Programming Explanation:

In this **Earthquake Detector Arduino Shield**, we have made **two codes**: one for Arduino to detect an earthquake and another for Processing IDE to plot the earthquake vibrations over the graph on Computer. We will learn about both the codes one by one:

Arduino code:

First of all, we **calibrate the accelerometer** with respect to its placing surface, so that it will not show alerts with respect to its normal surrounding vibrations. In this calibration, we take some samples and then take an average of them and stores in a variable.

```
for(int i=0;i<samples;i++)      // taking samples for calibration
{
  xsample+=analogRead(x);
  ysample+=analogRead(y);
  zsample+=analogRead(z);
}

xsample/=samples;   // taking avg for x
ysample/=samples;   // taking avg for y
zsample/=samples;   // taking avg for z

delay(3000);
lcd.clear();
lcd.print("Calibrated");
delay(1000);
lcd.clear();
lcd.print("Device Ready");
delay(1000);
lcd.clear();
lcd.print(" X     Y     Z   ");
```

Now whenever Accelerometer takes readings, we will subtract those sample values from the readings so that it can ignore surroundings vibrations.

```
  int value1=analogRead(x);   // reading x out
  int value2=analogRead(y);   //reading y out
  int value3=analogRead(z);   //reading z out

  int xValue=xsample-value1;    // finding change in x
  int yValue=ysample-value2;    // finding change in y
  int zValue=zsample-value3;    // finding change in z

/*displying change in x,y and z axis values over lcd*/
  lcd.setCursor(0,1);
  lcd.print(zValue);
  lcd.setCursor(6,1);
  lcd.print(yValue);
  lcd.setCursor(12,1);
```

```
    lcd.print(zValue);
    delay(100)
```

Then Arduino compares those calibrated (subtracted) values with predefined limits. And take action accordingly. If the values are higher than predefined values then it will beep the buzzer and plot the vibration graph on computer using Processing.

```
      /* comparing change with predefined limits*/
    if(xValue < minVal || xValue > maxVal  || yValue < minVal || yValue > maxVal  || zValue < minVal
    {
      if(buz == 0)
      start=millis();   // timer start
       buz=1;       // buzzer / led flag activated
    }

  else if(buz == 1)        // buzzer flag activated then alerting earthquake
  {
     lcd.setCursor(0,0);
     lcd.print("Earthquake Alert   ");
     if(millis()>= start+buzTime)
     buz=0;
  }
```

Processing code:

Below is the Processing Code attached, you can download the code from below link:
Earth Quake Detector Processing Code

We have designed a graph using Processing, for earth quake vibrations, in which we defined the size of the window, units, font size, background, reading and displaying serial ports, open selected serial port etc.

```
// set the window size:  and Font size
 f6 = createFont("Arial",6,true);
 f8 = createFont("Arial",8,true);
 f10 = createFont("Arial",10,true);
 f12 = createFont("Arial",12,true);
 f24 = createFont("Arial",24,true);
 size(1200, 700);

// List all the available serial ports
println(Serial.list());
myPort = new Serial(this, "COM43", 9600);
println(myPort);
myPort.bufferUntil('\n');
background(80)
```

In below function, we have received data from serial port and extract required data and then mapped it with the size of the graph.

```
// extracting all required values of all three axis:
int l1=inString.indexOf("x=")+2;
String temp1=inString.substring(l1,l1+3);
l1=inString.indexOf("y=")+2;
String temp2=inString.substring(l1,l1+3);
l1=inString.indexOf("z=")+2;
String temp3=inString.substring(l1,l1+3);

//mapping x, y and z value with graph dimensions
float inByte1 = float(temp1+(char)9);
inByte1 = map(inByte1, -80,80, 0, height-80);
float inByte2 = float(temp2+(char)9);
inByte2 = map(inByte2,-80,80, 0, height-80);
float inByte3 = float(temp3+(char)9);
inByte3 = map(inByte3,-80,80, 0, height-80);
float x=map(xPos,0,1120,40,width-40)
```