Computer Programming
Fall 2018
TEAM PROJECT
Due Jan. 2, 2020
Time Limit: 1 Week

Name (Print): ____________________

University ID: ____________________

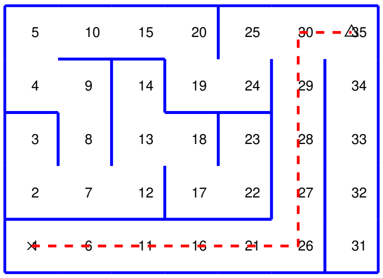
3 pages (including the cover)

1. Write a computer program that can automatically solve the maze.

   (a) (50 points) For example shown below is a 5 by 7 maze created randomly (see "maze_prob1_5_by_7.txt"). Your program should be able to solve it, which starts at location marked by △ (1) and reaches the location marked ×(35) (see "maze_prob1_5_by_7_in_and_out.txt"). Find such a path and store your result in a file named "maze_route.txt". The format is a sequence of number separated by a comma, for example,

   1,6,11,16,21,26,27,28,29,30,35

   is stored in "maze_route.txt".



   A package is also provided to plot the maze map and the calculated route as they are shown above. The matlab scirpt should work when all the script files (*.m, *.fig) are placed in the same folder.

   1 Execute "maze2_plot_gui.m" in matlab command window.

   2 Push the bottom "show map first" and select a map file and the initial and destinamtion file. For example "maze_prob1_5_by_7.txt" is the map file and the initial and final position is in "maze_prob1_5_by_7_in_and_out.txt".

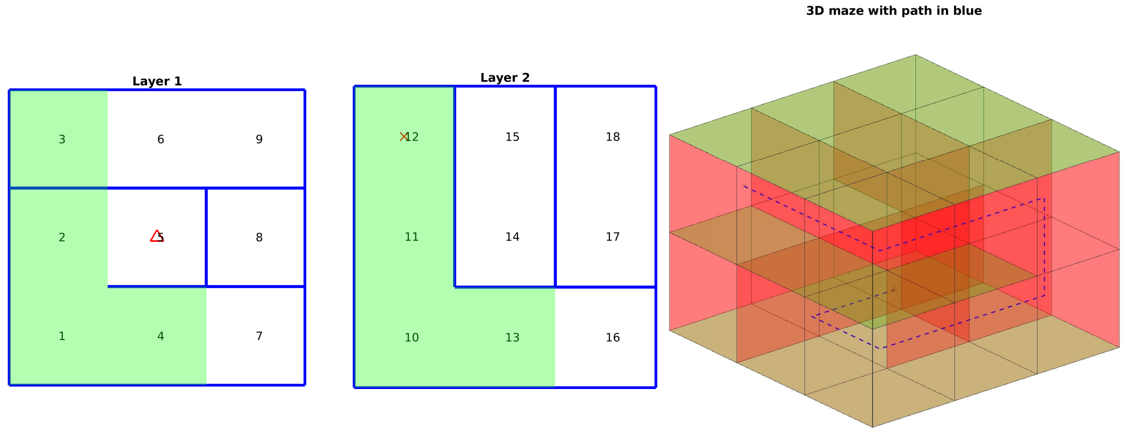
   3 If you successfully find the path and stored in the result file named after "maze_route.txt". You can push the buttom "Put Route on map" to see the path found along with the map of the maze.

   4 You can also check the capability of your program by solving additional mazes that are also included in the subfolders.

   (b) (50 points) A 3D maze solving. In the following example a 3 by 3 by 2 maze is shown (maze3_3_3x3x2.txt). Suppose a task is given to start at 12 and ultimately reach 5. Shaded area in maps layer 1 and 2 are the walls between the two layers, and the connecting routes for layer 1 and 2 are in white. Your program should be able to solve the route:

12,11,10,13,16,7,4,1,2,5

and store your result in a file named ”maze_3D_route.txt”.



Sample mazes are attached.

- 2D mazes are placed in folder ”maze2_data_route”. The first two columns in data are the position (x,y) of a cell and the last 4 columns are the ”wall” of a cell arranged by (N,S,E,W). 0 means ”NO wall”, 1 means there is ”wall” and 2 means the maze boundary. It is important to note that the index, **q**, shown on the maze map of *row*-rows by *col*-columns is converted to the position (x,y) of cell by

  $$\mathbf{q} = y + (x - 1) * row$$

  where $1 \leq x \leq col$ and $1 \leq y \leq row$

- As to the 3D maze data in .txt, the first three columns are the position (x,y,z) of the cell and the remaining 6 columns are walls (N, S, E, W, U, D) where N is north, S is south, E is east, W is west, U is the ceiling, D is floor. 0 means ”NO wall”, 1 means there is ”wall” and 2 means the maze boundary. It is important to note that the index, **q**, shown on the maze map of *row*-rows by *col*-columns by *h*-height is converted to the position (x,y,z) of cell by

  $$\mathbf{q} = y + (x - 1) * row + (z - 1) * row * col$$

  where $1 \leq x \leq col$, $1 \leq y \leq row$ and $1 \leq z \leq h$

Report Format (Print and Submit in class)

- Names and university ID of the team members should be printed. Number of student in one group should be less and equal to ”TWO”.
- Program analysis: Input, output and data format; Functions; (Specifications of the program is required)
- Algorithm: Programming logic and mathematical backgrounds
- Program tracing: illustration of how program works, memory maps
- Example of solving the mazes (Maze samples will be given online)

Program test:You should bring your source code that can be compiled and executed. A randomly generated maze of different sizes will be given during test.