CS 51 – Spring 2016

Final Project Proposal

Alex Lin & Melissa Yu

**Title:** the-o-maze-ing-caml

**Objective:** To create an OCaml-based application that generates perfect mazes.

**Project Goals/Implementation:**

* Take advantage of OCaml’s functional paradigm to implement the recursive division algorithm (check the .gif attached for an example) for generating a perfect, rectangular maze.
* Utilize OCaml’s Graphics Module to display the maze on the user’s screen.
* Structure:
  + Maze module: data types for maze - grid and cells; create maze function - divides a grid with specified coordinates, width, and height using the specified resolution; draw maze function
* If extra time permits:
  + Functionally program the reverse backtracking algorithm to provide solutions for the generated mazes
  + Create mazes of different shapes (i.e. circular, triangular, etc.)
  + Implement other maze algorithms (Prim’s, Kruskal’s, etc.) and compare relative efficiencies/performance

**Division of Labor:**

* We will divide the tasks of our project based on the abstraction barrier (creation of module / implementation of module).
  + Alex – create a module with functions that takes in the coordinates of a rectangle and subdivides the rectangle into four other rectangles, while also poking walls in three out of the four generated walls. This is a single, iterative step of the recursive division algorithm.
  + Melissa – recursively call the module’s functions to generate the maze, starting from the entire screen. In each step, four new walls will be created. The algorithm ends when every cell is made. This is the recursive part of the recursive division algorithm.
* Graphics
  + Melissa – writes a graphics function that will render the appropriate line segments to represent the walls on the GUI
  + Alex – formats the computational information into a list of wall coordinates that can be passed to Melissa’s function