**Write user-defined functions to perform repetitive tasks**

def area\_rectangle(width, height):

"""

This function calculates the area of a rectangle.

Args:

width: The width of the rectangle.

height: The height of the rectangle.

Returns:

The area of the rectangle.

"""

area = width \* height

return area

# Calculate the area of a rectangle with width 5 and height 10

rectangle\_area = area\_rectangle(5, 10)

print(f"The area of the rectangle is: {rectangle\_area}")

**Create and manipulate numpy arrays; create and manipulate pandas series; create and manipulate dataframes.**

import numpy as np

# Create an array of zeros

array = np.zeros(10)

print(array)

# Create an array of ones

array = np.ones((3, 4))

print(array)

# Create an array with specific values

array = np.array([1, 2, 3, 4, 5])

print(array)

# Create an empty DataFrame

data = {}

df = pd.DataFrame(data)

print(df)

# Create a DataFrame from a list of dictionaries

data = [{'name': 'John', 'age': 30}, {'name': 'Jane', 'age': 25}]

df = pd.DataFrame(data)

print(df)

# Create a DataFrame from a NumPy array

array = np.array([[1, 2, 3], [4, 5, 6]])

df = pd.DataFrame(array, columns=['col1', 'col2', 'col3'])

print(df)

# Create an empty DataFrame

data = {}

df = pd.DataFrame(data)

print(df)

# Create a DataFrame from a list of dictionaries

data = [{'name': 'John', 'age': 30}, {'name': 'Jane', 'age': 25}]

df = pd.DataFrame(data)

print(df)

# Create a DataFrame from a NumPy array

array = np.array([[1, 2, 3], [4, 5, 6]])

df = pd.DataFrame(array, columns=['col1', 'col2', 'col3'])

print(df)

**Describe how to index and "type" pandas Series and DataFrames.**

**Indexing Pandas Series and DataFrames**

**Pandas provides various indexing methods to access and manipulate specific data within Series and DataFrames. Here's an overview of the most common methods:**

**1. Bracket indexing:**

**This is the most basic indexing method, using square brackets [].**

**It allows accessing elements by position (integer) or label (string).**

**For Series, use series[index] to access a single element or series[start:end] for a slice.**

**For DataFrames, use df[column\_name] to access a single column, df[row\_index] to access a single row, or df[start\_row:end\_row, start\_column:end\_column] for subsetting rows and columns.**

**2. loc attribute:**

**This attribute allows indexing based on both position and label, providing more flexibility.**

**For Series, use series.loc[index] or series.loc[start:end] similar to bracket indexing.**

**For DataFrames, use df.loc[row\_label, column\_label] to access a single element, df.loc[row\_slice, column\_slice] for subsetting, and boolean indexing with df.loc[condition] to select rows based on a condition.**

**3. iloc attribute:**

**This attribute allows purely integer-based indexing, independent of labels.**

**For Series and DataFrames, use series.iloc[index] or df.iloc[row\_index, column\_index] to access single elements or series.iloc[start:end] or df.iloc[row\_slice, column\_slice] for subsetting.**

**4. Boolean indexing:**

**This method allows selecting rows or columns based on a boolean condition.**

**For Series and DataFrames, use series[condition] or df[condition] where condition is a boolean expression.**

**5. Advanced indexing:**

**Pandas also offers advanced indexing features like multi-level indexing, slicing with .at and .iat, and fancy indexing with custom functions.**

**These methods provide more control and flexibility for complex data manipulation.**

**Create histograms and scatter plots for basic exploratory data analysis.**

**# Create a histogram for the "age" column**

**plt.hist(data["age"])**

**plt.xlabel("Age")**

**plt.ylabel("Frequency")**

**plt.title("Distribution of Age in sample\_data.csv")**

**plt.show()**

**# Create a scatter plot between "age" and "salary" columns**

**sns.scatterplot(x="age", y="salary", data=data)**

**plt.xlabel("Age")**

**plt.ylabel("Salary")**

**plt.title("Relationship between Age and Salary in sample\_data.csv")**

**plt.show()**