NumPy

1. How to extract all odd numbers from arr?

```
[102] import numpy as np
      arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

      odd_numbers = arr[arr % 2 != 0]
      print("Original array:", arr)
      print("Odd numbers:", odd_numbers)
```

```
Original array: [0 1 2 3 4 5 6 7 8 9]
Odd numbers: [1 3 5 7 9]
```

2. Replace all odd numbers in arr with -1 without changing

```
[103] import numpy as np
      arr = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])

      arr_modified = np.where(arr % 2 != 0, -1, arr)
      print("modified array : ",arr_modified)
```

```
modified array :  [ 0 -1  2 -1  4 -1  6 -1  8 -1]
```

3. Convert a 1D array to a 2D array with 2 rows

```
[104] import numpy as np
      arr=np.arange(10)

      #Desired Output:
      #> array([[0, 1, 2, 3, 4],
      #>        [5, 6, 7, 8, 9]])

      arr_2d = arr.reshape((2, -1))

      print("Original 1D array:")
      print(arr)

      print("\nReshaped 2D array:")
      print(arr_2d)
```

```
Original 1D array:
[0 1 2 3 4 5 6 7 8 9]

Reshaped 2D array:
[[0 1 2 3 4]
 [5 6 7 8 9]]
```

4. Stack arrays a and b vertically

```
[105] import numpy as np
      a = np.arange(10).reshape(2,-1)
      b = np.repeat(1, 10).reshape(2,-1)
      result = np.vstack((a, b))
      print(result)
      #Desired Output:

      #> array([[0, 1, 2, 3, 4],
      #>        [5, 6, 7, 8, 9],
      #>        [1, 1, 1, 1, 1],
      #>        [1, 1, 1, 1, 1]])
```

```
[[0 1 2 3 4]
 [5 6 7 8 9]
 [1 1 1 1 1]
 [1 1 1 1 1]]
```

5. Stack the arrays a and b horizontally.

```
[106] import numpy as np
      a = np.arange(10).reshape(2,-1)
      b = np.repeat(1, 10).reshape(2,-1)
      result = np.hstack((a, b))
      print(result)
```

```
#Desired Output:

#> array([[0, 1, 2, 3, 4, 1, 1, 1, 1, 1],
#>        [5, 6, 7, 8, 9, 1, 1, 1, 1, 1]])
```

```
[[0 1 2 3 4 1 1 1 1 1]
 [5 6 7 8 9 1 1 1 1 1]]
```

6.How to get the common items between two python numpy arrays?

Get the common items between a and b

```
[107] import numpy as np
      a = np.array([1,2,3,2,3,4,3,4,5,6])
      b = np.array([7,2,10,2,7,4,9,4,9,8])
      common_elements = np.intersect1d(a, b)
      print(common_elements)
```

```
[2 4]
```

7.How to remove from one array those items that exist in another?

Q. From array a remove all items present in array b

```
[108] import numpy as np

      a = np.array([1,2,3,4,5])
      b = np.array([5,6,7,8,9])
      result = np.setdiff1d(a, b)
      print(result)
```

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

8. How to get the positions where elements of two arrays match?

Q. Get the positions where elements of a and b match

```
[109] import numpy as np
      a = np.array([1,2,3,2,3,4,3,4,5,6])
      b = np.array([7,2,10,2,7,4,9,4,9,8])
      matching_positions = np.where(np.in1d(a, b))[0]

      print(matching_positions)
      #Desired Output:

      #> (array([1, 3, 5, 7]),)
```

```
[1 3 5 7]
```

9. How to extract all numbers between a given range from a numpy array?

Get all items between 5 and 10 from a

```
[110] import numpy as np

      a = np.array([2, 6, 1, 9, 10, 3, 27])
      indices = np.where(a > 5)
      print(indices)
      #Desired Output:

      #(array([6, 9, 10]),)
```

```
(array([1, 3, 4, 6]),)
```

Pandas

Pandas Data Series

1. Write a Pandas program to convert a dictionary to a Pandas series.

```
Sample Series:
Original dictionary:
{'a': 100, 'b': 200, 'c': 300, 'd': 400, 'e': 800}

Converted series:
```

```
a    100
b    200
c    300
d    400
e    800
dtype: int64
```

```
[111] import pandas as pd
      original_dict = {'a': 100, 'b': 200, 'c': 300, 'd': 400, 'e': 800}
      converted_series = pd.Series(original_dict)
      print("Converted series:")
      print(converted_series)
```

```
Converted series:
a    100
b    200
c    300
d    400
e    800
dtype: int64
```

2. Write a Pandas program to convert a NumPy array to a Pandas series.

```
Sample Series:
  NumPy array:
[10 20 30 40 50]
Converted Pandas series:
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

```
[112] import pandas as pd
      import numpy as np
      numpy_array = np.array([10, 20, 30, 40, 50])
      converted_series = pd.Series(numpy_array)
      print("Converted Pandas series:")
      print(converted_series)
```

```
Converted Pandas series:
0    10
1    20
2    30
3    40
4    50
dtype: int64
```

3. Write a Pandas program to change the data type of given a column or a Series.

```
Sample Series:
    Original Data Series:
0      100
1      200
2    python
3    300.12
4      400
dtype: object
Change the said data type to numeric:
0    100.00
1    200.00
2       NaN
3    300.12
4    400.00
dtype: float64
```

```
[113] import pandas as pd
      original_series = pd.Series(['100', '200', 'python', '300.12', '400'])
      converted_series = pd.to_numeric(original_series, errors='coerce')
      print("original_series : \n",original_series)
      print("converted_series : \n",converted_series)
```

```
original_series :
 0       100
1       200
2    python
3    300.12
4       400
dtype: object
```

```
converted_series :
 0    100.00
1    200.00
2       NaN
3    300.12
4    400.00
dtype: float64
```

4. Write a Pandas program to convert the first column of a DataFrame as a Series.

```
Sample Output:
Original DataFrame
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
3     4     9    12
4     7     5     1
5    11     0    11

1st column as a Series:
0     1
1     2
2     3
3     4
4     7
5    11
Name: col1, dtype: int64
<class 'pandas.core.series.Series'>
```

```
[114] import pandas as pd
      data = {'col1': [1, 2, 3, 4, 7, 11],
              'col2': [4, 5, 6, 9, 5, 0],
              'col3': [7, 5, 8, 12, 1, 11]}
      df = pd.DataFrame(data)
      print("Original DataFrame")
      print(df)
      first_column_series = df.iloc[:, 0]
      print("\n1st column as a Series:")
      print(first_column_series)
      print(type(first_column_series))
```

```
Original DataFrame
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
3     4     9    12
4     7     5     1
5    11     0    11

1st column as a Series:
0     1
1     2
2     3
3     4
4     7
5    11
Name: col1, dtype: int64
<class 'pandas.core.series.Series'>
```

5. Write a Pandas program to convert a given Series to an array.

```
Sample Output:
Original Data Series:
0       100
1       200
2    python
3    300.12
4       400
dtype: object
Series to an array
['100' '200' 'python' '300.12' '400']
<class 'numpy.ndarray'>
```

```
[115] import pandas as pd
      import numpy as np
      original_series = pd.Series(['100', '200', 'python', '300.12', '400'])

      print("Original Data Series:")
      print(original_series)
      array_from_series = original_series.to_numpy()
```

```
print("\nSeries to an array")
print(array_from_series)
print(type(array_from_series))
```

```
Original Data Series:
0       100
1       200
2    python
3    300.12
4       400
dtype: object

Series to an array
['100' '200' 'python' '300.12' '400']
<class 'numpy.ndarray'>
```

6 Write a Pandas program to convert Series of lists to one Series.

```
Sample Output:
Original Series of list
0    [Red, Green, White]
1          [Red, Black]
2             [Yellow]
dtype: object
One Series
0       Red
1     Green
2     White
3       Red
4     Black
5    Yellow
dtype: object
```

```
[116] import pandas as pd

original_series = pd.Series([['Red', 'Green', 'White'], ['Red', 'Black'], ['Yellow']])

print("Original Series of list")
print(original_series)

one_series = original_series.explode()

print("\nOne Series")
print(one_series)
```

```
Original Series of list
0    [Red, Green, White]
1          [Red, Black]
2             [Yellow]
dtype: object

One Series
0       Red
0     Green
0     White
1       Red
1     Black
2    Yellow
dtype: object
```

7.Write a Pandas program to sort a given Series.

```
Sample Output:
Original Data Series:
0       100
1       200
2    python
3    300.12
4       400
dtype: object
0       100
1       200
3    300.12
4       400
2    python
dtype: object
```

```
[117] import pandas as pd
```

```
original_series = pd.Series(['100', '200', 'python', '300.12', '400'])

print("Original Data Series:")
print(original_series)

sorted_series = original_series.sort_values()

print("\nSorted Data Series:")
print(sorted_series)
```

```
Original Data Series:
0       100
1       200
2    python
3    300.12
4       400
dtype: object

Sorted Data Series:
0       100
1       200
3    300.12
4       400
2    python
dtype: object
```

Pandas DataFrame

1.Write a Pandas program to create a dataframe from a dictionary and display it.

```
Sample data: {'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':[86,97,96,72,83]}
Expected Output:
    X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
```

```
[118] import pandas as pd

data = {'X': [78, 85, 96, 80, 86], 'Y': [84, 94, 89, 83, 86], 'Z': [86, 97, 96, 72, 83]}

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

```
    X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
```

2. Write a Pandas program to create and display a DataFrame from a specified dictionary data which has the index labels.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
   attempts       name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
....
i         2      Kevin      no    8.0
j         1      Jonas     yes   19.0
```

```
[119] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
```

```
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
    }
    labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

    df = pd.DataFrame(exam_data, index=labels)
    print(df)
```

```
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

3. Write a Pandas program to get the first 3 rows of a given DataFrame.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
First three rows of the data frame:
   attempts       name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
c         2  Katherine     yes   16.5
```

```
[120] import pandas as pd
      import numpy as np

      # Sample dictionary data and list labels
      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      first_three_rows = df.head(3)

      print("First three rows of the data frame:")
      print(first_three_rows)
```

```
First three rows of the data frame:
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
```

4.Write a Pandas program to select the 'name' and 'score' columns from the following DataFrame.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Select specific columns:
        name  score
a  Anastasia   12.5
b       Dima    9.0
c  Katherine   16.5
...
h      Laura    NaN
i      Kevin    8.0
```

```
j     Jonas   19.0
```

```
[121] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)
      selected_columns = df[['name', 'score']]

      print("Select specific columns:")
      print(selected_columns)
```

```
Select specific columns:
        name  score
a  Anastasia   12.5
b       Dima    9.0
c  Katherine   16.5
d      James    NaN
e      Emily    9.0
f    Michael   20.0
g    Matthew   14.5
h      Laura    NaN
i      Kevin    8.0
j      Jonas   19.0
```

5. Write a Pandas program to select the specified columns and rows from a given data frame.

```
Sample Python dictionary data and list labels:
Select 'name' and 'score' columns in rows 1, 3, 5, 6 from the following data frame.
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Select specific columns and rows:
   score qualify
b    9.0      no
d    NaN      no
f   20.0     yes
g   14.5     yes
```

```
[122] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      selected_data = df.loc[['b', 'd', 'f', 'g'], ['score', 'qualify']]
      print("Select specific columns and rows:")
      print(selected_data)
```

```
Select specific columns and rows:
   score qualify
b    9.0      no
d    NaN      no
f   20.0     yes
g   14.5     yes
```

6. Write a Pandas program to select the rows where the number of attempts in the examination is greater than 2.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
```

```
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Number of attempts in the examination is greater than 2:
      name  score  attempts qualify
b     Dima    9.0         3      no
d    James    NaN         3      no
f  Michael   20.0         3     yes
```

```
[123] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      selected_rows = df[df['attempts'] > 2]

      print("Number of attempts in the examination is greater than 2:")
      print(selected_rows)
```

```
Number of attempts in the examination is greater than 2:
      name  score  attempts qualify
b     Dima    9.0         3      no
d    James    NaN         3      no
f  Michael   20.0         3     yes
```

7. Write a Pandas program to select the rows where the score is missing, i.e. is NaN.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Rows where score is missing:
   attempts   name qualify  score
d         3  James      no    NaN
h         1  Laura      no    NaN
```

```
[124] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      missing_score_rows = df[df['score'].isna()]

      print("Rows where score is missing:")
      print(missing_score_rows)
```

```
Rows where score is missing:
     name  score  attempts qualify
d   James    NaN         3      no
h   Laura    NaN         1      no
```

8. Write a Pandas program to select the rows the score is between 15 and 20 (inclusive).

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
```

```
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Rows where score between 15 and 20 (inclusive):
   attempts       name qualify  score
c         2  Katherine     yes   16.5
f         3    Michael     yes   20.0
j         1      Jonas     yes   19.0
```

```
[125] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      selected_rows = df[(df['score'] >= 15) & (df['score'] <= 20)]

      print("Rows where score between 15 and 20 (inclusive):")
      print(selected_rows)
```

```
Rows where score between 15 and 20 (inclusive):
        name  score  attempts qualify
c  Katherine   16.5         2     yes
f    Michael   20.0         3     yes
j      Jonas   19.0         1     yes
```

9. Write a Pandas program to select the rows where number of attempts in the examination is less than 2 and score greater than 15.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Number of attempts in the examination is less than 2 and score greater than 15 :
    name  score  attempts qualify
j  Jonas   19.0         1     yes
```

```
[126] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      selected_rows = df[(df['attempts'] < 2) & (df['score'] > 15)]
      print("Number of attempts in the examination is less than 2 and score greater than 15:")
      print(selected_rows)
```

```
Number of attempts in the examination is less than 2 and score greater than 15:
    name  score  attempts qualify
j  Jonas   19.0         1     yes
```

10. Write a Pandas program to change the score in row 'd' to 11.5.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
```

```
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Change the score in row 'd' to 11.5:
   attempts      name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
c         2  Katherine     yes   16.5
...
i         2     Kevin      no    8.0
j         1     Jonas     yes   19.0
```

[127]
```
import pandas as pd
import numpy as np

exam_data = {
    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
    'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

df.loc['d', 'score'] = 11.5

print("Change the score in row 'd' to 11.5:")
print(df)
```

```
Change the score in row 'd' to 11.5:
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d      James   11.5         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

11. Write a Pandas program to calculate the sum of the examination attempts by the students.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Sum of the examination attempts by the students:
19
```

[128]
```
import pandas as pd
import numpy as np

exam_data = {
    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
    'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

sum_attempts = df['attempts'].sum()

print("Sum of the examination attempts by the students:")
print(sum_attempts)
```

```
Sum of the examination attempts by the students:
19
```

12. Write a Pandas program to calculate the mean of all students' scores. Data is stored in a dataframe.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Mean score for each different student in data frame:
13.5625
```

```
[129] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      mean_score = df['score'].mean()
      print("Mean score for each different student in data frame:")
      print(mean_score)
```

```
Mean score for each different student in data frame:
13.5625
```

13. Write a Pandas program to replace the 'qualify' column contains the values 'yes' and 'no' with True and False.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Replace the 'qualify' column contains the values 'yes' and 'no'  with T
rue and  False:
   attempts      name  qualify  score
a         1  Anastasia     True   12.5
b         3       Dima    False    9.0
......
i         2     Kevin    False    8.0
j         1     Jonas     True   19.0
```

```
[130] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      df['qualify'] = df['qualify'].replace({'yes': True, 'no': False})

      print("Replace the 'qualify' column contains the values 'yes' and 'no' with True and False:")
      print(df)
```

```
Replace the 'qualify' column contains the values 'yes' and 'no' with True and False:
        name  score  attempts  qualify
a  Anastasia   12.5         1     True
b       Dima    9.0         3    False
c  Katherine   16.5         2     True
d      James    NaN         3    False
e      Emily    9.0         2    False
f    Michael   20.0         3     True
g    Matthew   14.5         1     True
```

```
h      Laura    NaN         1    False
i      Kevin    8.0         2    False
j      Jonas   19.0         1     True
```

14. Write a Pandas program to change the name 'James' to 'Suresh' in name column of the DataFrame.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Change the name 'James' to \?Suresh\?:
   attempts       name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
.......
i         2      Kevin      no    8.0
j         1      Jonas     yes   19.0
```

```
[131] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
          'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
          'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
      }
      labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

      df = pd.DataFrame(exam_data, index=labels)

      df['name'] = df['name'].replace({'James': 'Suresh'})

      print("Change the name 'James' to 'Suresh':")
      print(df)
```

```
Change the name 'James' to 'Suresh':
          name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
d     Suresh    NaN         3      no
e      Emily    9.0         2      no
f    Michael   20.0         3     yes
g    Matthew   14.5         1     yes
h      Laura    NaN         1      no
i      Kevin    8.0         2      no
j      Jonas   19.0         1     yes
```

15. Write a Pandas program to delete the 'attempts' column from the DataFrame.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
Delete the 'attempts' column from the data frame:
       name qualify  score
a  Anastasia     yes   12.5
b       Dima      no    9.0
.....
i      Kevin      no    8.0
j      Jonas     yes   19.0
```

```
[132] import pandas as pd
      import numpy as np

      exam_data = {
          'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
          'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
```

```
        'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
        'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
    }
    labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

    df = pd.DataFrame(exam_data, index=labels)

    df = df.drop(columns='attempts')

    print("Delete the 'attempts' column from the data frame:")
    print(df)
```

```
Delete the 'attempts' column from the data frame:
       name  score qualify
a  Anastasia   12.5     yes
b      Dima    9.0      no
c  Katherine   16.5     yes
d      James    NaN      no
e      Emily    9.0      no
f    Michael   20.0     yes
g    Matthew   14.5     yes
h      Laura    NaN      no
i      Kevin    8.0      no
j      Jonas   19.0     yes
```

16. Write a Pandas program to insert a new column in existing DataFrame.

```
Sample Python dictionary data and list labels:
exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
```

```
Expected Output:
New DataFrame after inserting the 'color' column
   attempts       name qualify  score   color
a         1  Anastasia     yes   12.5     Red
b         3       Dima      no    9.0    Blue
.......
i         2      Kevin      no    8.0   Green
j         1      Jonas     yes   19.0     Red
```

[133]
```
import pandas as pd
import numpy as np

exam_data = {
    'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
    'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],
    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']
}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels)

df['color'] = ['Red', 'Blue', 'Green', 'Yellow', 'Blue', 'Red', 'Red', 'Yellow', 'Green', 'Red']

print("New DataFrame after inserting the 'color' column:")
print(df)
```

```
New DataFrame after inserting the 'color' column:
       name  score  attempts qualify   color
a  Anastasia   12.5         1     yes     Red
b      Dima    9.0         3      no    Blue
c  Katherine   16.5         2     yes   Green
d      James    NaN         3      no  Yellow
e      Emily    9.0         2      no    Blue
f    Michael   20.0         3     yes     Red
g    Matthew   14.5         1     yes     Red
h      Laura    NaN         1      no  Yellow
i      Kevin    8.0         2      no   Green
j      Jonas   19.0         1     yes     Red
```

17. Write a Pandas program to rename columns of a given DataFrame

```
Sample data:
Original DataFrame
   col1  col2  col3
0     1     4     7
1     2     5     8
2     3     6     9
New DataFrame after renaming columns:
```

```
   Column1  Column2  Column3
0        1        4        7
1        2        5        8
2        3        6        9
```

```
[134] import pandas as pd

      data = {'col1': [1, 2, 3], 'col2': [4, 5, 6], 'col3': [7, 8, 9]}
      df = pd.DataFrame(data)
      print("Original DataFrame:")
      print(df)

      df = df.rename(columns={'col1': 'Column1', 'col2': 'Column2', 'col3': 'Column3'})

      print("\nNew DataFrame after renaming columns:")
      print(df)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     8
2     3     6     9

New DataFrame after renaming columns:
   Column1  Column2  Column3
0        1        4        7
1        2        5        8
2        3        6        9
```

18. Write a Pandas program to select rows from a given DataFrame based on values in some columns.

```
Sample data:
Original DataFrame
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1
Rows for colum1 value == 4
   col1  col2  col3
1     4     5     8
3     4     7     0
```

```
[135] import pandas as pd

      data = {'col1': [1, 4, 3, 4, 5],
              'col2': [4, 5, 6, 7, 8],
              'col3': [7, 8, 9, 0, 1]}
      df = pd.DataFrame(data)
      print("Original DataFrame:")
      print(df)

      selected_rows = df[df['col1'] == 4]

      print("\nRows for 'col1' value == 4:")
      print(selected_rows)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1

Rows for 'col1' value == 4:
   col1  col2  col3
1     4     5     8
3     4     7     0
```

19. Write a Pandas program to add one row in an existing DataFrame.

```
Sample data:
Original DataFrame
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
```

```
4    5    8    1
After add one row:
   col1  col2  col3
0    1    4    7
1    4    5    8
2    3    6    9
3    4    7    0
4    5    8    1
5   10   11   12
```

```
[136] import pandas as pd

      data = {'col1': [1, 4, 3, 4, 5],
              'col2': [4, 5, 6, 7, 8],
              'col3': [7, 8, 9, 0, 1]}
      df = pd.DataFrame(data)

      print("Original DataFrame:")
      print(df)

      new_row = pd.Series([10, 11, 12], index=['col1', 'col2', 'col3'])
      df = df.append(new_row, ignore_index=True)

      print("\nAfter adding one row:")
      print(df)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1

After adding one row:
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1
5    10    11    12
<ipython-input-136-249750dae259>:12: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat in
  df = df.append(new_row, ignore_index=True)
```

20. Write a Pandas program to replace all the NaN values with Zero's in a column of a dataframe.

```
Sample data:
Original DataFrame

   attempts       name qualify  score
0         1  Anastasia     yes   12.5
1         3       Dima      no    9.0
2         2  Katherine     yes   16.5
.....
8         2      Kevin      no    8.0
9         1      Jonas     yes   19.0
```

```
New DataFrame replacing all NaN with 0:
   attempts       name qualify  score
0         1  Anastasia     yes   12.5
1         3       Dima      no    9.0
2         2  Katherine     yes   16.5
.....
8         2      Kevin      no    8.0
9         1      Jonas     yes   19.0
```

```
[137] import pandas as pd
      import numpy as np

      data = {'attempts': [1, 3, 2, np.nan, 2, 3, 1, 1, 2, 1],
              'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
              'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes'],
              'score': [12.5, 9.0, 16.5, np.nan, 9.0, 20.0, 14.5, np.nan, 8.0, 19.0]}

      df = pd.DataFrame(data)

      print("Original DataFrame:")
      print(df)

      df['score'] = df['score'].fillna(0)
```

```
print("\nNew DataFrame replacing all NaN with 0:")
print(df)
```

```
Original DataFrame:
   attempts       name qualify  score
0       1.0  Anastasia     yes   12.5
1       3.0       Dima      no    9.0
2       2.0  Katherine     yes   16.5
3       NaN      James      no    NaN
4       2.0      Emily      no    9.0
5       3.0    Michael     yes   20.0
6       1.0    Matthew     yes   14.5
7       1.0      Laura      no    NaN
8       2.0      Kevin      no    8.0
9       1.0      Jonas     yes   19.0

New DataFrame replacing all NaN with 0:
   attempts       name qualify  score
0       1.0  Anastasia     yes   12.5
1       3.0       Dima      no    9.0
2       2.0  Katherine     yes   16.5
3       NaN      James      no    0.0
4       2.0      Emily      no    9.0
5       3.0    Michael     yes   20.0
6       1.0    Matthew     yes   14.5
7       1.0      Laura      no    0.0
8       2.0      Kevin      no    8.0
9       1.0      Jonas     yes   19.0
```

21. Write a Pandas program to count the NaN values in one or more columns in DataFrame.

```
Sample data:
Original DataFrame
   attempts       name qualify  score
0         1  Anastasia     yes   12.5
1         3       Dima      no    9.0
2         2  Katherine     yes   16.5
3         3      James      no    NaN
4         2      Emily      no    9.0
5         3    Michael     yes   20.0
6         1    Matthew     yes   14.5
7         1      Laura      no    NaN
8         2      Kevin      no    8.0
9         1      Jonas     yes   19.0

Number of NaN values in one or more columns:
2
```

```
[138] import pandas as pd
      import numpy as np

      data = {'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
              'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
              'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes'],
              'score': [12.5, 9.0, 16.5, np.nan, 9.0, 20.0, 14.5, np.nan, 8.0, 19.0]}

      df = pd.DataFrame(data)

      print("Original DataFrame:")
      print(df)

      nan_count = df.isna().sum().sum()

      print("\nNumber of NaN values in one or more columns:", nan_count)
```

```
Original DataFrame:
   attempts       name qualify  score
0         1  Anastasia     yes   12.5
1         3       Dima      no    9.0
2         2  Katherine     yes   16.5
3         3      James      no    NaN
4         2      Emily      no    9.0
5         3    Michael     yes   20.0
6         1    Matthew     yes   14.5
7         1      Laura      no    NaN
8         2      Kevin      no    8.0
9         1      Jonas     yes   19.0

Number of NaN values in one or more columns: 2
```

22. Write a Pandas program to drop a list of rows from a specified DataFrame.

```
Sample data:
Original DataFrame
```

```
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1


New DataFrame after removing 2nd & 4th rows:
   col1  col2  col3
0     1     4     7
1     4     5     8
3     4     7     0
```

```
[139] import pandas as pd

      data = {'col1': [1, 4, 3, 4, 5],
              'col2': [4, 5, 6, 7, 8],
              'col3': [7, 8, 9, 0, 1]}

      df = pd.DataFrame(data)
      rows_to_remove = [1, 3]
      new_df = df.drop(rows_to_remove)

      print("Original DataFrame:")
      print(df)
      print("\nNew DataFrame after removing specified rows:")
      print(new_df)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     4     5     8
2     3     6     9
3     4     7     0
4     5     8     1

New DataFrame after removing specified rows:
   col1  col2  col3
0     1     4     7
2     3     6     9
4     5     8     1
```

23. . Write a Pandas program to convert DataFrame column type from string to datetime.

```
Sample data:
String Date:
0    3/11/2000
1    3/12/2000
2    3/13/2000
dtype: object


Original DataFrame (string to datetime):
           0
0 2000-03-11
1 2000-03-12
2 2000-03-13
```

```
[140] import pandas as pd

      string_dates = pd.Series(['3/11/2000', '3/12/2000', '3/13/2000'])
      datetime_dates = pd.to_datetime(string_dates, format='%m/%d/%Y')

      df = pd.DataFrame({0: datetime_dates})
      print("String Date:")
      print(string_dates)
      print("\nOriginal DataFrame (string to datetime):")
      print(df)
```

```
String Date:
0    3/11/2000
1    3/12/2000
2    3/13/2000
dtype: object

Original DataFrame (string to datetime):
           0
0 2000-03-11
1 2000-03-12
2 2000-03-13
```

24. Write a Pandas program to find the row for where the value of a given column is maximum.

```
Sample Output:
Original DataFrame
   col1  col2  col3
0     1     4     7
1     2     5     8
2     3     6    12
3     4     9     1
4     7     5    11
Row where col1 has maximum value:
4
Row where col2 has maximum value:
3
Row where col3 has maximum value:
2
```

```
[141] import pandas as pd
      data = {'col1': [1, 2, 3, 4, 7],
              'col2': [4, 5, 6, 9, 5],
              'col3': [7, 8, 12, 1, 11]}

      df = pd.DataFrame(data)
      print("Original DataFrame:")
      print(df)
      max_row_col1 = df['col1'].idxmax()
      max_row_col2 = df['col2'].idxmax()
      max_row_col3 = df['col3'].idxmax()

      print("\nRow where col1 has maximum value:")
      print(max_row_col1)

      print("\nRow where col2 has maximum value:")
      print(max_row_col2)

      print("\nRow where col3 has maximum value:")
      print(max_row_col3)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     8
2     3     6    12
3     4     9     1
4     7     5    11

Row where col1 has maximum value:
4

Row where col2 has maximum value:
3

Row where col3 has maximum value:
2
```

25. Write a Pandas program to get the datatypes of columns of a DataFrame.

```
Sample data:
Original DataFrame:
   attempts      name qualify  score
0         1  Anastasia     yes   12.5
1         3       Dima      no    9.0
.......
8         2     Kevin      no    8.0
9         1     Jonas     yes   19.0
Data types of the columns of the said DataFrame:
attempts      int64
name         object
qualify      object
score       float64
dtype: object
```

```
[142] import pandas as pd
      data = {'attempts': [1, 3, 2, 1],
              'name': ['Anastasia', 'Dima', 'Kevin', 'Jonas'],
              'qualify': ['yes', 'no', 'no', 'yes'],
              'score': [12.5, 9.0, 8.0, 19.0]}

      df = pd.DataFrame(data)

      print("Original DataFrame:")
      print(df)
```

```
column_data_types = df.dtypes
print("\nData types of the columns of the said DataFrame:")
print(column_data_types)
```

```
Original DataFrame:
   attempts       name qualify  score
0         1  Anastasia     yes   12.5
1         3       Dima      no    9.0
2         2      Kevin      no    8.0
3         1      Jonas     yes   19.0

Data types of the columns of the said DataFrame:
attempts      int64
name         object
qualify      object
score       float64
dtype: object
```

26. Write a Pandas program to group by the first column and get second column as lists in rows.

```
Sample data:
Original DataFrame
col1 col2
0 C1 1
1 C1 2
2 C2 3
3 C2 3
4 C2 4
5 C3 6
6 C2 5


Group on the col1:
col1
C1 [1, 2]
C2 [3, 3, 4, 5]
C3 [6]
Name: col2, dtype: object
```

```
[143] import pandas as pd
      data = {'col1': ['C1', 'C1', 'C2', 'C2', 'C2', 'C3', 'C2'],
              'col2': [1, 2, 3, 3, 4, 6, 5]}

      df = pd.DataFrame(data)
      grouped_data = df.groupby('col1')['col2'].agg(list)

      print("Original DataFrame:")
      print(df)

      print("\nGroup on the col1:")
      print(grouped_data)
```

```
Original DataFrame:
  col1  col2
0   C1     1
1   C1     2
2   C2     3
3   C2     3
4   C2     4
5   C3     6
6   C2     5

Group on the col1:
col1
C1          [1, 2]
C2    [3, 3, 4, 5]
C3             [6]
Name: col2, dtype: object
```

27 Write a Pandas program to count number of columns of a DataFrame.

```
Sample Output:
Original DataFrame
  col1 col2 col3
0  1    4   7
1  2    5   8
2  3    6   12
3  4    9   1
4  7    5   11

Number of columns:
3
```

```
[144] import pandas as pd

      data = {'col1': [1, 2, 3, 4, 7],
              'col2': [4, 5, 6, 9, 5],
              'col3': [7, 8, 12, 1, 11]}

      df = pd.DataFrame(data)
      print("Original DataFrame:")
      print(df)
      num_columns = df.shape[1]
      print("\nNumber of columns:")
      print(num_columns)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     8
2     3     6    12
3     4     9     1
4     7     5    11

Number of columns:
3
```

28. Write a Pandas program to get first n records of a DataFrame.

```
Sample Output:
Original DataFrame
col1 col2 col3
0 1 4 7
1 2 5 5
2 3 6 8
3 4 9 12
4 7 5 1
5 11 0 11

First 3 rows of the said DataFrame':
col1 col2 col3
0 1 4 7
1 2 5 5
2 3 6 8
```

```
[145] import pandas as pd
      data = {'col1': [1, 2, 3, 4, 7, 11],
              'col2': [4, 5, 6, 9, 5, 0],
              'col3': [7, 5, 8, 12, 1, 11]}

      df = pd.DataFrame(data)
      print("Original DataFrame:")
      print(df)
      first_n_rows = df.head(3)
      print("\nFirst 3 rows of the said DataFrame:")
      print(first_n_rows)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
3     4     9    12
4     7     5     1
5    11     0    11

First 3 rows of the said DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
```

29. Write a Pandas program to get last n records of a DataFrame.

```
Sample Output:
```

Runtime disconnected

```
col1 col2 col3
0 1 4 7
1 2 5 5
2 3 6 8
3 4 9 12
4 7 5 1
5 11 0 11

Last 3 rows of the said DataFrame':
```

```
Last 3 rows of the said DataFrame :
col1 col2 col3
3 4 9 12
4 7 5 1
5 11 0 11
```

```
[ ] import pandas as pd

    data = {'col1': [1, 2, 3, 4, 7, 11],
            'col2': [4, 5, 6, 9, 5, 0],
            'col3': [7, 5, 8, 12, 1, 11]}

    df = pd.DataFrame(data)

    print("Original DataFrame:")
    print(df)
    last_n_rows = df.tail(3)
    print("\nLast 3 rows of the said DataFrame:")
    print(last_n_rows)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
3     4     9    12
4     7     5     1
5    11     0    11

Last 3 rows of the said DataFrame:
   col1  col2  col3
3     4     9    12
4     7     5     1
5    11     0    11
```

30. Write a Pandas program to get topmost n records within each group of a DataFrame.

```
Sample Output:
Original DataFrame
col1 col2 col3
0 1 4 7
1 2 5 5
2 3 6 8
3 4 9 12
4 7 5 1
5 11 0 11
```

```
topmost n records within each group of a DataFrame:
col1 col2 col3
5 11 0 11
4 7 5 1
3 4 9 12
col1 col2 col3
3 4 9 12
2 3 6 8
1 2 5 5
4 7 5 1
col1 col2 col3
3 4 9 12
5 11 0 11
2 3 6 8
```

```
import pandas as pd

data = {
    'col1': [1, 2, 3, 4, 7, 11],
    'col2': [4, 5, 6, 9, 5, 0],
    'col3': [7, 5, 8, 12, 1, 11]
}

df = pd.DataFrame(data)
print("Original DataFrame:")
print(df)
topmost_records = df.groupby('col1').apply(lambda x: x.head(2))

print("\ntopmost n records within each group of a DataFrame:")
print(topmost_records)
```

```
Original DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
```

```
3     4     9    12
4     7     5     1
5    11     0    11

topmost n records within each group of a DataFrame:
   col1  col2  col3
0     1     4     7
1     2     5     5
2     3     6     8
3     4     9    12
4     7     5     1
5    11     0    11
<ipython-input-147-51aea9f91cfb>:12: FutureWarning: Not prepending group keys to the result index of transform-like apply. In the future, the group keys will be in
To preserve the previous behavior, use

        >>> .groupby(..., group_keys=False)

To adopt the future behavior and silence this warning, use

        >>> .groupby(..., group_keys=True)
  topmost_records = df.groupby('col1').apply(lambda x: x.head(2))
```

31. Write a Pandas program to add a prefix or suffix to all columns of a given DataFrame.

```
Sample Output:
Original DataFrame
W X Y Z
0 68 78 84 86
1 75 85 94 97
2 86 96 89 96
3 80 80 83 72
4 66 86 86 83

Add prefix:
A_W A_X A_Y A_Z
0 68 78 84 86
1 75 85 94 97
2 86 96 89 96
3 80 80 83 72
4 66 86 86 83

Add suffix:
W_1 X_1 Y_1 Z_1
0 68 78 84 86
1 75 85 94 97
2 86 96 89 96
3 80 80 83 72
4 66 86 86 83
```

```
[ ]  import pandas as pd

     data = {
         'W': [68, 75, 86, 80, 66],
         'X': [78, 85, 96, 80, 86],
         'Y': [84, 94, 89, 83, 86],
         'Z': [86, 97, 96, 72, 83]
     }

     df = pd.DataFrame(data)
     print("Original DataFrame:")
     print(df)
     df_prefixed = df.add_prefix('A_')
     print("\nAdd prefix:")
     print(df_prefixed)
     df_suffixed = df.add_suffix('_1')
     print("\nAdd suffix:")
     print(df_suffixed)
```

```
Original DataFrame:
    W   X   Y   Z
0  68  78  84  86
1  75  85  94  97
2  86  96  89  96
3  80  80  83  72
4  66  86  86  83

Add prefix:
   A_W  A_X  A_Y  A_Z
0   68   78   84   86
1   75   85   94   97
2   86   96   89   96
3   80   80   83   72
4   66   86   86   83

Add suffix:
   W_1  X_1  Y_1  Z_1
0   68   78   84   86
```

```
1   75   85   94   97
2   86   96   89   96
3   80   80   83   72
4   66   86   86   83
```

32. Write a Pandas program to convert continuous values of a column in a given DataFrame to categorical.

```
Input:
{ 'Name': ['Alberto Franco','Gino Mcneill','Ryan Parkes', 'Eesha Hinton', 'Syed Wharton'],
'Age': [18, 22, 40, 50, 80, 5] }


Output:
Age group:
0 kids
1 adult
2 elderly
3 adult
4 elderly
5 kids
Name: age_groups, dtype: category
Categories (3, object): [kids < adult < elderly]
```

```
[ ]  import pandas as pd

     data = {
         'Name': ['Alberto Franco', 'Gino Mcneill', 'Ryan Parkes', 'Eesha Hinton', 'Syed Wharton'],
         'Age': [18, 22, 40, 50, 80]
     }

     df = pd.DataFrame(data)
     bins = [0, 18, 35, 100]
     labels = ['kids', 'adult', 'elderly']
     df['age_groups'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)
     print("Age group:")
     print(df['age_groups'])
```

```
Age group:
0      adult
1      adult
2    elderly
3    elderly
4    elderly
Name: age_groups, dtype: category
Categories (3, object): ['kids' < 'adult' < 'elderly']
```

33. Write a Pandas program to append rows to an existing DataFrame and display the combined data.

```
Test Data:
tudent_data1
  student_id             name  marks
0         S1  Danniella Fenton    200
1         S2      Ryder Storey    210
2         S3      Bryce Jensen    190
3         S4         Ed Bernal    222
4         S5       Kwame Morin    199


New Row(s)
student_id                 S6
name          Scarlette Fisher
marks                      205
dtype: object
```

```
[ ]  import pandas as pd
     student_data1 = pd.DataFrame({
         'student_id': ['S1', 'S2', 'S3', 'S4', 'S5'],
         'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],
         'marks': [200, 210, 190, 222, 199]
     })

     new_row = pd.Series({'student_id': 'S6', 'name': 'Scarlette Fisher', 'marks': 205})
     combined_data = student_data1.append(new_row, ignore_index=True)
     print("Combined Data:")
     print(combined_data)
```

```
Combined Data:
  student_id              name  marks
0         S1  Danniella Fenton    200
1         S2      Ryder Storey    210
2         S3      Bryce Jensen    190
3         S4         Ed Bernal    222
4         S5       Kwame Morin    199
5         S6  Scarlette Fisher    205
<ipython-input-150-9885b687a90c>:9: FutureWarning: The frame.append method is deprecated and will be removed from pandas in a future version. Use pandas.concat ins
  combined_data = student_data1.append(new_row, ignore_index=True)
```

34 Write a Pandas program to join the two given dataframes along rows and merge with another dataframe along the common column id.

```
Test Data:
student_data1:
  student_id              name  marks
0         S1  Danniella Fenton    200
1         S2      Ryder Storey    210
2         S3      Bryce Jensen    190
3         S4         Ed Bernal    222
4         S5       Kwame Morin    199
student_data2:
  student_id              name  marks
0         S4  Scarlette Fisher    201
1         S5  Carla Williamson    200
2         S6       Dante Morse    198
3         S7    Kaiser William    219
4         S8   Madeeha Preston    201
```

```
exam_data:
   student_id  exam_id
0          S1       23
1          S2       45
2          S3       12
3          S4       67
4          S5       21
5          S7       55
6          S8       33
7          S9       14
8         S10       56
9         S11       83
10        S12       88
11        S13       12
```

```
[151] import pandas as pd
      student_data1 = pd.DataFrame({
          'student_id': ['S1', 'S2', 'S3', 'S4', 'S5'],
          'name': ['Danniella Fenton', 'Ryder Storey', 'Bryce Jensen', 'Ed Bernal', 'Kwame Morin'],
          'marks': [200, 210, 190, 222, 199]
      })

      student_data2 = pd.DataFrame({
          'student_id': ['S4', 'S5', 'S6', 'S7', 'S8'],
          'name': ['Scarlette Fisher', 'Carla Williamson', 'Dante Morse', 'Kaiser William', 'Madeeha Preston'],
          'marks': [201, 200, 198, 219, 201]
      })

      exam_data = pd.DataFrame({
          'student_id': ['S1', 'S2', 'S3', 'S4', 'S5', 'S7', 'S8', 'S9', 'S10', 'S11', 'S12', 'S13'],
          'exam_id': [23, 45, 12, 67, 21, 55, 33, 14, 56, 83, 88, 12]
      })
      students_combined = pd.concat([student_data1, student_data2])
      result = pd.merge(students_combined, exam_data, on='student_id')
      print("Combined Data:")
      print(students_combined)
      print("\nMerged Data:")
      print(result)
```

```
Combined Data:
  student_id              name  marks
0         S1  Danniella Fenton    200
1         S2      Ryder Storey    210
2         S3      Bryce Jensen    190
3         S4         Ed Bernal    222
4         S5       Kwame Morin    199
0         S4  Scarlette Fisher    201
1         S5  Carla Williamson    200
2         S6       Dante Morse    198
3         S7    Kaiser William    219
4         S8   Madeeha Preston    201
```

```
Merged Data:
  student_id             name  marks  exam_id
0         S1  Danniella Fenton    200       23
1         S2      Ryder Storey    210       45
2         S3      Bryce Jensen    190       12
3         S4         Ed Bernal    222       67
4         S4  Scarlette Fisher    201       67
5         S5       Kwame Morin    199       21
6         S5  Carla Williamson    200       21
7         S7    Kaiser William    219       55
8         S8   Madeeha Preston    201       33
```