Código de Arduino del RC security locker

```
#include <deprecated.h>
#include <MFRC522.h>
#include <MFRC522Debug.h>
#include <MFRC522Extended.h>
#include <MFRC522Hack.h>
#include <require_cpp11.h>
#include <Keypad.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <SPI.h>
#include <RFID.h>
#include <Servo.h>



#define SS_PIN 53
#define RST_PIN 5 // PIN de RESET (PWN)
RFID rfid(SS_PIN, RST_PIN);
int ID[5];
int TARJETA[5] = {16, 149, 181, 164, 148};
int correct = 0; // PIN de RESET (PWN)

char tecla;

int conteo_puerta = 0;
int conteo_alarma = 0;

int sensor_puerta = 3;
int m_sensor_puerta = 0;
int cerradura_1 = 8;

int lampara_1 = 6;
int alarma = 2;

int LED_ROJO = 22;
int LED_VERDE = 23;
int LED_AZUL = 24;

int sensor_m = 3;
int led_screen = 9;
int LED_ON = 10;

int corriente = 49;
int DC = 0;
int posicion = 0;
int strike = 0;
int color = 78;
```

```
int incorrecto = 0;
int desbloqueado = 0;
int sistema_bloqueado = 0;
int timer = 0;
int time_block = 0;
int desbloquear = 0;
int menu = 0;

boolean alarmIsOn = false;
boolean ledIsOn = false;
boolean puerta_cerrada = false;
boolean alarma_activada = false;

char secuencia[8];
int posicion_sec = 0;

unsigned long previousTime = 0;
unsigned long currentTime;
unsigned long previousTimeStrike = 0;
unsigned long currentTimeStrike;
unsigned long previousTimeAlarm = 0;
unsigned long currentTimeAlarm;
unsigned long previousTimeCerradura = 0;
unsigned long currentTimeCerradura;
unsigned long previousTimeLED = 0;
unsigned long currentTimeLED;

const byte ROWS = 3; //horizontal
const byte COLS = 3; //vertical

LiquidCrystal_I2C lcd(0x27, 20, 4);

char keys[ROWS][COLS] = {
{'1', '2', '3'},
{'4', '5', '6'},
{'7', '8', '9'}
};
char patron[4] = {'1', '2', '3', '4'};

char patron_sec[9] = {'0', '0', '0', '0', '0', '0', '0', '0', '0'};

byte KEY_PRESSED[] =
{
0b00000000,
0b00001110,
0b00011111,
0b00011111,
0b00011111,
0b00001110,
0b00000000,
0b00000000
};

byte KEY_NOT_PRESSED[] =
{
```

```
0b00000000,
0b00001110,
0b00011011,
0b00010001,
0b00011011,
0b00001110,
0b00000000,
0b00000000
};

byte BATERIA[] = {
B00000,
B00000,
B01010,
B11111,
B10001,
B11111,
B11111,
B00000
};

byte FUENTE[] = {
B01010,
B01010,
B11111,
B10001,
B10001,
B01110,
B00100,
B00100
};

byte ADD_NFC[] = {
B11111,
B10001,
B10101,
B11111,
B10101,
B10001,
B10001,
B11111
};

byte NFC[] = {
B11111,
B11111,
B11111,
B11111,
B11111,
B11111,
B11111,
B11111
};

byte NEW_PATTERN[] = {
B00000,
```

```
B00000,
B11111,
B10101,
B11111,
B10101,
B11111,
B00000
};



byte LAMP[] = {
B00100,
B01010,
B10001,
B10101,
B10101,
B01110,
B01110,
B01110
};

byte CANDADO[] = {
B00100,
B01010,
B01010,
B11111,
B10001,
B10001,
B10001,
B11111
};

byte rowPins[ROWS] = {38, 39, 40}; //connect to the row pinouts of the
keypad
byte colPins[COLS] = {41, 42, 43}; //connect to the column pinouts of the
keypad

Keypad teclado = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
Serial.begin(9600);

lcd.begin();
lcd.backlight();
digitalWrite(led_screen, 1);
lcd.setCursor(5, 1);
lcd.print("RC LOCKER");
lcd.setCursor(2, 2);
lcd.print("by RC Sec. Sys.");

delay(2000);
lcd.clear();

pinMode(corriente, INPUT);
```

```
pinMode(sensor_m, OUTPUT);
digitalWrite(sensor_m, 1);
pinMode(lampara_1, OUTPUT);
pinMode(alarma, OUTPUT);
pinMode(LED_ROJO, OUTPUT);
pinMode(LED_VERDE, OUTPUT);
pinMode(LED_AZUL, OUTPUT);
pinMode(sensor_puerta, INPUT);
pinMode(cerradura_1, OUTPUT);
pinMode(led_screen, OUTPUT);
pinMode(LED_ON, OUTPUT);


lcd.createChar(1, KEY_PRESSED);
lcd.createChar(0, KEY_NOT_PRESSED);
lcd.createChar(2, BATERIA);
lcd.createChar(3, FUENTE);
lcd.createChar(4, ADD_NFC);
lcd.createChar(5, NFC);
lcd.createChar(6, NEW_PATTERN);
lcd.createChar(7, LAMP);
SPI.begin();
rfid.init();
}

void CheckID()
{
correct = 0;

for (int z = 0; z < 5; z++)
{
if (ID[z] == TARJETA[z])
{
correct ++;
}
if (correct == 5)
{
lcd.setCursor(3, 1);
lcd.print("TARJETA VALIDA");

sistema_bloqueado = 0;
timer = 0;
strike = 0;
desbloqueado = 1;
}
}
for (int x = 0; x < 5; x++)
{
ID[x] = 0;
}
}

void set_color() {
switch (color) {
case 1:
digitalWrite(LED_ROJO, 1);
```

```
digitalWrite(LED_AZUL, 0);
digitalWrite(LED_VERDE, 0);
break;
case 2:
digitalWrite(LED_ROJO, 0);
digitalWrite(LED_AZUL, 0);
digitalWrite(LED_VERDE, 1);
break;
case 3:
digitalWrite(LED_ROJO, 0);
digitalWrite(LED_AZUL, 1);
digitalWrite(LED_VERDE, 0);
break;

case 4:
digitalWrite(LED_ROJO, 1);
digitalWrite(LED_AZUL, 1);
digitalWrite(LED_VERDE, 0);
break;
case 5:
digitalWrite(LED_ROJO, 0);
digitalWrite(LED_AZUL, 1);
digitalWrite(LED_VERDE, 1);
break;
case 6:
digitalWrite(LED_ROJO, 1);
digitalWrite(LED_AZUL, 0);
digitalWrite(LED_VERDE, 1);
break;
case 7:
digitalWrite(LED_ROJO, 1);
digitalWrite(LED_AZUL, 1);
digitalWrite(LED_VERDE, 1);
break;
case 78:
digitalWrite(LED_ROJO, 1);
digitalWrite(LED_AZUL, 0);
digitalWrite(LED_VERDE, 0);
delay(1000);
digitalWrite(LED_ROJO, 0);
digitalWrite(LED_AZUL, 0);
digitalWrite(LED_VERDE, 1);
delay(1000);
break;
}
}

void fijar_contrasena() {
lcd.setCursor(0, 0);
lcd.print("Nueva secuencia");
}

void reset() {
posicion = 0;
posicion_sec = 0;
patron_sec[0] = '0';
```

```
patron_sec[1] = '0';
patron_sec[2] = '0';
patron_sec[3] = '0';
patron_sec[4] = '0';
patron_sec[5] = '0';
patron_sec[6] = '0';
patron_sec[7] = '0';
patron_sec[8] = '0';
incorrecto = 0;
}
void alarma_on() {
analogWrite(alarma, 255);
digitalWrite(LED_ROJO, 1);
alarmIsOn = true;
}
void alarma_off() {
analogWrite(alarma, 0);
digitalWrite(LED_ROJO, 0);
alarmIsOn = false;
}

void led_on() {
digitalWrite(LED_ON, 1);
ledIsOn = true;
}
void led_off() {
digitalWrite(LED_ON, 0);
ledIsOn = false;
}

void led_indicador() {
if (currentTimeLED - previousTimeLED > 1000) {
if (ledIsOn) {
led_off();
}
else {
led_on();
}
previousTimeLED = currentTimeLED;
}
}

void iniciar_alarma() {
lcd.setCursor(1, 1);
lcd.print("Calling the police...");
if (currentTimeAlarm - previousTimeAlarm > 250) {
if (alarmIsOn) {
alarma_off();
}
else {
alarma_on();
}
previousTimeAlarm = currentTimeAlarm;
}
}
void bloquear() {
```

```
switch (strike) {
case 3:
time_block = 30000;
break;
case 4:
time_block = 60000;
break;
case 5:
time_block = 180000;
break;
case 6:
time_block = 300000;
break;
case 7:
time_block = 600000;
break;
case 8:
time_block = 1800000;
break;
case 9:
time_block = 3200000;
break;
case 10:
time_block = 43200000;
break;
case 11:
time_block = 86400000;
break;
}
if (strike > 11) {
time_block = 86400000;
}
if (currentTimeStrike - previousTimeStrike < time_block) {
sistema_bloqueado = 1;
desbloquear = 0;
}
else sistema_bloqueado = 0;
}

void loop(
) {
currentTime = millis();
currentTimeStrike = millis();
currentTimeAlarm = millis();
currentTimeLED = millis();
led_indicador();
DC = digitalRead(corriente);
if (DC == 1) {
lcd.setCursor(19, 0);
lcd.write(3);
digitalWrite(led_screen, 1);
}
else
{
lcd.setCursor(19, 0);
lcd.write(2);
digitalWrite(led_screen, 0);
```

```
}

if (rfid.isCard()) //Verifica si hay una tarjeta
{
if (rfid.readCardSerial()) //Funcion que lee la tarjeta
{
Serial.println("El numero de serie de la tarjeta es :");
for (int i = 0; i <= 4; i++)
{
Serial.print(rfid.serNum[i], HEX); //rfid.serNum lee el número de
serie unico de la tarjeta
Serial.print(" ");
ID[i] = rfid.serNum[i];
if ( TARJETA[i] != ID[i]) {
if (i == 4) {
lcd.setCursor(1, 1);
lcd.print("TARJETA INVALIDA");
digitalWrite(alarma, 25);
digitalWrite(led_screen, 1);
delay(250);
digitalWrite(alarma, 0);
delay(750);
lcd.clear();
digitalWrite(led_screen, 0);
}
}
}
Serial.flush();
}
}

CheckID();
rfid.halt();

char tecla = teclado.getKey();
m_sensor_puerta = digitalRead(sensor_puerta);

if (conteo_puerta == 0 && desbloqueado == 1) {
digitalWrite(cerradura_1, 1);
}
if (conteo_puerta == 1 && desbloqueado == 1) {
digitalWrite(cerradura_1, 0);
set_color();
}
if (conteo_puerta == 2 && desbloqueado == 1) {
reset();
digitalWrite(LED_ROJO, 0);
digitalWrite(LED_AZUL, 0);
digitalWrite(LED_VERDE, 0);
desbloqueado = 0;
conteo_puerta = 0;
}

if (m_sensor_puerta == 1 && conteo_puerta == 0) {
conteo_puerta++;
}
```

```
if (m_sensor_puerta == 0 && conteo_puerta == 1) {
conteo_puerta++;
}

if (conteo_alarma == 1 && desbloqueado == false) {
iniciar_alarma();
}
else {
analogWrite(alarma, 0);
}

if (conteo_puerta == 1 && desbloqueado == 0 && conteo_alarma == 0) {
conteo_alarma++;
}

if (sistema_bloqueado == 1) {
lcd.setCursor(1, 0);
lcd.print("Sistema Bloqueado");
lcd.setCursor(0, 2);
lcd.print("Intente de nuevo en:");
lcd.setCursor(4, 3);
lcd.print((time_block - ((currentTimeStrike - previousTimeStrike))) /
1000);
lcd.setCursor(7, 3);
lcd.print("Segundos");
desbloquear = 0;
}

if (strike >= 3) {
if (currentTimeStrike - previousTimeStrike > time_block) {
sistema_bloqueado = 0;
timer = 0;
desbloquear = 1;
sistema_bloqueado = 0;
lcd.clear();
}
else {
sistema_bloqueado = 1;
}
}

if (tecla != NO_KEY && sistema_bloqueado == 0) {
if (desbloqueado)
{
conteo_alarma = 0;
analogWrite(alarma, 0);
if (tecla == '2')
{
fijar_contrasena();
}
}
if (tecla != patron_sec[1] && tecla != patron_sec[2] && tecla !=
patron_sec[3] && tecla != patron_sec[4] && tecla != patron_sec[5] &&
tecla != patron_sec[6] && tecla != patron_sec[7] && tecla !=
patron_sec[8] && tecla != patron_sec[0]) {
if (posicion_sec < 9) {
```

```
patron_sec[posicion_sec] = tecla;
posicion_sec++;
}
previousTime = currentTime;
lcd.setCursor(9, 1);
lcd.write(byte(0));
lcd.setCursor(10, 1);
lcd.write(byte(0));
lcd.setCursor(11, 1);
lcd.write(byte(0));
lcd.setCursor(9, 2);
lcd.write(byte(0));
lcd.setCursor(10, 2);
lcd.write(byte(0));
lcd.setCursor(11, 2);
lcd.write(byte(0));
lcd.setCursor(9, 3);
lcd.write(byte(0));
lcd.setCursor(10, 3);
lcd.write(byte(0));
lcd.setCursor(11, 3);
lcd.write(byte(0));
for (int i = 0; i < 9; i++) {
if (patron_sec[i] == '1') {
lcd.setCursor(9, 1);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '2') {
lcd.setCursor(10, 1);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '3') {
lcd.setCursor(11, 1);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '4') {
lcd.setCursor(9, 2);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '5') {
lcd.setCursor(10, 2);
lcd.write(byte(1));
}
}
```

```
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '6') {
lcd.setCursor(11, 2);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '7') {
lcd.setCursor(9, 3);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {
if (patron_sec[i] == '8') {
lcd.setCursor(10, 3);
lcd.write(byte(1));
}
}
for (int i = 0; i < 9; i++) {

if (patron_sec[i] == '9') {
lcd.setCursor(11, 3);
lcd.write(byte(1));
}
}
if (tecla == patron[posicion]) {
if (incorrecto != 1) {
posicion++;
}
}
if (posicion == 4) {
desbloqueado = 1;
lcd.clear();
lcd.setCursor(2, 1);
lcd.print("Codigo correcto");
strike = 0;
}
if (posicion_sec == 4 && posicion != 4)
{
incorrecto = 1;
}
if (posicion_sec == 4 && incorrecto == 1)
{
strike++;
if (strike >= 3) {
bloquear();
}
previousTimeStrike = currentTimeStrike;
lcd.clear();
lcd.setCursor(1, 1);
lcd.print("Codigo Incorrecto");
analogWrite(alarma, 10);
delay(250);
analogWrite(alarma, 0);
```

```
delay(1000);
reset();
lcd.clear();
}
}
}
}
```