

4. This question involves reasoning about a number puzzle that is represented as a two-dimensional array of integers. Each element of the array initially contains a value between 1 and 9, inclusive. Solving the puzzle involves clearing pairs of array elements by setting them to 0. Two elements can be cleared if their values sum to 10 or if they have the same value. The puzzle is considered solved if all elements of the array are cleared.

You will write the constructor and one method of the `SumOrSameGame` class, which contains the methods that manipulate elements of the puzzle.

```
public class SumOrSameGame
{
    private int[][] puzzle;

    /**
     * Creates a two-dimensional array and fills it with random integers,
     * as described in part (a)
     * Precondition: numRows > 0; numCols > 0
     */
    public SumOrSameGame(int numRows, int numCols)
    { /* to be implemented in part (a) */ }

    /**
     * Identifies and clears an element of puzzle that can be paired with
     * the element at the given row and column, as described in part (b)
     * Preconditions: row and col are valid row and column indices in puzzle.
     * The element at the given row and column is between 1 and 9, inclusive.
     */
    public boolean clearPair(int row, int col)
    { /* to be implemented in part (b) */ }

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

- A. Write the constructor for the `SumOrSameGame` class. The constructor initializes the instance variable `puzzle` to be a two-dimensional integer array with the number of rows and columns specified by the parameters `numRows` and `numCols`, respectively. Array elements are initialized with random integers between 1 and 9, inclusive, each with an equal chance of being assigned to each element of `puzzle`.

When an element of the two-dimensional array is accessed, the first index is used to specify the row and the second index is used to specify the column.

Complete the `SumOrSameGame` constructor.

```
/**
 * Creates a two-dimensional array and fills it with random integers,
 * as described in part (a)
 * Precondition: numRows > 0; numCols > 0
 */
public SumOrSameGame(int numRows, int numCols)
```

B. Write the `clearPair` method, which takes a valid row index and valid column index as its parameters. The array element specified by those indices, which has a value between 1 and 9, inclusive, is compared to other array elements in `puzzle` in an attempt to pair it with another array element that meets both of the following conditions.

- The row index of the second element is greater than or equal to the parameter `row`.
- The two elements have equal values or have values that sum to 10.

If such an array element is found, both array elements of the pair are cleared (set to 0) and the method returns `true`. If more than one such array element is found, any one of those identified array elements can be used to complete the pair and can be cleared. If no such array element is found, no changes are made to `puzzle` and the method returns `false`.

The following table shows the possible results of several calls to `clearPair`.

puzzle before call to <code>clearPair</code>	Method Call	puzzle after call to <code>clearPair</code>	Return Value	Explanation
0 7 9 0 7 4 1 6 8 4 1 8	<code>clearPair(0, 1)</code>	0 0 9 0 0 4 1 6 8 4 1 8	<code>true</code>	The value 7 in row 0, column 1 is matched with the value 7 in row 1, column 0.
1 2 3 4 5 6 7 8 5 4 1 2	<code>clearPair(2, 2)</code>	1 2 3 4 5 6 7 8 5 4 1 2	<code>false</code>	There is no element in row 2 or a later row to pair with the value 1.
8 1 0 5 0 4 3 6 3 4 5 8	<code>clearPair(1, 1)</code>	8 1 0 5 0 0 3 0 3 4 5 8	<code>true</code>	The value 4 in row 1, column 1 is matched with the value 6 in row 1, column 3. It could also have been matched with the value 4 in row 2, column 1.
1 7 9 2 6 5 4 4 4	<code>clearPair(0, 2)</code>	0 7 0 2 6 5 4 4 4	<code>true</code>	The value 9 in row 0, column 2 is matched with the value 1 in row 0, column 0.

Complete method `clearPair`.

```
/**
 * Identifies and clears an element of puzzle that can be paired with
 * the element at the given row and column, as described in part (b)
 * Preconditions: row and col are valid row and column indices in puzzle.
 * The element at the given row and column is between 1 and 9, inclusive.
 */
public boolean clearPair(int row, int col)
```

STOP
END OF EXAM