Here’s the implementation and explanation for each question:

**Q1. Print the data in the second row of df.**

import pandas as pd

course\_name = ['Data Science', 'Machine Learning', 'Big Data', 'Data Engineer']

duration = [2, 3, 6, 4]

df = pd.DataFrame(data={'course\_name': course\_name, 'duration': duration})

# Printing the second row

print(df.iloc[1]) # iloc is used for positional indexing

**Q2. Difference between loc and iloc in pandas.DataFrame.**

* **loc**: Label-based indexing. Access rows and columns by labels or boolean arrays.
* **iloc**: Integer-based positional indexing. Access rows and columns by numerical indices.

Example:

# Using loc

df.loc[1] # Access row with index label '1'

# Using iloc

df.iloc[1] # Access second row by its position

**Q3. Reindex the dataframe and analyze the output.**

# Reindexing

reindex = [3, 0, 1, 2]

new\_df = df.reindex(reindex)

# Outputs

print(new\_df.loc[2]) # Access row with label '2'

print(new\_df.iloc[2]) # Access third row by position

* new\_df.loc[2]: Access row with index label 2 (from original dataframe).
* new\_df.iloc[2]: Access the third row of the new reindexed dataframe.

**Q4. Statistical measurements for df1.**

import numpy as np

# Creating df1

columns = ['column\_1', 'column\_2', 'column\_3', 'column\_4', 'column\_5', 'column\_6']

indices = [1, 2, 3, 4, 5, 6]

df1 = pd.DataFrame(np.random.rand(6, 6), columns=columns, index=indices)

# (i) Mean of each column

print(df1.mean())

# (ii) Standard deviation of 'column\_2'

print(df1['column\_2'].std())

**Q5. Replace row data with a string and calculate mean.**

# Replacing second row of 'column\_2' with a string

df1.loc[2, 'column\_2'] = 'string'

# Attempt to calculate mean

try:

print(df1['column\_2'].mean())

except Exception as e:

print(e)

**Explanation**: Pandas columns must have consistent data types to perform numerical operations like mean(). Replacing a value with a string changes the column's data type, causing errors.

**Q6. Understanding the windows function in Pandas.**

* **Windows functions** process data in a sliding window.
* Types:
  + **Rolling**: Apply a function over a moving window.
  + **Expanding**: Apply a function over an expanding window.
  + **EWM (Exponentially Weighted Mean)**: Apply exponentially weighted calculations.

**Q7. Print current month and year.**

from datetime import datetime

# Current month and year

now = datetime.now()

print(f"{now.strftime('%B')}, {now.year}")

**Q8. Calculate date differences in days, hours, and minutes.**

# Prompt user for two dates

date1 = pd.to\_datetime(input("Enter first date (YYYY-MM-DD): "))

date2 = pd.to\_datetime(input("Enter second date (YYYY-MM-DD): "))

# Calculate difference

delta = date2 - date1

print(f"Days: {delta.days}, Hours: {delta.total\_seconds() // 3600}, Minutes: {delta.total\_seconds() // 60}")

**Q9. Convert a column to categorical type.**

# Read CSV and convert column to categorical

file\_path = input("Enter file path: ")

col\_name = input("Enter column name to convert: ")

categories = input("Enter category order (comma-separated): ").split(',')

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

df[col\_name] = pd.Categorical(df[col\_name], categories=categories, ordered=True)

# Display sorted data

print(df.sort\_values(by=col\_name))

**Q10. Stacked bar chart for sales data.**

import matplotlib.pyplot as plt

file\_path = input("Enter file path: ")

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

# Assuming the file has columns 'Date', 'Product', and 'Sales'

df.pivot\_table(values='Sales', index='Date', columns='Product', aggfunc='sum').plot(kind='bar', stacked=True)

plt.title("Sales by Product Category Over Time")

plt.xlabel("Date")

plt.ylabel("Sales")

plt.legend(title="Product")

plt.show()

**Q11. Statistical summary for student data.**

file\_path = input("Enter file path: ")

# Reading data

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

# Calculating statistics

mean = df['Test Score'].mean()

median = df['Test Score'].median()

mode = df['Test Score'].mode().tolist()

# Displaying results in a table

print("+-----------+--------+")

print("| Statistic | Value |")

print("+-----------+--------+")

print(f"| Mean | {mean:.2f} |")

print(f"| Median | {median:.2f} |")

print(f"| Mode | {', '.join(map(str, mode))} |")

print("+-----------+--------+")