# Motion detection

1. Implement a function that returns TRUE or FALSE for motion detected
2. Implement a Buzzer that makes sound when motion is detected
3. Add 3 LEDs and booleans (also for debugging)
   1. System armed ON/OFF (TRUE/FALSE)
   2. Timer before alarm ON/OFF
   3. Alarm/buzzer ON/OFF (TRUE/FALSE)
4. Add 2 buttons
   1. Simulating “correct password”
   2. Arming the system
5. Implement logic
   1. Setup
      1. System armed = **FALSE**
      2. Buttons are inputs
      3. LEDs as output
      4. All LEDs are off
   2. Loop
      1. Check motion
      2. If system armed = **FALSE:**
         1. Buzzer = **FALSE**
         2. Timer = **FALSE**
      3. If system armed = **FALSE** AND arm\_button = **TRUE:**
         1. Armed = **TRUE**
      4. If system armed = **TRUE** AND motion = **TRUE** AND timer = **FALSE**
         1. Start timer
         2. Timer LED = **TRUE**
      5. While timer = **TRUE:**
         1. If password\_button = **TRUE**:
            1. System armed = **FALSE**
            2. Timer = **FALSE**
            3. Break
         2. If timer > 10 seconds:
            1. Buzzer = **TRUE**

# Keypad module

1. Figure out what library to use
2. Test the following individual buttons:
   1. Buttons 0-9 prints its respective number
   2. Submit button
   3. Backspace button
3. Buttons 0-9 and other buttons in combination are used to print a 4 digit number
4. Add one LED for showing if the system is armed
5. Implement logic
   1. Setup
      1. Buttons as inputs
      2. LED as output
      3. System armed = **FALSE**
      4. Password string
   2. Loop
      1. Input = read\_keypad()
      2. If password == input:
         1. Armed != armed (toggle armed)
         2. armed\_LED = armed
   3. Int read\_keypad()
      1. Char input[4] = ””
      2. While strlen(input) < 4:
         1. If backspace is pressed:
            1. Pop the last element from input[]
         2. If 0-9 is pressed:
            1. Print to console, debugging
            2. Append to input[]
      3. Return convert\_to\_int(input)

# LCD