Introduction

Write a program that automatically generates mazes for a player to solve. The purpose of this assignment is to help you learn to build a basic 2D HTML Canvas game, along with exercising some of those Data Structures & Algorithms neurons from your CS1/2/3 courses. The program will generate random mazes according to a size chosen by the player, then allow the player to solve it, while keeping score and providing the ability to offer hints.

Assignment

Write a 2D HTML Canvas game according to the following specifications...

- Random generation of mazes using one of the Randomized Kruskal's, Randomized Prim's, or the Recursive Division methods, but not the Depth-First algorithm. Refer to the following Wikipedia page for maze generation algorithms: http://en.wikipedia.org/wiki/Maze_generation_algorithm (Links to an external site.)Links to an external site.

- User may choose mazes of size 5x5, 10x10, 15x15 and 20x20

- Hint Option : Tells the user the next best square to choose

- In Game Scoring (just do something good, it doesn't have to be this exactly)
    - 5 points for each correct square along shortest path; first time it is entered.

    - -1 for square adjacent to shortest path; first time it is entered.

    - -2 for all other squares; first time it is entered.

- Simple UI : New Game, High Scores, Credits
    - High Scores only need to be kept for the current session, they do not have to be persisted.

- User Interface that shows the current player score and elapsed time (accurate to the number of seconds).

- Show the location for the end of the maze.

- When starting the game, place a player marker the first cell in the maze.

- As the player moves, they leave a breadcrumb trail behind showing which cells have been visited.

- Other Requirements
    - Ability to toggle a breadcrumbs trail.

    - Ability to toggle shortest path to the finish (remember the stack concept).

    - Ability to toggle the live display of the score.
      Many of the above capabilities mean you have to write code that can find the shortest path. This can be done by using a breadth-first search, but feel free to use any reasonable technique. Here is a wiki link that talks about different maze solving algorithms, with a reference to using breadth-first search to find the shortest path (with multiple solutions)
      http://en.wikipedia.org/wiki/Maze_solving_algorithm (Links to an external site.)Links to an external site.

Control Scheme

Player movement is controlled by using the arrow keys and WASD/IJKL (all three must be active) keys; one key-press moves one square. For the additional requirements, please use the following controls:

- Hint toggle : Keyboard H
- Breadcrumbs toggle : Keyboard B
- Path to finish toggle : Keyboard P
- Score toggle : Keyboard Y

Technical Notes

Your code must have a game loop that follows the pattern learned from the first assignment. It should loop something like...

```
function gameLoop(time) {
    ...compute elapsed time...
    processInput(elapsedTime); // elapsedTime possi
bly not needed
    update(elapsedTime);
    render(elapsedTime);
    requestAnimationFrame(gameLoop);
}
```

I encourage you to place all of your game code inside of a namespace or module named MazeGame. You may spread your code over multiple .js files if you like.

Development Notes

At the time this assignment is given, I haven't presented all the lectures lectures on how to do Canvas rendering and how to collect keyboard input. These techniques are relatively easy and come in the first few days. My very, very strong recommendation is to begin the assignment by first working on developing the JavaScript code to generate the random mazes. You can create a simple HTML page (or using JS and node from a console) and associated JavaScript file and begin working on the maze generation algorithm. You can also work

on the game model, such as modeling the player position in the maze and the different move and scoring capabilities. Once you then know how to render shapes/textures/fonts on an HTML Canvas, along with accepting user input, you'll be in good shape.

JavaScript is different from C++/C#/Java, and the way you think about and organize your code will be different. Take the time to "think" in JavaScript and code accordingly. Again, I don't expect perfection on this assignment, but looking to see progress made.