PyQt5 Automotive Exam: Real-Time Car Performance Monitor

Exercise: Real-Time Car Performance Monitor

Create a PyQt5 application that monitors and visualizes real-time car performance metrics such as **RPM (Revolutions Per Minute)**, **Torque**, and **Horsepower**. The application should use ‘QThread‘ to simulate data generation, ‘QTimer‘ to update the UI, and custom signals to communicate between threads and the main UI.

Requirements and Questions

Question 1: Create the Main Window (3 Marks)

- Create a ‘QMainWindow‘ with a fixed size of 800x600 pixels.
- Add three ‘QProgressBar‘ widgets to display RPM, Torque, and Horsepower.
- Add a ‘QStatusBar‘ to display the current status (e.g., ”Monitoring”, ”Stopped”).

Question 2: Implement the Data Generation Thread (5 Marks)

- Create a ‘QThread‘ subclass named ‘PerformanceDataThread‘ to simulate real-time data generation for RPM, Torque, and Horsepower.
- Use a ‘QTimer‘ inside the thread to generate data at regular intervals (e.g., every 300 milliseconds).
- Emit custom signals to send the generated data to the main UI.

Question 3: Update the UI Dynamically (4 Marks)

- Connect the custom signals from ‘PerformanceDataThread‘ to slots in the main window to update the ‘QProgressBar‘ widgets.
- Ensure the UI updates in real-time as new data is received.
- Use appropriate ranges for the progress bars (e.g., 0-8000 for RPM, 0-500 for Torque, 0-700 for Horsepower).

Question 4: Handle Warnings and Alerts (3 Marks)

- If the RPM exceeds 7000, display a warning message in the status bar and change the progress bar color to red.
- If the Torque exceeds 450 Nm, display a warning message in the status bar and change the progress bar color to yellow.
- If the Horsepower exceeds 600 HP, display a warning message in the status bar and change the progress bar color to red.

Question 5: Add Start/Stop Functionality (3 Marks)

- Add two buttons: ”Start Monitoring” and ”Stop Monitoring”.
- When ”Start Monitoring” is clicked, start the ‘PerformanceDataThread‘ and update the status bar to ”Monitoring”.
- When ”Stop Monitoring” is clicked, stop the thread and update the status bar to ”Stopped”.

Question 6: Save Performance Data to a File (2 Marks)

- Add a "Save Data" button to save the current performance data (RPM, Torque, Horsepower) to a CSV file.
- The file should include a timestamp for each data entry.

Total Marks: 20

- Question 1: 3 Marks
- Question 2: 5 Marks
- Question 3: 4 Marks
- Question 4: 3 Marks
- Question 5: 3 Marks
- Question 6: 2 Marks

Estimated Duration: 3 Hours

This exam tests your ability to work with ‘QThread‘, ‘QTimer‘, custom signals, ‘QProgressBar‘, and file handling in PyQt5. It also evaluates your understanding of dynamic UI updates and real-time data processing. Good luck!