**CIS 263 Assignment 3 – Algorithm Design**

**Name: Sean Crowley**

**Due Date**

* at the start of class on Friday, April 17.

**Before Starting the Project**

* Read the entire project description before starting. This may be a pair (no more than 2) or individual assignment.

**Learning Objectives**

After completing this project you should be able to design and implement algorithms using several paradigms.

**Rubric**

25 pts commented and readable code \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

25 pts elegant source code \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

25 pts concise design \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

25 pts results during testing \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Background**

This program will take in a maze input file with starting and ending locations marked on the maze. The program must attempt to reach the end of the maze. A successful run through the maze should be highlighted and output to the screen. Assume the maze has an actual solution (not all dead ends) and proper boundaries (can’t fall off the edge). The maze matrix may be of any dimension, not necessarily square. The starting location will be marked with an S and the ends with an F. There may be multiple correct paths through the maze. You might print the length of the path you find or attempt to find the optimal shortest solution. You MUST mark the path through the maze with an X or other symbol. Solve a maze problem using two or more approaches and compare their performance.

**Step 0: Pick two algorithm approaches**

* Pick two classes of algorithm designs: brute force, backtracking, greedy, divide and conquer, dynamic programming, or random algorithms. Using the template provided, design two solutions using one of the design paradigms and implement the designs using appropriate code. Justify why each of your designs fall into the category you intended it to.

**Step 1: Test with Maze.txt**

* Read in the sample maze files and ensure that your code is correctly solving the maze.

**Step 2: Analysis**

* Without a very large maze, timing code might not detect a noticeable difference in your algorithm designs. Provide the theoretical performance for each of your algorithms.
  + Design: Backtracking O(N)
  + Design: Greedy O(N)

**Step 4: Bundle your program**

* Bundle your program and turn in all files required to run your program.
* Turn in output solution on maze.txt for each of your two algorithms.
* Ensure you include suitable documentation for your 1) overall project (e.g., an appropriate comment block at the top of your program stating which algorithms you used) and 2) source code.
* Hardcopy: Print off the cover page to this document with the theoretical performance, documentation for why your algorithm is in the class you intended, the source code, and output results.
* Softcopy: Upload source code to blackboard.

**Grading Criteria**

* There is a 50% penalty on programming projects if your solution does not execute or if it generates errors.
* There is a 50% penalty for not turning in a hardcopy (both pages of this document and printed code) and softcopy of code (zipped if needed) to blackboard.
* Any options/approaches/requirements not specified in this document are left for your own decision making, in keeping with the spirit of the assignment.

**Late Policy**

Projects are due at the START of the class period and not accepted later. Not turning in the hard copy or soft copy by the due date is considered a late/missing project unless PRIOR arrangements are made.