

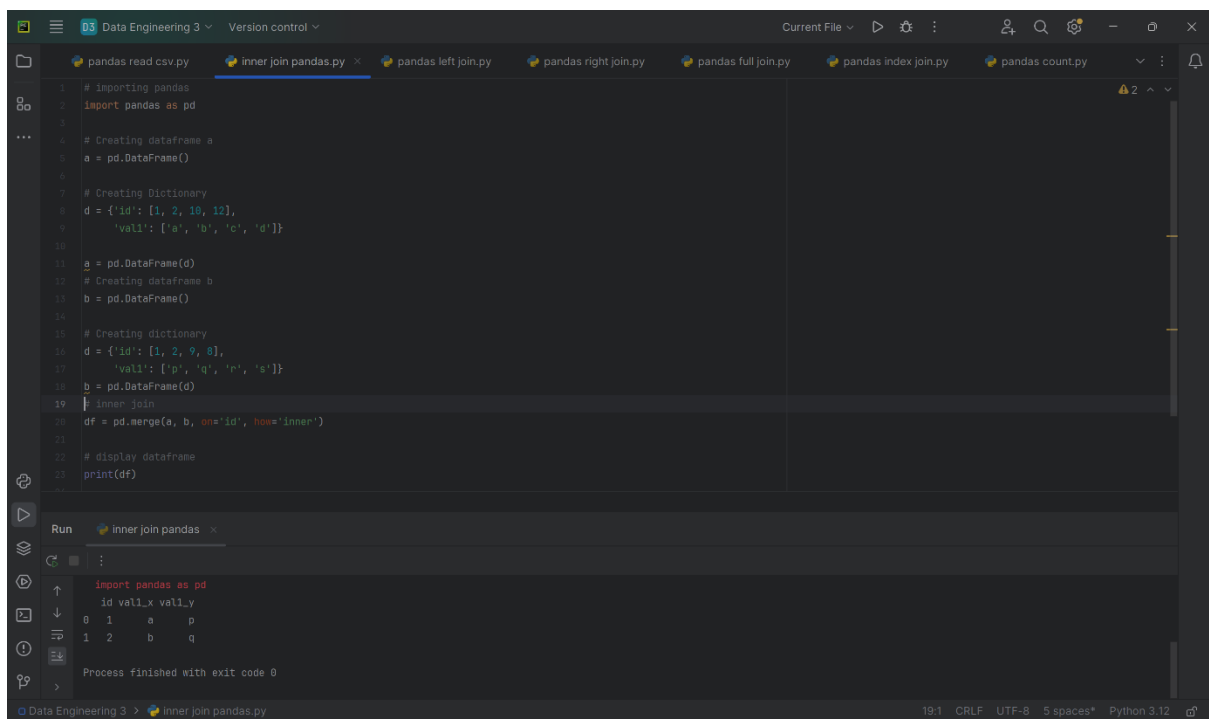
Name: Pradip Bochare

Joins Using Pandas

The Pandas module contains various features to perform various operations on Dataframes like join, concatenate, delete, add, etc. In this article, we are going to discuss the various types of join operations that can be performed on Pandas Dataframe. 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

❖ Inner Join



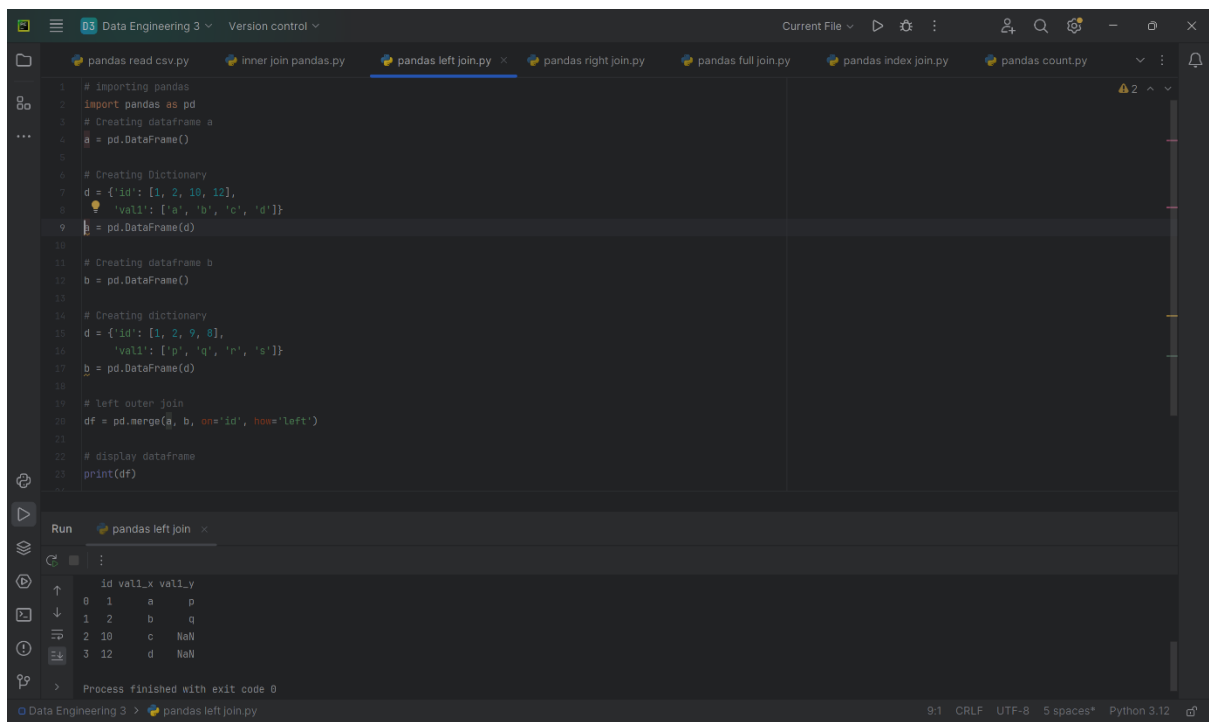
```
1 # importing pandas
2 import pandas as pd
3
4 # Creating dataframe a
5 a = pd.DataFrame()
6
7 # Creating Dictionary
8 d = {'id': [1, 2, 10, 12],
9      'val1': ['a', 'b', 'c', 'd']}
10
11 a = pd.DataFrame(d)
12 # Creating dataframe b
13 b = pd.DataFrame()
14
15 # Creating dictionary
16 d = {'id': [1, 2, 9, 8],
17      'val1': ['p', 'q', 'r', 's']}
18 b = pd.DataFrame(d)
19
20 # inner join
21 df = pd.merge(a, b, on='id', how='inner')
22
23 # display dataframe
24 print(df)
```

Run inner join pandas

```
import pandas as pd
id val1_x val1_y
0 1 a p
1 2 b q
```

Process finished with exit code 0

❖ Left Outer Join



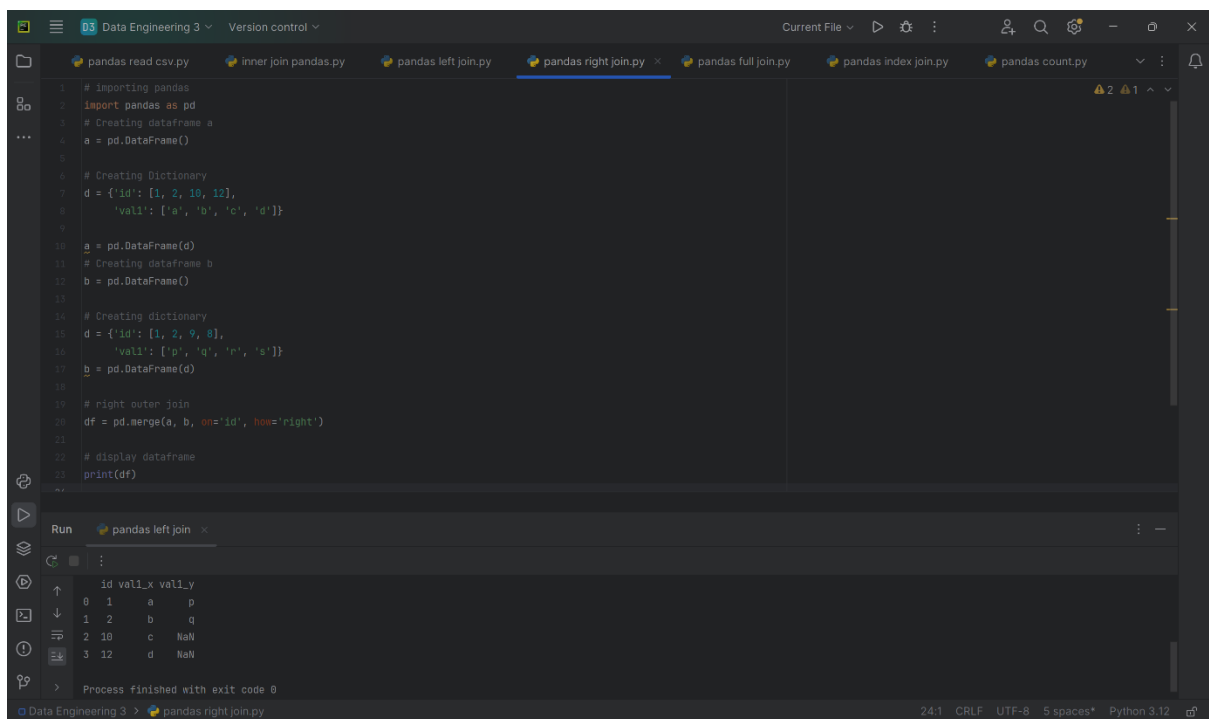
The screenshot shows a Jupyter Notebook with a file explorer at the top containing several Python files. The active file is 'pandas left join.py'. The code defines two DataFrames, 'a' and 'b', and performs a left outer join using 'pd.merge(a, b, on='id', how='left')'. The output is displayed in a table below the code.

```
1 # importing pandas
2 import pandas as pd
3 # Creating dataframe a
4 a = pd.DataFrame()
5
6 # Creating Dictionary
7 d = {'id': [1, 2, 10, 12],
8      'val1': ['a', 'b', 'c', 'd']}
9 a = pd.DataFrame(d)
10
11 # Creating dataframe b
12 b = pd.DataFrame()
13
14 # Creating dictionary
15 d = {'id': [1, 2, 9, 8],
16      'val1': ['p', 'q', 'r', 's']}
17 b = pd.DataFrame(d)
18
19 # left outer join
20 df = pd.merge(a, b, on='id', how='left')
21
22 # display dataframe
23 print(df)
```

	id	val1_x	val1_y
0	1	a	p
1	2	b	q
2	10	c	NaN
3	12	d	NaN

Process finished with exit code 0

❖ Right Outer Join



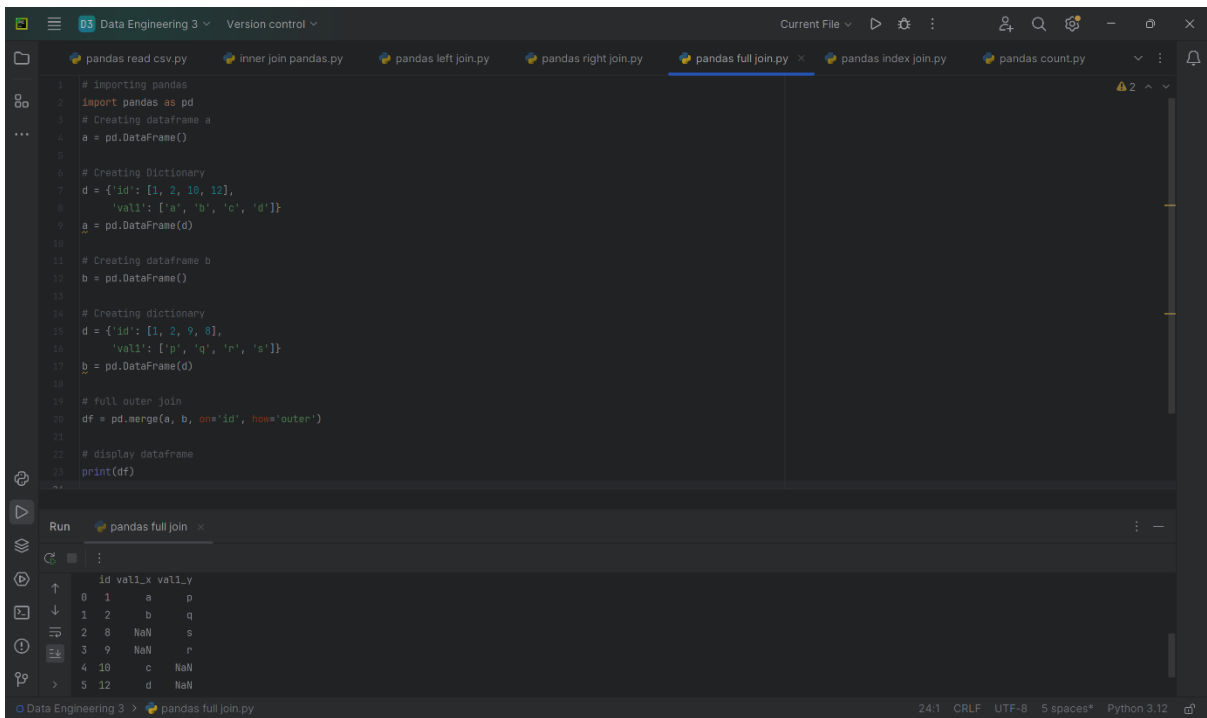
The screenshot shows a Jupyter Notebook with a file explorer at the top containing several Python files. The active file is 'pandas right join.py'. The code defines two DataFrames, 'a' and 'b', and performs a right outer join using 'pd.merge(a, b, on='id', how='right')'. The output is displayed in a table below the code.

```
1 # importing pandas
2 import pandas as pd
3 # Creating dataframe a
4 a = pd.DataFrame()
5
6 # Creating Dictionary
7 d = {'id': [1, 2, 10, 12],
8      'val1': ['a', 'b', 'c', 'd']}
9 a = pd.DataFrame(d)
10
11 # Creating dataframe b
12 b = pd.DataFrame()
13
14 # Creating dictionary
15 d = {'id': [1, 2, 9, 8],
16      'val1': ['p', 'q', 'r', 's']}
17 b = pd.DataFrame(d)
18
19 # right outer join
20 df = pd.merge(a, b, on='id', how='right')
21
22 # display dataframe
23 print(df)
```

	id	val1_x	val1_y
0	1	a	p
1	2	b	q
2	10	c	NaN
3	12	d	NaN

Process finished with exit code 0

❖ Full Outer Join

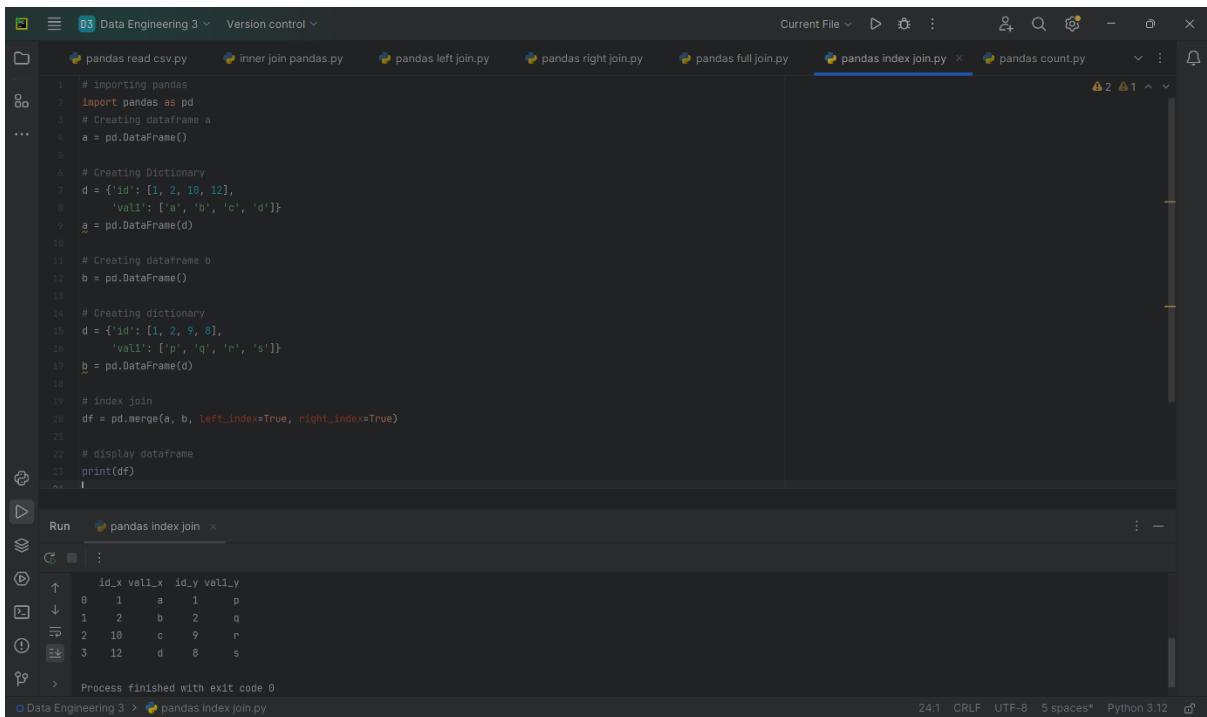


The screenshot shows a Jupyter Notebook with a file explorer on the left containing several Python files. The active file is 'pandas full join.py'. The code defines two DataFrames, 'a' and 'b', and performs a full outer join using 'pd.merge(a, b, on='id', how='outer')'. The output is displayed in a table below the code.

```
1 # importing pandas
2 import pandas as pd
3 # Creating dataframe a
4 a = pd.DataFrame()
5
6 # Creating Dictionary
7 d = {'id': [1, 2, 10, 12],
8      'val1': ['a', 'b', 'c', 'd']}
9 a = pd.DataFrame(d)
10
11 # Creating dataframe b
12 b = pd.DataFrame()
13
14 # Creating dictionary
15 d = {'id': [1, 2, 9, 8],
16      'val1': ['p', 'q', 'r', 's']}
17 b = pd.DataFrame(d)
18
19 # full outer join
20 df = pd.merge(a, b, on='id', how='outer')
21
22 # display dataframe
23 print(df)
```

	id	val1_x	val1_y
0	1	a	p
1	2	b	q
2	8	NaN	s
3	9	NaN	r
4	10	c	NaN
5	12	d	NaN

❖ Index Join



The screenshot shows a Jupyter Notebook with a file explorer on the left. The active file is 'pandas index join.py'. The code defines two DataFrames, 'a' and 'b', and performs an index join using 'pd.merge(a, b, left_index=True, right_index=True)'. The output is displayed in a table below the code.

```
1 # importing pandas
2 import pandas as pd
3 # Creating dataframe a
4 a = pd.DataFrame()
5
6 # Creating Dictionary
7 d = {'id': [1, 2, 10, 12],
8      'val1': ['a', 'b', 'c', 'd']}
9 a = pd.DataFrame(d)
10
11 # Creating dataframe b
12 b = pd.DataFrame()
13
14 # Creating dictionary
15 d = {'id': [1, 2, 9, 8],
16      'val1': ['p', 'q', 'r', 's']}
17 b = pd.DataFrame(d)
18
19 # index join
20 df = pd.merge(a, b, left_index=True, right_index=True)
21
22 # display dataframe
23 print(df)
```

	id_x	val1_x	id_y	val1_y
0	1	a	1	p
1	2	b	2	q
2	10	c	9	r
3	12	d	8	s

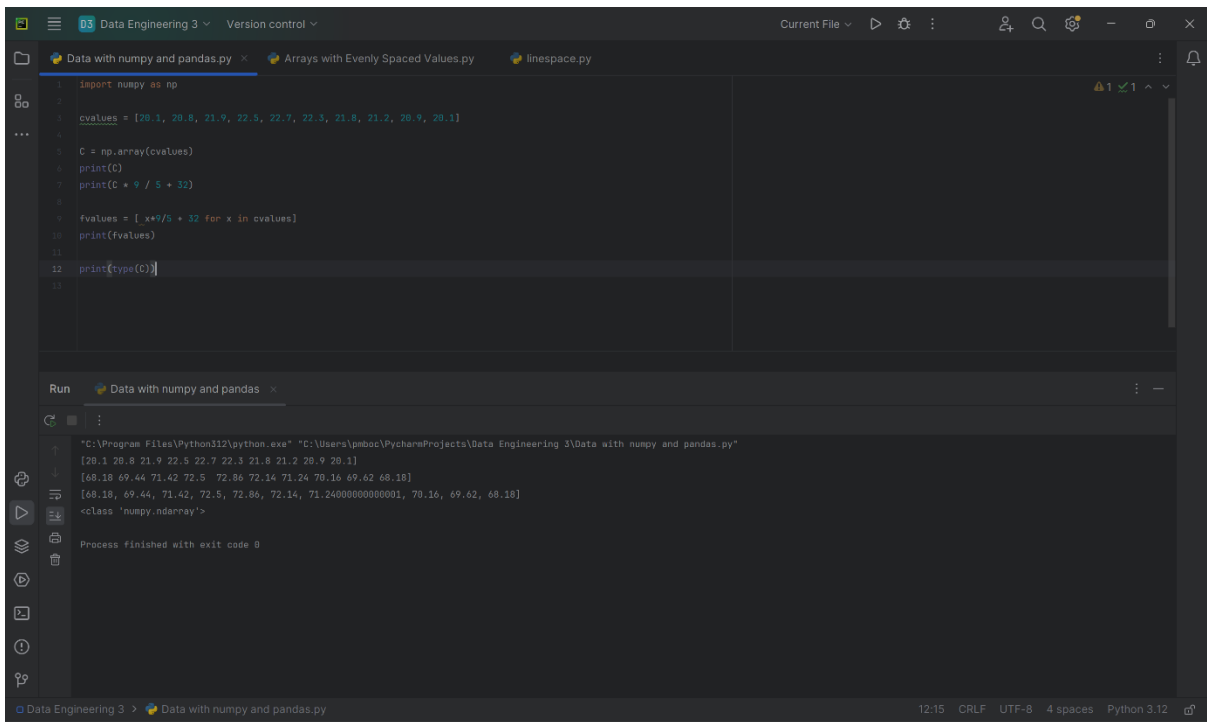
Count Values in Pandas Dataframe

```
1 import numpy as np
2 import pandas as pd
3
4 NaN = np.nan
5 dataframe = pd.DataFrame({'Name': ['Shobhit', 'Vaihbhav',
6                                   'Vimal', 'Sourabh',
7                                   'Rahul', 'Shobhit'],
8                           'Physics': [11, 12, 13, 14, NaN, 11],
9                           'Chemistry': [10, 14, NaN, 18, 20, 10],
10                          'Math': [13, 18, 15, NaN, NaN, 13]})
11
12 print(dataframe.count())
13 print(dataframe)
14
15 print(dataframe.count(axis=1))
16 print(dataframe.count(axis='columns'))
17
18 print(dataframe.isnull().sum())
19 print("Total Null values count: ",
20       dataframe.isnull().sum().sum())
21
22 print("Count of students with physics marks greater than 11 is->",
23       dataframe[dataframe['Physics'] > 11]['Name'].count())
24
25 print(dataframe[dataframe['Physics'] > 11])
26
27 print("Count of students ->",
28       dataframe[(dataframe['Physics'] > 18) &
29                 (dataframe['Chemistry'] > 11) &
30                 (dataframe['Math'] > 9)]['Name'].count())
31
32 print(dataframe[(dataframe['Physics'] > 18) &
33                 (dataframe['Chemistry'] > 11) &
34                 (dataframe['Math'] > 9)])
```

Result

```
Run pandas count
import pandas as pd
Name      6
Physics    5
Chemistry  5
Math       4
dtype: int64
Name Physics Chemistry Math
0 Shobhit  11.0      10.0  13.0
1 Vaihbhav 12.0      14.0  10.0
2 Vimal    13.0      NaN   15.0
3 Sourabh  14.0      18.0   NaN
4 Rahul    NaN      20.0   NaN
5 Shobhit  11.0      10.0  13.0
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
Name      0
Physics    1
Chemistry  1
Math       2
dtype: int64
Total Null values count: 4
Count of students with physics marks greater than 11 is-> 3
```


Pandas Linespace



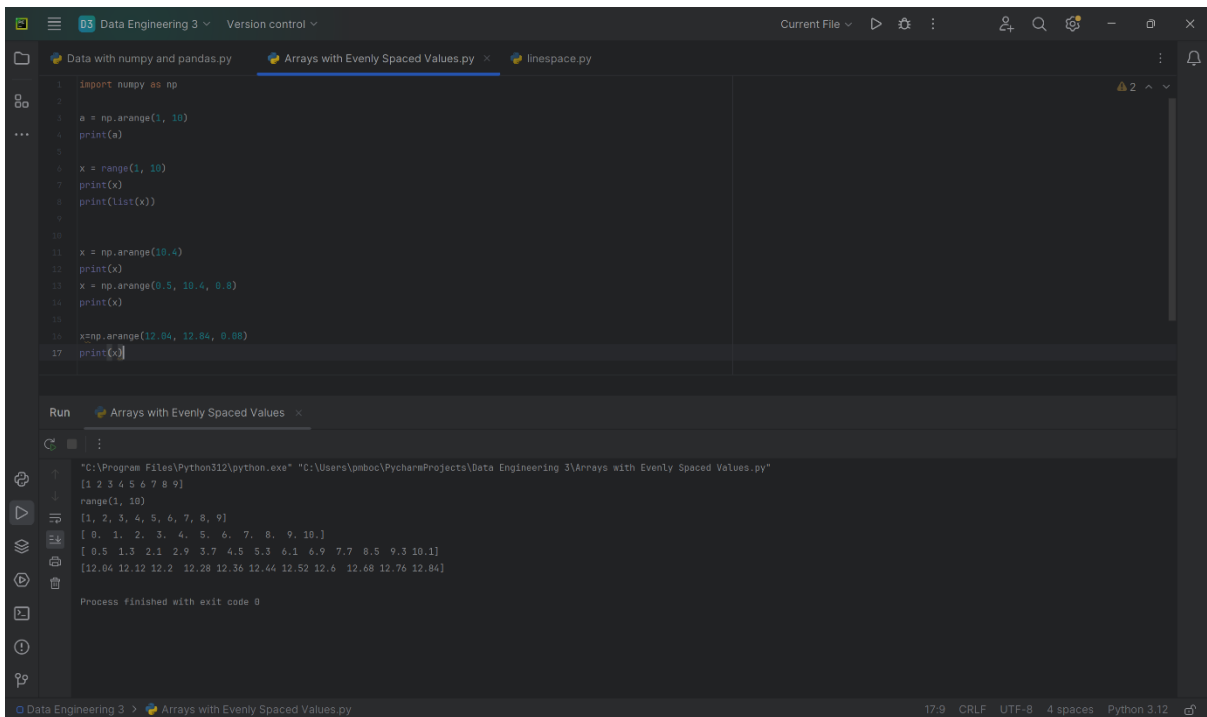
The screenshot shows the PyCharm IDE with a project named "Data Engineering 3". The active file is "Data with numpy and pandas.py". The code in the editor is as follows:

```
1 import numpy as np
2
3 cvalues = [28.1, 28.8, 21.9, 22.5, 22.7, 22.3, 21.8, 21.2, 28.9, 28.1]
4
5 C = np.array(cvalues)
6 print(C)
7 print(C * 9 / 5 + 32)
8
9 fvalues = [ x*9/5 + 32 for x in cvalues]
10 print(fvalues)
11
12 print(type(C))
13
```

The Run console shows the output of the script:

```
"C:\Program Files\Python312\python.exe" "C:\Users\pmboc\PycharmProjects\Data Engineering 3\Data with numpy and pandas.py"
[28.1 28.8 21.9 22.5 22.7 22.3 21.8 21.2 28.9 28.1]
[68.18 69.44 71.42 72.5 72.86 72.14 71.24 70.16 69.62 68.18]
[68.18, 69.44, 71.42, 72.5, 72.86, 72.14, 71.24000000000001, 70.16, 69.62, 68.18]
<class 'numpy.ndarray'>
Process finished with exit code 0
```

The status bar at the bottom indicates the file encoding is CRLF, UTF-8, 4 spaces, and the Python version is 3.12.



The screenshot shows the PyCharm IDE with a project named "Data Engineering 3". The active file is "Arrays with Evenly Spaced Values.py". The code in the editor is as follows:

```
1 import numpy as np
2
3 a = np.arange(1, 10)
4 print(a)
5
6 x = range(1, 10)
7 print(x)
8 print(list(x))
9
10 x = np.arange(10.4)
11 print(x)
12
13 x = np.arange(0.5, 10.4, 0.8)
14 print(x)
15
16 x=np.arange(12.04, 12.84, 0.08)
17 print(x)
```

The Run console shows the output of the script:

```
"C:\Program Files\Python312\python.exe" "C:\Users\pmboc\PycharmProjects\Data Engineering 3\Arrays with Evenly Spaced Values.py"
[1 2 3 4 5 6 7 8 9]
range(1, 10)
[1, 2, 3, 4, 5, 6, 7, 8, 9]
[0. 1. 2. 3. 4. 5. 6. 7. 8. 9. 10.]
[0.5 1.3 2.1 2.9 3.7 4.5 5.3 6.1 6.9 7.7 8.5 9.3 10.1]
[12.04 12.12 12.2 12.28 12.36 12.44 12.52 12.6 12.68 12.76 12.84]
Process finished with exit code 0
```

The status bar at the bottom indicates the file encoding is CRLF, UTF-8, 4 spaces, and the Python version is 3.12.

```
1 import numpy as np
2
3 # 50 values between 1 and 10:
4 print(np.linspace(start=1, stop=10))
5
6 # 7 values between 1 and 10:
7 print(np.linspace(start=1, stop=10, num=7))
8
9 # excluding the endpoint:
10 print(np.linspace(start=1, stop=10, num=7, endpoint=False))
11
12 samples, spacing = np.linspace(start=1, stop=10, retstep=True)
13 print(samples)
14 print(spacing)
15
16 samples, spacing = np.linspace(start=1, stop=10, num=20, endpoint=True, retstep=True)
17 print(samples)
18 print(spacing)
19
20 samples, spacing = np.linspace(start=1, stop=10, num=20, endpoint=False, retstep=True)
21 print(samples)
22 print(spacing)
```

Run linespace x

Data Engineering 3 > linespace.py 8:1 CRLF UTF-8 4 spaces Python 3.12

Result

```
Run linespace x
"C:\Program Files\Python312\python.exe" "C:\Users\pmboc\PycharmProjects\Data Engineering 3\linespace.py"
[ 1.  1.18367347  1.36734694  1.55102041  1.73469388  1.91836735
 2.10204082  2.28571429  2.46938776  2.65306122  2.83673469  3.02040816
 3.20408163  3.3877551  3.57142857  3.75510204  3.93877551  4.12244898
 4.30612245  4.48979592  4.67346939  4.85714286  5.04081633  5.2244898
 5.40816327  5.59183673  5.7755102  5.95918367  6.14285714  6.32653061
 6.51020408  6.69387755  6.87755102  7.06122449  7.24489796  7.42857143
 7.6122449  7.79591837  7.97959184  8.16326531  8.34693878  8.53061224
 8.71428571  8.89795918  9.08163265  9.26530612  9.44897959  9.63265306
 9.81632653 10. ]
[ 1.  2.5  4.  5.5  7.  8.5 10. ]
[ 1.  2.28571429  3.57142857  4.85714286  6.14285714  7.42857143
 8.71428571]
[ 1.  1.18367347  1.36734694  1.55102041  1.73469388  1.91836735
 2.10204082  2.28571429  2.46938776  2.65306122  2.83673469  3.02040816
 3.20408163  3.3877551  3.57142857  3.75510204  3.93877551  4.12244898
 4.30612245  4.48979592  4.67346939  4.85714286  5.04081633  5.2244898
 5.40816327  5.59183673  5.7755102  5.95918367  6.14285714  6.32653061
 6.51020408  6.69387755  6.87755102  7.06122449  7.24489796  7.42857143
 7.6122449  7.79591837  7.97959184  8.16326531  8.34693878  8.53061224
 8.71428571  8.89795918  9.08163265  9.26530612  9.44897959  9.63265306
 9.81632653 10. ]
0.1836734693877551
[ 1.  1.47368421  1.94736842  2.42105263  2.89473684  3.36842105
 3.84210526  4.31578947  4.78947368  5.26315789  5.73684211  6.21052632
 6.68421053  7.15789474  7.63157895  8.10526316  8.57894737  9.05263158
 9.52631579 10. ]
0.47368421052631576
[ 1.  1.45 1.9 2.35 2.8 3.25 3.7 4.15 4.6 5.05 5.5 5.95 6.4 6.85
 7.3 7.75 8.2 8.65 9.1 9.55]
0.45
Process finished with exit code 0
```

Data Engineering 3 > linespace.py 8:1 CRLF UTF-8 4 spaces Python 3.12