Perfect! Let’s go **step by step**, cleaning your code, explaining each operation, and including **expected output** at each step. I’ll use a small example DataFrame so outputs are clear.

Step 0: Setup

```
import pandas as pd
import numpy as np

# Sample DataFrame (simulating an Excel read)
df = pd.DataFrame({
    'Name': ['Aarav', 'Diya', 'Kabir', 'Riya', 'Raj'],
    'Score': [85, 92, 78, 88, 75],
    'Sales': [5000, 6000, 4000, 5500, 4500],
    'Region': ['North', 'South', 'East', 'West', 'South']
})
```

Output:

```
     Name  Score  Sales Region
0   Aarav     85   5000  North
1    Diya     92   6000  South
2   Kabir     78   4000   East
3    Riya     88   5500   West
4     Raj     75   4500  South
```

1. Inspecting Data

```
df.head()   # First 5 rows
```

Output:

```
     Name  Score  Sales Region
0   Aarav     85   5000  North
1    Diya     92   6000  South
2   Kabir     78   4000   East
3    Riya     88   5500   West
4     Raj     75   4500  South
```

```
df.tail()   # Last 5 rows (same here)
```

Output: Same as above.

```
df.shape
```

Output:

```
(5, 4)
```

```
df.columns
```

Output:

```
Index(['Name', 'Score', 'Sales', 'Region'], dtype='object')
df.info()
```
Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5 entries, 0 to 4
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Name    5 non-null      object
 1   Score   5 non-null      int64
 2   Sales   5 non-null      int64
 3   Region  5 non-null      object
dtypes: int64(2), object(2)
memory usage: 288.0+ bytes
df.describe()
```
Output:

```
           Score        Sales
count   5.000000     5.000000
mean   83.600000  5000.000000
std     6.817033   707.106781
min    75.000000  4000.000000
25%    78.000000  4500.000000
50%    85.000000  5000.000000
75%    88.000000  5500.000000
max    92.000000  6000.000000
```

2. Selecting Columns and Rows

```
df['Score']       # Single column
```
Output:

```
0    85
1    92
2    78
3    88
4    75
Name: Score, dtype: int64
df[['Name', 'Sales']]  # Multiple columns
```
Output:

```
    Name  Sales
0  Aarav   5000
1   Diya   6000
2  Kabir   4000
```

```
3     Riya   5500
4      Raj   4500
df.iloc[0]   # First row by index
```
Output:

```
Name      Aarav
Score        85
Sales      5000
Region    North
Name: 0, dtype: object
df.loc[0]    # First row by label
```
Output: Same as above.

```
df.iloc[0:3, 0:2]   # Slice first 3 rows, first 2 columns
```
Output:

```
     Name  Score
0   Aarav     85
1    Diya     92
2   Kabir     78
```

3. Modifying Columns and Rows

```
df['Total'] = df['Score'] + df['Sales']  # Create new column
```
Output:

```
      Name  Score  Sales Region  Total
0    Aarav     85   5000  North   5085
1     Diya     92   6000  South   6092
2    Kabir     78   4000   East   4078
3     Riya     88   5500   West   5588
4      Raj     75   4500  South   4575
df.drop('Total', axis=1, inplace=True)   # Drop column
```
- Column `'Total'` removed.

```
df.drop([0,1], axis=0)   # Drop first two rows (without
modifying df permanently)
```
Output:

```
    Name  Score  Sales Region
2  Kabir     78   4000   East
3   Riya     88   5500   West
4    Raj     75   4500  South
df.rename(columns={'Score': 'Marks'}, inplace=False)
```
Output:

```
      Name  Marks  Sales Region
```

```
0    Aarav     85   5000  North
1     Diya     92   6000  South
2    Kabir     78   4000   East
3     Riya     88   5500   West
4      Raj     75   4500  South
```

4. Handling Missing Values

```
df.fillna(0)    # Replace NaN with 0
df.dropna()     # Remove rows with NaN
```

- No NaN in this example, so DataFrame remains the same.

5. Sorting and Filtering

```
df.sort_values(by='Score', ascending=False)
```

Output:

```
     Name  Score  Sales Region
1    Diya     92   6000  South
3    Riya     88   5500   West
0   Aarav     85   5000  North
2  Kabir     78   4000   East
4     Raj     75   4500  South
```

```
df[df['Region'] == 'South']
```

Output:

```
   Name  Score  Sales Region
1  Diya     92   6000  South
4   Raj     75   4500  South
```

```
df[(df['Score'] > 80) & (df['Name'] == 'Diya')]
```

Output:

```
   Name  Score  Sales Region
1  Diya     92   6000  South
```

6. Conditional Column Creation

```
df['Result'] = df['Score'].apply(lambda x: 'Pass' if x >= 80
else 'Fail')
```

Output:

```
      Name  Score  Sales Region Result
0    Aarav     85   5000  North   Pass
1     Diya     92   6000  South   Pass
```

```
2    Kabir      78   4000   East   Fail
3     Riya      88   5500   West   Pass
4      Raj      75   4500  South   Fail
```

```
df['Grade'] = np.where(df['Score'] >= 90, 'A', 'B')
```

Output:

```
      Name  Score  Sales Region Result Grade
0    Aarav     85   5000  North   Pass     B
1     Diya     92   6000  South   Pass     A
2    Kabir     78   4000   East   Fail     B
3     Riya     88   5500   West   Pass     B
4      Raj     75   4500  South   Fail     B
```

7. Replacing Values

```
df['Name'] = df['Name'].replace('Aarav', 'Aarav Gupta')
```

Output:

```
         Name  Score  Sales Region Result Grade
0  Aarav Gupta     85   5000  North   Pass     B
1         Diya     92   6000  South   Pass     A
2        Kabir     78   4000   East   Fail     B
3         Riya     88   5500   West   Pass     B
4          Raj     75   4500  South   Fail     B
```

8. Updating Values Conditionally

```
df.loc[df['Score'] < 80, 'Score'] += 5
```

Output:

```
         Name  Score  Sales Region Result Grade
0  Aarav Gupta     85   5000  North   Pass     B
1         Diya     92   6000  South   Pass     A
2        Kabir     83   4000   East   Fail     B
3         Riya     88   5500   West   Pass     B
4          Raj     80   4500  South   Fail     B
```

Everything above is **stepwise, with output for each step**, using a clear example DataFrame.

If you want, I can also make a **visual “pandas workflow diagram” showing all operations: reading → inspecting → selecting → modifying → filtering → conditional columns → updating**, which makes it extremely easy to memorize.

Do you want me to make that diagram?