

CSC221 – Laboratory #1*

David John

Fall 2008

This laboratory assignment focuses on an application of *stack* and a use of a two-dimensional array. You will be using the *Stack* template library that is part of *C++*.

Mazes have fascinated people for years. Today for entertainment people pay to find their way through mazes in corn fields, and mazes with mirrored walls at amusement parks. Every year MIT conducts a contest for with the winner being the best autonomous robot to negotiate a maze. There are many serious applications of searching for a path in a maze, one is an application for robots to search burning buildings and report the shortest path to a victim.

There are many strategies for searching for a path in a maze. We will focus on a stack based approach. You will represent a maze by a two-dimension array of entries of 0 or 1. For example,

```
      010011
      100010  <--  exit
enter -> 011110
      000000
      110101
```

A value of 1 indicates a blocked path and a 0 represents an open path. From any position in the maze the possible moves are the eight directions on a compass.

*Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and I am aware of a trademark claim, the designations have been printed in initial caps, all caps or italics.

Algorithm 1 Rough algorithm that indicates the flow of the overall logic of a stack based solution to find a path from the entrance to the exit of a maze.

```
1: Maze(maze,istart,jstart,direction,iexit,jexit)
2: stack.push(istart, jstart, direction
3: while stack.notempty() do
4:    $(i, j, dir) \leftarrow \text{stack.pop}()$ 
5:   while moves still to be made do
6:      $(g, h) \leftarrow$  coordinates of next move
7:     if  $(g, h) == (iexit, jexit)$  then
8:       Done
9:     end if
10:    if  $(g, h)$  is a legal move and  $(g, h)$  has not been visited before then
11:      mark location  $(g, h)$  as visited
12:       $dir \leftarrow$  next direction to try
13:      stack.push(i, j, dir)
14:       $i \leftarrow g$ 
15:       $j \leftarrow h$ 
16:       $dir \leftarrow north$ 
17:    end if
18:  end while
19: end while
20: No Solution
```

You need to design, implement and test a well written *C++* program that reads the maze configuration from a file, traverses the maze (printing moves as it goes) and then prints the shortest traversal at the end. Algorithm 1 is a simple algorithm to solve the maze program. We will talk about this lab further in class.

If you search on the keyword *maze* in a browser you will find many resources, probably including the source to existing programs. The work that you submit for grading must be the product of your efforts.

This lab is due no later than 5pm on Tuesday, September 23.