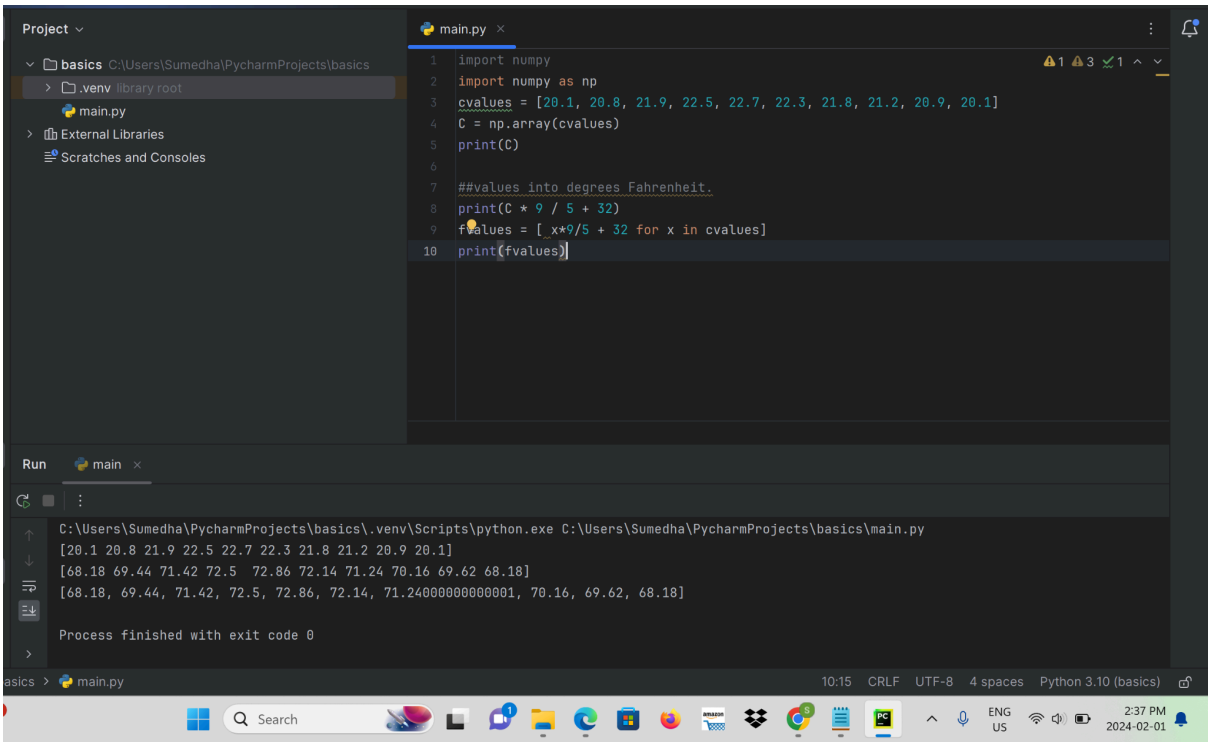
NAME : AKULA SHARATH CHANDRA
BATCH : DATA ENGINEERING
DATE : 01-02-24
TOPIC : NUMPY,CREATE DATAFRAMES USING DYNAMIC COLUMN LIST ON CSV DATA,PERFORMING JOINS BETWEEN PANDA DATA FRAMES,GET UNIQUE VALUES FROM A LIST IN PYTHON,GET COUNT BY STATUS USING PANDAS DATAFRAME APIS

1)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.



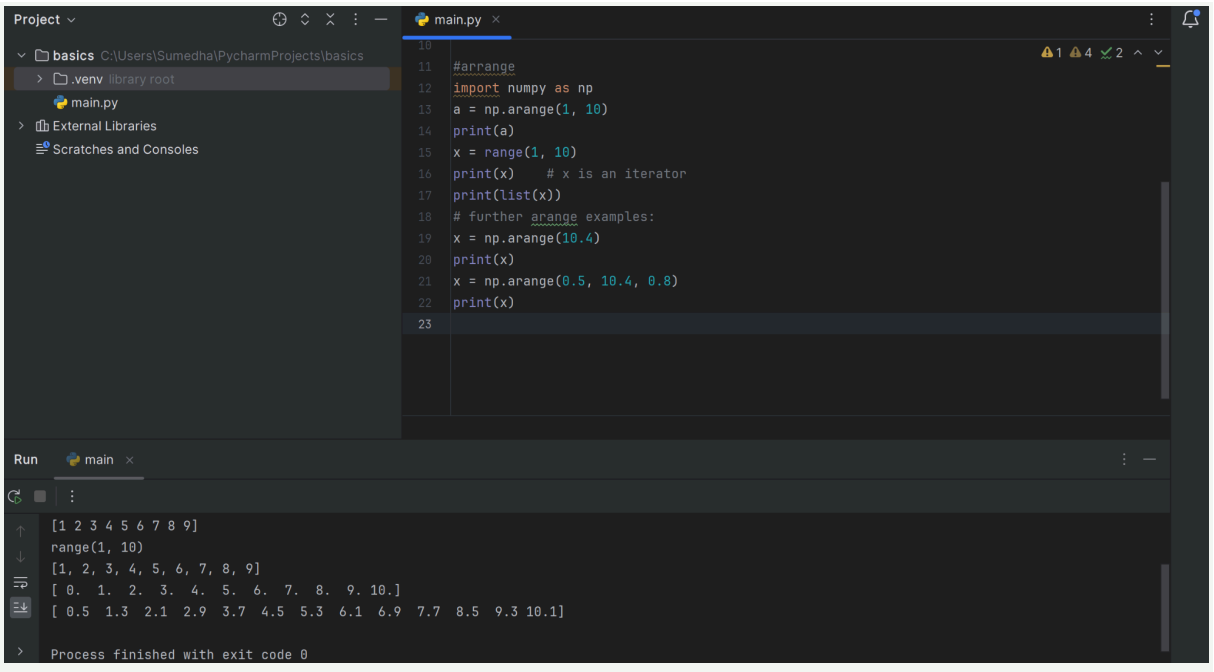
2)numpy.ndarray: There are functions provided by Numpy to create arrays with evenly spaced values within a given interval. One 'arange' uses a given distance and the other one 'linspace' needs the number of elements and creates the distance automatically.

—>Creation of Arrays with Evenly Spaced Values

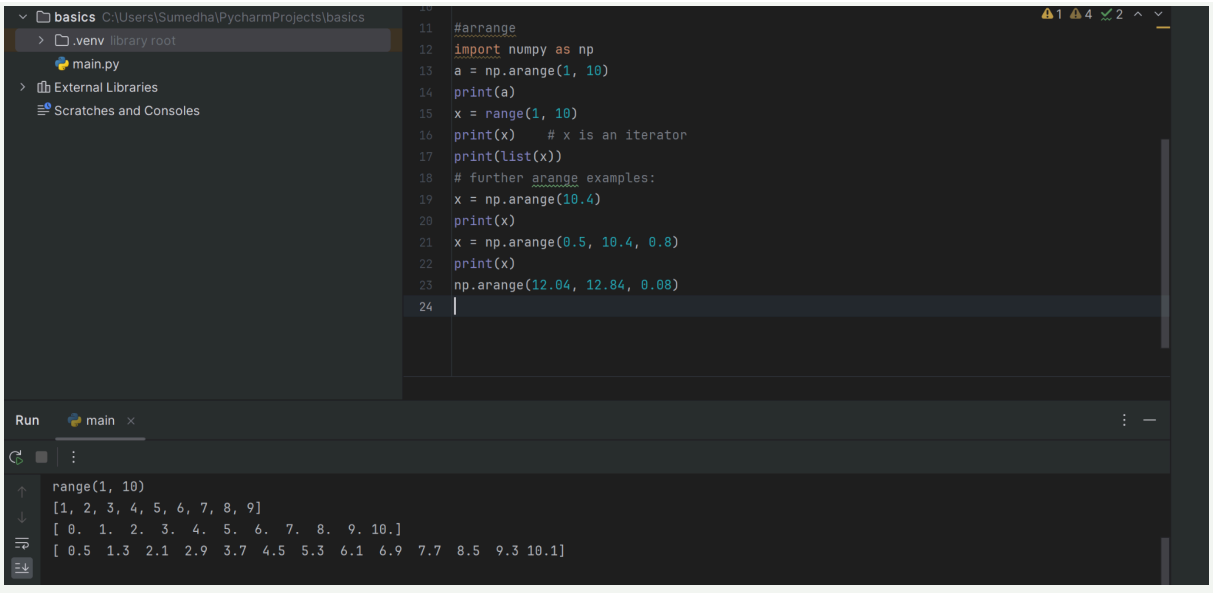
i)Arrange :

The syntax of arange:

`arange([start,] stop[, step], [, dtype=None])`



if we use a float value for the step parameter, as you can see in the following example:

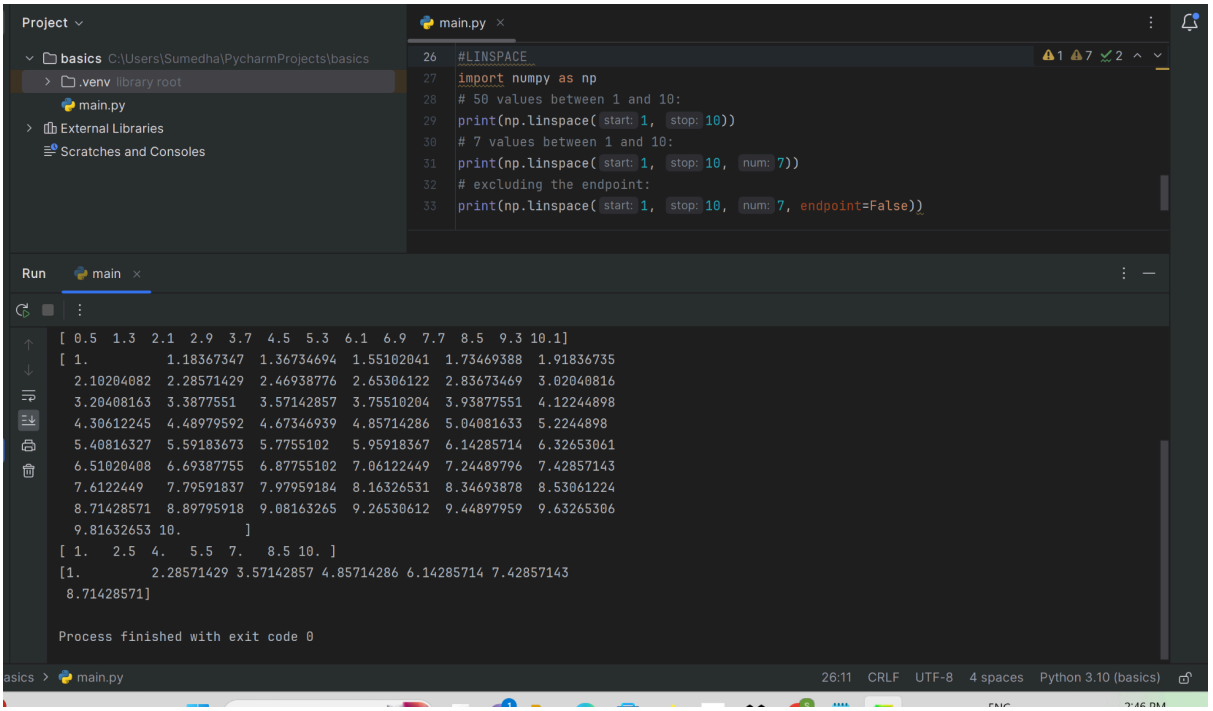


ii) linspace : The syntax of linspace:

linspace(start, stop, num=50, endpoint=True, retstep=False)

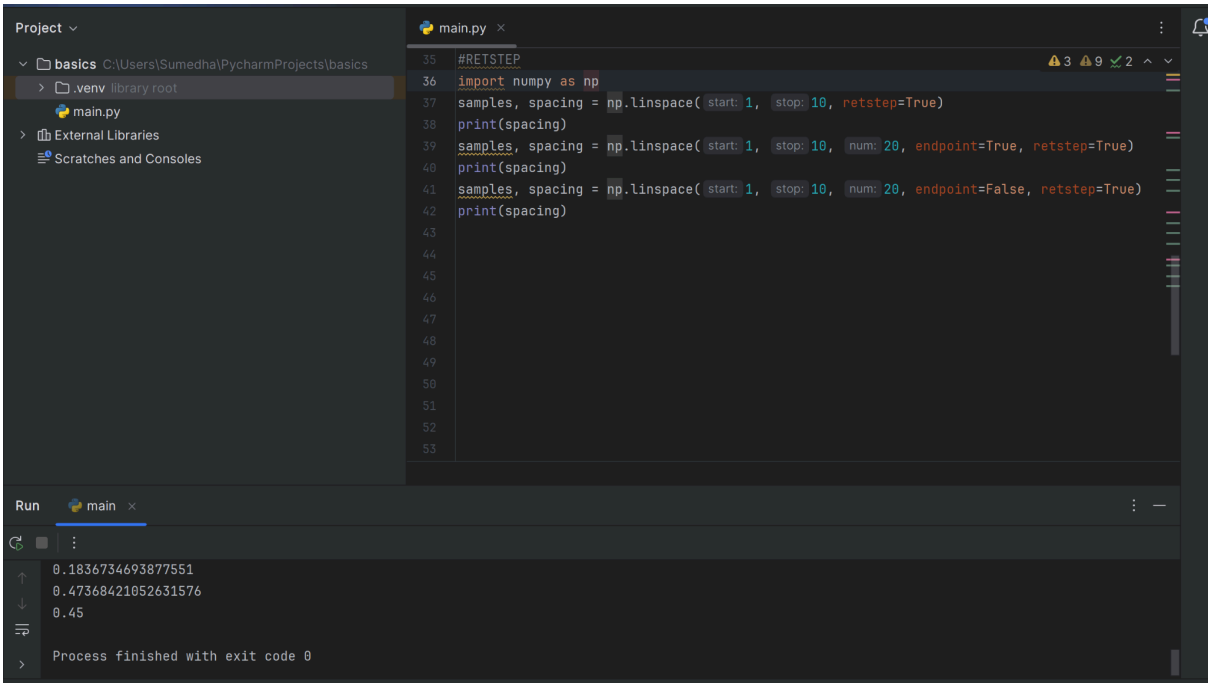
linspace returns an ndarray, consisting of 'num' equally spaced samples in the closed interval [start, stop] or the half-open interval [start, stop).

example:



iii) Retstep: The retstep parameter in the np.linspace function determines whether to return the step size along with the array of values. If retstep is set to true, the function returns a tuple containing the array of values and the step size. If retstep is set to false (default), only the array of values is returned.

Example:

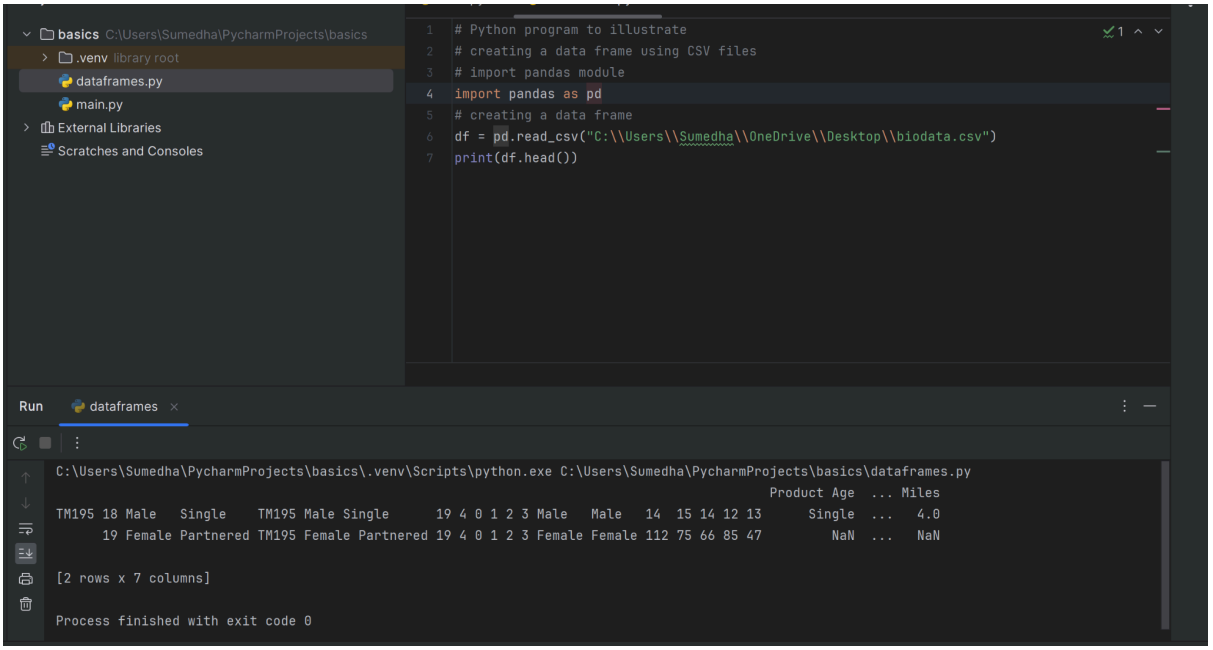


3)CREATE DATAFRAMES USING DYNAMIC COLUMN LIST ON CSV DATA:

Creating a pandas data frame using CSV files can be achieved in multiple ways.

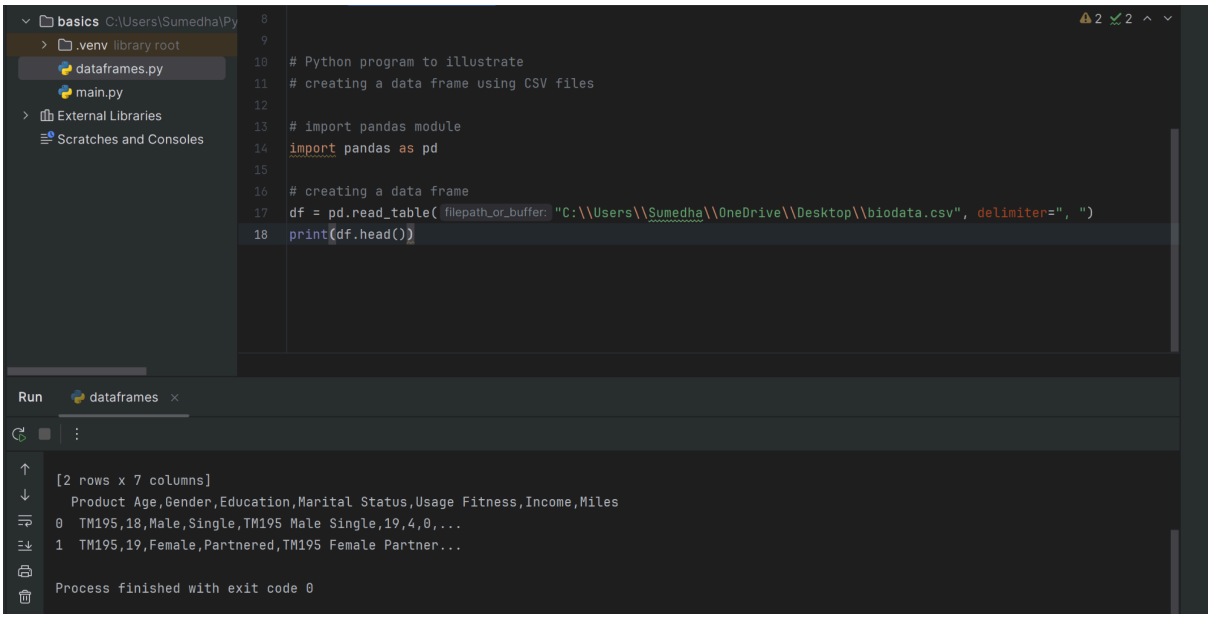
Method #1: Using **read_csv()** method: **read_csv()** is an important pandas function to read csv files and do operations on it.

Example:



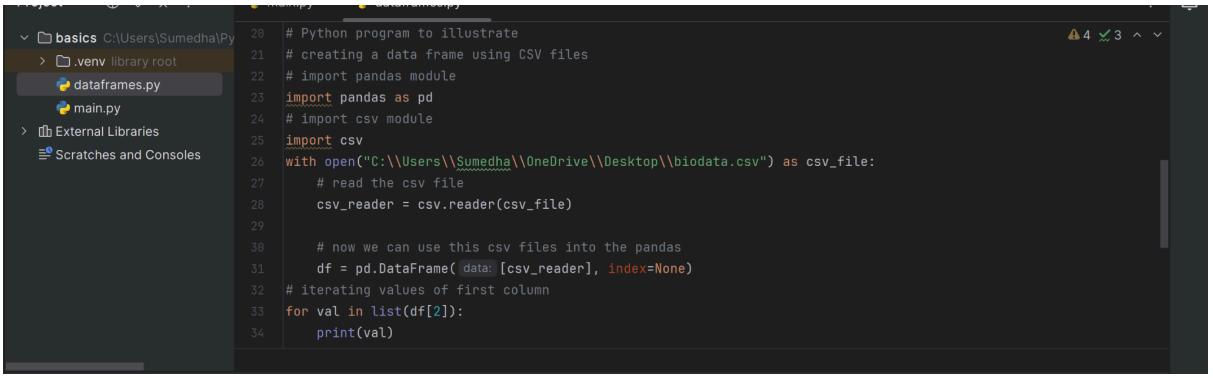
Method #2: Using **read_table()** method: **read_table()** is another important pandas function to read csv files and create data frame from it.

Example:

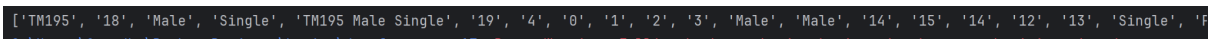


Method #3: Using the **csv** module: One can directly import the csv files using the csv module and then create a data frame using that csv file.

Example:



output:



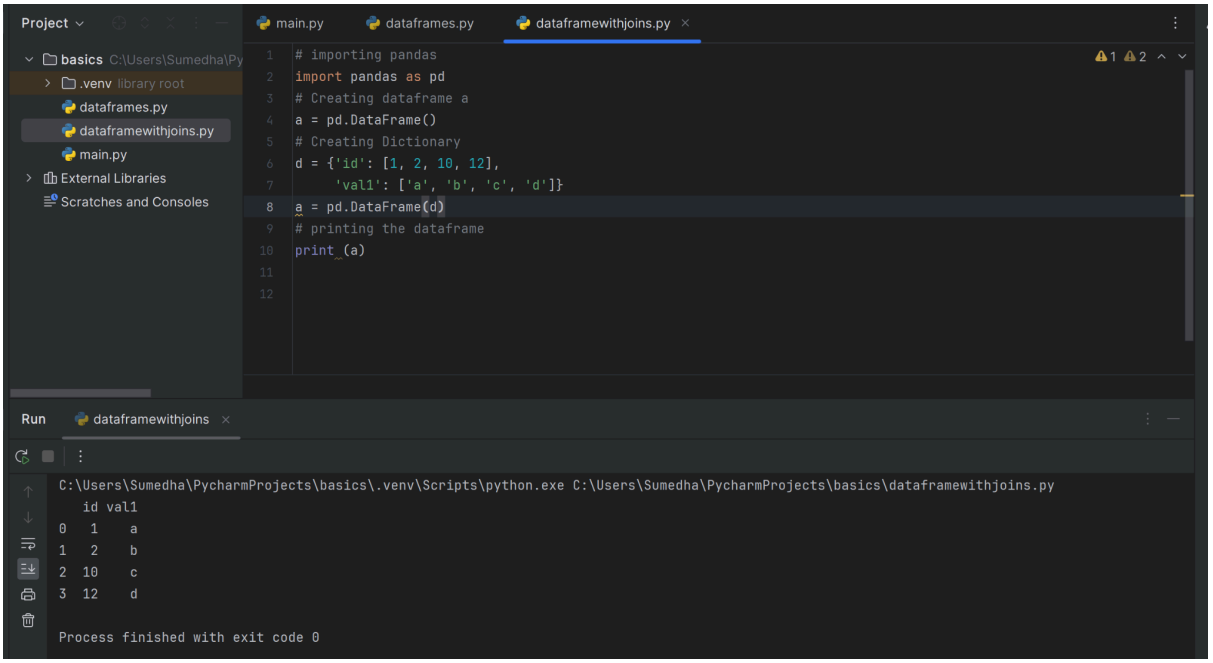
4)PERFORMING JOINS BETWEEN PANDA DATA FRAMES:

—>There are five types of Joins in pandas

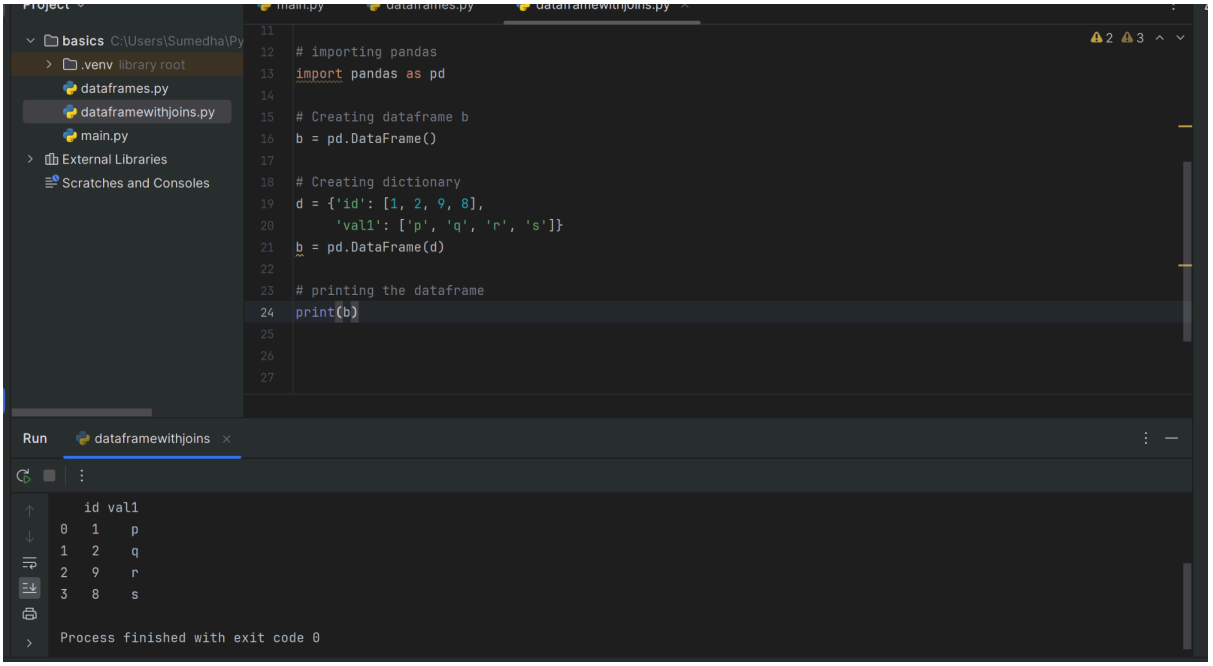
- Inner Join
- Left Outer Join
- Right Outer Join
- Full Outer Join or simply Outer Join
- Index Join

To understand different types of joins, we will first make two DataFrames, namely **a** and **b**.

Dataframe a:



DataFrame b:



—>Pandas Inner Join

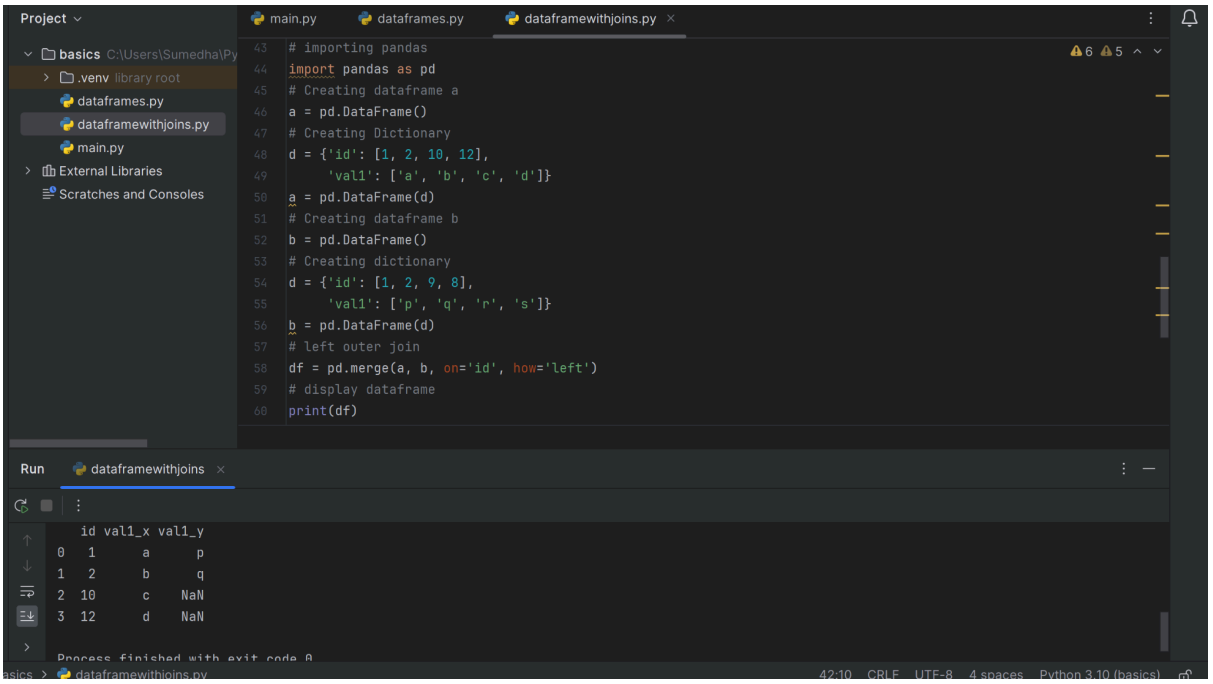
Inner join is the most common type of join you'll be working with. It returns a Dataframe with only those rows that have common characteristics.

Example:

—>**Pandas Left Join**

With a left outer join, all the records from the first Dataframe will be displayed, irrespective of whether the keys in the first Dataframe can be found in the second Dataframe. Whereas, for the second Dataframe, only the records with the keys in the second Dataframe that can be found in the first Dataframe will be displayed.

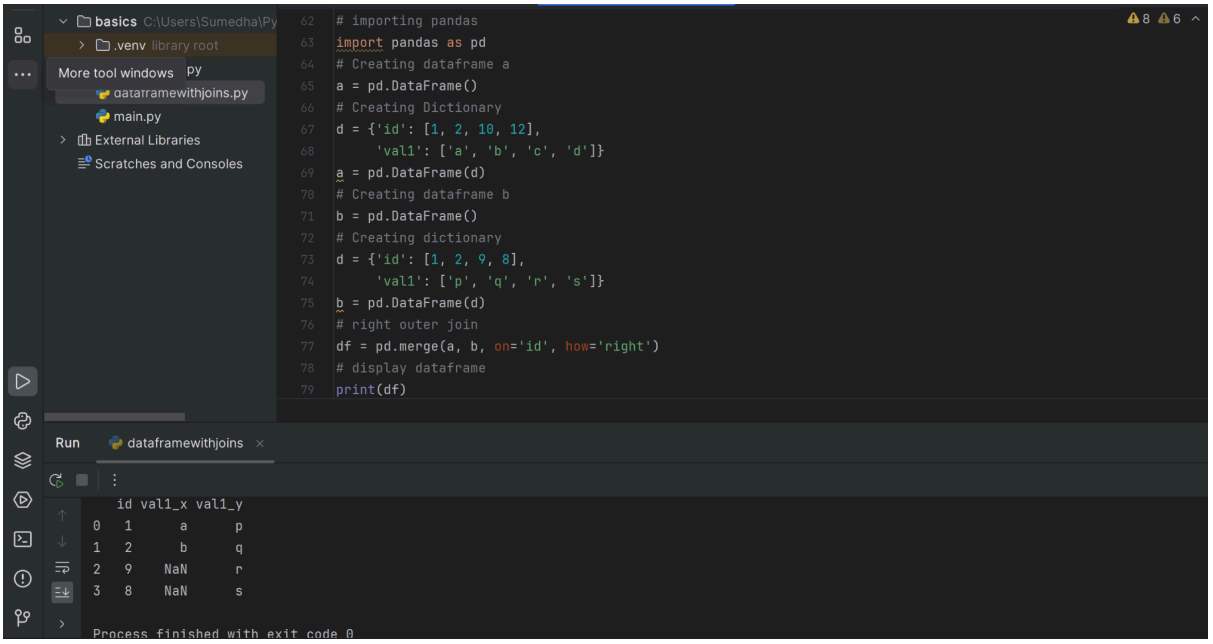
Example:



—>Pandas Right Outer Join

For a right join, all the records from the second Dataframe will be displayed. However, only the records with the keys in the first Dataframe that can be found in the second Dataframe will be displayed.

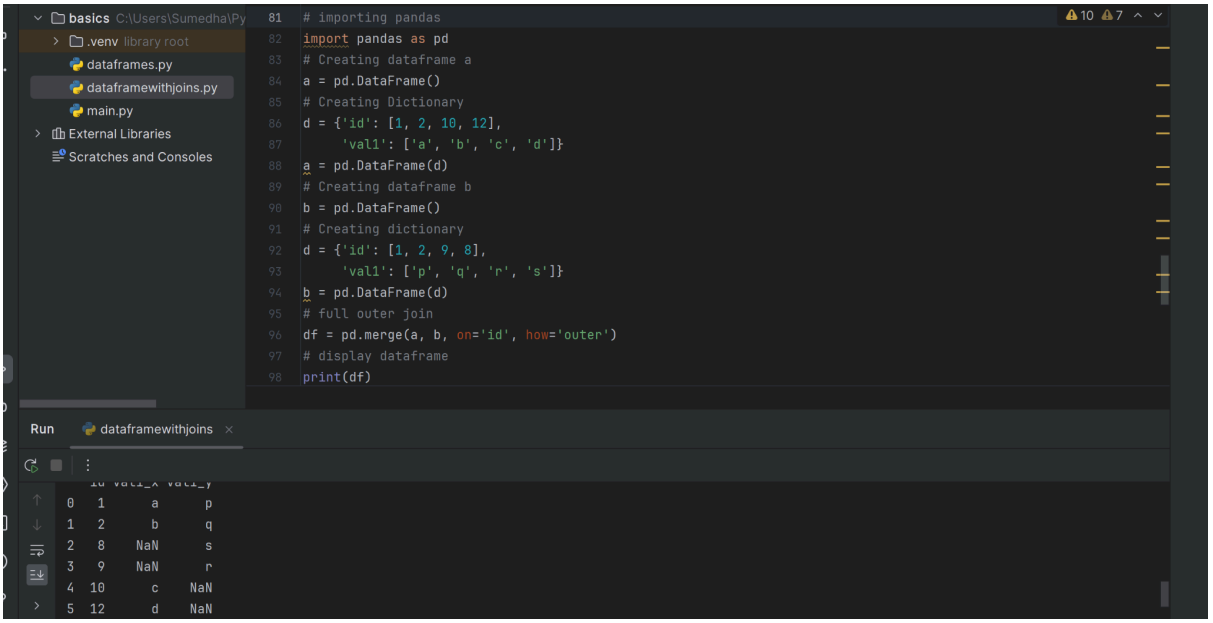
Example:



—>Pandas Full Outer Join

A full outer join returns all the rows from the left Dataframe, and all the rows from the right Dataframe, and matches up rows where possible, with NaNs elsewhere.

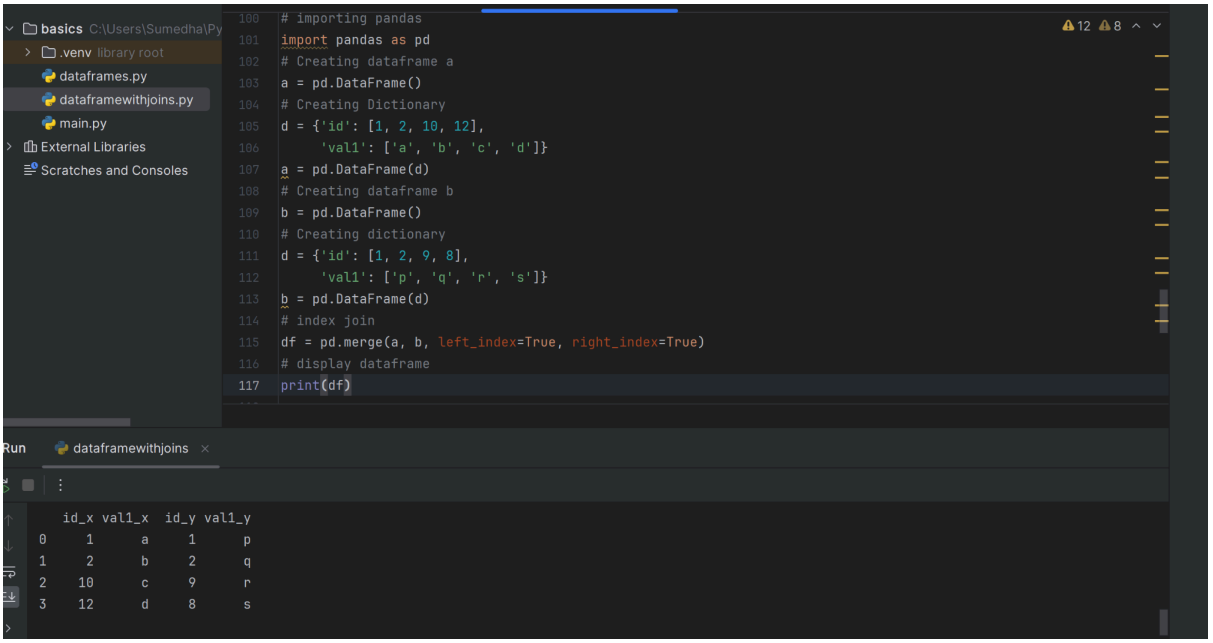
Example:



—>Pandas Index Join

To merge the Dataframe on indices pass the *left_index* and *right_index* arguments as True i.e. both the Dataframes are merged on an index using default Inner Join.

Example:

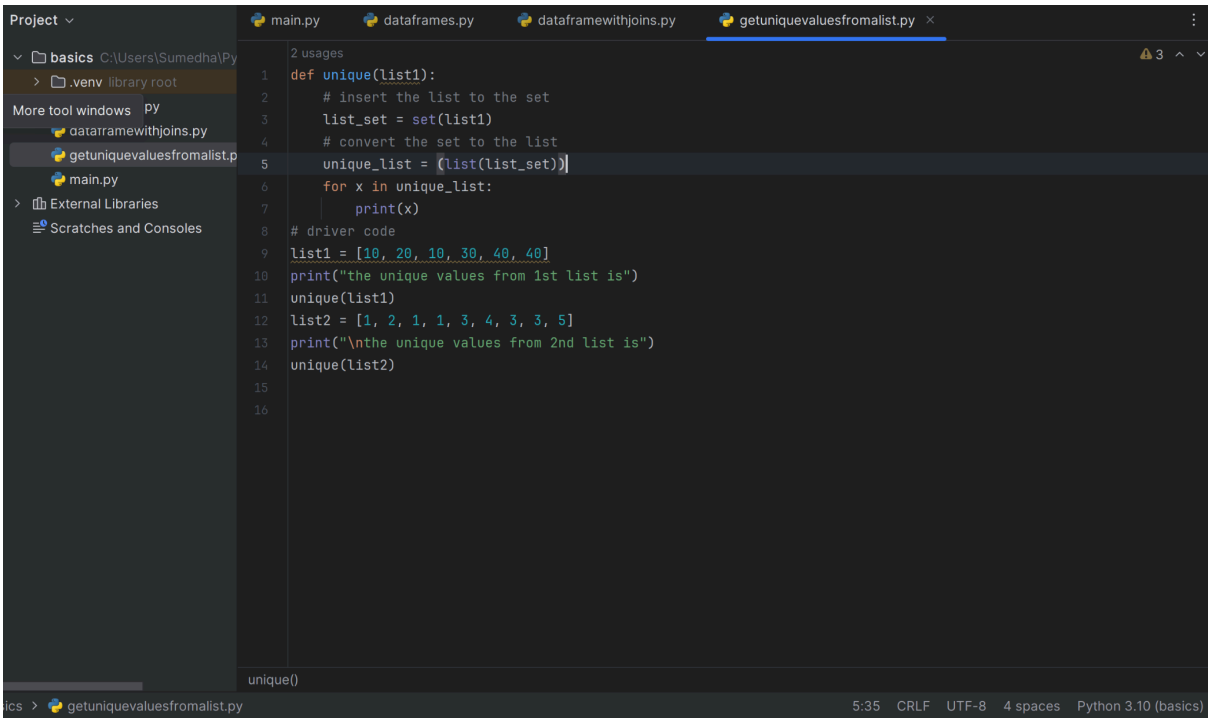


5)Get Unique Values from a List in Python:

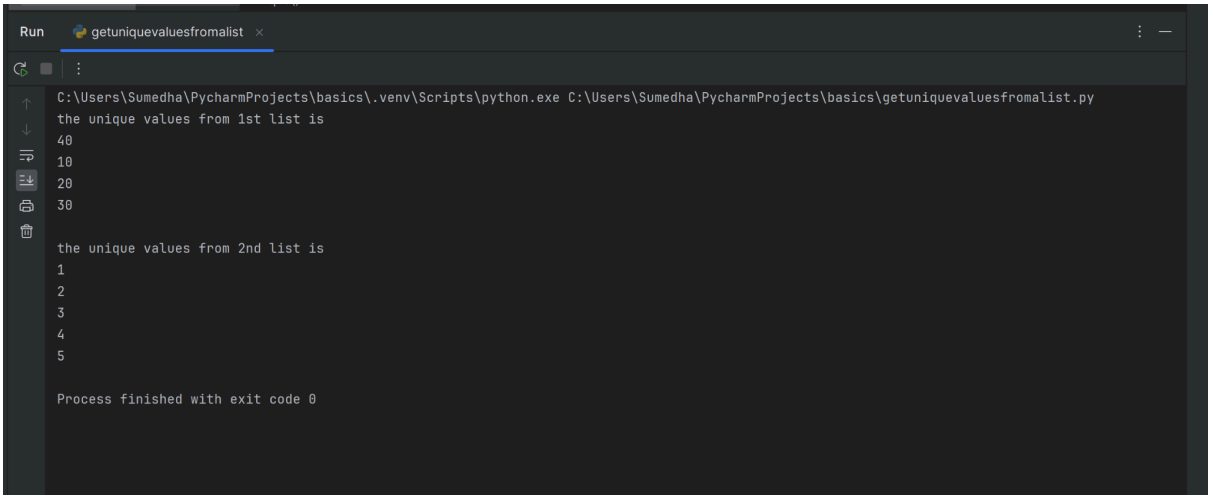
Get Unique Values from a List Using Set Method

Using set() property of Python, we can easily check for the unique values. Insert the values of the list in a set. Set only stores a value once even if it is inserted more than once. After inserting all the values in the set by list_set=set(list1), convert this set to a list to print it.

Example:

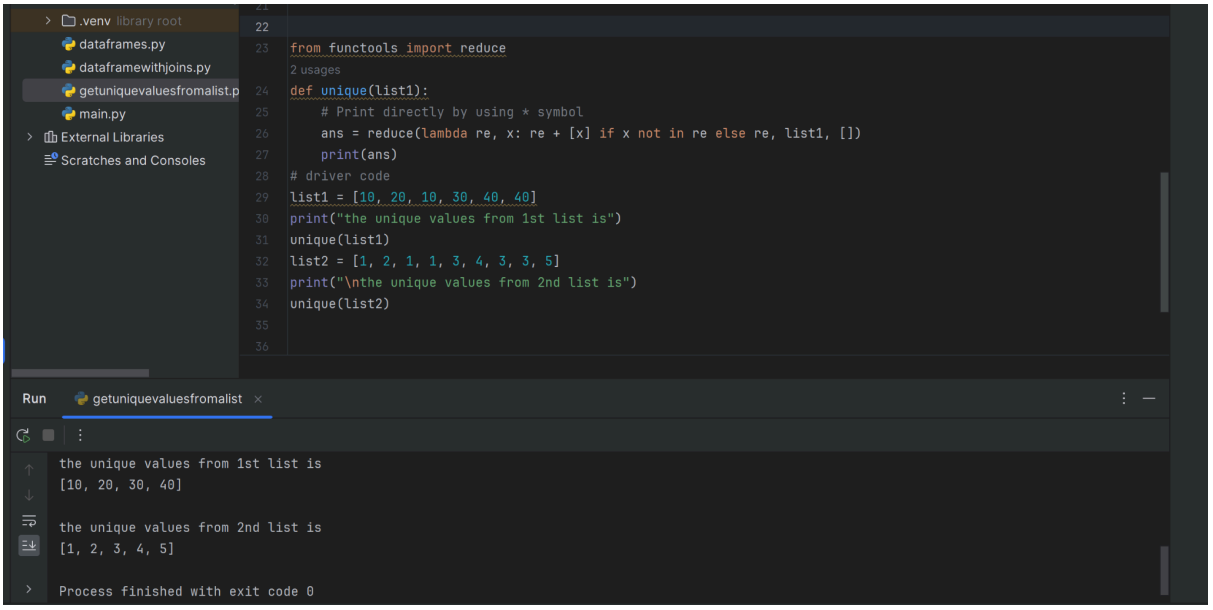


Output:



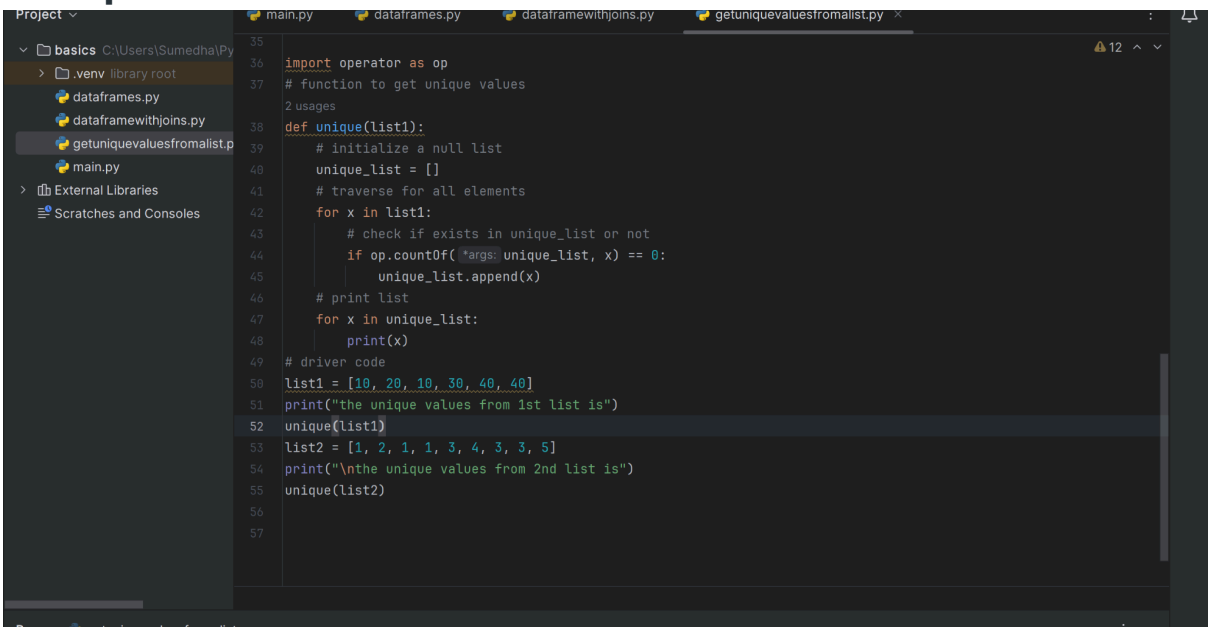
—>**Get Unique Values From a List in Python Using reduce() function:**Using Python import reduce() from functools and iterate over all element and checks if the element is a duplicate or unique value.

Example:

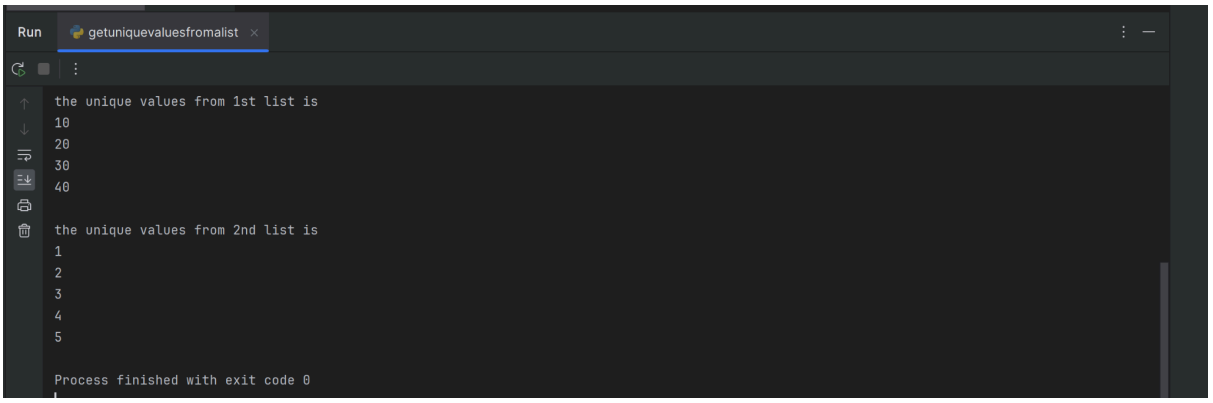


—>**Get Unique Values From a List in Python Using Operator.countOf() method:**The ‘unique’ function initializes an empty ‘unique_list’, then iterates through ‘list1’. For each element ‘x’, it employs ‘op.countof()’ to check if ‘x’ is present in ‘unique_list’. If not found (count is 0), ‘x’ is appended to ‘unique_list’. The final unique values are printed using a loop.

example:

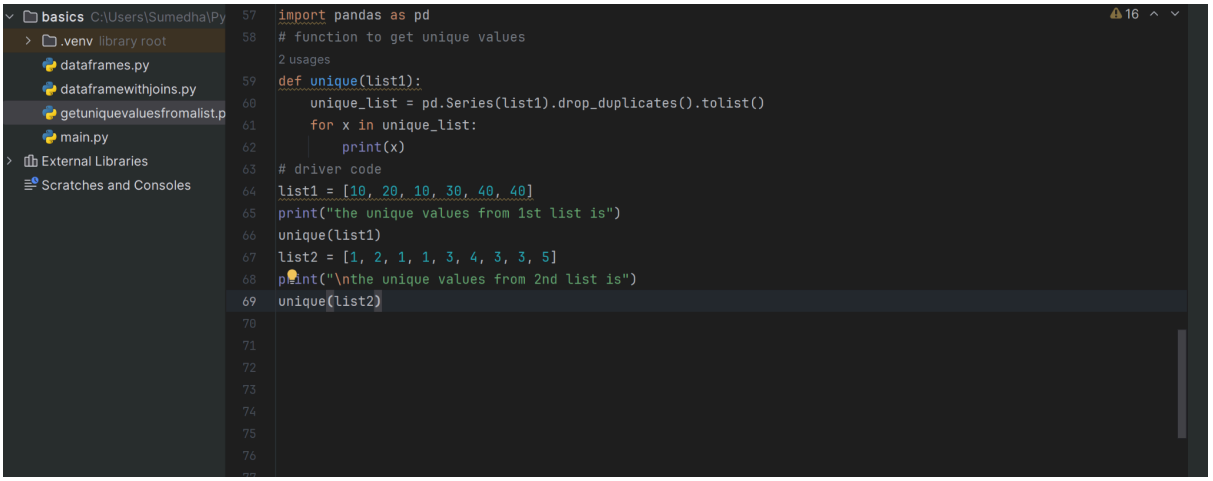


Output:

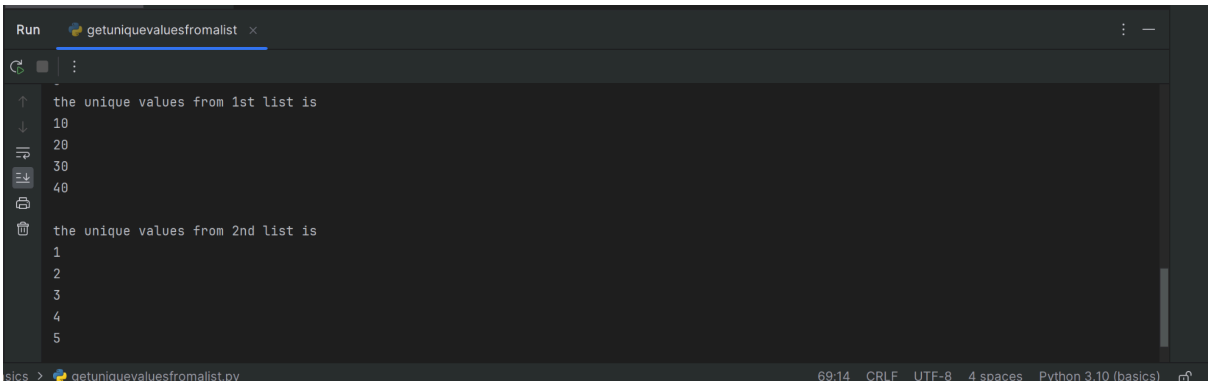


—>Get Unique Values From a List in Python Using pandas module: The ‘unique’ function utilize pandas() to create a Series from ‘list1’, then employs ‘drop_duplicates()’ to eliminate duplicates and obtain a list of unique values. Subsequently, it iterates through the unique list and prints each element. The driver code demonstrates the process for two lists, ‘list1’ and ‘list2’, providing distinct values for each list.

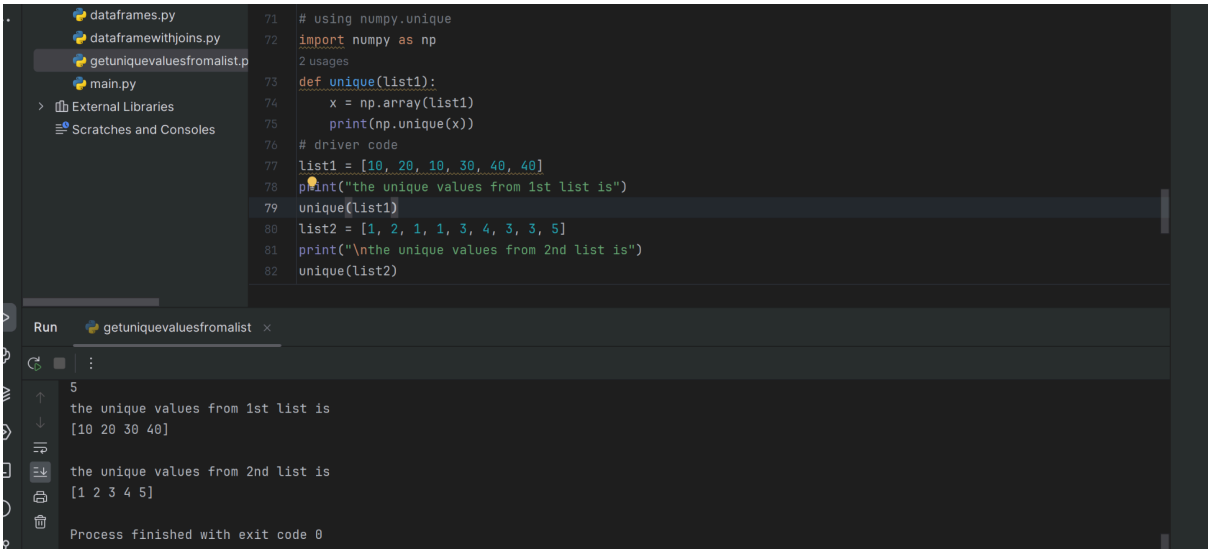
Example:



Output:

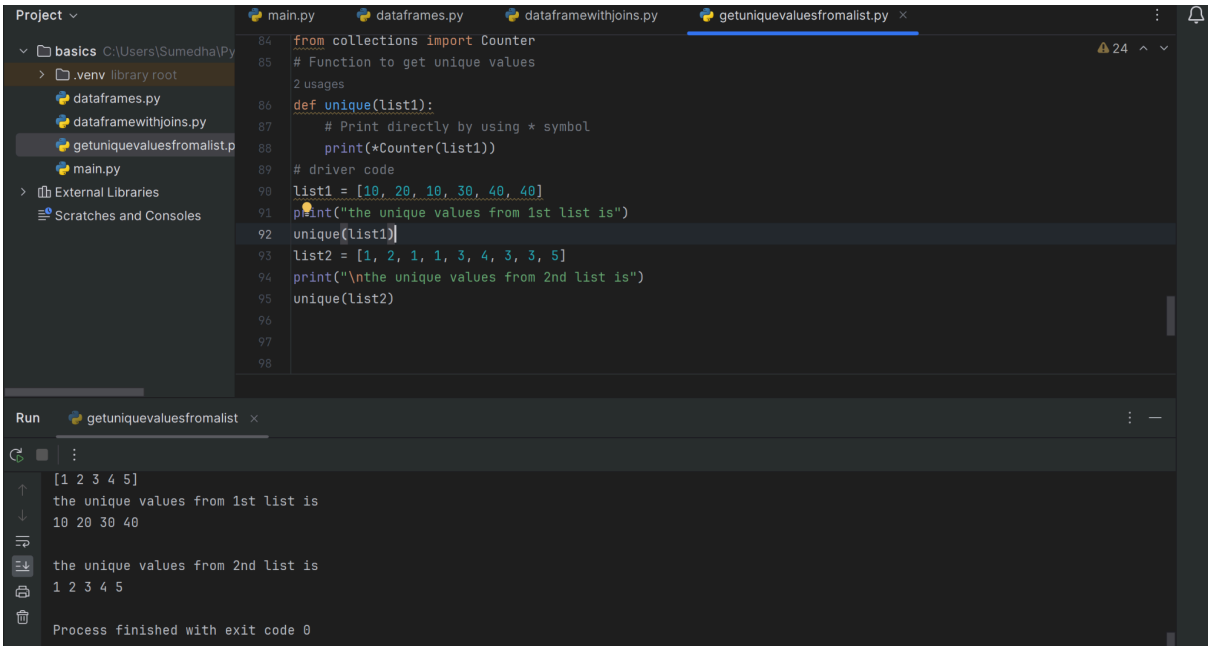


—>**Get Unique Values From a List Using numpy.unique:**
Using Python's import numpy()the unique elements in the array are also obtained. In the first step convert the list to **x=numpy.array(list)** and then use numpy.unique(x) function to get the unique values from the list. **numpy.unique()** returns only the unique values in the list.
Example:



—>Get Unique Values From a List in Python Using collections.Counter():
Using Python to import Counter() from collections() print all the keys of Counter elements or we print directly by using the **"*"** symbol. Below is the implementation of the above approach.

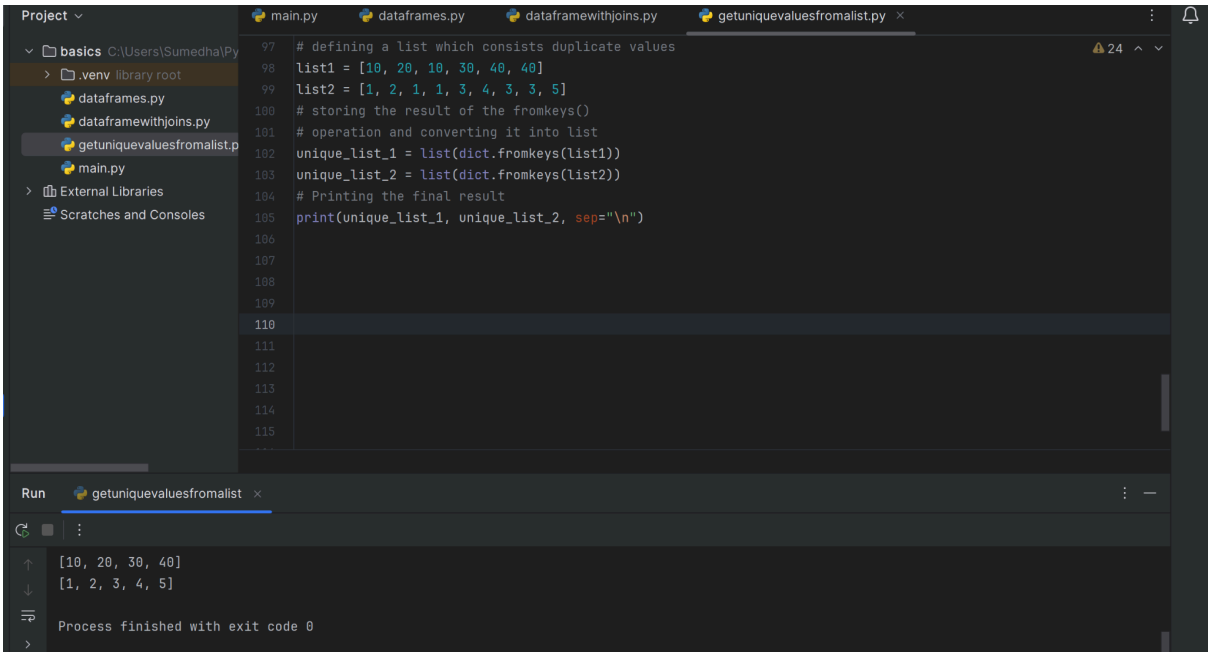
Example:



—>Get Unique Values From a List Using dict.fromkeys():

Using the fromkeys() method of dictionary data structure we can fetch the unique elements.Firstly we need to define a list that consists of duplicate elements.Then we need to use a variable in which we will store the result after using the fromkeys() method.We need to convert that result into a list, as the fromkeys() method is part of the dictionary so by default it returns a dictionary with all the unique keys and None as their values.

Example:


Project
basics C:\Users\Sumedha\Py
.venv library root
dataframes.py
dataframewithjoins.py
getuniquevaluesfromalist.p
main.py
External Libraries
Scratches and Consoles
main.py
dataframes.py
dataframewithjoins.py
getuniquevaluesfromalist.py
24
# defining a list which consists duplicate values
list1 = [10, 20, 10, 30, 40, 40]
list2 = [1, 2, 1, 1, 3, 4, 3, 3, 5]
# storing the result of the fromkeys()
# operation and converting it into list
unique_list_1 = list(dict.fromkeys(list1))
unique_list_2 = list(dict.fromkeys(list2))
# Printing the final result
print(unique_list_1, unique_list_2, sep="\n")
Run
getuniquevaluesfromalist
[10, 20, 30, 40]
[1, 2, 3, 4, 5]
Process finished with exit code 0

6)GET COUNT BY STATUS USING PANDAS DATAFRAME APIS:

To count values in Pandas dataframe. First, we will create a data frame, and then we will count the values of different attributes.

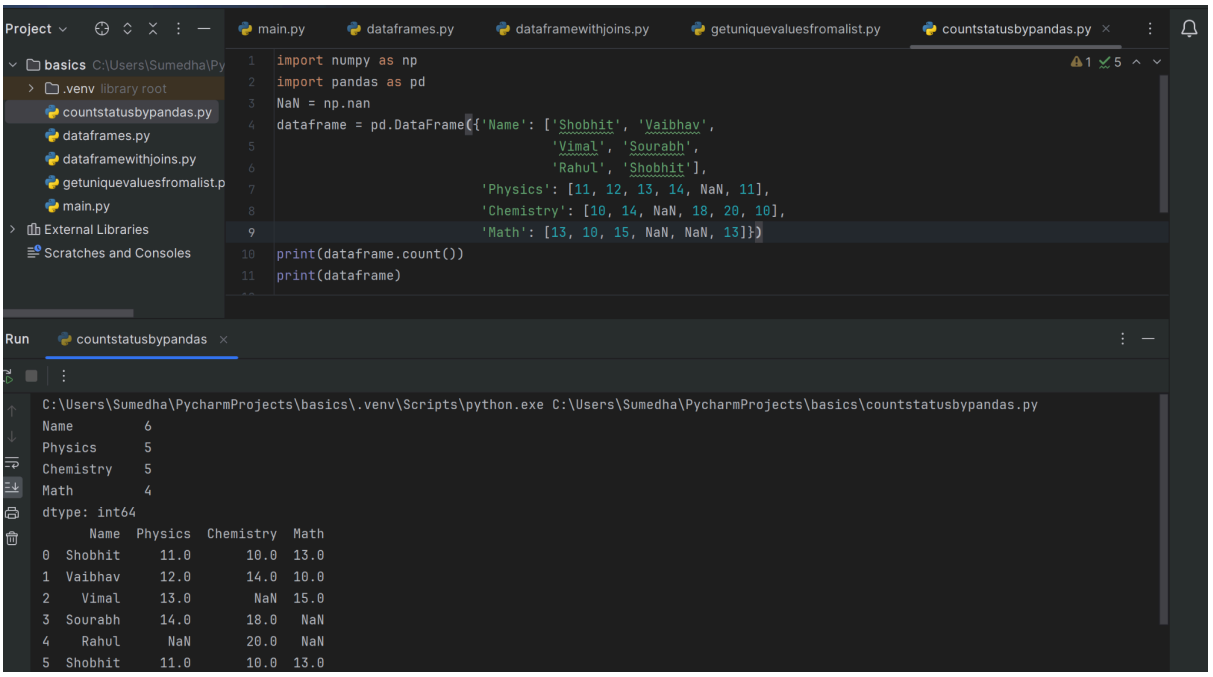
Syntax: DataFrame.count(axis=0, level=None, numeric_only=False)

Example:

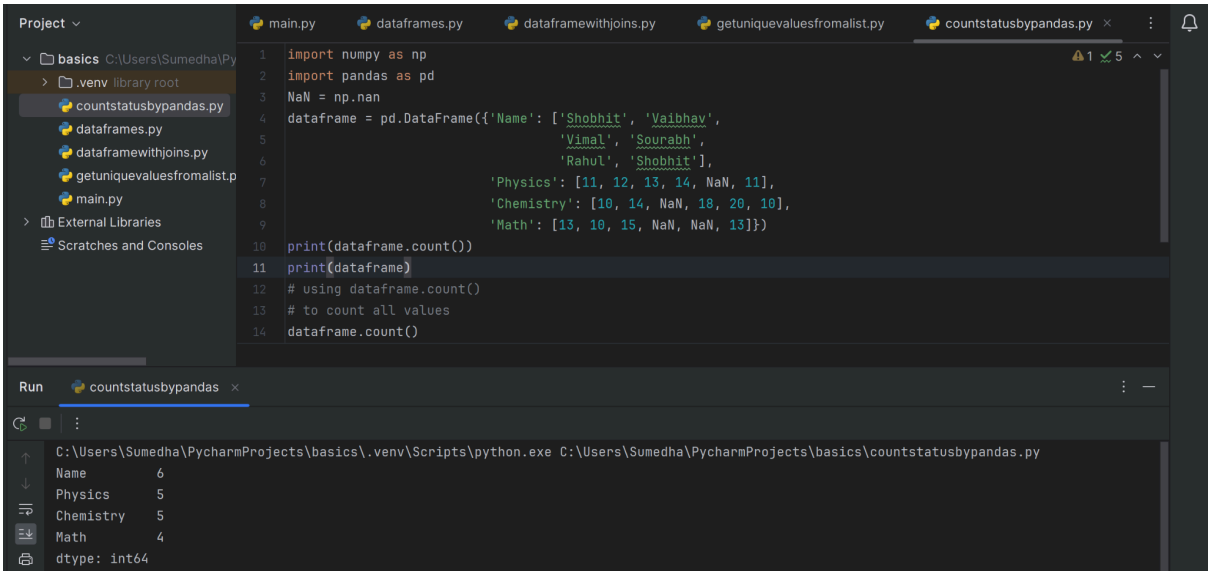
Count Values in Pandas Dataframe

Step 1: Importing libraries.

Step 2: Creating Dataframe.



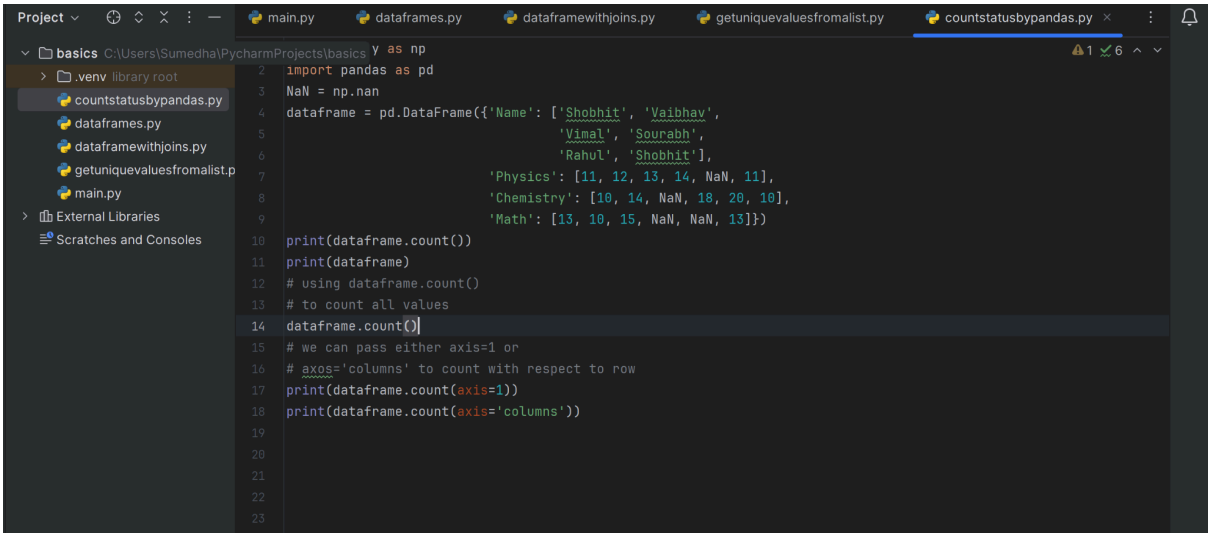
Step 3: In this step, we just simply use the **.count() function** to count all the values of different columns.



```
import numpy as np
import pandas as pd
NaN = np.nan
dataframe = pd.DataFrame({'Name': ['Shobhit', 'Vaibhav',
                                   'Vimal', 'Sourabh',
                                   'Rahul', 'Shobhit'],
                          'Physics': [11, 12, 13, 14, NaN, 11],
                          'Chemistry': [10, 14, NaN, 18, 20, 10],
                          'Math': [13, 10, 15, NaN, NaN, 13]})
print(dataframe.count())
print(dataframe)
# using dataframe.count()
# to count all values
dataframe.count()
```

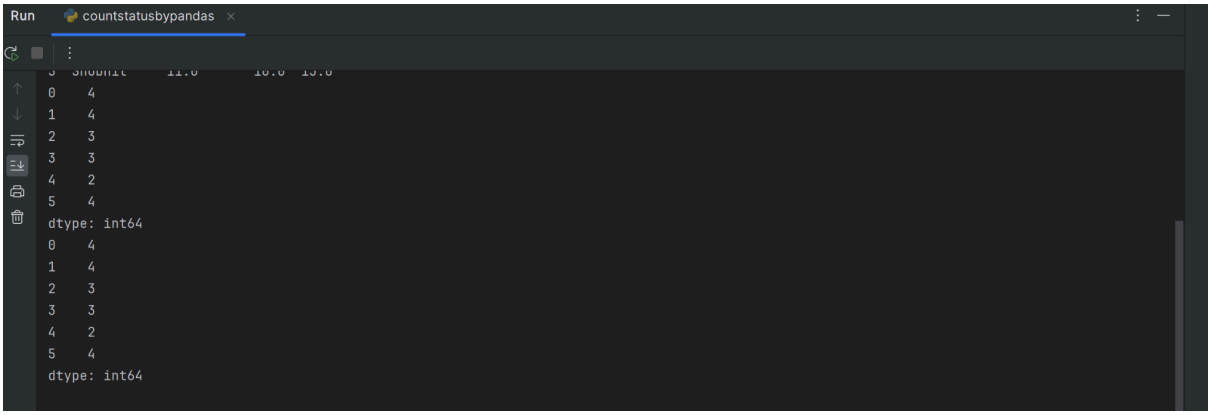
```
C:\Users\Sumedha\PycharmProjects\basics\.venv\Scripts\python.exe C:\Users\Sumedha\PycharmProjects\basics\countstatusbypandas.py
Name         6
Physics      5
Chemistry    5
Math         4
dtype: int64
```

Step 4: If we want to count all the values with respect to row then we have to pass axis=1 or ‘columns’.



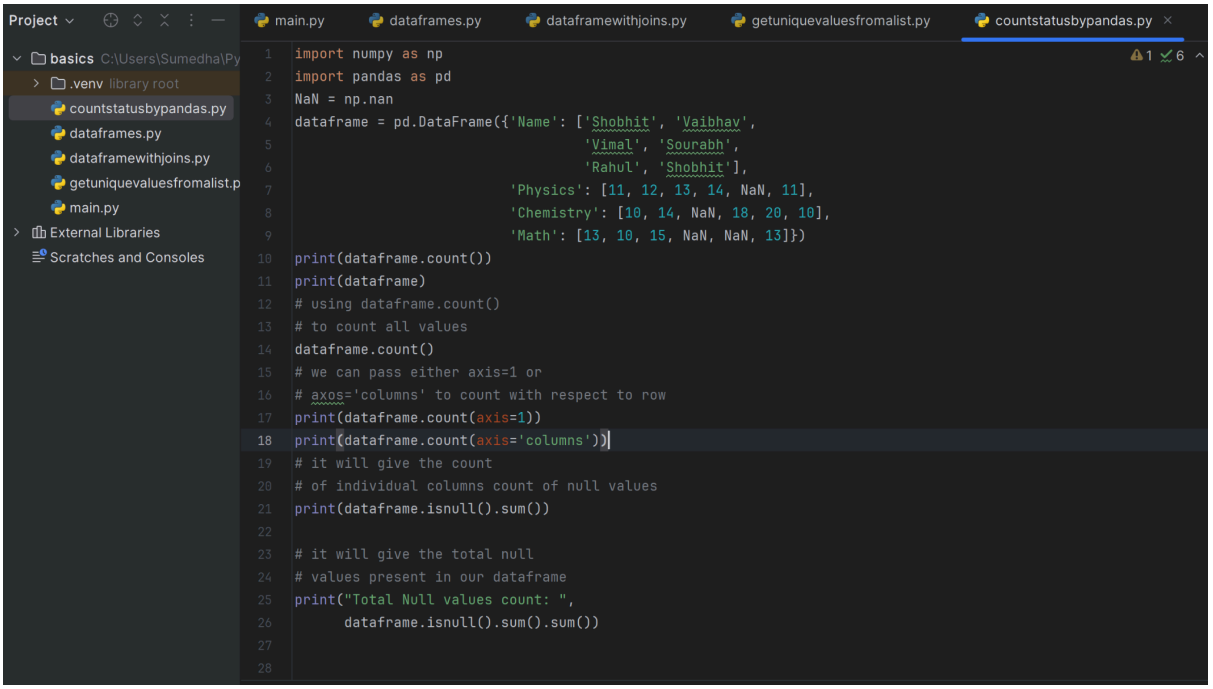
```
import numpy as np
import pandas as pd
NaN = np.nan
dataframe = pd.DataFrame({'Name': ['Shobhit', 'Vaibhav',
                                   'Vimal', 'Sourabh',
                                   'Rahul', 'Shobhit'],
                          'Physics': [11, 12, 13, 14, NaN, 11],
                          'Chemistry': [10, 14, NaN, 18, 20, 10],
                          'Math': [13, 10, 15, NaN, NaN, 13]})
print(dataframe.count())
print(dataframe)
# using dataframe.count()
# to count all values
dataframe.count()
# we can pass either axis=1 or
# axos='columns' to count with respect to row
print(dataframe.count(axis=1))
print(dataframe.count(axis='columns'))
```

OUTPUT:



```
0    4
1    4
2    3
3    3
4    2
5    4
dtype: int64
0    4
1    4
2    3
3    3
4    2
5    4
dtype: int64
```

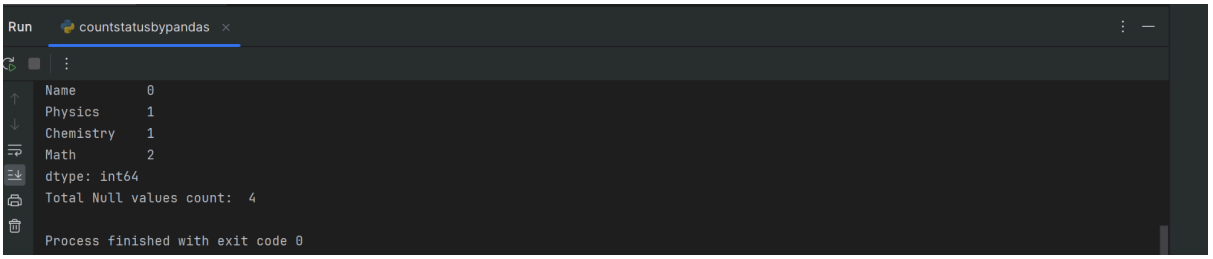
Step 5: Now if we want to count null values in our dataframe.



```
import numpy as np
import pandas as pd
NaN = np.nan
dataframe = pd.DataFrame({'Name': ['Shobhit', 'Vaibhav',
                                   'Vimal', 'Sourabh',
                                   'Rahul', 'Shobhit'],
                          'Physics': [11, 12, 13, 14, NaN, 11],
                          'Chemistry': [10, 14, NaN, 18, 20, 10],
                          'Math': [13, 10, 15, NaN, NaN, 13]})
print(dataframe.count())
print(dataframe)
# using dataframe.count()
# to count all values
dataframe.count()
# we can pass either axis=1 or
# axos='columns' to count with respect to row
print(dataframe.count(axis=1))
print(dataframe.count(axis='columns'))
# it will give the count
# of individual columns count of null values
print(dataframe.isnull().sum())

# it will give the total null
# values present in our dataframe
print("Total Null values count: ",
      dataframe.isnull().sum().sum())
```

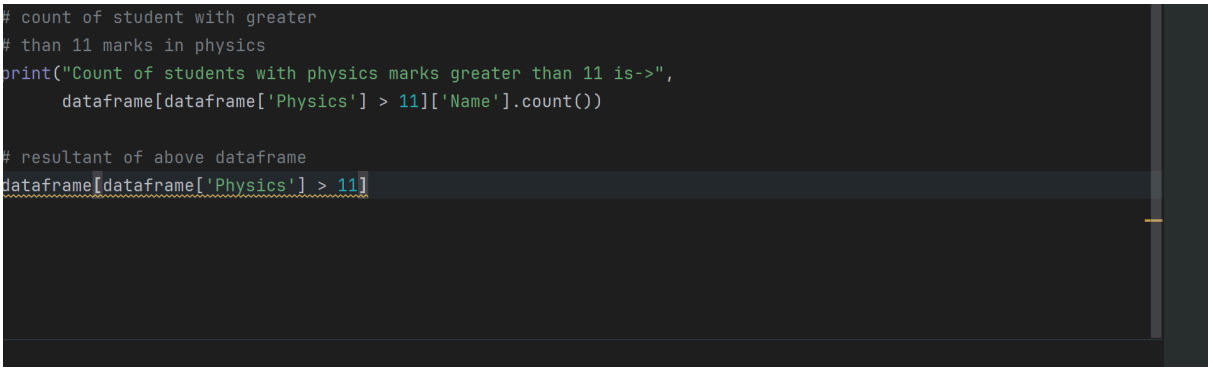
OUTPUT:



```
Name         0
Physics      1
Chemistry    1
Math         2
dtype: int64
Total Null values count:  4

Process finished with exit code 0
```

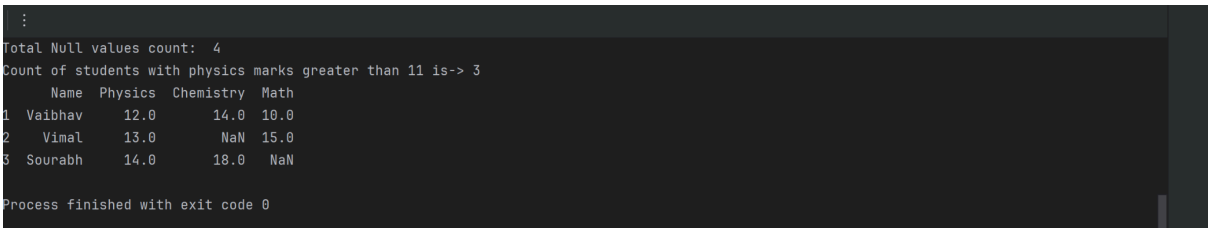
Step 6:. Some examples to use **.count()**
Now we want to count no of students whose physics marks are greater than 11.



```
# count of student with greater
# than 11 marks in physics
print("Count of students with physics marks greater than 11 is->",
      dataframe[dataframe['Physics'] > 11]['Name'].count())

# resultant of above dataframe
dataframe[dataframe['Physics'] > 11]
```

OUTPUT:



```
Total Null values count:  4
Count of students with physics marks greater than 11 is-> 3
      Name  Physics  Chemistry  Math
1  Vaibhav     12.0       14.0  10.0
2    Vimal     13.0        NaN  15.0
3  Sourabh     14.0       18.0   NaN

Process finished with exit code 0
```