Lab9: Maps and Maze Game

Terminology:

A Map is going to be represented by a 2d array of ints where 0 represents an empty spot, 1 represents you, 2 represents the finish line or goal, and 3 represents a wall or obstacle.

Task 1: Create the following Map constructor functions:

```
// Creates an empty map with r rows and c columns.
public static int[][] createMap(int r, int c);
```

//Insert Code with comments here:

```
// Creates a map with the following layout, where "Y" = you, "F" =
// finish line, and * represents wall, and _ represents empty:
//    Y * _ _ _ _ _ _ _ _
//    _ * _ * _ * _ * * _
//    _ _ _ * _ _ _ * _ _
//    * _ * _ * * * _ _ *
//    _ _ _ _ _ _ * _ * _
//    _ * * * _ _ _ * _ _
//    _ _ _ _ * * _ _ _ *
//    _ * _ _ * _ * _ * _
//    _ _ _ * _ _ _ _ _ _
//    _ * _ _ * _ _ * _ F
public static int[][] createMap();
```

//Insert Code with comments here:

```
// Creates a 10x10 map where you and the finish line are randomly placed,
// and there are exactly n walls placed randomly throughout the map.
// If this is too hard, you may place you in the upper left, and the finish
// in the bottom right.
public static int[][] createMap(int n);
```

//Insert Code with comments here:

Task 2: Create the following utility functions:

```
// Returns the String representation of the int map element.
// IE, symbol(0) => "_", symbol(1) => "Y", symbol(2) => "F", and symbol(3) => "*"
public static String getSymbol(int value);
```

//Insert Code with comments here:

Lab9: Maps and Maze Game

```
// Prints a nice looking map such as the following, with 2 space indentation:
  Y * _ _ _ _ _ _ _ _
  _ * _ * _ * _ * * _
  _ _ _ * _ _ _ * _ _
  * _ * _ * * * _ _ *
  _ _ _ _ _ _ * _ * _
  _ * * * _ _ _ * _ _
  _ _ _ _ * * _ _ _ *
  _ * _ _ * _ * _ * _
  _ _ _ * _ _ _ _ _ _
  _ * _ _ * _ _ * _ F
```

`public static void printMap(int[][] m);`

//Insert Code with comments here:

```
// Returns a nice looking map as a String.
```

`public static String mapToString(int[][] m);`

//Insert Code with comments here:

Lab9: Maps and Maze Game

```
// Returns the coordinates of any instance of value in the Map m.
// Return null if the value cannot be found on the map.
// This function is intended to find unique values only.
// Ex, find( {{0,0,3},{3,3,2},{1,3,0}} , 1) => {2,0}
// Ex, find( {{0,0,3},{3,3,2},{1,3,0}} , 4) => null
public static int[] find(int[][] m, int value);
```

//Insert Code with comments here:

```
// Returns true if you can move north/south/east/west, false otherwise.
public static boolean canGoN(int[][] m);
public static boolean canGoS(int[][] m);
public static boolean canGoE(int[][] m);
public static boolean canGoW(int[][] m);
```

//Insert Code with comments here:

Lab9: Maps and Maze Game

Challenge #7:

Create a function that returns a minimal path from you to finish. A minimal path is any path from you to finish where any other path is the same number of steps or longer. Minimal paths may not exist, and are not guaranteed to be unique. The function interface is left to you.

//Insert Code with comments here:

Challenge #8:

Create the groundwork to implement a 3d Maze Game. 4 should represent a ladder up, denoted as "U" when printing, and 5 should represent a ladder down, denoted as "D" when printing.

```
// Create a blank 3d Map with the given sizes.
public static int[][][] create3dMap(int r, int c, int numFloors);
```

//Insert Code with comments here:

```
// Create an interesting 3d Map with walls and ladders prefabricated, 19x19 base.
public static int[][][] create3dMap();
```

//Insert Code with comments here:

```
// Create a pyramid with randomized ladders, but no walls, and no finish.
// You always start on the bottom floor in the upper left.
// We will assume the goal is always to get the the very top of the pyramid.
// Ex, the floors of the pyramid might look like this:
Level 1:
  Y _ _ _ _
  _ _ _ _ _
  _ _ _ _ _
  _ _ _ U _
  _ _ _ _ _
Level 2:
  * * * * *
  * _ _ _ *
  * _ U _ *
  * _ _ D *
  * * * * *
Level 3:
  * * * * *
  * * * * *
  * * F * *
  * * * * *
```

Lab9: Maps and Maze Game

```
  * * * * *
public static int[][][] create3dMap(int numFloors);
```

//Insert Code with comments here:

```
public static boolean canGoU(int[][][] m);
public static boolean canGoD(int[][][] m);
```

//Insert Code with comments here:

```
// mapToString() should return a String with that looks like this:
Level 1:
  Y * _ U
  _ * _ *
  _ _ _ *
Level 2:
  _ F * D
  _ * * _
  _ _ _ _
public static String mapToString(int[][][] m);
```

//Insert Code with comments here: