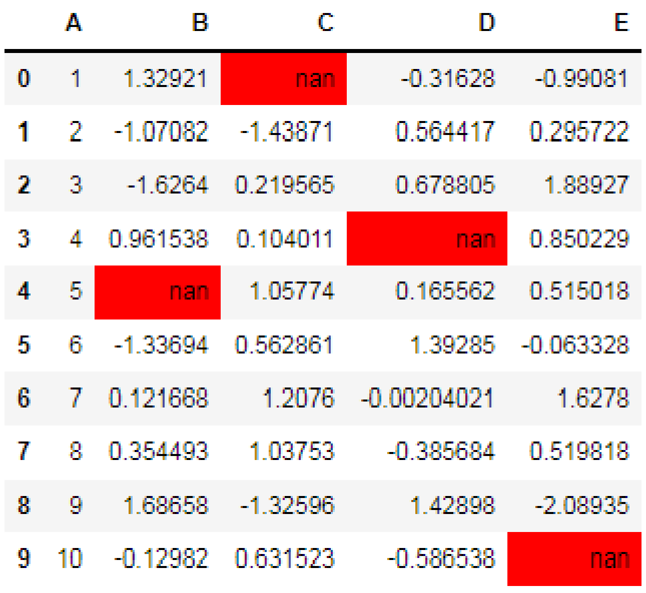
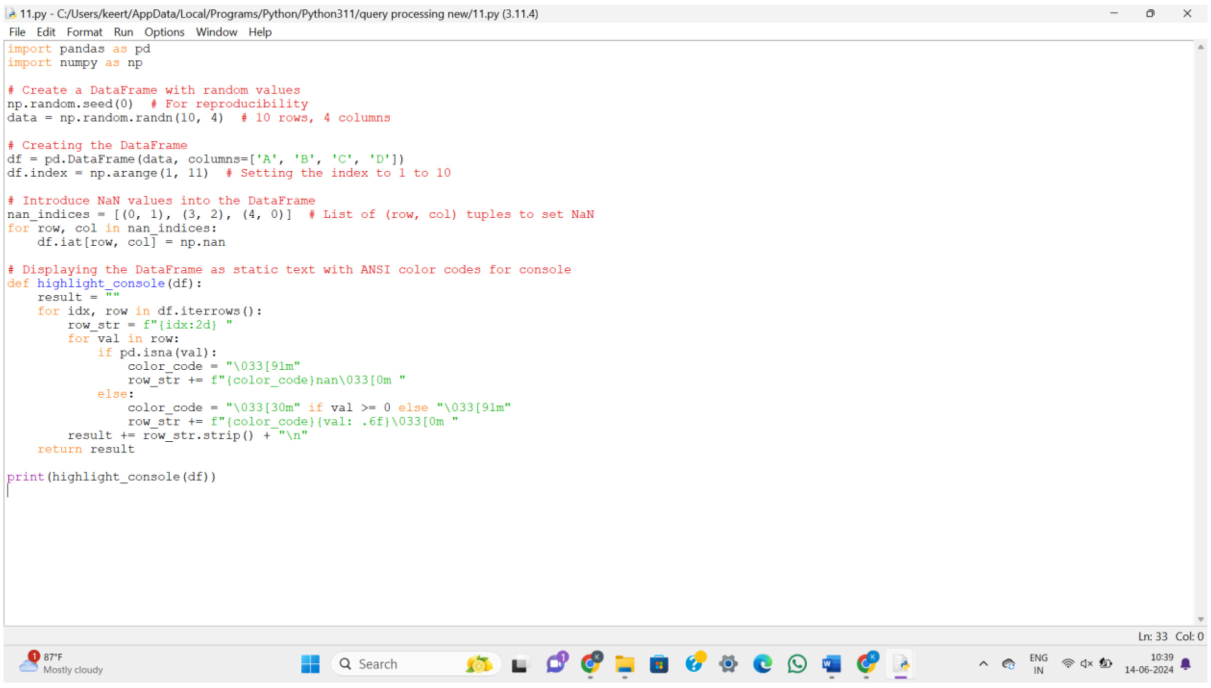
Create a dataframe of ten rows, four columns with random values. Convert some values to nan values. Write a Pandas program which will highlight the nan values.



	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

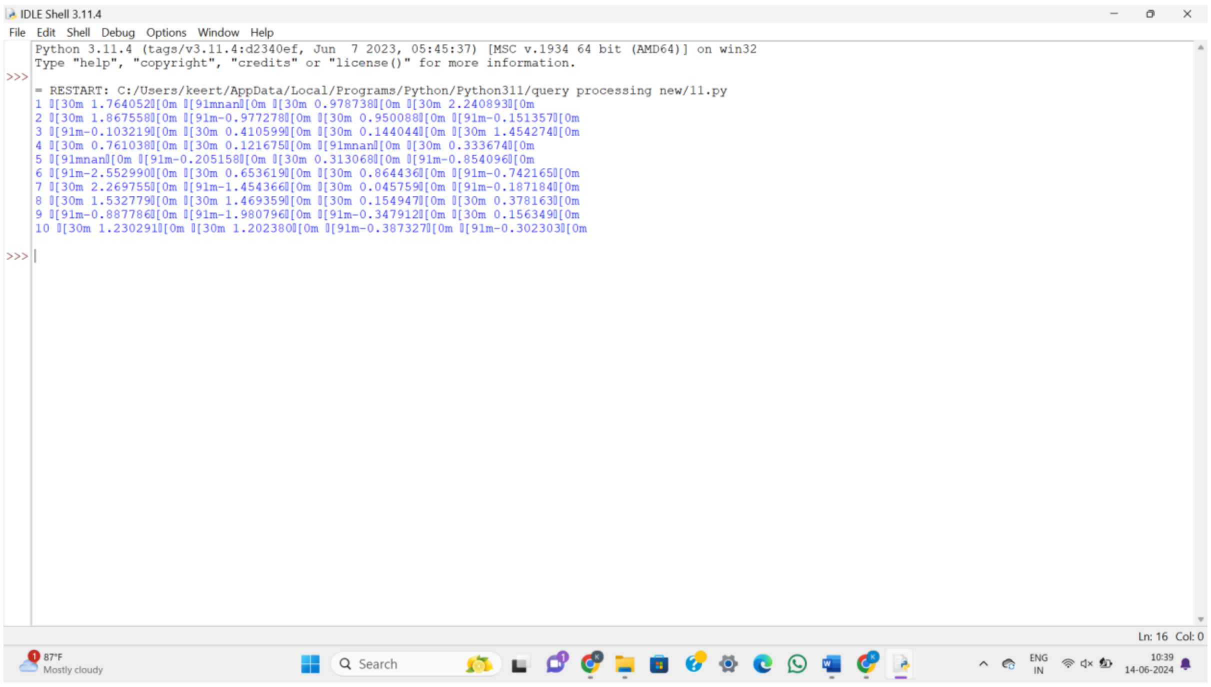
CODE:



11.py - C:/Users/keert/AppData/Local/Programs/Python/Python311/query processing new/11.py (3.11.4)

```python
import pandas as pd
import numpy as np

# Create a DataFrame with random values
np.random.seed(0)  # For reproducibility
data = np.random.randn(10, 4)  # 10 rows, 4 columns

# Creating the DataFrame
df = pd.DataFrame(data, columns=['A', 'B', 'C', 'D'])
df.index = np.arange(1, 11)  # Setting the index to 1 to 10

# Introduce NaN values into the DataFrame
nan_indices = [(0, 1), (3, 2), (4, 0)]  # List of (row, col) tuples to set NaN
for row, col in nan_indices:
    df.iat[row, col] = np.nan

# Displaying the DataFrame as static text with ANSI color codes for console
def highlight_console(df):
    result = ""
    for idx, row in df.iterrows():
        row_str = f"{idx:2d} "
        for val in row:
            if pd.isna(val):
                color_code = "\033[91m"
                row_str += f"{color_code}nan\033[0m "
            else:
                color_code = "\033[30m" if val >= 0 else "\033[91m"
                row_str += f"{color_code}{val: .6f}\033[0m "
        result += row_str.strip() + "\n"
    return result

print(highlight_console(df))
```

OUTPUT:



```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/keert/AppData/Local/Programs/Python/Python311/query processing new/11.py
1 ▯[30m 1.764052▯[0m ▯[91mnan▯[0m ▯[30m 0.978738▯[0m ▯[30m 2.240893▯[0m
2 ▯[30m 1.867558▯[0m ▯[91m-0.977278▯[0m ▯[30m 0.950088▯[0m ▯[91m-0.151357▯[0m
3 ▯[91m-0.103219▯[0m ▯[30m 0.410599▯[0m ▯[30m 0.144044▯[0m ▯[30m 1.454274▯[0m
4 ▯[30m 0.761038▯[0m ▯[30m 0.121675▯[0m ▯[91mnan▯[0m ▯[30m 0.333674▯[0m
5 ▯[91mnan▯[0m ▯[91m-0.205158▯[0m ▯[30m 0.313068▯[0m ▯[91m-0.854096▯[0m
6 ▯[91m-2.552990▯[0m ▯[30m 0.653619▯[0m ▯[30m 0.864436▯[0m ▯[91m-0.742165▯[0m
7 ▯[30m 2.269755▯[0m ▯[91m-1.454366▯[0m ▯[30m 0.045759▯[0m ▯[91m-0.187184▯[0m
8 ▯[30m 1.532779▯[0m ▯[30m 1.469359▯[0m ▯[30m 0.154947▯[0m ▯[30m 0.378163▯[0m
9 ▯[91m-0.887786▯[0m ▯[91m-1.980796▯[0m ▯[91m-0.347912▯[0m ▯[30m 0.156349▯[0m
10 ▯[30m 1.230291▯[0m ▯[30m 1.202380▯[0m ▯[91m-0.387327▯[0m ▯[91m-0.302303▯[0m

>>> |
```