Demonstration: (NOTE: these change based on the lab - enter the demo tasks specified in each lab)

Setting and running the digital clock, green LED	________
Setting, enabling, disabling alarm	________
Keeping the board idle to clear the display and making a rotation to keep the running clock	________

Requirements:

Display a 24-hour clock on the LCD display of the booster pack	________
Display the time as TIME: HH:MM:SS, alarm as ALARM: HH:MM:SS (E/D), both center aligned	________
Green LED should quick blink every second change	________
Change time when command SET TIME xx:yy:zz is input to the serial monitor, do not accept invalid times but signify which unit was wrong (i.e. invalid hours, etc)	________
SET ALARM xx:yy:zz sets the alarm in the same manner	________
Enable or disable (toggle) the alarm with S1 (J4.33)	________
When the alarm goes off, generate a beeping sound that can be stopped with S2 (J4.32)	________
When a command is executed from the serial monitor, generate a beep	________
The end of a command is where the enter key is pressed	________
When a command is accepted, print “Executed” on the monitor	________
When a command is not accepted, print “Invalid command”	________
Keep all commands and response messages on new lines	________
Clear the LCD screen when the board is stationary for 5 seconds	________
Display the clock again after the board rotates about the x-axis (45°)	________
Configure the push buttons as interrupt-based inputs and do not use delays	________

Learning Objectives:
The general learning objectives of this lab were to apply a lot of the knowledge we have learned throughout the semester to a practical application, and to also get a little bit of insight into programming the LCD screen of the Educational MKII BoosterPack.

General Steps:
The general steps needed to complete this lab were to create a clock and display it to the LCD, create an alarm, implement inputs from the serial monitor, implement interrupt input buttons, add the blinking light, and add an idle timer.

Detailed Steps:
Some detailed steps to complete this lab were . . . .

1. Look through example code provided with the board to learn about implementing the LCD display.
2. Create a clock using three variables for hours, minutes, and seconds.
3. Use what you learned from the LCD display program to print the clock to the LCD display every second

```
myScreen.gText(23, 48, "TIME: " + tHrStr + ":" + tMinStr + ":" + tSecStr, whiteColour);
```

4. Create another display for the alarm that does not change for now, and displays an alarm mode next to it.
5. Add an implementation to accept inputs from the serial monitor, testing that the phrase “SET _____ XX:YY:ZZ” and that it is a valid input
6. Implement interrupt buttons so that S1 (J4.33) disables/enables the alarm, and S2(J4.32) stops the alarm while it is beeping.
7. Add the blinking green light ever time a second is incremented.
8. Create another variable that checks for how long the board has been idle (i.e. the time since it has rotaed 45+ degrees), and add a clear function when it goes idle.

CODE:

```
void loop()
{
 if(millis() - lastSec > 1000) {
  tSec++;
  lastSec = millis();
  blinkGreen();
 }
 if(tSec == 60) {
  tMin++;
  tSec = 0;
 }
 if(tMin == 60) {
  tHr++;
  tMin = 0;
  }
  if(tHr == 24) {
    tHr = 0;
    tMin = 0;
    tSec = 0;
  }

  if(Serial.available()) {
    readCommand();
  }

  if(alarming) {
    tone(buzzerPin, 30);
    tone(buzzerPin, 500, 50);
    idleTime = millis();
  }

  currPos = (int) analogRead(xpin) - 2048;

  if((currPos- pos) > 105 || (pos - currPos) > 105) {
    idle = false;
    pos = currPos;
    idleTime = millis();
  }
  else if(millis() - idleTime > 5000) {
    idle = true;
  }

  if(aHr == tHr && aMin == tMin && aSec == tSec && aEnabled) {
    alarming = true;
    idle = false;
  }

  if(!idle || alarming) {
    timePrint(tSec, tMin, tHr);
    alarmPrint(aSec, aMin, aHr);
    blank = false;
  }
  else if(!blank) {
    myScreen.clear(darkGrayColour);
    blank = true;
  }

}
```

(Include selected code snippets/procedures that were written by you that are important. Answer all the questions asked in the lab document.)
(DO NOT INCLUDE PRINTOUTS OF REDILY AVAILABLE CODE FROM THE BOARD DISK)
(DO NOT PRINT ALL CODE FROM YOUR EFFORTS – IF WE ASK FOR YOUR CODE, YOU WILL BE ASKED TO SEND IT VIA EMAIL)

Observations:
Some important observations while completing/testing this lab were that the functions for the LCD screen are implicitly defined, so we only have to use functions like "myscreen.gtext" or "myscreen.clear" to accomplish the advanced tasks for this lab, proving further that Energia is a very high level coding environment and is not necessarily true to what we will be using when working in the field. Also, while the system was idle, it seemed to only loop once every second, which was odd. I'm thinking it could possibly be because the loop got stuck at the first if statement (in code) waiting for it to be true to move on with the program.

Summary:
In this lab we learned how to implement an LCD screen using implicit functions with the board, and how to implement several things we have learned in the past into a more practical and realistic application.