Task 1

Create a Pandas Series with custom index labels. Use the data [10, 20, 30, 40, 50] and index labels ['a', 'b', 'c', 'd', 'e'] and display it.

Task 2

Access the element at position 2 from a Series created from the list [100, 200, 300, 400, 500]. Also give any example of labelled indexing for that list and display it.

Task 3

Create two Series: Series1 with values [5, 10, 15, 20] and index ['w', 'x', 'y', 'z'], and Series2 with values [2, 4, 6, 8] and index ['w', 'x', 'y', 'p']. Add, Subtract, Multiply and divide these two Series together. Handle possible NaN values.

Task 4

Create a DataFrame with at least 5 rows and use the head() method to display the first 3 rows. Then use the tail() method to display the last 2 rows. Also, try experimenting with the default value and negative arguments for the these methods.

Task 5

Create a Series from [15, 22, 15, 30, 22, 45, 30, 15]. Find the unique values, count how many unique values there are using nunique(), and display the frequency of each value using value_counts().

Task 6

Create a DataFrame with columns 'Item' and 'Cost' containing at least 5 items. Add a new column called 'Tax' that is 10% of the Cost. Add another column called 'Total' which is Cost plus Tax. Then filter and display only the rows where Total is greater than 50.

Task 7

Create a DataFrame with columns 'Product', 'Category', 'Price', 'Stock' containing at least 8 products from different categories. Then perform the following operations:

- Use isnull() and notnull() to check for missing values
- Use value_counts() on the Category column to count products in each category
- Use nlargest() to find the top 3 most expensive products
- Use nsmallest() to find the 2 products with the lowest stock

- Display the shape and ndim of the DataFrame

Task 8

Create a DataFrame with columns 'Student', 'Subject', 'Score', 'Attendance' containing at least 10 students with some duplicate student names taking different subjects. Then perform the following operations:

- Use duplicated() to identify duplicate student names
- Use drop_duplicates() to remove duplicate student names keeping the first occurrence
- Use set_index() to set Student as the index
- Use loc[] to retrieve data for a specific student
- Use iloc[] to retrieve the first 3 rows
- Use query() method to find students with Score greater than 80 and Attendance greater than 85
- Use between() to find students with scores between 70 and 90