3. A crossword puzzle grid is a two-dimensional rectangular array of black and white squares. Some of the white squares are labeled with a positive number according to the *crossword labeling rule*.

   The crossword labeling rule identifies squares to be labeled with a positive number as follows.

   A square is labeled with a positive number if and only if
   - the square is white and
   - the square does not have a white square immediately above it, or it does not have a white square immediately to its left, or both.

   The squares identified by these criteria are labeled with consecutive numbers in row-major order, starting at 1.

   The following diagram shows a crossword puzzle grid and the labeling of the squares according to the crossword labeling rule.
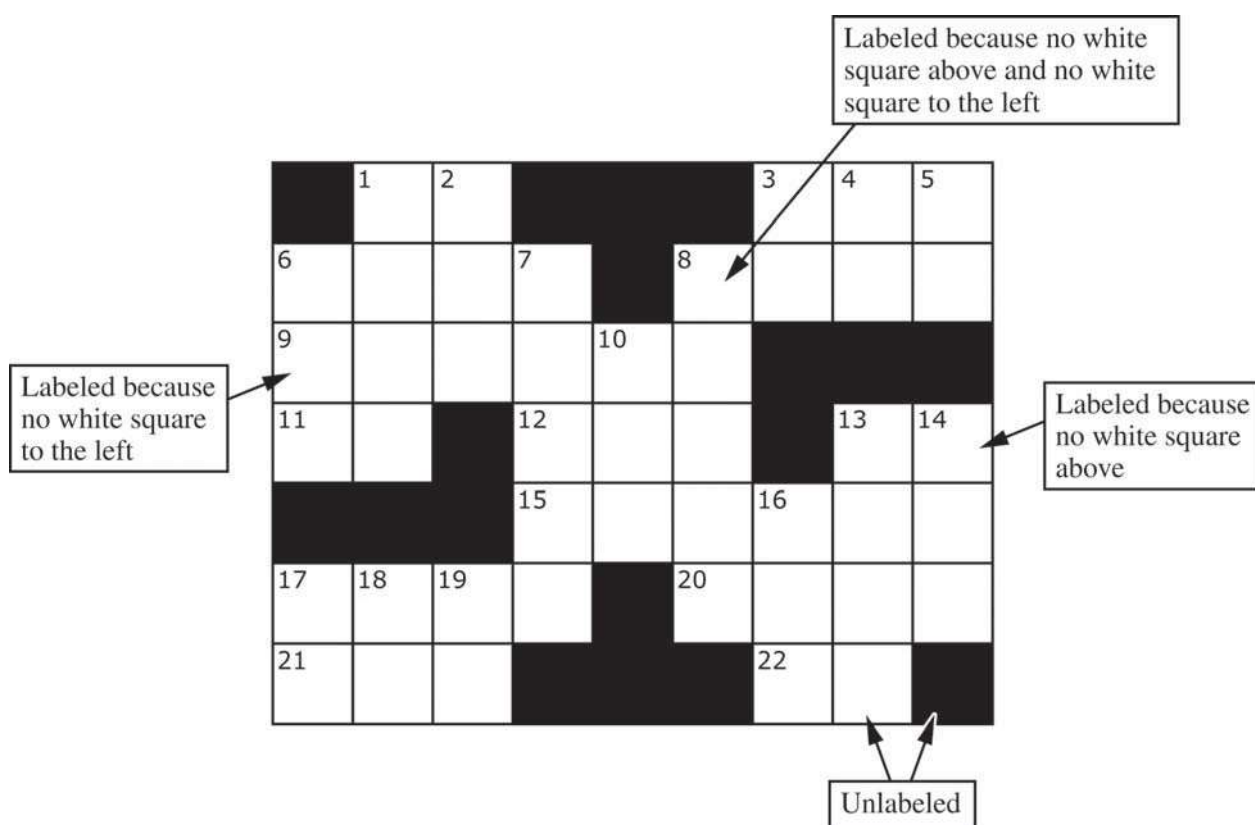
This question uses two classes, a `Square` class that represents an individual square in the puzzle and a `Crossword` class that represents a crossword puzzle grid. A partial declaration of the `Square` class is shown below.

```
public class Square
{
    /**  Constructs one square of a crossword puzzle grid.
     *   Postcondition:
     *      - The square is black if and only if isBlack is true.
     *      - The square has number num.
     */
    public Square(boolean isBlack, int num)
    {   /*  implementation not shown  */   }


    //  There may be instance variables, constructors, and methods that are not shown.
}
```

A partial declaration of the `Crossword` class is shown below. You will implement one method and the constructor in the `Crossword` class.

```
public class Crossword
{
    /**  Each element is a Square object with a color (black or white) and a number.
     *   puzzle[r][c]  represents the square in row  r,  column  c.
     *   There is at least one row in the puzzle.
     */
    private Square[][] puzzle;

    /**  Constructs a crossword puzzle grid.
     *   Precondition: There is at least one row in blackSquares.
     *   Postcondition:
     *      - The crossword puzzle grid has the same dimensions as blackSquares.
     *      - The Square object at row  r,  column  c  in the crossword puzzle grid is black
     *        if and only if blackSquares[r][c]  is true.
     *      - The squares in the puzzle are labeled according to the crossword labeling rule.
     */
    public Crossword(boolean[][] blackSquares)
    {   /*  to be implemented in part (b)  */   }

    /**  Returns  true  if the square at row  r,  column  c  should be labeled with a positive number;
     *           false  otherwise.
     *   The square at row  r,  column  c  is black if and only if blackSquares[r][c]  is true.
     *   Precondition:  r  and  c  are valid indexes in  blackSquares.
     */
    private boolean toBeLabeled(int r, int c, boolean[][] blackSquares)
    {   /*  to be implemented in part (a)  */   }


    //  There may be instance variables, constructors, and methods that are not shown.
}
```

Part (a) begins on page 14.

GO ON TO THE NEXT PAGE.

(a) Write the `Crossword` method `toBeLabeled`. The method returns `true` if the square indexed by row `r`, column `c` in a crossword puzzle grid should be labeled with a positive number according to the crossword labeling rule; otherwise it returns `false`. The parameter `blackSquares` indicates which squares in the crossword puzzle grid are black.

```
Class information for this question

public class Square

public Square(boolean isBlack, int num)

public class Crossword

private Square[][] puzzle

public Crossword(boolean[][] blackSquares)
private boolean toBeLabeled(int r, int c, boolean[][] blackSquares)
```

WRITE YOUR SOLUTION ON THE NEXT PAGE.

Complete method `toBeLabeled` below.

```
/** Returns true if the square at row r, column c should be labeled with a positive number;
 *          false otherwise.
 * The square at row r, column c is black if and only if blackSquares[r][c] is true.
 * Precondition: r and c are valid indexes in blackSquares.
 */
private boolean toBeLabeled(int r, int c, boolean[][] blackSquares)
```

Part (b) begins on page 16.

GO ON TO THE NEXT PAGE.

(b) Write the `Crossword` constructor. The constructor should initialize the crossword puzzle grid to have the same dimensions as the parameter `blackSquares`. Each element of the puzzle grid should be initialized with a reference to a `Square` object with the appropriate color and number. The number is positive if the square is labeled and 0 if the square is not labeled.

Class information for this question

<u>public class Square</u>

public Square(boolean isBlack, int num)

<u>public class Crossword</u>

private Square[][] puzzle

public Crossword(boolean[][] blackSquares)
private boolean toBeLabeled(int r, int c, boolean[][] blackSquares)

WRITE YOUR SOLUTION ON THE NEXT PAGE.

Assume that `toBeLabeled` works as specified, regardless of what you wrote in part (a). You must use `toBeLabeled` appropriately to receive full credit.

Complete the `Crossword` constructor below.

```
/** Constructs a crossword puzzle grid.
 *   Precondition: There is at least one row in blackSquares.
 *   Postcondition:
 *      - The crossword puzzle grid has the same dimensions as blackSquares.
 *      - The Square object at row r, column c in the crossword puzzle grid is black
 *        if and only if blackSquares[r][c] is true.
 *      - The squares in the puzzle are labeled according to the crossword labeling rule.
 */
public Crossword(boolean[][] blackSquares)
```

GO ON TO THE NEXT PAGE.