#define USE\_ARDUINO\_INTERRUPTS true // Set-up low-level interrupts for most acurate BPM math.

#include <PulseSensorPlayground.h> // Includes the PulseSensorPlayground Library.

#include<LiquidCrystal.h>

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

// Variables

const int PulseWire = 1; // PulseSensor PURPLE WIRE connected to ANALOG PIN 0

const int LED13 = 13; // The on-board Arduino LED, close to PIN 13.

int Threshold = 550; // Determine which Signal to "count as a beat" and which to ignore.

// Use the "Gettting Started Project" to fine-tune Threshold Value beyond default setting.

// Otherwise leave the default "550" value.

PulseSensorPlayground pulseSensor; // Creates an instance of the PulseSensorPlayground object called "pulseSensor"

void setup() {

Serial.begin(9600); // For Serial Monitor

lcd.begin(20,4);

// Configure the PulseSensor object, by assigning our variables to it.

pulseSensor.analogInput(PulseWire);

pulseSensor.blinkOnPulse(LED13); //auto-magically blink Arduino's LED with heartbeat.

pulseSensor.setThreshold(Threshold);

// Double-check the "pulseSensor" object was created and "began" seeing a signal.

if (pulseSensor.begin()) {

Serial.println("We created a pulseSensor Object !"); //This prints one time at Arduino power-up, or on Arduino reset.

lcd.setCursor(0,0);

lcd.print(" Heart Rate Monitor");

}

}

void loop() {

int myBPM = pulseSensor.getBeatsPerMinute(); // Calls function on our pulseSensor object that returns BPM as an "int".

// "myBPM" hold this BPM value now.

if (pulseSensor.sawStartOfBeat()) { // Constantly test to see if "a beat happened".

Serial.println("♥ A HeartBeat Happened ! "); // If test is "true", print a message "a heartbeat happened".

Serial.print("BPM: "); // Print phrase "BPM: "

Serial.println(myBPM); // Print the value inside of myBPM.

lcd.setCursor(0,2);

lcd.print("HeartBeat Happened !"); // If test is "true", print a message "a heartbeat happened".

}

delay(20); // considered best practice in a simple sketch.

myBPM=0;

}