



**Computing Using Python**

**LAB MANUAL**

AMITY UNIVERSITY CHHATTISGARH

A picture containing text, clipart

Description automatically generated**Amity University Chhattisgarh**

**Amity School of Engineering & Technology Batch : 2020 – 2024**

**Enrolment Number : A80105220044**

This is to certify that this is a bonafide record of the work done by **Mr. Manmohan Balaji**

bearing enrollment number **A80105220044** of B. Tech **Computer Science & Engineering** semester **V** , from Amity School of Engineering & Technology, Amity University Chhattisgarh in the **Computing Using Python** Laboratory with Course Code **CSE3507**.

University Examination held on --.

Faculty in-charge Director-ASET

Examiner- 1 Examiner-2

**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. NO.** | **PARTICULARS** | **PAGE NO.** | **SIGN.** |
| 1. | Introduction to Microsoft s Macro Assembler (MASM). | 4-6 |  |
| 2. | Programs for 16 bit arithmetic operations of 8086(using various Addressing Modes). | 7-9 |  |
| 3. | PROGRAM FOR SORTING AN ARRAY FOR 8086 | 10-12 |  |
| 4. | Program for searching for a number or character in a string for 8086. | 13-15 |  |
| 5. | Program for String Manipulation for 8086. | 16-17 |  |
| 6. | Interfacing ADC&DAC to 8086. | 18-19 |  |
| 7. | Interfacing stepper to 8086. | 20-21 |  |
| 8. | Program Using Arithmetic, Logical, and Bit Manipulation Instructions Of 8051. | 22-23 |  |
| 9. | Timer/Counters in 8051. | 24-25 |  |
| 10. | PROGRAM AND VERIFY INTERRUPT HANDLING IN 8051 | 26-27 |  |
| 11. | PROGRAM FOR UART OPERATION IN 8051. | 28-29 |  |
| 12. | Interfacing LCD to 8051. | 29-31 |  |

1. **Creating An Array**

a)1-Dimensional Array

import numpy as np

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

print(a)

**OUTPUT: [1 2 3 4]**

b) 2-Dimensional Array

import numpy as np

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

print(a)

**OUTPUT: [[1 2 3 4]**

**[5 6 7 8]**

c) 3-Dimensional Array

import numpy as np

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

print(a)

**OUTPUT: [[[1 2]**

**[5 6]**

**[[4 7]**

**[9 10]]]**

1. No. of Dimensions

import numpy as np

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

print(a.ndim)

**OUTPUT: 2**

1. **Array Slicing**

import numpy as np

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

print(arr[1:5])

OUTPUT: [2 3 4 5]

1. **Slicing 2D Arrays**

import numpy as np

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

print(arr[0, 1:4])

OUTPUT: [2 3 4]

1. **Negative Slicing**

import numpy as np

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

print(arr[-3:-1])

OUTPUT: [5 6]

1. **Steps in Array Slicing**

import numpy as np

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

print(arr[1:5:2])

OUTPUT: [2 4 ]

1. **Checking Datatype of Array**

import numpy as np

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

print(arr.dtype)

OUTPUT: int64

1. **Creating Arrays with Defined Datatypes**

import numpy as np

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

print(arr)

print(arr.dtype)

OUTPUT: [“1” “2” “3” “4”]

S1

1. **Array Shaping**

import numpy as np

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

print(arr.shape)

OUTPUT: [2 4]

**b**) 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)

1. **Array Reshaping**

import numpy as np

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

newarr = arr.reshape(4,2)

print(newarr)

OUTPUT: [[1 2]

[3 4]

[[5 6]

[7 8]

**b) Reshaping from 1D to 3D**

import numpy as np

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

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

print(newarr)

OUTPUT: [[[1 2]

[3 4]]

[[5 6]

[7 8]]]

1. **Flattening the Array**

import numpy as np

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

newarr = arr.reshape(-1)

print(newarr)

OUTPUT: [1 2 3 4 5 6]

1. **Array Join**

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]

**b) Array Join Using Stack**

import numpy as np

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

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

arr = np.stack((arr1, arr2), axis=1)

print(arr)

OUTPUT: [[[1 2]

[3 4]

[5 6]]]

**c)Stacking along Height(Depth)**

import numpy as np

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

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

arr = np.dstack((arr1, arr2))

print(arr)

OUTPUT: [[[1 4]

[2 5]

[3 6]]]

1. **Array Splitting**

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])]

**b) Splitting 2D Arrays**

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]])

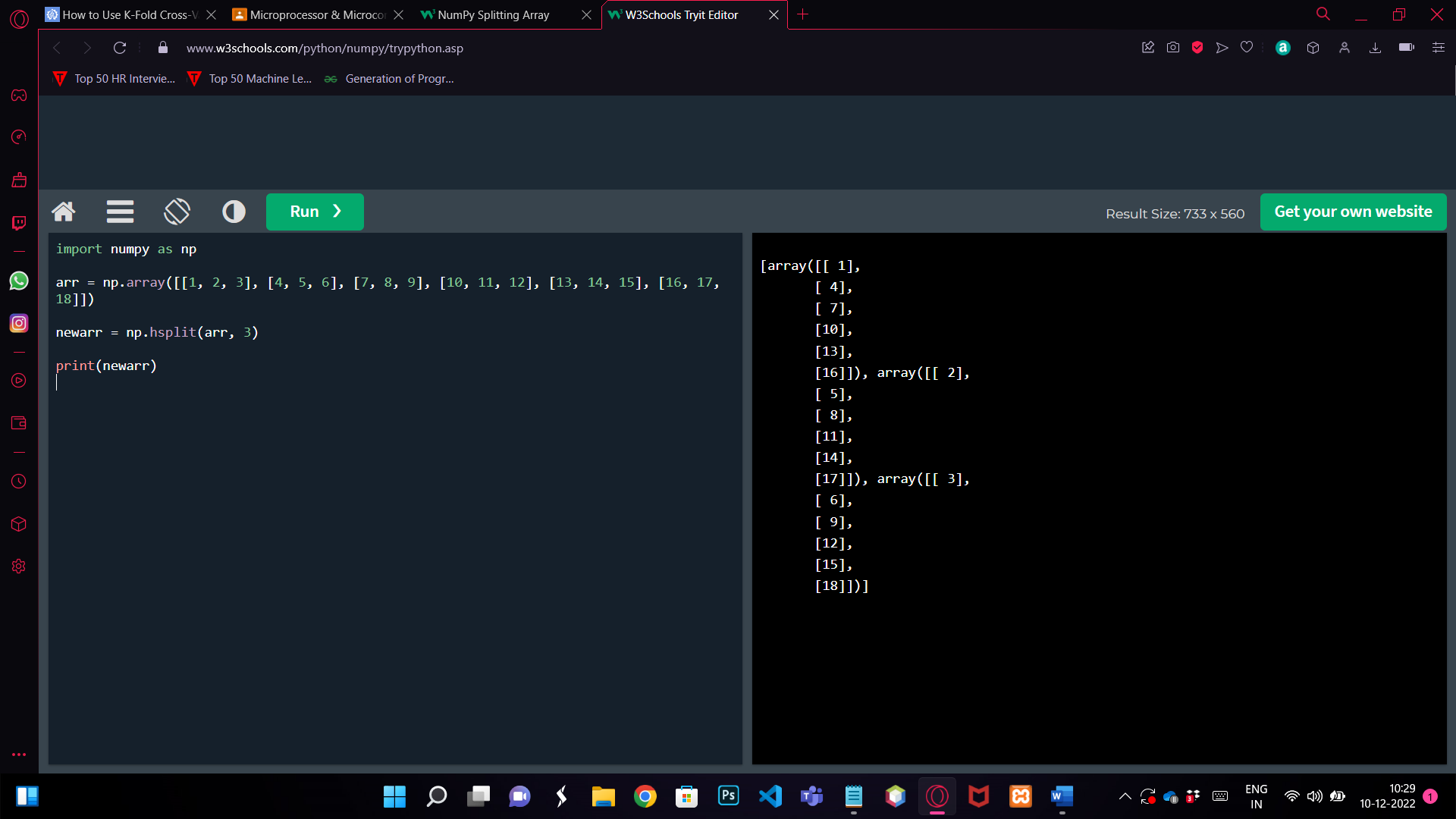
**c)Splitting along Height**

import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12], [13, 14, 15], [16, 17, 18]])

newarr = np.hsplit(arr, 3)

print(newarr)



1. **Array Searching**

import numpy as np

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

x = np.where(arr == 4)

print(x)

OUTPUT: [3 5 6]

**b) Search Sorted**

import numpy as np

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

x = np.searchsorted(arr, 7)

print(x)

OUTPUT: 1

**c) Search Sorted with Multiple Values**

import numpy as np

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

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

print(x)

OUTPUT: [1 2 3]

1. **Array Sorting**

import numpy as np

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

print(np.sort(arr))

OUTPUT: [0 1 2 3]

**b) With Strings**

import numpy as np

arr = np.array(['banana', 'cherry', 'apple'])

print(np.sort(arr))

OUTPUT:[apple banana cherry]

1. **Sorting a 2D Array**

import numpy as np

arr = np.array([[3, 2, 4], [5, 0, 1]])

print(np.sort(arr))

OUTPUT: [[2 3 4]

[0 1 5]]

1. **Sorting Array in Descending Order**

import numpy as np

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

x=np.sort(arr)[::-1]

print(x)

OUTPUT: [3 2 1 0]

1. **Filtering Array**

**a**) import numpy as np

arr = np.array([41, 42, 43, 44])

x = [True, False, True, False]

newarr = arr[x]

print(newarr)

OUTPUT: [41 43]

**b)** import numpy as np

arr = np.array([41, 42, 43, 44])

filter\_arr = arr > 42

newarr = arr[filter\_arr]

print(filter\_arr)

print(newarr)

OUTPUT:[False False True True]

[43 44]

**c)** import numpy as np

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

filter\_arr = arr % 2 == 0

newarr = arr[filter\_arr]

print(filter\_arr)

print(newarr)

OUTPUT: [False True False True False]

[2 4]

1. **Pandas Series**

import pandas as pd

a = [1, 7, 2]

myvar = pd.Series(a)

print(myvar)

OUTPUT: 0 1

1 7

2 2

Dtype:int64

**b) Creating Labels**

import pandas as pd

a = [1, 7, 2]

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

print(myvar)

OUPUT: x 1

Y 7

Z 2

Dtype:int64

1. **Pandas Dataframe**

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

**b**) import pandas as pd

data = {

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

"duration": [50, 40, 45]

}

df = pd.DataFrame(data)

print(df.loc[[0, 1]])

OUTPUT:

calories duration

0 420 50

1 380 40

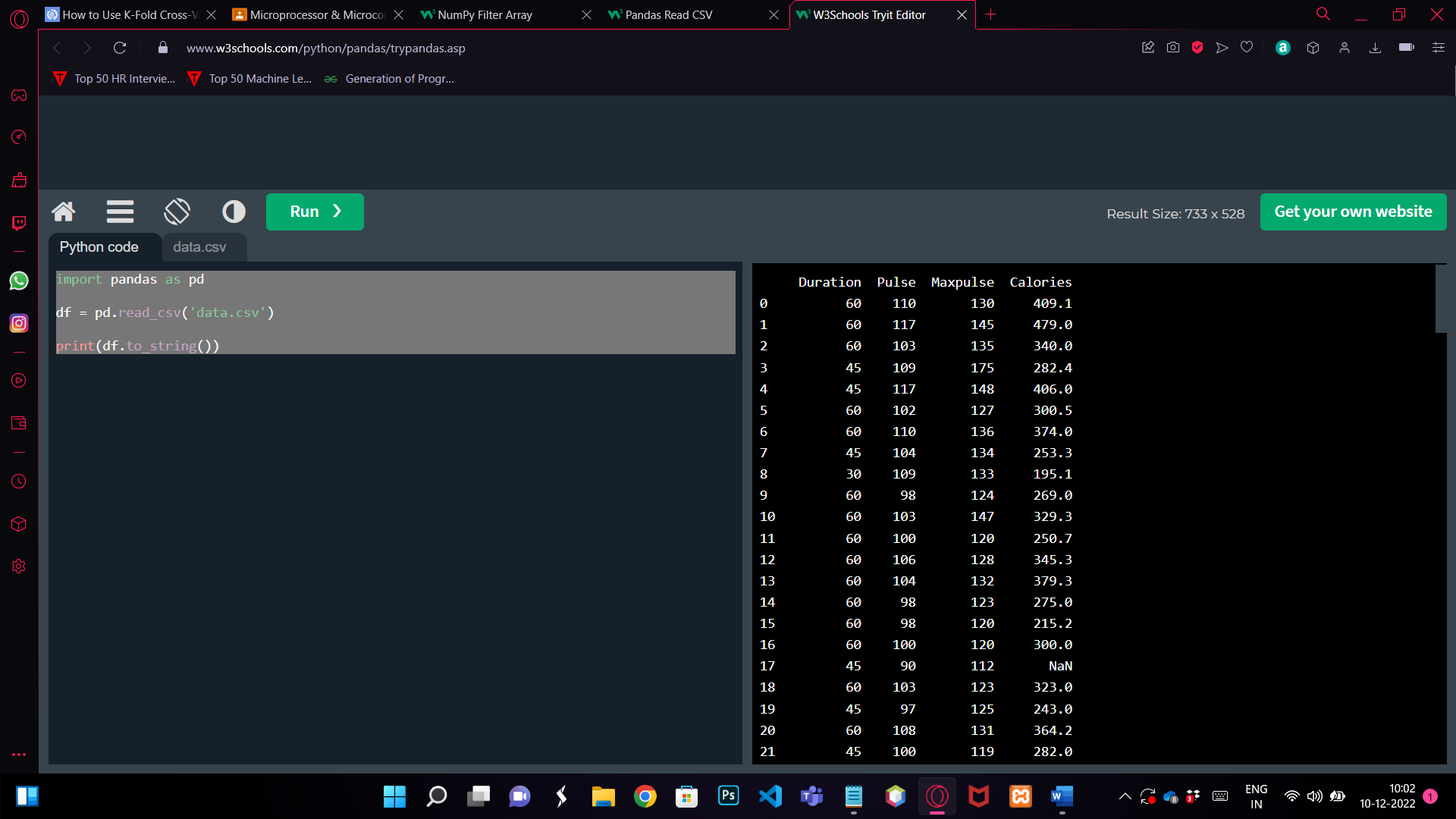
1. **Pandas Read CSV File**

import pandas as pd

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

print(df.to\_string())

OUTPUT:



**b) Checking the System’s Maximum No. of Rows**

import pandas as pd

print(pd.options.display.max\_rows)

OUTPUT: 60

**c)Changing the Maximum No. of Rows**

import pandas as pd

pd.options.display.max\_rows = 9999

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

print(df)

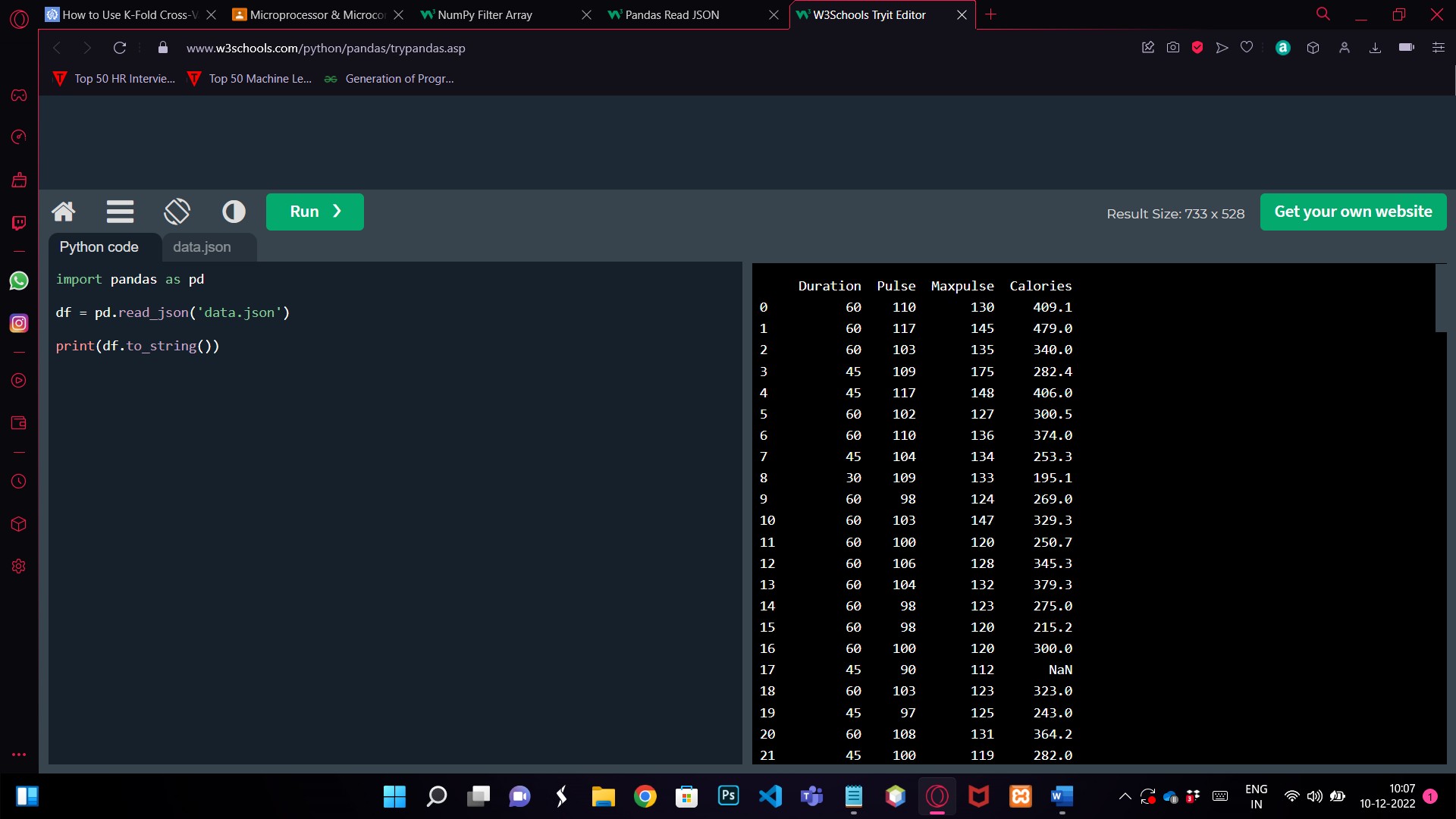
1. **Pandas Read JSON File**

import pandas as pd

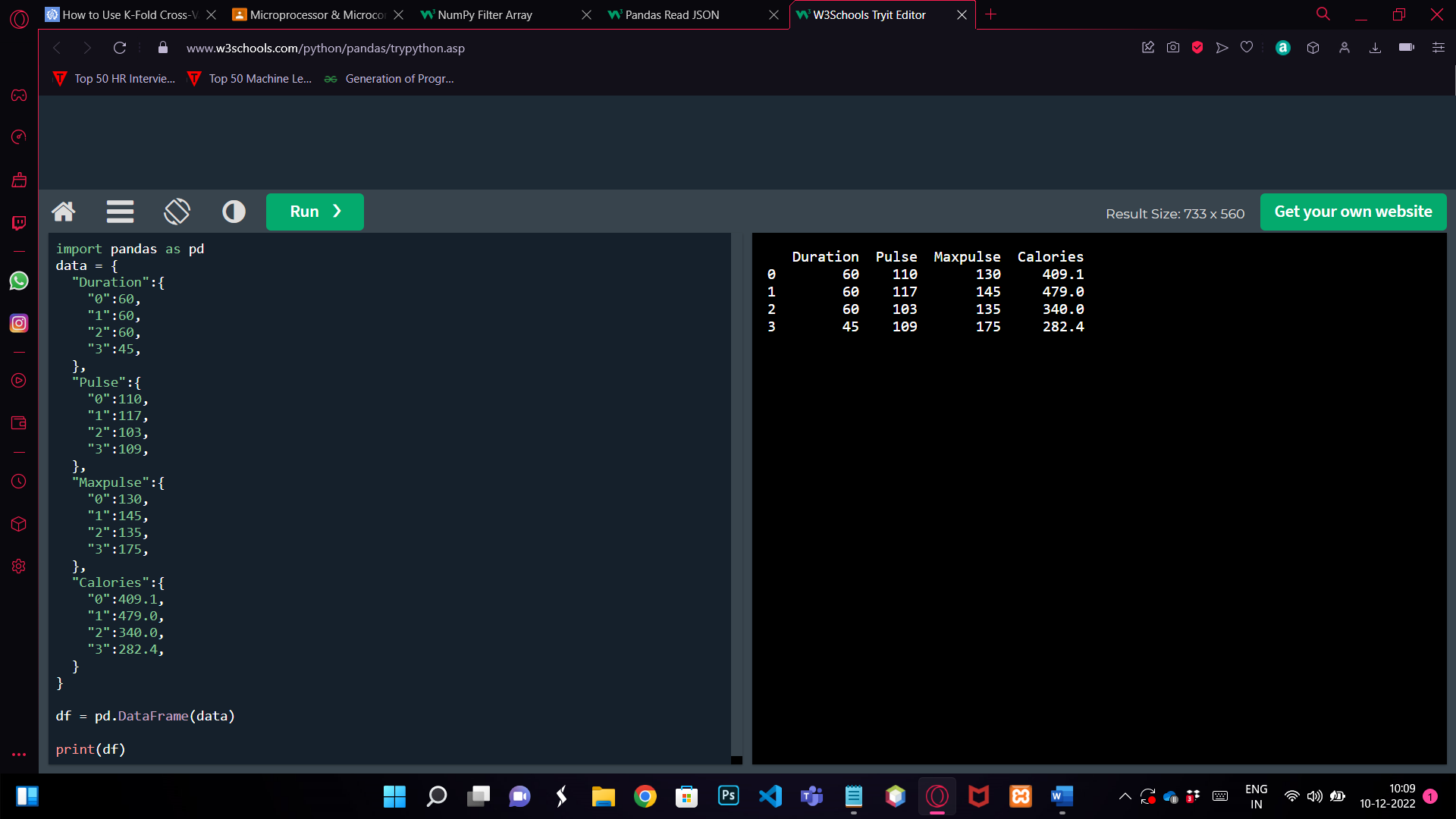
df = pd.read\_json('data.json')

print(df.to\_string())

OUTPUT:



1. **Loading A Python Dictionary into DataFrame**



1. **MatplotLib PyPlot**

import matplotlib.pyplot as plt

import numpy as np

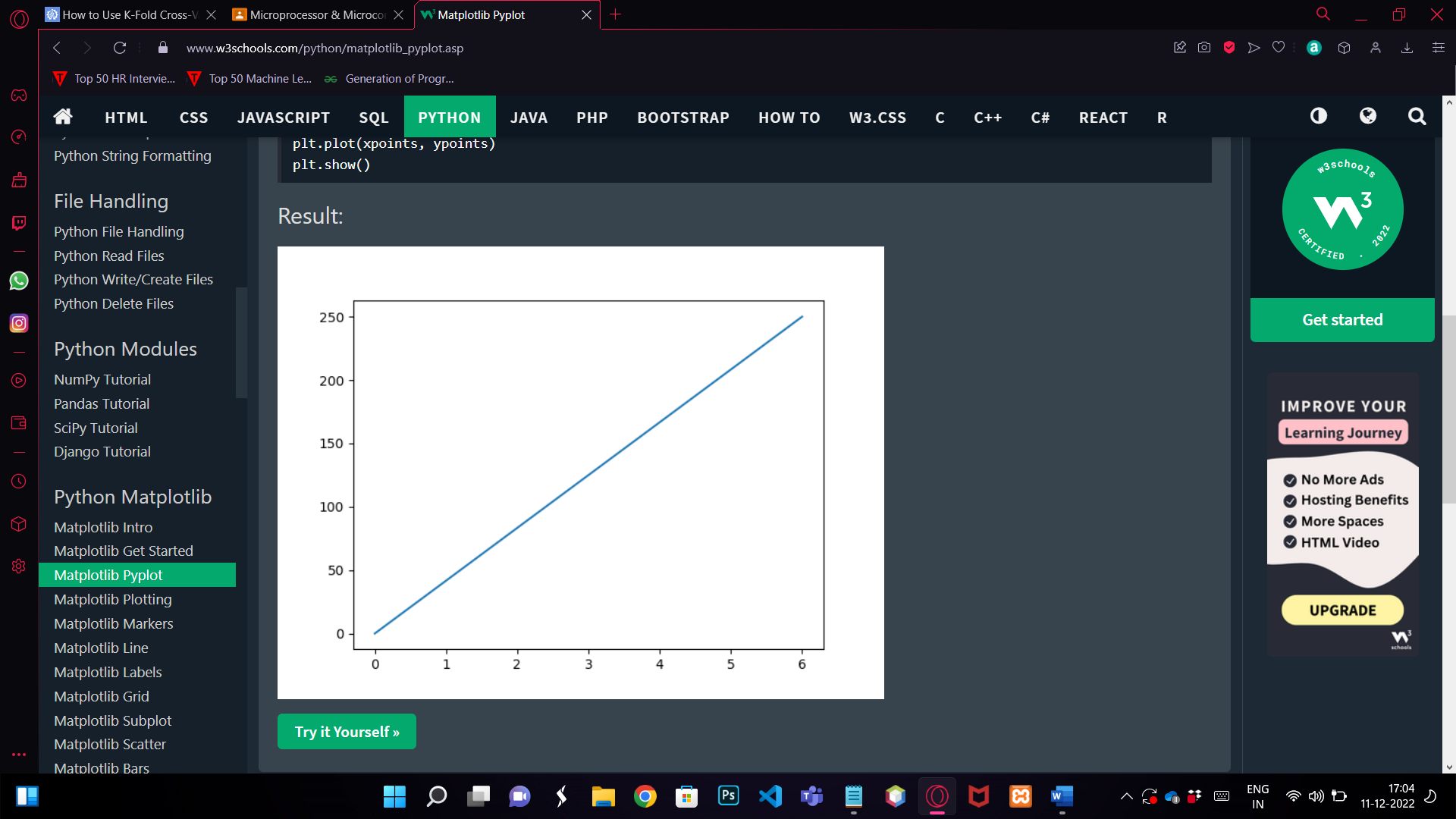
xpoints = np.array([0, 6])

ypoints = np.array([0, 250])

plt.plot(xpoints, ypoints)

plt.show()

OUTPUT:



1. **MatPlotLib Plotting**

import matplotlib.pyplot as plt

import numpy as np

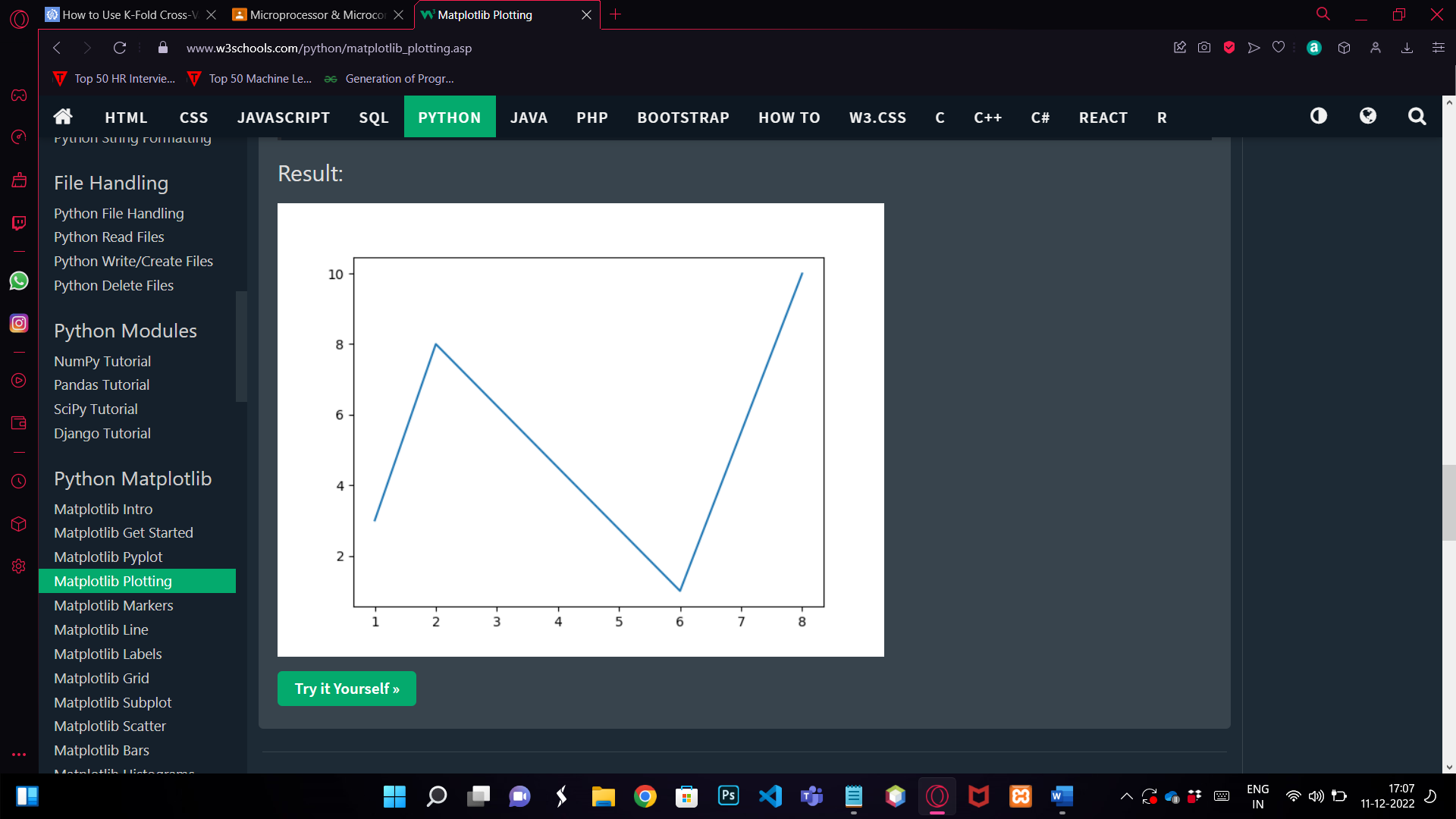
xpoints = np.array([1, 2, 6, 8])

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

plt.plot(xpoints, ypoints)

plt.show()

OUTPUT:



1. **MatPlotLib Markers**

import matplotlib.pyplot as plt

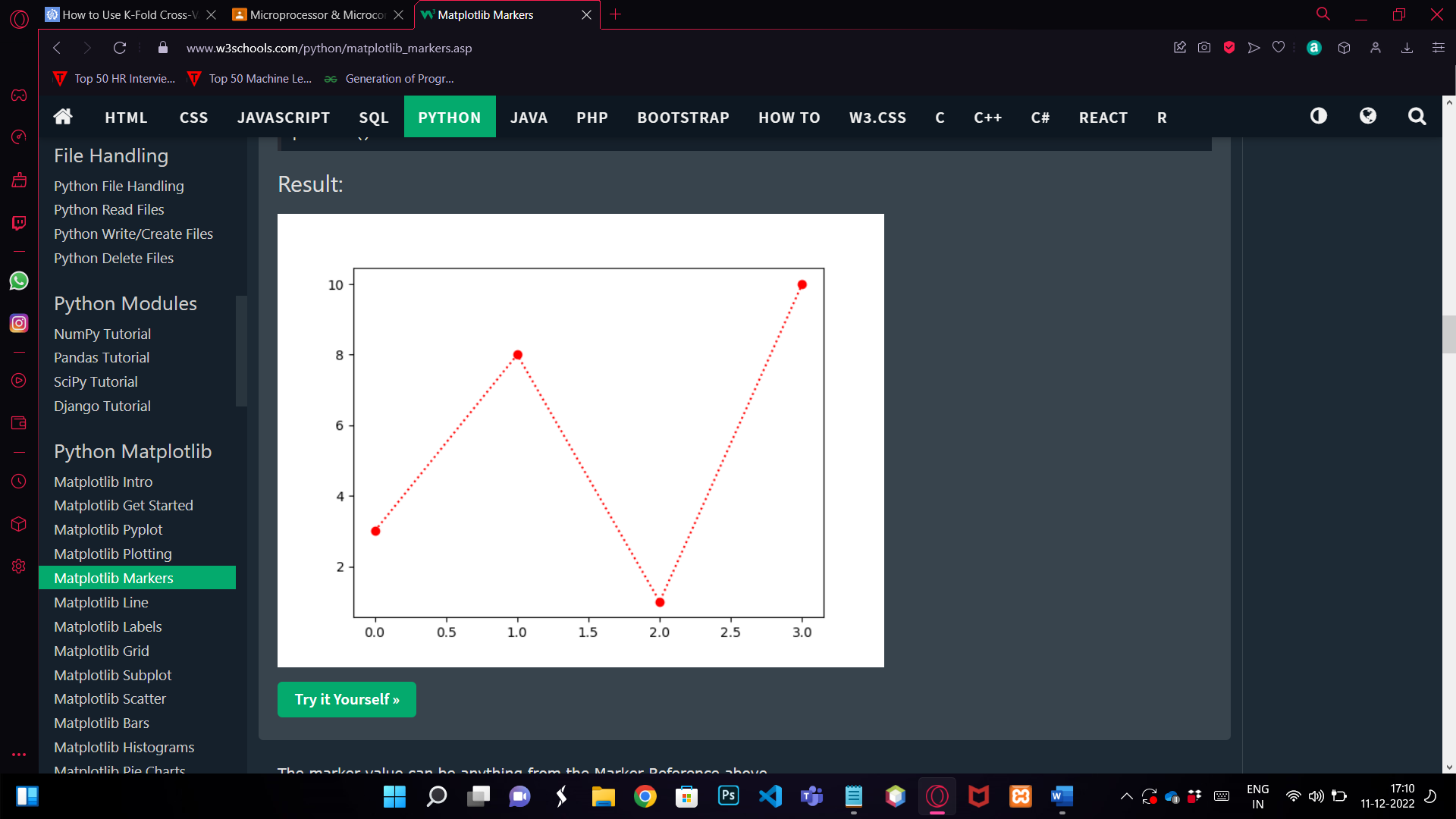
import numpy as np

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

plt.plot(ypoints, 'o:r')

plt.show()

OUTPUT:



1. **MatPlotLib Lines**

import matplotlib.pyplot as plt

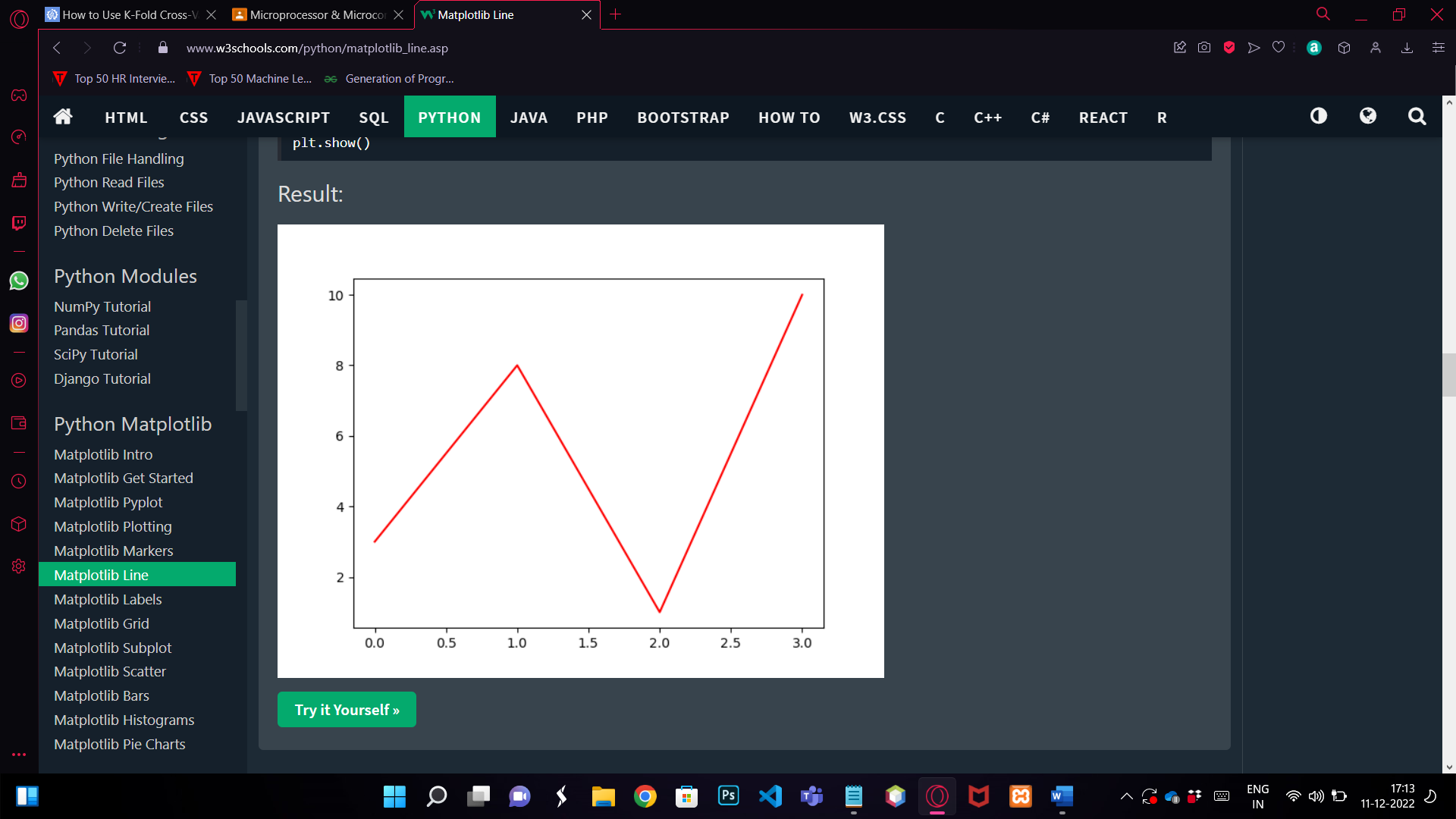
import numpy as np

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

plt.plot(ypoints, color = 'r')

plt.show()

OUTPUT:



**b**) import matplotlib.pyplot as plt

import numpy as np

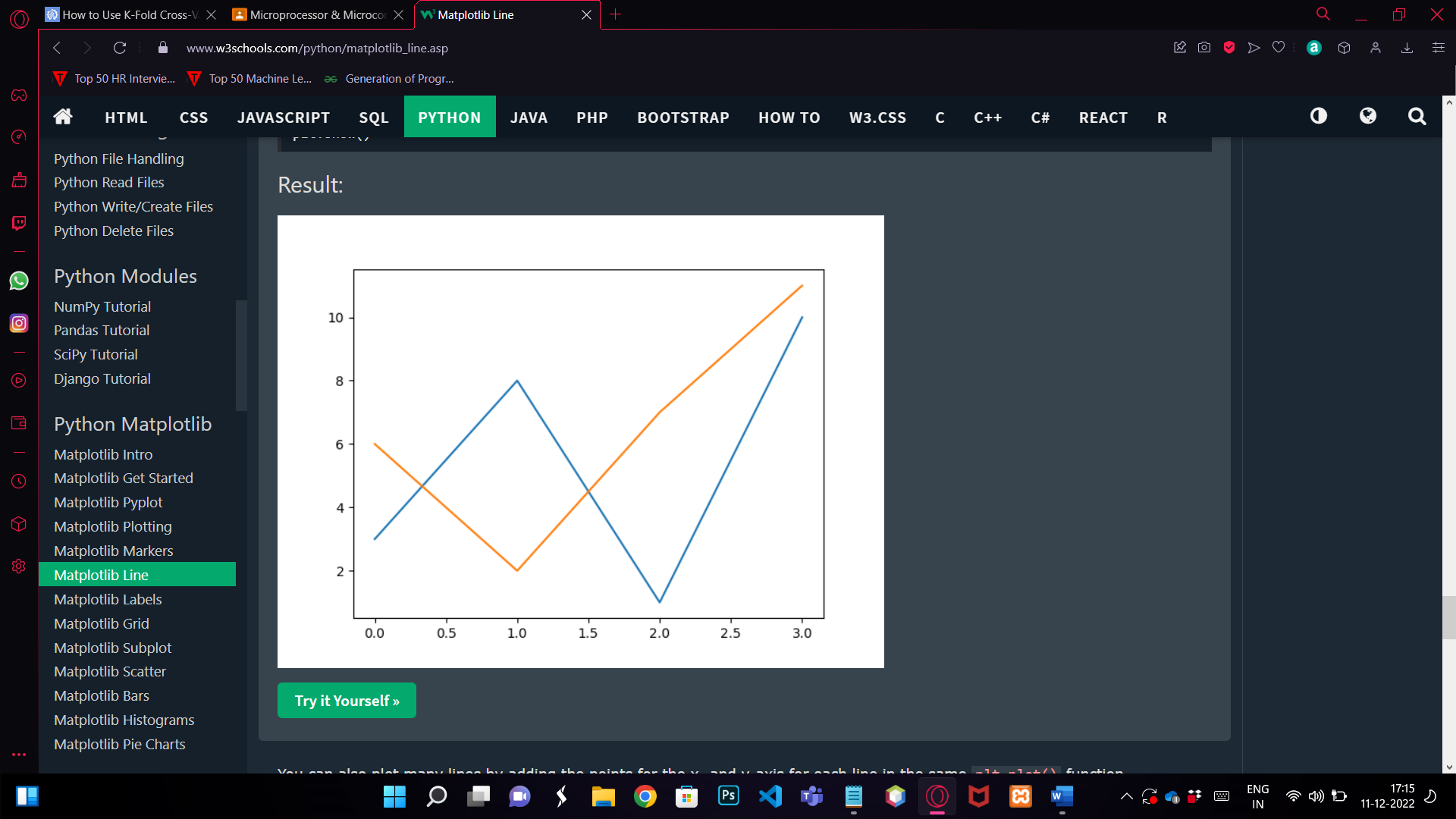
y1 = np.array([3, 8, 1, 10])

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

plt.plot(y1)

plt.plot(y2)

plt.show()



1. **MatPlotLib Labels**

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)

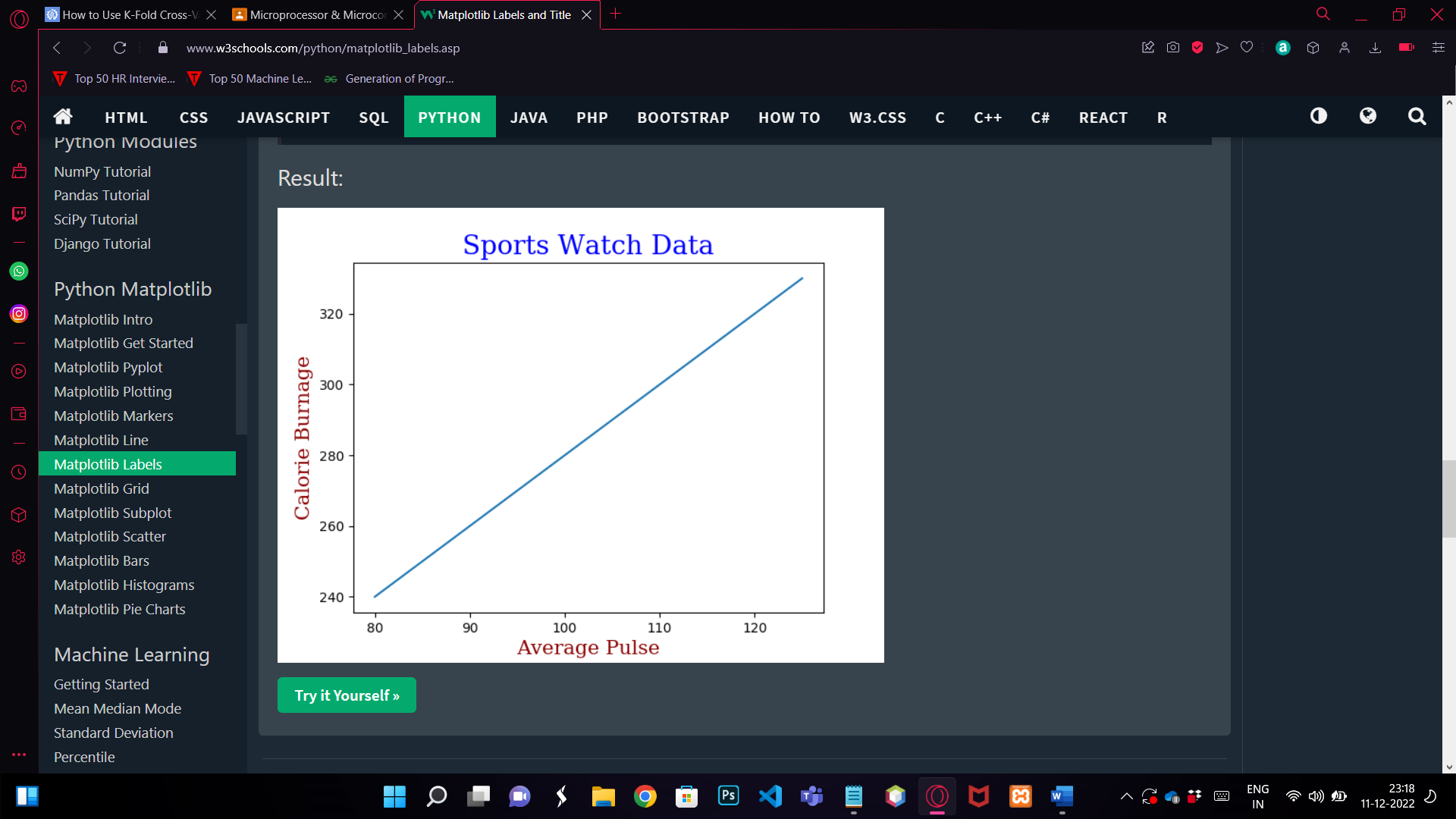
plt.xlabel("Average Pulse", fontdict = font2)

plt.ylabel("Calorie Burnage", fontdict = font2)

plt.plot(x, y)

plt.show()

OUTPUT:



1. **MatPlotLib Grid**

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:



1. **MatPlotLib SubPlot**

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])

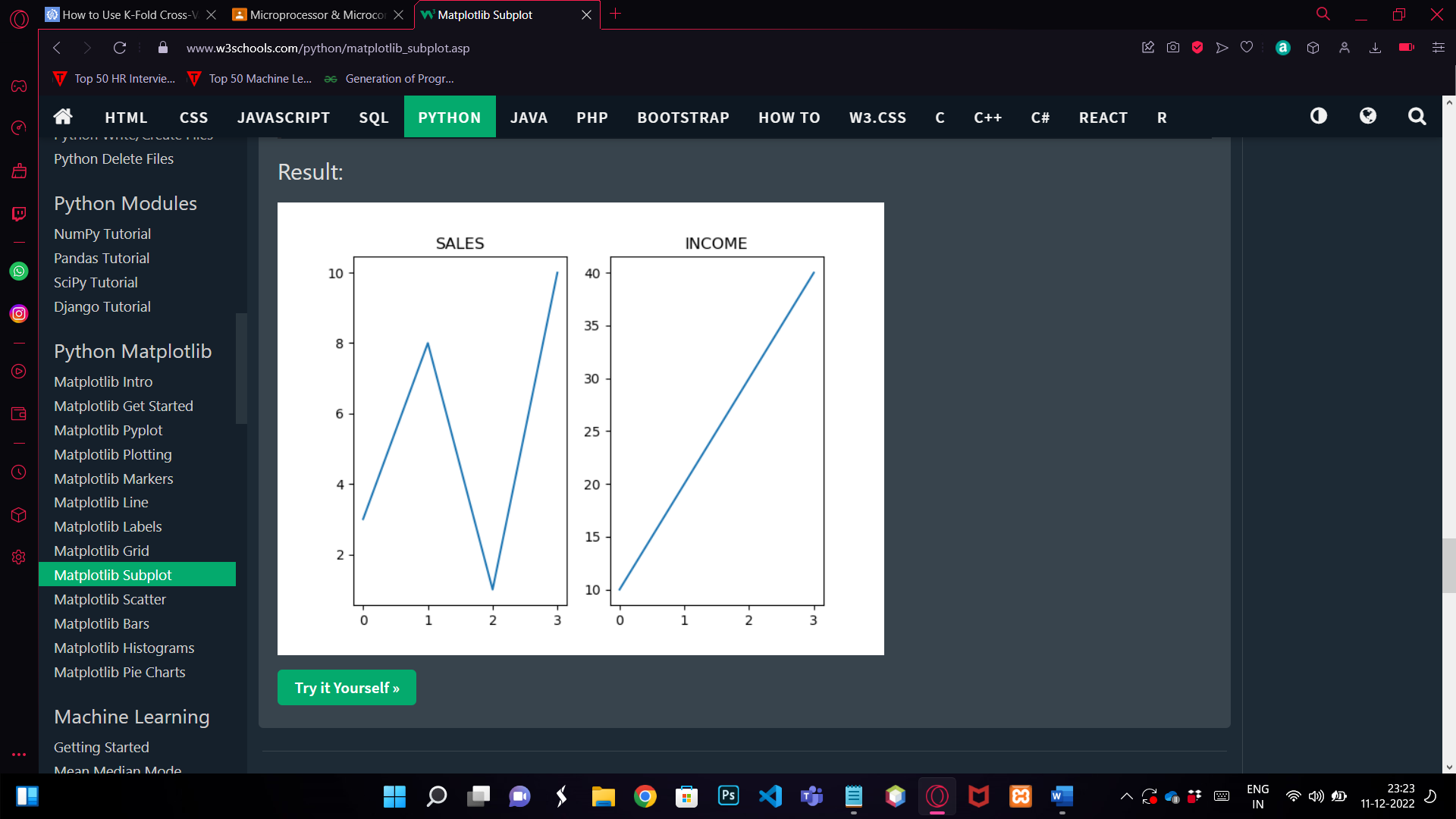
plt.subplot(1, 2, 2)

plt.plot(x,y)

plt.title("INCOME")

plt.show()

OUTPUT:



1. **MatPlotLib Scatter**

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:



1. **MatPlotLib Bar**

import matplotlib

matplotlib.use('Agg')

import matplotlib.pyplot as plt

import numpy as np

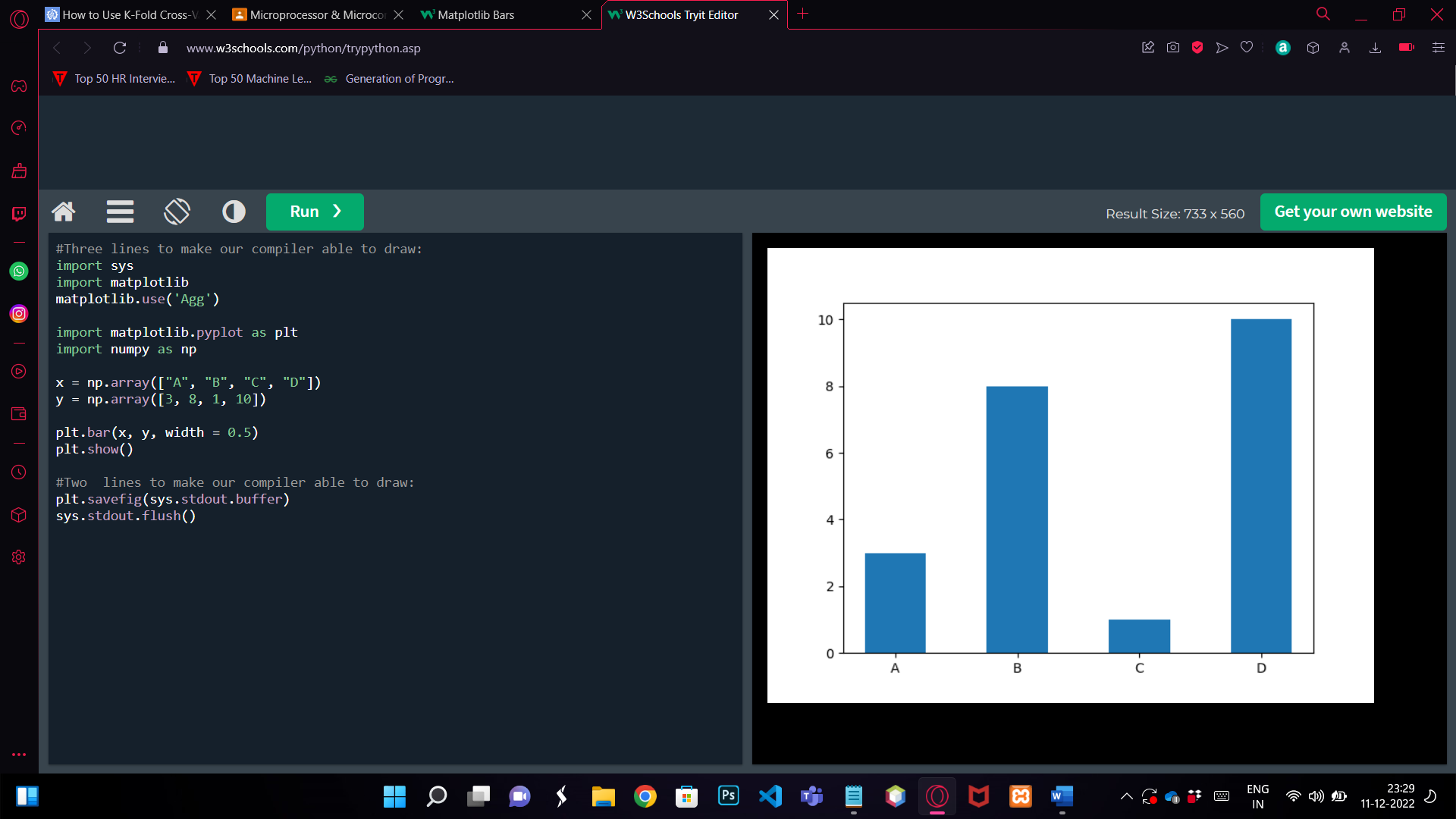
x = np.array(["A", "B", "C", "D"])

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

plt.bar(x, y, width = 0.5)

plt.show()

OUTPUT:



1. **MatPlotLib Histograms**

import matplotlib.pyplot as plt

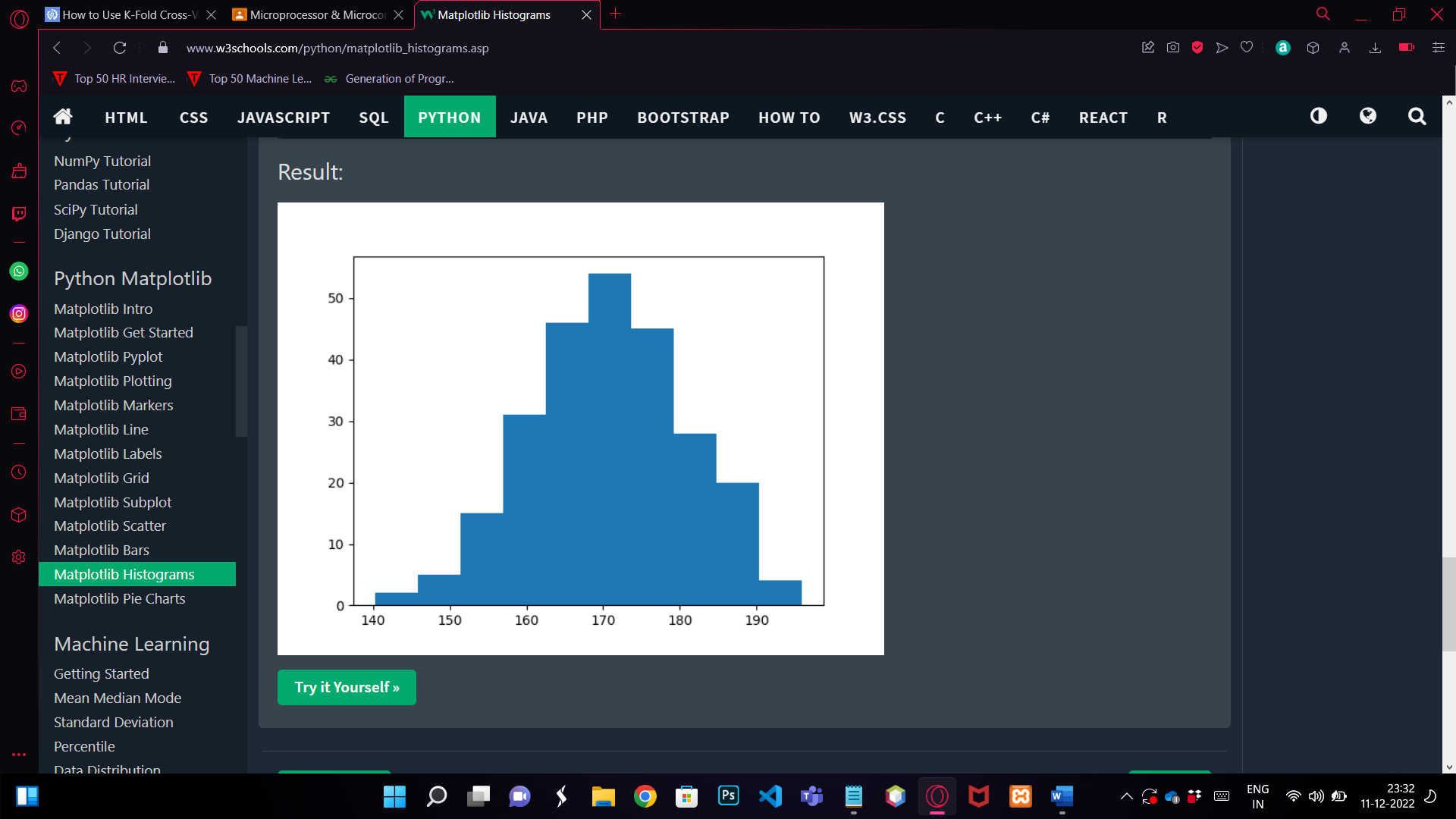
import numpy as np

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

plt.hist(x)

plt.show()

OUTPUT:



1. **MatPlotLib PieChart**

import matplotlib.pyplot as plt

import numpy as np

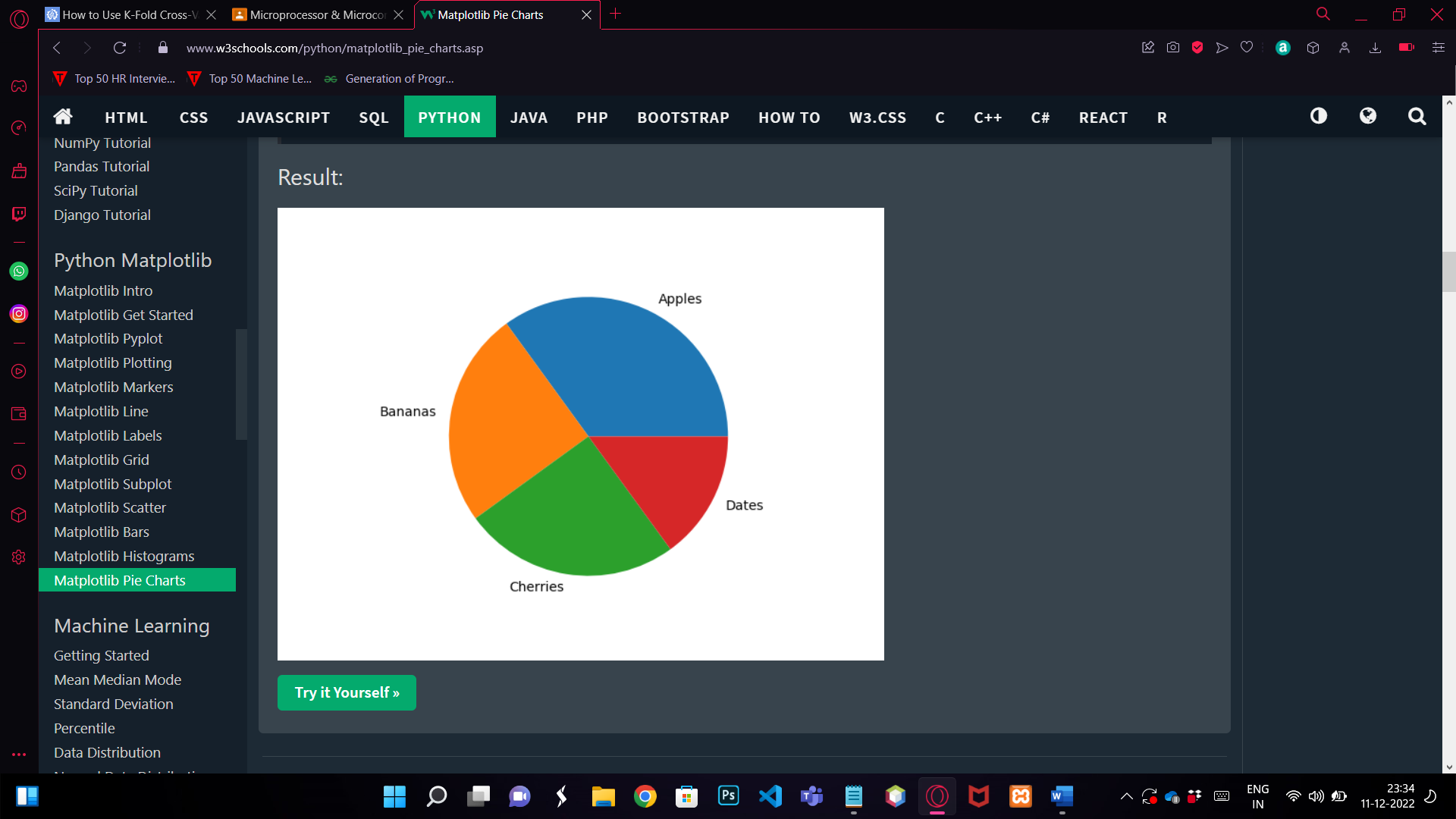
y = np.array([35, 25, 25, 15])

mylabels = ["Apples", "Bananas", "Cherries", "Dates"]

plt.pie(y, labels = mylabels)

plt.show()

OUTPUT:



1. **MatPlotLib Mean Median Mode**

**a) Mean**

import numpy

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

x = numpy.mean(speed)

print(x)

OUTPUT: 89.77

**b) Median**

import numpy

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

x = numpy.median(speed)

print(x)

OUTPUT: 87.0

**c)Mode**

import numpy

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

x = numpy.mode(speed)

print(x)

OUTPUT: 86.0

1. **MatPlotLib Standard Deviation**

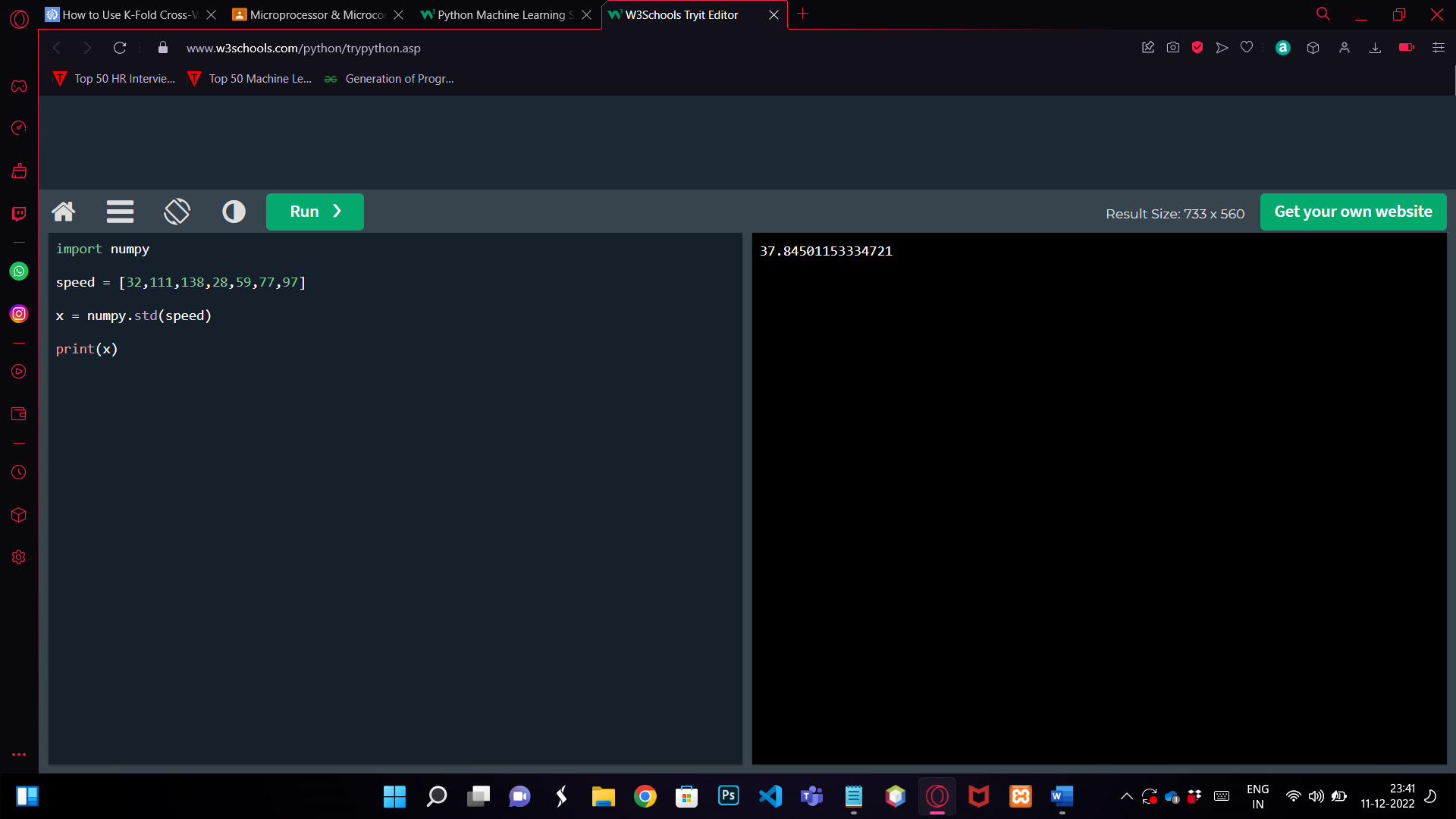
import numpy

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

x = numpy.std(speed)

print(x)

OUTPUT:



1. **Linear Regression**

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:



1. **Multiple Progression**

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: 107.208

1. **Polynomial Regression**

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:

