Light Wars

Let's make a game with the LED shift register, buttons and LCD screen. The game will be a battle between 2 players, where you need to press the button fast enough when it arrives at your end.

1. Make Player 2 exist!

Goals
- ❏ Define player 2 and give it the correct pin
- ❏ Define pressed_p2
- ❏ Copy the pinMode line of code for player2 to set it as an input
- ❏ Copy the digitalWrite line of code to initialise the button to HIGH

1. **Near the top of the file** you'll find this code, which sets the pin number for player1:

   ```
   int player1 = 14;
   ```

   Add a similar declaration for player 2, **but change the number** to the correct pin number for the right hand side button. (What's the correct pin number? The answer is on your Arduino board.)

2. Next, find this line of code and **add a similar one for player 2:**

   ```
   boolean pressed_p1;
   ```

3. Find this line of code in **`setup()`** and **make a similar one for player 2**:

   ```
   pinMode(player2, INPUT);
   ```

4. Find this line of code in **`setup()`** and **make a similar one for player 2**:

   ```
   digitalWrite(player2, HIGH);
   ```

2. Ready, Set, Go! Game Over!

Goals
- ❏ Get the LCD screen to print "Ready", "Set", Go"
- ❏ Get the LCD screen to print "Game Over!"
- ❏ Make the game wait 3 seconds before starting again

1. **Go to the loop()** part of the code
2. **Clear** the LCD screen
3. **Move the cursor** to the top left corner
4. **Print "Ready"** to the screen
5. **Add a delay** of 700 milliseconds
6. **Repeat the above steps to add "Set", "Go", and "Game Over"**
7. **Set the delay after the "Game Over"** to be 3000 milliseconds.

Nice work! You've got the main outline of a game.

3. Moving Lights

Goals
- ❏ Make a variable that stores how fast the lights should move
- ❏ Make the lights go from right to left
- ❏ Then make the lights go back from left to right

Remember ***8 LEDs, 8 Times The Fun*** from earlier today? Like in those programs this game will make use of the eight LED lights below the board's green display.

1. **Create a variable called `gameSpeed`**, **set it to 500**. `gameSpeed` represents how many milliseconds the game will pause at each step of the moving lights. Put it in the Variable Declaration Section.

The next pieces of code represent the main body of your program, so it should be placed between the "Go" and "Game Over" prompts in `loop()`.

2. **Get the lights to move from right to left**. Under the comment `// PART 3.2: LIGHTS GO FROM RIGHT TO LEFT`, **use this code eight times** to move the light from right to left. You'll have to edit it slightly to make it work...

   ```
   displayBinary(<SOME NUMBER HERE>);
   delay(gameSpeed);
   ```

3. Under the comment `// PART 3.3: LIGHTS GO FROM LEFT TO RIGHT`, **make the lights move from left to right** by doing the same thing in reverse.

4. Timer Time

Goals
- ❏ Create a variable called timeLimit to store the reaction time allowed for pressing the button
- ❏ Add a while loop that delays the code by 1 millisecond every time it runs.
- ❏ Make the loop run the number of times set in timeLimit

Each player should have a brief moment when the light hovers at their end, and they can press their button to "capture" it. **This needs a timer.**

Why can't we just use `delay()` to pause the light at the ends? The problem with delay is that **it freezes the code**. While in a delay, the program can't listen for the players pressing their buttons. Using a timer solves this problem.

1. Look all the way up to the top of the code, in the variable declaration section. **Create a variable called timeLimit and set it to 3000.**

2. Also in the variable declaration section at the top, **declare an integer variable called timer.**

3. Look back down to the first block of light-moving code; the block that moves the light from right to left. After the eighth call to `displayBinary()`, **remove the delay.** It's time to replace it with the new timer system.

4. Replace the `delay()` with code that **sets `timer` to equal 0.**

5. Below that, **make a while loop. Check if `timer` is less than the time limit** of `gameSpeed` in the condition section of the while loop

6. Inside the while loop, **add a `delay` for 1 millisecond.**

7. Inside the while loop, **increment `timer` by 1.**

8. **Run your code!** Check if this happens:
    a. "Ready", "Set", "Go" displays
    b. The light go from right to left
    c. The light pauses for 3 seconds
    d. The light goes from left to right
    e. "Game over" displays

5. Bring on the Button

Goals
- ❑ Update the variable pressed_p1 if player1's button is pressed while we're waiting
- ❑ Send the light back straight away if player1's button is pressed.

If we press the button we don't want to wait to send the light back the other way!

1. Above the while loop **set pressed_p1 to false**

2. Inside your while loop, **create an if statement that checks if player1's button has been pressed**. *Remember how to do this by looking at Button Bonanza! From the first workbook.*
   If the button is pushed we want to **update pressed_p1 to true**. Do this **inside the if statement**

3. Change the condition of the while loop so it checks if the timer is less than the time limit **AND checks to see if the button still hasn't been pressed** (pressed_p1 should still be false).

4. **Run your code!** It should do these things when the light reaches the left side:
   a. Pressing the button should return the light immediately.
   b. Not pressing the button should have the light wait three seconds, like before.

6. Double Trouble Buttons

Goals
- ❏ Update the variable pressed_p2 if player2's button is pressed while we're waiting
- ❏ Make your player2 button end the game immediately if pressed

It's now time to make Player 2's button work too.

1. Just like before but now for the left-to-right light sequence, **remove the `delay()` after the last call to `displayBinary()`.**

2. **Replace the `delay()` with a copy of the button and timer code you just wrote for player 1.** Don't forget to update all the references of p1 to p2.

3. **Run your code!** You already checked that it worked for right-to-left, but you should now check that it also works for left-to-right, when the light is hovering at the right side:
    a. Does "Game Over" happen straight away if you press the button on the right?
    b. Does it still wait 3 seconds if you don't press the button?

7. Again Again Again

Goals
- ❏ Add an infinite loop that makes our game go for ever

Right now our game only goes back and forwards once, that's not very fun! Let's make our game code run over and over again.

1. **Create an infinite loop** below your "Ready" "Set" "Go" code. *Remember the condition in an infinite while loop is always true.*

2. **Make sure the curly braces for your infinite while loop surround all of you game code.** This is everything after "Ready" "Set" "Go", but before "Game Over".

3. **Run your code!** Check it does this:
    a. The light goes left and right, over and over again.
    b. If you push the button it sends the light back faster
    c. If you don't push the button it waits 3 seconds before the light changes direction
    d. *The game goes on forever!* There is no way to get to "Game Over".

TROUBLESHOOTING TIP: Is the light waiting for Player 1 (on the left), but not Player 2 (on the right)? You might need a line to reset `timer` before Player 2's turn. Can you explain how this strange behaviour is happening?

8. Time to lose

Goals
- ❏ Make a variable that stores the winner
- ❏ When player 1 misses the button player 2 wins
- ❏ When player 2 misses the button player 1 wins
- ❏ When someone wins the game finishes
- ❏ The game prints the winner at the end of the game

At the moment the game never finishes, there is no way to lose and therefore no way for us to win!

1. **Create an integer variable called winner** in the variable declaration section at the top of the page. This will store the winner of the game. **Set it to 0**, since no player has won yet.

2. After the loop that waits for player1 to press the button (the first `while` loop), **add an `if` statement**. We want to see if the timer ran out, **so the `if` statement should check if `timer` is equal to `timeLimit`.**

3. **If the player did run out of time set the winner** to be the player who did not lose.

4. If the player did run out of time, add a **`break`** statement to break us out of the infinite loop.

5. **Run your code!** Check if it does this:
   a. When the light moves from right to left, **push the button in time**. Does the light still go back the other way?
   b. When the light moves from right to left, **DON'T push the button**. Does game over show?

6. You've got it working for the left side; now for the right side. **Repeat steps 2-5 for player2's button**.

7. Go to the "Game over" code. **Use an if/else if statement to check if player 1 or 2 won**. If player1 won, print "Player 1 wins!". If player2 won, print "Player 2 wins!".

8. **Add a long delay** to display the winner for 3000 milliseconds.

9. **Run your code!** Play the game, see what happens if you forget to press the buttons at each end. **Does it print the correct winner?**

Bonuses - Add to your game!

Automate the light-shifting code with while loops

1. Create a variable to store where the light is at.
2. At the start of the game set the light value to 1.
3. Add a while loop that displays the the light, has a delay, and increments the position of the light using the pattern we found in Eight LEDs Eight Times the fun.
4. Display the final light separately after the while loop (because it needs the different delay we created in Part 4)
5. Repeat this for sending the lights back the other direction

Add a scoreboard

1. Create two variables to store how many times each player has won. Initialize each player's score to zero.
2. When either player wins, add one to their score.
3. At the end of the game, print the scores to the LCD display.
4. *Double bonus:* Add more displays throughout your code to show the score throughout the whole game!

First To Five

1. Create 2 variables, one that stores how many games won by player 1, one that stores the number of games won by player 2. Set them both to 0 to start with
2. Every time someone wins increment their number of games one, only for the winning player
3. After each game,check whether either player has won 5 times. If so, display the final scores before the loop section starts over again

Too Early

1. Can you make it so a player loses if they press the button to early (before it gets to the end light)? It's a tricky one, try and figure it out!