48332195 -0.16110732  0.16110732  0.48332195  0.80553658  1.12775121
  1.44996584  1.77218047  2.0943951   2.41660973  2.73882436  3.061039
  3.38325363  3.70546826  4.02768289  4.34989752  4.67211215  4.99432678
  5.31654141  5.63875604  5.96097068  6.28318531]

out_array :  [ 1.          0.94853644  0.79944276  0.56806475  0.27821746 -0.04026594
 -0.35460489 -0.63244538 -0.84519009 -0.97094182 -0.99675731 -0.91997944
 -0.74851075 -0.5        -0.20002569  0.12053668  0.42869256  0.69272435
  0.88545603  0.98705026  0.98705026  0.88545603  0.69272435  0.42869256
  0.12053668 -0.20002569 -0.5        -0.74851075 -0.91997944 -0.99675731
 -0.97094182 -0.84519009 -0.63244538 -0.35460489 -0.04026594  0.27821746
  0.56806475  0.79944276  0.94853644  1.        ]
```

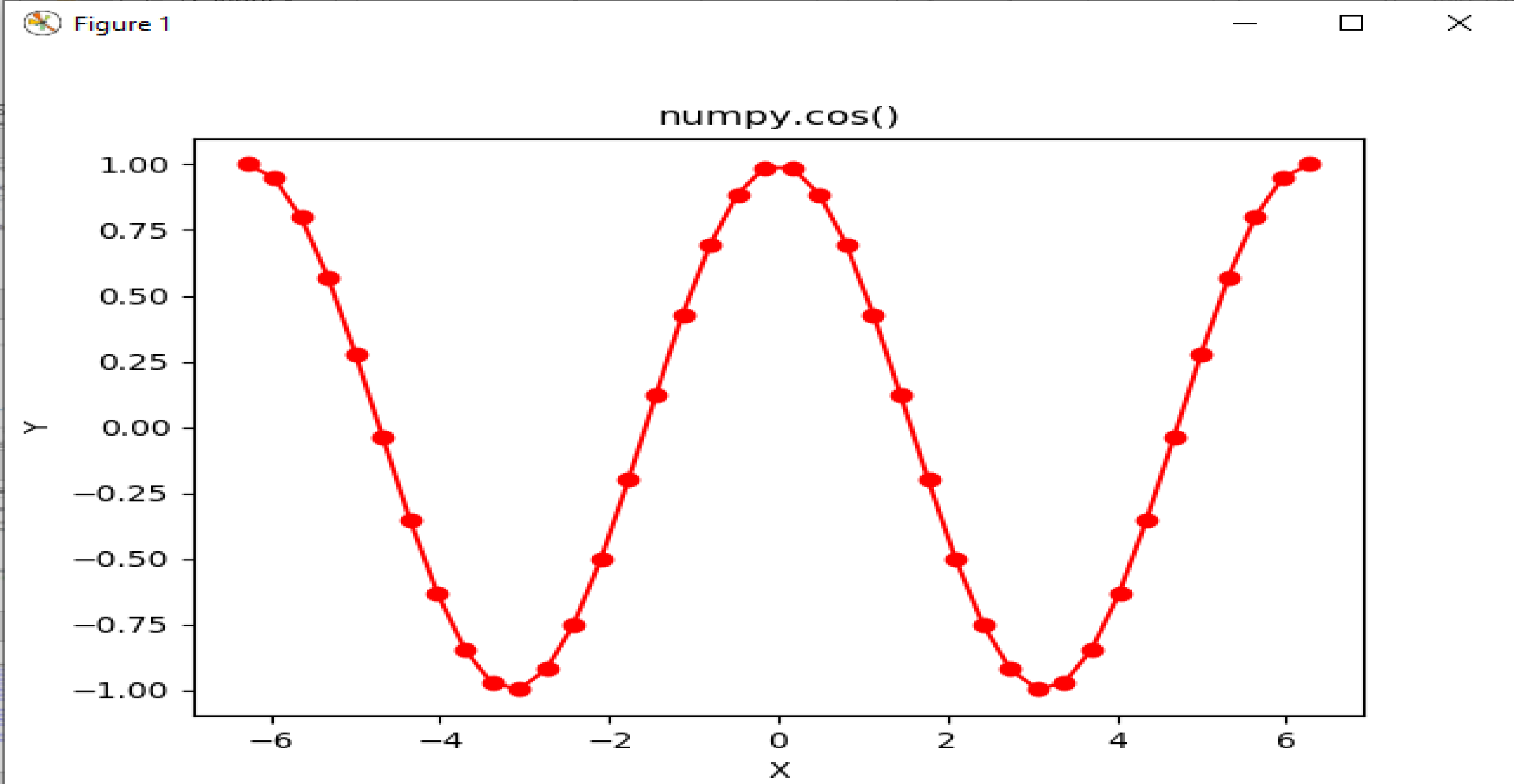


Figure 1
numpy.cos()
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Y
−6
−4
−2
0
2
4
6
X

# Scatter ?

```
import matplotlib.pyplot as plt

goods_price = [93.95, 112.12, 104.89, 165.98, 120.78]
currency_change = [39.01, 50.29, 57.08, 69.98, 94.38]
year = [2016, 2017, 2018, 2019, 2020]

plt.plot(year, goods_price)

plt.scatter(year, currency_change)

plt.show()
```

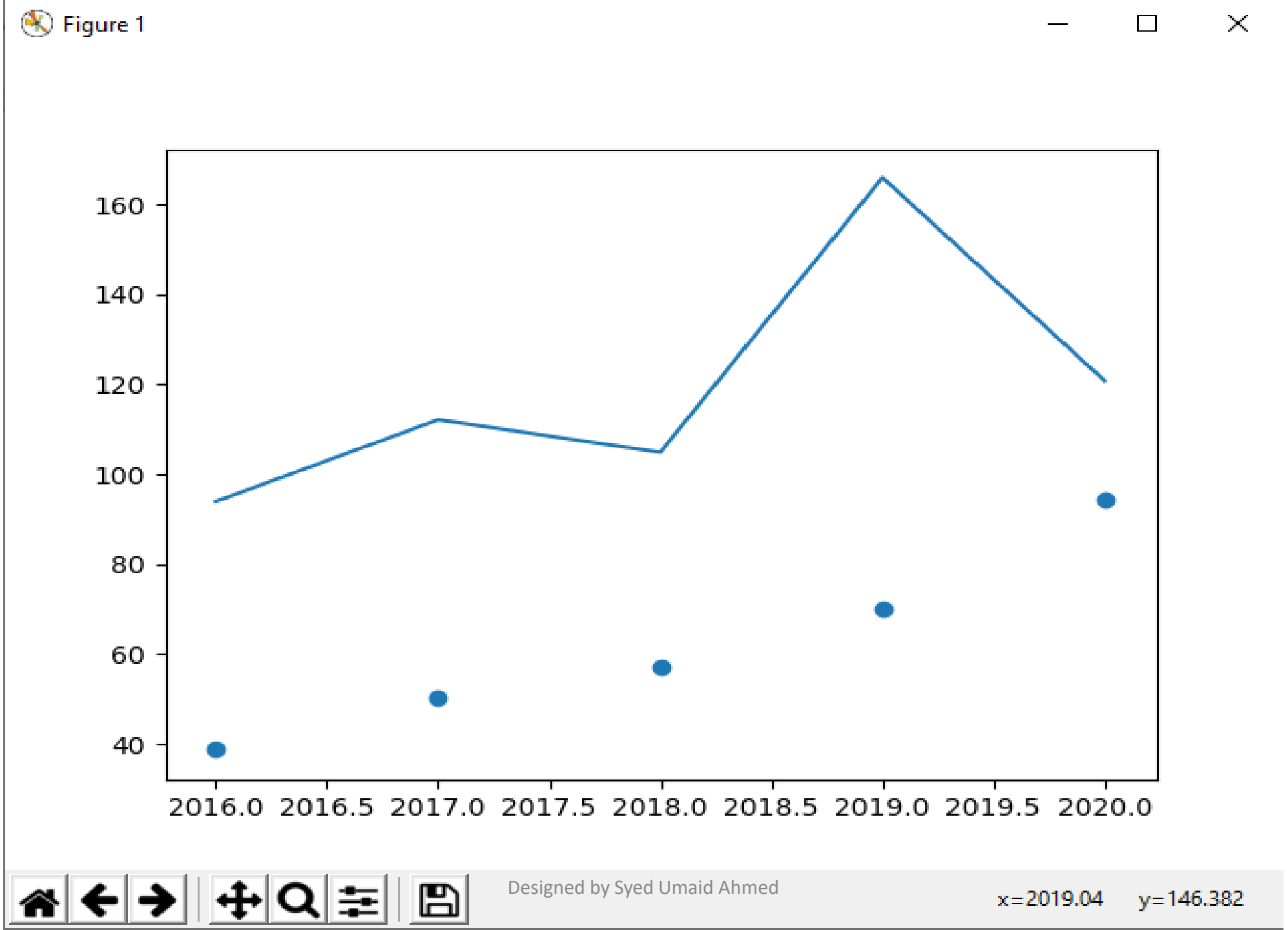


Figure 1
160
140
120
100
80
60
40
2016.0
2016.5
2017.0
2017.5
2018.0
2018.5
2019.0
2019.5
2020.0
x=2019.04
y=146.382

Linear_Algebra.py - C:\Users\Umaid\Desktop\Mathematics Department\Linear_Algebra.py (3.7.5)

File Edit Format Run Options Window Help

```
import numpy as np
A = np.array([[3,2],[1,2]])
print(A)

b = np.array([1,0])

#x = A^-1 * b


a =(np.linalg.inv(A))

#print(np.linalg.inv(A))

sol = np.dot(a,b)

print(sol)
```

# Solving Linear Equations

# X = A^-1 * b

## Advanced Method

```
#More Efficient way

sol2 = np.linalg.solve(A,b)

print(sol2)
```

## A Real-World Example

Let's see how a system of linear equation can be used to solve real-world problems.
Suppose, a fruit-seller sold 20 mangoes and 10 oranges in one day for a total of $350. The next day he sold 17 mangoes and 22 oranges for $500. If the prices of the fruits remained unchanged on both the days, what was the price of one mango and one orange? This problem can be easily solved with a system of two linear equations.

Price of mango is “x”, Price of orange is “y”

$$20x + 10y = 350$$
$$17x + 22y = 500$$

```
A = np.array([[20, 10], [17, 22]])
B = np.array([350, 500])
X = np.linalg.solve(A,B)

print(X)
```

And here is the output:

```
[10. 15.]
```

The output shows that the price of one mango is $10 and the price of one orange is $15.

# Solving Three by Three Matrices with np.linalg.det(b)

solving_three_by_three.py - C:/Users/Umaid/Desktop/Mathematics Department/solving_thre...

File Edit Format Run Options Window Help

```
import numpy as np

b = np.array([[6,1,1], [4, -2, 5], [2,8,7]])
print (b)
print (np.linalg.det(b))
print (6*(-2*7 - 5*8) - 1*(4*7 - 5*2) + 1*(4*8 - -2*2))
```

# Solving Two Equations with **sym.solve**

$$3x + 2y - 1 = 0$$

$$X + 2y = 0$$

symbolic_functions.py - C:\Users\Umaid\Desktop\Mathematics Department\symbolic_functions.py (3.7.5)

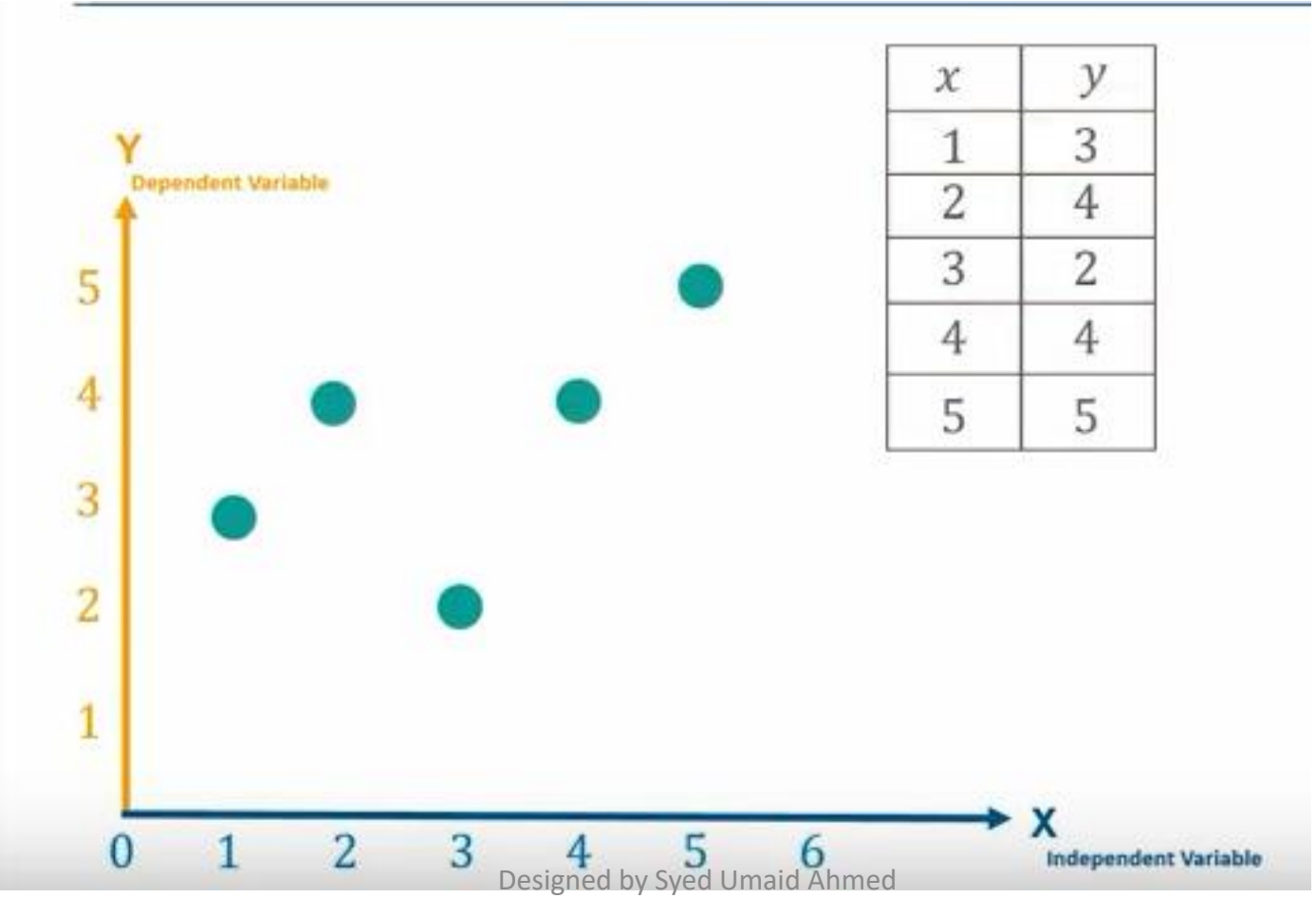
File Edit Format Run Options Window Help

```
import sympy as sym
#sym.init_printing()
x,y = sym.symbols('x y')
a=sym.linsolve([3*x+2*y-1, x+2*y], (x,y))

print(a)

sym.pprint(a)
```

# Linear Regression Algorithm

## Application: Prediction and Forecasting

## Concept: Machine Learning

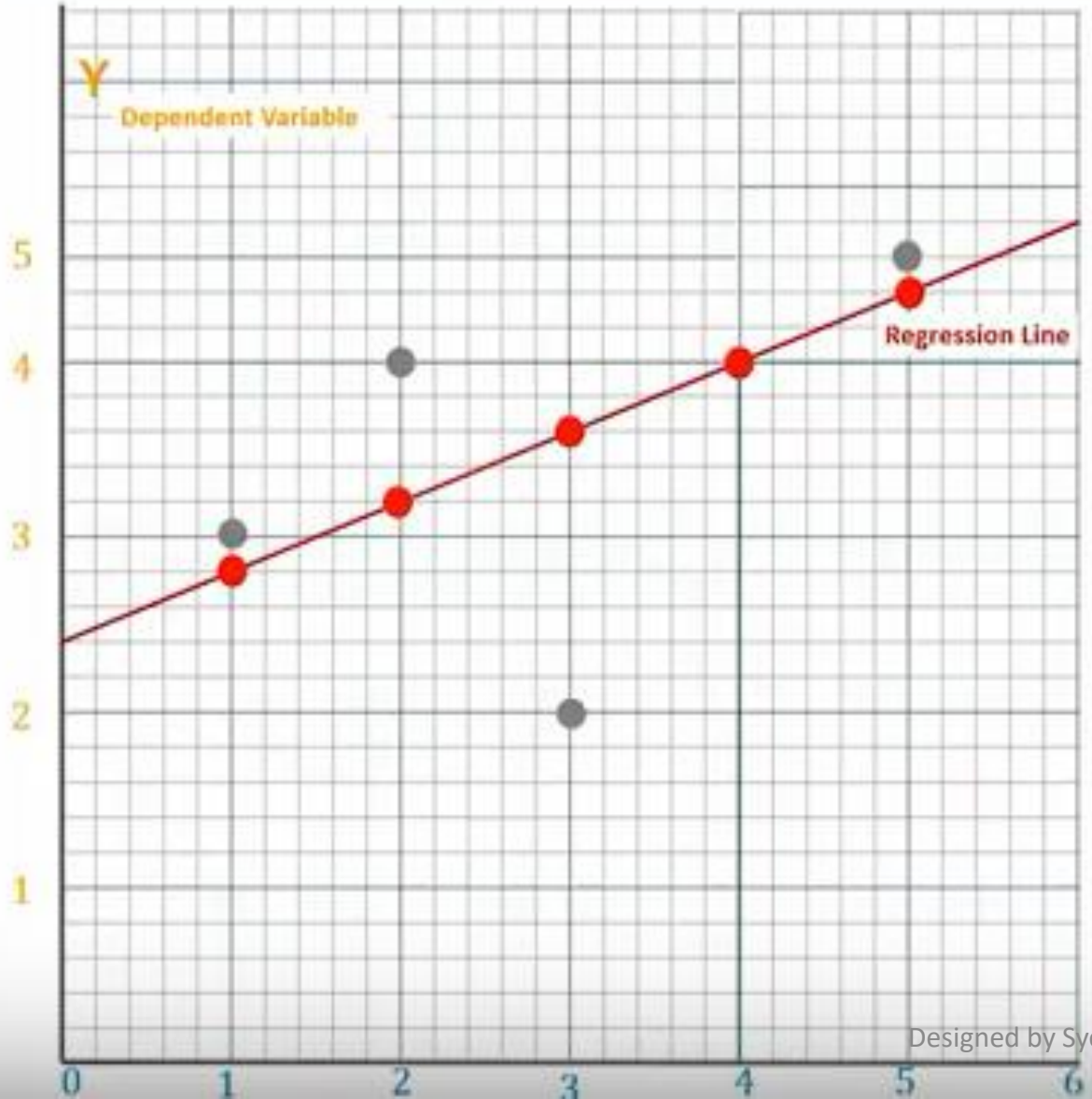



| $x$ | $y$ |
|---|---|
| 1 | 3 |
| 2 | 4 |
| 3 | 2 |
| 4 | 4 |
| 5 | 5 |

| $x$ | $y$ | $x - \bar{x}$ | $y - \bar{y}$ | $(x - \bar{x})^2$ | $(x - \bar{x})(y - \bar{y})$ |
|---|---|---|---|---|---|
| 1 | 3 | -2 | -0.6 | 4 | 1.2 |
| 2 | 4 | -1 | 0.4 | 1 | -0.4 |
| 3 | 2 | 0 | -1.6 | 0 | 0 |
| 4 | 4 | 1 | 0.4 | 1 | 0.4 |
| 5 | 5 | 2 | 1.4 | 4 | 2.8 |
| $\bar{x} = 3$ | $\bar{y} = 3.6$ | | | $\Sigma = 10$ | $\Sigma = 4$ |

$$m = \sum \frac{(x - \bar{x})(y - \bar{y})}{(x - \bar{x})^2}$$

# Mean Square Error

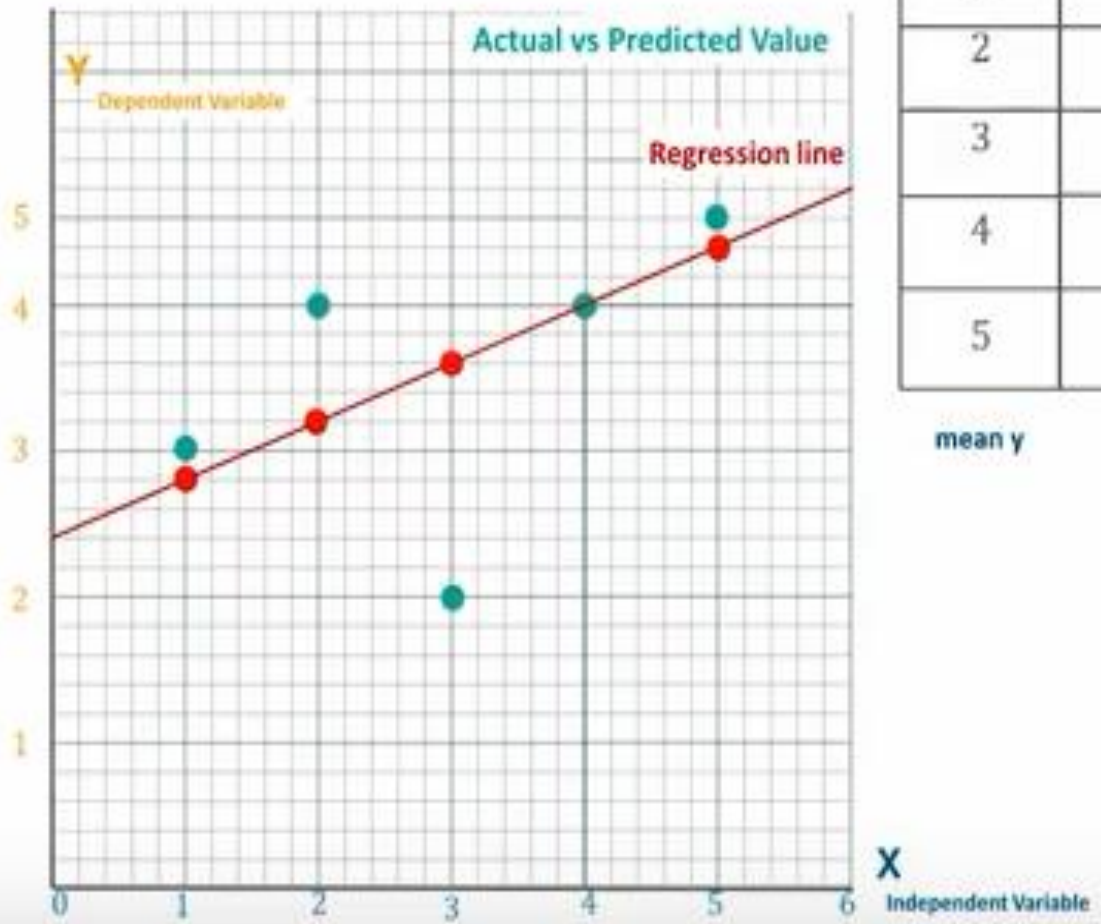



$$m = 0.4$$
$$c = 2.4$$
$$y = 0.4x + 2.4$$

For given $m = 0.4$ & $c = 2.4$, lets predict values for y for $x = \{1,2,3,4,5\}$

$$y = 0.4 \times 1 + 2.4 = 2.8$$
$$y = 0.4 \times 2 + 2.4 = 3.2$$
$$y = 0.4 \times 3 + 2.4 = 3.6$$
$$y = 0.4 \times 4 + 2.4 = 4.0$$
$$y = 0.4 \times 5 + 2.4 = 4.4$$

# Calculation of $R^2$




| $x$ | $y$ | $y - y$ | $(y - y)^2$ | $y_p$ | $(y_p - y)$ | $(y_p - y)^2$ |
|---|---|---|---|---|---|---|
| 1 | 3 | - 0.6 | 0.36 | 2.8 | -0.8 | 0.64 |
| 2 | 4 | 0.4 | 0.16 | 3.2 | -0.4 | 0.16 |
| 3 | 2 | -1.6 | 2.56 | 3.6 | 0 | 0 |
| 4 | 4 | 0.4 | 0.16 | 4.0 | 0.4 | 0.16 |
| 5 | 5 | 1.4 | 1.96 | 4.4 | 0.8 | 0.64 |
| mean y | 3.6 | | $\sum$ 5.2 | | | $\sum$ 1.6 |

$$R^2 = \frac{1.6}{5.2} = \frac{\sum \left((y_p - \bar{y}\right)^2}{\sum (y - \dot{y})^2}$$

# Lambdas Function : Comprehensive Explanation

**If we are using some function in the program only once than lambda is the best choice.**
**Start with an example problem** *(Bonus Learning: maps, reduce and filter)*

*Lambdas are called single-line functions*

*They are also known as anonymous functions*

**Example # 1**

```
cuber =[2,4,5,12,18]

def solver(n):
    return n*n*n

print(list(map(solver,cuber)))

# We use a map function here
# map takes two arguments 1st is the function,
# second is iterator
#(func,iter)
# list is also built-in of python
```

```
Run - maths_department
Run:  Chapter_4
C:\Users\DELL\PycharmProjects\maths_
[8, 64, 125, 1728, 5832]

Process finished with exit code 0
```

## Example # 2

```
cuber =[2,4,5,12,18]

#def solver(n):
#    return n*n*n
#print(list(map(solver,cuber)))

print(list(map(lambda n:n*n*n,cuber)))
```

```
Run:  Chapter_5
C:\Users\DELL\PycharmProjects\maths_department
[8, 64, 125, 1728, 5832]

Process finished with exit code 0
```

## Example # 3

```
linear = lambda x,y,z: 2*x+3*y+14*z

print(linear(1,2,3))
```

```
C:\Users\DELL\PycharmProjects\maths_
50

Process finished with exit code 0
```

## Example # 4

```
list1 =[2,4,5,12,18]
list2=[1,5,7,8,12]


print(list(map(lambda x,y:x+y, list1,list2)))
```

```
Run:  Chapter_7
C:\Users\DELL\PycharmProjects\maths
[3, 9, 12, 20, 30]

Process finished with exit code 0
```

**This will add elements of list, Check this:**

**2+1 =3**
**4+5 =9**
**5+7 =12**
**12+8 = 20**
**12+18 =30**

**Note: Use “list” with print variable**

**Example # 5**

```
list1 =[2,4,5,12,18]

c = filter(lambda x: x % 2 == 0, list1)

print(list(c))
```

```
Run:  Chapter_8
C:\Users\DELL\PycharmProjects\maths_
[2, 4, 12, 18]

Process finished with exit code 0
```

**Example # 6**

```
from functools import reduce

list1 =[2,4,5,12,18]

c = reduce(lambda x,y: x+y, list1)

print(c)
```

```
Run - maths_department
Run:  Chapter_9
C:\Users\DELL\PycharmProjects\maths_department\
41

Process finished with exit code 0
```

For Video Explanation of Week 12:

# Integration & Plotting the Responses (Outputs)

Today we will learn integration in python using “quad” rule using python programming.

The rule we are using is accurate and faster than the other rules

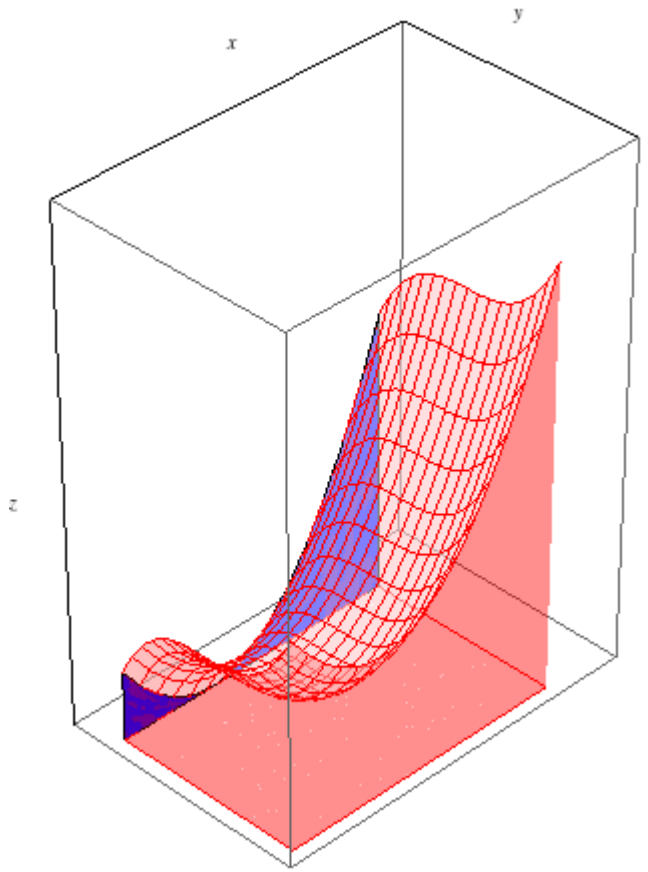
We will work on:

*1. Simple Integrals*

*2. Double Integrals*

*3. Triple Integrals*

*4. Plotting results of some questions through matplotlib*



## Question # 1

**Problem statement**

find the integral of a function f(x) from a to b i.e.

$$\int_a^b f(x)dx$$

In python we use numerical quadrature to achieve this with the scipy.integrate.quad command.

as a specific example, lets integrate

$$y = x^2$$

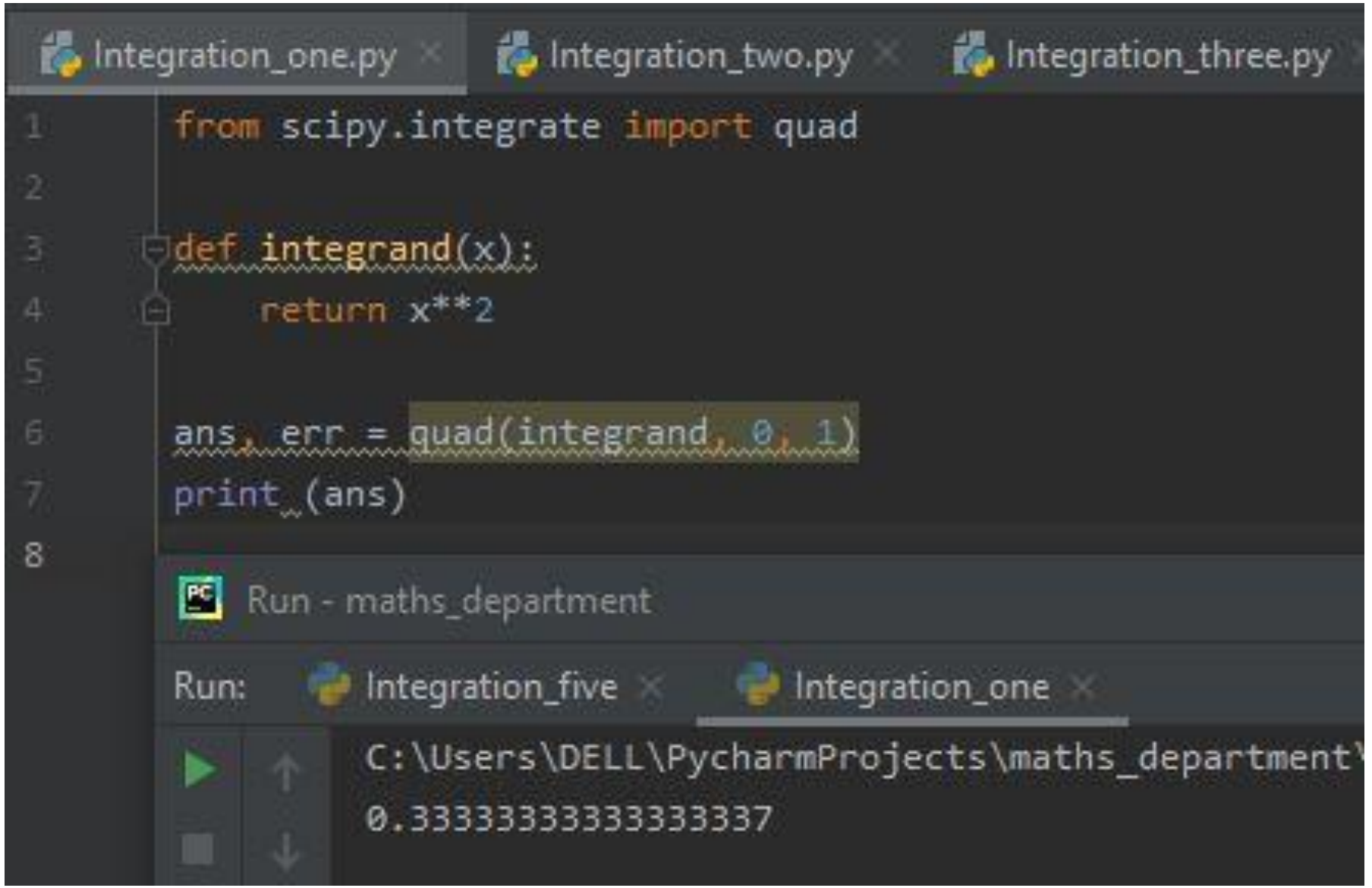
from x=0 to x=1. You should be able to work out that the answer is 1/3.

## Manual Solution # 1

$$\int_0^1 x^2 = \frac{x^3}{3}\Big|_0^1$$

$$= \frac{1}{3} - 0 = 1/3 \quad \text{Ans!}$$

## Python Solution # 1



# Question # 2

## double integrals

we use the scipy.integrate.dblquad command

Integrate $f(x, y) = ysin(x) + xcos(y)$ over

$\pi <= x <= 2\pi$

$0 <= y <= \pi$

i.e.

$$\int_{x=\pi}^{2\pi} \int_{y=0}^{\pi} ysin(x) + xcos(y) dydx$$

## Manual Solution # 2

$f(x, y) = y \sin(x) + x \cos(y)$

$$\int_{\pi}^{2\pi} \int_{0}^{\pi} y \sin(x) + x \cos(y) \, dy \, dx$$

$$\int_{\pi}^{2\pi} \left[ \int_{0}^{\pi} y \sin x + x \cos y \, dy \right] dx$$

$$\int_{\pi}^{2\pi} \left[ \frac{\sin x}{2} \left[ y^2 \right]_0^{\pi} + x \left( +\sin y \right)_0^{\pi} \right] dx$$

$$\int_{\pi}^{2\pi} \left[ \frac{\sin x}{2} (\pi^2 - 0) + x (+\sin \pi) \right] dx$$

$$\int_{\pi}^{2\pi} \frac{\sin x (\pi^2)}{2} + x \sin \pi \, dx$$

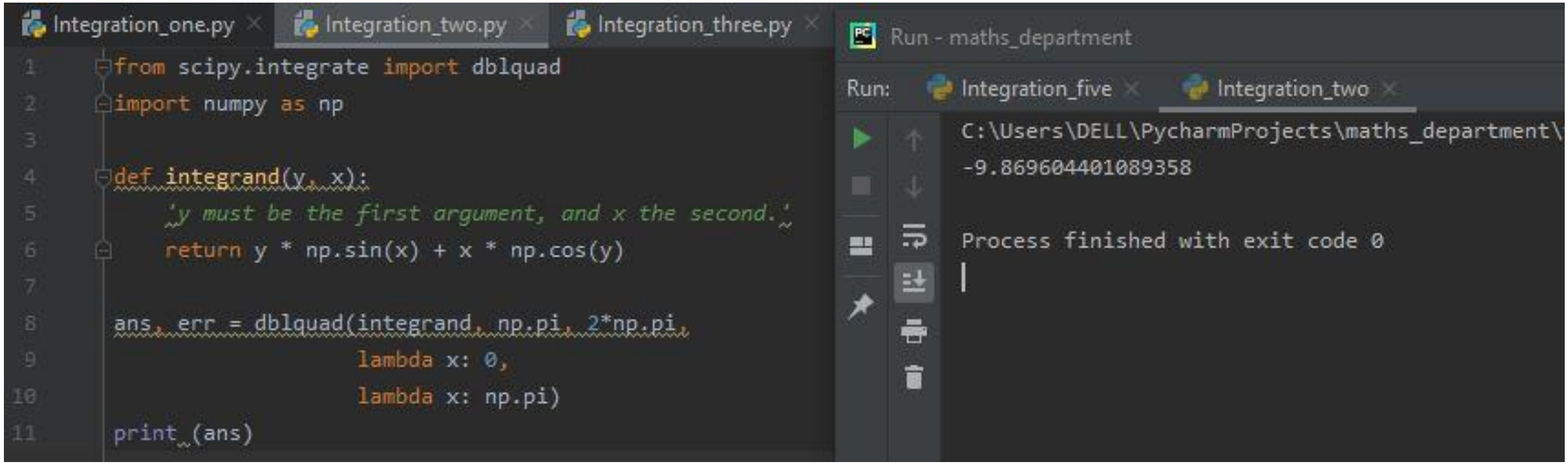
$$\frac{\pi^2}{2} (\cos x)_{\pi}^{2\pi} + \frac{\sin \pi}{2} \left[ x^2 \right]_{\pi}^{2\pi}$$

$$\frac{\pi^2}{2} (\cos 2\pi - \cos \pi) + \frac{\sin \pi}{2} \left[ 4\pi^2 - \pi^2 \right]$$

$$-\pi^2 + \frac{\sin \pi}{2} \left[ 3\pi^2 \right]$$

$$= -9.868$$

## Python Solution # 2




*For the third and upcoming questions I will only guide python solutions,*

*you have to do manual solutions by yourselves*

## Question # 3

we use the tplquad command to integrate $f(x, y, z) = ysin(x) + zcos(x)$ over the region

$0 <= x <= \pi$

$0 <= y <= 1$

$-1 <= z <= 1$

## Python Solution # 3

Integration_one.py | Integration_two.py | Integration_three.py

```
from scipy.integrate import tplquad
import numpy as np

def integrand(z, y, x):
    return y * np.sin(x) + z * np.cos(x)

ans, err = tplquad(integrand,
                   0, np.pi,  # x limits
                   lambda x: 0,
                   lambda x: 1,# y limits
                   lambda x,y: -1,
                   lambda x,y: 1)# z limits
print(ans)
```

Run - maths_department

Run: Integration_five | Integration_three

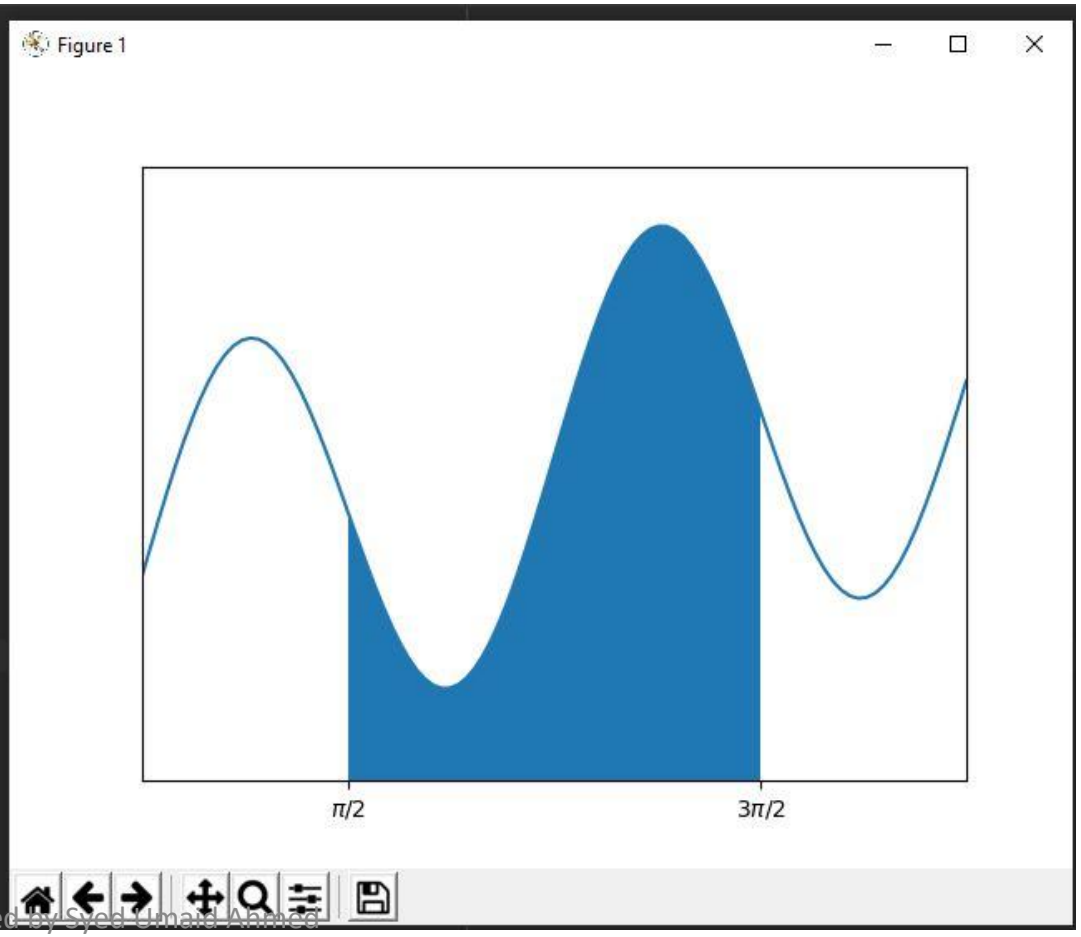
```
C:\Users\DELL\PycharmProjects\maths_depar
1.9999999999999998

Process finished with exit code 0
```

## Question # 4

$$\int_{\pi/2}^{3\pi/2} \sin(0.2x) + \sin(2x) + 1 \, dx$$

## Python Solution with Plot

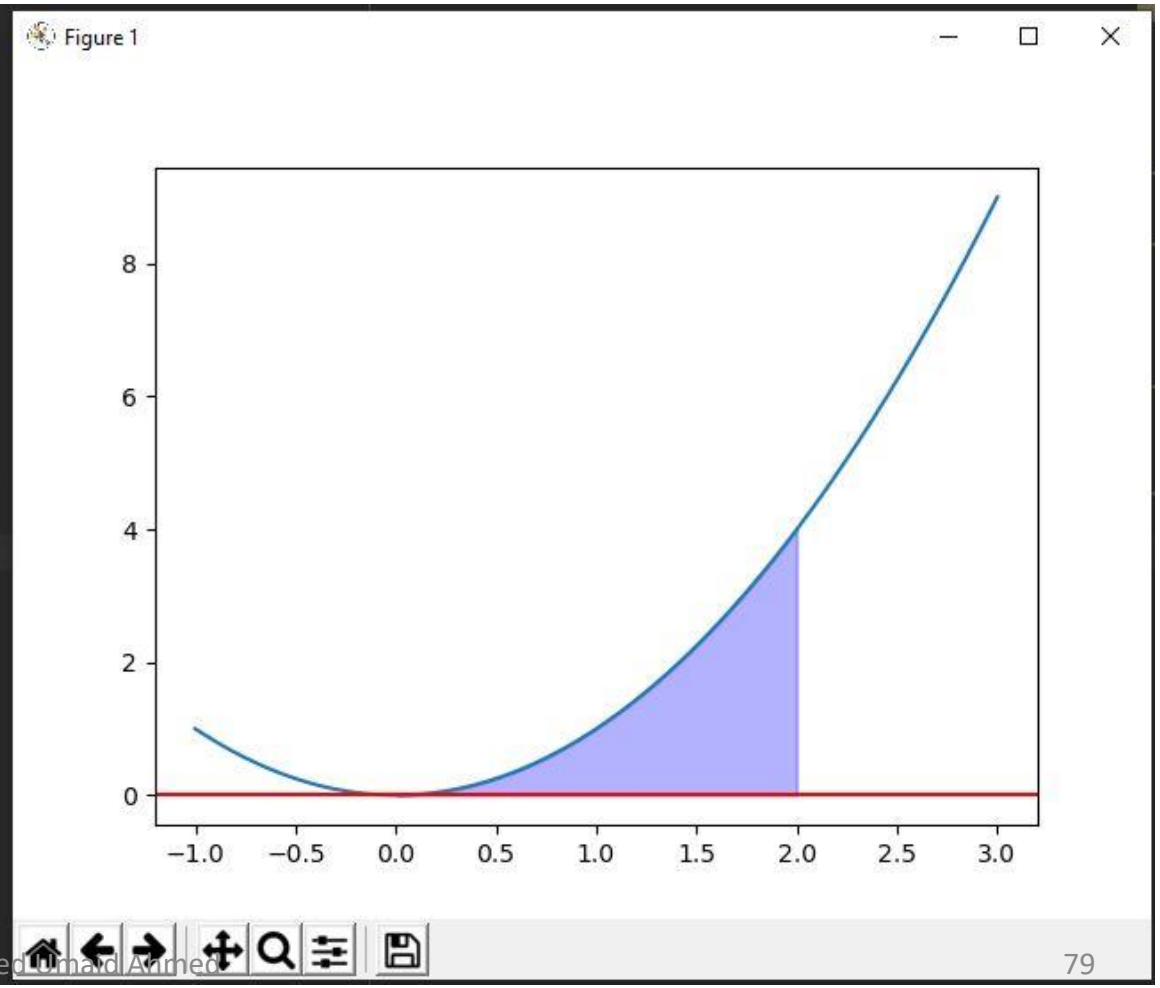
```
import numpy as np
import matplotlib.pyplot as plt


f = lambda x: np.sin(0.2*x) + np.sin(2*x) + 1

x = np.linspace(0,2*np.pi,100)
y = f(x)
print(y)
plt.plot(x,y)

X = np.linspace(np.pi/2,3*np.pi/2,100)
Y = f(X)
print(y)
plt.fill_between(X,Y)

plt.xticks([np.pi/2,3*np.pi/2],['$\pi/2$','$3\pi/2$']); plt.yticks([]);
plt.xlim([0,2*np.pi]); plt.ylim([0,3]);
plt.show()
```



# Question # 5 and Python Solution with the Plot

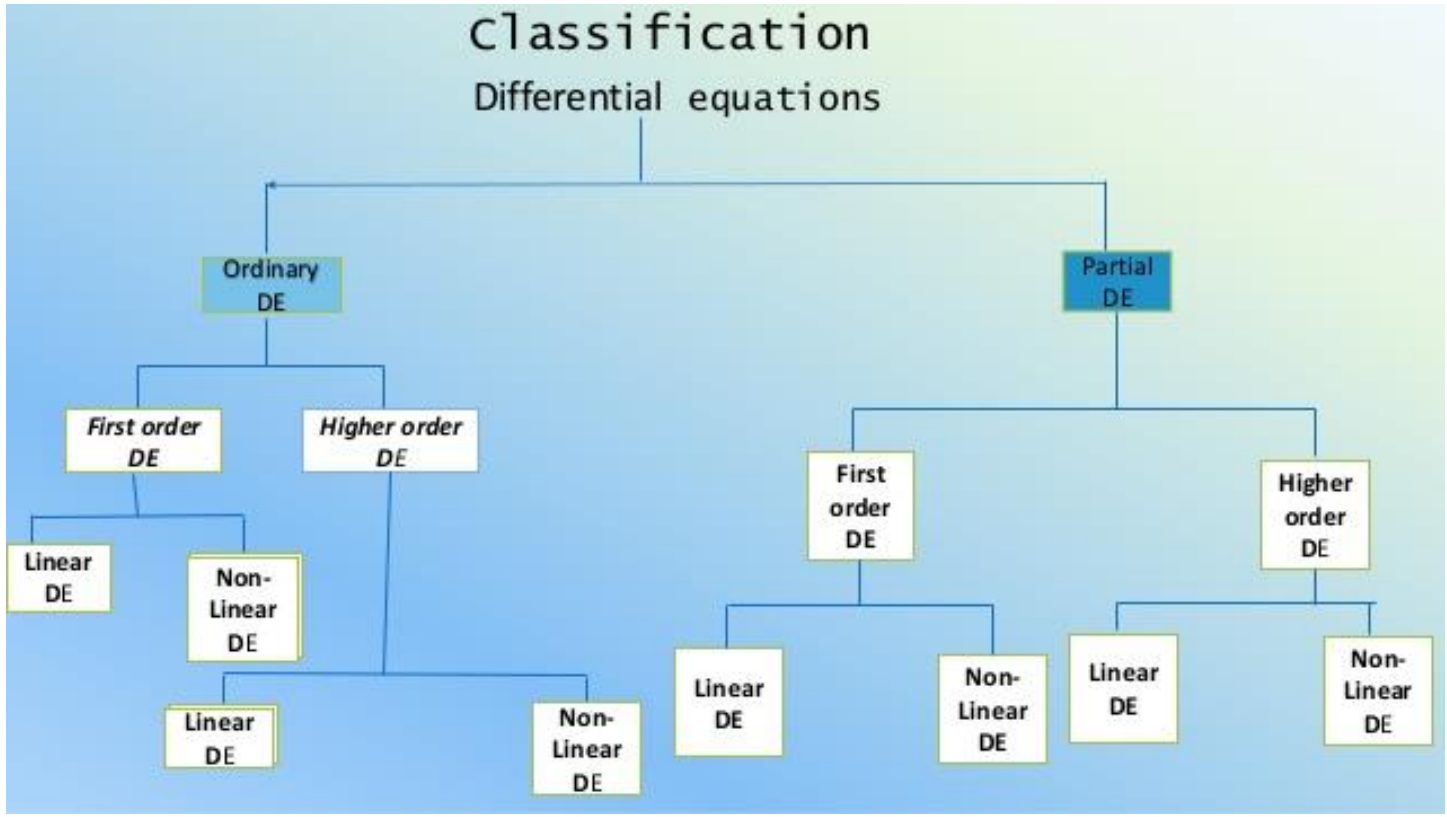
$$y = \int_{-1}^{3} x^2$$

```python
import numpy as np
import matplotlib.pyplot as plt


x = np.linspace(-1,3,1000)

def f(x):
    return x**2


plt.plot(x,f(x))
plt.axhline(color='red')

plt.fill_between(x, f(x), where=[(x>0) and (x<2) for x in x], color='blue', alpha=0.3)

# alpha is for transparency

plt.show()
```



# Ordinary Differential Equation : Python

An equation containing the derivatives of one or more dependent variables with respect to one or more independent variable is said to be a differential equation




*We will study ODE through python in this lecture*

# Applications:

- **It is an essential tool for describing nature of the physical universe**
- **These helps to produce shapes, interior and exterior designs of machines**
- **Modeling electrical circuits**
- **Used in the laws of physics i.e. Newton law of cooling**
- **Artificial Intelligence, Computer Vision and Machine Learning**

# Past Practices: (Manual)

$$\frac{dy}{dt} = -2yt \qquad y(0) = 1$$

$$\frac{1}{-2y}\, dy = t\, dt$$

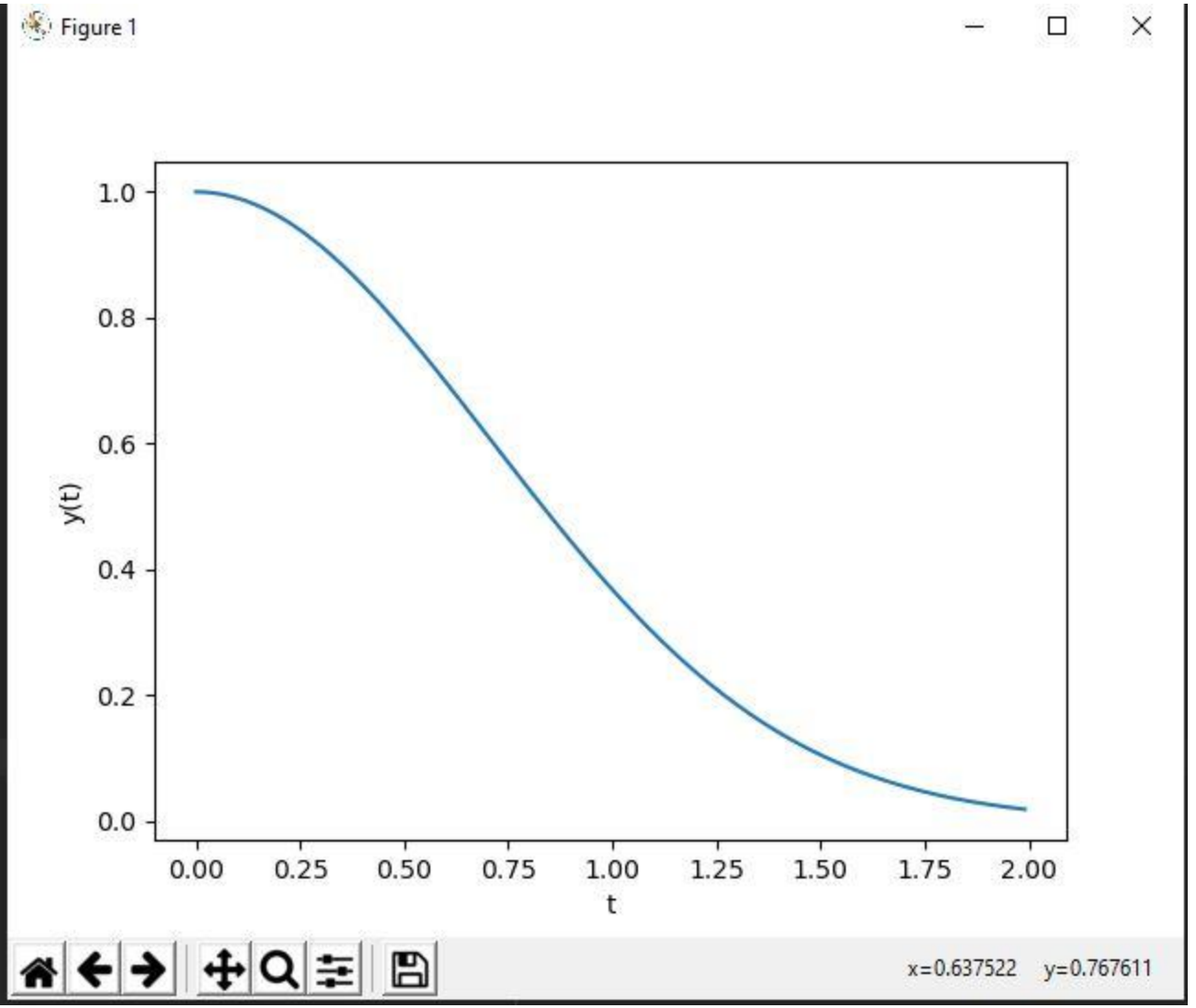
$$-\frac{1}{2}\int \frac{1}{y}\, dy = \int t\, dt$$

$$-\frac{1}{2} \ln y = \frac{t^2}{2} + C$$

**We are not interested in general solutions because these have been done in earlier studies, If someone really Want to see general solution search "dsolve"**

# Solution Programming & Graphical

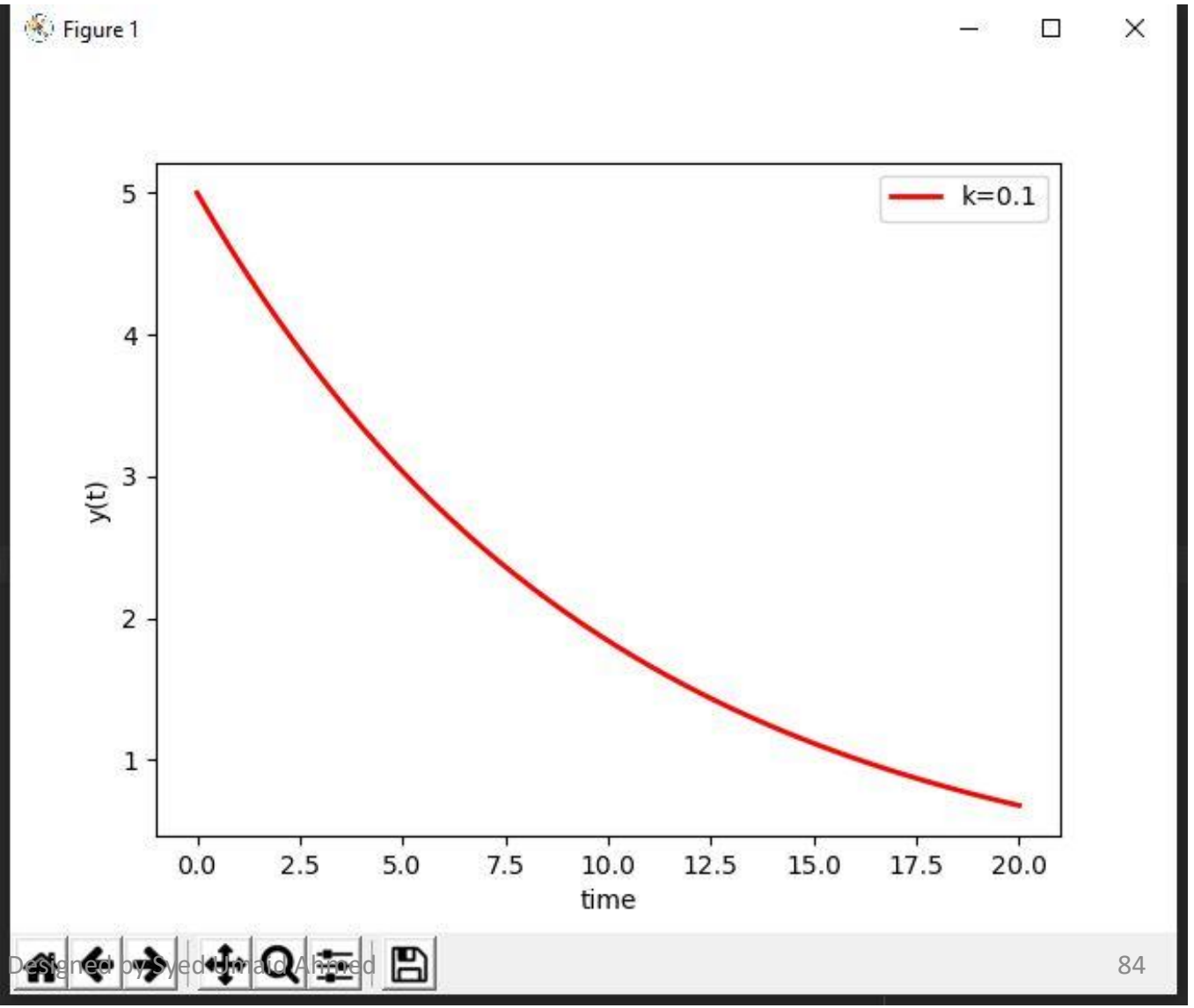
```
from scipy.integrate import odeint

import numpy as N

import matplotlib
import pylab

def f(y, t):
    return -2 * y * t

y0 = 1
a = 0
b = 2

t = N.arange(a, b, 0.01)  # values of t for which we require the solution y(t)
y = odeint(f, y0, t)  # actual computation of y(t)....func, value, put into


pylab.plot(t, y)
pylab.xlabel('t'); pylab.ylabel('y(t)')
pylab.show()
```



## Example Question:

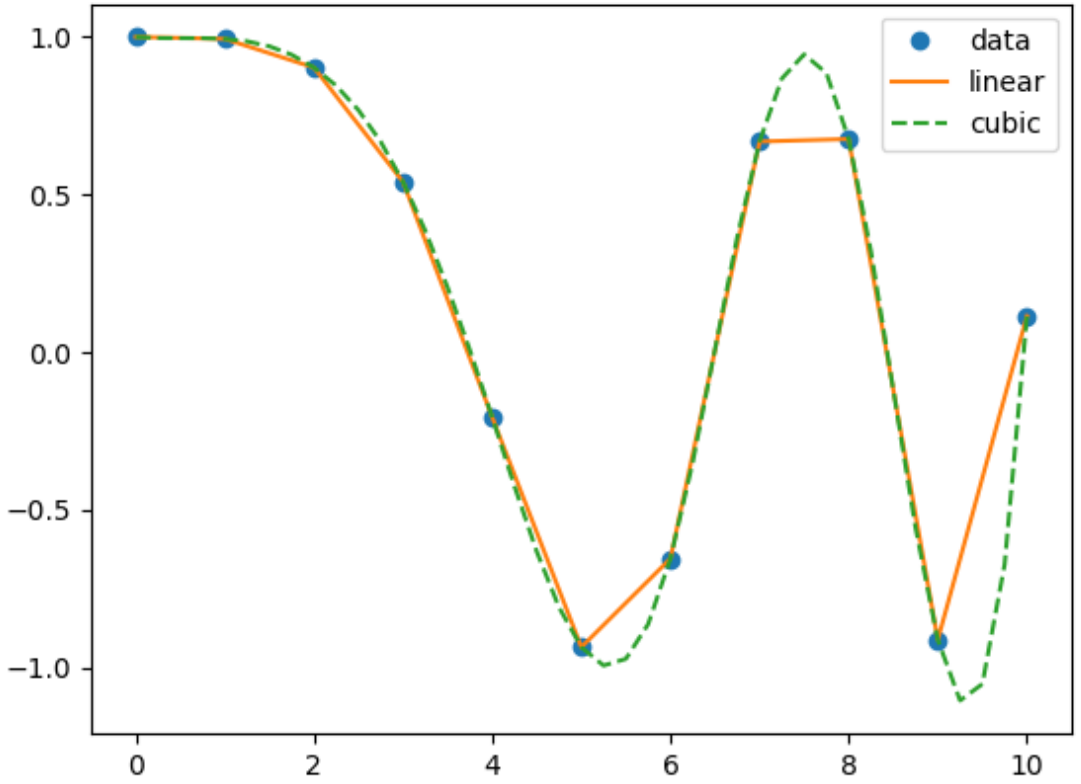
$$\frac{dy(t)}{dt} = -Ky(t) \qquad y(0) = 5$$

```python
# function that returns dy/dt
def model(y,t,k):
    dydt = -k * y
    return dydt

# initial condition
y0 = 5

# time points
t = np.linspace(0,20)

# solve ODEs
k = 0.1
y1 = odeint(model,y0,t,args=(k,))

# plot results
plt.plot(t,y1,'r-',linewidth=2,label='k=0.1')


plt.xlabel('time')
plt.ylabel('y(t)')
plt.legend()
plt.show()
```



# Interpolation with Python Programming

In Interpolation we take data and draw curve between data points. You are given the points And you have to draw or connect them.



**Example Questions:**

$$y = e^{-x/4} * x$$

$$y = \sin x$$

Put the values and find the answers, How we can put the values ?

What is the use of interpolation in it ?

# Fourier Transform with Python

Fourier transform is a function that transforms a time domain signal into frequency domain.

The function accepts a time signal as input and produces the frequency representation

of the signal as an output.

Every signal in the real world is a time signal and is made up of many sinusoids

of different frequencies.

So, time domain signal can be converted into the frequency domain to view

different frequency components.

Fourier transform doesn’t change the signal.

It just provides a different view to analyze your time signal because some

properties and features of the signal can be fully explored in the frequency domain.

For Real life applications of Fourier Transform Follow the Lecture of Professor

Dennis Freeman from MIT University:

https://www.youtube.com/watch?v=tp_MdKz3fC8

# Mathematics Department Karachi University


## Engineer Syed Umaid Ahmed
## Lecturer

***BE (EE), ME (Mechatronics) Contd. NED***
***Senior Team Lead IoT Department Zaaviyah Solutions***
***Trainer & Senior Raspberry Pi Developer at Projexels***
***Embedded Design Engineer at TechOzean***

***For Queries & Suggestions:***

**Email Address:** syedumaidahmed96@gmail.com
**GitHub Profile:** www.github.com/SyedUmaidAhmed
**Research Profile Google Scholar:** https://scholar.google.com/citations?user=g9ZAtyEAAAAJ&hl=en
**YouTube Channel Link:** https://www.youtube.com/channel/UCGP_OdkIWCuBdoc1EdxWsVQ/featured?view_as=subscriber
**Contact:** +923323117626