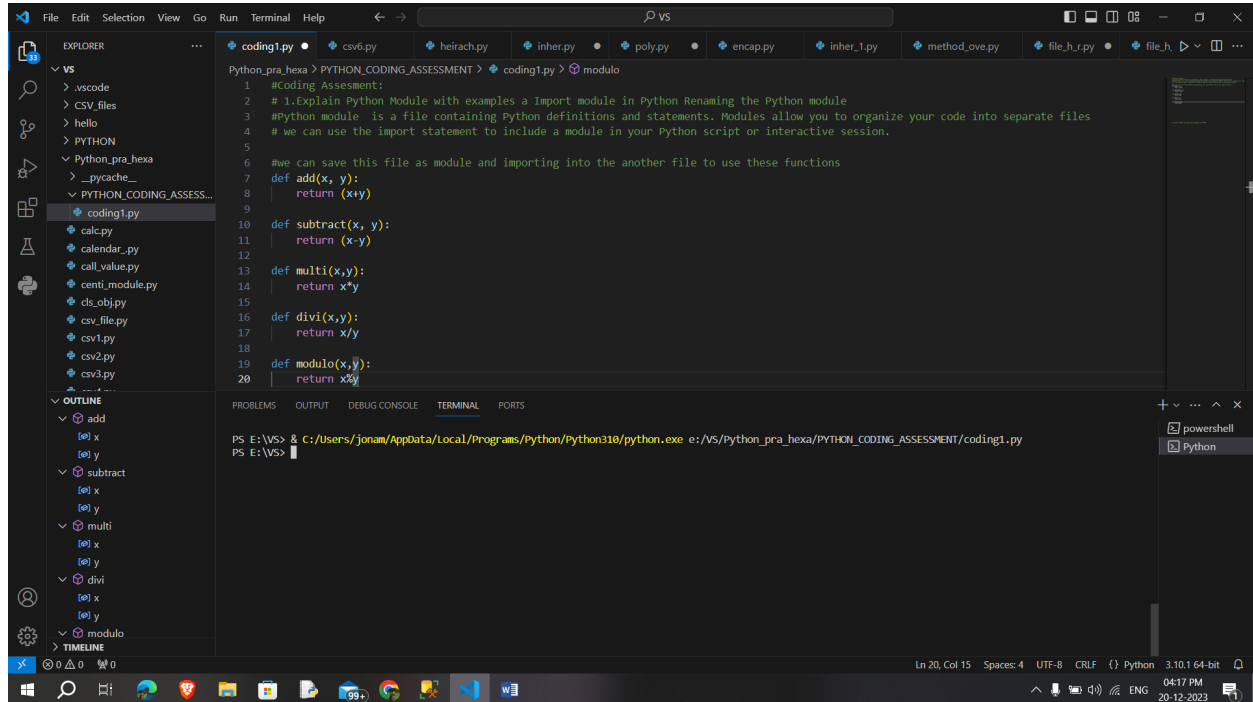


## 1.Explain Python Module with examples.Import module in Python.Renaming the Python module

Python module-a module is a file containing Python definitions and statements. Modules allow you to organize your code into separate files, making it easier to manage and maintain.

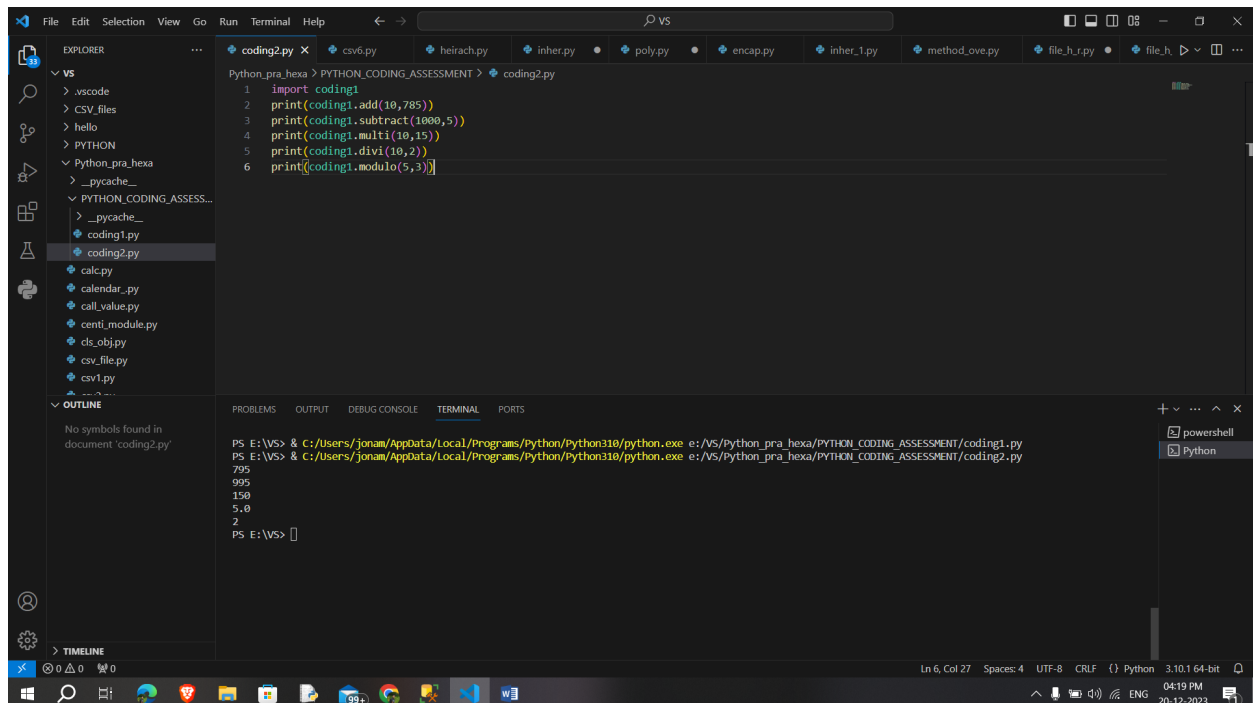
**Creating own module and importing it in the another file,**



The screenshot shows the Visual Studio Code interface with a file explorer on the left and a code editor in the center. The file explorer shows a project structure with a folder named 'PYTHON\_CODING\_ASSESSMENT' containing several Python files, including 'coding1.py'. The code editor displays the content of 'coding1.py', which defines several functions: 'add', 'subtract', 'multi', 'divi', and 'modulo'. The terminal at the bottom shows the command to run the script: 'PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/python310/python.exe e:/VS/Python\_pra\_hexa/PYTHON\_CODING\_ASSESSMENT/coding1.py'.

```
1 #Coding Assessment:
2 # 1.Explain Python Module with examples a Import module in Python Renaming the Python module
3 #Python module is a file containing Python definitions and statements. Modules allow you to organize your code into separate files
4 # we can use the import statement to include a module in your Python script or interactive session.
5
6 #we can save this file as module and importing into the another file to use these functions
7 def add(x, y):
8     return x+y
9
10 def subtract(x, y):
11     return (x-y)
12
13 def multi(x,y):
14     return x*y
15
16 def divi(x,y):
17     return x/y
18
19 def modulo(x,y):
20     return x%y
```

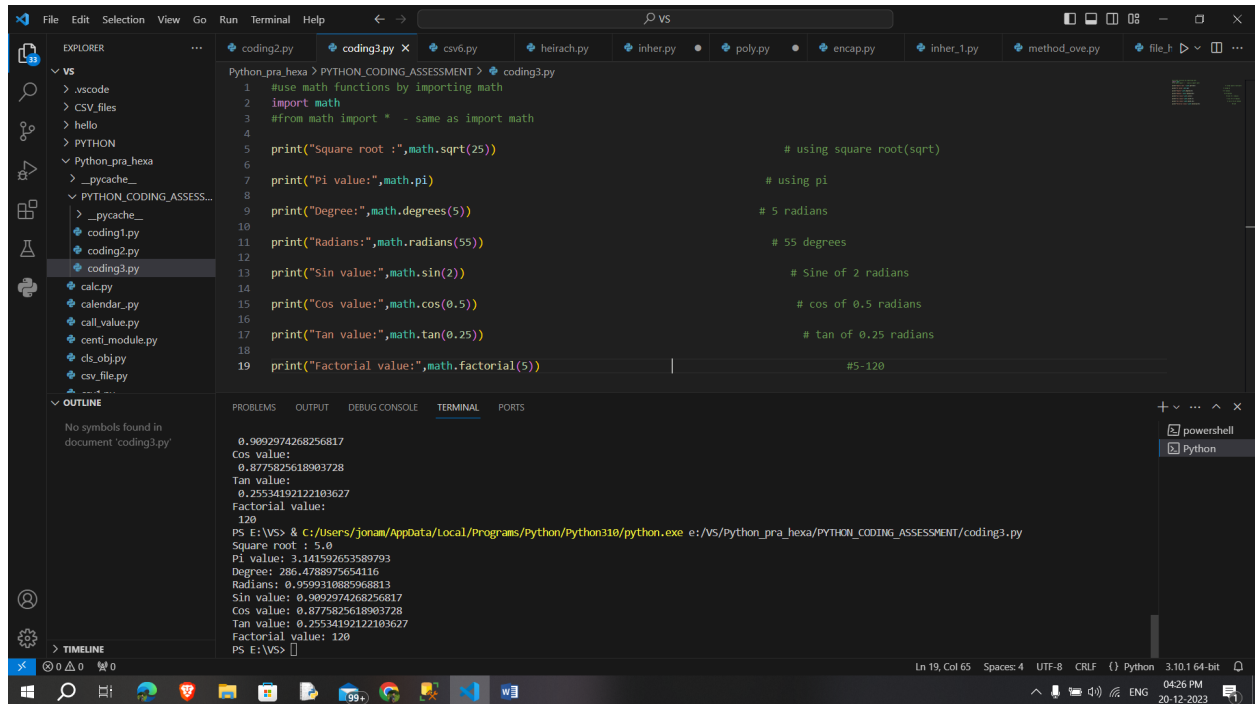
## Importing the own module and printing the output,



The screenshot shows the Visual Studio Code interface with a file explorer on the left and a code editor in the center. The file explorer shows the same project structure as the previous screenshot, but now 'coding2.py' is selected. The code editor displays the content of 'coding2.py', which imports the 'coding1' module and uses its functions to calculate and print the sum, difference, product, quotient, and remainder of 10 and 785. The terminal at the bottom shows the command to run the script: 'PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/python310/python.exe e:/VS/Python\_pra\_hexa/PYTHON\_CODING\_ASSESSMENT/coding2.py', followed by the output: '795', '995', '150', '5.0', and '2'.

```
1 import coding1
2 print(coding1.add(10,785))
3 print(coding1.subtract(1000,5))
4 print(coding1.multi(10,15))
5 print(coding1.divi(10,2))
6 print(coding1.modulo(5,3))
```

Importing math functions are square root,pi value,degree,radians etc,



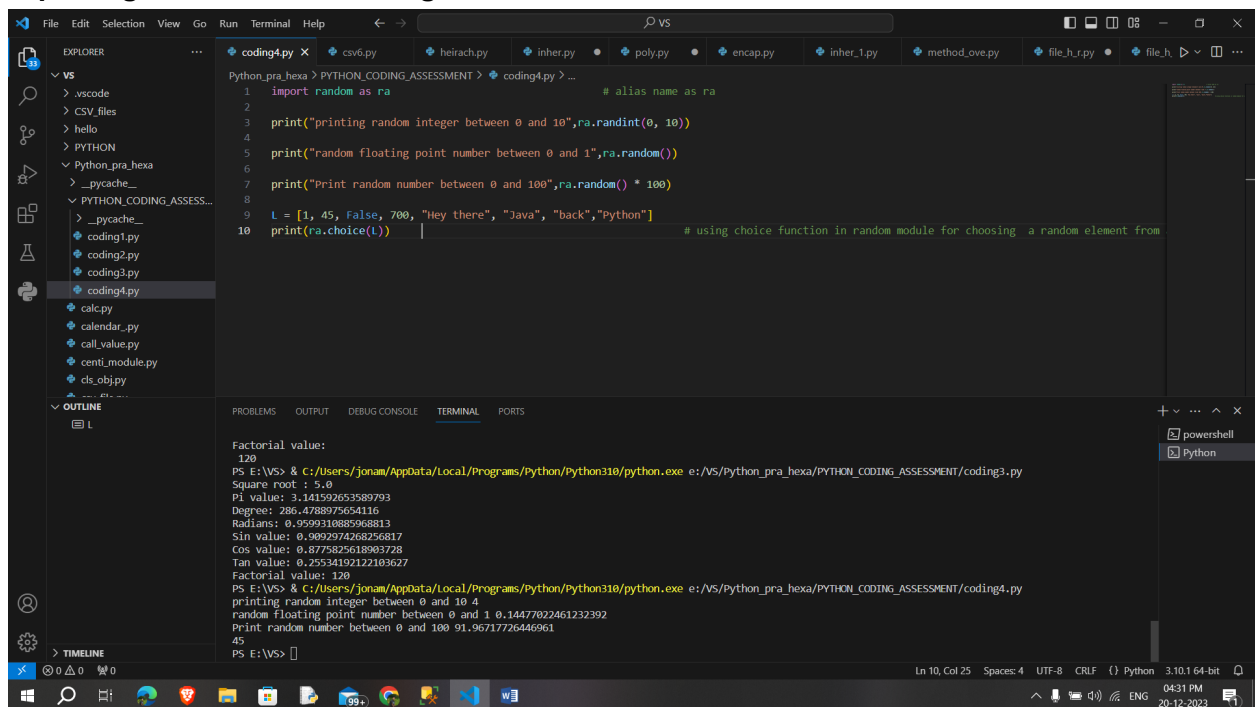
The screenshot shows the VS Code editor with a Python file named `coding3.py`. The code imports the `math` module and prints several mathematical values. The terminal output shows the results of these calculations.

```
1 #use math functions by importing math
2 import math
3 #from math import * - same as import math
4
5 print("Square root :",math.sqrt(25))           # using square root(sqrt)
6
7 print("Pi value:",math.pi)                   # using pi
8
9 print("Degree:",math.degrees(5))              # 5 radians
10
11 print("Radians:",math.radians(55))            # 55 degrees
12
13 print("Sin value:",math.sin(2))               # Sine of 2 radians
14
15 print("Cos value:",math.cos(0.5))             # cos of 0.5 radians
16
17 print("Tan value:",math.tan(0.25))            # tan of 0.25 radians
18
19 print("Factorial value:",math.factorial(5))   #5-120
```

Terminal Output:

```
0.9092974268256817
Cos value:
0.8775825618903728
Tan value:
0.25534192122103627
Factorial value:
120
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding3.py
Square root : 5.0
Pi value: 3.141592653589793
Degree: 286.4788975654116
Radians: 0.9599310885968813
Sin value: 0.9092974268256817
Cos value: 0.8775825618903728
Tan value: 0.25534192122103627
Factorial value: 120
PS E:\VS>
```

Importing random and creating alias name as ra,



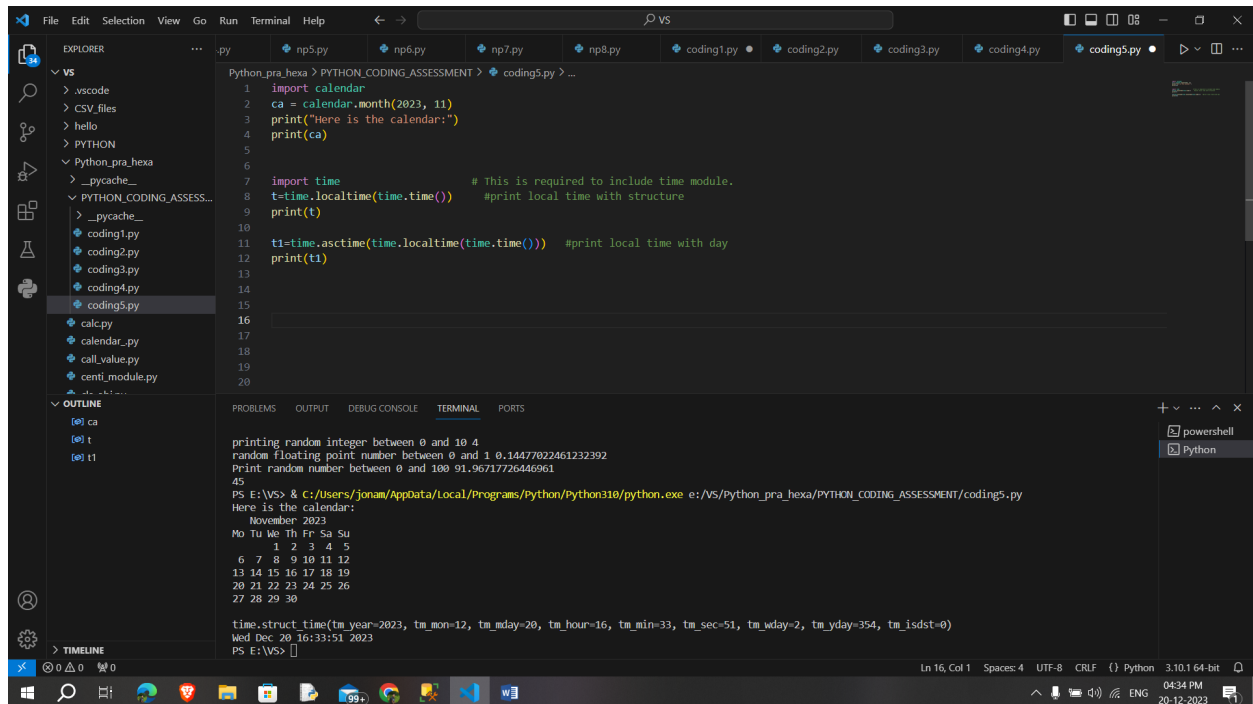
The screenshot shows the VS Code editor with a Python file named `coding4.py`. The code imports the `random` module with an alias `ra` and prints various random values. The terminal output shows the results of these calculations.

```
1 import random as ra           # alias name as ra
2
3 print("printing random integer between 0 and 10",ra.randint(0, 10))
4
5 print("random floating point number between 0 and 1",ra.random())
6
7 print("Print random number between 0 and 100",ra.Random() * 100)
8
9 L = [1, 45, False, 700, "Hey there", "Java", "back", "Python"]
10 print(ra.choice(L))           # using choice function in random module for choosing a random element from
```

Terminal Output:

```
Factorial value:
120
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding3.py
Square root : 5.0
Pi value: 3.141592653589793
Degree: 286.4788975654116
Radians: 0.9599310885968813
Sin value: 0.9092974268256817
Cos value: 0.8775825618903728
Tan value: 0.25534192122103627
Factorial value: 120
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding4.py
printing random integer between 0 and 10 4
random floating point number between 0 and 1 0.14477022461232392
Print random number between 0 and 100 91.96717726446961
45
PS E:\VS>
```

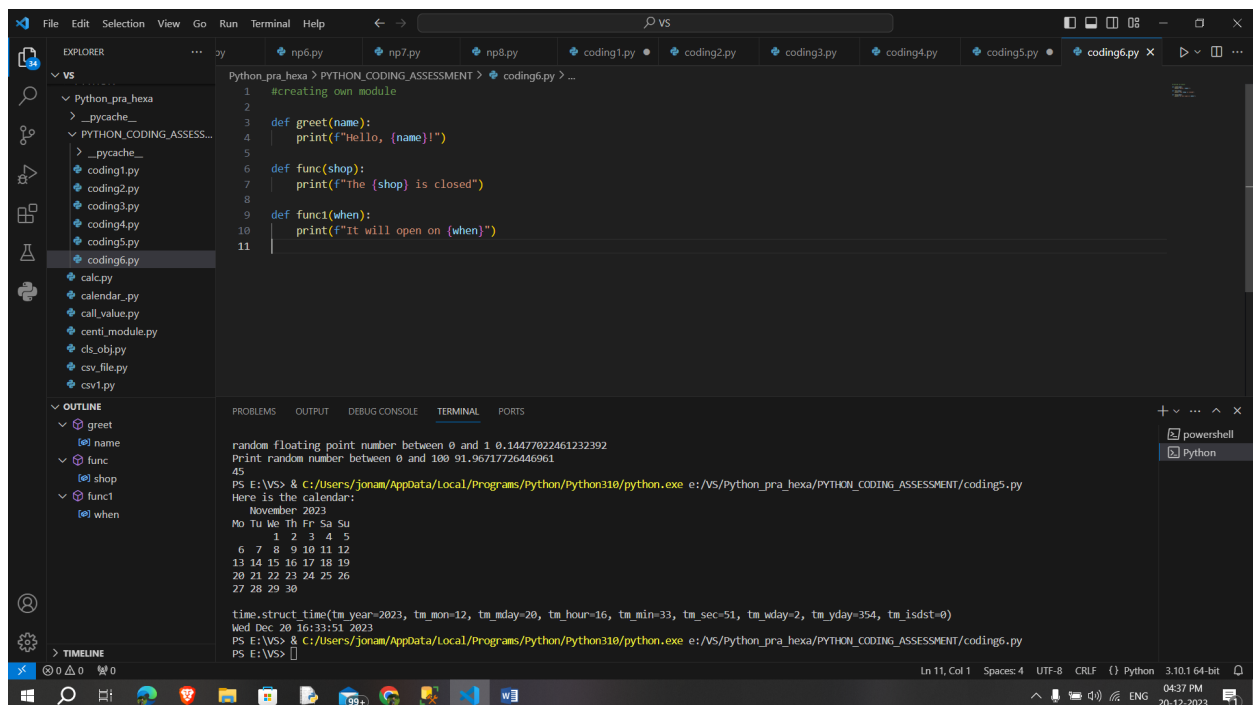
Importing calendar and time module to get the local time and the particular month,



```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding5.py > ...
1 import calendar
2 ca = calendar.month(2023, 11)
3 print("Here is the calendar:")
4 print(ca)
5
6
7 import time # This is required to include time module.
8 t=time.localtime(time.time()) #print local time with structure
9 print(t)
10
11 ti=time.asctime(time.localtime(time.time())) #print local time with day
12 print(ti)
13
14
15
16
17
18
19
20
```

printing random integer between 0 and 10.4  
random floating point number between 0 and 1 0.14477022461232392  
Print random number between 0 and 100 91.96717726446961  
45  
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python\_pra\_hexa/PYTHON\_CODING\_ASSESSMENT/coding5.py  
Here is the calendar:  
November 2023  
Mo Tu We Th Fr Sa Su  
1 2 3 4 5  
6 7 8 9 10 11 12  
13 14 15 16 17 18 19  
20 21 22 23 24 25 26  
27 28 29 30  
time.struct\_time(tm\_year=2023, tm\_mon=12, tm\_mday=20, tm\_hour=16, tm\_min=33, tm\_sec=51, tm\_wday=2, tm\_yday=354, tm\_isdst=0)  
Wed Dec 20 16:33:51 2023  
PS E:\VS> []

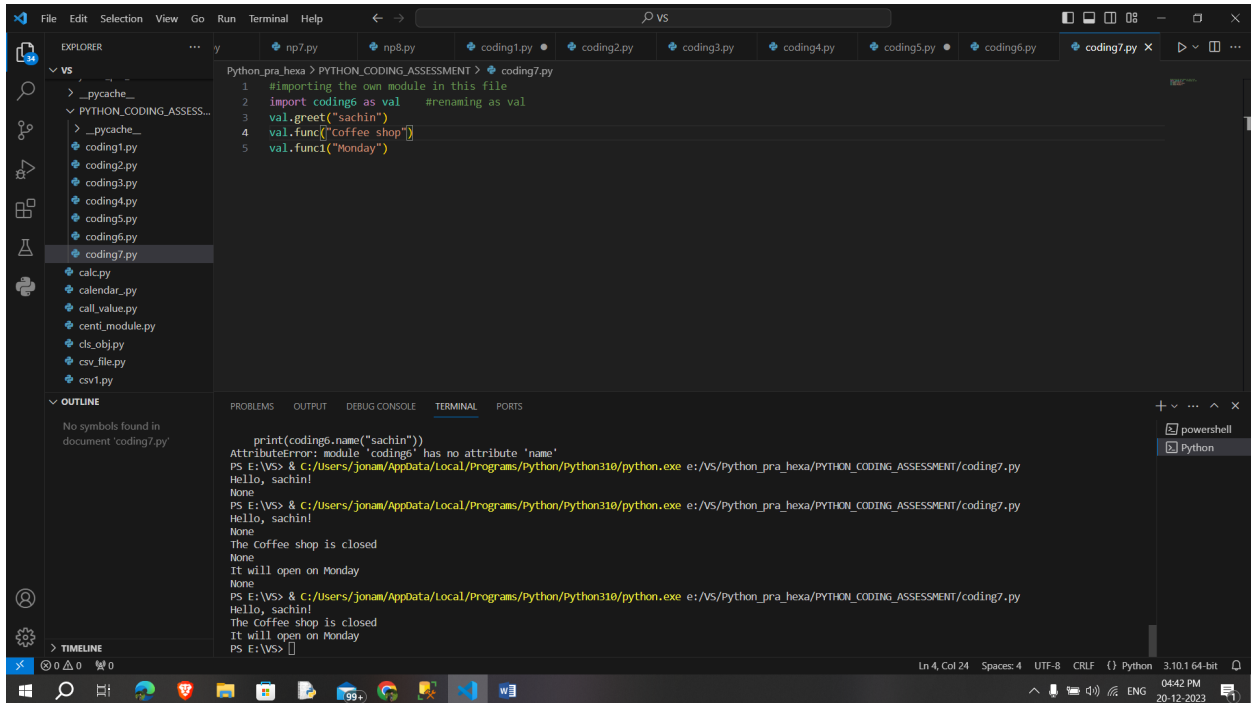
Creating own module called coding6 and importing in the next file,



```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding6.py > ...
1 #creating own module
2
3 def greet(name):
4     print(f"Hello, {name}!")
5
6 def func(shop):
7     print(f"The {shop} is closed")
8
9 def func1(when):
10    print(f"It will open on {when}")
11
```

random floating point number between 0 and 1 0.14477022461232392  
Print random number between 0 and 100 91.96717726446961  
45  
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python\_pra\_hexa/PYTHON\_CODING\_ASSESSMENT/coding5.py  
Here is the calendar:  
November 2023  
Mo Tu We Th Fr Sa Su  
1 2 3 4 5  
6 7 8 9 10 11 12  
13 14 15 16 17 18 19  
20 21 22 23 24 25 26  
27 28 29 30  
time.struct\_time(tm\_year=2023, tm\_mon=12, tm\_mday=20, tm\_hour=16, tm\_min=33, tm\_sec=51, tm\_wday=2, tm\_yday=354, tm\_isdst=0)  
Wed Dec 20 16:33:51 2023  
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python\_pra\_hexa/PYTHON\_CODING\_ASSESSMENT/coding6.py  
PS E:\VS> []

Importing coding6 module to another file and renaming as val and executing it,



```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding7.py
1 #importing the own module in this file
2 import coding6 as val #renaming as val
3 val.greet("sachin")
4 val.func1("coffee shop")
5 val.func1("Monday")

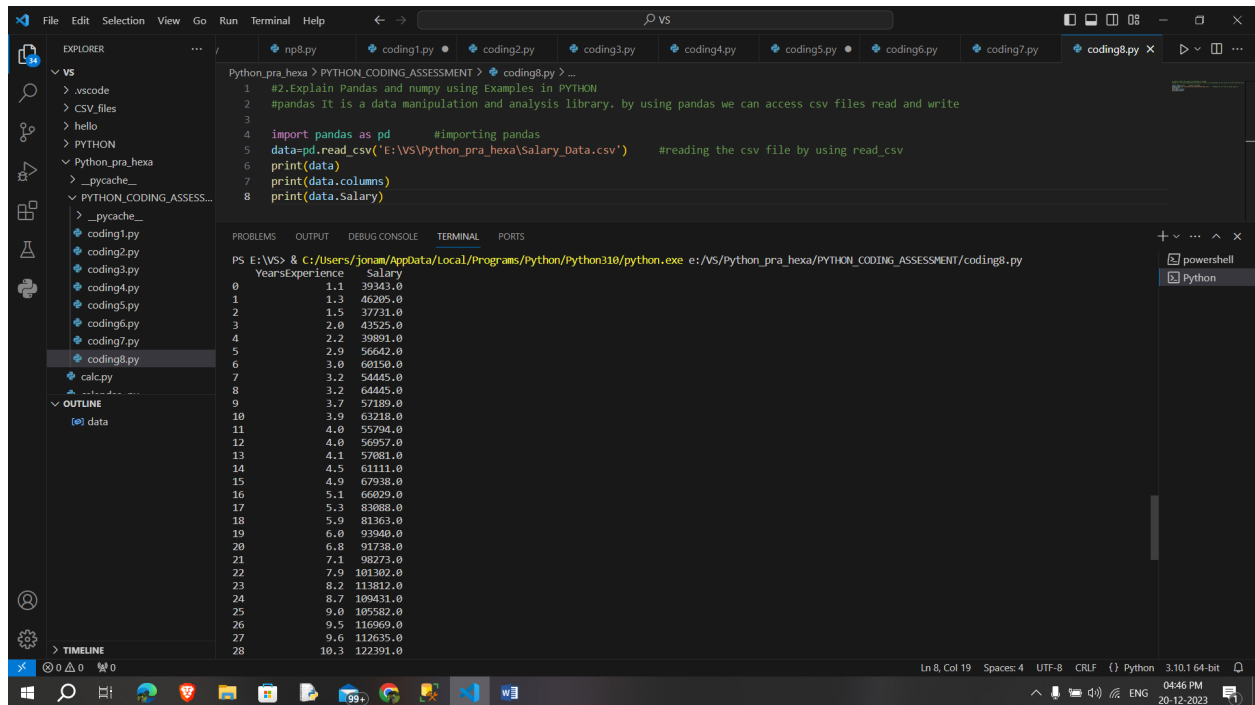
print(coding6.name("sachin"))
AttributeError: module 'coding6' has no attribute 'name'
PS E:\VS> & c:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding7.py
Hello, sachin!
None
PS E:\VS> & c:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding7.py
Hello, sachin!
None
The Coffee shop is closed
None
It will open on Monday
None
PS E:\VS> & c:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding7.py
Hello, sachin!
The Coffee shop is closed
It will open on Monday
PS E:\VS>
```

## 2.Explain Pandas and numpy using Examples in PYTHON

Pandas-Pandas is a data manipulation and analysis library. It provides data structures like Series and DataFrame, which are particularly useful for working with structured data such as CSV files or SQL tables.

Numpy-NumPy is a powerful numerical computing library in Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions to operate on these arrays.

By using pandas we can access the csv files,reading the file

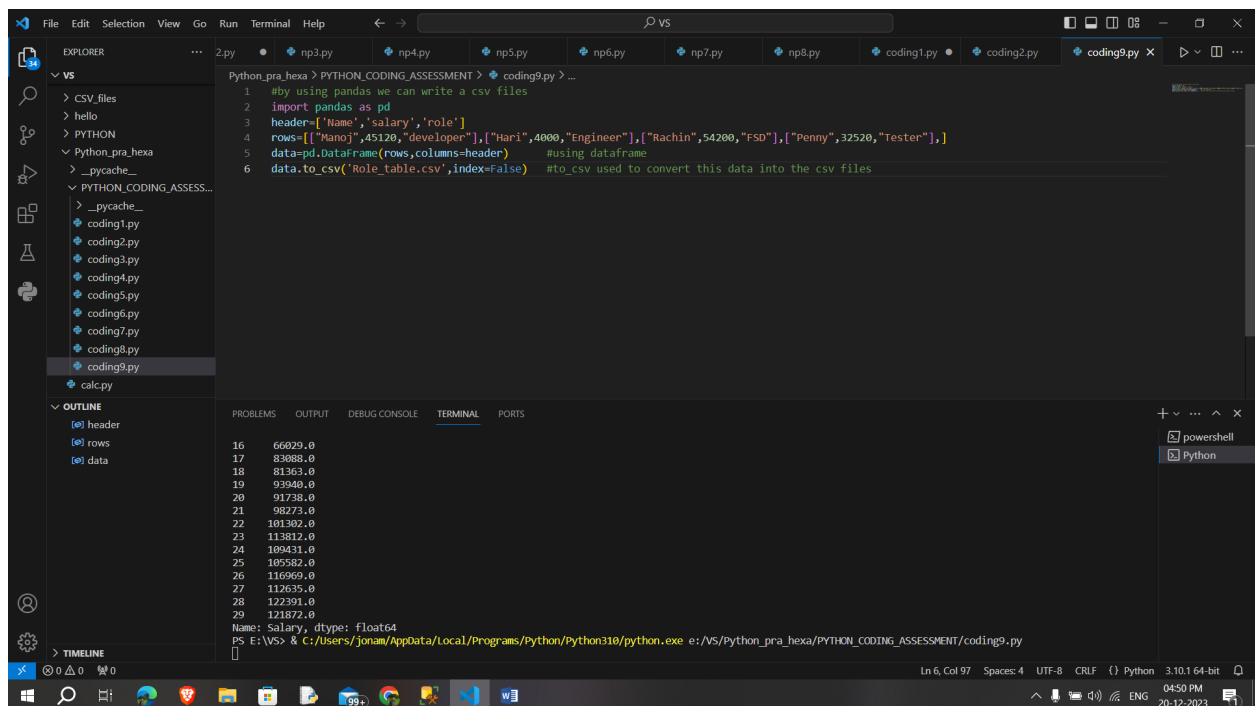


The screenshot shows the VS Code interface with a Python script in the editor. The script reads a CSV file named 'Salary\_Data.csv' and prints its columns and salary data. The terminal output shows the execution of the script, displaying the columns 'YearsExperience' and 'Salary' along with 28 rows of data.

```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding8.py > ...
1 #2.Explain Pandas and numpy using Examples in PYTHON
2 #pandas It is a data manipulation and analysis library. by using pandas we can access csv files read and write
3
4 import pandas as pd #importing pandas
5 data=pd.read_csv('E:\VS\Python_pra_hexa\Salary_Data.csv') #reading the csv file by using read_csv
6 print(data)
7 print(data.columns)
8 print(data.Salary)
```

```
PS E:\VS> & C:\Users\jonam\AppData\Local\Programs\Python\Python310\python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding8.py
YearsExperience Salary
0 1.1 39343.0
1 1.3 46205.0
2 1.5 37731.0
3 2.0 43525.0
4 2.2 39891.0
5 2.9 56642.0
6 3.0 60150.0
7 3.2 54445.0
8 3.2 64445.0
9 3.7 57189.0
10 3.9 63218.0
11 4.0 55794.0
12 4.0 56957.0
13 4.1 57881.0
14 4.5 61111.0
15 4.9 67938.0
16 5.1 66029.0
17 5.3 83088.0
18 5.9 81363.0
19 6.0 93940.0
20 6.8 91738.0
21 7.1 98273.0
22 7.9 101302.0
23 8.2 113812.0
24 8.7 109431.0
25 9.0 105582.0
26 9.5 116969.0
27 9.6 112635.0
28 10.3 122391.0
```

Using pandas writing and creating the csv file,

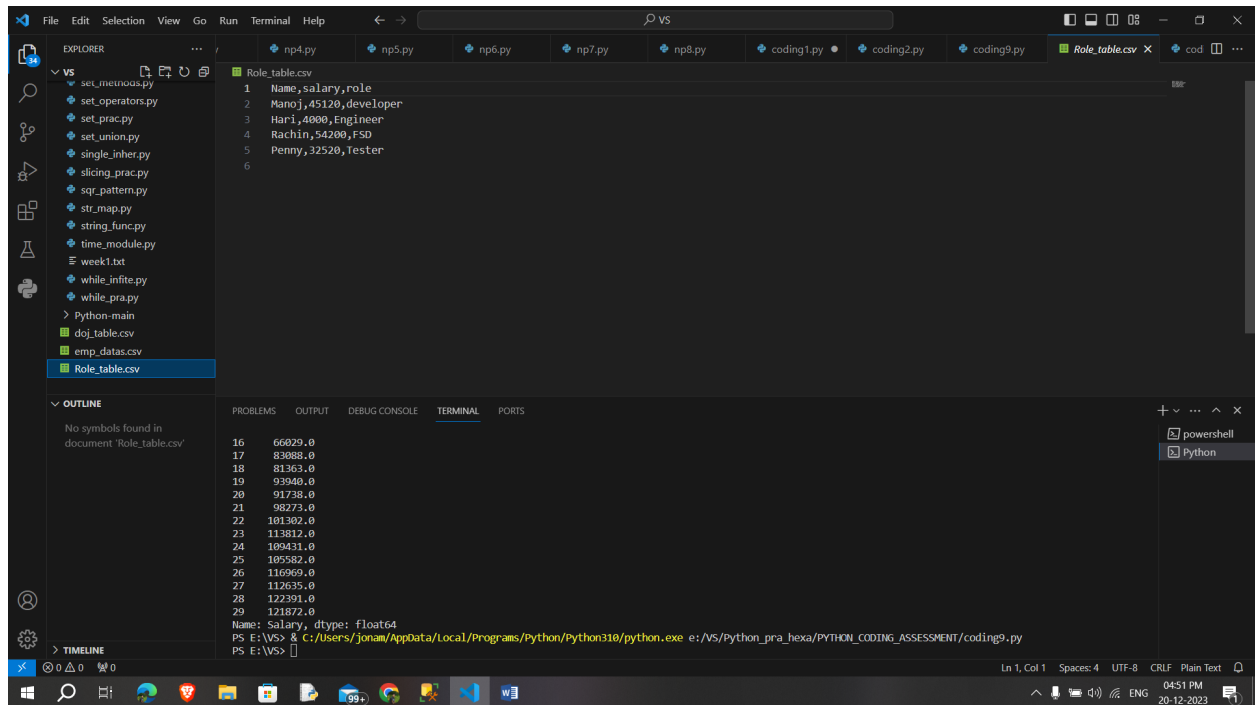


The screenshot shows the VS Code interface with a Python script in the editor. The script creates a DataFrame with employee data and writes it to a CSV file named 'Role\_table.csv'. The terminal output shows the execution of the script, displaying the DataFrame structure and the file path.

```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding9.py > ...
1 #by using pandas we can write a csv files
2 import pandas as pd
3 header=['Name','salary','role']
4 rows=[["Manoj",45120,"developer"],["Hari",4000,"Engineer"],["Rachin",54200,"FSD"],["Penny",32520,"Tester"],]
5 data=pd.DataFrame(rows,columns=header) #using dataframe
6 data.to_csv('Role_table.csv',index=False) #to_csv used to convert this data into the csv files
```

```
PS E:\VS> & C:\Users\jonam\AppData\Local\Programs\Python\Python310\python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding9.py
Name: Salary, dtype: float64
```

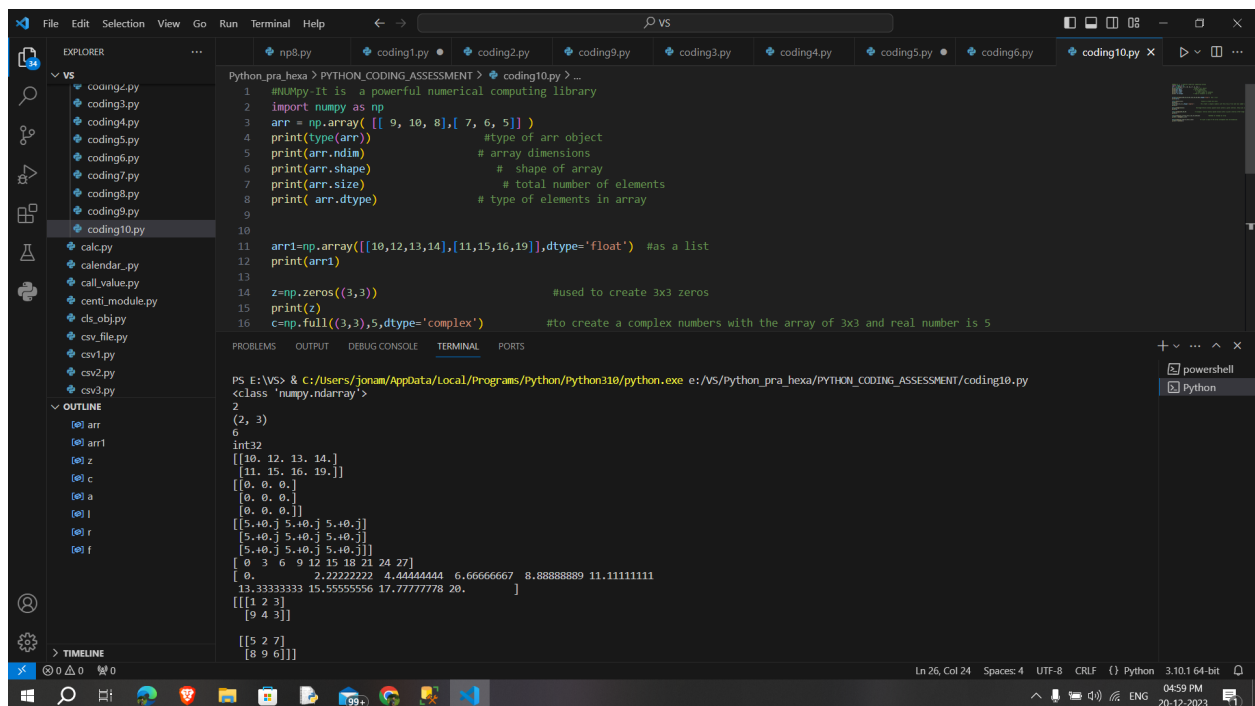
## The created csv file,



```
Role_table.csv
1 Name,salary,role
2 Manoj,45120,developer
3 Hari,4000,Engineer
4 Rachin,54200,FSD
5 Penny,32520,Tester
6

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
16 66029.0
17 83088.0
18 81263.0
19 93040.0
20 91738.0
21 98273.0
22 101302.0
23 113812.0
24 109431.0
25 105582.0
26 116969.0
27 112635.0
28 122391.0
29 121872.0
Name: Salary, dtype: Float64
PS E:\VS> & C:\Users\jonam\AppData\Local\Programs\Python\python310\python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding9.py
PS E:\VS>
```

## Using numpy



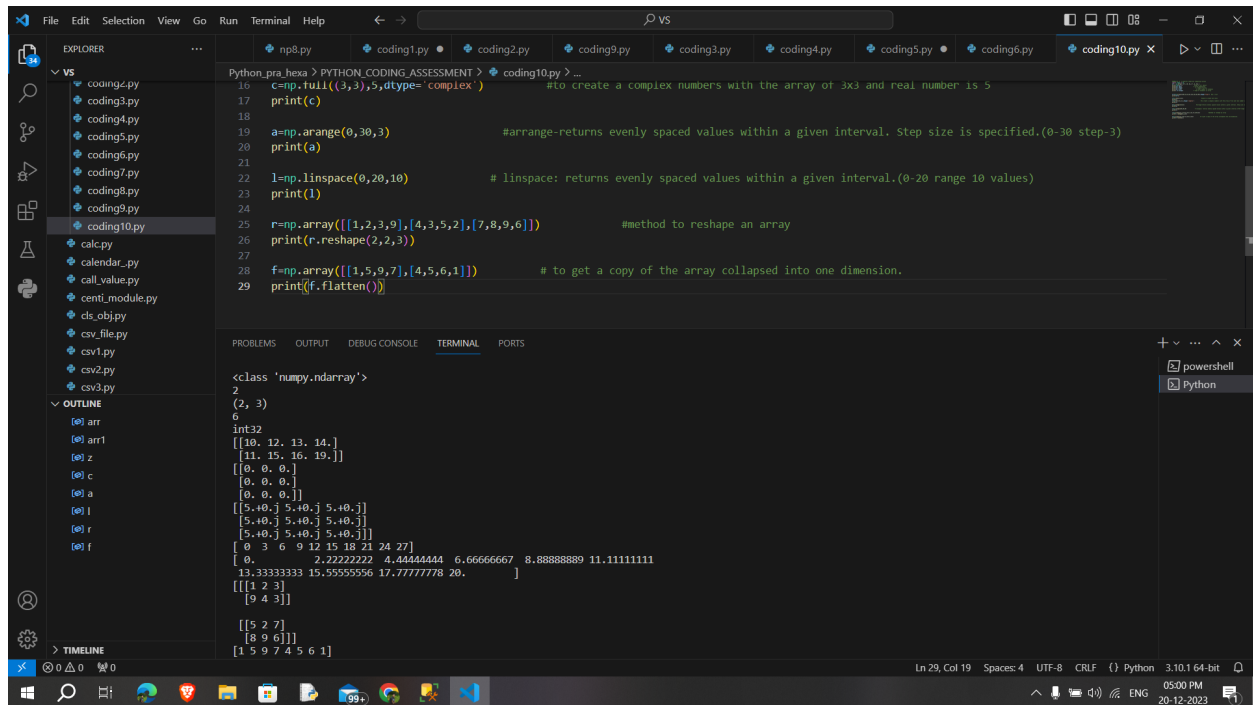
```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding10.py > ...
1 #Numpy-It is a powerful numerical computing library
2 import numpy as np
3 arr = np.array([ [ 9, 10, 8], [ 7, 6, 5] ])
4 print(type(arr)) #type of arr object
5 print(arr.ndim) # array dimensions
6 print(arr.shape) # shape of array
7 print(arr.size) # total number of elements
8 print(arr.dtype) # type of elements in array
9
10
11 arr1=np.array([[10,12,13,14],[11,15,16,19]],dtype='float') #as a list
12 print(arr1)
13
14 z=np.zeros((3,3)) #used to create 3x3 zeros
15 print(z)
16 c=np.full((3,3),5,dtype='complex') #to create a complex numbers with the array of 3x3 and real number is 5

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\VS> & C:\Users\jonam\AppData\Local\Programs\Python\python310\python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding10.py
<class 'numpy.ndarray'>
2
(2, 3)
6
int32
[[10. 12. 13. 14.]
 [11. 15. 16. 19.]]
[0. 0. 0.]
[0. 0. 0.]
[0. 0. 0.]
[[5.+0.j 5.+0.j 5.+0.j]
 [5.+0.j 5.+0.j 5.+0.j]
 [5.+0.j 5.+0.j 5.+0.j]]
[ 0  3  6  9 12 15 18 21 24 27]
[ 0. 2.22222222 4.44444444 6.66666667 8.88888889 11.11111111
 13.33333333 15.55555556 17.77777778 20. ]
[[1 2 3]
 [9 4 3]]

[[5 2]
 [8 9 6]]

Ln 26, Col 24 Spaces: 4 UTF-8 CRLF {} Python 3.10.1 64-bit
04:59 PM 20-12-2023
```

## Continuation,



The screenshot shows a VS Code editor with a Python file named `coding10.py`. The code defines several NumPy arrays and performs operations on them. The terminal output shows the execution results.

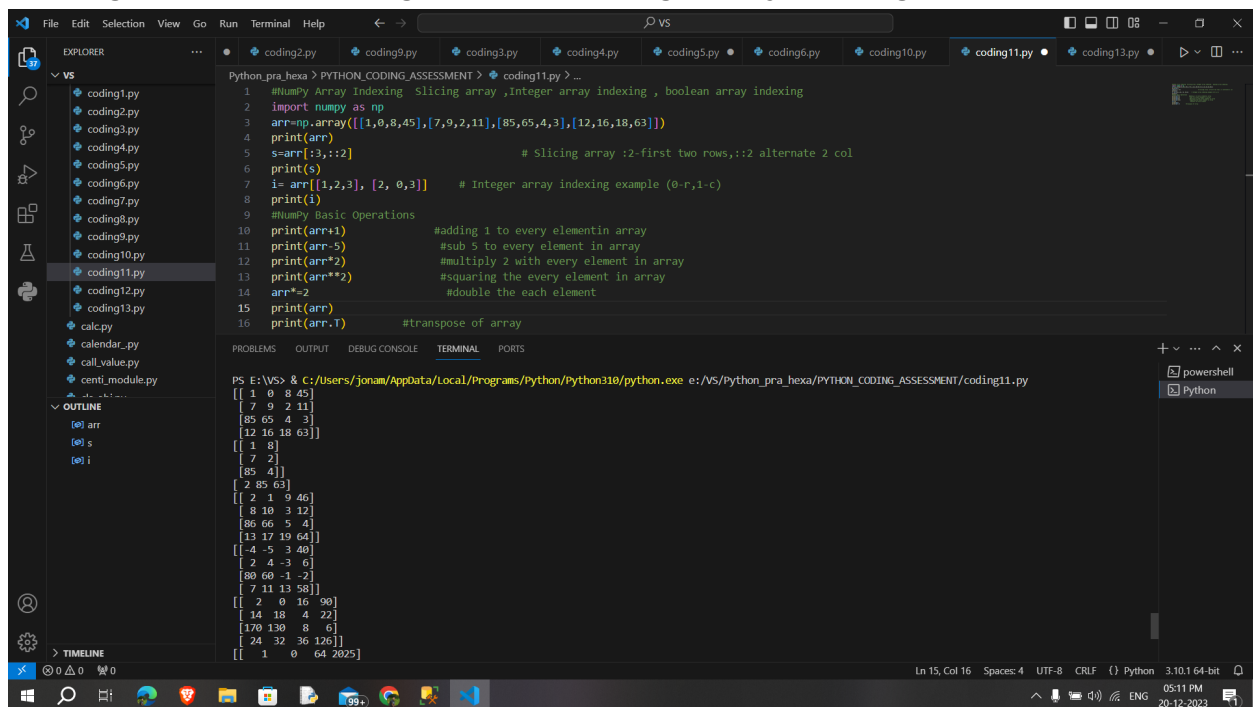
```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding10.py > ...
16 c=np.full((3,3),5,dtype= complex') #to create a complex numbers with the array of 3x3 and real number is 5
17 print(c)
18
19 a=np.arange(0,30,3) #arrange-returns evenly spaced values within a given interval. Step size is specified.(0-30 step-3)
20 print(a)
21
22 l=np.linspace(0,20,10) # linspace: returns evenly spaced values within a given interval.(0-20 range 10 values)
23 print(l)
24
25 r=np.array([[1,2,3,9],[4,3,5,2],[7,8,9,6]]) #method to reshape an array
26 print(r.reshape(2,2,3))
27
28 f=np.array([[1,5,9,7],[4,5,6,1]]) # to get a copy of the array collapsed into one dimension.
29 print(f.flatten())
```

Terminal Output:

```
<class 'numpy.ndarray'>
(2, 3)
6
int32
[[10. 12. 13. 14.]
 [11. 15. 16. 19.]]
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
[[5.+0.j 5.+0.j 5.+0.j]
 [5.+0.j 5.+0.j 5.+0.j]
 [5.+0.j 5.+0.j 5.+0.j]]
[ 0  3  6  9 12 15 18 21 24 27]
[ 0. 2.22222222 4.44444444 6.66666667 8.88888889 11.11111111
 13.33333333 15.55555556 17.77777778 20. ]
[[[1 2 3]
  [9 4 3]]

 [[5 2 7]
  [8 9 6]]]
[1 5 9 7 4 5 6 1]
```

Numpty operations and indexing methods, operations are add,sub,mul,square,double and indexing methods are slicing,boolean and integer array indexing.



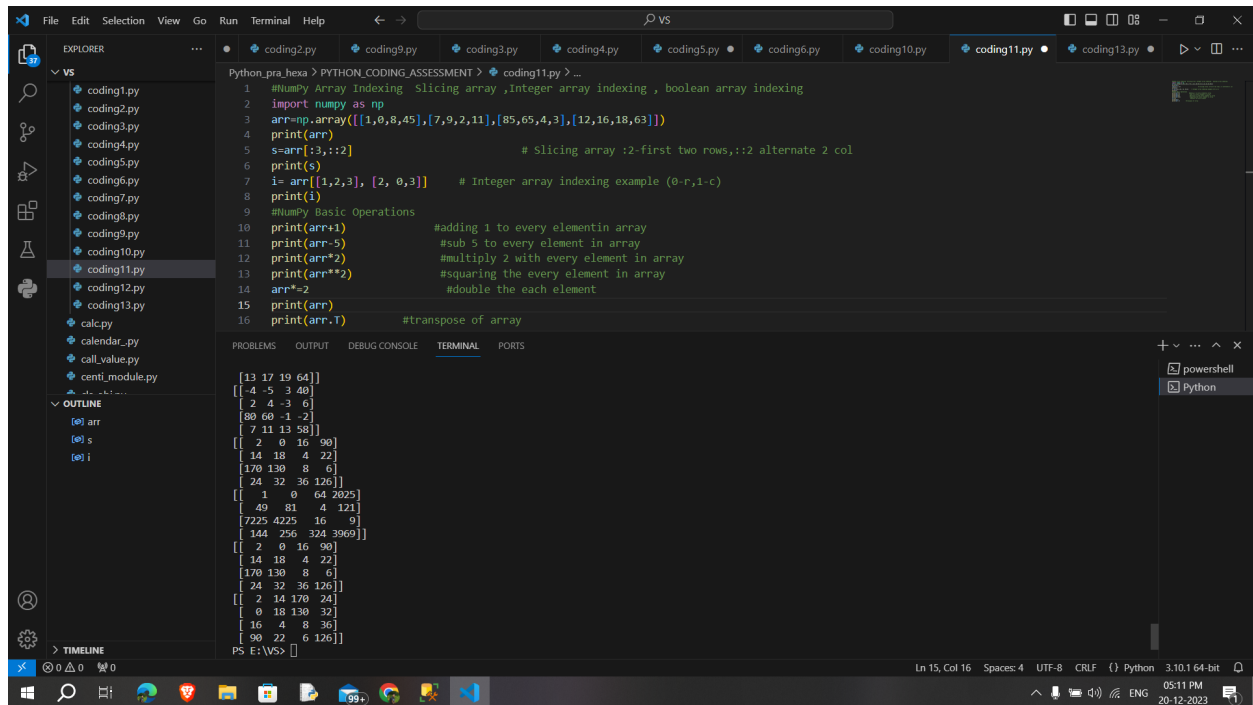
The screenshot shows a VS Code editor with a Python file named `coding11.py`. The code demonstrates various NumPy operations and indexing methods. The terminal output shows the execution results.

```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding11.py > ...
1 #Numpty Array Indexing Slicing array ,Integer array indexing , boolean array indexing
2 import numpy as np
3 arr=np.array([[1,0,8,45],[7,9,2,11],[85,65,4,3],[12,16,18,63]])
4 print(arr)
5 s=arr[:3,:12] # Slicing array :2-first two rows,:2 alternate 2 col
6 print(s)
7 i=arr[[1,2,3], [2, 0,3]] # Integer array indexing example (0-r,1-c)
8 print(i)
9 #Numpty Basic Operations
10 print(arr+1) #adding 1 to every element in array
11 print(arr-5) #sub 5 to every element in array
12 print(arr*2) #multiply 2 with every element in array
13 print(arr**2) #squaring the every element in array
14 arr*=2 #double the each element
15 print(arr)
16 print(arr.T) #transpose of array
```

Terminal Output:

```
PS E:\VS> & C:/Users/jonam/AppData/Local/Programs/Python/Python310/python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding11.py
[[ 1  0  8 45]
 [ 7  9  2 11]
 [85 65  4  3]
 [12 16 18 63]]
[[ 1  8]
 [ 7  2]
 [85  4]]
[[ 2 85 63]
 [ 2  1  9 46]
 [ 8 10  3 12]
 [86 66  5  4]
 [13 17 19 64]]
[[-4 -5  3 40]
 [ 2  4 -3  6]
 [80 60 -1 -2]
 [ 7 11 13 58]]
[[ 2  0 16 90]
 [14 18  4 22]
 [170 130  8  6]
 [ 24 32 36 126]]
[[ 1  0  64 2025]
```

## Continuation,

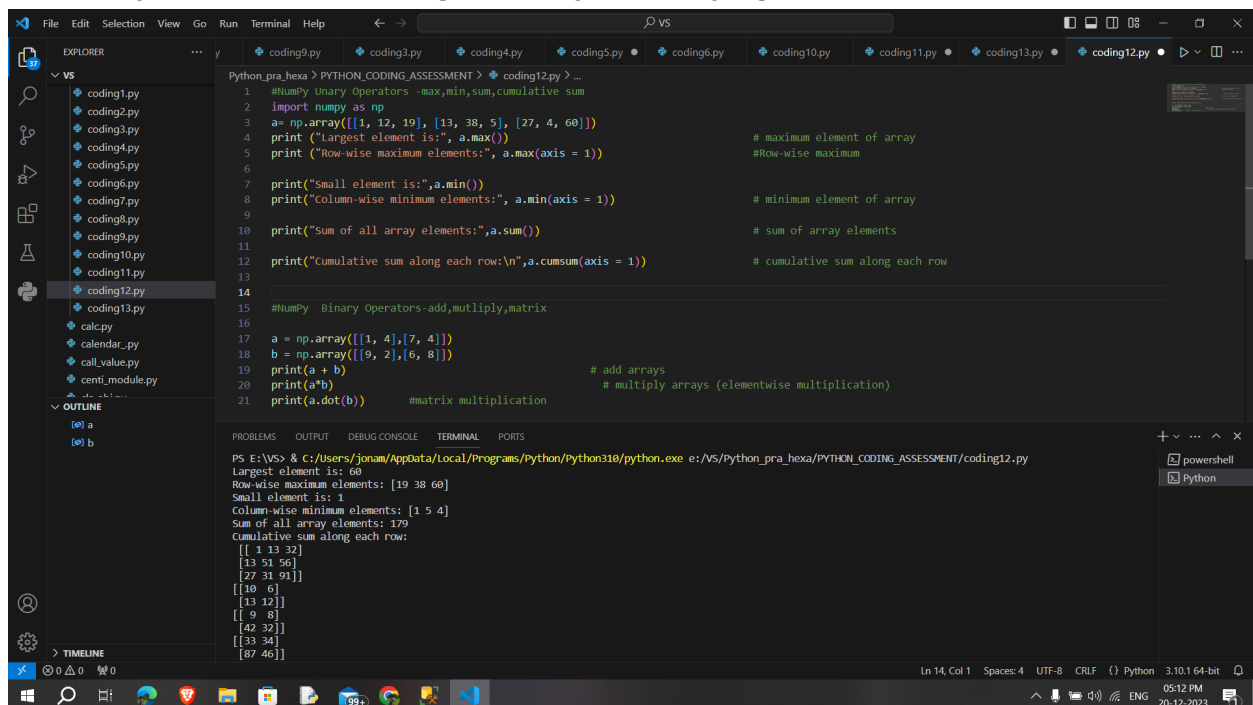


The screenshot shows a Visual Studio Code editor with a Python file named `coding11.py`. The code demonstrates NumPy array indexing and basic operations. The terminal output shows the results of the operations.

```
1 #NumPy Array Indexing Slicing array ,Integer array indexing , boolean array indexing
2 import numpy as np
3 arr=np.array([[1,0,8,45],[7,9,2,11],[85,65,4,3],[12,16,18,63]])
4 print(arr)
5 s=arr[:3,:2] # Slicing array :2-first two rows,:2 alternate 2 col
6 print(s)
7 i= arr[[1,2,3], [2, 0,3]] # Integer array indexing example (0-r,1-c)
8 print(i)
9 #NumPy Basic Operations
10 print(arr+1) #adding 1 to every element in array
11 print(arr-5) #sub 5 to every element in array
12 print(arr*2) #multiply 2 with every element in array
13 print(arr**2) #squaring the every element in array
14 arr*=2 #double the each element
15 print(arr)
16 print(arr.T) #transpose of array
```

```
[13 17 19 64]
[[ 4 -5  3 48]
 [ 2  4 -3  6]
 [80 60 -1 -2]
 [ 7 11 13 58]]
[[ 2  0 16 90]
 [14 18  4 22]
 [170 130  8  6]
 [24 32 36 126]]
[[ 1  0  64 2025]
 [49 81  4 121]
 [7225 4225 16  9]
 [144 256 324 3609]]
[[ 2  0 16 90]
 [14 18  4 22]
 [170 130  8  6]
 [24 32 36 126]]
[[ 2 14 170 24]
 [ 0 18 130 32]
 [16  4  8 36]
 [90 22  6 126]]
PS E:\VS>
```

NumPy unary and binary operators, unary operators are max,min,sum,cumulative sum and binary operators are adding two arrays,multiplying and matrix multiplication.



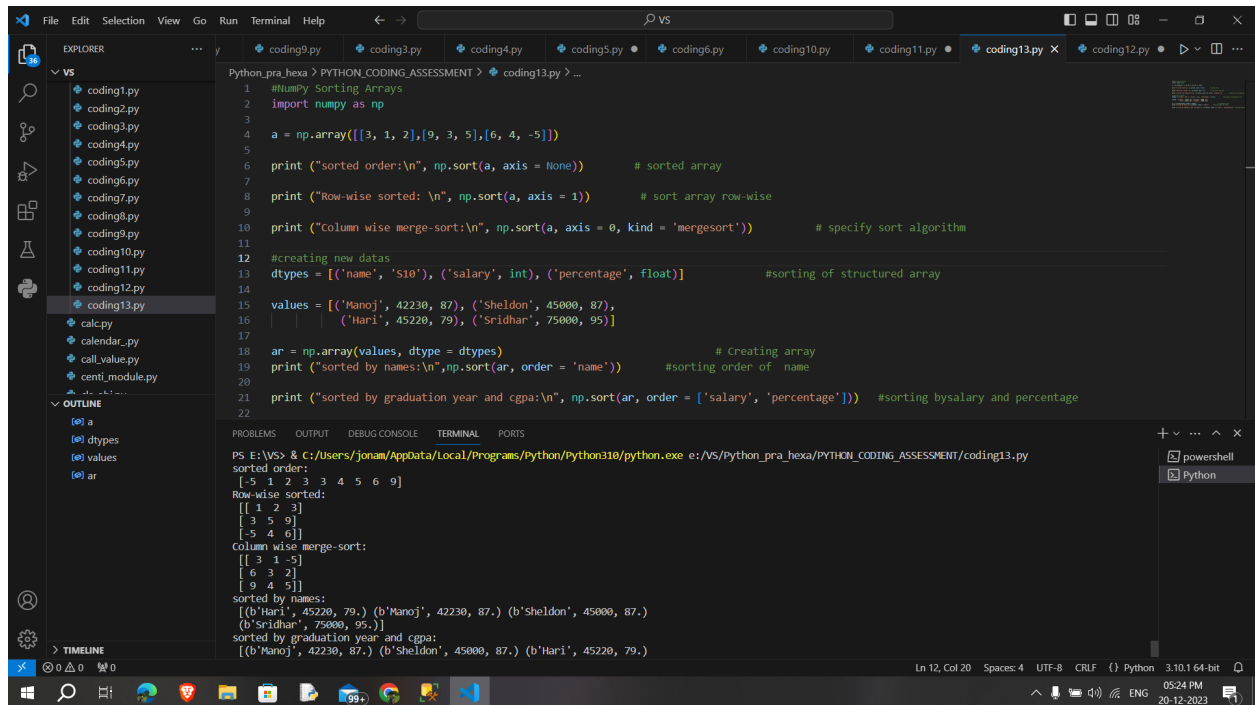
The screenshot shows a Visual Studio Code editor with a Python file named `coding12.py`. The code demonstrates NumPy unary and binary operators. The terminal output shows the results of the operations.

```
1 #NumPy Unary Operators -max,min,sum,cumulative sum
2 import numpy as np
3 a= np.array([[1, 12, 19], [13, 38, 5], [27, 4, 60]])
4 print ("Largest element is:", a.max()) # maximum element of array
5 print ("Row-wise maximum elements:", a.max(axis = 1)) #Row-wise maximum
6
7 print("Small element is:",a.min())
8 print("Column-wise minimum elements:", a.min(axis = 1)) # minimum element of array
9
10 print("Sum of all array elements:",a.sum()) # sum of array elements
11
12 print("Cumulative sum along each row:\n",a.cumsum(axis = 1)) # cumulative sum along each row
13
14
15 #NumPy Binary Operators-add,multiply,matrix
16
17 a = np.array([[1, 4],[7, 4]])
18 b = np.array([[9, 2],[6, 8]])
19 print(a + b) # add arrays
20 print(a*b) # multiply arrays (elementwise multiplication)
21 print(a.dot(b)) #matrix multiplication
```

```
PS E:\VS> & C:\Users\jonam\AppData\Local\Programs\Python\Python310\python.exe e:\VS\Python_pra_hexa\PYTHON_CODING_ASSESSMENT\coding12.py
Largest element is: 60
Row-wise maximum elements: [19 38 60]
Small element is: 1
Column-wise minimum elements: [1 5 4]
Sum of all array elements: 179
Cumulative sum along each row:
[[ 1 13 32]
 [13 51 56]
 [27 31 91]]
[[10  6]
 [13 12]]
[[ 9  8]
 [42 32]]
[[33 24]
 [87 46]]
```



## Numpy sorting methods,



```
Python_pra_hexa > PYTHON_CODING_ASSESSMENT > coding13.py > ...
1  #Numpy Sorting Arrays
2  import numpy as np
3
4  a = np.array([[3, 1, 2],[9, 3, 5],[6, 4, -5]])
5
6  print ("sorted order:\n", np.sort(a, axis = None))      # sorted array
7
8  print ("Row-wise sorted: \n", np.sort(a, axis = 1))      # sort array row-wise
9
10 print ("Column wise merge-sort:\n", np.sort(a, axis = 0, kind = 'mergesort'))      # specify sort algorithm
11
12 #creating new datas
13 dtypes = [('name', 'S10'), ('salary', int), ('percentage', float)]      #sorting of structured array
14
15 values = [('Manoj', 42230, 87), ('Sheldon', 45000, 87),
16           ('Harli', 45220, 79), ('Sridhar', 75000, 95)]
17
18 ar = np.array(values, dtype = dtypes)      # Creating array
19 print ("sorted by names:\n", np.sort(ar, order = 'name'))      #sorting order of name
20
21 print ("sorted by graduation year and cgpa:\n", np.sort(ar, order = ['salary', 'percentage']))      #sorting by salary and percentage
22
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS E:\VS> & C:\Users\jonam\AppData\Local\Programs\Python\Python310\python.exe e:/VS/Python_pra_hexa/PYTHON_CODING_ASSESSMENT/coding13.py
sorted order:
[-5  1  2  3  3  4  5  6  9]
Row-wise sorted:
[[ 1  2  3]
 [ 3  5  9]
 [-5  4  6]]
Column wise merge-sort:
[[ 3  1 -5]
 [ 6  3  2]
 [ 9  4  5]]
sorted by names:
[(b'Harli', 45220, 79.) (b'Manoj', 42230, 87.) (b'Sheldon', 45000, 87.)
 (b'Sridhar', 75000, 95.)]
sorted by graduation year and cgpa:
[(b'Manoj', 42230, 87.) (b'Sheldon', 45000, 87.) (b'Harli', 45220, 79.)]
```

Ln 12, Col 20 Spaces: 4 UTF-8 CRLF Python 3.10.1 64-bit

05:24 PM 20-12-2023