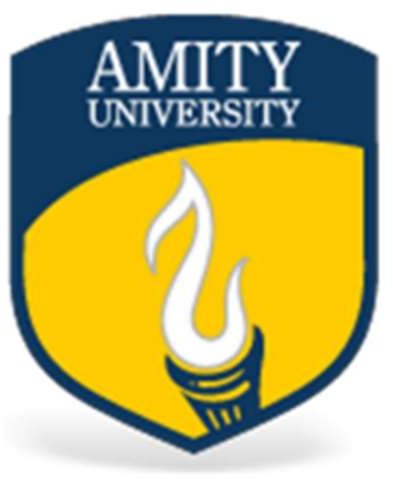


Laboratory Manual

**Amity University Chhattisgarh Amity School of Institute & Technology**

Batch: - 2020 – 2024

**Enrollment Number: - A80105220077**

This is to certify that this is a bonafide record of the work done by **Mr. Jay Patel** Enrollment number **A80105220077** of B. Tech **Computer Science & Engineering semester V**, from Amity School of Engineering & Technology, Amity University Chhattisgarh in the **Computing Using Python Lab** with Course Code **CSE3507**.

University Examination held on .

Faculty in-charge Director - ASET

Examiner – 1

Examiner – 2

# # NumPy Array

**Exp. 1: Create a NumPy array.**

Program:

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr)

print(type(arr))

Output:

[1 2 3 4 5]

<class 'numpy.ndarray'> **Exp. 2: Create a 0-D array.** Program:

import numpy as np arr = np.array(42) print(arr)

Output:

42

# Exp. 3: Create a 1-D array.

Program:

import numpy as np

arr = np.array([1, 2, 3, 4, 5]) print(arr)

Output:

[1 2 3 4 5]

# Exp. 4: Create a 2-D array.

Program:

import numpy as np

arr = np.array([[1, 2, 3],

[4, 5, 6]])

print(arr)

Output: [ [1 2 3]

[4 5 6]]

# Exp.5: Create a 3-D array.

Program:

import numpy as np

arr = np.array([[[1, 2, 3],

[4, 5, 6]], [[1, 2, 3],

[4, 5, 6]]])

print(arr)

Output:

[ [1 2 3]

[4 5 6]]

[[1 2 3]

[4 5 6]]]

# Exp. 6: WAP to check the dimensions of array.

Program:

import numpy as np

a = np.array(42)

b = np.array([1, 2, 3, 4, 5])

c = np.array([[1, 2, 3],

[4, 5, 6]])

d = np.array([[[1, 2, 3],

[4, 5, 6]], [[1, 2, 3],

[4, 5, 6]]])

print(a.ndim) print(b.ndim) print(c.ndim) print(d.ndim)

Output: 0

1

2

3

# Exp. 7: WAP to get the first element from the array.

Program:

import numpy as np

arr = np.array([1, 2, 3, 4]) print(arr[0])

Output :

1

# Exp. 8: WAP to get the third and fourth element from the array.

Program:

import numpy as np

arr = np.array([1, 2, 3, 4])

print(arr[2] , arr[3])

Output :

3 4

# Exp. 9: WAP to access the element on the first row, second column of a 2-D array.

Program:

import numpy as np

arr = np.array([[1,2,3,4,5],

[6,7,8,9,10]])

print('2nd element on 1st row: ', arr[0, 1])

Output :

2nd element on 1st row: 2

# Exp. 10: WAP to access the third element on the second array of the first array of a 3-D array.

Program:

import numpy as np

arr = np.array([[[1, 2, 3],

[4, 5, 6]], [[7, 8, 9],

[10, 11, 12]]])

print(arr[0, 1, 2])

Output : 6

# Exp. 11: WAP to print the last element from the 2nd dim.

Program:

import numpy as np

arr = np.array([[1,2,3,4,5],

[6,7,8,9,10]])

print('Last element from 2nd dim: ', arr[1, -1])

Output :

Last element from 2nd dim: 10

# Exp. 12: WAP to slice elements from index 1 to index 5.

Program:

import numpy as np arr =

np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[1:5])

Output :

[2 3 4 5]

# Exp. 13: WAP to slice elements from index to the end of the array.

Program:

import numpy as np arr =

np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[4:])

Output :

[5 6 7]

# Exp. 14: WAP to slice from the index 3 from the end of index 1 from the end.

Program:

import numpy as np arr =

np.array([1, 2, 3, 4, 5, 6, 7])

print(arr[-3:-1])

Output : [5 6]

# #Slicing 2 – D Arrays:

**Exp. 15: WAP to slice elements from index 1 to index 4 (not included) from the second elemnt.**

Program:

import numpy as np

arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

print(arr[1, 1:4])

Output : [7 8 9]

# # Data types of an Array:

**Exp. 16: WAP to get the data type of an array object.**

Program:

import numpy as np

arr = np.array([1, 2, 3, 4]) print(arr.dtype)

Output :

Int64

# Exp. 17: WAP to create an array with data type string.

Program:

import numpy as np arr =

np.array([1, 2, 3, 4], dtype='S') print(arr)

print(arr.dtype)

Output :

[b’1’ b’2’ b’3’ b’4’]

|S1

# Exp. 18: WAP to change the datatype from float to integer by using int as parameter value.

Program:

import numpy as np

arr = np.array([1.1, 2.1, 3.1]) newarr = arr.astype(int) print(newarr) print(newarr.dtype)

Output : [1 2 3]

Int64

# # Array Shape

**Exp. 19: WAP to print the shape of a 2 – D array.**

Program:

import numpy as np

arr = np.array([1.1, 2.1, 3.1]) newarr = arr.astype(int)

print(newarr) print(newarr.dtype)

Output: (2, 4)

# Exp. 20: WAP to create an array with 5 dimensions using ndmin mean using a vector with values 1,2,3,4 and verify that last dimension has value 4.

Program:

import numpy as np

arr = np.array([1, 2, 3, 4],

ndmin=5) print(arr)

print('shape of array :', arr.shape)

Output :

[[[[[1 2 3 4]]]]]

Shape of array : (1, 1, 1, 1, 4)

# # Array Reshaping

**Exp. 21: WAP to convert 1-D array with 12 elements into a 2-D array.**

Program:

import numpy as np

arr =

np.array([1, 2, 3, 4, 5, 6, 7, 8, 9

, 10, 11, 12])

newarr = arr.reshape(4, 3) print(newarr)

Output :

[[ 1 2 3]

[ 4 5 6]

[ 7 8 9]

[ 10 11 12]]

# Exp. 22: WAP to convert 1-D array with elements into a 3-D array.

Program:

import numpy as np arr =

np.array([1, 2, 3, 4, 5, 6, 7, 8, 9

, 10, 11, 12])

newarr = arr.reshape(2, 3, 2) print(newarr)

Output : [[[ 1 2]

[ 3 4]

[ 5 6]]

[[ 7 8]

[ 9 10]

[11 12]]]

# Exp. 23: WAP to iterate on the elements of 1-D array.

Program:

import numpy as np

arr = np.array([1, 2, 3]) for x in arr:

print(x)

Output :

1

2

3

# Exp. 24: WAP to iterate on the elements of 2-D array.

Program:

import numpy as np

arr = np.array([[1, 2, 3],

[4, 5, 6]])

for x in arr: print(x)

Output : [1 2 3]

[4 5 6]

# Exp. 25: WAP to iterate on the elements of 3-D array.

Program:

import numpy as np

arr = np.array([[[1, 2, 3],

[4, 5, 6]],

[[7, 8, 9], [10, 11, 12]]])

for x in arr: print(x)

Output : [[1 2 3]

[4 5 6]]

[[ 7 8 9]

[10 11 12]]

# # Joining Array

**Exp. 26: WAP to join two arrays.**

Program:

import numpy as np

arr1 = np.array([1, 2, 3])

arr2 = np.array([4, 5, 6])

arr = np.concatenate((arr1, arr2)) print(arr)

Output :

[ 1 2 3 4 5 6]

# Exp. 27: WAP to join two 2-D arrays along rows (axis = 1).

Program:

import numpy as np

arr1 = np.array([[1, 2], [3, 4]])

arr2 = np.array([[5, 6], [7, 8]]) arr = np.concatenate((arr1, arr2), axis=1)

print(arr)

Output : [[ 1 2 5 6]

[ 3 4 7 8]]

# # Splitting Array

**Exp. 28: WAP to split the array in 3 parts.**

Program:

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6]) newarr = np.array\_split(arr, 3) print(newarr)

Output :

[array([1, 2]), array([3, 4]), array([5, 6])]

# Exp. 29: WAP to split the array in 4 parts.

Program:

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6]) newarr = np.array\_split(arr, 4) print(newarr)

Output :

[array([1, 2]), array([3, 4]), array([5]), array([6])]

# Exp. 30: WAP to access the splitted arrays.

Program:

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6]) newarr = np.array\_split(arr, 3) print(newarr[0])

print(newarr[1]) print(newarr[2])

Output : [1 2]

[3 4]

[5 6]

# Exp. 31: WAP to split the 2-D array into three 2-D arrays.

Program:

import numpy as np

arr = np.array([[1, 2], [3, 4],

[5, 6], [7, 8], [9, 10], [11, 12]])

newarr = np.array\_split(arr, 3) print(newarr)

Output :

[array([1, 2],

[3, 4]]), array([[5, 6],

[7, 8]]), array([[ 9, 10],

[11, 12]])]

# # Array Searching

**Exp. 32: WAP to find the indexes where the value is 4.**

Program:

import numpy as np arr =

np.array([1, 2, 3, 4, 5, 4, 4])

x = np.where(arr == 4) print(x)

Output :

(array([3, 5, 6]),)

# Exp. 33: WAP to find the indexes where the values are even.

Program:

import numpy as np arr =

np.array([1, 2, 3, 4, 5, 6, 7, 8])

x = np.where(arr%2 == 0) print(x)

Output :

(array([1, 3, 5, 7]),)

# Exp. 34: WAP to find the indexes where the value 7 should be inserted.

Program:

import numpy as np

arr = np.array([6, 7, 8, 9]) x = np.searchsorted(arr, 7) print(x)

Output :

1

# Exp. 35: WAP to find the indexes where the value 7 should be inserted, starting from the right.

Program:

import numpy as np

arr = np.array([6, 7, 8, 9]) x = np.searchsorted(arr, 7, side='right')

print(x)

Output : 2

# Exp. 36: WAP to find the indexes where the values 2, 4, and 6 should be inserted.

Program:

import numpy as np

arr = np.array([1, 3, 5, 7])

x = np.searchsorted(arr, [2, 4, 6]) print(x)

Output : [1 2 3]

# # Sorting Arrays

**Exp. 37: WAP to sort the array.**

Program:

import numpy as np

arr = np.array([3, 2, 0, 1]) print(np.sort(arr))

Output : [0 1 2 3]

# Exp. 38: WAP to sort the array alphabetically.

Program:

import numpy as np arr =

np.array(['banana', 'cherry', 'appl e'])

print(np.sort(arr))

Output :

[‘apple’ ‘banana’ ‘cherry’]

# # Array Filter

**Exp. 39: WAP to create an array from the elements on index 0 and 2.**

Program:

import numpy as np

arr = np.array([41, 42, 43, 44]) x = [True, False, True, False] newarr = arr[x]

print(newarr)

Output :

[41 43]

# Exp. 40: WAP to create a filter array that will return only values higher than 42.

Program:

import numpy as np

arr = np.array([41, 42, 43, 44]) # Create an empty list filter\_arr = []

# go through each element in arr for element in arr:

# if the element is higher than

42, set the value to True, otherwise False:

if element > 42: filter\_arr.append(True)

else:

filter\_arr.append(False) newarr = arr[filter\_arr] print(filter\_arr) print(newarr)

Output :

[False, False, True, True] [43 44]

# Exp. 41: WAP to create a filter array that will return only even elements from the original array.

Program:

import numpy as np

arr =

np.array([1, 2, 3, 4, 5, 6, 7])

# Create an empty list filter\_arr = []

# go through each element in arr for element in arr:

# if the element is completely divisble by 2, set the value to True, otherwise False

if element % 2 == 0: filter\_arr.append(True)

else:

filter\_arr.append(False)

newarr = arr[filter\_arr]

print(filter\_arr) print(newarr)

Output :

[False, True, False, True, False, True, False] [2 4 6]

# # Matplotlib Pyplot

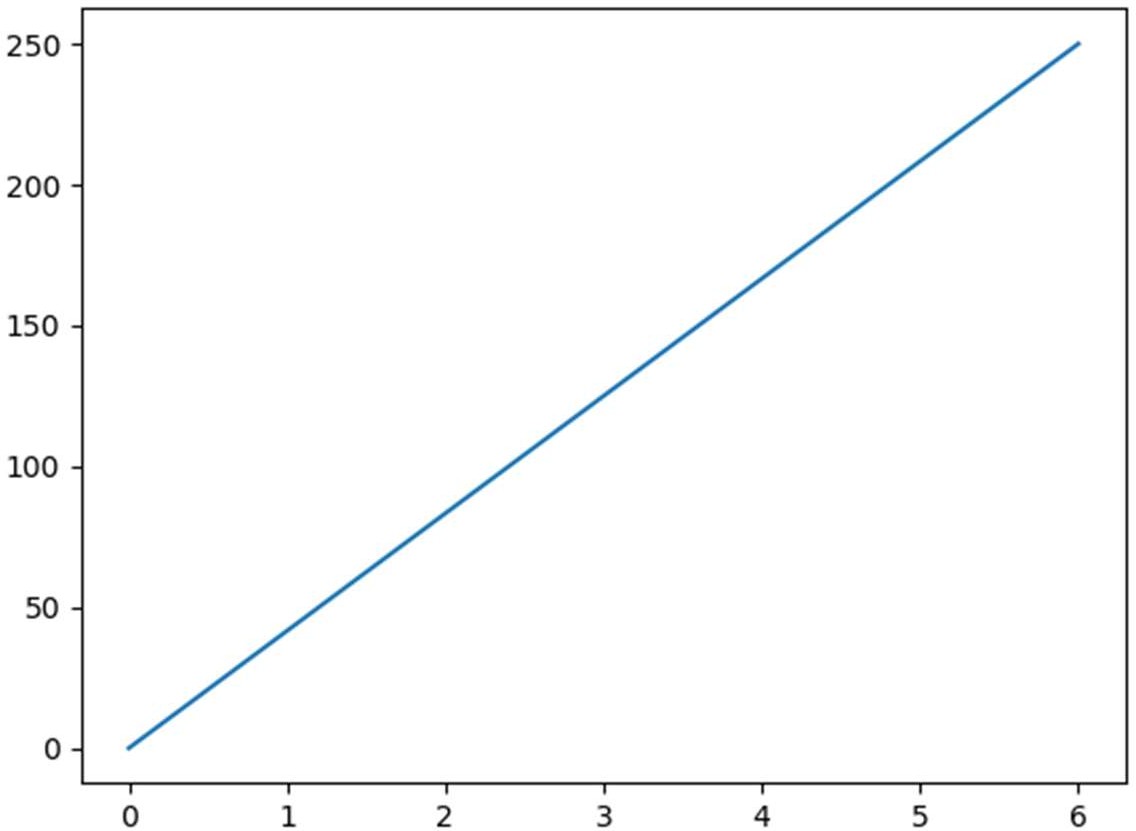
**Exp. 42: WAP to draw a line in a diagram from position (0,0) to position (6,250).**

Program:

import matplotlib.pyplot as plt import numpy as np

xpoints = np.array([0, 6]) ypoints = np.array([0, 250]) plt.plot(xpoints, ypoints) plt.show()

Output :



# # Matplotlib Plotting

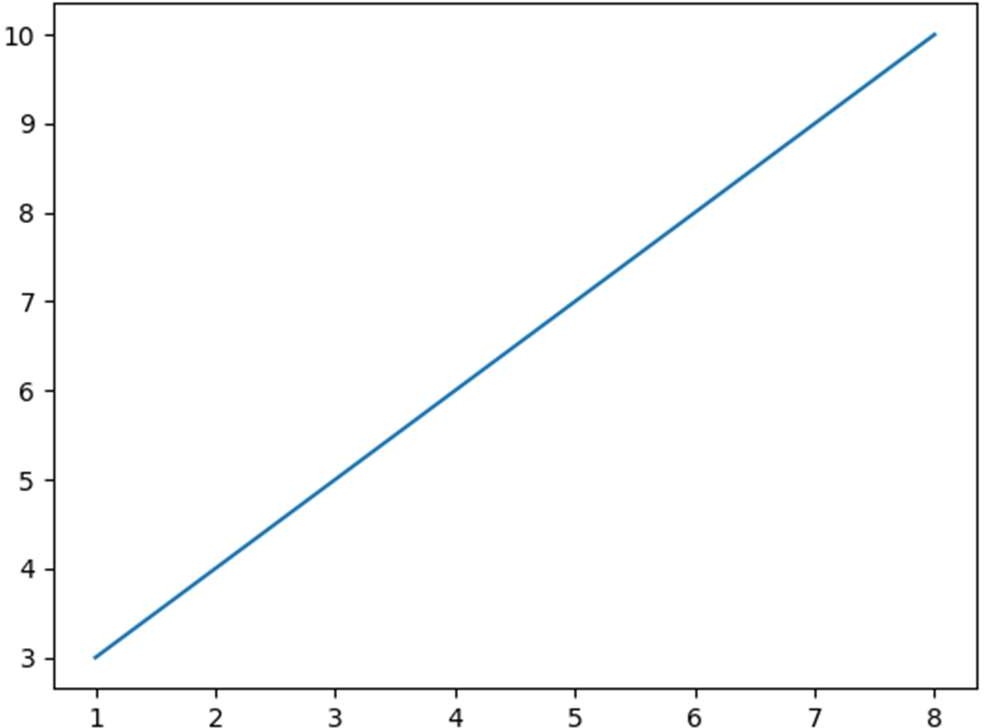
**Exp. 43: WAP to draw a line in a diagram from position (1,3) to position (8,10).**

Program:

import matplotlib.pyplot as plt import numpy as np

xpoints = np.array([1, 8]) ypoints = np.array([3, 10]) plt.plot(xpoints, ypoints) plt.show()

Output :

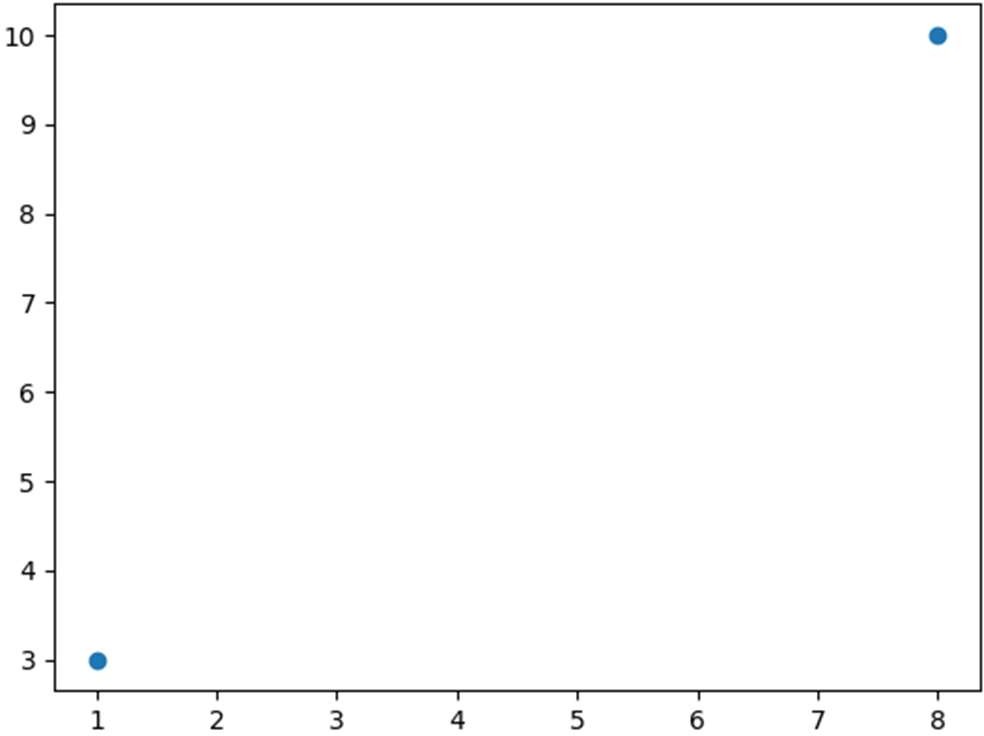


# Exp. 44: WAP to draw two points in the diagram, one at position (1, 3) and one in position (8, 10).

Program:

import matplotlib.pyplot as plt import numpy as np

xpoints = np.array([1, 8]) ypoints = np.array([3, 10]) plt.plot(xpoints, ypoints, 'o') plt.show()

Output :

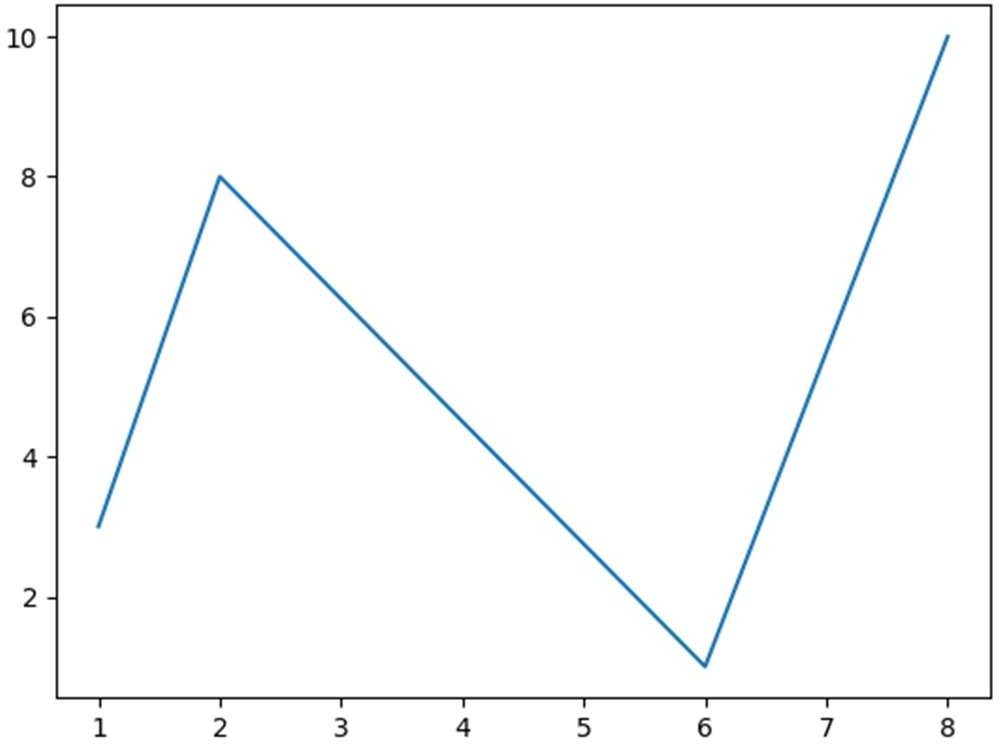
# Exp. 45: WAP to draw a line in a diagram from position (1, 3) to (2, 8) then to (6, 1) and finally to position (8, 10).

Program:

import matplotlib.pyplot as plt import numpy as np

xpoints = np.array([1, 2, 6, 8])

ypoints = np.array([3, 8, 1, 10]) plt.plot(xpoints, ypoints) plt.show()

Output :

# Exp. 46: WAP for plotting without x-points.

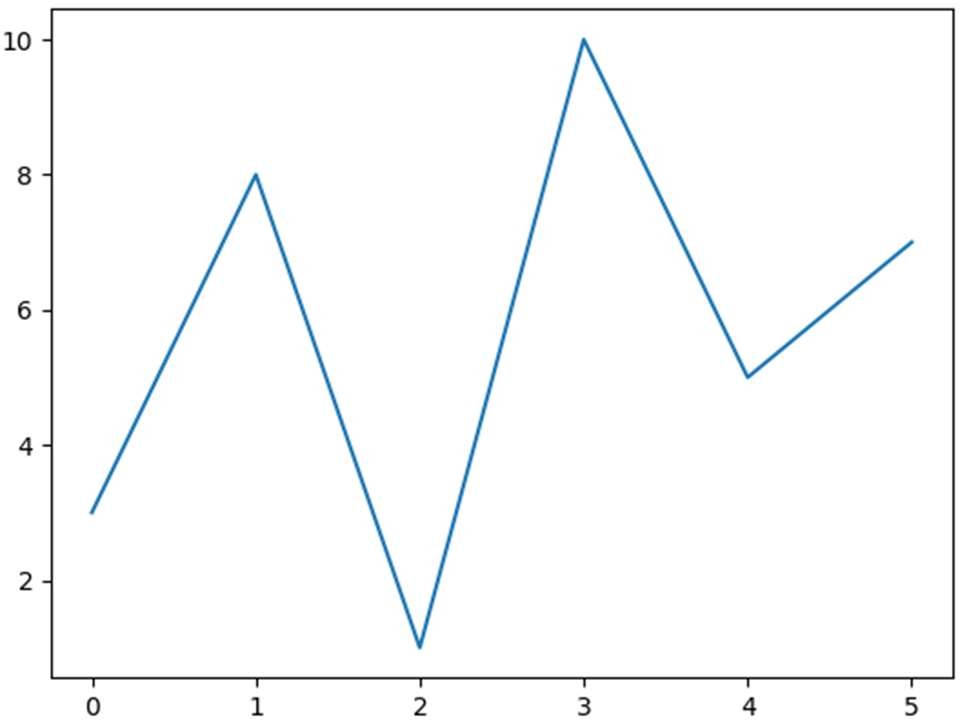
Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10, 5, 7]) plt.plot(ypoints)

plt.show()

Output :



# # Matplotlib Markers

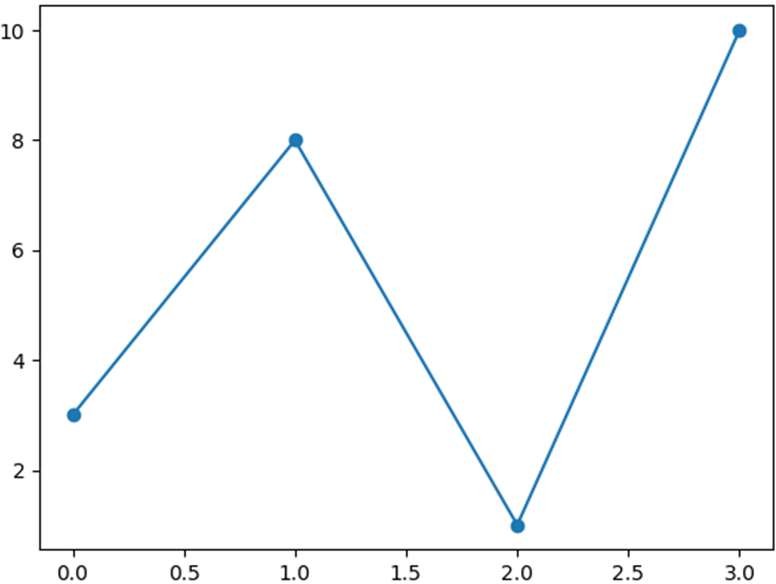
**Exp. 47: WAP to mark each point with a circle.**

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, marker = 'o') plt.show()

Output :



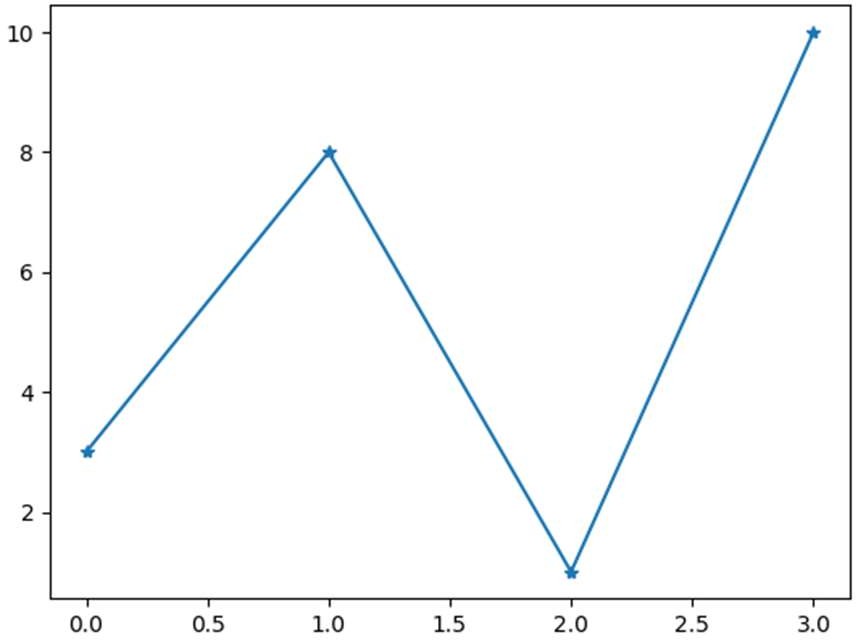
# Exp. 48: WAP to mark each point with a star.

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, marker = '\*') plt.show()

Output :



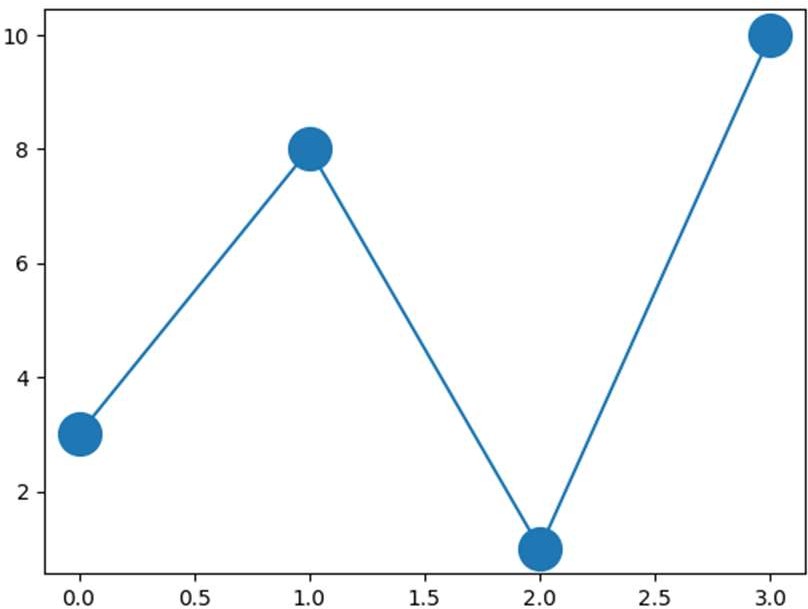
# Exp. 49: WAP to set the size of the markers to 20.

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, marker = 'o', ms = 20) plt.show()

Output :



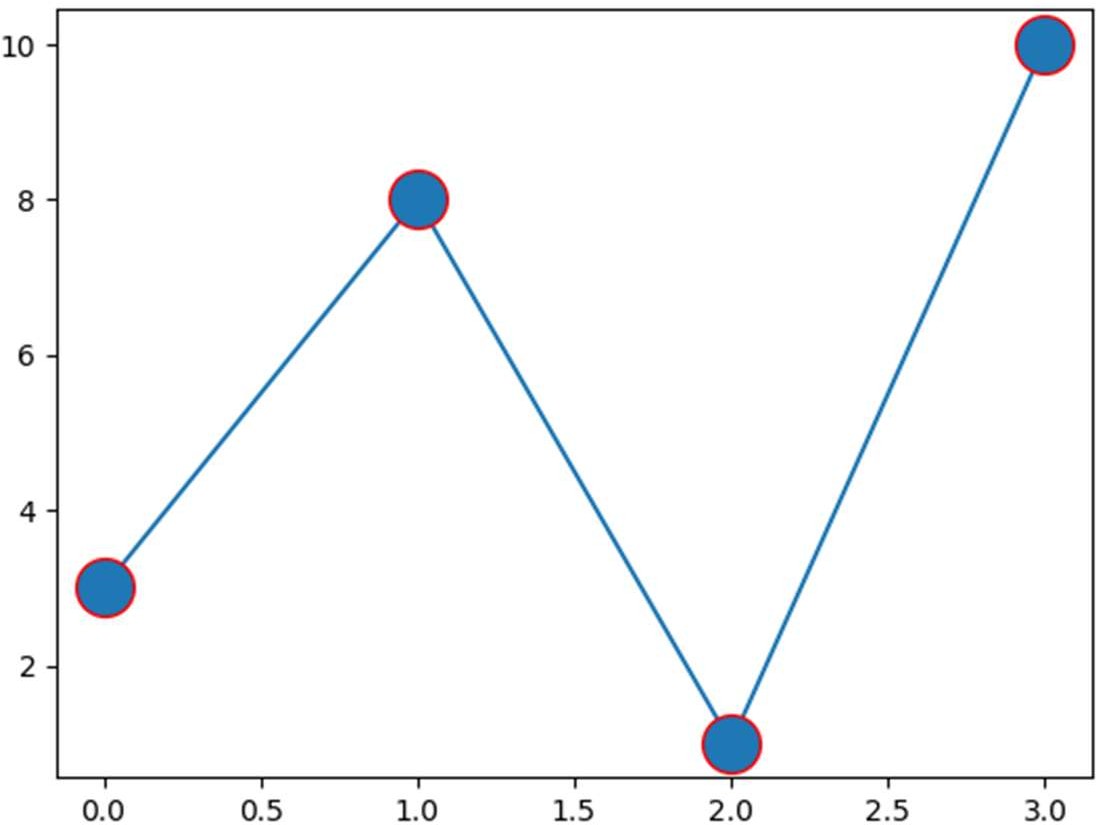
# Exp. 50: WAP to set the edge colour to red.

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, marker = 'o', ms = 20, mec = 'r') plt.show()

Output :



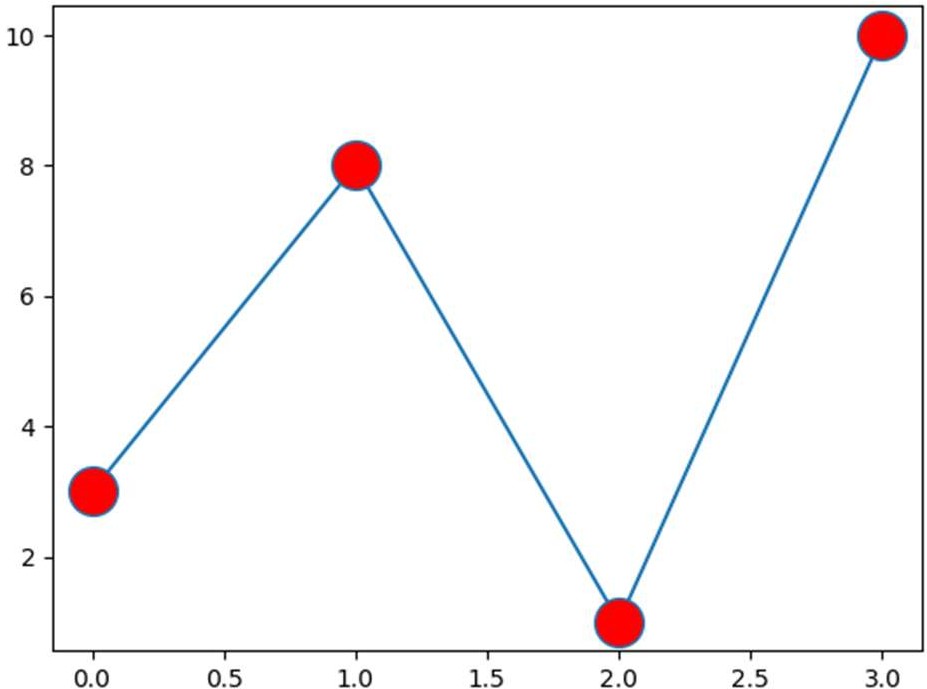
# Exp. 51: WAP to set the FACE colour to red.

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, marker = 'o', ms = 20, mfc = 'r') plt.show()

Output :



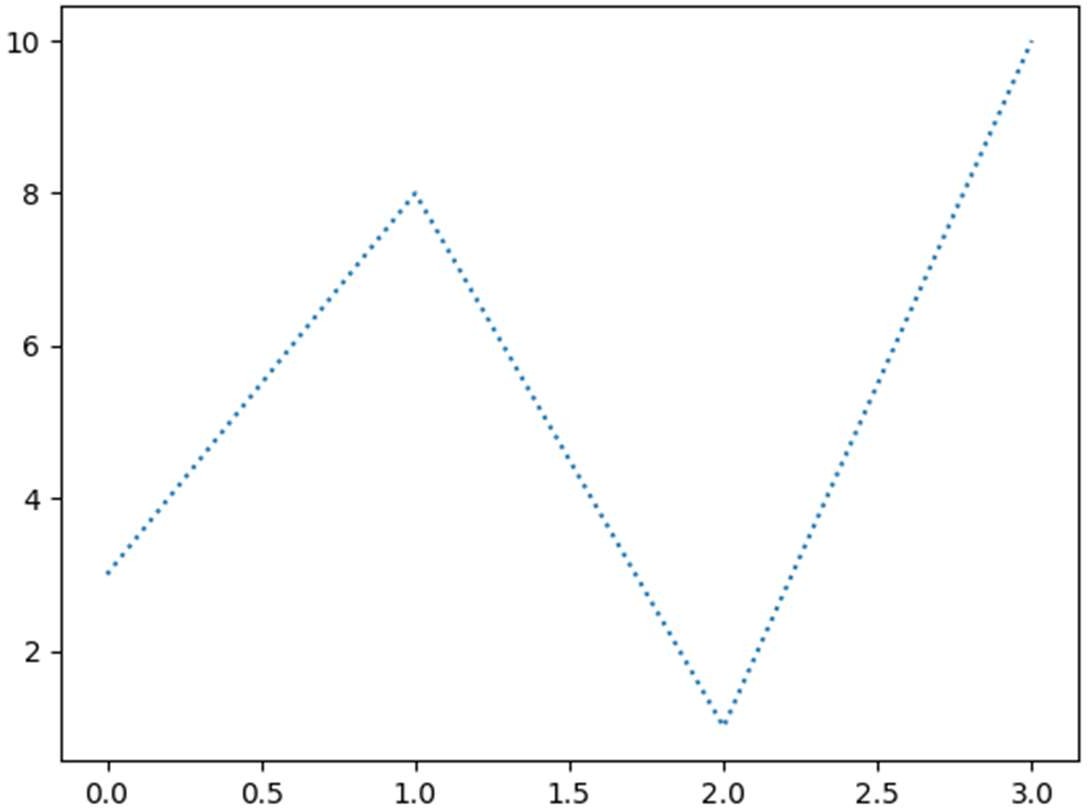
# # Matplotlib Line

**Exp. 52: WAP to use a dotted line.**

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, linestyle = 'dotted') plt.show()

Output :

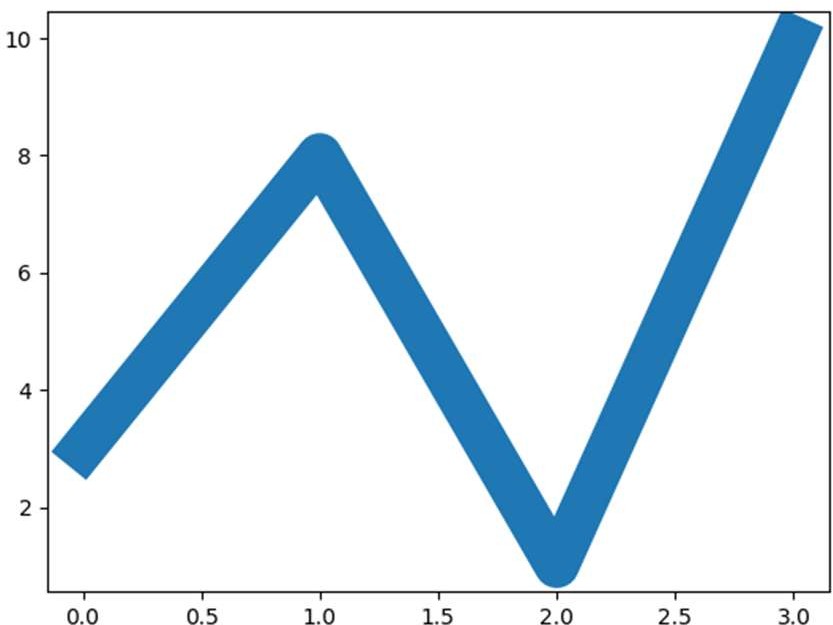
# Exp. 53: WAP to plot with a 20.5pt wide line.

Program:

import matplotlib.pyplot as plt import numpy as np

ypoints = np.array([3, 8, 1, 10]) plt.plot(ypoints, linewidth = '20.5') plt.show()

Output :



# # Multiple Lines

**Exp. 54: WAP to draw two lines by specifying a plt.plot() function for each line.**

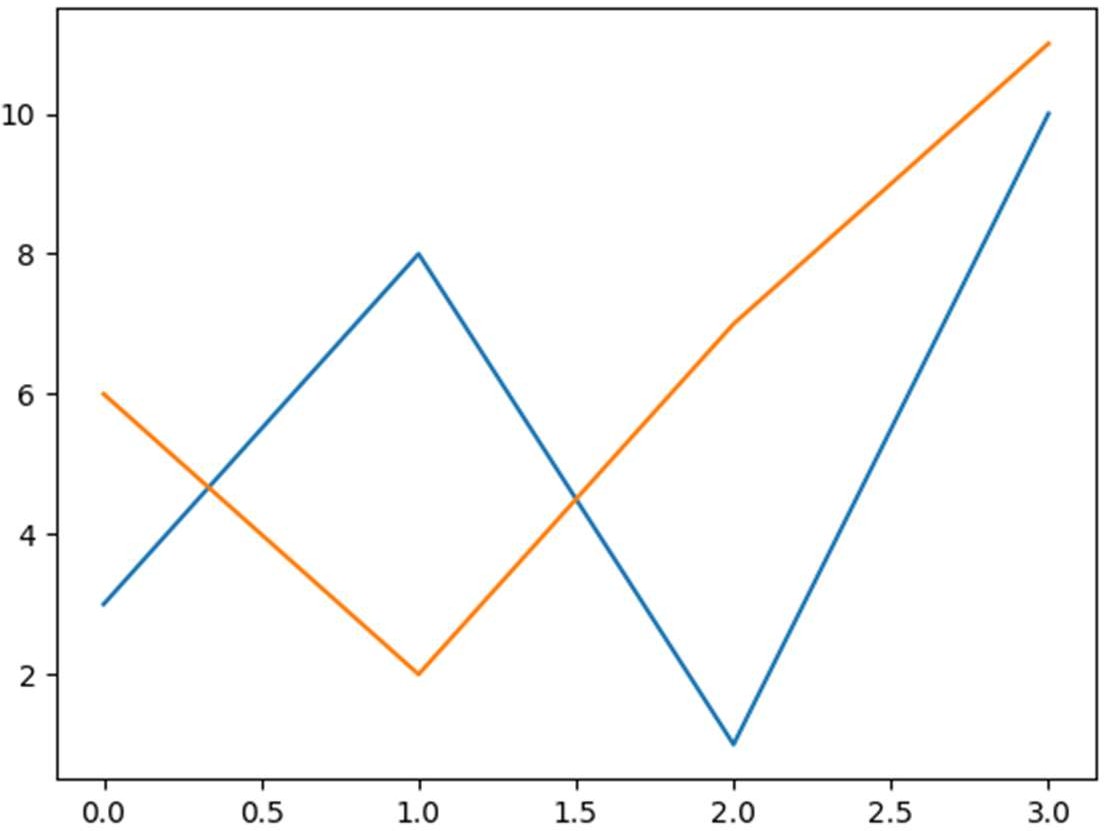
Program:

import matplotlib.pyplot as plt import numpy as np

y1 = np.array([3, 8, 1, 10])

y2 = np.array([6, 2, 7, 11]) plt.plot(y1)

plt.plot(y2) plt.show()

Output :

# # Labels

**Ex. 55: WAP to add labels to the x and y axis.**

Program:

import numpy as np

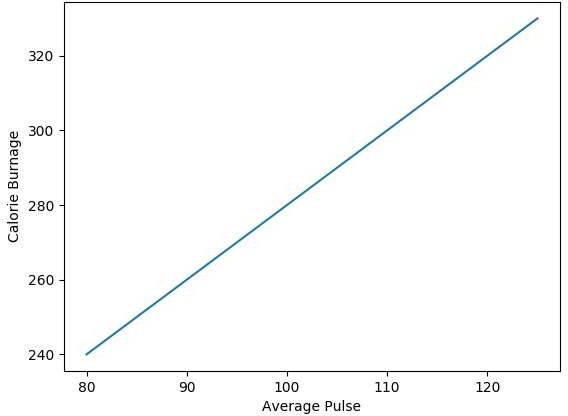
import matplotlib.pyplot as plt

x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.plot(x, y) plt.xlabel("Average Pulse") plt.ylabel("Calorie Burnage") plt.show()

Output :



# Exp. 56: WAP to add a plot title and labels for the x and y axis.

Program:

import numpy as np

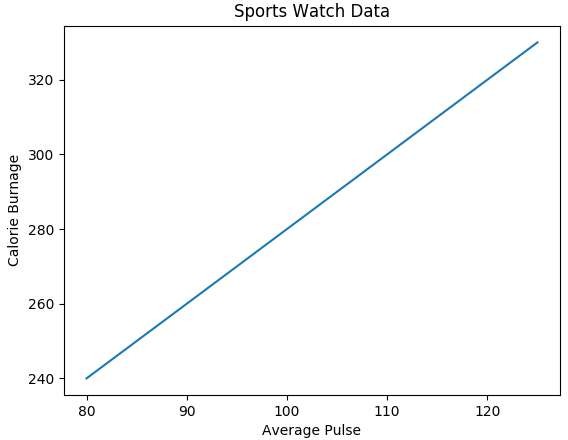
import matplotlib.pyplot as plt

x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.plot(x, y) plt.title("Sports Watch Data") plt.xlabel("Average Pulse") plt.ylabel("Calorie Burnage") plt.show()

Output:



# Exp. 57: WAP to set font properties for the title and labels.

Program:

import numpy as np

import matplotlib.pyplot as plt

x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

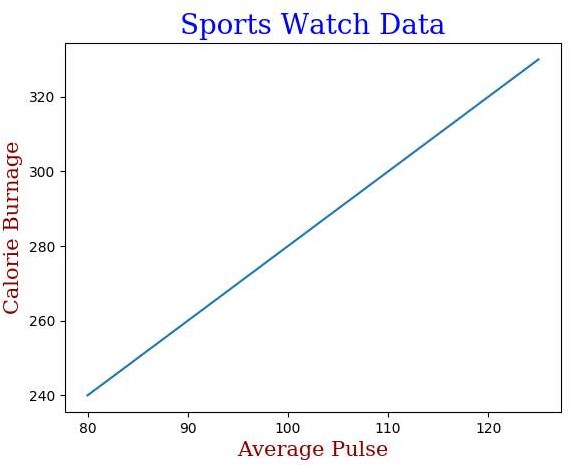
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

font1 = {'family':'serif','color':'blue','size':20}

font2 = {'family':'serif','color':'darkred','size':15} plt.title("Sports Watch Data", fontdict = font1) plt.xlabel("Average Pulse", fontdict = font2) plt.ylabel("Calorie Burnage", fontdict = font2) plt.plot(x, y)

plt.show()

Output:



# Exp. 58: WAP to position the title to the left.

Program:

import numpy as np

import matplotlib.pyplot as plt

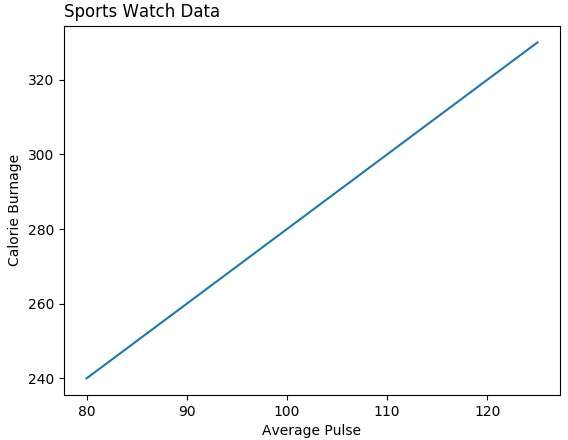
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.title("Sports Watch Data", loc = 'left') plt.xlabel("Average Pulse") plt.ylabel("Calorie Burnage")

plt.plot(x, y) plt.show()

Output:



# # Grid

**Exp. 59: WAP to add grid lines to the plot.**

Program:

import numpy as np

import matplotlib.pyplot as plt

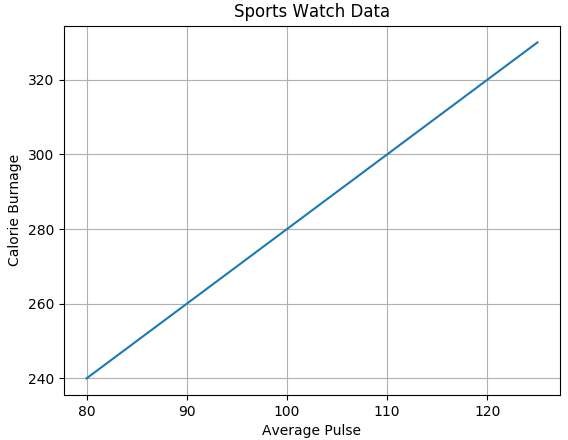
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.title("Sports Watch Data") plt.xlabel("Average Pulse") plt.ylabel("Calorie Burnage") plt.plot(x, y)

plt.grid() plt.show()

Output:



# Exp. 60: WAP to display only grid lines for the x-axis.

Program:

import numpy as np

import matplotlib.pyplot as plt

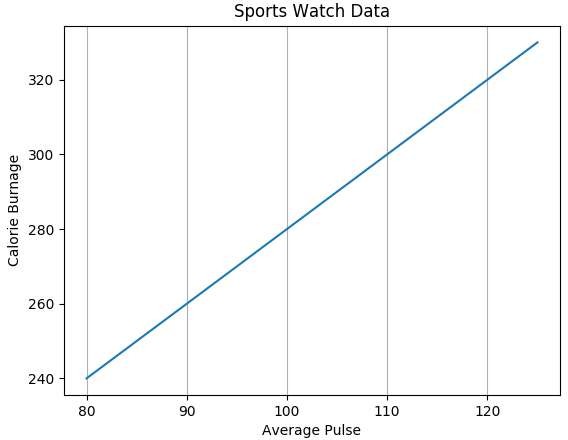
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.title("Sports Watch Data") plt.xlabel("Average Pulse") plt.ylabel("Calorie Burnage") plt.plot(x, y)

plt.grid(axis = 'x') plt.show()

Output:



# Exp. 61: WAP to set the line properties of the grid.

Program:

import numpy as np

import matplotlib.pyplot as plt

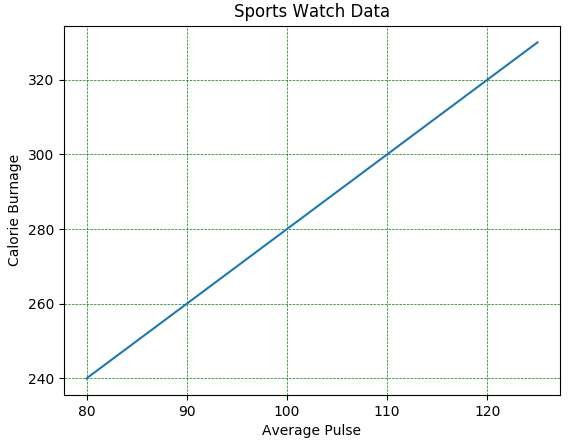
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])

y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])

plt.title("Sports Watch Data") plt.xlabel("Average Pulse") plt.ylabel("Calorie Burnage") plt.plot(x, y)

plt.grid(color = 'green', linestyle = '--', linewidth = 0.5) plt.show()

Output:



# # Subplot

**Exp. 62: WAP to draw 2 plots.**

Program:

import matplotlib.pyplot as plt import numpy as np

#plot 1:

x = np.array([0, 1, 2, 3])

y = np.array([3, 8, 1, 10])

plt.subplot(1, 2, 1) plt.plot(x,y)

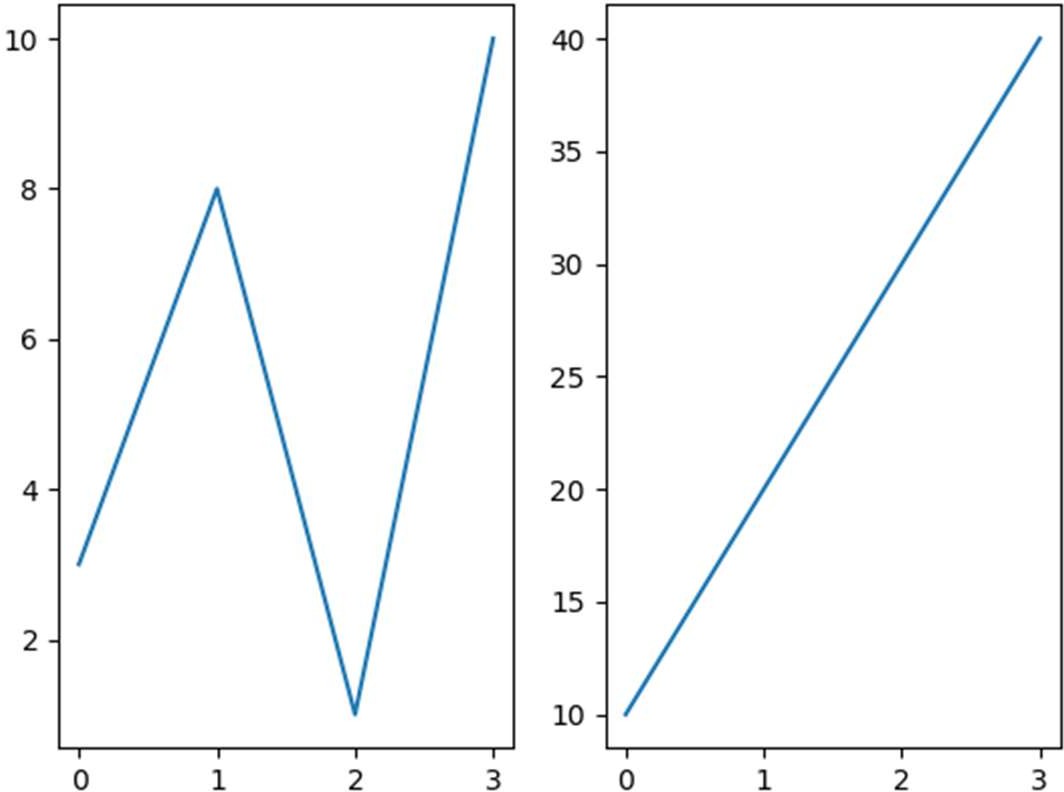
#plot 2:

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(1, 2, 2) plt.plot(x,y) plt.show()

Output:



# Exp. 63: WAP to draw 2 plots on top of each other.

Program:

import matplotlib.pyplot as plt import numpy as np

#plot 1:

x = np.array([0, 1, 2, 3])

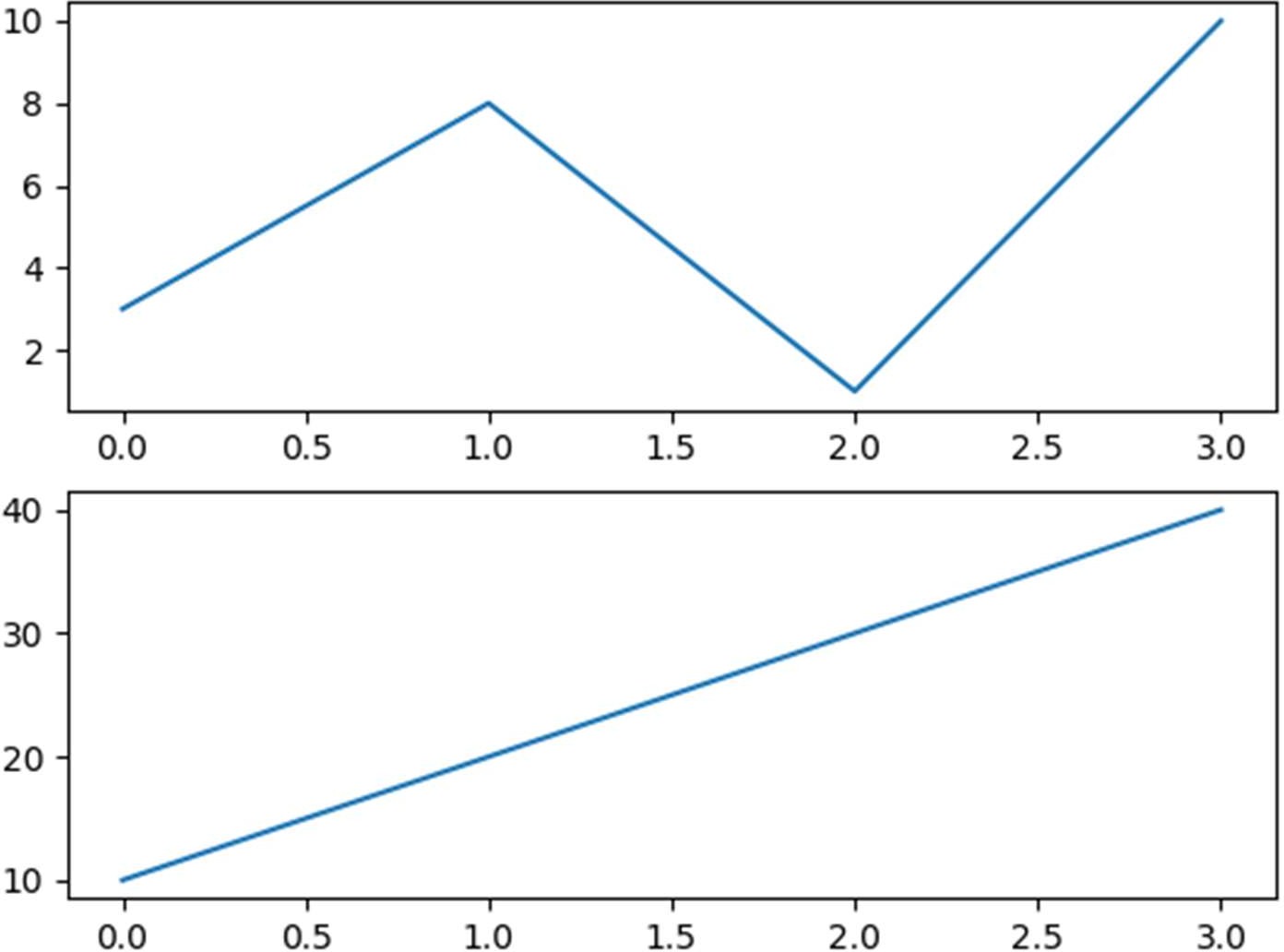
y = np.array([3, 8, 1, 10])

plt.subplot(2, 1, 1) plt.plot(x,y)

#plot 2:

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(2, 1, 2) plt.plot(x,y) plt.show()

Output:

# Exp. 64: WAP to draw 6 plots.

Program:

import matplotlib.pyplot as plt import numpy as np

x = np.array([0, 1, 2, 3])

y = np.array([3, 8, 1, 10])

plt.subplot(2, 3, 1) plt.plot(x,y)

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(2, 3, 2) plt.plot(x,y)

x = np.array([0, 1, 2, 3])

y = np.array([3, 8, 1, 10])

plt.subplot(2, 3, 3) plt.plot(x,y)

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(2, 3, 4) plt.plot(x,y)

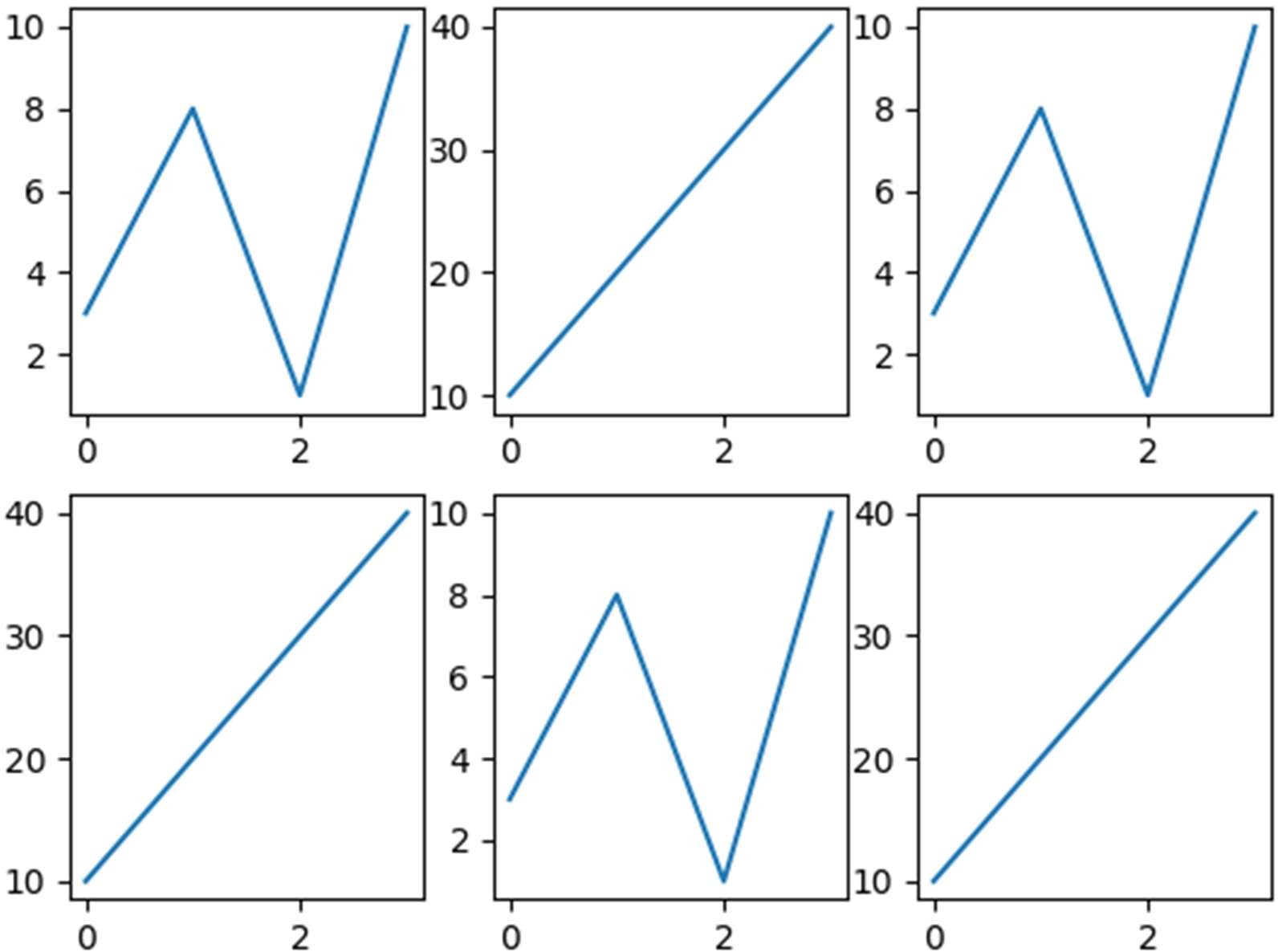
x = np.array([0, 1, 2, 3])

y = np.array([3, 8, 1, 10])

plt.subplot(2, 3, 5) plt.plot(x,y)

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(2, 3, 6) plt.plot(x,y) plt.show()

Output:

# Exp. 65: WAP for 2 plots with titles.

Program:

import matplotlib.pyplot as plt import numpy as np

#plot 1:

x = np.array([0, 1, 2, 3])

y = np.array([3, 8, 1, 10])

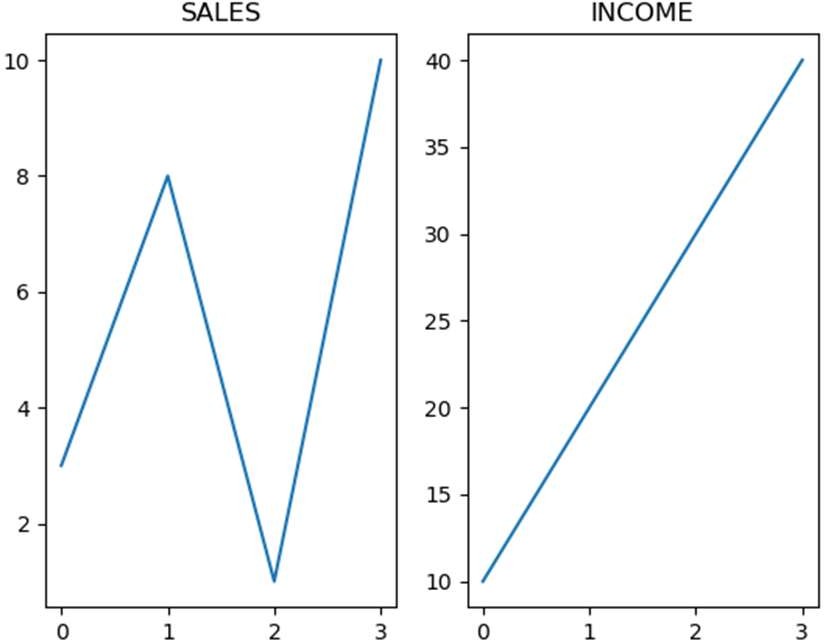
plt.subplot(1, 2, 1) plt.plot(x,y) plt.title("SALES") #plot 2:

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(1, 2, 2) plt.plot(x,y) plt.title("INCOME") plt.show()

Output:



# Exp. 66: WAP to add a title for the entire figure.

Program:

import matplotlib.pyplot as plt import numpy as np

#plot 1:

x = np.array([0, 1, 2, 3])

y = np.array([3, 8, 1, 10])

plt.subplot(1, 2, 1) plt.plot(x,y) plt.title("SALES") #plot 2:

x = np.array([0, 1, 2, 3])

y = np.array([10, 20, 30, 40])

plt.subplot(1, 2, 2)

plt.plot(x,y) plt.title("INCOME") plt.suptitle("MY SHOP") plt.show()

Output:



# # Scatter

**Exp. 67: WAP for a simple scatter plot.**

Program:

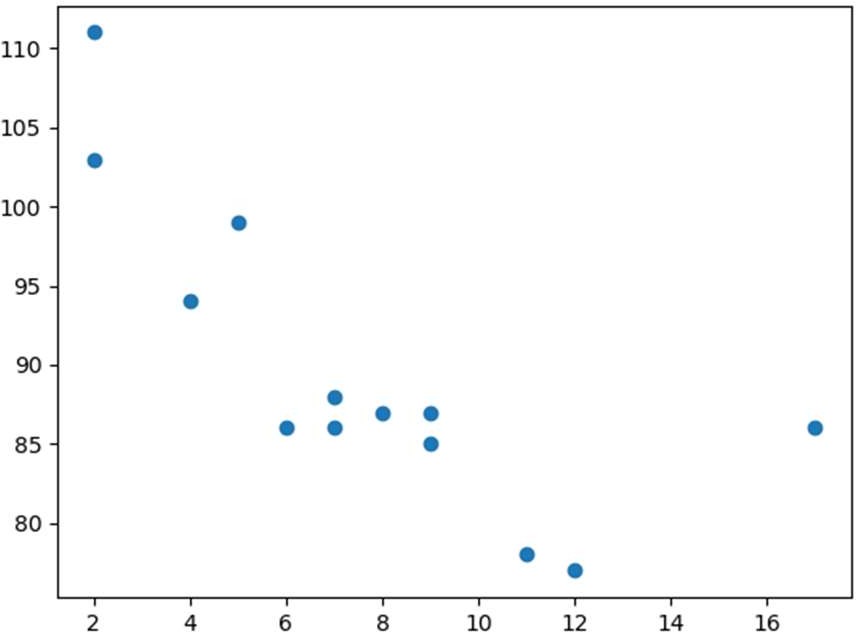
import matplotlib.pyplot as plt import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])

y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

plt.scatter(x, y) plt.show()

Output:



# Exp. 68: WAP to draw two plots on the same figure.

Program:

import matplotlib.pyplot as plt import numpy as np

#day one, the age and speed of 13 cars:

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])

y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

plt.scatter(x, y)

#day two, the age and speed of 15 cars:

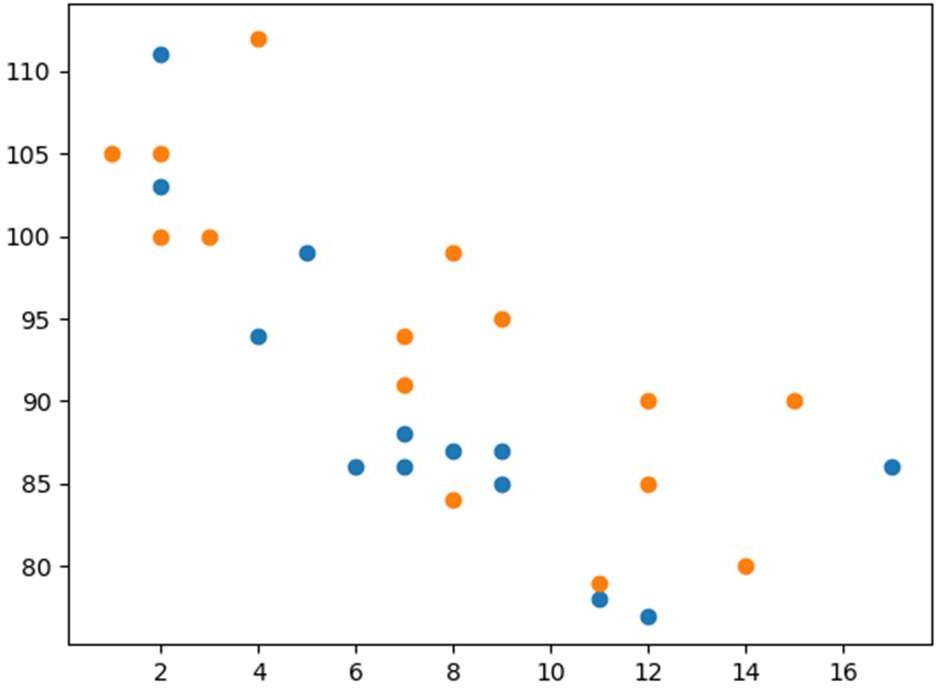
x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])

y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])

plt.scatter(x, y)

plt.show()

Output:



# Exp. 69: WAP to set your own colour of the markers.

Program:

import matplotlib.pyplot as plt import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])

y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

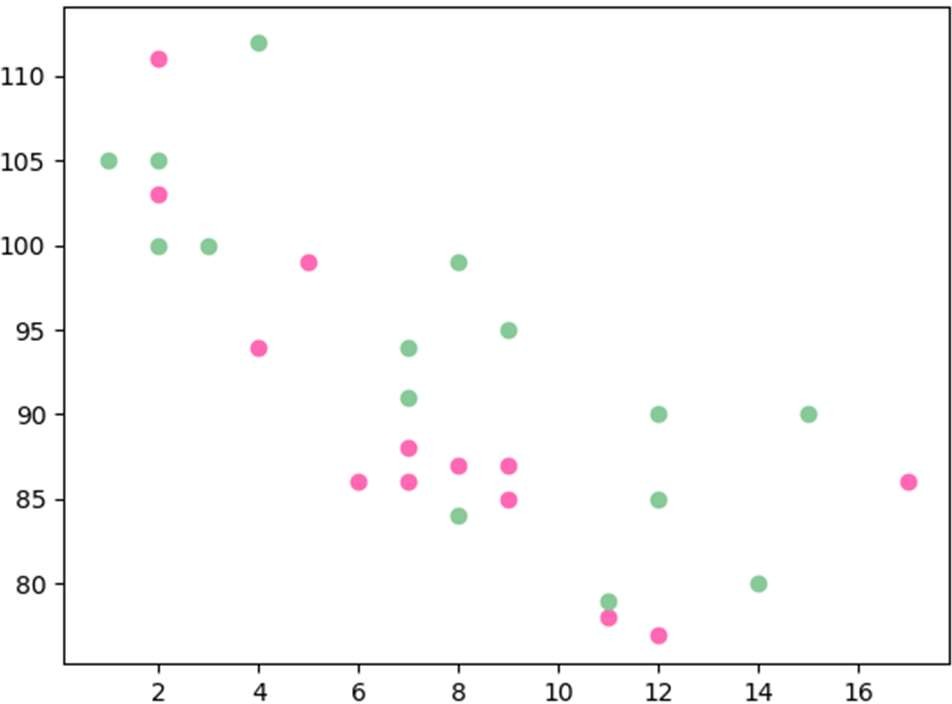
plt.scatter(x, y, color = 'hotpink')

x = np.array([2,2,8,1,15,8,12,9,7,3,11,4,7,14,12])

y = np.array([100,105,84,105,90,99,90,95,94,100,79,112,91,80,85])

plt.scatter(x, y, color = '#88c999') plt.show()

Output:



# Exp. 70: WAP to include the actual colourmap.

Program:

import matplotlib.pyplot as plt import numpy as np

x = np.array([5,7,8,7,2,17,2,9,4,11,12,9,6])

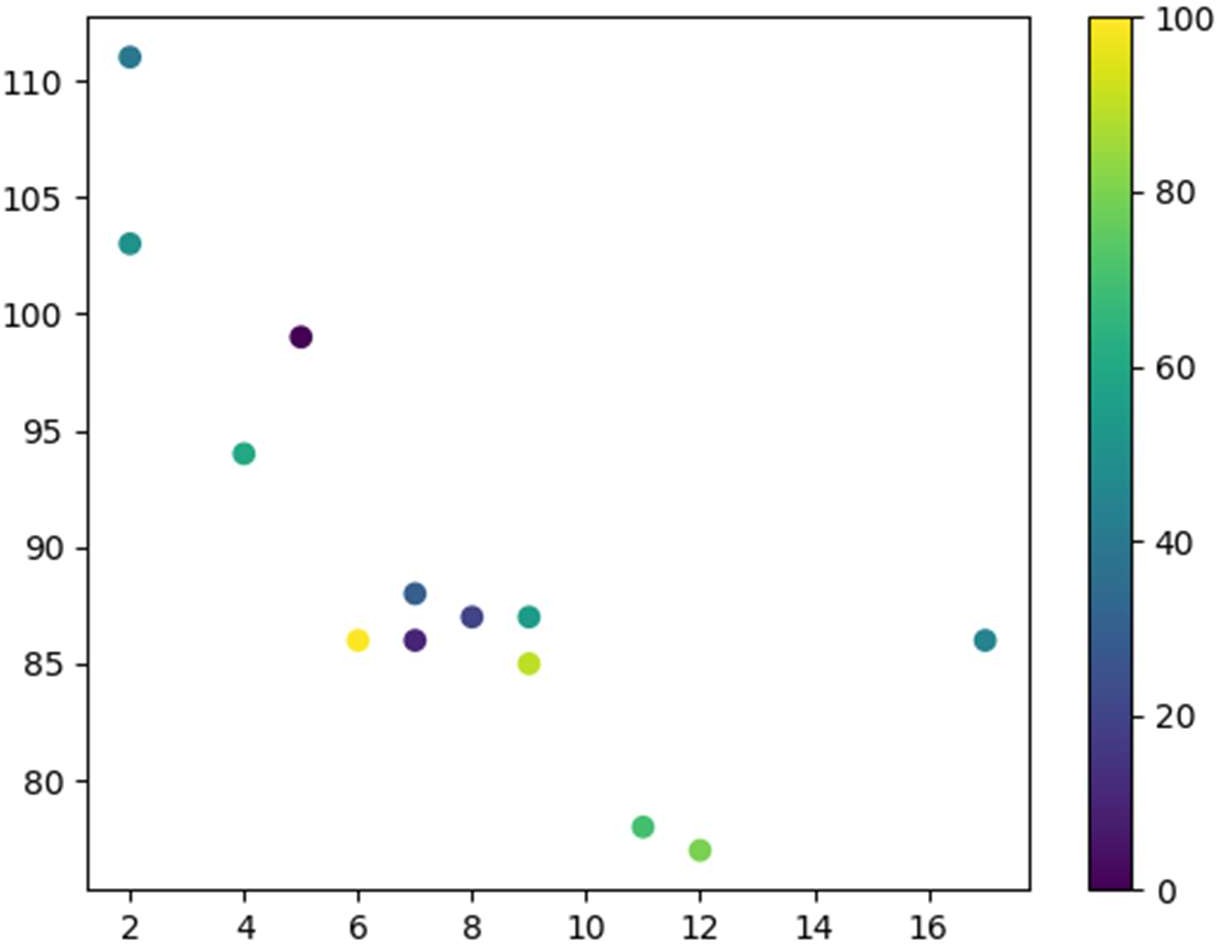
y = np.array([99,86,87,88,111,86,103,87,94,78,77,85,86])

colors = np.array([0, 10, 20, 30, 40, 45, 50, 55, 60, 70, 80, 90, 100])

plt.scatter(x, y, c=colors, cmap='viridis') plt.colorbar()

plt.show()

Output:



# # Bars

**Exp. 71: WAP to draw 4 bars.**

Program:

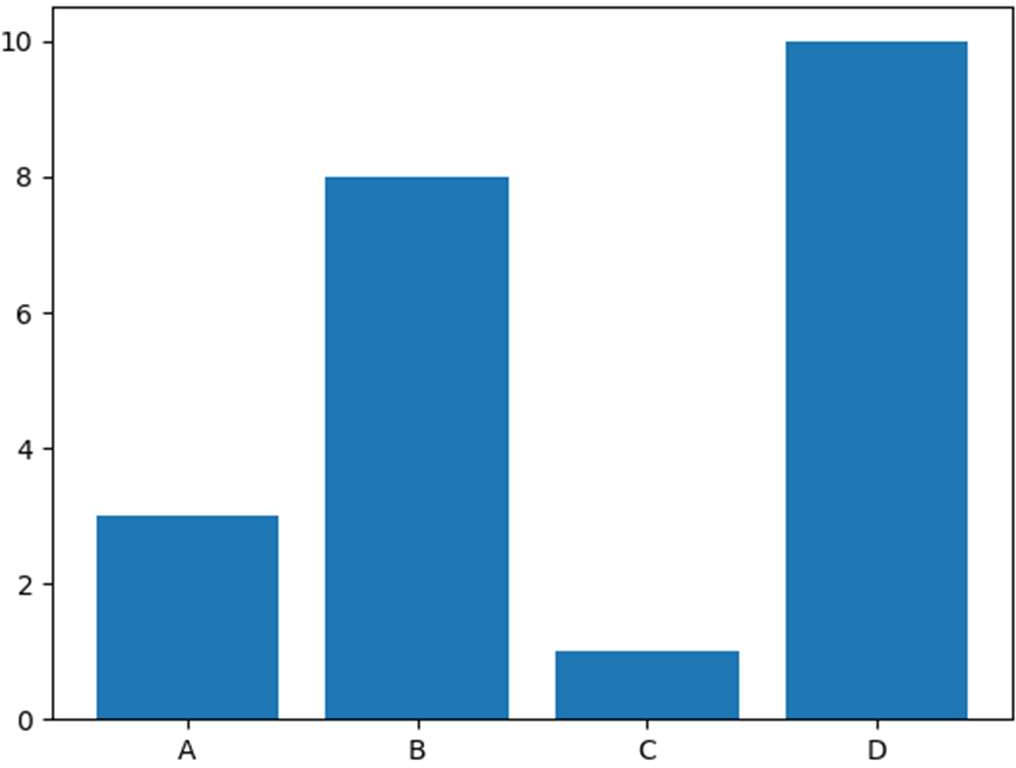
import matplotlib.pyplot as plt import numpy as np

x = np.array(["A", "B", "C", "D"])

y = np.array([3, 8, 1, 10]) plt.bar(x,y)

plt.show()

Output:

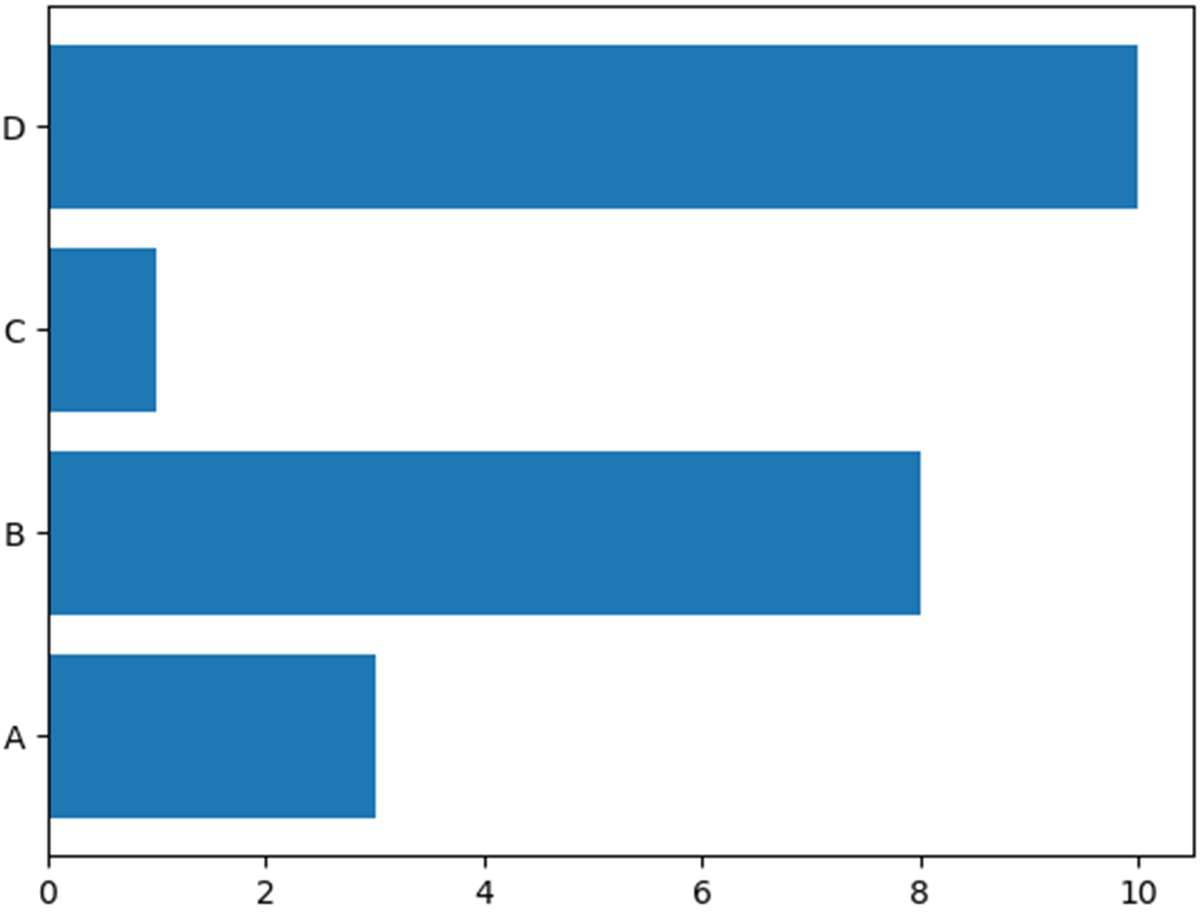


# Exp. 72: WAP to draw 4 horizontal bars.

Program:

import matplotlib.pyplot as plt import numpy as np

x = np.array(["A", "B", "C", "D"])

y = np.array([3, 8, 1, 10]) plt.barh(x, y)

plt.show()

Output:

# Exp. 73: WAP to draw 4 very thin bars.

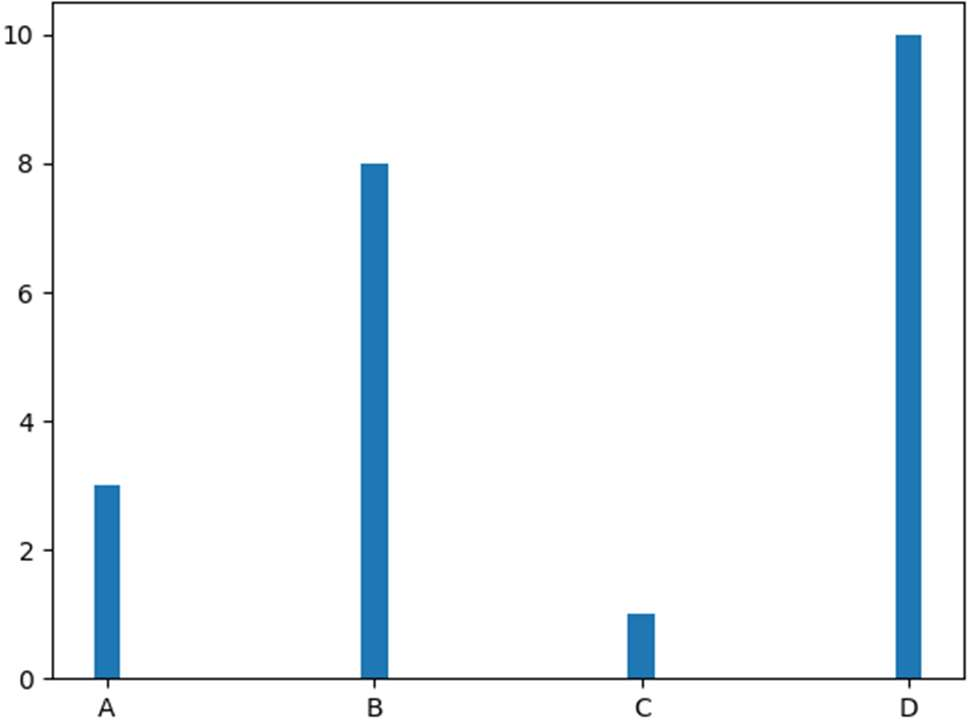
Program:

import matplotlib.pyplot as plt import numpy as np

x = np.array(["A", "B", "C", "D"])

y = np.array([3, 8, 1, 10]) plt.bar(x, y, width = 0.1) plt.show()

Output:

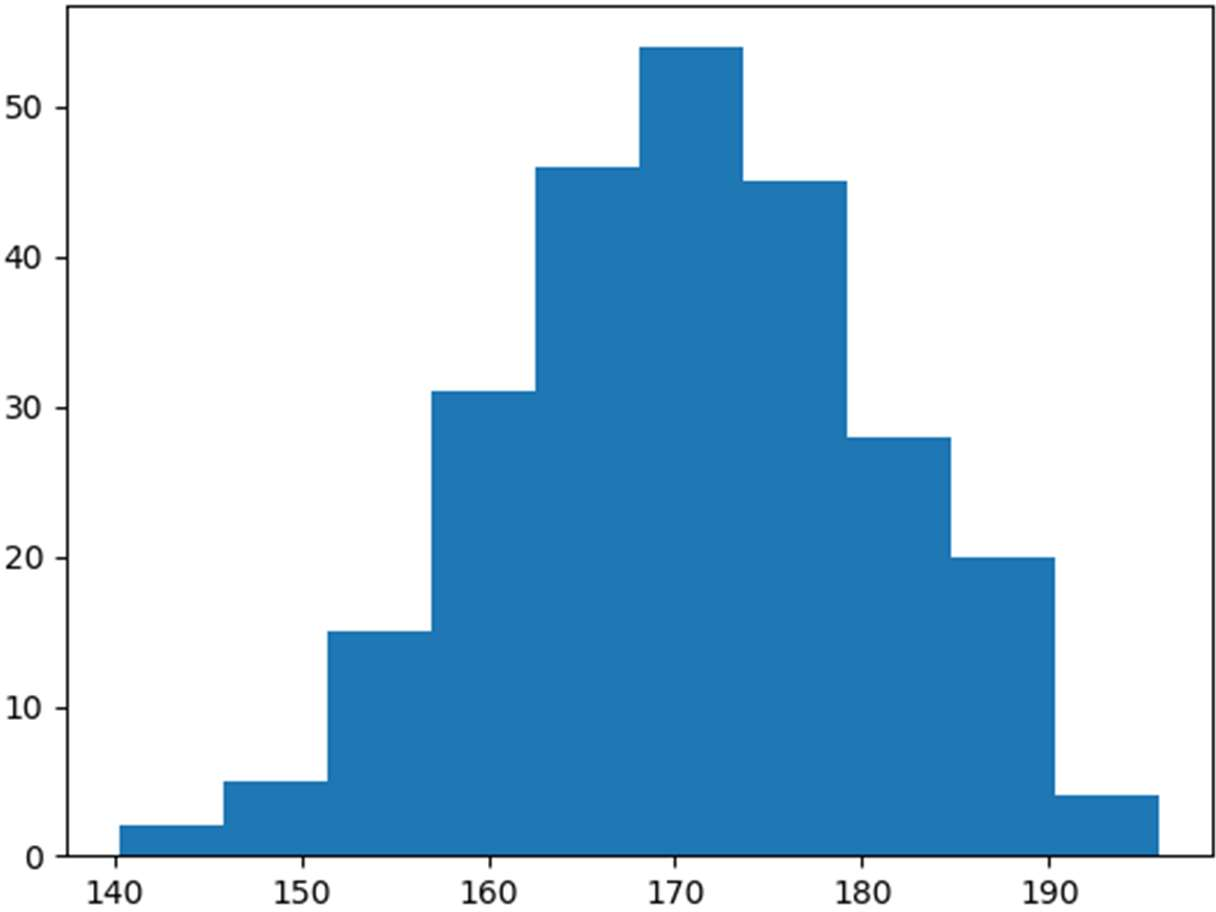


# # Histograms

**Exp. 74: WAP for a simple histogram.**

Program:

import matplotlib.pyplot as plt import numpy as np

x = np.random.normal(170, 10, 250) plt.hist(x)

plt.show()

Output:

# # Pie Charts

**Exp.75: WAP for a simple pie chart.**

Program:

import matplotlib.pyplot as plt import numpy as np

y = np.array([35, 25, 25, 15]) plt.pie(y)

plt.show()

Output:



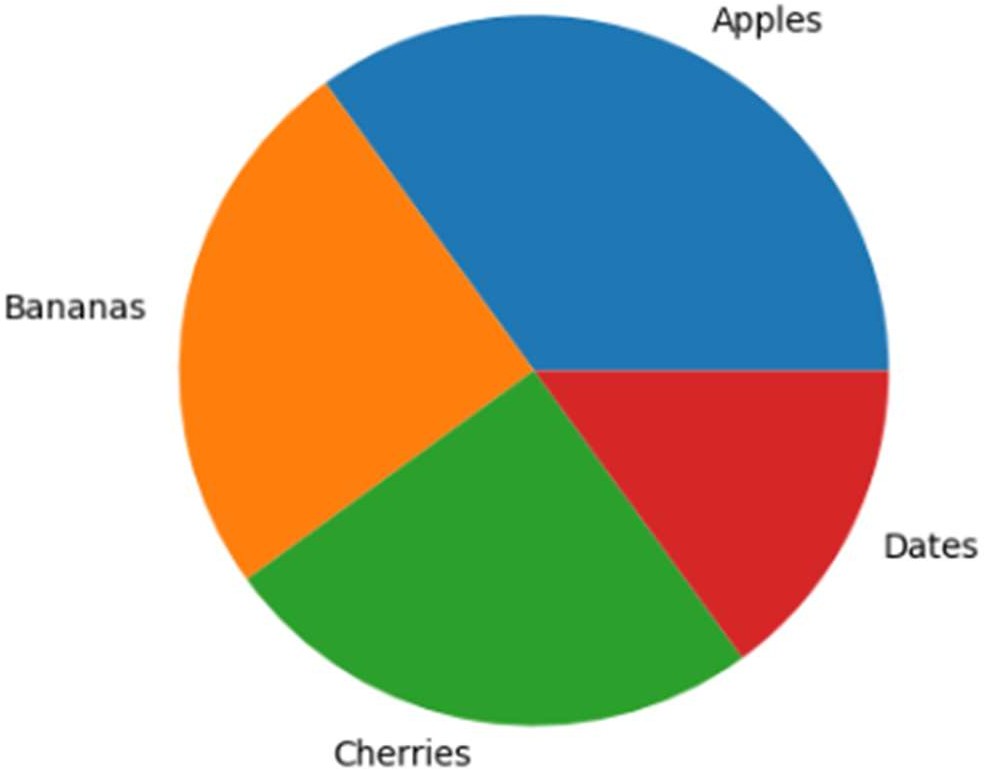
# Exp. 76: WAP to add labels in pie chart.

Program:

import matplotlib.pyplot as plt import numpy as np

y = np.array([35, 25, 25, 15])

mylabels = ["Apples", "Bananas", "Cherries", "Dates"] plt.pie(y, labels = mylabels)

plt.show()

Output:

# Exp. 77: WAP to start the first wedge at 90 degrees.

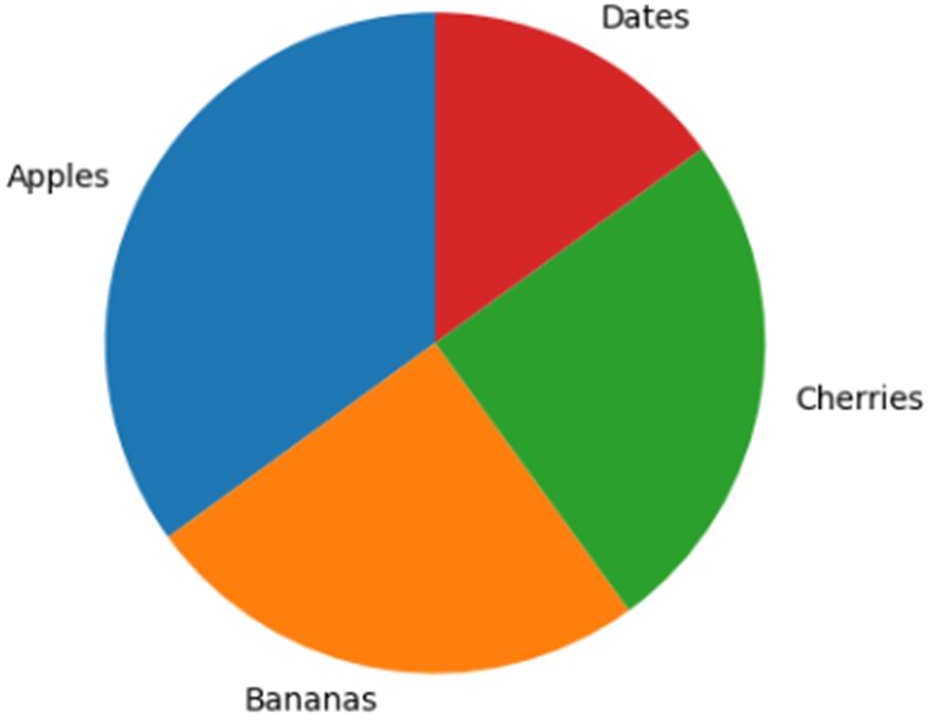
Program:

import matplotlib.pyplot as plt import numpy as np

y = np.array([35, 25, 25, 15])

mylabels = ["Apples", "Bananas", "Cherries", "Dates"] plt.pie(y, labels = mylabels, startangle = 90) plt.show()

Output:



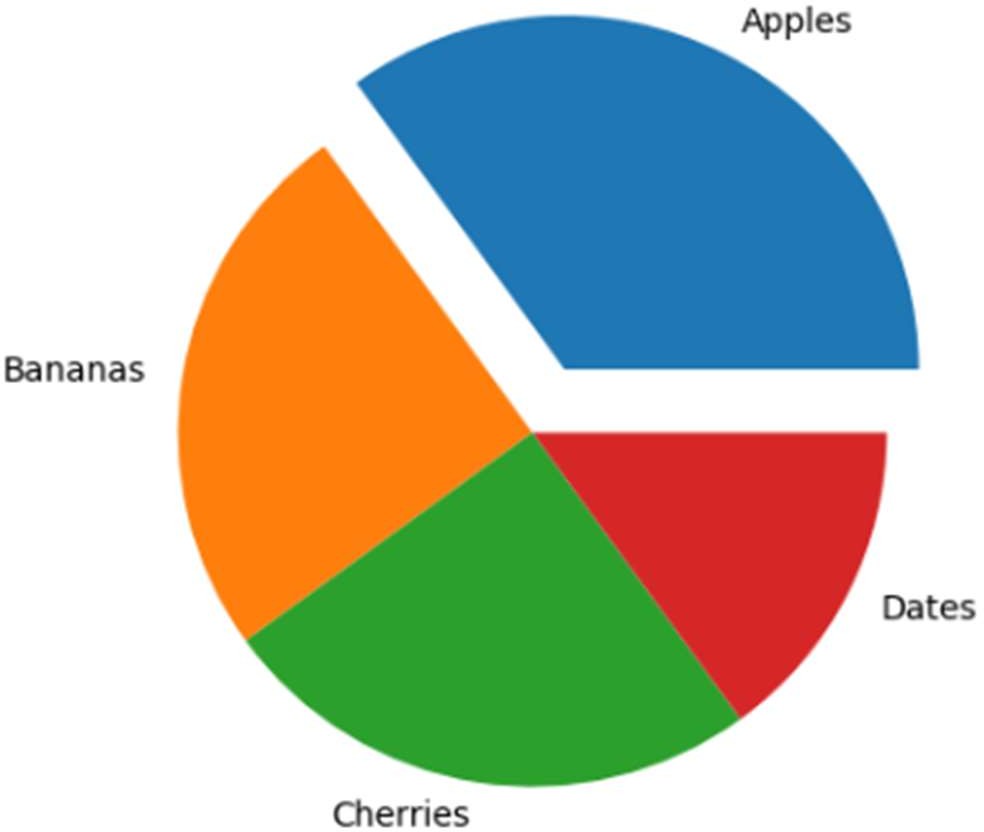
# Exp. 78: WAP to pull the “Apples” wedge 0.2 from the center of the pie.

Program:

import matplotlib.pyplot as plt import numpy as np

y = np.array([35, 25, 25, 15])

mylabels = ["Apples", "Bananas", "Cherries", "Dates"] myexplode = [0.2, 0, 0, 0]

plt.pie(y, labels = mylabels, explode = myexplode) plt.show()

Output:

# Exp. 79: WAP to add a shadow.

Program:

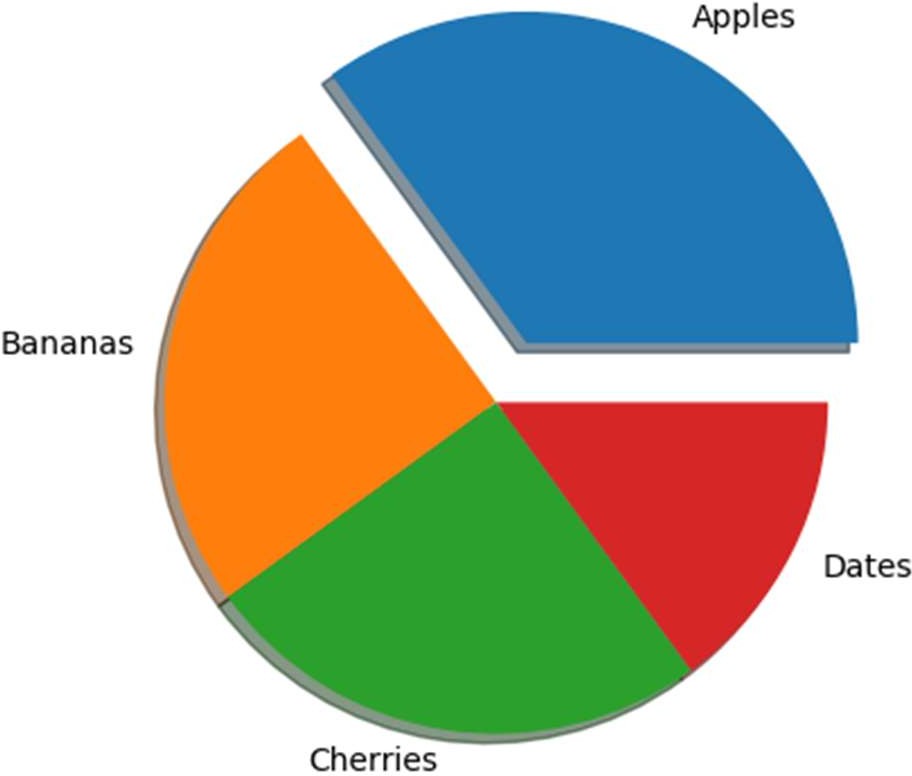
import matplotlib.pyplot as plt import numpy as np

y = np.array([35, 25, 25, 15])

mylabels = ["Apples", "Bananas", "Cherries", "Dates"] myexplode = [0.2, 0, 0, 0]

plt.pie(y, labels = mylabels, explode = myexplode, shadow = True) plt.show()

Output:

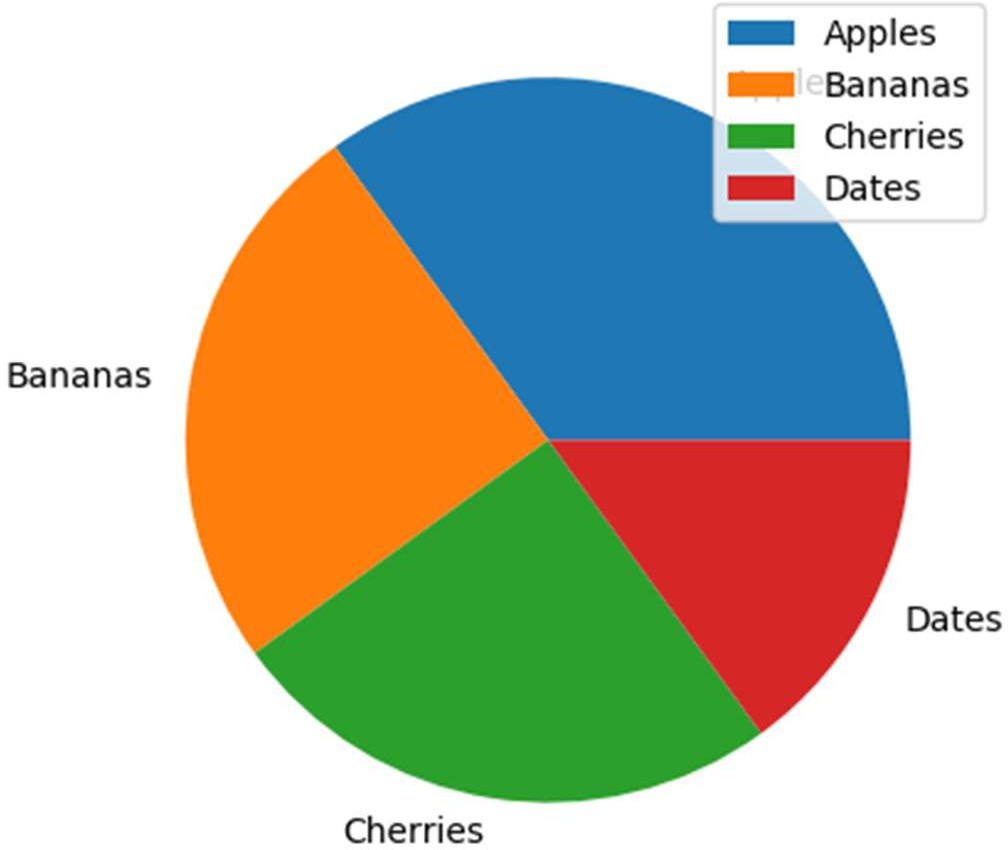


# Exp. 80: WAP to add a legend.

Program:

import matplotlib.pyplot as plt import numpy as np

y = np.array([35, 25, 25, 15])

mylabels = ["Apples", "Bananas", "Cherries", "Dates"] plt.pie(y, labels = mylabels)

plt.legend() plt.show()

Output:

# # Mean, Median and Mode

**Exp. 81: WAP to use the NumPy mean() method to find the average speed.**

Program:

import numpy speed =

[99,86,87,88,111,86,103,87,94,78,77

,85,86]

x = numpy.mean(speed) print(x)

Output: 89.76923076923077

# Exp.82: WAP using NumPy module.

Program:

import numpy speed =

[99,86,87,88,86,103,87,94,78,77,85,

86]

x = numpy.median(speed) print(x)

Output:

86.5

# Exp. 83: WAP to use the SciPy mode( method to find the number that appears the most.

Program:

from scipy import stats speed

= [99,86,87,88,111,86,103,87,94,78,

77,85,86]

x = stats.mode(speed) print(x)

Output:

ModeResult(mode=array([86]), count=array([3]))

# # Standard Deviation

**Exp. 84: WAP using the NumPy std() method to find the standard deviation.**

Program:

import numpy

speed = [86,87,88,86,87,85,86]

x = numpy.std(speed) print(x)

Output: 0.9035079029052513

# Exp. 85: WAP using the NumPy var() method to find the variance.

Program:

import numpy

speed = [32,111,138,28,59,77,97]

x = numpy.var(speed) print(x)

Output: 1432.2448979591834

# # Linear Regression

**Exp. 86: WAP to start by drawing a scatter plot.**

Program:

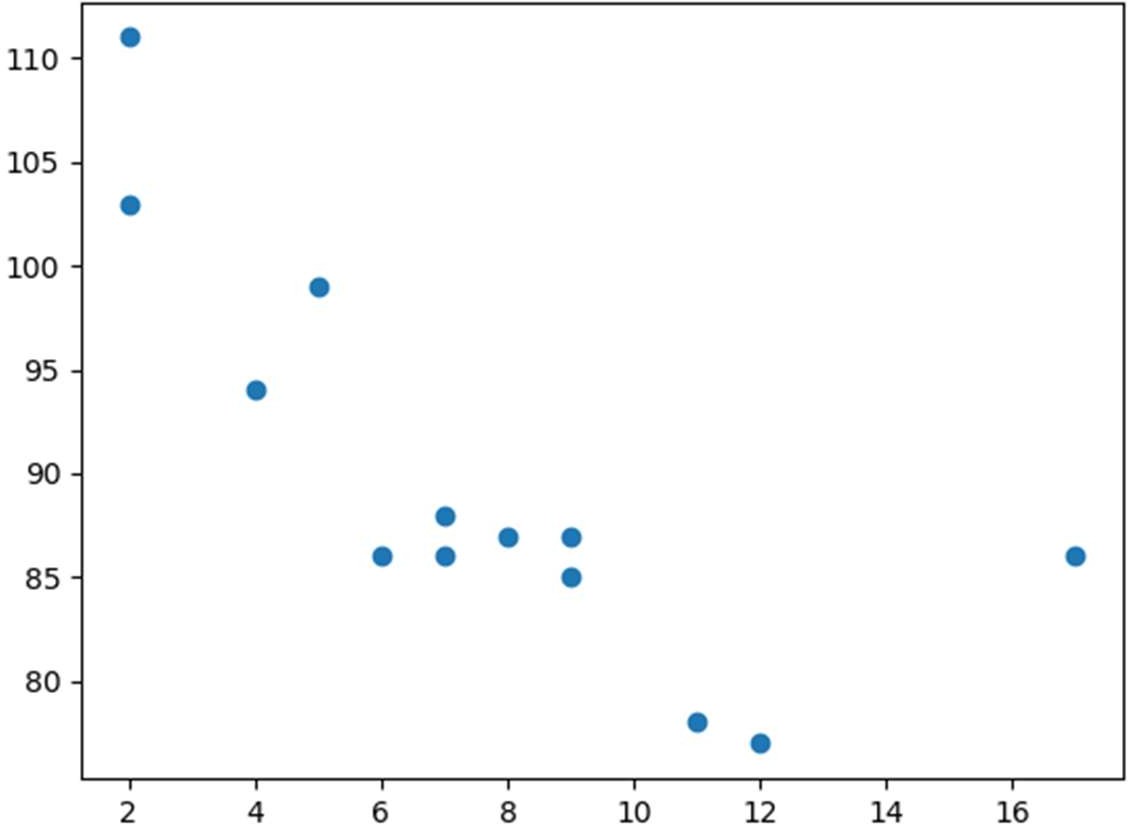
import matplotlib.pyplot as plt

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]

y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

plt.scatter(x, y) plt.show()

Output:



# Exp. 87: WAP to import scipy and draw the line of linear regression.

Program:

import matplotlib.pyplot as plt from scipy import stats

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]

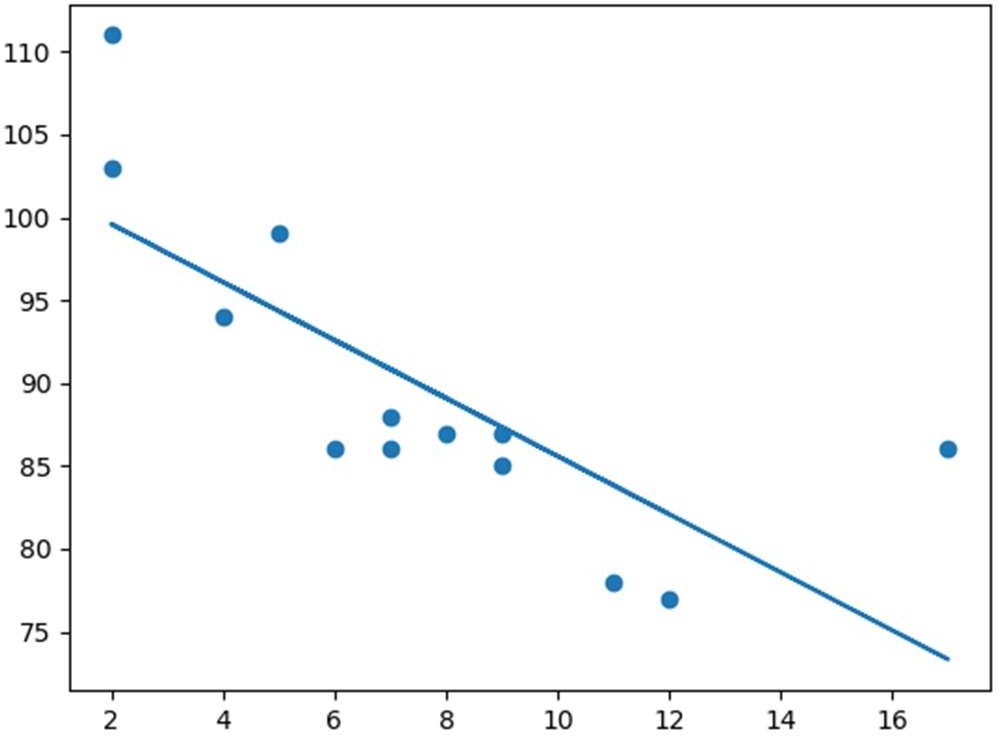
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

slope, intercept, r, p, std\_err = stats.linregress(x, y) def myfunc(x):

return slope \* x + intercept mymodel = list(map(myfunc, x)) plt.scatter(x, y)

plt.plot(x, mymodel) plt.show()

Output:



# Exp. 88: WAP to predict the speed of a 10 years old car.

Program:

from scipy import stats

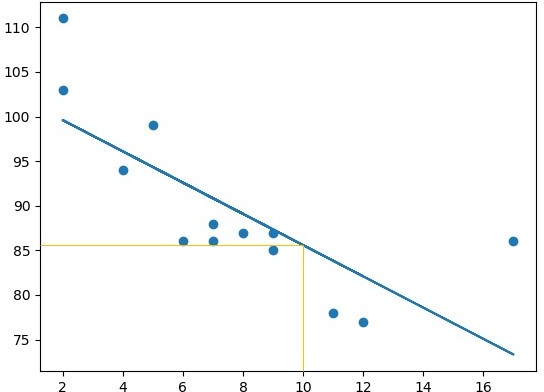
x = [5,7,8,7,2,17,2,9,4,11,12,9,6]

y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

slope, intercept, r, p, std\_err = stats.linregress(x, y) def myfunc(x):

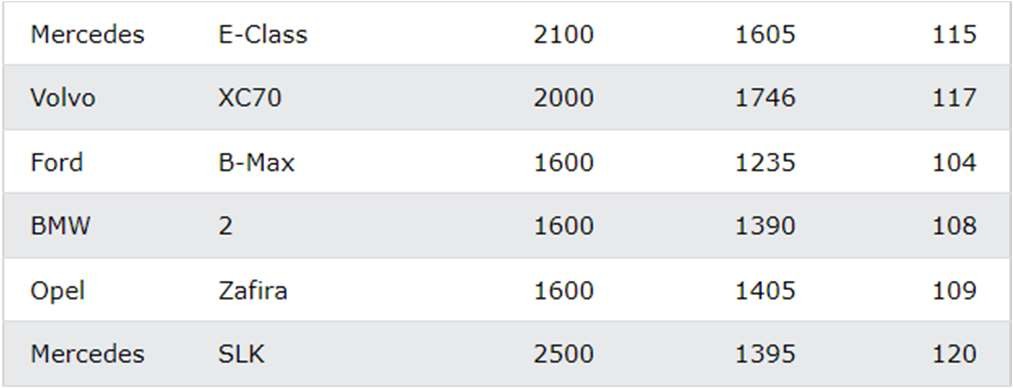
return slope \* x + intercept speed = myfunc(10) print(speed)

Output:



# # Multiple Regression

Dataset:



# Exp. 89: WAP for multiple regression.

Program:

import pandas

from sklearn import linear\_model df = pandas.read\_csv("data.csv") X = df[['Weight', 'Volume']]

y = df['CO2']

regr

= linear\_model.LinearRegression() regr.fit(X, y)

#predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3:

predictedCO2 = regr.predict([[2300, 1300]]) print(predictedCO2)

Output:

107.2087328

# Exp. 90: WAP to print the coefficient values of the regression objects.

Program:

import pandas

from sklearn import linear\_model df = pandas.read\_csv("data.csv")

X = df[['Weight', 'Volume']] y = df['CO2']

regr

= linear\_model.LinearRegression() regr.fit(X, y)

print(regr.coef\_)

Output:

0.00755095 0.00780526

# Exp. 91: WAP to copy the example before, but change the weight from 2300 to 3300.

Program:

import pandas

from sklearn import linear\_model df = pandas.read\_csv("data.csv") X = df[['Weight', 'Volume']]

y = df['CO2']

regr

= linear\_model.LinearRegression() regr.fit(X, y)

predictedCO2 = regr.predict([[3300, 1300]]) print(predictedCO2)

Output: 114.75968007

# # Polynomial Regression

**Exp. 92: WAP to import numpy and matplotlib then draw the line of polynomial regression.**

Program:

import numpy

import matplotlib.pyplot as plt

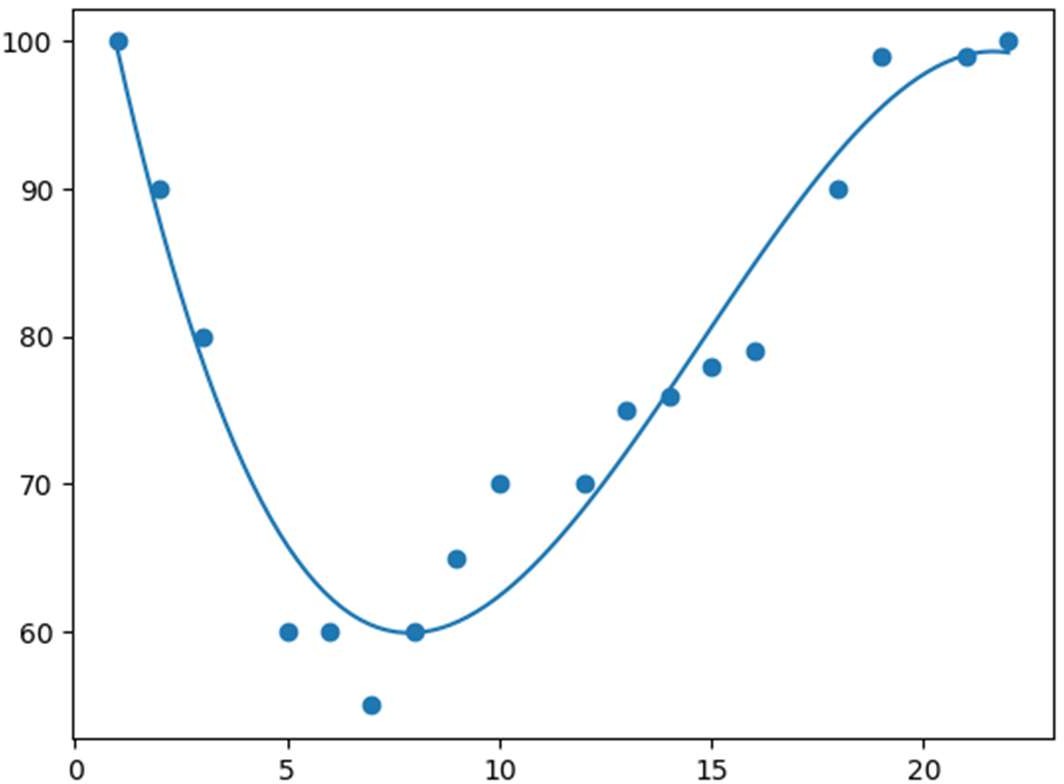
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]

y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]

mymodel = numpy.poly1d(numpy.polyfit(x, y, 3)) myline = numpy.linspace(1, 22, 100) plt.scatter(x, y)

plt.plot(myline, mymodel(myline)) plt.show()

Output:



# Exp. 93: WAP to predict the speed of a car passing at 17.00.

Program:

import numpy

from sklearn.metrics import r2\_score

x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]

y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]

mymodel = numpy.poly1d(numpy.polyfit(x, y, 3)) speed = mymodel(17)

print(speed) Output: 88.87331269697987

# # Pandas

**Exp. 94: WAP for import pandas.**

Program:

import pandas mydataset = {

'cars': ["BMW", "Volvo", "Ford"], 'passings': [3, 7, 2]

}

myvar = pandas.DataFrame(mydataset) print(myvar)

Output:

cars passings

1. BMW 3
2. Volvo 7
3. Ford 2

# Exp. 95: WAP to create a simple Pandas series from a list.

Program:

import pandas as pd a = [1, 7, 2]

myvar = pd.Series(a) print(myvar)

Output:

0 1

1 7

2 2

Dtype: int64

# Exp. 96: WAP to create your own label.

Program:

import pandas as pd a = [1, 7, 2]

myvar = pd.Series(a, index = ["x", "y", "z"])

print(myvar)

Output:

x 1

y 7

z 2

dtype: int64.

# Exp. 97: WAP to create a simple Pandas Series from a dictionary.

Program:

import pandas as pd calories =

{"day1": 420, "day2": 380, "day3":

390}

myvar = pd.Series(calories) print(myvar)

Output:

|  |  |
| --- | --- |
| day1 | 420 |
| day2 | 380 |
| day3 | 390 |

dtype: int64

# Exp. 98: WAP to create a Data Frame from two series.

Program:

import pandas as pd data = {

"calories": [420, 380, 390],

"duration": [50, 40, 45]

}

myvar = pd.DataFrame(data) print(myvar)

Output:

|  |  |  |
| --- | --- | --- |
|  | calories | duration |
| 0 | 420 | 50 |
| 1 | 380 | 40 |
| 2 | 390 | 45 |

# # Data Frame

**Exp. 99: WAP to create a simple Pandas Data Frame.**

Program:

import pandas as pd data = {

"calories": [420, 380, 390],

"duration": [50, 40, 45]

}

#load data into a DataFrame object: df = pd.DataFrame(data)

print(df)

Output:

|  |  |  |
| --- | --- | --- |
|  | calories | duration |
| 0 | 420 | 50 |
| 1 | 380 | 40 |
| 2 | 390 | 45 |

# Exp. 100: WAP to add a list of names to give each row a name.

Program:

import pandas as pd data = {

"calories": [420, 380, 390],

"duration": [50, 40, 45]

}

df = pd.DataFrame(data, index = ["day1", "day2", "day3"])

print(df)

Output:

|  |  |  |
| --- | --- | --- |
|  | calories | duration |
| day0 | 420 | 50 |
| day1 | 380 | 40 |
| day2 | 390 | 45 |

# # JSON

**Exp. 101: WAP to load JOSN file into a Data Frame.**

Program:

import pandas as pd

df = pd.read\_json('data.json') print(df.to\_string())

# Exp. 102: WAP to load a Python Dictionary into a Data Frame.

Program:

import pandas as pd data = {

"Duration":{

"0":60,

"1":60,

"2":60,

"3":45,

"4":45,

"5":60

},

"Pulse":{

"0":110,

"1":117,

"2":103,

"3":109,

"4":117,

"5":102

},

"Maxpulse":{

"0":130,

"1":145,

"2":135,

"3":175,

"4":148,

"5":127

},

"Calories":{

"0":409,

"1":479,

"2":340,

"3":282,

"4":406,

"5":300

}

}

df = pd.DataFrame(data) print(df)

Output:

Duration Pulse Maxpulse Calories 0 60 110 130 409.1

1 60 117 145 479.0

2 60 103 135 340.0

3 45 109 175 282.4

4 45 117 148 406.0

5 60 102 127 300.5

# # CSV

**Exp. 103: WAP to load the CSV into a Data frame.**

Program:

import pandas as pd

df = pd.read\_csv('data.csv') print(df.to\_string())

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Output: |  | | | |
|  | Duration | Pulse | Maxpulse | Calories |
| 0 | 60 | 110 | 130 | 409.1 |
| 1 | 60 | 117 | 145 | 479.0 |
| 2 | 60 | 103 | 135 | 340.0 |
| 3 | 45 | 109 | 175 | 282.4 |
| 4 | 45 | 117 | 148 | 406.0 |
| 5 | 60 | 102 | 127 | 300.5 |

# Exp. 104: WAP to print the Data Frame without the to\_string() method.

Program:

import pandas as pd

df = pd.read\_csv('data.csv') print(df)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Output: |  | | | |
|  | Duration | Pulse | Maxpulse | Calories |
| 0 | 60 | 110 | 130 | 409.1 |
| 1 | 60 | 117 | 145 | 479.0 |
| 2 | 60 | 103 | 135 | 340.0 |
| 3 | 45 | 109 | 175 | 282.4 |
| 4 | 45 | 117 | 148 | 406.0 |
| .. | ... | ... | ... | ... |
| 164 | 60 | 105 | 140 | 290.8 |
| 165 | 60 | 110 | 145 | 300.4 |
| 166 | 60 | 115 | 145 | 310.2 |
| 167 | 75 | 120 | 150 | 320.4 |
| 168 | 75 | 125 | 150 | 330.4 |

# Exp. 105: WAP to increase the maximum number of rows to display the entire Data Frame.

[169 rows x 4 columns]

Program:

import pandas as pd pd.options.display.max\_rows = 9999 df = pd.read\_csv('data.csv') print(df)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Output: |  | | | |
|  | Duration | Pulse | Maxpulse | Calories |
| 0 | 60 | 110 | 130 | 409.1 |
| 1 | 60 | 117 | 145 | 479.0 |
| 2 | 60 | 103 | 135 | 340.0 |
| 3 | 45 | 109 | 175 | 282.4 |
| 4 | 45 | 117 | 148 | 406.0 |
| 5 | 60 | 102 | 127 | 300.5 |

**Data Pre-processing and Data Cleansing**