```
/*This is the code for an escargo nursery. The system needs to measure the temperature,
humidity and light level.
    It reads the information then controls the environment automatically to create a comfortable
environment for the snails.
    The system has outputs : 2 LED module and a 12V fan to provide airflow and cooling.
    There is also an Nokia 5110 LCD screen in order to allow users to know the snail life conditions in
real time
    Coding by Juliette PORTEFAIX, Team 1, Caracol. May 2017 */

    /* Connections between the components and the arduino
    pin A5 - SDA TSL2561
    pin A4 - SCL TSL2561
    pin 12 - Fan
    pin 11 - LED module
    pin 7 - LCD Serial clock out (SCLK)
    pin 6 - LCD Serial data out (DIN)
    pin 5 - LCD Data/Command select (D/C)
    pin 4 - LCD chip select (CS)
    pin 3 - LCD reset (RST)
    pin 2 - DHT 22
    */

    #include <TimeLib.h>
    #include <Adafruit_TSL2561_U.h>
    #include <pgmspace.h>
    #include <Wire.h>
    #include <Adafruit_Sensor.h>
    #include <DHT.h>
    #include <DHT_U.h>
    #include <Adafruit_GFX.h>
    #include <Adafruit_PCD8544.h> //declaration of all the libraries


    #define TIME_HEADER  "T"   // Header tag for serial time sync message
    #define TIME_REQUEST  7    // ASCII bell character requests a time sync message
    #define DHTPIN 2
    #define DHTTYPE DHT22

    DHT dht(DHTPIN, DHTTYPE);

    Adafruit_PCD8544 display = Adafruit_PCD8544(7, 6, 5, 4, 3);

    Adafruit_TSL2561_Unified tsl = Adafruit_TSL2561_Unified(TSL2561_ADDR_FLOAT, 12345);

    #define NUMFLAKES 10
    #define XPOS 0
    #define YPOS 1
    #define DELTAY 2

    #define LOGO16_GLCD_HEIGHT 16
    #define LOGO16_GLCD_WIDTH  16
```

```
static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

void setup()  {

  Serial.begin(9600);

  display.begin();
  display.setContrast(50);

  display.display(); // show splashscreen
  delay(2000);
  display.clearDisplay();   // clears the screen and buffer


  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);

  setSyncProvider(requestSync);
  Serial.println("Waiting for sync message");
  boolean timeWasSet=false;
  while(!timeWasSet){
    //Serial.println("Waiting for the time");
    if (Serial.available()) {
      processSyncMessage();
    }
    if (timeStatus()!= timeNotSet) {
       digitalClockDisplay();
       timeWasSet=true;
    }
  } //time

  Serial.println("Light Sensor Test"); Serial.println("");
   /* Initialise the sensor */
  if(!tsl.begin())
  {
    /* There was a problem detecting the TSL2561 ... check your connections */
```

```
    Serial.print("Ooops, no TSL2561 detected ... Check your wiring or I2C ADDR!");
    while(1);
  }

  /* Setup the sensor gain and integration time */
  configureSensor();

  /* We're ready to go! */
  Serial.println("");
} //light sensor

void loop(){
  checkBrightness();
  checkHumidityAndTemperature();
  delay(5000);
}

void digitalClockDisplay(){
  // digital clock display of the time
  Serial.print(hour());
  printDigits(minute());
  printDigits(second());
  Serial.print(" ");
  Serial.print(day());
  Serial.print(" ");
  Serial.print(month());
  Serial.print(" ");
  Serial.print(year());
  Serial.println();
}


void printDigits(int digits){
  // utility function for digital clock display: prints preceding colon and leading 0
  Serial.print(":");
  if(digits < 10)
    Serial.print('0');
  Serial.print(digits);
}


void processSyncMessage() {
  unsigned long pctime;
  const unsigned long DEFAULT_TIME = 1357041600; // Jan 1 2013

  if(Serial.find(TIME_HEADER)) {
     pctime = Serial.parseInt();
     if( pctime >= DEFAULT_TIME) { // check the integer is a valid time (greater than Jan 1 2013)
       setTime(pctime); // Sync Arduino clock to the time received on the serial port
     }
  }
}
```

```
void checkBrightness(){

  display.setTextSize(1);
  display.setTextColor(BLACK);
  display.setCursor(0,0);
  /* Get a new sensor event */
  sensors_event_t event;
  tsl.getEvent(&event);

  /* Display the results (light is measured in lux) */
  if (event.light)
  {
    Serial.print(event.light); Serial.println(" lux");
     display.display();
  delay(2000);
  display.clearDisplay();
    display.println(event.light); display.println(" lux");
  }
  else
  {
    /* If event.light = 0 lux the sensor is probably saturated
       and no reliable data could be generated! */
    Serial.println("Sensor overload");
  }

  if (hour()>=7 && hour()<= 23 && event.light < 50)
    {
    digitalWrite(11, HIGH); // LED on
  }
  else {
    digitalWrite(11, LOW);  // LED off
  }
}

void checkHumidityAndTemperature(){
  Serial.println("DHTxx test!");
  dht.begin();
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

    // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
```

```
 // Compute heat index in Celsius (isFahreheit = false)
 float hic = dht.computeHeatIndex(t, h, false);

 Serial.print("Humidity: ");
 Serial.print(h);
 Serial.print(" %\t");
 Serial.print("Temperature: ");
 Serial.print(t);
 Serial.print(" *C ");
 Serial.print(f);
 Serial.print(" *F\t");
 Serial.print("Heat index: ");
 Serial.print(hic);
 Serial.print(" *C ");
 Serial.print(hif);
 Serial.println(" *F");

display.display();
 delay(2000);
 display.clearDisplay();

 display.setTextSize(1


 );
 display.setTextColor(BLACK);
 display.setCursor(0,0);
 display.println("Humidity: ");
 display.println(h);
 display.println(" %\t");
 display.println("Temperature: ");
 display.println(t);
 display.println(" *C ");


 if (t > 25){
  digitalWrite(12,HIGH);
 }
 else {
  digitalWrite(12, LOW);
 }
}


time_t requestSync()
{
 Serial.write(TIME_REQUEST);
 return 0; // the time will be sent later in response to serial mesg
}

void configureSensor()
{
```

```
      tsl.enableAutoRange(true);          /* Auto-gain ... switches automatically between 1x and 16x */
      tsl.setIntegrationTime(TSL2561_INTEGRATIONTIME_101MS);  /* medium resolution and speed
*/
     }
```