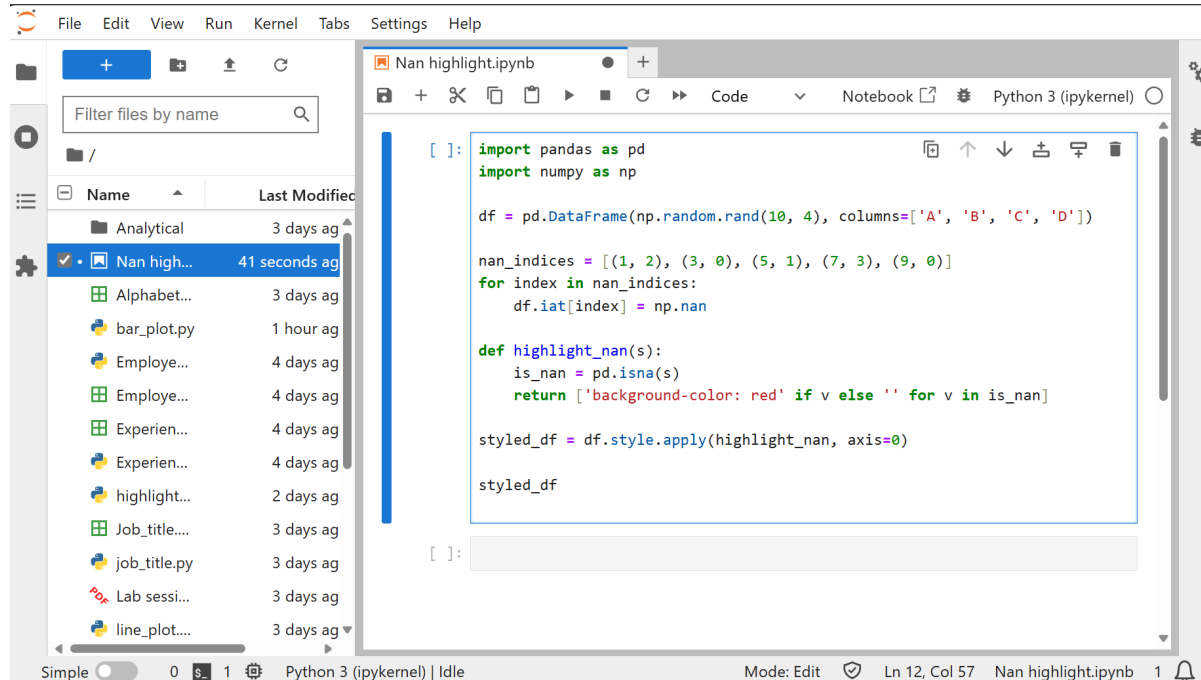


# Experiment 11

## Aim:

To develop a Pandas program to highlight Nan values.

## Code:



The screenshot shows a Jupyter Notebook with a file explorer on the left and a code editor on the right. The code in the notebook is as follows:

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

df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])

nan_indices = [(1, 2), (3, 0), (5, 1), (7, 3), (9, 0)]
for index in nan_indices:
    df.iat[index] = np.nan

def highlight_nan(s):
    is_nan = pd.isna(s)
    return ['background-color: red' if v else '' for v in is_nan]

styled_df = df.style.apply(highlight_nan, axis=0)

styled_df
```

## Input:

```
df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])
```

## Output:

[5]:

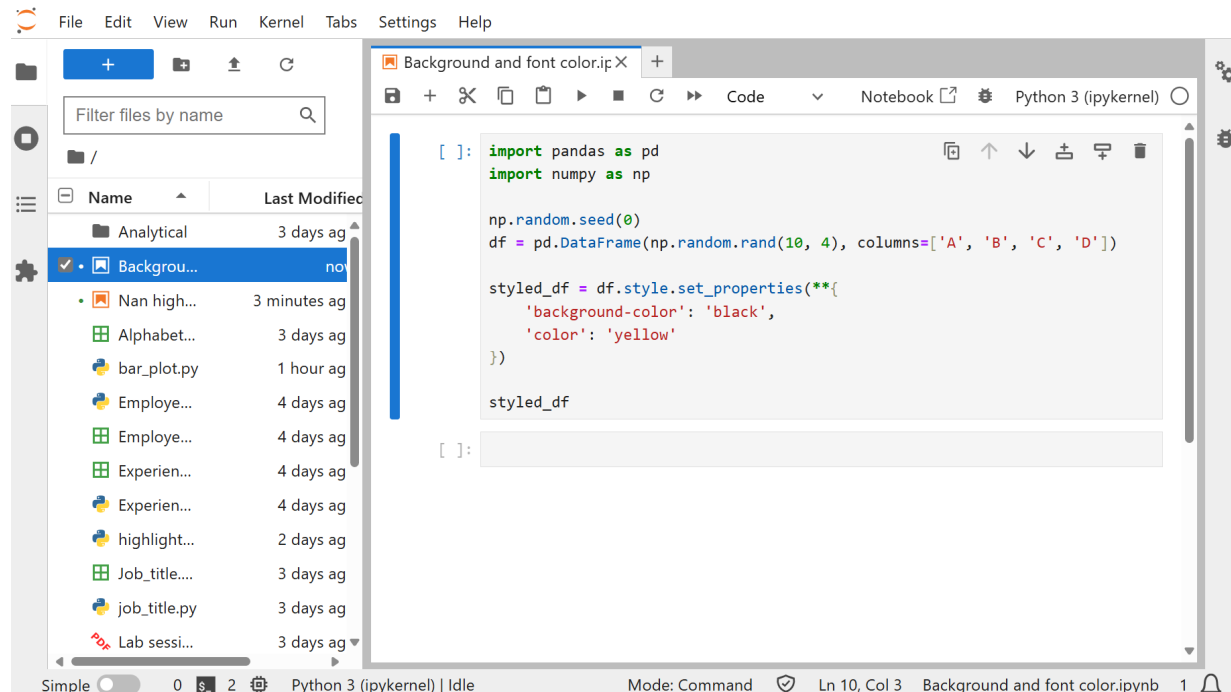
	A	B	C	D
0	0.770346	0.866002	0.387929	0.982454
1	0.141348	0.494301	nan	0.467524
2	0.508943	0.017011	0.932910	0.075325
3	nan	0.142506	0.782765	0.379285
4	0.717285	0.774044	0.693669	0.771902
5	0.038193	nan	0.387158	0.875157
6	0.085609	0.870489	0.253774	0.824059
7	0.412102	0.223124	0.394731	nan
8	0.844768	0.977309	0.995110	0.533583
9	nan	0.990504	0.003779	0.750633

# Experiment 12

## Aim:

To develop a Pandas program to set Data frame background Color black and font color yellow.

## Code:



The screenshot shows a Jupyter Notebook titled 'Background and font color.ipynb'. The code in the cell is as follows:

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

np.random.seed(0)
df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])

styled_df = df.style.set_properties(**{
    'background-color': 'black',
    'color': 'yellow'
})

styled_df
```

The output of the cell is a DataFrame with 10 rows and 4 columns (A, B, C, D). The background is black and the text is yellow.

## Input:

```
df = pd.DataFrame(np.random.rand(10, 4), columns=['A', 'B', 'C', 'D'])
```

## Output:

```
[2]:
```

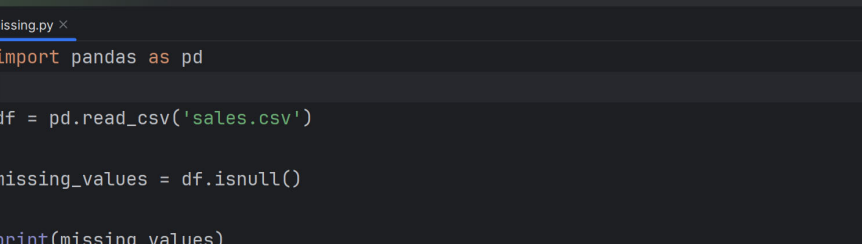
	A	B	C	D
0	0.548814	0.715189	0.602763	0.544883
1	0.423655	0.645894	0.437587	0.891773
2	0.963663	0.383442	0.791725	0.528895
3	0.568045	0.925597	0.071036	0.087129
4	0.020218	0.832620	0.778157	0.870012
5	0.978618	0.799159	0.461479	0.780529
6	0.118274	0.639921	0.143353	0.944669
7	0.521848	0.414662	0.264556	0.774234
8	0.456150	0.568434	0.018790	0.617635
9	0.612096	0.616934	0.943748	0.681820

## Experiment 13

**Aim:**

To develop a Pandas program to display missing values in a dataset as True  
else false

**Code:**



The screenshot shows a code editor with a dark theme. The menu bar at the top includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The toolbar below the menu has buttons for Codes, Version control, Current File, a play button, a bug icon, a vertical ellipsis, a user icon, a search icon, and a settings icon. The file explorer on the left shows a folder icon and a file named 'detect missing.py'. The code editor displays the following Python code:

```
1 import pandas as pd
2 |
3 df = pd.read_csv('sales.csv')
4
5 missing_values = df.isnull()
6
7 print(missing_values)
```

The status bar at the bottom shows 'Codes > detect missing.py', '2:1', 'CRLF', 'UTF-8', '4 spaces', and 'Python 3.12 (Codes)'.

**Input:**

sales - Excel

File Home Insert Draw Page Layout Formulas Data Review View Help Acrobat Tell me what you want to do

Paste Clipboard Font Alignment Number Conditional Formatting Styles Cell Styles Cells Editing Add-ins Create and Share Adobe PDF

P18 fx

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	ord_no	purch_amt	ord_date	customer_id	salesman_id														
2	70001	150.5	05-10-2012	3002	5002														
3	Nan	270.65	10-09-2012	3001	5003														
4	70002	65.26	Nan	3001	5001														
5	70004	110.5	17-08-2012	3003	Nan														
6	Nan	948.5	10-09-2012	3002	5002														
7	70005	2400.6	27-07-2012	3001	5001														
8	Nan	5760	10-09-2012	3001	5001														
9	70010	1983.43	10-10-2012	3004	Nan														
10	70003	2480.4	10-10-2012	3003	5003														
11	70012	250.45	27-06-2012	3002	5002														
12	Nan	75.29	17-08-2012	3001	5003														
13	70013	3045.6	25-04-2012	3001	Nan														
14																			
15																			
16																			
17																			
18																			
19																			
20																			
21																			
22																			
23																			
24																			
25																			
26																			
27																			
28																			

sales

Ready Accessibility: Unavailable

# Output:

Run detect missing x

↶

↷

⌵

⌶

⌷

"C:\Users\maano\_0waenfu\OneDrive\College\Query Processing\Codes\.venv\Scripts\python.exe" "C:\Us

	ord_no	purch_amt	ord_date	customer_id	salesman_id
0	False	False	False	False	False
1	True	False	False	False	False
2	False	False	True	False	False
3	False	False	False	False	True
4	True	False	False	False	False
5	False	False	False	False	False
6	True	False	False	False	False
7	False	False	False	False	False
8	False	False	False	False	False
9	False	False	False	False	False
10	True	False	False	False	False
11	False	False	False	False	True

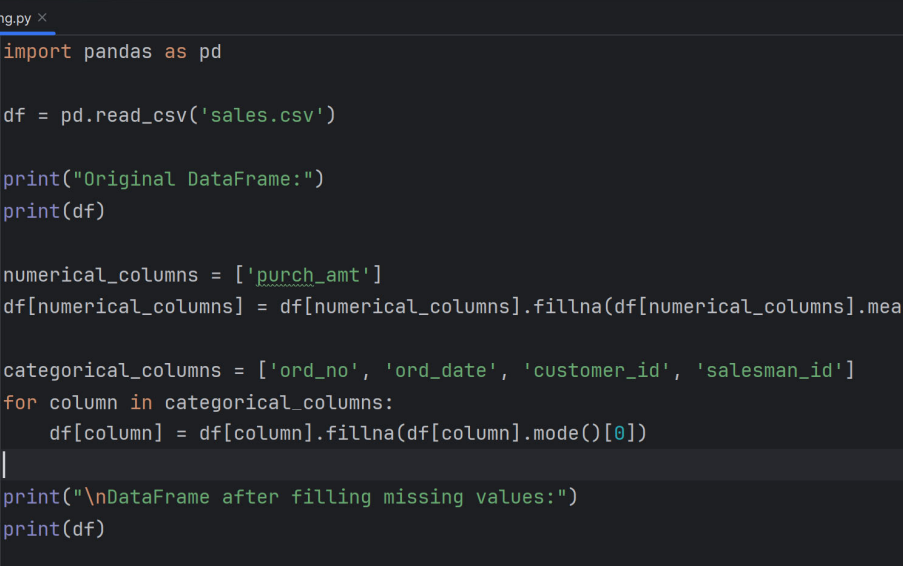
Process finished with exit code 0

## Experiment 14

### Aim:

To develop a Pandas program to fill missing values.

**Code:**



The screenshot shows a code editor with a dark theme. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. Below the menu bar, there are tabs for 'Codes' and 'Version control'. The main editor area displays a Python script named 'fill\_missing.py'. The script uses the pandas library to read a CSV file, handle missing values in numerical columns using fillna, and handle missing values in categorical columns using fillna with the mode. The script also prints the original DataFrame and the DataFrame after filling missing values.

```

1 import pandas as pd
2
3 df = pd.read_csv('sales.csv')
4
5 print("Original DataFrame:")
6 print(df)
7
8 numerical_columns = ['purch_amt']
9 df[numerical_columns] = df[numerical_columns].fillna(df[numerical_columns].mean())
10
11 categorical_columns = ['ord_no', 'ord_date', 'customer_id', 'salesman_id']
12 for column in categorical_columns:
13     df[column] = df[column].fillna(df[column].mode()[0])
14
15 print("\nDataFrame after filling missing values:")
16 print(df)
17

```

The bottom status bar shows the current file is 'fill\_missing.py', the encoding is 'UTF-8', the line length is '14:1', and the Python version is 'Python 3.12 (Codes)'.

**Input:**

sales - Excel

File Home Insert Draw Page Layout Formulas Data Review View Help Acrobat Tell me what you want to do

Paste Font Alignment Number Conditional Formatting Styles Cell Styles Cells Editing Sort & Find & Filter Select Add-ins Create and Share Adobe PDF

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	ord_no	purch_amt	ord_date	customer_id	salesman_id														
2	70001	150.5	05-10-2012	3002	5002														
3	Nan	270.65	10-09-2012	3001	5003														
4	70002	65.26	Nan	3001	5001														
5	70004	110.5	17-08-2012	3003	Nan														
6	Nan	948.5	10-09-2012	3002	5002														
7	70005	2400.6	27-07-2012	3001	5001														
8	Nan	5760	10-09-2012	3001	5001														
9	70010	1983.43	10-10-2012	3004	Nan														
10	70003	2480.4	10-10-2012	3003	5003														
11	70012	250.45	27-06-2012	3002	5002														
12	Nan	75.29	17-08-2012	3001	5003														
13	70013	3045.6	25-04-2012	3001	Nan														
14																			
15																			
16																			
17																			
18																			
19																			
20																			
21																			
22																			
23																			
24																			
25																			
26																			
27																			
28																			

Ready | sales | Accessible | Unavailable

# Output:

Run fill\_missing x

↑

↓

↺

↻

🖨

🗑

10

NaN

75.29

17-08-2012

3001

5003

11

70013.0

3045.60

25-04-2012

3001

NaN

DataFrame after filling missing values:

	ord_no	purch_amt	ord_date	customer_id	salesman_id
0	70001.0	150.50	05-10-2012	3002	5002
1	70001.0	270.65	10-09-2012	3001	5003
2	70002.0	65.26	10-09-2012	3001	5001
3	70004.0	110.50	17-08-2012	3003	5001
4	70001.0	948.50	10-09-2012	3002	5002
5	70005.0	2400.60	27-07-2012	3001	5001
6	70001.0	5760.00	10-09-2012	3001	5001
7	70010.0	1983.43	10-10-2012	3004	Nan
8	70003.0	2480.40	10-10-2012	3003	5003
9	70012.0	250.45	27-06-2012	3002	5002
10	70001.0	75.29	17-08-2012	3001	5003
11	70013.0	3045.60	25-04-2012	3001	5001

Process finished with exit code 0

## Experiment 15

### Aim:

To develop a Pandas program to display missing values in a dataset as True else false

**Code:**

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Codes Version control Current File
2 Nan values.py
1 import pandas as pd
2
3 df = pd.read_csv('sales.csv')
4
5 print("Original DataFrame:")
6 print(df)
7
8 df_filtered = df[df.isna().sum(axis=1) >= 2]
9
10 print("\nDataFrame with at least 2 NaN values:")
11 print(df_filtered)
12
13 df_filtered.to_csv('filtered_sales.csv', index=False)
14
```

**Input:**

sales - Excel

File Home Insert Draw Page Layout Formulas Data Review View Help Acrobat Tell me what you want to do

Calibri 11 A<sup>+</sup>

B I U Font

Alignment Number

General

Conditional Formatting Styles

Format as Table

Cell Styles

Insert Delete Format Cells

Σ 2 Sort & Find & Filter Select Editing

Add-ins

Create and Share Adobe PDF

Adobe Acrobat

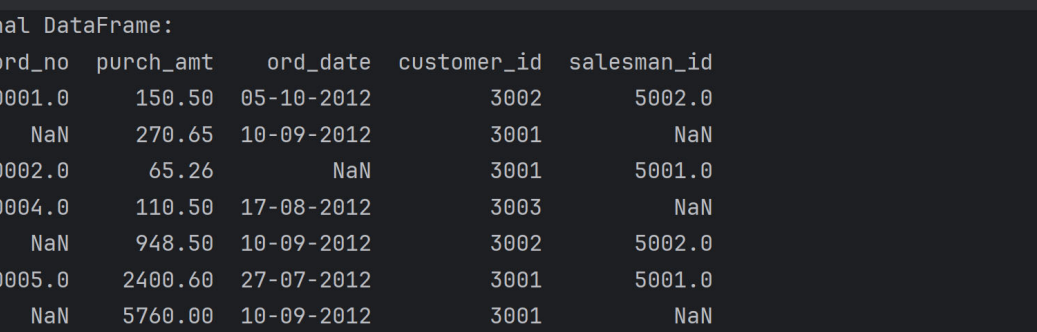
N6

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
	ord_no	purch_amord_date	customer_id	salesman_id															
1																			
2	70001	150.5	05-10-2012	3002	5002														
3		270.65	10-09-2012	3001															
4	70002	65.26		3001	5001														
5	70004	110.5	17-08-2012	3003															
6		948.5	10-09-2012	3002	5002														
7	70005	2400.6	27-07-2012	3001	5001														
8		5760	10-09-2012	3001															
9	70010	1983.43	10-10-2012	3004	5001														
10	70003	2480.4	10-10-2012	3003	5003														
11	70012	250.45	27-06-2012	3002	5002														
12		75.29	17-08-2012	3001															
13	70013	3045.6	25-04-2012	3001	5003														
14																			
15																			
16																			
17																			
18																			
19																			
20																			
21																			
22																			
23																			
24																			
25																			
26																			
27																			
28																			

sales

Ready Accessibility Unavailable

## Output:



The screenshot shows a Jupyter Notebook interface with a dark theme. At the top, a status bar indicates 'Run' and '2 Nan values'. The main area displays a pandas DataFrame titled 'Original DataFrame:'. The DataFrame has six columns: 'ord\_no', 'purch\_amt', 'ord\_date', 'customer\_id', and 'salesman\_id'. The rows are indexed from 0 to 11. Some cells contain NaN values, specifically in the 'ord\_no' and 'salesman\_id' columns for rows 1, 4, 6, and 10. Below the original DataFrame, a new DataFrame is shown, titled 'DataFrame with at least 2 NaN values:'. This filtered DataFrame contains only the rows from the original DataFrame that have at least two NaN values, which are rows 1, 6, and 10. The columns are the same as the original DataFrame.

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

	ord_no	purch_amt	ord_date	customer_id	salesman_id
1	NaN	270.65	10-09-2012	3001	NaN
6	NaN	5760.00	10-09-2012	3001	NaN
10	NaN	75.29	17-08-2012	3001	NaN