```
/*******************************************************************************
 * lab_2
 * Created by Matt Mason,
 * CWID 11800439
*******************************************************************************/

// switch pin & debounce threshold
static constexpr int SW = 53;
static constexpr int DEBOUNCE_THRESH = 100; // ~100 ms

// LED pins
static constexpr int LED = 13;
static constexpr int EXT_LED = 52;

// seven-segment display pins & character mappings
#define SEVEN_SEG PORTC
#define SEVEN_SEG_DDR DDRC
static constexpr byte CHARACTERS[] = {
    B00000011, // 0
    B10011111, // 1
    B00100101, // 2
    B00001101, // 3
    B10011001, // 4
    B01001001, // 5
    B01000001, // 6
    B00011111, // 7
    B00000001, // 8
    B00001001, // 9
    B00010000, // A.
    B11000000, // b.
    B01100010, // C.
    B10000100, // d.
    B01100000, // E.
    B01110000  // F.
};

/*******************************************************************************
 * Function:        readSwitch
 * Parameters:      int count — the current count of switch state changes
 * Return value:    int — an updated count of switch state changes
 * Purpose:         Read the switch input, debounce it, and calculate a new
 *                  value for the count variable accordingly
*******************************************************************************/
int readSwitch(const int count)
{
    // initialize debounce variables
    static bool switchState = digitalRead(SW);
    static int readCount = 0;

    // debounce logic
    if (switchState != digitalRead(SW))
    {
        if (readCount == DEBOUNCE_THRESH) // switch has been flipped
        {
            // reset read counter
            readCount = 0;
            // toggle saved switch state
            switchState = !switchState;
            // calculate new value for switch state-change counter
            return (count + 1) % 16;
        }
        else
            // increment read counter on successive reads != saved switch state
            readCount++;
    }
    else
        // reset read counter when the read value == saved switch state
        readCount = 0;

    // no change to switch state-change counter
    return count;
}
```

```
73
74  void setup()
75  {
76      // configure switch pin as input w/ internal pullup
77      pinMode(SW, INPUT_PULLUP);
78
79      // configure LED pins as outputs
80      pinMode(LED, OUTPUT);
81      pinMode(EXT_LED, OUTPUT);
82
83      // configure all seven-seg. pins as outputs
84      SEVEN_SEG_DDR = 0xFF;
85
86      // open serial connection
87      Serial.begin(9600);
88      Serial.println("lab_2 by Matt Mason");
89
90      // print initial switch state-change count to serial terminal
91      Serial.println("count = (decimal) 0   (hex) 0");
92  }
93
94  void loop()
95  {
96      // initialize switch state-change counter
97      static int count = 0;
98      // initialize LED blinking state to initial switch value
99      static bool blinking = digitalRead(SW);
100     // get current time
101     unsigned long currentTime = millis();
102     // initialize LED toggle timepoint to 1 second from now
103     static unsigned long toggleTime = currentTime + 1000;
104
105     // read switch and get updated count
106     int newCount = readSwitch(count);
107
108     // if switch state-change count has changed
109     if (newCount != count)
110     {
111         // update switch state-change counter
112         count = newCount;
113         // print new value to serial terminal
114         Serial.print("count = (decimal) ");
115         Serial.print(count);
116         Serial.print((count < 10) ? "   (hex) " : "  (hex) ");
117         Serial.println(count, HEX);
118         // toggle LED blinking state
119         blinking = !blinking;
120         // set LED toggle timepoint to 1 second from now
121         toggleTime = currentTime + 1000;
122     }
123
124     // when switch is on, blink LEDs according to specified pattern
125     if (blinking)
126     {
127         // if the toggle cycle has finished, update LED toggle timepoint
128         if (currentTime > (toggleTime + 1000))
129             toggleTime = currentTime + 1000;
130
131         // turn on the appropriate LED based on the toggle timepoint
132         bool toggleState = (currentTime < toggleTime);
133         digitalWrite(LED, toggleState);
134         digitalWrite(EXT_LED, !toggleState);
135     }
136     else
137     {
138         // turn LEDs off when switch is off
139         digitalWrite(LED, 0);
140         digitalWrite(EXT_LED, 0);
141     }
142
143     // update seven-segment display
144     SEVEN_SEG = CHARACTERS[count];
145
```

```
146       // delay 1 ms to regulate loop speed
147       delay(1);
148   }
```