DASL 130 – C Programming Course

Final Project – Part 1

Due August 19th at 10 AM

The overall goal of this project is to write code that would simulate navigating a robot through a maze. You are given a framework or "skeleton code" to start with, you need to fill in some functions and add more of your own. You are given five maps to start with, in the form of bitmaps. The given code loads the map, and navigates through it using "turn right or left," first trying left then right. Examine the code to see how it works. For part one use the first map. Your goals for part 1 are as follows:

1) Add functions *findStart(map, &startX, &startY, w)* and *findEnd(map, &endX, &endY, w)* which will fill in startX, startY, endX, and endY with the start and end positions in the maze. Pass the map to the function, as well as the variables you want to get back (via reference), and then write that function to search the map for the start and end position respectively. The grayscale value of the start pixel is 29, and the end pixel is 36. – **20 Points**

2) Add code in your main loop that will paint the path you took to get to the end of the map, in gray. In other words make the traversed parts of the map have a value of 128. Uncomment *setMapBmp("inMap1out.bmp", map, w);* to print out the map and see how it worked. Also print out the number of iterations your program took to reach the end. **– 10 Points**

3) Run through all the maps with your code so far, save the outputted bitmaps and the amount of iterations needed to solve each one. Then modify the navigation algorithm so that instead of trying to turn right then trying to turn left, it randomly chooses which direction to turn if it can't go the current direction. Try each map with this algorithm and compare the results to the original. Save your code with the old algorithm before you modify it. **– 20 Points**

E-mail your assignment to nrkuntz@gmail.com

Final Project – Part 2

Due August 26th at 10 AM

4) Implement breadth first search for your navigation algorithm. Refer to the slides for how it should work. Again run this on all the maps and compare your results. Print out the map to different file names several times as it navigates, to show the path. **– 20 Points**

5) Implement depth first search for your navigation algorithm. Refer to the slides for how it should work. Again run this on all the maps and compare your results. Print out the map to different file names several times as it navigates, to show the path. **– 20 Points**

6) Make your own map and run the various methods on this too, try to make it interesting. **– 5 Points**

E-mail your assignment to nrkuntz@gmail.com