**1️⃣ Main Tasks for Week 4**

**🔹 Task 1: Guided Final Project Selection & AI-Assisted Problem Solving**

📌 **Goal:** Students select a real-world coding challenge and use AI assistance to build a solution.  
**User-Generated Prompt Example:**  
*"I need a Python script to analyze weather data. The script should:"*

* **Load a dataset containing daily weather measurements.**
* **Calculate moving averages for temperature over a rolling 7-day window.**
* **Identify days with extreme weather conditions (e.g., heatwaves, cold spells).**
* **Generate a visualization of temperature trends.**  
  *"Can you generate an efficient Python script for this?"*

**Expected Code:**

* **AI-generated script for time-series analysis using Pandas and Matplotlib.**
* **Code for detecting extreme weather patterns based on threshold values.**
* **Efficient implementation of rolling averages for trend analysis.**

**Week 4 - Task 1: Guided Final Project - AI-Assisted Weather Data Analysis**

### **AI-Generated Python Script for Time-Series Analysis & Extreme Weather Detection**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load and preprocess weather dataset

def load\_weather\_data(file\_path):

try:

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

df['Date'] = pd.to\_datetime(df['Date']) # Ensure Date column is in datetime format

df = df.sort\_values(by='Date') # Sort data by date

return df

except Exception as e:

print(f"Error loading data: {e}")

return None

# Calculate rolling averages for temperature

def calculate\_rolling\_avg(df, window=7):

df['Temp\_7Day\_Avg'] = df['Temperature'].rolling(window=window, min\_periods=1).mean()

return df

# Identify extreme weather conditions

def detect\_extreme\_weather(df, high\_threshold=90, low\_threshold=32):

df['Extreme\_Heat'] = df['Temperature'] > high\_threshold

df['Extreme\_Cold'] = df['Temperature'] < low\_threshold

return df

# Visualize temperature trends

def plot\_temperature\_trends(df):

plt.figure(figsize=(12, 6))

sns.lineplot(x=df['Date'], y=df['Temperature'], label='Daily Temperature', color='blue')

sns.lineplot(x=df['Date'], y=df['Temp\_7Day\_Avg'], label='7-Day Moving Average', color='red')

plt.axhline(90, color='orange', linestyle='dashed', label='Extreme Heat Threshold')

plt.axhline(32, color='purple', linestyle='dashed', label='Extreme Cold Threshold')

plt.xlabel("Date")

plt.ylabel("Temperature (°F)")

plt.title("Temperature Trends and Extreme Conditions")

plt.legend()

plt.xticks(rotation=45)

plt.show()

# Save processed dataset

def save\_processed\_data(df, output\_path):

df.to\_csv(output\_path, index=False)

print(f"Processed data saved as {output\_path}")

# Example usage

file\_path = "weather\_data\_example.csv"

output\_path = "processed\_weather\_data.csv"

df = load\_weather\_data(file\_path)

if df is not None:

df = calculate\_rolling\_avg(df)

df = detect\_extreme\_weather(df)

plot\_temperature\_trends(df)

save\_processed\_data(df, output\_path)

**🔹 Task 2: Iterative Development & Debugging for the Project**

📌 **Goal:** Improve AI-generated scripts through **structured debugging and optimization.**  
**User-Generated Prompt Example:**  
*"I generated a Python script for analyzing weather data, but it's inefficient. Can you help optimize it by:"*

* **Replacing inefficient loops with vectorized Pandas operations?**
* **Adding exception handling to manage missing data?**
* **Ensuring the visualization uses best practices for readability?**  
  *"Can you refine and optimize my script?"*

**Expected Code:**

* **Optimized AI-generated script with vectorized operations.**
* **Debugging and exception handling improvements.**
* **Refactored visualization using Seaborn/Matplotlib.**

**Week 4 - Task 2: Iterative Development & Debugging - Optimizing Weather Data Analysis**

### **AI-Generated Python Script for Debugging & Optimization**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

def load\_weather\_data(file\_path):

try:

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

df['Date'] = pd.to\_datetime(df['Date'], errors='coerce') # Handle incorrect date formats

df = df.sort\_values(by='Date')

df.dropna(subset=['Temperature'], inplace=True) # Remove rows with missing temperature values

return df

except Exception as e:

print(f"Error loading data: {e}")

return None

def calculate\_rolling\_avg(df, window=7):

if 'Temperature' in df.columns:

df['Temp\_7Day\_Avg'] = df['Temperature'].rolling(window=window, min\_periods=1).mean()

else:

print("Error: 'Temperature' column not found in dataset.")

return df

def detect\_extreme\_weather(df, high\_threshold=90, low\_threshold=32):

if 'Temperature' in df.columns:

df['Extreme\_Heat'] = df['Temperature'] > high\_threshold

df['Extreme\_Cold'] = df['Temperature'] < low\_threshold

return df

def plot\_temperature\_trends(df):

if 'Date' in df.columns and 'Temperature' in df.columns:

plt.figure(figsize=(12, 6))

sns.lineplot(x=df['Date'], y=df['Temperature'], label='Daily Temperature', color='blue')

sns.lineplot(x=df['Date'], y=df.get('Temp\_7Day\_Avg', pd.Series([None] \* len(df))), label='7-Day Moving Average', color='red')

plt.axhline(90, color='orange', linestyle='dashed', label='Extreme Heat Threshold')

plt.axhline(32, color='purple', linestyle='dashed', label='Extreme Cold Threshold')

plt.xlabel("Date")

plt.ylabel("Temperature (°F)")

plt.title("Temperature Trends and Extreme Conditions")

plt.legend()

plt.xticks(rotation=45)

plt.show()

else:

print("Error: Required columns missing from dataset.")

def save\_processed\_data(df, output\_path):

try:

df.to\_csv(output\_path, index=False)

print(f"Processed data saved as {output\_path}")

except Exception as e:

print(f"Error saving data: {e}")

# Example usage

file\_path = "weather\_data\_example.csv"

output\_path = "optimized\_weather\_data.csv"

df = load\_weather\_data(file\_path)

if df is not None:

df = calculate\_rolling\_avg(df)

df = detect\_extreme\_weather(df)

plot\_temperature\_trends(df)

save\_processed\_data(df, output\_path)

**🔹 Task 3: Presentation & Peer Review**

📌 **Goal:** Students present AI-assisted projects and discuss improvements.  
**User-Generated Prompt Example:**  
*"Summarize the key optimizations made in my Python script, explaining why they improve efficiency and readability."*

**Expected Code:**

* **AI-generated summary of debugging improvements.**
* **Code documentation improvements for readability.**

**Week 4 - Task 3: Presentation & Peer Review - Documenting & Explaining Code Improvements**

### **Task Overview**

In this session, students will present their AI-assisted projects, highlighting key improvements made to their scripts. They will engage in peer reviews, discuss optimization strategies, and reflect on how AI-assisted coding influenced their problem-solving approach.

### **Key Objectives:**

* Summarize and document optimizations made to AI-generated scripts.
* Discuss how debugging and iterative development improved efficiency.
* Participate in structured peer review to assess code quality and readability.
* Reflect on AI’s role in enhancing programming workflows.

### **Presentation Guidelines:**

1. **Project Overview:**
   * Briefly introduce the dataset and problem addressed.
   * Explain the initial AI-generated script and its limitations.
2. **Key Improvements:**
   * **Data Handling & Error Prevention:** Describe any missing data handling, formatting corrections, and data validation measures.
   * **Efficiency Optimization:** Highlight improvements made to reduce redundant calculations or enhance performance.
   * **Visualization & Interpretation:** Explain changes made to improve the clarity and impact of graphical outputs.
3. **Lessons Learned & AI's Role:**
   * How did AI assist in structuring the project?
   * What were the biggest limitations of AI-generated code, and how were they addressed?
   * What strategies helped refine AI-generated scripts into well-optimized solutions?

### **Peer Review Process:**

* **Code Readability & Documentation:** Is the script well-commented and easy to follow?
* **Efficiency & Optimization:** Have unnecessary computations been removed? Is the code structured efficiently?
* **Error Handling & Debugging:** Are common issues anticipated and handled appropriately?
* **Overall Effectiveness:** Does the final implementation meet the project goals?

### **Wrap-Up Discussion:**

* **Future Applications:** How can AI-assisted development be applied to more complex programming challenges?
* **Best Practices:** What strategies should be used when incorporating AI-generated code in real-world projects?
* **Final Reflections:** How has AI-assisted programming changed students’ approach to structured problem-solving?

By the end of this session, students will have gained experience in presenting technical solutions, providing and receiving feedback, and critically evaluating AI-assisted programming in an applied setting.

**2️⃣ Appendices for Additional Exercises**

### **Appendix 1: AI in Collaborative Development**

**Appendix 1: AI in Collaborative Development**

📌 **Goal:** Students work in pairs using AI-assisted pair programming.  
**User-Generated Prompt Example:**  
*"I am working with a partner on a Python script for analyzing sales data. Can you generate a modularized script that:"*

* **Includes functions for loading, cleaning, and visualizing data?**
* **Allows easy collaboration by structuring the script with docstrings?**
* **Handles edge cases for missing and inconsistent data?**

**Expected Code:**

* **Well-structured AI-generated script with modular functions.**
* **Commented sections to facilitate collaboration.**

**Week 4 - Appendices: Additional Exercises**

### **Appendix 1: AI in Collaborative Development**

#### **Exercise: AI-Assisted Pair Programming**

**Goal:** Students work in pairs using AI-assisted programming to develop a structured script collaboratively.

**Task:**

* **Divide roles:** One student generates AI-assisted code, while the other reviews and refines it.
* **Use AI to modularize code:** Ensure the script is well-structured with functions and clear documentation.
* **Enhance collaboration:** Use docstrings and comments to explain functions for better teamwork.

**User-Generated Prompt Example:** "I am working with a partner on a Python script for analyzing sales data. Can you generate a structured and modular Python script that:\*

* **Loads, cleans, and visualizes sales data.**
* **Uses well-defined functions with proper docstrings and inline comments.**
* **Handles missing and inconsistent data.** Can you ensure the script is collaborative-friendly with clear documentation?"

**AI-Generated Python Code for Collaborative Development:**

import pandas as pd

import matplotlib.pyplot as plt

# Function to load data

def load\_data(file\_path):

"""Loads sales data from a CSV file."""

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

df.columns = df.columns.str.strip().str.lower()

return df

# Function to clean data

def clean\_data(df):

"""Cleans missing values and standardizes columns."""

df.dropna(inplace=True)

return df

# Function to visualize sales trends

def plot\_sales\_trends(df):

"""Generates a line plot for sales trends over time."""

df['date'] = pd.to\_datetime(df['date'])

df.groupby('date')['sales'].sum().plot(figsize=(10,5), title='Sales Trends')

plt.xlabel("Date")

plt.ylabel("Total Sales")

plt.show()

# Example usage

file\_path = "sales\_data.csv"

df = load\_data(file\_path)

df = clean\_data(df)

plot\_sales\_trends(df)

**Discussion Questions:**

1. How does modularization improve collaboration?
2. What are the benefits of AI-assisted documentation in teamwork?
3. How can AI help in debugging collaboratively?

### **Appendix 2: Analyzing AI Bias in Code Generation**

**Appendix 2: Analyzing AI Bias in Code Generation**

📌 **Goal:** Students evaluate AI-generated scripts for potential biases.  
**User-Generated Prompt Example:**  
*"I need a Python script to analyze hiring data. The script should:"*

* **Identify gender and demographic distributions in hiring trends.**
* **Ensure the analysis does not introduce biased conclusions.**
* **Apply fairness checks to avoid misrepresenting data.**

**Expected Code:**

* **AI-generated script for demographic analysis.**
* **Bias detection checks using statistical methods.**

#### **Exercise: AI Bias Detection in Data Analysis**

**Goal:** Students evaluate AI-generated scripts for potential biases in data analysis.

**Task:**

* **Generate AI-assisted code to analyze hiring data.**
* **Identify potential biases in dataset interpretation.**
* **Discuss ethical considerations in AI-generated code.**

**User-Generated Prompt Example:** "I need a Python script to analyze hiring trends based on demographic data. The script should:\*

* **Load and clean the dataset.**
* **Analyze hiring trends by gender and age group.**
* **Ensure fairness in representation and avoid misleading conclusions.** Can you generate a Python script that accounts for bias detection?"

**AI-Generated Python Code for Bias Detection in Hiring Data:**

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

# Function to load hiring data

def load\_hiring\_data(file\_path):

"""Loads hiring data from a CSV file."""

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

return df

# Function to analyze hiring trends

def analyze\_hiring\_trends(df):

"""Analyzes hiring distribution by gender and age group."""

hiring\_by\_gender = df['gender'].value\_counts(normalize=True) \* 100

hiring\_by\_age = df.groupby('age\_group')['hired'].mean() \* 100

print("Hiring by Gender (%):")

print(hiring\_by\_gender)

print("\nHiring by Age Group (%):")

print(hiring\_by\_age)

# Function to visualize hiring trends

def plot\_hiring\_distribution(df):

"""Plots hiring trends by demographic category."""

plt.figure(figsize=(10,5))

sns.countplot(data=df, x='gender', hue='hired')

plt.title("Hiring Distribution by Gender")

plt.show()

plt.figure(figsize=(10,5))

sns.barplot(x=df['age\_group'], y=df['hired'])

plt.title("Hiring Probability by Age Group")

plt.show()

# Example usage

file\_path = "hiring\_data.csv"

df = load\_hiring\_data(file\_path)

analyze\_hiring\_trends(df)

plot\_hiring\_distribution(df)

**Discussion Questions:**

1. How can AI-generated code unintentionally introduce bias?
2. What precautions should be taken when analyzing demographic data?
3. How can AI tools be improved to mitigate bias in automated analysis?

### **Outcome:**

By completing these exercises, students will:

* Learn to collaborate effectively with AI-assisted programming.
* Recognize and mitigate bias in AI-generated analysis.
* Develop structured workflows for fair and accurate data interpretation.