2048 Exercises

Warm-ups:

1. Change DrawBoard to use and not use "run without screen refresh" and notice the difference.

2. Make it so the board starts with four random cells filled in, two of them with 2 and two of them with 4.

3. Make it so there is more space between the cells on the board.
4. Make it so there is no space between the cells on the board.
5. Make it so the board is positioned in the lower-right of the graphics area.
6. Change it so the board is 5x5 instead of 4x4. Then go back to 4x4.

Finishing the game:

7. Write TryMove using the following logic:

    - set “did any” to 0
    - do a forever loop that keeps trying to make a move, and as long as it made a move it keeps going, but as soon as it didn't make a move, it stops – use the following steps:

        - set “did it” to 0
        - call TryMoveOnce to see if moves can be made in the given directions
        - if “did it” is 0, it means no move was made, so we're done and we can do the following:

            - if “did any” is 1, it means that some move was made, so we add a new cell in a random location, the new cell should have a 50/50 chance of being 2 or 4
            - draw the board
            - stop this script (we stop the forever loop since no more sliding or compressing could be done in the given direction)

        - otherwise (if “did it” is 1) set “did any” to 1 to flag that at least one move was made

8. Write TryMoveOnce, which uses a repeat loop nested inside another repeat loop to go across the columns and rows of the board. For each col/row combination it calls TryMoveOnceAtLocation.

9. Write TryMoveOnceAtLocation using the following logic:

    - get the value at the give board space
    - if the value is 0, the cell is empty so stop this script
    - otherwise, check if adding dx to c or adding dy to r would make a number that's off the board (less than 1 or more than 4); if so, stop this script
    - put the value in the “first value holder” variable, to save it
    - get the board value at the space c + dx, r + dy
    - if the value of that space is 0 (meaning the cell is empty), we move this cell into that cell by:

    - set the value of this cell (c, r) to 0 (empty)
    - set the value of that cell (c + dx, r + dy) to our saved “first value holder”
    - set “did it” to 1 to flag that we made the move
- otherwise, if the value of that space is not 0, check if its the same value as the value we stored in “first value holder”; if so, it means the two numbers are the same so we can do a compression:
    - set this value of this cell (c, d) to 0 (empty)
    - set the value of that cell (c + dx, r + dy) to double the value
    - set “did it” to 1 to flag that we made the move

Extra challenges:

10. Make it so the newly appearing cells are shown in a different color so the player can tell the new cells from the old cells.

11. Make it so it tells you when you've lost. That's when the board is full (no slides can be done) and also no more compressions can be done. One way to do this is to check if the board is full (how?), and if so:

    - make a backup copy of the board in a separate list variable
    - call TryMove for all four directions and see if you ever get “did any” set to 0
    - if not, no moves were possible and the game is over