Board

• contains a 2-D array storing all position of cars • contains a map storing all cars • contains number of cars • find an empty spot and add new car in • change the position of a particular car • update the 2-D array of board if any car has been changed • provide possible moves for the current board	• BoardGenerator • BoardIO • Settings • Car • SearchAlgorithm • MoveComponent • CarCreate • GameFrame

BoardIO

• print the board generated from BoardGenerator and the solution for this board to a file • load a board from file and return the Board to caller • print Score of a player of specific difficulty and level to a file • load all the high score players of specific difficulty and level, return the sorted high score player list	• SearchAlgorithm • Player • BoardGenerator

BoardGenerator

• generate a random board and check the solution fits the number of moves wanted, keep generate until get the wanted board and print it to the difficulty related puzzles file • randomly generate a board with wanted number of cars	• SearchAlgorithm • Settings

Car

• contains carID, position in the board, length and direction of the car • change the position of car	• board • MoveComponent • Settings • CarCreate

Player	
• contains the information of the player with its name and score	• GameFrame • BoardIO

Interface SearchAlgorithm SearchAlgorithmAStar, SearchAlgorithmBFS	
• print the solution board by required way (print cars or board) • get the solution path of required board and return the list of board represents each step	• Board

SearchAlgorithmAStar	
• find solution path of required board using provided heuristic • reconstruct the path by getting the final board and retrieve • get and set heuristic for the current AStarNode	• Heuristic • Board • AStarNode

AStarNode	
• State of AStar search Algorithm, contains the board of current state, g, h and f value, the previous state	• AStarSearchAlgorithm • Board

SearchAlgorithmBFS	
• Using BFS to find the solution of board	• Board

Interface Heuristic HeuristicCarsInfront, HeuristicZero	
• calculate heuristic value for the current Node	• SearchAlgorithmAStar

HeuristicZero

• Always return 0	

HeuristicCarsInfront

• The heuristic that use in AStar, which gets the number of cars that in red cars way to exit point	• SearchAlgorithmAStar

CarCreate

• Contains a list of all car colors • For each car in the given board, assign its colour, bound and wont allow any other car overlap it. store it into the carList • Create a new MoveComponent object for each car, store all its information into it. store it into moveList • Contains the board for these cars	• MoveComponent • Car • Settings

MoveComponent

• Every MoveComponent is a car, stores its information including carID, board, carLabel and Size. • Whenever mouse pressed any car on the board, it will set up the car for dragging. • SetupDragging will detect the component that is dragging and change cursor to the current position • Override the mouseDragged method to avoid any overlapping	• CarCreate

GameFrame

• Set up for welcome Screen, select difficulty screen, different level screen and their related buttons • After entering the game, print grid and cars, show difficulty, current move number, level, Highest score. Hint could be loaded by using AStar to find next move and update the board • After successfully finished game, will show the score and ask for name to put in the HighScore file • Background music for the game • Settings containing start or stop the bgm	• SearchAlgorithm • CarCreate • Player • BoardIO • Board • Grid • main • Settings

Settings

• contains all predefined data and different types of difficulty	• Board • Car • CarCreate • GameFrame • BoardGenerator • SearchAlgorithm

Main

• Show the Game window	• GameFrame