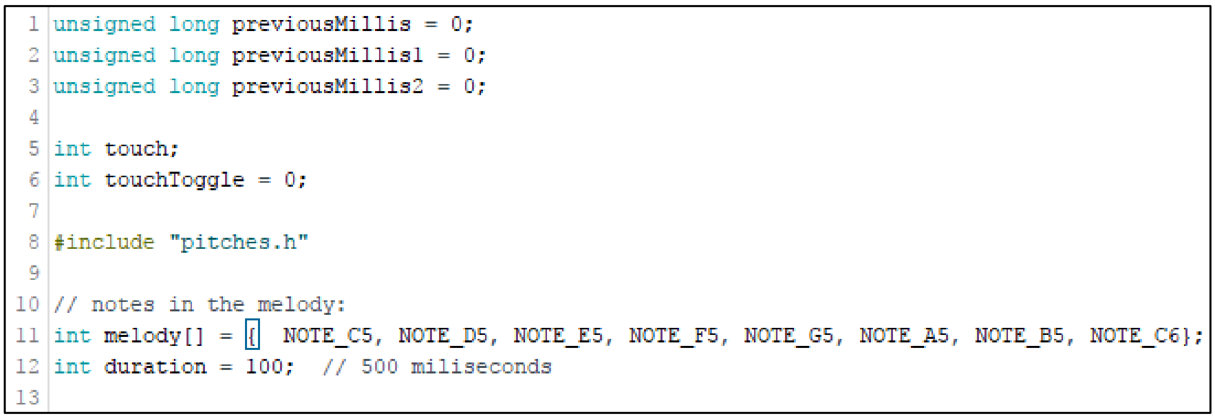
## Alarm system

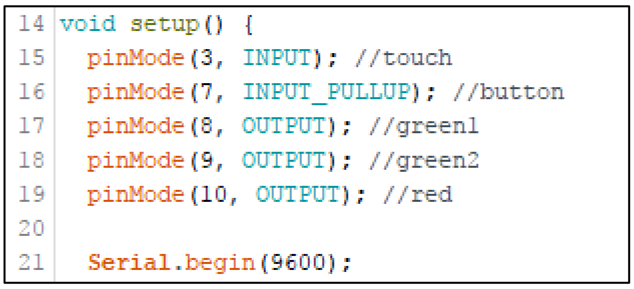
Our Arduino alarm system works like a simple alarm with a touch sensor that is activated with the touch of a finger. After which a set of three LED lights will turn on indicating the alarms progress whereas it will start a buzzer. The alarm can be then reset with the press of a button.

Before the setup we define a few variables like millis, intergers and the library needed for the notes in the buzzer



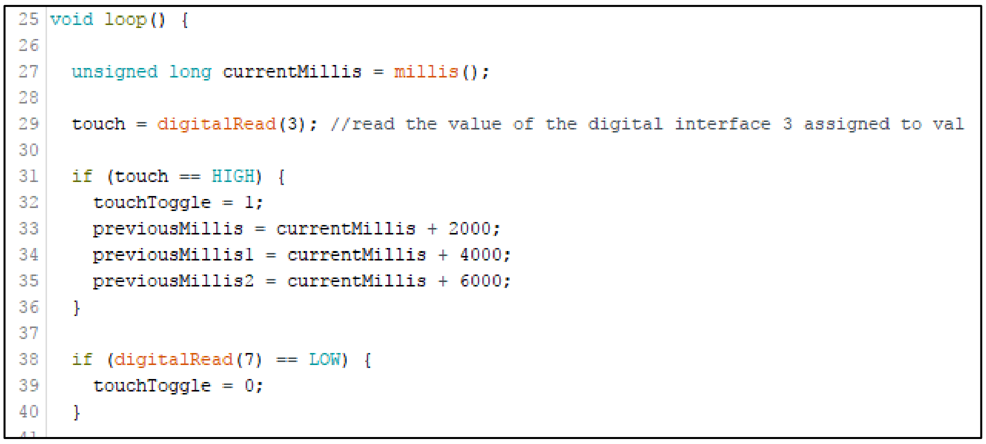
```
unsigned long previousMillis = 0;
unsigned long previousMillis1 = 0;
unsigned long previousMillis2 = 0;

int touch;
int touchToggle = 0;

#include "pitches.h"

// notes in the melody:
int melody[] = {  NOTE_C5, NOTE_D5, NOTE_E5, NOTE_F5, NOTE_G5, NOTE_A5, NOTE_B5, NOTE_C6};
int duration = 100;  // 500 miliseconds

```

Afterwards the setup comes, where both inputs and outputs are defined. These inputs and outputs are for buttons, LED’s and the touch sensor.



```
void setup() {
  pinMode(3, INPUT); //touch
  pinMode(7, INPUT_PULLUP); //button
  pinMode(8, OUTPUT); //green1
  pinMode(9, OUTPUT); //green2
  pinMode(10, OUTPUT); //red

  Serial.begin(9600);
```

Now as we are in the void loop we need to make sure the program checks if the touch sensor is activated. This is done with an If statement that says if it is HIGH (activated) then it sets the variable “touchToggle” to 1, and then it also sets three millis variables with 2000 millis difference. These millis are used to make the delay between the LED’s without using the built in delay function that Arduino has because that function will basically stop the whole program for the duration of the delay.

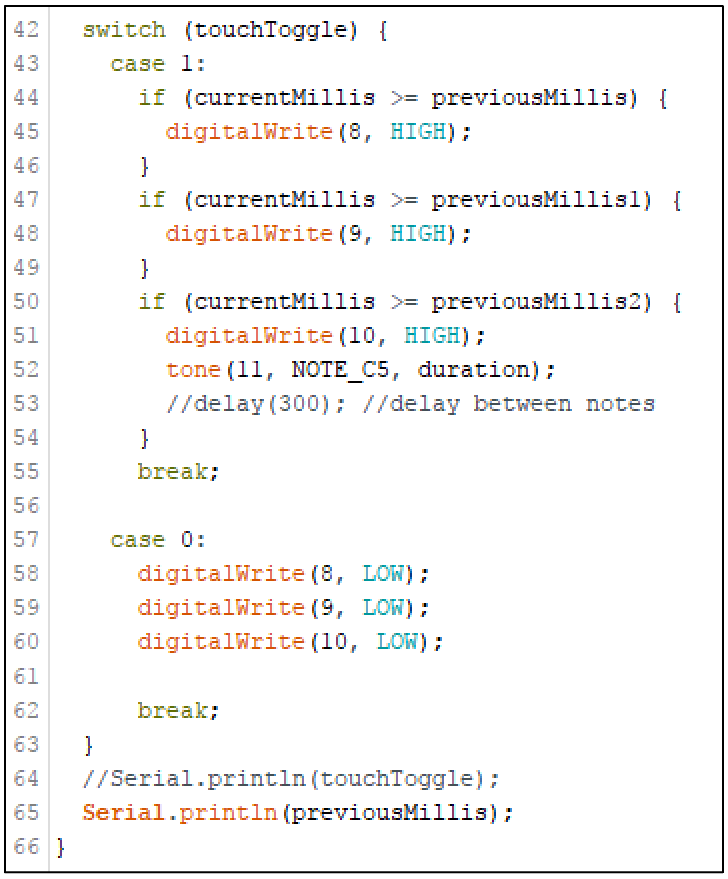


```
void loop() {

  unsigned long currentMillis = millis();

  touch = digitalRead(3); //read the value of the digital interface 3 assigned to val

  if (touch == HIGH) {
    touchToggle = 1;
    previousMillis = currentMillis + 2000;
    previousMillis1 = currentMillis + 4000;
    previousMillis2 = currentMillis + 6000;
  }

  if (digitalRead(7) == LOW) {
    touchToggle = 0;
  }
```

When the touch sensor is activated we can decide what happens using a switch case function. This function checks the variable “touchToggle” and what it is currently set to. For example if it is 1 (case 1) then it will check the millis variables and turn on LED’s when they are equal of less than the previous millis value (essentially just turning on an LED every 2 second). After the third LED is activated (the red one) it will also turn on the buzzer which emits a continuous buzzing sound until interrupted.

If the variable is 0 (case 0) then it will turn off all LED’s and also turn off the buzzer if it was activated.

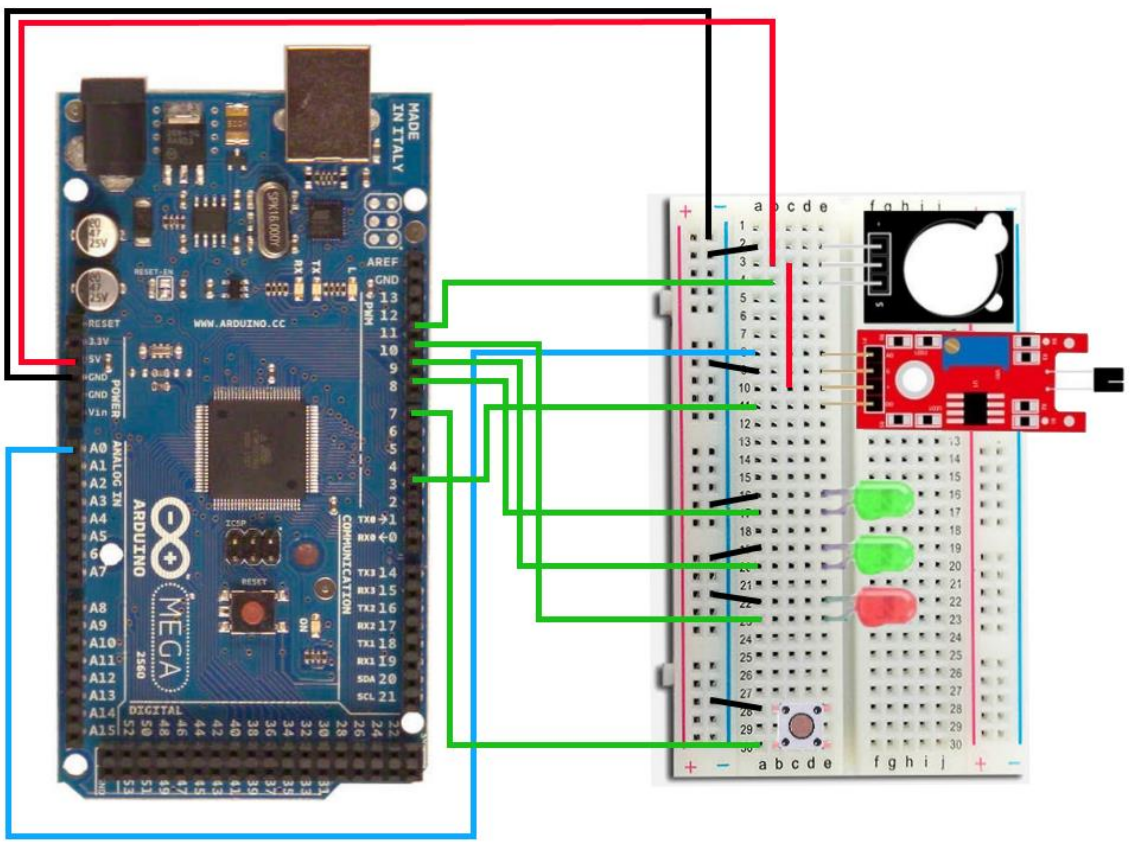
Afterwards there is a break which stops the whole switch case.

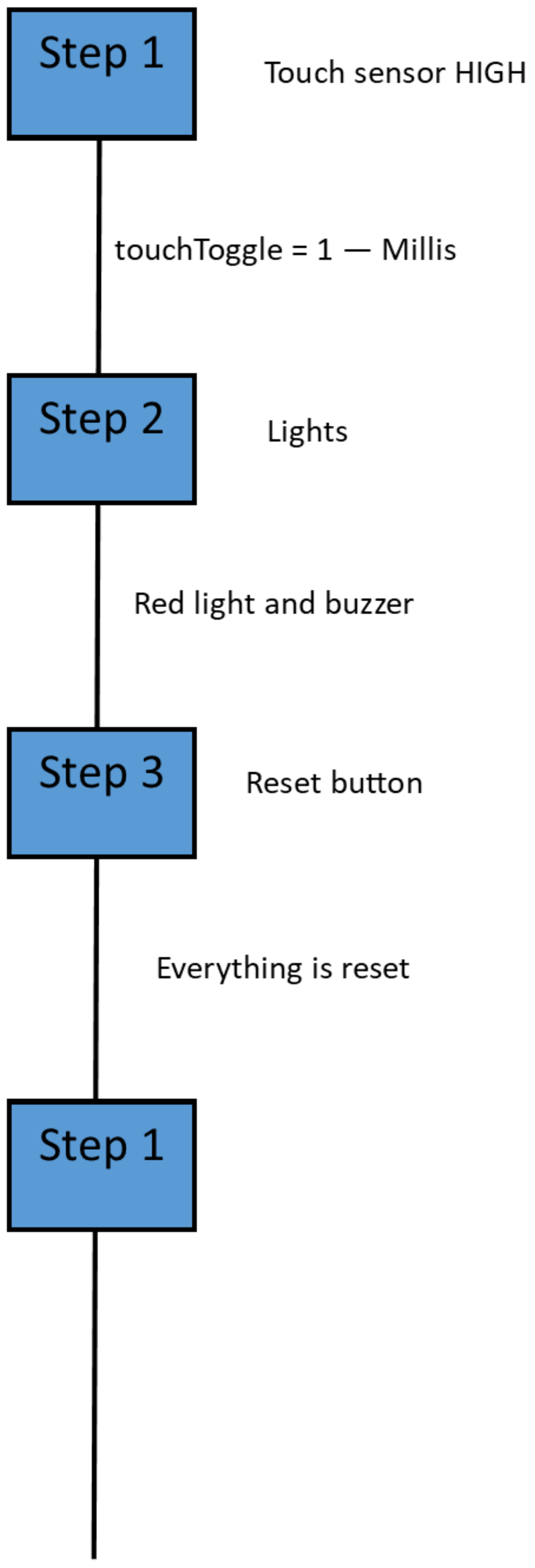


```
  switch (touchToggle) {
    case 1:
      if (currentMillis >= previousMillis) {
        digitalWrite(8, HIGH);
      }
      if (currentMillis >= previousMillis1) {
        digitalWrite(9, HIGH);
      }
      if (currentMillis >= previousMillis2) {
        digitalWrite(10, HIGH);
        tone(11, NOTE_C5, duration);
        //delay(300); //delay between notes
      }
      break;

    case 0:
      digitalWrite(8, LOW);
      digitalWrite(9, LOW);
      digitalWrite(10, LOW);

      break;
  }
  //Serial.println(touchToggle);
  Serial.println(previousMillis);
}
```

## IO list

| **Defined data type** | **Defined I/O – intern logic** | **Comment** |
|---|---|---|
| unsigned long | previousMillis | Remember last millis |
| unsigned long | previousMillis1 | Remember last millis |
| unsigned long | previousMillis2 | Remember last millis |
| Int | Touch | Touch sensor for alarm trigger |
| Int | touchToggle | Touch sensor on/off |
| pinMode 3 | INPUT | Touch sensor |
| pinMode 7 | INPUT_PULLUP | Reset button |
| pinMode 8 | OUTPUT | Green led 1 |
| pinMode 9 | OUTPUT | Green led 2 |
| pinMode 10 | OUTPUT | Red led |




Step 1
Touch sensor HIGH
touchToggle = 1 — Millis
Step 2
Lights
Red light and buzzer
Step 3
Reset button
Everything is reset
Step 1