

onoma ergou = Parking Roda

Onoma Programmatisth = Nikolaos Damianos

hmeromineia (Version 1.6) = 14/1/2019

version = 1.6

PINS

Ir sensor = 20 , analog pin = A0 /// sensor gia antikeimeno katw apo tin mpara

BuzzerPin = 6; // pin you buzzer

buttonPin = 7; // pin tou Service button

Stepper pins = 8,9// pin 9 einai to pin gia ta steps kai to pin8 einai gia to ama stripsei deksia h aristera

keyboard pins = {grammis : (46, 47, 48, 49) , stillis : (50, 51, 52, 53) }

pin othoneis = 31 (gia to contrast) , 12 (gia to rs) , 11 (gia to enable), {22,23,24,25,26,27,28,29} gia ta data

servo mparas = 13

```
////////////////////////////////////  
#include <Arduino.h>  
  
#include <Keypad.h> // vivliothiki gia to plhketrologio  
  
#include <EEPROM.h> // vivliothiki gia thn mnimi arduino  
  
#include <Wire.h> // xreiazetai gia tin o8onh  
  
// #include <LiquidCrystal_I2C.h> // vivliothiki gia tin o8onei  
  
#include <BasicStepperDriver.h>  
  
#include <LiquidCrystal.h>  
  
#include <Servo.h> //vivliothiki gia to servo tis mparas  
  
#include <String.h> //voithitiki
```

```
int IRsensorValue = 0; // ir sensor (sensoras empodiou katw apo tin mpara) arxikopoihsi timis tou
const int analogInPinForIRsensor = A0; // ir sensor analogiko pin tou
int BuzzerPin = 6; // pin tou buzzer
int buttonPin = 7; // pin tou service button
```

```

int available_Spots[N] = {1, 1, 1, 1, 1, 1}; // eleftheres theseis an 1 tote einai elefterh h thesei
int Virtual_array[N] = {0, 0, 0, 0, 0, 0}; // pinakas 8esewn tou parking 1 an exei amaksi 0 an einai keno
int num_Array[N] = {1, 2, 3, 4, 5, 6}; // noumera ka8e vagoniou

int eeprom_FirstTime_open = 0; // dieuthinsi stin mnimi eeprom gia elenxo an anigi protoi fora to
arduino

int eeprom_V_array_address = 1; // arxiki dieuthinsi stin mnimi eeprom pou apothikevetai to
Virtual_array 1 mexri N

int eeprom_NUM_array_address = N + 1; // dieuthinsi stin mnimi eeprom pou apothikevetai to
num_Array N+1 ews 2*N

int eeprom_available_address = (2*N)+1; // dieuthinsi stin mnimi eeprom pou apothikevetai to
available_Spots 2*N+1 ews 3*N

int notfirstTime;

//const int Stepper_stepPin = 3; // pin stepper (rodas) gia ta step
//const int Stepper_dirPin = 4; // pin stepper (rodas) gia ta fora tis rodas (aristera dekcia)
bool Service_mode = false; // metavliti pou dilwnei an eimaste se service mode
Spot theseis; // structure pou dilwnei pou 8a mpi to kainourgio amaksi etsi wste na yparxh isoropoia

const byte ROWS = 4; // poses grammes exei to pliktrologio
const byte COLS = 4; // poses stilles exei to pliktrologio

char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
}; // mapping koumpion tou pliktrologiou

byte rowPins[ROWS] = {46, 47, 48, 49}; // pin ton grammwn tou pliktrologiou

```

```
byte colPins[COLS] = {50, 51, 52, 53}; // pin ton stillwn tou pliktrologiou
```

```
int IR_obstacle_sensor = 20; // pin tou ir sensor
```

```
///// gia o8onei
```

```
int rs = 12;
```

```
int enable=11;
```

```
int contrast=20;
```

```
int d0=22;
```

```
int d1=23;
```

```
int d2=24;
```

```
int d3=25;
```

```
int d4=26;
```

```
int d5=27;
```

```
int d6=28;
```

```
int d7=29;
```

```
////////////////////////////////////
```

```
//LiquidCrystal_I2C lcd(0x27, 20, 4); // antikeimeno tis klassis LiquidCrystal_I2C gia ton elenxo tis  
o8oneis h parametroi einai (0x27) mnimi tis o8oneis , 20 xaraktires ana grammi ,4 grammes
```

```
LiquidCrystal lcd(rs, enable, d0, d1,d2,d3, d4, d5, d6, d7);
```

```
BasicStepperDriver stepper(MOTOR_STEPS,DIR,STEP);
```

```
Servo myseervo;// antikeimeno tis klassis Servo gia ton elenxo tis mparas
```

```
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);// antikeimeno tis klassis  
Keypad gia ton elenxo tou pliktrologiou
```

```
////////////////////////////////////  
///
```

```
void update_eeprom() {  
  
    for (int i = eeprom_V_array_address; i < N + 1; i++) {  
        EEPROM.write(i, Virtual_array[i - 1]);  
    }  
  
    int cnt = 0;  
    for (int i = eeprom_NUM_array_address; i < (2 * N + 1); i++) {  
        EEPROM.write(i, num_Array[cnt]);  
        cnt++;  
    }  
  
    cnt = 0;  
    for (int i = eeprom_available_address; i < (3*N + 1); i++){  
        EEPROM.write(i, available_Spots[i - 1]);  
        cnt++;  
    }  
}  
  
/*  
void update_eeprom  
Parametro : {-}  
Ti kanei :update tin mnimi eeprom  
Xreisimotita : update tin mnimi eeprom  
*/  
  
/////////////////////////////////////  
///  
void rotate_array(int Array[]) {  
    int temp = Array[N - 1];
```

```

for (int i = 1 ; i > 0; i--) {
    Array[i] = Array[i - 1];
}
Array[0] = temp;

}

```

```

void rotate_array(int VorN) {
    if (VorN == __virtual__) {
        int temp = Virtual_array[N - 1];

        for (int i = 1 ; i > 0; i--) {
            Virtual_array[i] = Virtual_array[i - 1];
        }
        Virtual_array[0] = temp;
    } else if (VorN == __num_a__) {
        int temp = num_Array[N - 1];

        for (int i = 1 ; i > 0; i--) {
            num_Array[i] = num_Array[i - 1];
        }
        num_Array[0] = temp;

    }
}

/*

```

```

void rotate_array

```

```

Parametroi : {Array: enas pinakas me akeraious ari8mous}

```

Ti kanei :girizei ton pinaka kata mia 8esi

Xreisimotita : mimitai tin roda etsi wste na girnane kai to virtual array ktlp

```
*/  
////////////////////////////////////  
////  
void moveRoda(int steps) {  
  
    digitalWrite(DIR,LOW);  
    stepper.rotate(steps*34*1.8);  
    /*  
    digitalWrite(Stepper_dirPin, HIGH);  
  
    int Nn = 200; /*steps;  
    for (int x = 0; x < Nn; x++) {  
        digitalWrite(Stepper_stepPin, HIGH);  
        delayMicroseconds(500);  
        digitalWrite(Stepper_stepPin, LOW);  
        delayMicroseconds(500);  
    }  
    /*  
    for (int i = 0; i < steps; i++) {  
        rotate_array(__virtual__);  
        rotate_array(__num_a__);  
    }  
    delayMicroseconds(1000);  
    update_eeprom();  
    delayMicroseconds(500);  
}
```

void moveRoda

Parametroi : {steps : enas akeraios ari8mos pou dilonei posa steps 8a kanei h roda (1 step = me apostasi apo vagonei se vagonei)}

Ti kanei : girizei tin roda kata step vagonia

Xreisimotita : girizei tin roda etsi wste na vgei amksi h na mpei h gia to service

*/

```
void screen_print(String minima, int row = 0, int col = 0) {  
    lcd.setCursor(row,col);  
    lcd.print(minima);  
}
```

```
void screen_print

Parametroi : { minima = String me to ti 8a ektiposei stin o8onei , row = akeraios ari8mos pou dilwnei se
pia grammei 8a ektuposei (default timi = 0), col = akeraios ari8mos pou dilwnei se pia stilli 8a ektuposei
(default timi = 0) }

Ti kanei : ektiponei (grafei) stin o8onei

Xreisimotita : grafei stin o8onei

*/
```

```
void clearScreen() {  
    lcd.clear();//Clean the screen  
}
```



```
void clearScreen
```

```
Parametroi : -
```

```
Ti kanei : ka8arizei tin o8onei
```

```
Xreisimotita : ka8arizei tin o8onei
```

```
*/
```

```
////////////////////////////////////  
/////
```

```
void openMpara() {
```

```
if (myseervo.read() == 180) {
```

```
for (int pos = 180; pos >= 95; pos -= 1) {
```

```
myseervo.write(pos);
```

```
delayMicroseconds(500);
```

```
}
```

```
}
```

```
}
```

```
/*
```

```
void openMpara
```

```
Parametroi : -
```

```
Ti kanei : anoigi tin mpara
```

```
Xreisimotita : anoigi tin mpara
```

```
*/
```

```
////////////////////////////////////  
/////
```

```
bool Find_object() {
```

```
IRsensorValue = analogRead(analogInPinForIRsensor);
```

```
if (IRsensorValue <= 200) {
```

```
return true;
```

```

    }

    return false;
}

/*
void Find_object
Parametroi : -
Ti kanei : anoixnevei antikeimenw an uparxh
Xreisimotita : anoixnevei an uparxh antikeimenw etsi wste na min kleisi h mpara
*/

////////////////////////////////////
////

void closeMpara() {
    while (Find_object()) {
        delayMicroseconds(500);
        continue;
    }

    for (int pos = 95; pos <= 180; pos++) {
        myseervo.write(pos);
        delayMicroseconds(500);
    }
}

/*
void closeMpara
Parametroi : -
Ti kanei : klinei tin mpara

```

Xreisimotita : klinei tin mpara

*/

////////////////////////////////////
/////

```
int sumA(int Array[], int len) {
```

```
    int sum = 0;
```

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

```
        sum = sum + Array[i];
```

```
    }
```

```
    return sum;
```

```
}
```

```
int sumA(int arr) {
```

```
    int sum = 0;
```

```
    if (arr == __virtual__) {
```

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

```
            sum = sum + Virtual_array[i];
```

```
        }
```

```
    }
```

```
    else if (arr == __num_a__) {
```

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

```
            sum = sum + num_Array[i];
```

```
        }
```

```
    }
```

```
    return sum;
```

```
}
```

/*

void sumA

Parametroï : {Array = pinakas akerewn, len = mege8os pinaka}

Ti kanei : a8roisma tou pinaka

Xreisimotita : a8roisma tou pinaka

*/

////////////////////////////////////
////

int isoropiaOut() {

int steps = 0;

int Tmp_V_roda[N];

int len = (N / 2) - 1;

memcpy(&Tmp_V_roda, &Virtual_array , sizeof(int)*N);

int minimum[len] = {100, 100};

for (int i = 0; i < N; i++) {

rotate_array(Tmp_V_roda);

int diaf[len];

for (int j = 1; j < (N / 2); j++) {

diaf[j - 1] = abs(Tmp_V_roda[j] - Tmp_V_roda[N - j]);

}

int Sum_diaf = sumA(diaf, len);

if (Sum_diaf == 0) {

minimum[0] = 0;

minimum[1] = 0;

```

        steps = i + 1;
        break;
    } else {
        int cnt = 0;
        for (int i = 0; i < len; i++) {
            if (diaf[i] <= minimum[i]) {
                cnt++;
            }
        }
        if (cnt == len) {

            minimum[0] = diaf[0];
            minimum[1] = diaf[1];

            steps = i + 1;
        } else {
            continue;
        }
    }

}

if (steps >= N - 1) {
    steps = 0;
}

return steps;
}

```

```

/*
    int isoropiaOut

    Parametroi : {-}

    Ti kanei : vriskei posa steps thelei gia na exei isoropia meta apo apoxwrisei autokinitou

    Xreisimotita : isoropia rodas meta apo apoxwrisei

    return : steps gia na exei isoropia

*/

////////////////////////////////////
///

void moveRodaOut() {
    int steps = isoropiaOut();
    moveRoda(steps);
}

/*
    int moveRodaOut

    Parametroi : {-}

    Ti kanei : kounaei tin roda esti wste na einai se isoropia meta apo apoxwriseis autokinitou

    Xreisimotita : isoropia rodas meta apo apoxwrisei

*/

////////////////////////////////////
///

void InputIsoropia() {
    int Tmp_V_roda[N];
    memcpy(&Tmp_V_roda, &Virtual_array, sizeof(int)*N);
    int len = (N / 2) - 1;
    int minimum[len] = {100, 100};
    int i = 0;

```

```

while (i < N) {
    rotate_array(Tmp_V_rodA);
    if (Tmp_V_rodA[Bottom] == 1) {
        i = i + 1;
    } else {
        Tmp_V_rodA[Bottom] = 1;

        for (int j = 0; j < N; j++) {
            rotate_array(Tmp_V_rodA);
            int diaf[len];

            for (int z = 1; z < (N / 2); z++) {
                diaf[z - 1] = abs(Tmp_V_rodA[z] - Tmp_V_rodA[N - z]);
            }

            int cnt = 0;
            for (int ii = 0; ii < len; ii++) {
                if (diaf[ii] <= minimum[ii]) {
                    cnt++;
                }
            }

            if (cnt == len) {

                minimum[0] = diaf[0];
                minimum[1] = diaf[1];

                if (j == N - 1) {

```

```

theseis.newTop = 0;

} else {
    theseis.newTop = j + 1;

}

if (i == N - 1) {
    theseis.newSteps = 0;
} else {
    theseis.newSteps = i + 1;
}
}
}
Tmp_V_roda[Bottom] = 0;
i = i + 1;
}
}
}

/*
int InputIsoropia
Parametroi : {-}

Ti kanei : vriskei kai apo8ikevei sto structure Spot theseis ta steps pou 8a 8elei gia na eixei isoropia
meta tin eisagogi autokinitou kai se pia 8esei 8a eisax8ei to autokinito

Xreisimotita : isoropia rodas kata tin isagogi

*/

```



```
////////////////////////////////////  
///
```

```
void rightSlotToBeFilled() {  
    InputIsoropia();  
    int tmpNumArray[N];  
    memcpy(&tmpNumArray, &num_Array , sizeof(int)*N);  
    for (int i = 0; i < theseis.newSteps; i++) {  
        rotate_array(tmpNumArray);  
    }  
    theseis.New = tmpNumArray[Bottom];  
    theseis.old = N - theseis.newSteps;  
}
```

```
/*
```

```
int rightSlotToBeFilled
```

```
Parametroi : {-}
```

Ti kanei : vriskei kai apo8ikevei sto structure Spot theseis ta steps pou 8a 8elei gia na eixei isoropia meta tin eisagogi autokinitou kai se pia 8esei 8a eisax8ei to autokinito kai to noumero tou vagoniou

Xreisimotita : isoropia rodas kata tin isagogi

```
*/
```

```
////////////////////////////////////  
///
```

```
int find_index(int Num) {  
    int index = 0;  
    for (int i = 0 ; i < N ; i++) {  
        if (num_Array[i] == Num) {  
            index = i;  
            break;  
        }  
    }  
}
```

```

    }
}
return index;
}

```

```

/*
int find_index
Parametroι : {Num = noumero vagoniou }

Ti kanei : vriskei ta steps pou 8elei gia to vagonei me noumero Num na einai sto Bottom

Xreisimotita : vriskei ta steps pou 8elei gia to vagonei me noumero Num na einai sto Bottom etsi wste
na vgi to autokinito
*/

```

```

////////////////////////////////////
///

```

```

String availableParkingSlots() {
    String av_Nums = "";
    for (int spot = 0; spot < N; spot++) {
        if (available_Spots[spot] == 1) {
            av_Nums = av_Nums + (spot + 1) + " ";
        }
    }
    return av_Nums;
}

```

```

/*
int availableParkingSlots
Parametroι : {-}

```

Ti kanei : vriskei tis kenos 8esis

Xreisimotita : gia na diksoumai pies einai oi kenos 8eseis

*/

```
////////////////////////////////////  
///
```

```
bool check_system(){  
    screen_print("Press any button");  
    char key = keypad.getKey();  
    int timer = 10000;  
    while(key == NO_KEY){  
        timer--;  
        key = keypad.getKey();  
        if (timer <= 0){  
            return false;  
        }  
    }  
    clearScreen();  
    screen_print("Press any key ");  
    screen_print("if bar is working",1,0);  
    timer = 10000;  
  
    openMpara();  
    closeMpara();
```

```
key = keypad.getKey();  
while(key == NO_KEY){  
    timer--;  
    key = keypad.getKey();  
    if (timer <= 0){
```

```

        return false;
    }
}

clearScreen();

screen_print("Press any key if ");
screen_print("roda is working",1,0);
timer = 10000;

moveRoda(N);

key = keypad.getKey();
while(key == NO_KEY){
    timer--;
    key = keypad.getKey();
    if (timer <= 0){
        return false;
    }
}

return true;
}

////////////////////////////////////
///

void setup() {

    analogWrite(31,contrast); // contrast o8oneis

```

```

pinMode(IR_obstacle_sensor, INPUT);
pinMode(BuzzerPin, OUTPUT);
//pinMode(Stepper_stepPin, OUTPUT);
//pinMode(Stepper_dirPin, OUTPUT);
pinMode(buttonPin, INPUT);

if (EEPROM.read(eeprom_FirstTime_open) != 1) {
    notfirstTime = 1;
    EEPROM.write(eeprom_FirstTime_open, 1);

    for (int i = 1; i < N + 1; i++) {
        EEPROM.write(i, 0);
    }

    for (int i = N + 1; i < (2 * N + 1); i++) {
        EEPROM.write(i, i - N);
    }

    for (int i = eeprom_available_address; i < (3*N + 1); i++){
        EEPROM.write(1, i - N);
    }

} else {
    notfirstTime = 1;
    for (int i = 0 ; i < N; i++) {
        Virtual_array[i] = EEPROM.read(eeprom_V_array_address + i);
        num_Array[i] = EEPROM.read(eeprom_NUM_array_address + i);
        available_Spots[i] = EEPROM.read(eeprom_available_address + i);
    }
}

```

```

    }
}

lcd.begin(20,4);

stepper.begin(RPM, MICROSTEPS);
myseervo.attach(13); //attaches the servo on pin 13 to the servo object


bool all_ok = check_system();
clearScreen();
while(!all_ok){

    screen_print("Call service immediately!" , 0 ,0);
}


//lcd.begin();

//lcd.backlight();


}


void loop() {
    clearScreen();

```

```
int sumVirtual = sumA(__virtual__);  
int service_button_isActive = digitalRead(buttonPin);
```

```
char key = keypad.getKey();
```

```
if (Service_mode == true) {  
    if (service_button_isActive == HIGH) {  
        moveRoda(1);  
    } else if (key != NO_KEY) {
```

```
        if (key == '#') {  
            Service_mode = false;  
        } else {  
            int steps = key - '0';  
            moveRoda(steps);  
        }  
    }  
}
```

```
} else {  
    if (sumVirtual == N) {  
        screen_print("PARKING IS FULL", 0, 0);  
        screen_print("No Available Spots", 1, 0);  
    } else {  
        screen_print("not full", 0, 0);
```

```
        String av_Nums = availableParkingSlots();  
        screen_print("Free : " + av_Nums, 1, 0);  
    }  
}
```

```

if (service_button_isActive == HIGH) {
    Service_mode = true;
} else {
    if (key == 'A') {
        if (sumVirtual == N) {
            screen_print("Sorry you can not", 2, 0);
            screen_print("Park. Park is Full.", 3, 0);
            delay(5000);
        } else {
            rightSlotToBeFilled();
            int timer = 60000;
            moveRoda(theseis.newSteps);
            openMpara();
            bool endLoop = false;
            while (!endLoop) {
                clearScreen();
                screen_print("Your Slot is : " + theseis.New, 0, 0);
                delay(1);
                timer--;
                char Cancel_Accept = keypad.getKey();
                if (Cancel_Accept != NO_KEY) {
                    if (Cancel_Accept == '*') {
                        Virtual_array[Bottom] = 1;
                        moveRoda(theseis.newTop);
                        available_Spots[theseis.New - 1] = 0;
                        closeMpara();
                        endLoop = true;
                    }
                }
            }
        }
    }
}

```



```

    } else if (Cancel_Accept == '#') {
        moveRoda(theseis.old);
        delayMicroseconds(1000);
        closeMpara();
        endLoop = true;
    } else {
        screen_print("Accept = * " + theseis.New, 1, 0);
        screen_print("Cancel = # " + theseis.New, 2, 0);
    }
} else {
    if (timer <= 15000 && timer > 0) {
        digitalWrite(BuzzerPin, HIGH);
    } else if ( timer <= 0) {
        //Virtual_array[Bottom] = 1;
        digitalWrite(BuzzerPin, LOW);
        moveRoda(theseis.newTop);
        //available_Spots[theseis.New-1]=0;
        endLoop = true;
    }
}

}

} else if (key == 'B') {
    clearScreen();
    screen_print("Your Parking slot :", 0, 0);
    int timer = 30000;
    char thesei = keypad.getKey();
    bool is_ok = true;

```

```

while (NO_KEY && timer > 0) {
    is_ok = false;
    timer--;
    delay(1);
    thesei = keypad.getKey();
    if (thesei) {
        is_ok = true;
    }
}

if (is_ok) {
    char Accept_Cancel = keypad.getKey();
    timer = 30000;
    while (NO_KEY && timer > 0) {
        timer--;
        delay(1);
        Accept_Cancel = keypad.getKey();
    }
    if (Accept_Cancel == '*' || timer <= 0) {
        int slot = thesei - '0';
        moveRoda(find_index(slot));

        Virtual_array[Bottom] = 0;
        delay(30000);
        moveRodaOut();
    } else if (Accept_Cancel == '#') {
        return;
    }
}

```

```
    } else {  
        return;  
    }  
  
    }  
  
    }  
    }  
}
```