* 1. Solution :

Python Code :

```
import pandas as pd
df = pd.DataFrame({'X':[78,85,96,80,86],
'Y':[84,94,89,83,86],'Z':[86,97,96,72,83]});
print(df)
```

Sample Output:

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

* 2. solution
 python code:

```
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)
```

Sample Output:

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

* 3. solution
Python Code :

```
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("Summary of the basic information about this DataFrame and its
data:")
print(df.info())
```

```
Sample Output:
Summary of the basic information about this DataFrame and its data:
Index: 10 entries, a to j
Data columns (total 4 columns):
attempts    10 non-null int64
name        10 non-null object
qualify     10 non-null object
score       8 non-null float64
dtypes: float64(1), int64(1), object(2)
memory usage: 400.0+ bytes
None
```

* 4. solution

```
Python Code :
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("First three rows of the data frame:")
print(df.iloc[:3])
```

```
Sample 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
```

* 5.solution

Python Code :

```
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("Select specific columns:")
print(df[['name', 'score']])
```

Sample Output:

```
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
```

* 6.solution

Python Code :

```
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("Select specific columns and rows:")
print(df.iloc[[1, 3, 5, 6], [1, 3]])
```

Sample Output:

```
Select specific columns and rows:
      name  score
b     Dima    9.0
d    James    NaN
f  Michael   20.0
g  Matthew   14.5
```

* 7.solution

Python Code :

```
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("Select specific columns and rows:")
print(df.ix[[1, 3, 5], ['name', 'score']])
```

Sample Output:

```
Select specific columns and rows:
      name  score
b     Dima    9.0
d    James    NaN
f  Michael   20.0
```

* 8. solution

Python Code :

```
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)
total_rows=len(df.axes[0])
total_cols=len(df.axes[1])
print("Number of Rows: "+str(total_rows))
```

```
print("Number of Columns: "+str(total_cols))
```

Sample Output:

```
Number of Rows: 10
Number of Columns: 4
```

* 9. solution

Python Code :

```
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("Rows where score is missing:")
print(df[df['score'].isnull()])
```

Sample Output:

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

* 10. solution

Python Code :

```
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("Rows where score between 15 and 20 (inclusive):")
print(df[df['score'].between(15, 20)])
Copy
```

Sample 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
```

* 11.solution

Python Code :

```
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("Rows where score between 15 and 20 (inclusive):")
print(df[(df['attempts'] < 3) & (df['score'] > 15)])
```

Sample Output:

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

* 12. solution

Python Code :

```
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("\nOriginal data frame:")
print(df)
print("\nChange the score in row 'd' to 11.5:")
df.loc['d', 'score'] = 11.5
print(df)
```

Sample Output:

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

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
d         3      James      no   11.5
e         2      Emily      no    9.0
f         3    Michael     yes   20.0
g         1    Matthew     yes   14.5
h         1      Laura      no    NaN
i         2      Kevin      no    8.0
j         1      Jonas     yes   19.0
```

* 13. solution

Python Code :

```
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("\nSum of the examination attempts by the students:")
print(df['attempts'].sum())
```

Sample Output:

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

* 14. solution

Python Code :

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

Copy

Sample Output:

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

* 15.solution

Python Code :

```
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("Orginal rows:")
print(df)
print("\nAppend a new row:")
df.loc['k'] = [1, 'Suresh', 'yes', 15.5]
print("Print all records after insert a new record:")
print(df)
print("\nDelete the new row and display the original  rows:")
df = df.drop('k')
print(df)
```

Sample Output:

```
Orginal rows:
   attempts       name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
c         2  Katherine     yes   16.5
d         3      James      no    NaN
e         2      Emily      no    9.0
f         3    Michael     yes   20.0
```

```
g         1    Matthew     yes   14.5
h         1      Laura      no    NaN
i         2      Kevin      no    8.0
j         1      Jonas     yes   19.0
```

* 16.solution

Python Code :

```
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("Orginal rows:")
print(df)
df.sort_values(by=['name', 'score'], ascending=[False, True])
print("Sort the data frame first by Â‘nameÂ’ in descending order, then by
Â‘scoreÂ’ in ascending order:")
print(df)
Copy
Sample Output:

Orginal rows:
   attempts       name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
c         2  Katherine     yes   16.5
d         3      James      no    NaN
e         2      Emily      no    9.0
f         3    Michael     yes   20.0
g         1    Matthew     yes   14.5
h         1      Laura      no    NaN
i         2      Kevin      no    8.0
j         1      Jonas     yes   19.0
Sort the data frame first by Â‘nameÂ’ in descending order, then by Â‘score
Â’ in ascending order:
   attempts       name qualify  score
a         1  Anastasia     yes   12.5
b         3       Dima      no    9.0
c         2  Katherine     yes   16.5
d         3      James      no    NaN
e         2      Emily      no    9.0
f         3    Michael     yes   20.0
g         1    Matthew     yes   14.5
h         1      Laura      no    NaN
i         2      Kevin      no    8.0
j         1      Jonas     yes   19.0
```

* 17.solution
Python Code :

```
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("Orginal rows:")
print(df)
print("\nReplace the Â‘qualify' column contains the values 'yes' and 'no'
with True and  False:")
df['qualify'] = df['qualify'].map({'yes': True, 'no': False})
print(df)
```

Copy
Sample Output:

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

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
c         2  Katherine     True   16.5
d         3      James    False    NaN
e         2      Emily    False    9.0
f         3    Michael     True   20.0
g         1    Matthew     True   14.5
h         1      Laura    False    NaN
i         2      Kevin    False    8.0
j         1      Jonas     True   19.0
```

* 18. solution

Python Code :

```
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("Orginal rows:")
print(df)
print("\nChange the name 'James' to Â‘SureshÂ’:")
df['name'] = df['name'].replace('James', 'Suresh')
print(df)
```

Sample Output:

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

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

* 19.solution

Python Code :

```
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("Orginal rows:")
print(df)
print("\nDelete the 'attempts' column from the data frame:")
df.pop('attempts')
print(df)
```

Sample Output:

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

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

* 20.solution

Python Code :

```
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("Orginal rows:")
print(df)
color =
['Red','Blue','Orange','Red','White','White','Blue','Green','Green','Red']
df['color'] = color
print("\nNew DataFrame after inserting the 'color' column")
print(df)
```

Sample Output:

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

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
c         2  Katherine     yes   16.5  Orange
d         3      James      no    NaN     Red
e         2      Emily      no    9.0   White
f         3    Michael     yes   20.0   White
g         1    Matthew     yes   14.5    Blue
h         1      Laura      no    NaN   Green
i         2      Kevin      no    8.0   Green
j         1      Jonas     yes   19.0     Red
```

* 21.solution
Python Code :

```
import pandas as pd
import numpy as np
exam_data = [{'name':'Anastasia', 'score':12.5},
{'name':'Dima','score':9}, {'name':'Katherine','score':16.5}]
df = pd.DataFrame(exam_data)
for index, row in df.iterrows():
    print(row['name'], row['score'])
```

Sample Output:

```
Anastasia 12.5
Dima 9.0
Katherine 16.5
```

* 22..solution
Python Code :

```
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(list(df.columns.values))
```

Sample Output:

```
['attempts', 'name', 'qualify', 'score']
```

* 23..solution
Python Code :

```
import pandas as pd
d = {'col1': [1, 2, 3], 'col2': [4, 5, 6], 'col3': [7, 8, 9]}
df = pd.DataFrame(data=d)
print("Original DataFrame")
print(df)
df.columns = ['Column1', 'Column2', 'Column3']
df = df.rename(columns={'col1': 'Column1', 'col2': 'Column2', 'col3':
'Column3'})
print("New DataFrame after renaming columns:")
print(df)
```

Sample Output:

```
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
```

* 24.solution
Python Code :

```
import pandas as pd
import numpy as np
d = {'col1': [1, 4, 3, 4, 5], 'col2': [4, 5, 6, 7, 8], 'col3': [7, 8, 9,
0, 1]}
df = pd.DataFrame(data=d)
print("Original DataFrame")
print(df)
print('After altering col1 and col3')
df = df[['col3', 'col2', 'col1']]
print(df)
```

Sample Output:

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

* 25.solution
Python Code :

```
import pandas as pd
import numpy as np
d = {'col1': [1, 4, 3, 4, 5], 'col2': [4, 5, 6, 7, 8], 'col3': [7, 8, 9,
0, 1]}
df = pd.DataFrame(data=d)
print("Original DataFrame")
print(df)
print('After add one row:')
df2 = {'col1': 10, 'col2': 11, 'col3': 12}
df = df.append(df2, ignore_index=True)
print(df)
```

Sample Output:

```
   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
```

* 26. solution

Python Code :

```
import pandas as pd
df1 = pd.DataFrame({'name': ['Anastasia', 'Dima', 'Katherine', 'James',
'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],
'city': ['California', 'Los Angeles', 'California', 'California',
'California', 'Los Angeles', 'Los Angeles', 'Georgia', 'Georgia', 'Los
Angeles']})
g1 = df1.groupby(["city"]).size().reset_index(name='Number of people')
print(g1)
```

Sample Output:

```
          city  Number of people
0   California                 4
1. Georgia                2
2. Los Angeles                 4
```