1.Scenario: You are working on a project that involves analyzing student performance data for a class of 32 students. The data is stored in a NumPy array named student\_scores, where each row represents a student and each column represents a different subject. The subjects are arranged in the following order: Math, Science, English, and History. Your task is to calculate the average score for each subject and identify the subject with the highest average score.

Question: How would you use NumPy arrays to calculate the average score for each subject and determine the subject with the highest average score? Assume 4x4 matrix that stores marks of each student in given order.

**CODE:**

**import pandas as pd**

**import numpy as np**

**# Load the data**

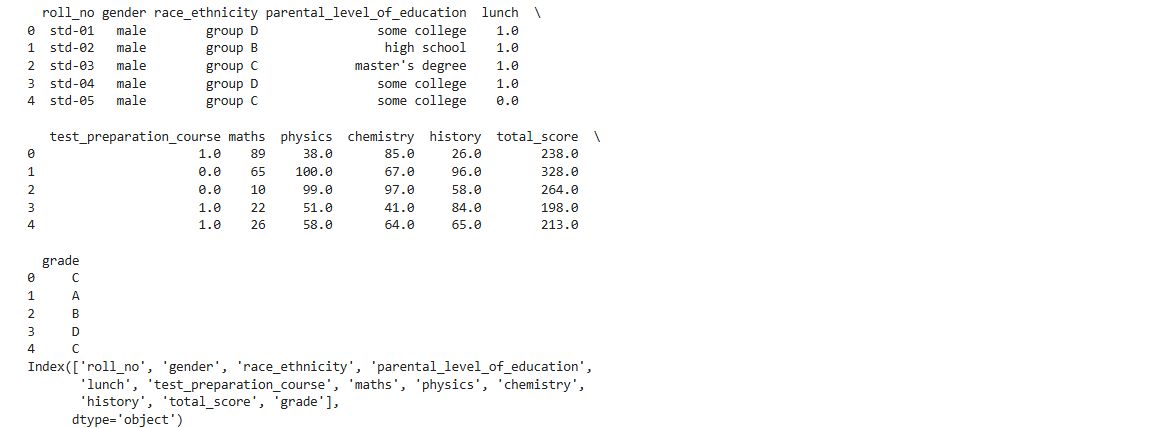
**file\_path = r"D:\datasets\Student\_performance\_10k.csv"**

**df = pd.read\_csv(file\_path)**

**# Show the first few rows to understand the structure**

**print(df.head())**

**print(df.columns)**



**# Drop rows with any missing values**

**df\_clean = df.dropna()**

**# Check how many rows are left**

**print(f"Original rows: {df.shape[0]}")**

**print(f"Rows after dropping missing values: {df\_clean.shape[0]}")**



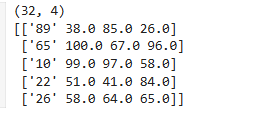
df\_clean = df\_clean.iloc[:32]

subjects = ['maths', 'physics', 'chemistry', 'history']

student\_scores = df\_clean[subjects].to\_numpy()

print(student\_scores.shape)

print(student\_scores[:5])



student\_scores = student\_scores.astype(float)

subject\_averages = np.mean(student\_scores, axis=0)

subject\_names = ['Maths', 'Physics', 'Chemistry', 'history']

highest\_avg\_index = np.argmax(subject\_averages)

highest\_avg\_subject = subject\_names[highest\_avg\_index]

# Output

for subject, avg in zip(subject\_names, subject\_averages):

print(f"{subject}: {avg:.2f}")

print(f"\nSubject with the highest average score: {highest\_avg\_subject} ({subject\_averages[highest\_avg\_index]:.2f})")

