**REPORT**

**Here's the explanation of how the design requirements is met:**

1. **Data Loading and Parsing:**
   * The `loadStrings("cars.data")` function is used to load the data from a file named "cars.data".
   * The data is then split and processed using various `splitTokens` and `split` functions to extract relevant information.
2. **Data Structures:**
   * Arrays like `cylinders`, `mpgValues`, `modelYears`, etc., are used to store different attributes of the cars.
   * Arrays are initialized and populated based on the data read from the file.
3. **Visualization:**
   * I use multiple functions (`program1()`, `program2()`, etc.) that draw different types of charts and graphs based on the processed data.
4. **User Interaction:**
   * We implements a basic menu system that allows the user to switch between different visualization programs.
   * Mouse clicks (`mousePressed()`) are used to change the `currentProgram` variable, which determines which visualization program to run.
5. **Handling Missing Data:**
   * We also checks for the presence of necessary values and ensures they are not equal to "NA" before using them.
6. **Labels and Axis Scaling:**
   * Labels are drawn along both the X and Y axes to provide context for the visualizations.
7. **Color Coding:**
   * Different colors are used to represent data points in the visualizations. For example, outlier bars are colored differently.
8. **Statistics and Calculations:**
   * Various calculations are performed on the data, such as averaging and counting occurrences based on specific criteria.
9. **Error Handling:**
   * We also includes exception handling (`try`-`catch` blocks) to handle potential errors when reading the data file.

Overall, the code effectively processes and visualizes the car data based on the provided design requirements. It provides an interactive interface for users to explore different aspects of the dataset.

**Menu -**

**A screenshot of a computer

Description automatically generated**

The menu is implemented as a set of buttons displayed on the right side of the window. Each button corresponds to a program, and clicking a button activates the associated program.

* **Menu Layout**:
  + The menu displays buttons for each program, labeled with their corresponding names (Line Chart, Histogram, etc.).
  + Buttons are arranged vertically with appropriate spacing.
* **Button Functionality**:
  + The **mousePressed()** function is used to detect which button was clicked.
  + Upon clicking a button, the **currentProgram** variable is updated to activate the corresponding program.

**Task 1 - Line Chart (Model Year vs MPG)**

* **Objective**: This program generates a line chart displaying the average MPG (Miles Per Gallon) for cars from 1970 to 1982.
* **Implementation**:
  + **Data Extraction**: Data is loaded from the "cars.data" file to extract MPG values and corresponding model years.
  + **Data Processing**: The program calculates the average MPG for each year.
  + **Visualization**: The line chart is drawn with model years on the x-axis and average MPG on the y-axis. Data points are plotted and connected with lines.

**A graph with numbers and lines

Description automatically generated**

* The program1() function begins by setting up the environment with a white background and drawing the X and Y axis.
* It calculates average MPG for each model year, divides it into groups based on model years, and calculates the average MPG for each group.
* It then proceeds to draw the X and Y axis, labels, and data points on the chart.
* The design requirements are met by creating a clear, labeled line chart with appropriate data points and labels for model years and MPG values.

**Task 3 - Bar Chart**

* **Objective**: This program generates a bar chart displaying the number of cars for each model year.
* **Implementation**:
  + **Data Extraction**: Model year data is extracted from the "cars.data" file.
  + **Data Processing**: The program counts the number of cars for each year.
  + **Visualization**: The bar chart is drawn with model years on the x-axis and the number of cars on the y-axis.

A screenshot of a computer

Description automatically generated

* The program3() function sets up the environment with a white background and draws the X and Y axis, labels, and data bars.
* It calculates the relationship between model year and number of cars and uses a bar chart to visualize this data.
* The bars display additional information about the model year and number of cars when you hover over them.
* The design requirements are met by creating a labelled bar chart that correctly represents the relationship between model year and number of cars.

**Task 4 - Pie Chart**

* **Objective**: This program creates a pie chart showing the distribution of cars based on their origin.
* **Implementation**:
  + **Data Extraction**: Origin information is extracted from the "cars.data" file.
  + **Data Processing**: The program calculates the proportion of cars for each origin.
  + **Visualization**: The pie chart is drawn with segments representing different origins.

A screenshot of a computer screen

Description automatically generated

* The program4() function starts by setting up the environment with a white background.
* It calculates the distribution of cars based on their origin and creates a pie chart to visualize this data.
* The pie chart segments are labeled with their origin, percentage, and total car count.
* The design requirements are met by creating a labeled pie chart that correctly represents the distribution of cars by origin.

**Task 5 - Scatter Plot**

* **Objective**: This program generates a scatter plot displaying the relationship between MPG and horsepower.
* **Implementation**:
  + **Data Extraction**: MPG and horsepower values are extracted from the "cars.data" file.
  + **Data Filtering**: Invalid or missing data points are excluded.
  + **Visualization**: The scatter plot is drawn with MPG on the x-axis and horsepower on the y-axis. Data points are displayed with tooltips.

A screenshot of a computer

Description automatically generated

* The program5() function sets up the environment with a white background and draws the X and Y axis, labels, and data points.
* It calculates and visualizes the relationship between MPG and horsepower, displaying data points and showing additional information when you hover over the data points.
* The design requirements are met by creating a labeled scatter plot that correctly represents the relationship between MPG and horsepower.
* There is outlier in the screenshot showing with MPG value 33 and Horsepower value is 190.

**Conclusion**

The programs successfully meet their respective design requirements, providing visual representations of various aspects of the car dataset. The menu allows for easy navigation between programs, enhancing user interaction with the visualizations.

**Submitted By : Sameer Khan**

**Note :** In this report I only Mention the Tasks that is done by me.