Project Abstract

1. 成員: 資管二 403530051 廖士豪

資管二 403530055 蔡沅廷

資管二 403530035 張楷睿

1. We have chosen problem 18 Escaping a Maze.
2. Puzzle is very interesting!

想寫一些以前沒寫過的程式

最近Alpha Go正紅,所以想要練習一下有關Artificial Intelligence的題目

1. 牆壁設1

空地設0

走過的路徑設2

迷宮右手法則

EAST MAZE[x][y+1]

WEST MAZE[x][y-1]

SOUTH MAZE[x+1][y]

NORTH MAZE[x-1][y]

假設起點座標(0,0)

若走回原點回傳false

若走出地圖回傳true

目前大致上有的雛型長這樣

1. **import** DSwJ.S14.ALStack;
3. **public** **class** Maze {
4. **public** **static** **final** **int** ExitX = 10;
5. **public** **static** **final** **int** ExitY = 8;
6. **public** **int** curX=1;
7. **public** **int** curY=1;
8. **protected** ALStack history;
9. **private** **int**[][] MAZE = {{1,1,1,1,1,1,1,1,1,1,1,1},
10. {1,0,0,0,1,1,1,1,1,1,1,1},
11. {1,1,1,0,1,1,0,0,0,0,1,1},
12. {1,1,1,0,1,1,0,1,1,0,1,1},
13. {1,1,1,0,0,0,0,1,1,0,1,1},
14. {1,1,1,0,1,1,0,1,1,0,1,1},
15. {1,1,1,0,1,1,0,1,1,0,1,1},
16. {1,1,1,1,1,1,0,1,1,0,1,1},
17. {1,1,0,0,0,0,0,0,1,0,0,1},
18. {1,1,1,1,1,1,1,1,1,1,1,1}};
20. **public** Maze() {history = **new** ALStack();}
22. **public** **boolean** start(){
23. go(curX,curY);
24. **while**(**true**) {
25. **if**(north()==0){
26. goNorth();
27. } **else** **if**(south()==0) {
28. goSouth();
29. } **else** **if**(west()==0) {
30. goWest();
31. } **else** **if**(east()==0) {
32. goEast();
33. } **else** {
34. Pos backPos = back();
35. **if**(backPos==**null**){
36. System.out.println("Exit is unreachable!!!");
37. **return** **false**;
38. } **else** {
39. curX = backPos.getX();
40. curY = backPos.getY();
41. }
42. }
43. **if**(isExit())**return** **true**;;
44. }
45. }
47. **public** **int** north(){**return** MAZE[curY-1][curX];}
48. **public** **int** south(){**return** MAZE[curY+1][curX];}
49. **public** **int** west(){**return** MAZE[curY][curX-1];}
50. **public** **int** east(){**return** MAZE[curY][curX+1];}
52. **public** **void** goNorth(){go(curX,--curY);}
53. **public** **void** goSouth(){go(curX,++curY);}
54. **public** **void** goEast(){go(++curX,curY);}
55. **public** **void** goWest(){go(--curX,curY);}
56. **public** **void** go(**int** x, **int** y) {
57. MAZE[y][x]=2;
58. history.push(**new** Pos(x,y));
59. }
61. **public** Pos back(){
62. **return** history.pop();
63. }
65. **public** **boolean** isExit(){**if**(curX==ExitX&&curY==ExitY)**return** **true**;**return** **false**;}
67. **public** **void** showMaze(){
68. **for**(**int** i=0; i
69. **for**(**int** j=0; j
70. System.out.print(MAZE[i][j]+" ");
71. System.out.println();
72. }
73. }
75. **public** **static** **void** main(String args[]) {
76. Maze maze = **new** Maze();
77. **if**(maze.start()) {
78. System.out.println("Conquer the maze ^^.");
79. } **else** {
80. System.out.println("I lost in the maze ==\".");
81. }
82. maze.showMaze();
83. }
84. }