PA11

Generated by Doxygen 1.8.11

Contents

Chapter 1

Main Page

This program will simulate a rush hour game board and solve the board.

This program takes in input from the keyboard to make a rush hour game board and place a maximum of eighteen cars. this program then has the potential to solve said 6x6 board if there exist a solution with a maximum of ten move. futher more if you can manually manipulate the board and move individual cars if desired. the program includes the Board class with the following functions:

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 Board Class Reference

`#include <RushHour.h>`

Public Member Functions

- Board ()

  This function will initialize a board class variable by setting all the grid values to zero, this is the default constructor of the class.
- Board & operator= (const Board &other)

  default assignment operator that copies one board into another
- int setupBoard ()

  This function will set up the board values from keyboard inputs it also makes sure to clear the variables of the board class to start with a clean board before it sets any values. Furthermore it will return a int value if it is zero then there is no more inputs to be taken in other wise it is some other number of cars.
- void printBoard () const

  This Function prints the board's two dimensianol array values this function is mainly for debugging purposes.
- bool isWon () const

  bool function that checks the current state of the board, whether it is won or not
- int getNumCars () const

  function that returns the number of cars
- bool moveForward (int carNum) const

  This function checks if the specified vehicle can be moved forward within the board's two dimensional array.
- bool moveBackward (int carNum) const

  This function checks if the specified vehicle can be moved forward within the board's two dimensional array.
- void moveVehicleF (int carNum)

  This function checks if the specified vehicle can be moved forward within the board's two dimensional array.
- void moveVehicleB (int carNum)

  This function checks if the specified vehicle can be moved forward within the board's two dimensional array.
- string getKey () const

  returns the keyVal for the current board

4.1.1 Constructor & Destructor Documentation

4.1.1.1 Board::Board ( )

This function will initialize a board class variable by setting all the grid values to zero, this is the default constructor of the class.

Parameters

none	

Returns

none

Precondition

none

Postcondition

creates a board item

4.1.2 Member Function Documentation

4.1.2.1 string Board::getKey ( ) const

returns the keyVal for the current board

Parameters

none	

Returns

string keyVal

Precondition

none

Postcondition

none

4.1.2.2 int Board::getNumCars ( ) const

function that returns the number of cars

Parameters

none	

Returns

int numberOfCars

Precondition

none

Postcondition

none

4.1.2.3 bool Board::isWon ( ) const

bool function that checks the current state of the board, whether it is won or not

Parameters

none	

Returns

bool true if win condition is met, bool false if not

Precondition

none

Postcondition

none

4.1.2.4 bool Board::moveBackward (int *carNum*) const

This function checks if the specified vehicle can be moved forward within the board's two dimensional array.

Parameters

int	value that specifies which scenerio is being printed

Returns

none

Precondition

none

Postcondition

prints message to terminal

4.1.2.5 bool Board::moveForward (int *carNum*) const

This function checks if the specified vehicle can be moved forward within the board's two dimensional array.

Parameters

int	value that specifies which scenerio is being printed

Returns

none

Precondition

none

Postcondition

prints message to terminal

4.1.2.6 void Board::moveVehicleB (int *carNum*)

This function checks if the specified vehicle can be moved forward within the board's two dimensional array.

Parameters

int	value that specifies which scenerio is being printed

Returns

none

Precondition

none

Postcondition

prints message to terminal

4.1.2.7 void Board::moveVehicleF (int *carNum*)

This function checks if the specified vehicle can be moved forward within the board's two dimensional array.

Parameters

int	value that specifies which scenerio is being printed

Returns

none

Precondition

none

Postcondition

prints message to terminal

4.1.2.8 Board & Board::operator= (const Board & *other*)

default assignment operator that copies one board into another

Parameters

const	Board& other

Returns

∗this

Precondition

none

Postcondition

updates the current board with one assigned to it

4.1.2.9 void Board::printBoard ( ) const

This Function prints the board's two dimensianol array values this function is mainly for debugging purposes.

Parameters

none	

Returns

none

Precondition

none

Postcondition

prints message two dimensional array

4.1.2.10 int Board::setupBoard ( )

This function will set up the board values from keyboard inputs it also makes sure to clear the variables of the board class to start with a clean board before it sets any values. Furthermore it will return a int value if it is zero then there is no more inputs to be taken in other wise it is some other number of cars.

Parameters

noe	

Returns

int value that functions as a bool to demonstrate that there is no more values to be read into the board

Precondition

the input values to the board must be as specified and cars must not be out of bound or overlapping

Postcondition

allocates all inputed data in the proper place so that the board can be solved

The documentation for this class was generated from the following files:

- RushHour.h
- RushHour.cpp

4.2 rushBoard Class Reference

```
#include <RushHour.h>
```

Public Member Functions

- rushBoard ()
  default constructor for rushBoard
- int setupBoard ()
  function that sets up the default game board
- void printResults (int scenarioNum) const
  Get Key Function.
- void printBoard () const
  Get Key Function.
- bool isWon () const
  Get Key Function.
- void solve ()
  Get Key Function.

4.2.1 Constructor & Destructor Documentation

4.2.1.1 rushBoard::rushBoard ( )

default constructor for rushBoard

Parameters

none	

Returns

none

Precondition

none

Postcondition

sets bool won to false

4.2.2 Member Function Documentation

4.2.2.1 bool rushBoard::isWon ( ) const

Get Key Function.

Parameters

none	

Returns

bool true if board has been solved

Precondition

none

Postcondition

none

4.2.2.2 void rushBoard::printBoard ( ) const

Get Key Function.

Parameters

none	

Returns

none

Precondition

none

Postcondition

output to terminal the contents of the board

4.2.2.3 void rushBoard::printResults (int *scenarioNum*) const

Get Key Function.

Parameters

int	for the scenerio number

Returns

none

Precondition

none

Postcondition

prints out results of board, if the board has been solved

4.2.2.4 int rushBoard::setupBoard ( )

function that sets up the default game board

Parameters

none	

Returns

gameBoard.setUpBoard()

Precondition

setupBoard()

Postcondition

sets bool won to false

4.2.2.5 void rushBoard::solve ( )

Get Key Function.

Parameters

none	

Returns

none

Precondition

board must be set prior to being solved

Postcondition

solves for the number of moves required to solve a preset board

The documentation for this class was generated from the following files:

- RushHour.h
- RushHour.cpp

4.3 Vehicle Struct Reference

`#include <RushHour.h>`

Public Attributes

- int row
- int col
- int sizeType
- char orientation

4.3.1 Member Data Documentation

4.3.1.1 int Vehicle::col

This is the column number of the vehicle (for ID)

4.3.1.2 char Vehicle::orientation

Determines whether vehicle is horizontal or vertical

4.3.1.3 int Vehicle::row

This is the row number of the vehicle (for ID)

4.3.1.4 int Vehicle::sizeType

Determines whether vehicle is a car or truck

The documentation for this struct was generated from the following file:

- RushHour.h

Chapter 5

File Documentation

5.1 RushHour.cpp File Reference

This program will simulate a rush hour game board and solve the board.

```
#include "RushHour.h"
#include <iostream>
```

Functions

- int main ()

5.1.1 Detailed Description

This program will simulate a rush hour game board and solve the board.

Authors

Liliana Pacheco, Chantelle Marquez Suarez, Chris Eichstedt

Version

Revision 1.1

This program takes in input from the keyboard to make a rush hour game board and place a maximum of eighteen cars. this program then has the potential to solve said 6x6 board if there exist a solution with a maximum of ten move. futher more if you can manually manipulate the board and move individual cars if desired.

Date

Tuesday December 7, 2017

5.1.2 Function Documentation

5.1.2.1 int main ( )

5.2 RushHour.h File Reference

```
#include <iostream>
#include <string>
#include <map>
#include <queue>
```

Classes

- struct Vehicle
- class Board
- class rushBoard

Macros

- #define MAX_GRID_SIZE 6
- #define MAX_NUMBER_OF_VEHICLES 18

5.2.1 Macro Definition Documentation

5.2.1.1 #define MAX_GRID_SIZE 6

5.2.1.2 #define MAX_NUMBER_OF_VEHICLES 18

Index