Projects Descriptions for INFO - 6143 _ FALL 2024 _ FANSHAWE

1. Student Performance Dashboard

- **Concepts Covered**: File reading/writing, exceptions, classes, data visualization.
- **Description**: Build a student performance dashboard that reads student data from a file (CSV/JSON) and visualizes it using graphs and charts. You can visualize metrics like scores, attendance, and grades in different subjects over time.
- **Key Features**:
    - Import student data from CSV/JSON.
    - Visualize performance metrics (e.g., grades over time, attendance vs. performance) using libraries like **Matplotlib** and **Seaborn**.
    - Handle data inconsistencies and exceptions (e.g., missing or invalid data).
    - Allow the user to filter data by subjects, grade range, or time period.

2. Expense Tracker with Visualization

- **Concepts Covered**: File handling, exceptions, data visualization.
- **Description**: Build an expense tracker that not only stores and manages expenses but also visualizes spending trends over time. Categories like food, rent, transportation, etc., can be represented in bar charts, pie charts, or line graphs.
- **Key Features**:
    - Input and store expense data in a file (CSV/JSON).
    - Visualize expenses using **Matplotlib** or **Plotly** (e.g., pie charts for categories, line graphs for monthly expenses).
    - Exception handling for invalid input (e.g., non-numeric data, negative values).
    - Create a dashboard that allows users to interact with the data and view trends for different periods.

3. Weather Data Visualizer

- **Concepts Covered**: File I/O, exceptions, data visualization.
- **Description**: Develop a program that reads weather data from a CSV/JSON file and visualizes key metrics like temperature, humidity, or rainfall over time. The data can be retrieved from a weather API or entered manually by users.
- **Key Features**:
    - Read and write weather data (e.g., date, temperature, humidity) to/from a file.
    - Visualize trends using **Matplotlib** (e.g., temperature over time, humidity vs. temperature scatter plots).
    - Handle data inconsistencies or missing data with appropriate exception handling.
    - Display visualizations interactively (e.g., with **Dash** or **Streamlit**).

4. Inventory Management with Data Visualization

- **Concepts Covered**: Python classes, exceptions, data visualization.
- **Description**: Create an inventory management system where products and stock levels are tracked over time. Use data visualization to show stock levels, sales trends, and the most popular products.
- **Key Features**:
    - Use classes to define products and manage inventory.
    - Read and write inventory data from CSV or JSON files.
    - Visualize inventory data, such as stock levels, product categories, and sales trends using **Matplotlib** or **Seaborn**.
    - Exception handling for missing or incorrect data inputs.
    - Create reports, such as bar charts for product sales or pie charts for category distribution.

5. COVID-19 Data Visualization Tool

- **Concepts Covered**: File handling, exceptions, data visualization.
- **Description**: Create a Python application that reads COVID-19 statistics from a CSV file and visualizes the data. You can show the number of cases, deaths, and recoveries over time for different regions or countries.
- **Key Features**:
    - Read COVID-19 data from CSV or external APIs.
    - Use **Matplotlib**, **Seaborn**, or **Plotly** to visualize trends (e.g., line graphs for cases, bar charts for regional comparisons).
    - Handle missing or incomplete data gracefully.
    - Provide interactive controls for the user to select the region and time frame for analysis.

6. Real-Time Stock Price Visualization

- **Concepts Covered**: File handling, exceptions, data visualization, API integration.
- **Description**: Develop a Python application that fetches real-time stock market data from an API (such as Yahoo Finance or Alpha Vantage) and visualizes the stock prices over time.
- **Key Features**:
    - Fetch real-time stock data using a stock market API.
    - Visualize stock price trends with line graphs, candlestick charts, or bar graphs.
    - Handle errors from API calls (e.g., network issues, invalid stock symbols).
    - Allow users to input stock symbols and view real-time visualizations.

7. Sales Dashboard with Interactive Visualizations

- **Concepts Covered**: Classes, exceptions, data visualization.
- **Description**: Create a sales dashboard that visualizes sales data for a company over time. Implement features like product performance, sales trends, and region-wise comparisons.
- **Key Features**:
    - Use Python classes to define sales data and manage the records.
    - Read sales data from CSV/JSON and write back after analysis.
    - Use **Matplotlib**, **Seaborn**, or **Plotly Dash** to create interactive visualizations (e.g., bar charts for product sales, heatmaps for regional sales).
    - Exception handling for missing or invalid data inputs.

8. To-Do List with Progress Tracker

- **Concepts Covered**: File I/O, exceptions, data visualization.
- **Description**: Build a to-do list application that not only allows users to add and manage tasks but also visualizes the progress of tasks over time. Tasks can be categorized by priority and completion status, and the progress can be displayed using a pie chart or progress bar.
- **Key Features**:
    - Store to-do list tasks in a file (CSV/JSON).
    - Visualize progress with pie charts, bar charts, or progress bars (tasks completed vs. remaining).
    - Allow the user to filter tasks based on priority or status.
    - Handle missing or incomplete task data.

9. Budgeting Tool with Visual Analytics

- **Concepts Covered**: File reading/writing, exceptions, data visualization.
- **Description**: Develop a budgeting tool that helps users track income and expenses. The tool should allow users to categorize expenses and visualize spending trends with graphs and charts.
- **Key Features**:
    - Store income and expense data in CSV/JSON files.
    - Visualize spending by category (e.g., pie chart for expense categories).
    - Track income and expenses over time (e.g., line graph for monthly trends).
    - Exception handling for missing or invalid data.

10. Crime Data Analysis Tool

- **Concepts Covered**: File handling, exceptions, classes, data visualization.
- **Description**: Create a tool to analyze and visualize crime data from different regions. The program should allow users to filter data by crime type, time, or location and visualize crime rates using various charts.
- **Key Features**:
  - Load crime data from CSV/JSON files.
  - Visualize crime trends with line graphs, bar charts, or heatmaps.
  - Implement data filtering based on crime types, locations, and time periods.
  - Handle missing or corrupt data and display appropriate error messages.