ID Name

미로 찾기

다음 코드는 미로(maze)를 찾는 코드입니다.

package backtrack;

import java.util.\*;

public class Maze {

private static final int NN = 10;

private static final int CC = 2;

private static Stack stack = new Stack( );

private static int exitRow = (NN - 1), exitCol = (NN - 1);

private static boolean[ ][ ] hasVisited = new boolean[ NN ][ NN ];

// boolean array is initialized to false by default

public static void stackPush( int [ ] cod ) {

stack.push( cod );

// System.out.println( "stackPush: [ " + stack.size( ) + " ][ " + cod[ 0 ] + " ][ " + cod[ 1 ] + " ]" );

return;

}

public static int[ ] stackPop( ) {

int[ ] cod = ( int[ ] )stack.pop( );

// System.out.println( "stackPop: [ " + cod[ 0 ] + " ][ " + cod[ 1 ] + " ]" );

return cod;

}

public static void checkFourDirections( int maze[ ][ ], int coord[ ] ) {

int row = coord[ 0 ];

int col = coord[ 1 ];

int dir = 0;

hasVisited[ row ][ col ] = true;

stackPush( coord );

// find an exit: success

if( row == exitRow && col == exitCol ) {

printSolution( );

}

while( dir >= 0 ) {

boolean keep = false;

// System.out.println( " Current row [ " + row + " ] col [ " + col + " ][ " + dir + " ]");

if( dir == 0 ) { // north -- up

if( ( row - 1 >= 0 ) && ( maze[ row - 1 ][ col ] == 1 ) && ( !hasVisited[ row - 1 ][ col ] ) ) {

coord[ 0 ] = row - 1;

coord[ 1 ] = col;

keep = true;

}

} else if( dir == 3 ) { // west -- left

if( ( col - 1 >= 0 ) && ( maze[ row ][ col - 1 ] == 1 ) && ( !hasVisited[ row ][ col - 1 ] ) ) {

coord[ 0 ] = row;

coord[ 1 ] = col - 1;

keep = true;

}

} else if( dir == 1 ) { // east -- right

if( ( col + 1 < NN ) && ( maze[ row ][ col + 1 ] == 1 ) && ( !hasVisited[ row ][ col + 1 ] ) ) {

coord[ 0 ] = row;

coord[ 1 ] = col + 1;

keep = true;

}

} else if( dir == 2 ) { // south -- down

if( ( row + 1 < NN ) && ( maze[ row + 1 ][ col ] == 1 ) && ( !hasVisited[ row + 1 ][ col ] ) ) {

coord[ 0 ] = row + 1;

coord[ 1 ] = col;

keep = true;

}

} else { // dir == 4?

stackPop( );

break;

}

++dir;

if (keep) {

checkFourDirections( maze, coord );

}

}

return;

}

public static void main(String[] args) {

int maze[][] = {

// 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

{1, 0, 0, 0, 1, 0, 0, 0, 0, 1}, // 0

{1, 1, 1, 0, 1, 1, 1, 1, 0, 0}, // 1

{0, 1, 0, 1, 1, 0, 0, 1, 0, 1}, // 2

{1, 1, 1, 1, 0, 1, 0, 1, 0, 0}, // 3

{0, 0, 1, 0, 1, 0, 1, 1, 1, 1}, // 4

{0, 1, 1, 0, 1, 1, 0, 0, 1, 0}, // 5

{0, 1, 0, 1, 0, 0, 1, 1, 1, 0}, // 6

{1, 1, 1, 1, 1, 0, 1, 1, 0, 0}, // 7

{0, 0, 1, 0, 0, 1, 0, 1, 0, 1}, // 8

{0, 0, 1, 1, 1, 1, 1, 1, 1, 1}, // 9

};

int[ ] coord = new int[ CC ];

checkFourDirections( maze, coord );

System.out.println( "No paths have been found" );

return;

}

public static void printSolution( ) {

System.out.println( "\n\n----------" );

System.out.println( " success " );

System.out.println( "----------\n\n" );

for( int idx = 0; idx < stack.size( ); idx++ ) {

int cod[ ] = ( int[ ] )stack.get( idx );

System.out.println( "[ " + cod[ 0 ] + " ][ " + cod[ 1 ] + " ] ==> " );

}

System.exit( 0 );

}

}

1. 위에 코드에서 미로를 찾는 알고리즘과 수업시간에 배운 알고리즘과 다릅니다. 그 차이를 알고리즘 관점에서 설명하세요.

수업에서 배운 알고리즘이 좀 더 divide & conquer 가 잘 되어 있습니다. 바로 절차 지향적으로 쓰는 이 코드와는 달리 수업에서 배운 코드는 method로 분화 시켜놓아 재사용이 더욱 쉽습니다.

1. 위에 코드는 미로를 찾는 경로를 바르게 출력을 하지 않습니다. 미로를 찾는 알고리즘은 올바르지만 스택을 다루를 부분에 문제가 있습니다. 코드를 수정해서 미로의 경로를 바르게 출력하세요.

* 코드명 Maze1.java

1. 위에 코드에서 미로를 찾는 경로는 여러 개가 있습니다.
   1. 위에 코드를 미로의 모든 경로를 찾는 코드로 수정하세요. 코드명

* 코드명 Maze2.java
  1. 찾은 경로들 중 최단 경로 하나만 출력하는 코드로 수정하세요.
* 코드명 Maze3.java

1. 위에 코드는 recursive 방식인데 이를 iterative 방식으로 수정하세요.

* 코드명 Maze4.java

MAC 주소

1. 노트북의 무선랜 MAC 주소를 적으세요. MAC 주소를 찾는 방법은 TA에게 질문하세요.

C8-F7-33-B9-D0-C0

* 예) 39-34-DE-AD-DC-29

코드 작성요령

* 수정하는 코드의 줄 수가 적을수록 더 높은 점수를 받습니다.
* 수정된 모든 코드의 행 말미에 /\* XXXX \*/ 를 붙입니다.
* 수정하는 코드에 의도하는 바를 바로 윗줄에 영문 또는 국문으로 주석표시 합니다.
  + 예)

1. /\* this routine will call a routine for this \*/
2. checkFourDirections( maze, coord ); /\* XXXX \*/

답안 제출

1. 문제 1과 문제 5 답안은 문제 파일에 직접 작성을 하세요. 파일명을 학생의 “학번\_한글이름.docx”로 변경하세요.
2. 문제 2, 문제 3 그리고 문제 4로 작성된 코드 파일명을 주어진 이름으로 변경하세요.
3. 새로운 폴더를 만들어 폴더명을 학생의 “학번한글이름”으로 변경하고 모든 파일을 이 폴더 아래로 복사하세요.
4. 폴더를 압축하고 파일명을 “학번\_한글.zip”으로 변경하세요.
5. 정해진 시간에 iCampus에 제출하세요.