

```

#include <DHT11.h>           // Βιβλιοθήκη αισθητήρα θερμοκρασίας και υγρασίας (dht11)
#include <DS3231.h>          // Βιβλιοθήκη ρολογιού
#include <Wire.h>             // Βιβλιοθήκη καλωδιώσεων
#include <LiquidCrystal_I2C.h> // Βιβλιοθήκη οθόνης
#include <SoftwareSerial.h>    // Βιβλιοθήκη σειριακής επικοινωνίας
#include <DFRobotDFPlayerMini.h> // Βιβλιοθήκη player mp3

DS3231 myRTC;                // ορισμός αντικειμένου ρολογιού
LiquidCrystal_I2C lcd(0x27, 16, 2); // ορισμός αντικειμένου οθόνης

const int pinVibration = 2;   // Έλεγχος για σεισμό
DHT11 dht11(3);              // Ορισμός αντικειμένου ελέγχου θερμοκρασίας και Υγρασίας από την είσοδο 3
const int pinLights = 4;      // Διαχείριση φώτων διαδρόμου
const int pinAskisi = 5;      // Άσκηση σεισμού
const int pinBell = 6;        // Έκτακτο κτύπημα κουδουνιού
const int pinLevita = 7;      // Διαχείριση λέβητα καλοριφέρ
const int pinTempLevita = 8;   // Ορισμός θερμοκρασίας χώρου για την θέρμανση
const int pinFan = 9;         // Διαχείριση ανεμιστήρων
SoftwareSerial softwareSerial(11, 10); // Ορισμός εισόδου-εξόδου player mp3 10-TX, 11-RX
DFRobotDFPlayerMini player;   // Ορισμός αντικειμένου player mp3
const int pinAlarm = 12;      // Ανίχνευση σήματος συναγερμού
const int pinAlarmLed = 13;   // Ενεργοποίηση συναγερμού
const int pinBrightness = A2; // Φωτεινότητα εξωτερικού χώρου

int out = 0;
int min = 0;
int maxTemp = 20;
bool century = false;
bool h12Flag;
bool pmFlag;
String dt;

```

```
void setup () {
  pinMode(LED_BUILTIN, OUTPUT);
  digitalWrite(LED_BUILTIN, LOW);    // σβήσιμο εσωτερικού led

  Wire.begin();
  Serial.begin(57600);
  myRTC.setClockMode(false);        // ορισμός μορφής εμφάνισης 24 ώρης
  softwareSerial.begin(9600);
  if (player.begin(softwareSerial)) player.volume(30);

  pinMode (pinVibration, INPUT);
  pinMode (pinAskisi, INPUT_PULLUP);
  pinMode (pinBrightness, INPUT);
  pinMode (pinBell, INPUT_PULLUP);
  pinMode (pinLights, OUTPUT);
  pinMode (pinLevita, OUTPUT);
  pinMode (pinTempLevita, INPUT);
  pinMode (pinFan, OUTPUT);
  pinMode (pinAlarm, INPUT);
  pinMode (pinAlarmLed, OUTPUT);

  //setDateTime (); //καθορισμός ημέρας και ώρας στο ρολόι DS3231, όταν χρειάζεται
  lcd.begin(16,2);
  lcd.clear();
  lcd.backlight();
  digitalWrite(pinLights, HIGH);
  delay(500);
}
```

```
void loop () {  
    displayDateTime();  
    displayTemperature();  
    bell();  
    emergencyBell();  
    seismos();  
    askisiSeismou();  
    levitas();  
    lights();  
    alarm ();  
}
```

```
void alarm() {                                     // Διαχείριση συναγερμού  
    if (digitalRead(pinAlarm) == HIGH ) {  
        digitalWrite(pinAlarmLed, HIGH);  
        ringBell(5);  
    }  
    else {  
        digitalWrite(pinAlarmLed, LOW);  
    }  
}
```

```
void displayTemperature() {                       // Εμφάνιση Θερμοκρασίας και Υγρασίας  
    float temp = (int)dht11.readTemperature();  
    float humd = (int)dht11.readHumidity();  
    lcd.setCursor(0,0);  
    lcd.print("T:");  
    lcd.setCursor(2,0);  
    lcd.print(temp);  
    lcd.setCursor(4,0);  
    lcd.print("C ");  
    lcd.setCursor(10,0);
```

```

    lcd.print("H:");
    lcd.setCursor(12,0);
    lcd.print(humd);
    lcd.setCursor(14,0);
    lcd.print("% ");
    if (humd>50) { // Άνοιγμα του εξαερισμού λόγω υψηλής υγρασίας
        digitalWrite(9,HIGH);
    }
    else {
        digitalWrite(9,LOW);
    }
}

```

```

void lights() { // Διαχείριση φωτισμού διαδρόμων
    if (myRTC.getHour(h12Flag, pmFlag)>8 && myRTC.getHour(h12Flag, pmFlag)<14) {
        out = analogRead(pinBrightness);
        if (out < 600) {
            digitalWrite(pinLights, LOW);
        }
        else {
            digitalWrite(pinLights, HIGH);
        }
    }
}

```

```

void levitas() { // Διαχείριση θέρμανσης κτιρίου
    if (myRTC.getHour(h12Flag, pmFlag)>7 && myRTC.getHour(h12Flag, pmFlag)<9) { if (dht11.readTemperature() <= maxTemp)
digitalWrite(pinLevita,HIGH); }
    else
        if (myRTC.getHour(h12Flag, pmFlag)>11 && myRTC.getHour(h12Flag, pmFlag)<13) { if (dht11.readTemperature() <= maxTemp)
digitalWrite(pinLevita,HIGH); }
}

```

```
void emergencyBell() { // Χειροκίνητο κτύπημα κουδουνιού
    if (digitalRead(pinBell) == LOW ) ringBell(3);
}
```

```
void askisiSeismou() { // Άσκηση σεισμού
    if (digitalRead(pinAskisi) == LOW ) player.play(2);
}
```

```
void seismos() { // Ανίχνευση σεισμού
    if (digitalRead(pinVibration) == HIGH) player.play(2);
}
```

```
void displayDateTime () { // εμφάνιση της ώρα και της ημέρας
    dt="";
    dt=dt+myRTC.getDate();
    dt=dt+"-";
    dt=dt+myRTC.getMonth(century);
    dt=dt+"-";
    dt=dt+myRTC.getYear();
    dt=dt+" ";
    dt=dt+myRTC.getHour(h12Flag, pmFlag);
    dt=dt+": ";
    dt=dt+myRTC.getMinute();
    lcd.setCursor(1,1);
    lcd.print(dt);
}
```

```

void bell (){                                     // Προγραμματισμός και Έλεγχος πότε χτυπάει το κουδούνι
    if (min != myRTC.getMinute() && myRTC.getHour(h12Flag, pmFlag)>=8 && myRTC.getHour(h12Flag, pmFlag)<=15) {
        if (myRTC.getHour(h12Flag, pmFlag) == 8 && myRTC.getMinute() == 12 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 9 && myRTC.getMinute() == 00 ) {min=myRTC.getMinute(); ringBell(1); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 9 && myRTC.getMinute() == 10 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 9 && myRTC.getMinute() == 55 ) {min=myRTC.getMinute(); ringBell(1); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 10 && myRTC.getMinute() == 05 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 10 && myRTC.getMinute() == 50 ) {min=myRTC.getMinute(); ringBell(1); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 11 && myRTC.getMinute() == 00 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 11 && myRTC.getMinute() == 45 ) {min=myRTC.getMinute(); ringBell(1); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 11 && myRTC.getMinute() == 55 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 12 && myRTC.getMinute() == 40 ) {min=myRTC.getMinute(); ringBell(1); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 12 && myRTC.getMinute() == 50 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 13 && myRTC.getMinute() == 30 ) {min=myRTC.getMinute(); ringBell(1); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 13 && myRTC.getMinute() == 35 ) {min=myRTC.getMinute(); ringBell(3); }
        else if (myRTC.getHour(h12Flag, pmFlag) == 14 && myRTC.getMinute() == 15 ) {min=myRTC.getMinute(); ringBell(3); }
    }
}

```

```

void ringBell(int track){                       // Χτύπημα κουδουνιού
    player.volume(30);
    player.play(track);
}

```

```

void setDateTime () {                          // καθορισμός της ημέρας και ώρας (όποτε χρειάζεται)
    myRTC.setYear(24);
    myRTC.setMonth(02);
    myRTC.setDate(26);
    myRTC.setHour(19);
    myRTC.setMinute(17);
    myRTC.setSecond(20);
}

```

