```
// Eugene Ngo
// 12/2/2022
// EE 271
// Lab 5

// centerLight module takes in clk, reset, L, R, NL, NR, restart as inputs and they are all 1-Bit and returns
// lightOn which is also 1-Bit. This module determines the next state based on the inputs and moves the light
// to the left if key 3 is pressed and moves the light to the right if key 0 is pressed. When the game is reset
// or restarted then this module recenters the light for the next game.

module centerLight (lightOn, clk, reset, L, R, NL, NR, restart);
   output logic lightOn;
   input logic clk, reset;
   input logic L, R, NL, NR, restart;

   enum {on, off} ps, ns;

   always_comb begin
      case(ps)
         on:    if(~R & L | R & ~L) ns = off;
                else ns = on;

         off:   if(NR & L & ~R | NL & R & ~L) ns = on;
                else ns = off;

      endcase

   end
   always_comb begin
      case(ps)
         on: lightOn = 1;

         off: lightOn = 0;
      endcase
   end

   always_ff @(posedge clk) begin
      if(reset | restart)
         ps <= on;
      else
         ps <= ns;
   end

endmodule

// centerLight_testbench tests all expected, unexpected, and edgecase behaviors
module centerLight_testbench();
   logic L, R, NL, NR;
   logic lightOn;
   logic clk, reset;

   centerLight dut (.lightOn, .clk, .reset, .L, .R, .NL, .NR);

   //Set up the clock
   parameter CLOCK_PERIOD = 100;
   initial begin
      clk <= 0;
      forever #(CLOCK_PERIOD / 2)
      clk <= ~clk;
   end

   //Set up the inputs to the design. Each line is a clock cycle
   initial begin
                                          @(posedge clk)
      reset <= 1;                         @(posedge clk)
                                          @(posedge clk)
      reset <= 0;                         @(posedge clk)
                                          @(posedge clk)
      L <= 1; R <= 0; NL <= 0; NR <= 1;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                      NL <= 1; NR <= 0;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      L <= 0; R <= 1;                     @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                      NL <= 0; NR <= 1;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      L <= 1;         NL <= 1; NR <= 0;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      reset <= 1;                         @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      L <= 0; R <= 1; NL <= 1; NR <= 0;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      L <= 1; R <= 0; NL <= 0; NR <= 0;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      L <= 0; R <= 1; NL <= 0; NR <= 0;   @(posedge clk)
                                          @(posedge clk)
                                          @(posedge clk)
      $stop;
   end
endmodule
```