```
 2  * simonDisplay.h
 7 #ifndef SIMONDISPLAY_H_
 8 #define SIMONDISPLAY_H_
 9
10 #include <stdbool.h>
11 #include <stdint.h>
12
13 // Width, height of the simon "buttons"
14 #define SIMON_DISPLAY_BUTTON_WIDTH 60
15 #define SIMON_DISPLAY_BUTTON_HEIGHT 60
16
17 // WIdth, height of the simon "squares.
18 // Note that the video shows the squares as larger but you
19 // can use this smaller value to make the game easier to implement speed-wise.
20 #define SIMON_DISPLAY_SQUARE_WIDTH  120
21 #define SIMON_DISPLAY_SQUARE_HEIGHT 120
22
23 // Given coordinates from the touch pad, computes the region number.
24
25 // The entire touch-screen is divided into 4 rectangular regions, numbered 0 - 3.
26 // Each region will be drawn with a different color. Colored buttons remind
27 // the user which square is associated with each color. When you press
28 // a region, computeRegionNumber returns the region number that is used
29 // by the other routines.
30 /*
31 |----------|----------|
32 |          |          |
33 |    0     |     1    |
34 |  (RED)   | (YELLOW) |
35 ------------------------
36 |          |          |
37 |     2    |    3     |
38 |  (BLUE)  |  (GREEN) |
39 ------------------------
40 */
41
42 // These are the definitions for the regions.
43 #define SIMON_DISPLAY_REGION_0 0
44 #define SIMON_DISPLAY_REGION_1 1
45 #define SIMON_DISPLAY_REGION_2 2
46 #define SIMON_DISPLAY_REGION_3 3
47
48 int8_t simonDisplay_computeRegionNumber(int16_t x, int16_t y);
49
50 // Draws a colored "button" that the user can touch.
51 // The colored button is centered in the region but does not fill the region.
52 void simonDisplay_drawButton(uint8_t regionNumber);
53
54 // Convenience function that draws all of the buttons.
55 void simonDisplay_drawAllButtons();
56
57 // Convenience function that erases all of the buttons.
58 void simonDisplay_eraseAllButtons();
59
60 // Draws a bigger square that completely fills the region.
61 // If the erase argument is true, it draws the square as black background to "erase"
   it.
62 void simonDisplay_drawSquare(uint8_t regionNo, bool erase);
```

```
63
64 // Runs a brief demonstration of how buttons can be pressed and squares lit up to
   implement the user
65 // interface of the Simon game. The routine will continue to run until the touchCount
   has been reached, e.g.,
66 // the user has touched the pad touchCount times.
67
68 // I used a busy-wait delay (utils_msDelay) that uses a for-loop and just blocks until
   the time has passed.
69 // When you implement the game, you CANNOT use this function as we discussed in class.
   Implement the delay
70 // using the non-blocking state-machine approach discussed in class.
71 void simonDisplay_runTest(uint16_t touchCount);
72
73 #endif /* SIMONDISPLAY_H_ */
74
```