RUSH HOUR GAME

Rush Hour is a single-player game where the player assumes the role of a taxi driver tasked with collecting money by transporting passengers to their destinations. The game begins with a start menu that offers options to view the leaderboard, select a taxi color, and enter the player's name. The leaderboard displays the top 10 scores and player names, which are read from a file called "highscores.txt" and stored in arrays. If the current game's score makes it to the leaderboard, the lowest score is replaced, and the updated leaderboard is saved in the file.

The Board and Game Play:

The game features a 20x20 board with roads (white) and buildings (black). The player's taxi and other cars, obstacles, and passengers are placed randomly on the board. At the start, the player's taxi is positioned at the top left corner. The board always has 3 to 5 passengers waiting to be picked up, located anywhere except where other cars or obstacles are present. The player controls the taxi using arrow keys to move in four directions and must navigate through obstacles, pick up passengers, and reach their desired destinations.

Passenger's Desired Destination, Points, and Negative Points:

Each passenger has a highlighted desired destination on the board, which cannot be the same as the pickup location or occupied by an obstacle. The player earns 10 points for picking up and dropping off passengers successfully, with additional points for overtaking cars. Negative points are incurred for hitting people, obstacles, or other cars, with different point deductions based on the type of taxi. The game also includes features such as increasing the number and speed of cars on the board after successful drops, and winning criteria of scoring 100 points or more within a 3-minute time limit.

Graphics, Bonus Features, and Coding Requirements:

The game incorporates graphics with stick figures representing passengers and rectangles with circles for cars. Bonus features include an auto mode for the taxi to find the shortest path and navigate automatically while avoiding obstacles and collisions. Each new game generates a new board with reachable corners, comprising 35% buildings and 65% roads. The project requires the use of functions, pointers, dynamic memory allocation, and file handling. Starter code is provided in C++ to draw shapes and can be modified and expanded upon to fulfill the project requirements.