Continue to work with your partner on this assignment. Don't forget to switch roles often.

Exercise 2.0: Unblocking

Complete the following private helper `removeBlocks` method in your `Tetrad` class.

```
// precondition:  Blocks are in the grid.
// postcondition: Returns old locations of blocks;
//                blocks have been removed from grid.
private Location[] removeBlocks()
```

Create a new `Location` array called `locations` with a length of 4. Loop through the `locations` array using a standard for-loop. Inside the loop, set `locations[i]` to `block[i].getLocation()`, and then tell `block[i]` to `removeSelfFromGrid()`. Then, outside the for-loop, return `locations`.

Run `TetradTest` to verify that you have completed this exercise before moving on.

Exercise 2.1 Anybody Home?

Complete the following private helper `areEmpty` method in your `Tetrad` class.

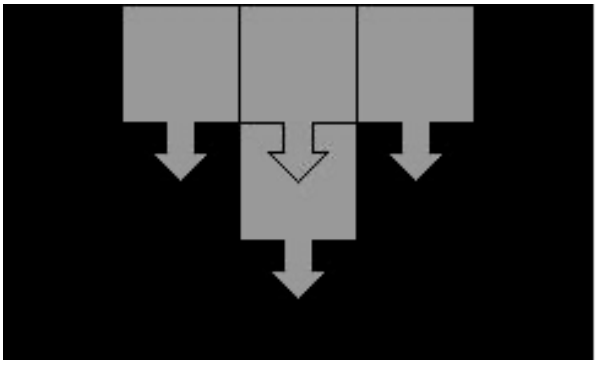
```
// postcondition: Returns true if each of locations is
//                valid (on the board) AND empty in
//                grid; false otherwise.
private boolean areEmpty(Grid grid, Location[] locations)
```

Loop though `locations`. Check each location to see if it is either not a valid location in the grid (`!grid.isValid(location)`), or it is not empty (`grid.get(location) != null`). If either of those is the case, return `false`. Then, outside of the loop, return `true` (all four locations are valid and empty in the grid).

Run `TetradTest` to verify that you have completed this exercise before moving on.

Exercise 2.2: Lost in Translation

We will now complete the `Tetrad` `translate` method, which will shift the tetrad over and down by the given amount, first making sure that the tetrad's potential new position is valid and empty. For example, suppose we have a T-shaped tetrad in the top middle of an empty grid, and we wish to move it down by one row, as shown below.



Clearly, we ought to allow this move. But notice that one of the locations we are attempting to move this tetrad into is already occupied by another block in the tetrad itself. So, it's not enough to make sure that the new locations are empty.

To make sure we handle this situation correctly, we will always translate a tetrad as follows:

1. Ask any block for its `grid`, and store it in a temporary variable.
2. Create a new array capable of holding 4 `Location` objects called `oldLocations`
3. Remove the blocks (but save the locations to `oldLocations`).
4. Create another array capable of holding 4 `Location` objects called `newLocations`.
5. Loop through `newLocations`, setting each element as follows.
   ```
   newLocations[i] = new Location(oldLocations[i].getRow() + deltaRow,
                                  oldLocations[i].getCol() + deltaCol);
   ```
6. Test `if` the new locations `areEmpty` using `newLocations`.
7. If the new locations are empty, call `addToLocations` using `newLocations` and return `true`. Otherwise, put the blocks back where they were by calling `addToLocations` using `oldLocations` and return `false`.

Go ahead and complete the `Tetrad` `translate` method, making use of the helper methods `addToLocations`, `removeBlocks`, and `areEmpty`. Use the steps outlined above as your guide.

```
// postcondition: Attempts to move this tetrad deltaRow
//                rows down and deltaCol columns to the
//                right, if those positions are valid
//                and empty; returns true if successful
//                and false otherwise.
public boolean translate(int deltaRow, int deltaCol)
```

Run `TetradTest` to verify that you have completed this exercise. Please submit a screenshot of the tester success message for Part - 2.