

Theory Activity No. 1

Name : Karan Ugale

Subject : EDS

PRN : 202401070135

Roll No. : ET1-62

Batch : E14

- Import Required Libraries

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

- Dataset

```
data = {
    'Paper_ID': range(1, 11),
    'Title': ['Paper A', 'Paper B', 'Paper C', 'Paper D', 'Paper E', 'Paper F', 'Paper G', 'Paper H', 'Paper I', 'Paper J'],
    'Author(s)': ['Author1', 'Author2', 'Author1', 'Author3', 'Author4', 'Author5', 'Author2', 'Author6', 'Author7', 'Author3'],
    'Submission_Date': pd.to_datetime(['2025-03-30', '2025-04-01', '2025-04-02', '2025-03-28', '2025-03-31', '2025-04-03', '2025-03-27', '2025-04-04', '2025-03-29', '2025-04-05']),
    'Review_Score': [8, 7, 9, 6, 5, 8, 7, 9, 6, 5],
    'Reviewer_Comments': ['Good work', 'Needs improvement', 'Excellent', 'Average', 'Poor', 'Well done', 'Good novelty', 'Very good', 'Could be better', 'Interesting concept'],
    'Decision': ['Accepted', 'Rejected', 'Accepted', 'Rejected', 'Rejected', 'Accepted', 'Accepted', 'Accepted', 'Rejected', 'Rejected'],
    'Keywords': ['AI, ML', 'IoT', 'AI, Robotics', 'ML, Security', 'Blockchain', 'IoT, AI', 'ML', 'Cybersecurity', 'IoT, Blockchain', 'Security'],
    'No_of_Pages': [8, 10, 12, 9, 11, 7, 8, 10, 13, 9],
    'Plagiarism_Score': [10, 35, 5, 20, 40, 8, 15, 10, 50, 25]
}

df = pd.DataFrame(data)
df
```

	Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score
0	1	Paper A	Author1	2025-03-30	8	Good work	Accepted	AI, ML	8	10
1	2	Paper B	Author2	2025-04-01	7	Needs improvement	Rejected	IoT	10	35
2	3	Paper C	Author1	2025-04-02	9	Excellent	Accepted	AI, Robotics	12	5
3	4	Paper D	Author3	2025-03-28	6	Average	Rejected	ML, Security	9	20
4	5	Paper E	Author4	2025-03-31	5	Poor	Rejected	Blockchain	11	40
5	6	Paper F	Author5	2025-04-03	8	Well done	Accepted	IoT, AI	7	8
6	7	Paper G	Author2	2025-03-27	7	Good novelty	Accepted	ML	8	15
7	8	Paper H	Author6	2025-04-01	9	Very good	Accepted	Cybersecurity	10	10
8	9	Paper I	Author7	2025-04-04	6	Could be better	Rejected	IoT, Blockchain	13	50
9	10	Paper J	Author3	2025-03-29	5	Interesting concept	Rejected	Security	9	25

1) Find the number of papers submitted

```
df.shape[0]
```

```
10
```

2) Calculate the average review score

```
df['Review_Score'].mean()
```

```
np.float64(7.0)
```

3) Find the paper(s) with the highest review score

```
df[df['Review_Score'] == df['Review_Score'].max()]
```

	Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score
2	3	Paper C	Author1	2025-04-02	9	Excellent	Accepted	AI, Robotics	12	5
7	8	Paper H	Author6	2025-04-01	9	Very good	Accepted	Cybersecurity	10	10

4) Find the number of papers accepted

```
df[df['Decision'] == 'Accepted'].shape[0]
```

```
5
```

5) List the titles of papers that were rejected

```
df[df['Decision'] == 'Rejected']['Title']
```

```
Title
1  Paper B
3  Paper D
4  Paper E
8  Paper I
9  Paper J
dtype: object
```

6) Find the percentage of papers with plagiarism score above 30%

```
(df[df['Plagiarism_Score'] > 30].shape[0] / df.shape[0]) * 100
```

```
30.0
```

7) Average number of pages for accepted papers

```
df[df['Decision'] == 'Accepted']['No_of_Pages'].mean()

np.float64(9.0)
```

8) Identify papers submitted after the deadline (2025-04-01)

```
df[df['Submission_Date'] > '2025-04-01']
```

	Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score
2	3	Paper C	Author1	2025-04-02	9	Excellent	Accepted	AI, Robotics	12	5
5	6	Paper F	Author5	2025-04-03	8	Well done	Accepted	IoT, AI	7	8
8	9	Paper I	Author7	2025-04-04	6	Could be better	Rejected	IoT, Blockchain	13	50

9) Find top 5 papers with the lowest plagiarism scores

```
df.nsmallest(5, 'Plagiarism_Score')
```

	Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score
2	3	Paper C	Author1	2025-04-02	9	Excellent	Accepted	AI, Robotics	12	5
5	6	Paper F	Author5	2025-04-03	8	Well done	Accepted	IoT, AI	7	8
0	1	Paper A	Author1	2025-03-30	8	Good work	Accepted	AI, ML	8	10
7	8	Paper H	Author6	2025-04-01	9	Very good	Accepted	Cybersecurity	10	10
6	7	Paper G	Author2	2025-03-27	7	Good novelty	Accepted	ML	8	15

10) List unique keywords used across all papers

```
df['Keywords'].str.split(',').explode().str.strip().unique()

array(['AI', 'ML', 'IoT', 'Robotics', 'Security', 'Blockchain', 'Cybersecurity'], dtype=object)
```

11) Correlation between review score and plagiarism score

```
df[['Review_Score', 'Plagiarism_Score']].corr()
```

	Review_Score	Plagiarism_Score
Review_Score	1.000000	-0.739644
Plagiarism_Score	-0.739644	1.000000

12) Find the standard deviation of the review scores

```
df['Review_Score'].std()

1.4907119849998598
```

13) Number of papers submitted by each author

```
df['Author(s)'].value_counts()
```

Author(s)	count
Author1	2
Author2	2
Author3	2
Author4	1
Author5	1
Author6	1
Author7	1

dtype: int64

14) Find papers with 'novel' in reviewer comments

```
df[df['Reviewer_Comments'].str.contains('novel', case=False)]
```

Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score	
6	7	Paper G	Author2	2025-03-27	7	Good novelty	Accepted	ML	8	15

15) Identify papers with outlier review scores

```
mean = df['Review_Score'].mean()
std = df['Review_Score'].std()

outliers = df[(df['Review_Score'] > mean + 2*std) | (df['Review_Score'] < mean - 2*std)]
outliers
```

Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score
----------	-------	-----------	-----------------	--------------	-------------------	----------	----------	-------------	------------------

16) Group papers by decision and calculate the average review score

```
df.groupby('Decision')['Review_Score'].mean()
```

Decision	Review_Score
Accepted	8.2
Rejected	5.8

dtype: float64

17) Create a new column categorizing papers based on plagiarism score

```

conditions = [
    (df['Plagiarism_Score'] <= 15),
    (df['Plagiarism_Score'] > 15) & (df['Plagiarism_Score'] <= 30),
    (df['Plagiarism_Score'] > 30)
]

choices = ['Low', 'Moderate', 'High']

df['Plagiarism_Level'] = np.select(conditions, choices, default='Unknown')
df[['Paper_ID', 'Plagiarism_Score', 'Plagiarism_Level']]

```

	Paper_ID	Plagiarism_Score	Plagiarism_Level
0	1	10	Low
1	2	35	High
2	3	5	Low
3	4	20	Moderate
4	5	40	High
5	6	8	Low
6	7	15	Low
7	8	10	Low
8	9	50	High
9	10	25	Moderate

18) Find how many papers have more than 10 pages

```

df[df['No_of_Pages'] > 10]

```

	Paper_ID	Title	Author(s)	Submission_Date	Review_Score	Reviewer_Comments	Decision	Keywords	No_of_Pages	Plagiarism_Score	Plagiarism_Level
2	3	Paper C	Author1	2025-04-02	9	Excellent	Accepted	AI, Robotics	12	5	Low
4	5	Paper E	Author4	2025-03-31	5	Poor	Rejected	Blockchain	11	40	High
8	9	Paper I	Author7	2025-04-04	6	Could be better	Rejected	IoT, Blockchain	13	50	High

19) Check for missing values in the dataset

```

df.isnull().sum()

```

	0
Paper_ID	0
Title	0
Author(s)	0
Submission_Date	0
Review_Score	0
Reviewer_Comments	0
Decision	0
Keywords	0
No_of_Pages	0
Plagiarism_Score	0
Plagiarism_Level	0

dtype: int64

20) Find the earliest and latest submission dates

```
print("Earliest Submission Date:", df['Submission_Date'].min())  
print("Latest Submission Date:", df['Submission_Date'].max())
```

```
Earliest Submission Date: 2025-03-27 00:00:00  
Latest Submission Date: 2025-04-04 00:00:00
```