Program Planning

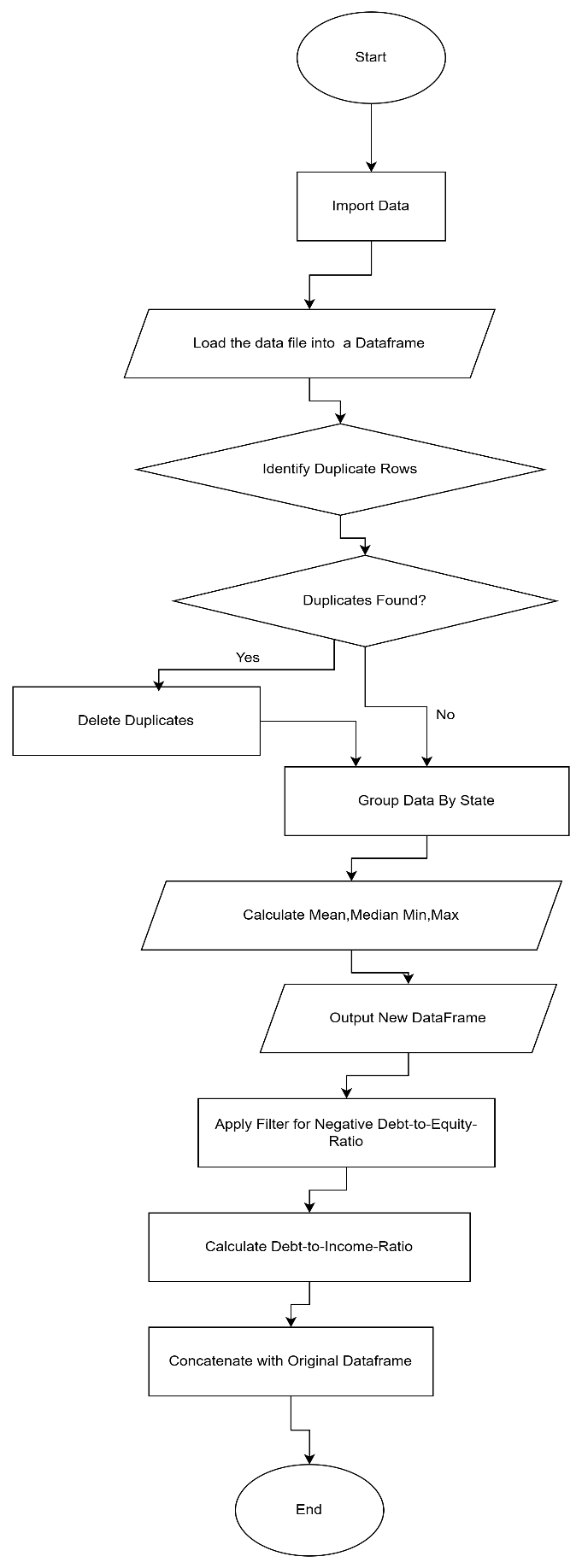
Trayvonious Pendleton

WGU

Course Number: D598

STUDENT ID: 011205284

PART A



PART B

Start

Load data from "D598 Data Set.xlsx" into a DataFrame

If loading fails:

Display an error message and terminate the program

Else:

Proceed to the next step

Check for duplicate rows in the DataFrame

If duplicates exist:

Remove the duplicates

Else:

Proceed to the next step

Group data by the "State" column

For each group:

Calculate the mean, median, minimum, and maximum for all numeric columns

Store these statistics in a new DataFrame

Filter rows where "Debt-to-Equity Ratio" is negative

Store the filtered data in a separate DataFrame

Create a new column in the DataFrame for "Debt-to-Income Ratio"

For each row:

Calculate "Long-Term Debt / Revenue" as Debt-to-Income Ratio

Concatenate the Debt-to-Income Ratio column with the original DataFrame

If concatenation fails:

Display an error message and terminate the program

Else:

Save the concatenated DataFrame as the final result

End

PART C1

The flowchart and pseudocode represent a structured approach to solving the data analysis task by breaking it into logical steps. The program begins with importing the dataset into a data frame, ensuring the data is accessible for processing. A decision point is included to handle errors during the data loading, ensuring the program does not proceed with incomplete or missing data. Next, the program identifies duplicate rows in the dataset. If duplicates are found, they are removed to maintain the integrity and accuracy of the dataset before moving to the next step.

The program then groups data by the "State" column, calculating descriptive statistics such as mean, median, minimum, and maximum for all numeric variables. These calculated statistics are stored in a new data frame, providing insights into state-level data patterns. The next step involves filtering rows with negative debt-to-equity ratios and isolating businesses with financial risks. Following this, the program calculates a new debt-to-income ratio metric by dividing long-term debt by revenue for each row. This calculation adds valuable information about the financial health of each business.

Finally, the debt-to-income ratio is concatenated with the original data frame to create a comprehensive dataset. A decision point is included to check if the concatenation is successful. If it fails, an error message is displayed, and the program terminates; otherwise, the concatenated DataFrame is saved as the result. Each step logically follows the previous one, ensuring a smooth progression of tasks and a complete data analysis process.

**PART C2**

The flowchart and pseudocode align closely, with each step in the flowchart directly corresponding to a detailed instruction in the pseudocode. For example, the flowchart step "Identify Duplicate Rows" aligns with the pseudocode logic that checks for duplicates and removes them if found. Similarly, the decision points in the flowchart, such as "Duplicates Found?" or "Data Loading Fails?" are mirrored by conditional statements in the pseudocode to ensure consistent logic.

The flowchart provides a high-level visual representation of the program’s flow, including decision points and processes. At the same time, the pseudocode offers a more detailed and precise set of instructions for implementing these steps programmatically. For instance, the flowchart step "Calculate Debt-to-Income Ratio" corresponds to the pseudocode instruction to compute "Long-Term Debt / Revenue" for each row. Additionally, the flowchart step "Concatenate with Original DataFrame" maps directly to pseudocode logic that checks for errors during concatenation and handles them appropriately.

The design ensures clarity, consistency, and logical completeness by aligning the flowchart’s high-level overview with the pseudocode’s detailed implementation. This alignment bridges the gap between conceptual planning and actual program execution, ensuring that all steps in the task are fully represented and effectively implemented.