```
#include  <LiquidCrystal.h>

#define LED_0            3
#define PUSH_BUTTON_0    8
#define LED_1            4
#define PUSH_BUTTON_1    9
#define LED_2            5
#define PUSH_BUTTON_2    10
#define LED_3            6
#define PUSH_BUTTON_3    11
#define LED_4            7
#define PUSH_BUTTON_4    12
LiquidCrystal lcd (18,17,16,15,14,19);
int PushButtons[5] = {PUSH_BUTTON_0, PUSH_BUTTON_1, PUSH_BUTTON_2,
PUSH_BUTTON_3, PUSH_BUTTON_4};
int LED[5] = {LED_0, LED_1, LED_2, LED_3, LED_4};
void setup() {
    // put your setup code here, to run once:
  lcd.begin(20,4);

  for(int i =3; i<8; i++)
  {
    pinMode(i,OUTPUT);
  }
  for(int i = 8; i<13; i++)
  {
      pinMode(i,INPUT_PULLUP);
  }
  lcd.autoscroll(); //This feature is used when seperating the sentence and
the number
}

void loop() {
  // put your main code here, to run repeatedly:
  int LoopCount = 0;

  for(LoopCount = 0; LoopCount <5; LoopCount++)
  {
       if(digitalRead(PushButtons[LoopCount]) == LOW)
      {
           digitalWrite(LED[LoopCount], HIGH);
          lcd.print("The LED ");
          lcd.print(LoopCount);
          lcd.print(" is on.");
```

```
          delay(5000);
          lcd.clear();
           digitalWrite(LED[LoopCount], LOW);
      }
  }
}



/*Start of program 1*
  #define LED_0       3
  void setup() {
  // put your setup code here, to run once:
  pinMode(LED_0, OUTPUT);
             }

  void loop() {

  digitalWrite(LED_0, HIGH); //Write the pin to high
  delay(500); //Wait for 500ms
  digitalWrite(LED_0, LOW); //Write the pin to low
  delay(500);//Wait for 500ms
  // put your main code here, to run repeatedly

  }
  End of program 1*/



/*Start of program 2
  #include<LiquidCrystal.h>  //Include LCD library to make use of LCD
functions
  LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to
LCD
  void setup() {
  // put your setup code here, to run once:
  lcd.begin(20, 4); //Initialize LCD
   lcd.print("Varad Vinayak");  //Write once to the LCD
  }

  void loop() {

  //No code needed here
  }
  End of program 2*/
```

```
/*Start of program 3
  #include<LiquidCrystal.h>  //Include LCD library to make use of LCD
functions
  LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to
LCD
  byte Count = 0;
  void setup() {
  // put your setup code here, to run once:
  lcd.begin(20, 4); //Initialize LCD
   lcd.print("Varad Vinayak");  //Write once to the LCD
  }

  void loop() {
  lcd.setCursor(0,1); // Set cursor to second line of the LCD
  lcd.print("   ");  //Clear the second line. This is done to erase all
previous data
  lcd.setCursor(0,1); //Set the cursor back to the start of the second line
  lcd.print(Count); //Print the count
  Count = Count + 20; //Increment the count
  delay(500);  // Delay for 500ms
  }
  /*
     The program prints the updated variable
     value every 200 ms. The variable is incremented
     by 20 each time. Since the variable is of type byte,
     it frequently rolls over after 255.
*/
/*End of Program 3*/



/*Start of program 4
  #include<LiquidCrystal.h>  //Include LCD library to make use of LCD
functions
  LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to
LCD
  byte Count = 0;
  void setup() {
  // put your setup code here, to run once:
  lcd.begin(20, 4); //Initialize LCD
   lcd.print("Varad Vinayak");  //Write once to the LCD
   lcd.setCursor(0,2); //Set cursor to row 2 to write the millis value
   lcd.print(millis());  //Print the millis value
```

```
  }

  void loop() {
  lcd.setCursor(0,1); // Set cursor to second line of the LCD
  lcd.print("    ");  //Clear the second line. This is done to erase all
previous data
  lcd.setCursor(0,1); //Set the cursor back to the start of the second line
  lcd.print(Count); //Print the count
  Count = Count + 20; //Increment the count
  delay(500);  // Delay for 500ms
  lcd.setCursor(8,2); //Set cursor to column 8
  lcd.print(millis());  //Print the millis value
  }
  /*
     The program is an extension of the previous program.
     By observing the output from Millis, we see that the delay of 500 given
     by the standard delay function "delay" is incorrect. A drift of several
ms
     is observed. These addup overtime and the error becomes quite large
after sometime.
*/
/*End of Program 4*/



/*Start of program 5
  #include<LiquidCrystal.h>  //Include LCD library to make use of LCD
functions
  LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to
LCD
  byte Count = 0;
  int PreviousTime = 0;
  void setup() {
  // put your setup code here, to run once:
  lcd.begin(20, 4); //Initialize LCD
  lcd.print("Varad Vinayak");  //Write once to the LCD
  lcd.setCursor(0, 2);
  lcd.print(millis()); //Print Millis for the first time
  PreviousTime = millis();  //Set previous time
  }

  void loop() {
  if ((millis() - PreviousTime) >= 500)
  {
```

```
    PreviousTime = millis();
    lcd.setCursor(0, 1); // Set cursor to second line of the LCD
    lcd.print("   ");  //Clear the second line. This is done to erase all
previous data
    lcd.setCursor(0, 1); //Set the cursor back to the start of the second lir
    lcd.print(Count); //Print the count
    Count = Count + 20; //Increment the count
    lcd.setCursor(8, 2); // Set cursor to 8th column and 3rd row.
    lcd.print(millis()); // Print the millis value
  }
  }
  /*
   The program is an extension of the previous program but done using millis
   Checking the output of millis, we see that the delay is exactly 500 and
there is
   no deviation like what was observed with delay().
*/
/*End of Program 5*/



/*Start of program 6
#include<LiquidCrystal.h>  //Include LCD library to make use of LCD functions
LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to LC
int PreviousTime = 0;
bool LEDState = 0;
void setup() {
  // put your setup code here, to run once:
  pinMode(5, OUTPUT);
  lcd.begin(20, 4); //Initialize LCD
  lcd.setCursor(0, 2);
  lcd.print(millis());//Print Millis for the first time
  PreviousTime = millis(); //Set previous time
}

void loop() {
  if ((millis() - PreviousTime) >= 500)
  {
    PreviousTime = millis(); //Update the previous time
    if (digitalRead (5) == HIGH) // Read the previous LED status to toggle it
    {
      digitalWrite(5, LOW);  //Set to low if it was high previously
    }
    else
```

```
    {
      digitalWrite(5, HIGH); //Set high if it was low previously
    }
    lcd.setCursor(8, 2);
     lcd.print(millis());
  }
}
/*
   LED blink using Millis.
   Delay is seen as exactly 500ms
*/
/*End of Program 6*/


/*Start of exercise 2, problem 1
#include "TimerOne.h"
int LEDLastActive = 0;
int LEDPins[5] = { 3, 4, 5, 6, 7}; //Define all LED port numbers in an array

void setup() {
  // put your setup code here, to run once:
  for(int i = 3; i < 8; i++)
  {
    pinMode(i, OUTPUT);  //Set all LED pins to output
  }
  Timer1.initialize(200000); //To call ISR every 200ms
  Timer1.attachInterrupt(isr); // Attach timer interrupt to ISR
  digitalWrite(LEDPins[LEDLastActive], HIGH); //Write the pin high for the
first time
}

void isr()
{
  digitalWrite(LEDPins[LEDLastActive], LOW);  //Switch off the LED
  LEDLastActive++; //Increment to activate next LED
  LEDLastActive = LEDLastActive % 5; //To make sure the sum is within 5 as we
only have 5 LEDs
  digitalWrite(LEDPins[LEDLastActive], HIGH);  //Switch on the next LED
}
void loop() {
  // Nothing is done in loop. Function handled in ISR

}
/*
```

```
   Blinking of LEDs in a sequencial manner
   Timing is handled by ISR.
*/

/*End of exercise 2, problem 1*/


/*Start of exercise 2, problem 2*
#include "TimerOne.h"
int LEDLastActive = 0;
int LEDPins[5] = { 4,6,3,5,7}; //Define all LED port numbers in an array.
NOTE: ORDER CHANGED TO BLINK IN DIFFERENT SEQUENCE

void setup() {
  // put your setup code here, to run once:
  for(int i = 3; i < 8; i++)
  {
    pinMode(i, OUTPUT);  //Set all LED pins to output
  }
  Timer1.initialize(200000); //To call ISR every 200ms
  Timer1.attachInterrupt(isr); // Attach timer interrupt to ISR
  digitalWrite(LEDPins[LEDLastActive], HIGH); //Write the pin high for the
first time
}

void isr()
{

  digitalWrite(LEDPins[LEDLastActive], LOW);   //Switch off the LED
  LEDLastActive++; //Increment to activate next LED
  LEDLastActive = LEDLastActive % 5; //To make sure the sum is within 5 as we
only have 5 LEDs
  digitalWrite(LEDPins[LEDLastActive], HIGH);   //Switch on the next LED

}
void loop() {
  // Nothing is done in loop. Function handled in ISR

}
/*
   Blinking of LEDs in a non sequencial manner
   Timing is handled by ISR.
*/
```

```
/*End of exercise 2, problem 2*/



/*Start of exercise 2, problem 3*/
#include "TimerOne.h"
int LEDLastActive = 0;
int LEDPins[5] = { 3,4,5,6,7}; //Define all LED port numbers in an array

void setup() {
  // put your setup code here, to run once:
  for(int i = 3; i < 8; i++)
  {
    pinMode(i, OUTPUT);  //Set all LED pins to output
  }
  pinMode(8, INPUT_PULLUP);
  Timer1.initialize(200000); //To call ISR every 200ms
  Timer1.attachInterrupt(isr); // Attach timer interrupt to ISR
  digitalWrite(LEDPins[LEDLastActive], HIGH); //Write the pin high for the
first time
}

void isr()
{
  if(digitalRead(8) == HIGH)
  {
  digitalWrite(LEDPins[LEDLastActive], LOW);   //Switch off the LED
  LEDLastActive++; //Increment to activate next LED
  LEDLastActive = LEDLastActive % 5; //To make sure the sum is within 5 as we
only have 5 LEDs
  digitalWrite(LEDPins[LEDLastActive], HIGH);  //Switch on the next LED
  }
}
void loop() {
  // Nothing is done in loop. Function handled in ISR

}
/*
   Blinking of LEDs in a  seauential manner.
   Blinking stops when button 8 is pressed.
   Timing is handled by ISR.
*/
```

```
/*End of exercise 2, problem 3*/



/* Program 1 start
#include<LiquidCrystal.h>  //Include LCD library to make use of LCD functions

LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to LC
volatile unsigned int Count = 0;  //Count declared globally
int CurrentLevel = LOW;  //Used to check the current level of the digital
encoder pin A(1)
 int PreviousLevel = LOW; // Used to check the current level of the digitql
encoder pin A(0)
void setup() {
  // put your setup code here, to run once:
  for(int i = 0; i < 2; i++)
  {
    pinMode(i, INPUT_PULLUP);  //Set the encoder pins to input
  }
  lcd.begin(20, 4); //Initialize LCD
  lcd.setCursor(0, 1);  //Set cursor to the second line
  PreviousLevel = digitalRead(0);  //Initially store the previous level
  PreviousTime = millis();  //Initially store the previous time
}

void loop() {
CurrentLevel = digitalRead(0);  //Read the current level of the encoder pin
if ((CurrentLevel == HIGH) && (PreviousLevel == LOW))  //Check the current
and previous states of the controller pin to check if the digital encoder was
turned
{
//Encoder turned, check the direction by reading pin B
  if(digitalRead(1) == LOW)
  {
  // Clockwise, increment count
    if(Count != 63) //Make sure count does not cross 64
    Count = (Count + 1);
  }
  else
  {
    if(Count != 0)  //Make sure the count does not go below 0
    Count = (Count - 1);
  }
```

```
}
PreviousLevel = CurrentLevel;  //Update the current and previous levels
if ((millis() - PreviousTime)    > 100)  //Calculate delay using millis
function. Delay is 100 to keep the refresh rate at 100ms.
{
  PreviousTime = millis();  //Update previous time with latest reading from
millis
  lcd.clear();  //Clear the display before writing another value
  lcd.setCursor(0,1);    // Reset the cursor
  lcd.print(Count);    //Write the latest value
}

}

/* Program 1 end */

/* Program 2 start*
#include<LiquidCrystal.h>  //Include LCD library to make use of LCD functions

LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to
volatile unsigned int Count = 0;
int CurrentLevel = LOW; //Used to check the current level of the digital
encoder pin A(1)
int PreviousLevel = LOW; // Used to check the current level of the digitql
encoder pin A(0)
int PreviousTime = 0; // Used to check previous time
void setup() {
  // put your setup code here, to run once:
  for(int i = 0; i < 2; i++)
  {
    pinMode(i, INPUT_PULLUP);  //Set all LED pins to output
  }
  lcd.begin(20, 4); //Initialize LCD
  lcd.setCursor(0, 1);  // set the cursor at 1st line
  PreviousLevel = digitalRead(0); //Initially store the previous level
  PreviousTime = millis(); //Initially store the previous time
  analogWriteFrequency(23,25000);  //Initialize PWM with frequency of 25Khz
  analogWriteResolution(6);  //Set resolution to 6 bit
}

void loop() {
CurrentLevel = digitalRead(0); //Read the current level of the encoder pin
if ((CurrentLevel == HIGH) && (PreviousLevel == LOW))  //Check the current
and previous states of the controller pin to check if the digital encoder was
```

```
turned
{
  if(digitalRead(1) == LOW)
  {
    if(Count != 63)
    {
      Count = (Count + 1);
    }
  }
  else
  {
    if(Count != 0)
    {
      Count = (Count - 1);
    }
  }
}
analogWrite(23,Count);  //Set the PWM Output based on count value. Count
value determines the resolution. Value is between 0 and 63. This inturn
changes the speed of the motor.
PreviousLevel = CurrentLevel; //Update the current and previous levels
if ((millis() - PreviousTime)   > 100)
{
  PreviousTime = millis();  //Update previous time with latest reading from
millis
  lcd.clear(); //Clear the display before writing another value
  lcd.setCursor(0,1);  //Initially store the previous level
  lcd.print(Count);  //Write the latest value
}

}
/* Program 2 end*/


/*Program 3 start*/
#include<LiquidCrystal.h>  //Include LCD library to make use of LCD functions

LiquidCrystal lcd (18, 17, 16, 15, 14, 19); //Select the pins connected to
volatile unsigned int Count = 0; //Count declared globally
int CurrentLevel = LOW; //Used to check the current level of the digital
encoder pin A(1)
int PreviousLevel = LOW; // Used to check the current level of the digitql
encoder pin A(0)
int PreviousTime = 0; // Used to check previous time
```

```
int Rest = 0;    //To count the rest time of 2 seconds
int Shutdown = 0;  //To indicate shutdown
int PWMActive = 0;  //To indicate that encoder was active last cycle
void setup() {
  // put your setup code here, to run once:
  for(int i = 0; i < 2; i++)
  {
    pinMode(i, INPUT_PULLUP);  //Set all LED pins to output
  }
  lcd.begin(20, 4); //Initialize LCD
  lcd.setCursor(0, 1); // set the cursor at 1st line
  PreviousLevel = digitalRead(0); //Initially store the previous level
  PreviousTime = millis(); //Initially store the previous time
  analogWriteFrequency(23,25000); //Initialize PWM with frequency of 25Khz
  analogWriteResolution(6); //Set resolution to 6 bit
}

void loop() {
CurrentLevel = digitalRead(0); //Read the current level of the encoder pin
if ((CurrentLevel == HIGH) && (PreviousLevel == LOW) && (Shutdown == 0))
//Also shutdown is checked because the motor shall not be controlled via the
encoder when auto shutdown is in place
{
  Rest = 0;  //Reset rest to 0 as pin has been changed
  if(digitalRead(1) == LOW)
  {
    if(Count != 63)
    {
      Count = (Count + 1);
    }
  }
  else
  {
    if(Count != 0)
    {
      Count = (Count - 1);
    }
  }
}
else
{

}
analogWrite(23,Count); //Set the PWM Output based on count value. Count value
```

```
determines the resolution. Value is between 0 and 63. This inturn changes the
speed of the motor.

PreviousLevel = CurrentLevel; //Update the current and previous levels
/*
 * The below section is executed every 100 ms. Delay calculated using millis.
The last section (outside the if and else blocks) is for updating the LCD
display.
 * If the shutdown has been initiated, count value driving the PWM is
decremented by 1 each time until it reaches 0.
 * Once the count reaches zero, the shutdown is mode is disabled.
 * The else if section handles the timeout if no input is received. Each
time, the rest counter is incremented by 100. Once the counter reaches a
value of 2000 (which indicates 2 seconds), shutdown is initiated. This
counter is reset
 * when the input is seen on the encoder or if the max value of 2000 is
reached.
 */
if ((millis() - PreviousTime)   > 100)  //Used to calculate the 100ms wait
time for LCD refresh, shutdown decrement or input timeout
{
  if(Shutdown == 1)  //Check if shutdown was invoked
  {
    if(Count == 0)  //Check if shutdown is complete
    {
      Shutdown = 0;  //Set shutdown to off
    }
    else
    {
       Count--;
    }
  }
  else
  {
    Rest += 100;  //Increment rest timer by 100ms
    if(Rest == 2000)  // If rest timer has reached the value of 2s initiate
shutdown
  {
    Shutdown = 1;  //Initiate shutdown
    Rest = 0;   //Reset rest timer
    PWMActive = 0;  //Set this variable to zero to indicate that there was no
recent digital encoder input
  }
  }
```

```
  PreviousTime = millis(); //Update previous time with latest reading from
millis
  lcd.clear(); //Clear the display before writing another value
  lcd.setCursor(0,1); //Initially store the previous level
  lcd.print(Count);  //Write the latest value
}

}
/*Program 3 end*/
```