Sokoban/Pengo Challenges Part 1

1. Playing with stamping

    a. Write a “forever” loop that uses “stamp” to stamp the “wall” costume from the “images” sprite in random places on the screen forever.
    b. Change your “forever” loop so that for each “stamp” it chooses a random costume from the “images” sprite.
    c. Change your “forever” loop so that it sets the sprite to a random size before each “stamp”.

    d. Remove the sizing code from part (c)

    Now, here’s an important exercise. Before, you were stamping the sprite just anywhere on the screen. Now let’s try to stamp it just on the actual “grid cells” of our game board. Remember the game board has its upper-left corner at

    (x = -180, y = 180)

    And remember that the board is 20 grid cells across and 20 grid cells down. So what you want to do here is first pick two numbers that you can put in variables called “xgrid” and “ygrid”. Each of these numbers should be between 0 and 19. (Counting from 0 to 19 is better than counting from 1 to 20. Can you explain why?) Then, to position the “images” sprite before stamping it, using the following:

    - For the x coordinate, take -180 and add 18 times “xgrid”
    - For the y coordinate, take 180 and subtract 18 times “ygrid”

    Your job is to turn those two statements into code. And also, try explaining those two lines to a parent or friend. Those lines are very important and they are what lets you turn the game’s “grid” coordinates into x, y coordinates for Scratch to draw the sprites.

    When you get this exercise working, it should look something like what you did in part (c), but instead of drawing the sprite randomly anywhere, it only draws the sprites on the board, and it draws each one properly inside a grid cell, so the image don’t overlap. (It will look like some of the images are “changing” into other images, but they all fit neatly in the game’s grid.)

    (Side note: If you look at the costumes for the “images” sprite, and you use the “set costume center” button in the top-right of the window, you will see that all the sprites are “centered” on their top-left corners. Can you explain how this helps when drawing the board using our “grid” coordinates?)

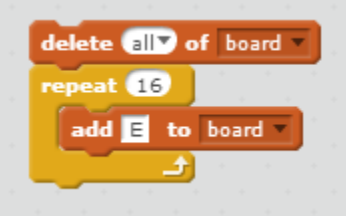
2. Playing with lists

   To actually make the game work, we need a way to keep track of what's where on the board. To do that, we are going to use a list.

   a. Under the "Data" section, make a list called "board" (for all sprites).

   b. In the real game, we are going to use a list of size 400 to hold a board of size 20 x 20. But to keep things simpler for now, we are going to use a list of size 16 to hold a board of size 4 x 4.

      Use the following code to empty the "board" list, and then fill it with 16 E's.



   c. We are going to use the following letters in the list to represent different things on the board:

      - "E" represents an empty space on the board.
      - "W" represents a wall.
      - "B" represents a box.
      - "P" represents the player.
      - "T" represents a target location.

      Modify the "repeat" loop so that instead of filling the board with just "E", it fills it with a random letter from the list above.

3. Combing stamping with lists to draw the game board

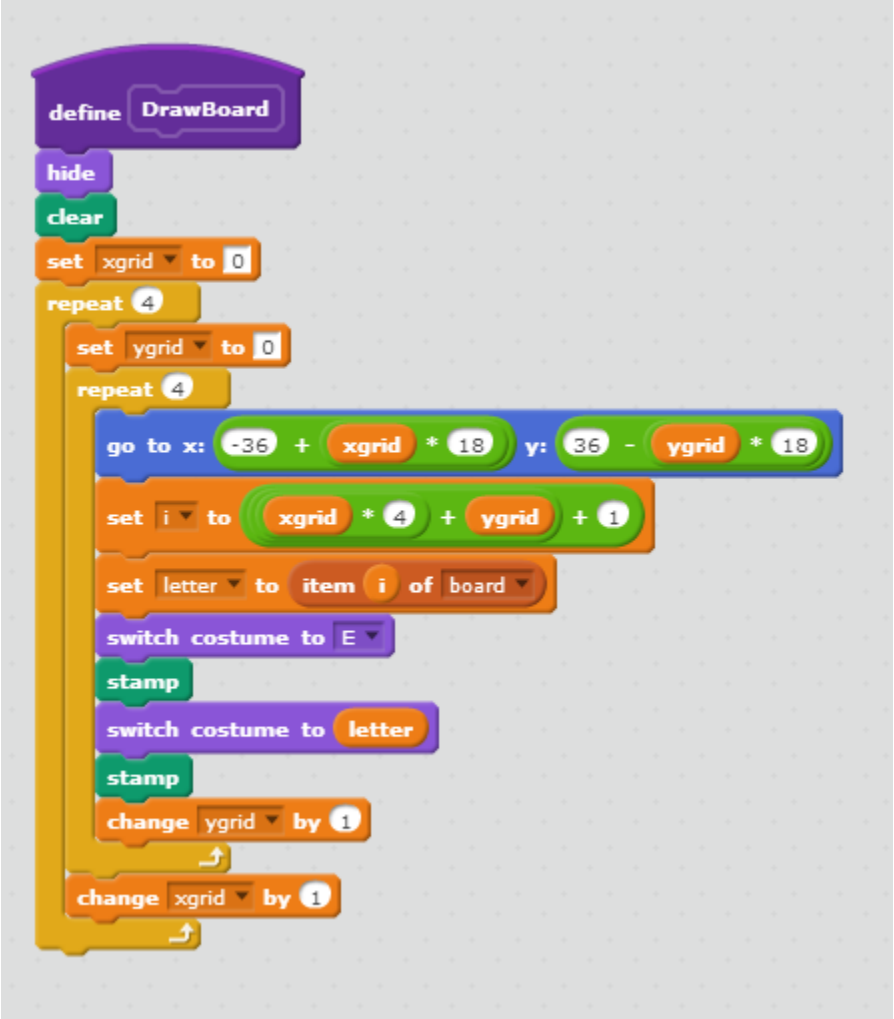
   a. We are now ready to combine the grid stamping we learned in exercise 1 with the list we made in exercise 2. There are two important things to know:

      - Since we are using just a 4 x 4 board for now, the top-left corner of the board will be at x = -36, y = 36. Otherwise the drawing is basically the same, using our same two rules:

- For the x coordinate, take -36 and add 18 times “xgrid”
- For the y coordinate, take 36 and subtract 18 times “ygrid”

- Remember that our list is just a “flat” list from 1 to 16. We want those 16 list items to represent the “cells” in our 4 x 4 board. How should we do that? Well, let’s do it this way: Let’s say that the first four list items represent the first column of the 4 x 4 grid. The next four list items represent the second column. The next four list items represent the third column. And the last four list items represent the fourth column.

  Here is what that means. It means that if we want to draw the board, we can write a loop like this:



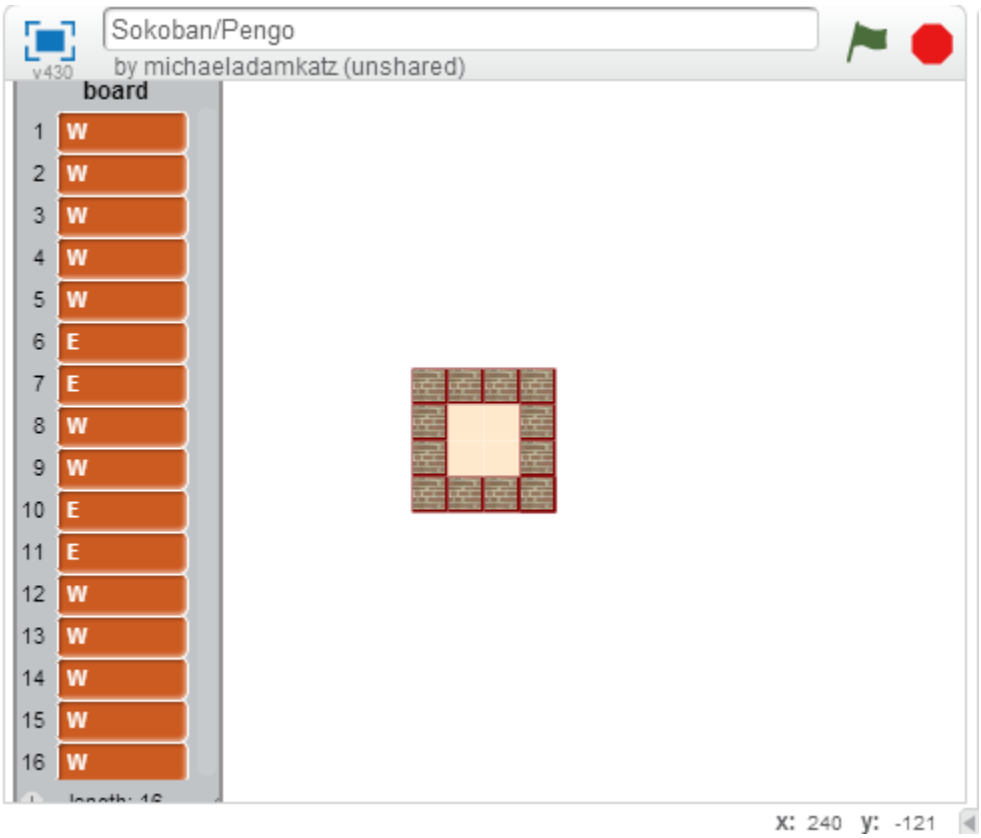
  Create that code now.

That code is not simple. But on the other hand, it's something that you can understand if you take it line by line. For now, don't worry if you don't understand it all.

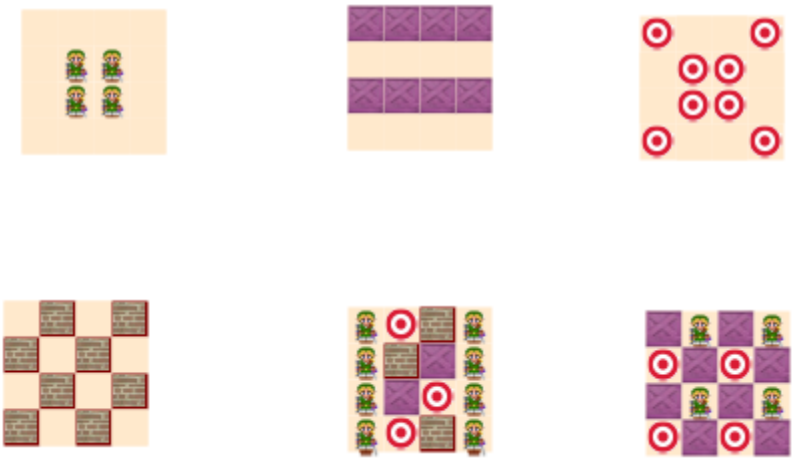
b. What I do want you to understand is what was said above about how the items in the list relate to what gets drawn on the board. I'll repeat that here:

- The first four list items represent the first column of the 4 x 4 grid. The next four list items represent the second column. The next four list items represent the third column. And the last four list items represent the fourth column.

To understand that better, try reproducing each of the boards below. To do this, click right on the list elements and enter values for them. Then click the DrawBoard function to "stamp" those list elements out on the board. Here is the first one, with its answer:



And here are some others for you to try:



4. Random boards

Previously we drew random grids on our 20 x 20 board. Now let's try the same idea with our 4 x 4 board, but this time instead of just stamping directly on the screen, we'll use our list.

Write a function to fill the list with 16 random items, and then call the DrawBoard function to draw those items. Do that repeatedly to draw on random board after the other.

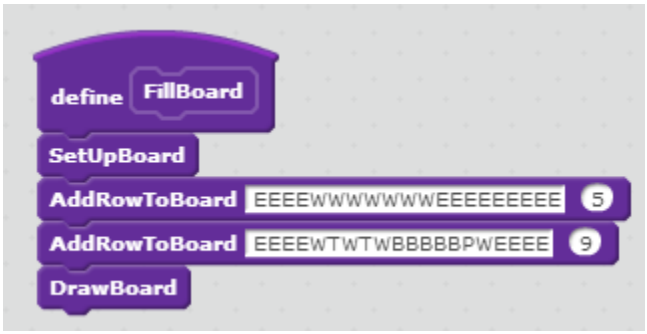
Trick: If you right-click on the DrawBoard function you can edit it and under "options" you can choose "Run without screen refresh". This will make DrawBoard go much faster and you can see your random boards very quickly.

Sokoban/Pengo Challenges Part 2

5. Filling the board list by hand or randomly is okay for an exercise, but to really fill a large board list, we are going to need an easier way.

   First, let's imagine a board. Here is a board from somebody else's Sokoban game. Let's try to make this one.

   Remember our total board size is going to be 20x20. If you count the spaces in the board about, you'll see it's 12x9. So that fits fine into our space.



   Notice that since the board is 12 wide, but our total board space is 20 wide, we have 8 spaces extra, so we put four extra E's on the left, and four extra E's on the right. Also, since the board is 9 high but out total board space is 20 high, we start on row 5 so it will come out more or less centered on the screen vertically.

   Your challenge: Add 7 more lines to FillBoard to create the board pictured at the start of this exercise.

6. Playing the game: move in all directions

   Write the other three key handlers – Down, Left, Right

7. Playing the game: detecting win

   Can you figure out how to write a function to know when the player has won the game? Ask for help if you need it!

8. Make more levels.

   Make a new function called FillBoard2 that's like FillBoard, but makes a different level.