EXPERIMENT :11

AIM:-

Pandas program which will highlight the nan values..

CODE:-

```
\Users\ankal\Downloads\DSA0511-Query Processing\11.py (3.12.1)
Format  Run  Options  Window  Help
andas as pd
umpy as np
n.seed(24)
DataFrame({'A': np.linspace(1, 10, 10)})
concat([df, pd.DataFrame(np.random.randn(10, 4), columns=list('B
       axis=1)
0, 2] = np.nan
3, 3] = np.nan
4, 1] = np.nan
9, 4] = np.nan
riginal array:")
)
r_negative_red(val):
r = 'red' if val < 0 else 'black'
n 'color: %s' % color
nNegative numbers red and positive numbers black:")
.highlight_null(null_color='red')
```

INPUT:-

	A	B	C	D	E
	1	1.32921	nan	-0.31628	-0.99081
	2	-1.07082	-1.43871	0.564417	0.295722
	3	-1.6264	0.219565	0.678805	1.88927
	4	0.961538	0.104011	nan	0.850229
	5	nan	1.05774	0.165562	0.515018
	6	-1.33694	0.562861	1.39285	-0.063328
	7	0.121668	1.2076	-0.00204021	1.6278
	8	0.354493	1.03753	-0.385684	0.519818
	9	1.68658	-1.32596	1.42898	-2.08935
	10	-0.12982	0.631523	-0.586538	nan

OUTPUT:-

```
==================================================
Original array:
      A         B         C         D         E
0   1.0  1.329212       NaN -0.316280 -0.990810
1   2.0 -1.070816 -1.438713  0.564417  0.295722
2   3.0 -1.626404  0.219565  0.678805  1.889273
3   4.0  0.961538  0.104011       NaN  0.850229
4   5.0       NaN  1.057737  0.165562  0.515018
5   6.0 -1.336936  0.562861  1.392855 -0.063328
6   7.0  0.121668  1.207603 -0.002040  1.627796
7   8.0  0.354493  1.037528 -0.385684  0.519818
8   9.0  1.686583 -1.325963  1.428984 -2.089354
9  10.0 -0.129820  0.631523 -0.586538       NaN

Negative numbers red and positive numbers black:
```

EXPERIMENT :12

AIM:-

Pandas program to set dataframe background Color black and font color yellow

CODE:-

```
- C:\Users\ankal\Downloads\DSA0511-Query Processing\11.py (3.12.1)
t  Format  Run  Options  Window  Help
 pandas as pd
 numpy as np
dom.seed(24)
i.DataFrame({'A': np.linspace(1, 10, 10)})
i.concat([df, pd.DataFrame(np.random.randn(10, 4), columns=list('BCD
          axis=1)
:[0, 2] = np.nan
:[3, 3] = np.nan
:[4, 1] = np.nan
:[9, 4] = np.nan
"Original array:")
df)
lor_negative_red(val):
lor = 'red' if val < 0 else 'black'
turn 'color: %s' % color
"\nNegative numbers red and positive numbers black:")
le.highlight_null(null_color='red')
```

INPUT:-

	A	B	C	D	E
0	1	1.32921	nan	-0.31628	-0.99081
1	2	-1.07082	-1.43871	0.564417	0.295722
2	3	-1.6264	0.219565	0.678805	1.88927
3	4	0.961538	0.104011	nan	0.850229
4	5	nan	1.05774	0.165562	0.515018
5	6	-1.33694	0.562861	1.39285	-0.063328
6	7	0.121668	1.2076	-0.00204021	1.6278
7	8	0.354493	1.03753	-0.385684	0.519818
8	9	1.68658	-1.32596	1.42898	-2.08935
9	10	-0.12982	0.631523	-0.586538	nan

OUTPUT:-

```
==================================================
Original array:
      A         B         C         D         E
0   1.0  1.329212       NaN -0.316280 -0.990810
1   2.0 -1.070816 -1.438713  0.564417  0.295722
2   3.0 -1.626404  0.219565  0.678805  1.889273
3   4.0  0.961538  0.104011       NaN  0.850229
4   5.0       NaN  1.057737  0.165562  0.515018
5   6.0 -1.336936  0.562861  1.392855 -0.063328
6   7.0  0.121668  1.207603 -0.002040  1.627796
7   8.0  0.354493  1.037528 -0.385684  0.519818
8   9.0  1.686583 -1.325963  1.428984 -2.089354
9  10.0 -0.129820  0.631523 -0.586538       NaN

Negative numbers red and positive numbers black:
```

EXPERIMENT :13

AIM:-

Pandas program to detect missing values of a given DataFrame.
Display True or False

CODE:-

File Edit Format Run Options Window Help

```
import pandas as pd
import numpy as np

data = {'A': [1, 2, np.nan, 4], 'B': [np.nan, 2, 3, 4], 'C': [1, np.nan, 3, 4]}
df = pd.DataFrame(data)

missing_values = df.isnull()
print(missing_values)
```

INPUT

	ord_no	purch_amt	ord_date	customer_id	salesman_id
0	70001.0	150.50	2012-10-05	3002	5002.0
1	NaN	270.65	2012-09-10	3001	5003.0
2	70002.0	65.26	NaN	3001	5001.0
3	70004.0	110.50	2012-08-17	3003	NaN
4	NaN	948.50	2012-09-10	3002	5002.0
5	70005.0	2400.60	2012-07-27	3001	5001.0
6	NaN	5760.00	2012-09-10	3001	5001.0
7	70010.0	1983.43	2012-10-10	3004	NaN
8	70003.0	2480.40	2012-10-10	3003	5003.0
9	70012.0	250.45	2012-06-27	3002	5002.0
10	NaN	75.29	2012-08-17	3001	5003.0
11	70013.0	3045.60	2012-04-25	3001	NaN

OUTPUT

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/13.py =====
       A      B      C
0  False   True  False
1  False  False   True
2   True  False  False
3  False  False  False
>>>
```

EXPERIMENT :14

AIM

Pandas program to find and replace the missing values in a given DataFrame which do not have any valuable information

CODE

14.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/14.py (3.11.0)

File Edit Format Run Options Window Help

```
import pandas as pd
import numpy as np

data = {'A': [1, 2, np.nan, 4], 'B': [np.nan, 2, 3, 4], 'C': [1, np.nan, 3, 4]}
df = pd.DataFrame(data)

df_filled = df.fillna(0)
print(df_filled)
```

INPUT

	ord_no	purch_amt	ord_date	customer_id	salesman_id
0	70001	150.5	?	3002	5002
1	NaN	270.65	2012-09-10	3001	5003
2	70002	65.26	NaN	3001	?
3	70004	110.5	2012-08-17	3003	5001
4	NaN	948.5	2012-09-10	3002	NaN
5	70005	2400.6	2012-07-27	3001	5002
6	--	5760	2012-09-10	3001	5001
7	70010	?	2012-10-10	3004	?
8	70003	12.43	2012-10-10	--	5003
9	70012	2480.4	2012-06-27	3002	5002
10	NaN	250.45	2012-08-17	3001	5003
11	70013	3045.6	2012-04-25	3001	--

OUTPUT

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/14.py =====
     A    B    C
0  1.0  0.0  1.0
1  2.0  2.0  0.0
2  0.0  3.0  3.0
3  4.0  4.0  4.0
>>>
```

EXPERIMENT :15

AIM

Pandas program to keep the rows with at least 2 NaN values in a given DataFrame

CODE

15.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/15.py (3.11.0)

File Edit Format Run Options Window Help

```
import pandas as pd
import numpy as np

data = {'A': [1, 2, np.nan, 4], 'B': [np.nan, 2, 3, 4], 'C': [1, np.nan, 3, 4]}
df = pd.DataFrame(data)

rows_with_nans = df[df.isnull().sum(axis=1) >= 2]
print(rows_with_nans)
```

INPUT

	ord_no	purch_amt	ord_date	customer_id
0	NaN	NaN	NaN	NaN
1	NaN	270.65	2012-09-10	3001.0
2	70002.0	65.26	NaN	3001.0
3	NaN	NaN	NaN	NaN
4	NaN	948.50	2012-09-10	3002.0
5	70005.0	2400.60	2012-07-27	3001.0
6	NaN	5760.00	2012-09-10	3001.0
7	70010.0	1983.43	2012-10-10	3004.0
8	70003.0	2480.40	2012-10-10	3003.0
9	70012.0	250.45	2012-06-27	3002.0
10	NaN	75.29	2012-08-17	3001.0
11	NaN	NaN	NaN	NaN

OUTPUT

```
IDLE Shell 3.11.0
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/15.py =====
Empty DataFrame
Columns: [A, B, C]
Index: []
>>>
```

EXPERIMENT :16

AIM

Pandas program to split the following dataframe into groups based on school code. Also check the type of GroupBy object.

CODE

16.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/16.py (3.11.0)

File Edit Format Run Options Window Help

```
import pandas as pd

data = {'school_code': ['s1', 's2', 's1', 's3'],
        'name': ['A', 'B', 'C', 'D'],
        'age': [15, 16, 15, 17]}
df = pd.DataFrame(data)

grouped = df.groupby('school_code')
print(grouped)
```

INPUT

	school	class	name	date_Of_Birth	age	height	weight	address
S1	s001	V	Alberto Franco	15/05/2002	12	173	35	street1
S2	s002	V	Gino Mcneill	17/05/2002	12	192	32	street2
S3	s003	VI	Ryan Parkes	16/02/1999	13	186	33	street3
S4	s001	VI	Eesha Hinton	25/09/1998	13	167	30	street1
S5	s002	V	Gino Mcneill	11/05/2002	14	151	31	street2
S6	s004	VI	David Parkes	15/09/1997	12	159	32	street4

OUTPUT

```
IDLE Shell 3.11.0
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/16.py =====
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x000002087BD66690>
>>>
```

EXPERIMENT :17

AIM

Pandas program to split the following dataframe by school code and get mean, min, and max value of age for each school

CODE

17.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/17.py (3.11.0)

File Edit Format Run Options Window Help

```
import pandas as pd

data = {'school_code': ['s1', 's2', 's1', 's3'],
        'name': ['A', 'B', 'C', 'D'],
        'age': [15, 16, 15, 17]}
df = pd.DataFrame(data)

grouped_stats = df.groupby('school_code')['age'].agg(['mean', 'min', 'max'])
print(grouped_stats)
```

INPUT

	school	class	name	date_Of_Birth	age	height	weight	address
S1	s001	V	Alberto Franco	15/05/2002	12	173	35	street1
S2	s002	V	Gino Mcneill	17/05/2002	12	192	32	street2
S3	s003	VI	Ryan Parkes	16/02/1999	13	186	33	street3
S4	s001	VI	Eesha Hinton	25/09/1998	13	167	30	street1
S5	s002	V	Gino Mcneill	11/05/2002	14	151	31	street2
S6	s004	VI	David Parkes	15/09/1997	12	159	32	street4

OUTPUT

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/17.py =====
             mean  min  max
school_code
s1           15.0   15   15
s2           16.0   16   16
s3           17.0   17   17
>>>
```

EXPERIMENT :18

AIM

Pandas program to split the following given dataframe into groups based on school code and class.

CODE

18.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/18.py (3.11.0)

File Edit Format Run Options Window Help

```
import pandas as pd

data = {'school_code': ['s1', 's2', 's1', 's3'],
        'class': ['c1', 'c2', 'c1', 'c3'],
        'name': ['A', 'B', 'C', 'D'],
        'age': [15, 16, 15, 17]}
df = pd.DataFrame(data)
grouped = df.groupby(['school_code', 'class'])
print(grouped)
```

INPUT

	school	class	name	date_Of_Birth	age	height	weight	address
S1	s001	V	Alberto Franco	15/05/2002	12	173	35	street1
S2	s002	V	Gino Mcneill	17/05/2002	12	192	32	street2
S3	s003	VI	Ryan Parkes	16/02/1999	13	186	33	street3
S4	s001	VI	Eesha Hinton	25/09/1998	13	167	30	street1
S5	s002	V	Gino Mcneill	11/05/2002	14	151	31	street2
S6	s004	VI	David Parkes	15/09/1997	12	159	32	street4

OUTPUT

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/18.py =====
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x0000027C15958C50>
>>>
```

EXPERIMENT :19

AIM

Pandas program to display the dimensions or shape of the World alcohol consumption dataset. Also extract the column names from the dataset

CODE

19.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/19.py (3.11.0)

File Edit Format Run Options Window Help

```python
import pandas as pd

data = {
    'Country': ['USA', 'Canada', 'Germany', 'UK', 'France'],
    'Year': [1985, 1986, 1987, 1988, 1989],
    'Alcohol_Type': ['Beer', 'Wine', 'Spirits', 'Beer', 'Wine'],
    'Consumption': [100, 200, 150, 300, 250],
    'WHO_Region': ['Americas', 'Americas', 'Europe', 'Europe', 'Europe']
}
df = pd.DataFrame(data)

print("Shape of dataset:", df.shape)

print("Column names:", df.columns.tolist())
```

INPUT

	Year	WHO region	Country	Beverage Types	Display Value
0	1986	Western Pacific	Viet Nam	Wine	0.00
1	1986	Americas	Uruguay	Other	0.50
2	1985	Africa	Cte d'Ivoire	Wine	1.62
3	1986	Americas	Colombia	Beer	4.27
4	1987	Americas	Saint Kitts and Nevis	Beer	1.98

OUTPUT

IDLE Shell 3.11.0

File Edit Shell Debug Options Window Help

```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/19.py =====
Shape of dataset: (5, 5)
Column names: ['Country', 'Year', 'Alcohol_Type', 'Consumption', 'WHO_Region']
>>>
```

EXPERIMENT :20

AIM

Pandas program to find the index of a given substring of a DataFrame column

CODE

20.py - C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/20.py (3.11.0)

File Edit Format Run Options Window Help

```
import pandas as pd

data = {'text': ['Hello world', 'Pandas is great', 'Hello Pandas']}
df = pd.DataFrame(data)

df['substring_index'] = df['text'].str.find('Pandas')
print(df)
```

INPUT

DATA= [HELLO WORLD, PANDAS IS GREAT, HELLO PANDAS]

OUTPUT

```
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: C:/Users/Vishnu/Desktop/DSA0511-Query processing lab/20.py =====
            text  substring_index
0     Hello world             -1
1  Pandas is great             0
2    Hello Pandas              6
>>>
```