Joe Listro
Prof. Cullen Clark
Data Structures and Algorithms

Project Write-Up

In solving a bmp maze, the biggest challenge I faced was reading the bitmap file in from memory to a data structure that I could work with. Having never worked with any images in C++ before, I started this project by searching for an appropriate library to use while interacting with the bitmap file. I settled on this (http://partow.net/programming/bitmap/) library written by Arash Partow.

Once I was able to read in the data from the bitmap file, the next step for me was constructing a maze object with a 2d array member that held a literal pixel by pixel representation of the bitmap maze file. I figured this was the best way to interact with the maze. I stored the white pixels, the open paths, as 0s and the black pixels, the walls, as 1s.

Then I designed a recursive function that would navigate the array one pixel at a time in each direction, returning false back until it finds an open path in another direction, continuing forward in the else case. The recursive nature of such a function that calls itself allowed for me to build a tree of pixel paths, where each pixel is a node and connects to the pixels around it, leaving a leaf where the recursive function is returning false, for instance, if it hits a wall or a pixel that it has already visited, and highlighting the correct path without having to construct a tree or graph object to keep track, as it all is stored in the stack with each recursion.

The pixel idea worked for me, but to optimize it, I designed a pixel block data struct that fits itself to the size of the maze cell blocks and can move as a block of pixels, which keeps the maze looking less like the image on the left, solved pixel by pixel, and outputs an image like the one on the right, solved navigating using the data structure.