Write a `Maze` class with a `search` method that uses recursion to traverses a maze by following space-character passageways through a maze whose walls are 'X' characters. The program driver below specifies a layout with an array of strings – one string for each row in the two-dimensional maze.

```
/*************************************************************
* MazeDriver.java
* Dean & Dean
*
* This drives a maze traverse.
*************************************************************/

public class MazeDriver
{
  public static void main(String[] args)
  {
    String[] mazeStrings = new String[]
    {"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
     "XXXXXXXXXXXX                 XX",
     "XXXX         XXXXXXXXXXXX XXXX",
     "XXXX XXXXX XXXXXXX          XXX",
     "XXXX XXXXX XXXXXXX XXXXXXXXXXX",
     "XXXX       XXXXXXXXXXXXXXXXXXX",
     "XXXX XXXXX                XXXXX",
     "     XXXXX XXXXXXXXXXXXX XXXXX",
     "X XXXXXXXX XXXXXXXXXXXXX XXXXX",
     "X XXXXXXXXXXXXX      XXX   XXX",
     "X   XXXXXXXXXXXXX   XXXXX XXX",
     "XXX XXXXXXXXXXXXX   XXXXX XXX",
     "XXX          XXXXX     XXXXXXX",
     "XXX XXXXXXXX XXXXXXXXX XXXXXXX",
     "XXX XXXXXXXX XXXXXXXXX XXXXXXX",
     "XXX     XXXX XXXXXXXXX         ",
     "XXX XXX XXXX XXXXXXXXX XXXXXXX",
     "XXX     XXXX XXXXXXXXX XXXXXXX",
     "XXXXXXXXXXXX            XXXXXXX",
     "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"};

    Maze maze = new Maze(mazeStrings);
    maze.search(7, 0, 0.0); // start row, start col, start angle
  } // end main
} // end MazeDriver class
```

This particular maze is 30 columns wide and 20 rows high. The `X`s are boundaries and the clear spaces are passages. There is an opening on the left side at row 7 and column 0, and there is another opening on the right at row 15 and column 29. There are two islands created by cross linking passageways, and there is one open area that is three columns wide.

To traverse the maze, specify a starting row and column and an initial heading angle that is not into a wall. For example, you might start with row 7, column 0, and angle 0 to indicate entrance at the left-side opening. Or you might start with row 15, column 29, and angle 180 to indicate entrance at the right-side opening. An angel of +90 is up and an angle of −90 is down. You could make the

traverse by always following the wall on the left or always following the wall on the right. Our solution always follows the wall on the left.

Your `search` method should contain a `for` loop that steps through four angles. To follow left walls, those four angles are: 90 degrees to the left of straight ahead, straight ahead, 90 degrees to the right of straight ahead, and straight behind − in that order. If the next cell in the currently selected direction is outside the array limits, you've found an exit, so print the exit location and return `true`. Otherwise, if the next cell in the selected direction is not 'X', and if the direction is not straight ahead, print a change-direction message and make a recursive call to proceed in the selected direction. If the recursive call returns `true`, then return `true`. If the `for` loop terminates, return `false`.

When driven by the `main` method above, your program should produce the output below. Although the solution that produces this output works, and the output makes it easy to follow what happens, the solution that produces this output is not the most efficient solution. For a more efficient solution reduce the `for` loop iterations to eliminate the final step that goes straight back. Explain why this more efficient solution is able to produce the same final result.

Sample session:

```
(7,0) (7,1) (7,2) (7,3) (7,4)
 change angle to 90
(6,4) (5,4) (4,4) (3,4) (2,4)
 change angle to  0
(2,5) (2,6) (2,7) (2,8) (2,9) (2,10) (2,11) (2,12)
 change angle to 90
(1,12)
 change angle to  0
(1,13) (1,14) (1,15) (1,16) (1,17) (1,18) (1,19) (1,20) (1,21)
(1,22) (1,23) (1,24) (1,25) (1,26) (1,27)
 change angle to -180
(1,26) (1,25)
 change angle to -90
(2,25) (3,25)
 change angle to  0
(3,26)
 change angle to -180
(3,25) (3,24) (3,23) (3,22) (3,21) (3,20) (3,19) (3,18)
 change angle to -90
(4,18)
 change angle to -270
(3,18)
 change angle to -360
(3,19) (3,20) (3,21) (3,22) (3,23) (3,24) (3,25)
 change angle to -270
(2,25) (1,25)
 change angle to -180
(1,24) (1,23) (1,22) (1,21) (1,20) (1,19) (1,18) (1,17) (1,16)
(1,15)(1,14) (1,13) (1,12)
 change angle to -90
(2,12)
 change angle to -180
(2,11) (2,10)
 change angle to -90
```

```
(3,10) (4,10) (5,10) (6,10)
 change angle to  0
(6,11) (6,12) (6,13) (6,14) (6,15) (6,16) (6,17) (6,18) (6,19)
(6,20) (6,21) (6,22) (6,23) (6,24)
 change angle to -90
(7,24) (8,24) (9,24)
 change angle to  0
(9,25) (9,26)
 change angle to -90
(10,26) (11,26)
 change angle to -270
(10,26) (9,26)
 change angle to -180
(9,25) (9,24)
 change angle to -270
(8,24) (7,24) (6,24)
 change angle to -180
(6,23) (6,22) (6,21) (6,20) (6,19) (6,18) (6,17) (6,16) (6,15)
(6,14) (6,13) (6,12) (6,11) (6,10)
 change angle to -90
(7,10) (8,10)
 change angle to -270
(7,10) (6,10) (5,10)
 change angle to -180
(5,9) (5,8) (5,7) (5,6) (5,5) (5,4)
 change angle to -90
(6,4) (7,4)
 change angle to -180
(7,3) (7,2) (7,1)
 change angle to -90
(8,1) (9,1) (10,1)
 change angle to  0
(10,2) (10,3)
 change angle to -90
(11,3) (12,3)
 change angle to  0
(12,4) (12,5) (12,6) (12,7) (12,8) (12,9) (12,10) (12,11) (12,12)
 change angle to -90
(13,12) (14,12) (15,12) (16,12) (17,12) (18,12)
 change angle to  0
(18,13) (18,14) (18,15) (18,16) (18,17) (18,18) (18,19) (18,20)
(18,21) (18,22)
 change angle to 90
(17,22) (16,22) (15,22) (14,22) (13,22) (12,22)
 change angle to 180
(12,21) (12,20) (12,19) (12,18)
 change angle to 90
(11,18) (10,18) (9,18)
 change angle to 180
(9,17) (9,16) (9,15)
 change angle to  0
(9,16) (9,17) (9,18) (9,19) (9,20)
```

```
 change angle to -90
(10,20) (11,20) (12,20)
 change angle to  0
(12,21) (12,22)
 change angle to -90
(13,22) (14,22) (15,22)
 change angle to  0
(15,23) (15,24) (15,25) (15,26) (15,27) (15,28) (15,29)
left maze at (15,30)
```

What happens if there is no second opening the in the maze's outer wall? What happens if there are no openings in the outer wall, and the seeker "parachutes" in to some interior starting point?