



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΗΛΕΚΤΡΟΝΙΚΩΝ ΑΙΣΘΗΤΗΡΙΩΝ

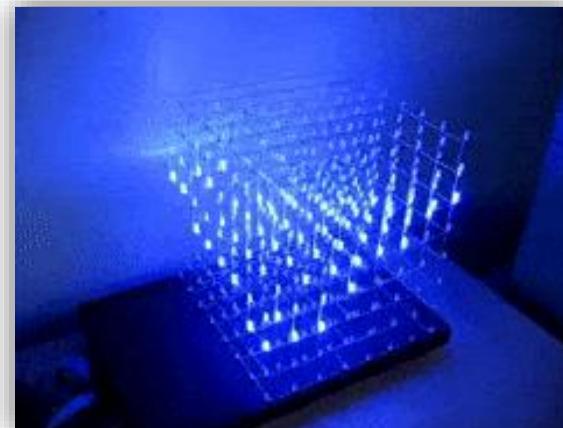
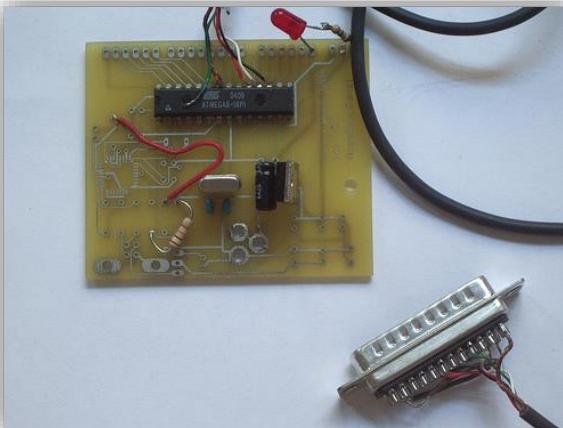
Εισαγωγή στο Arduino

Σπυρίδων Αγγελόπουλος

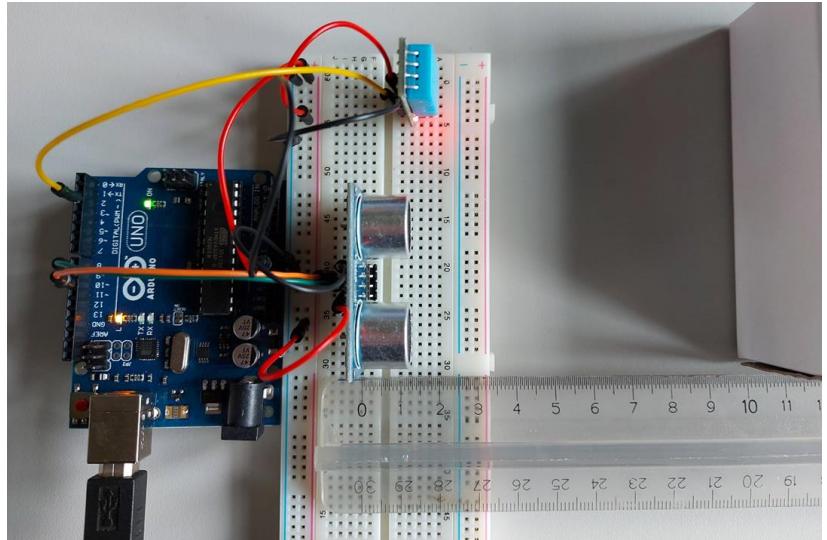


To Arduino

- Κατασκευάστηκε το **2005** στο Interaction Design Institute Ivrea (IDI) της Ιταλίας, από τους M. Banzi, D. Cuartielles, T. Igoe, G. Martino και D. Mellis.
- Προοριζόταν για μαθητές/φοιτητές, δημιουργούς και καλλιτέχνες που **δεν είναι εξοικειωμένοι** με την Ηλεκτρονική & την Πληροφορική.
- Αναπτύχθηκε τόσο το **υλικό**, όσο και το **λογισμικό**, ως open source.
- Το όνομα **Arduino** προήλθε από bar της περιοχής!



Τρόπος χρήσης



8_SR04_DHT11 | Arduino IDE 2.0.0-rc4

```

File Edit Sketch Tools Help
8_SR04_DHT11.ino
1 #include <dht.h>
2
3 #define trigPin 9
4 #define echoPin 10
5 #define DHT11_PIN 2
6
7 float duration, distance;
8 float sound_speed;
9 float temp, hmdt;
10
11 dht DHT;
12
13 void setup() {
14   pinMode(trigPin, OUTPUT);
15   pinMode(echoPin, INPUT);
16   pinMode(DHT11_PIN, INPUT);
17   Serial.begin(9600);
18 }
19
20 void loop() {
21   int chk = DHT.read11(DHT11_PIN);
22   temp = DHT.temperature;
23   hmdt = DHT.humidity;
24
25   digitalWrite(trigPin, LOW);
26   delayMicroseconds(2);
27   digitalWrite(trigPin, HIGH);
28   delayMicroseconds(10);
29   digitalWrite(trigPin, LOW);
30
31   duration = pulseIn(echoPin, HIGH);
32   sound_speed = 331.4 + (0.606 * temp) + (0.0124 * hmdt);
33   distance = (duration / 2) * (sound_speed / 10000);
34
35   if (distance < 400 || distance > 2) {
36     Serial.print(distance);
37     Serial.println(" cm");
}

```

Ln 1, Col 1 UTF-8 C++ X No board selected

Serial Monitor ×

Message (Ctrl + Enter to send message to 'Arduino')

```

=====
10.25 cm
19.00 C
46.00 %
=====
10.36 cm
19.00 C
46.00 %
=====
```

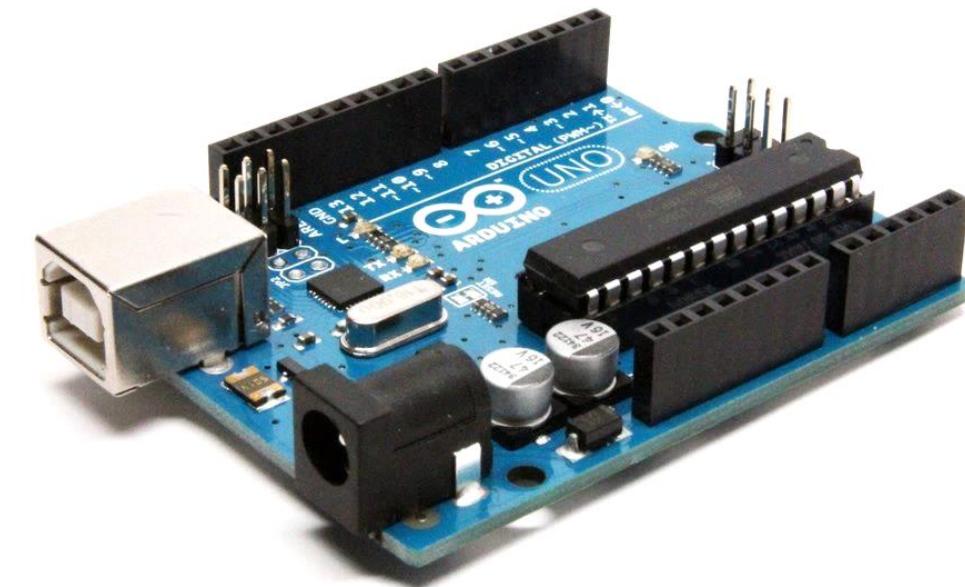
Σύνδεση Arduino με
αισθητήρες, ηλεκτρονικά
εξαρτήματα κ.λπ.

Ανάπτυξη κώδικα
και αποστολή του
στο Arduino

Λήψη
αποτελεσμάτων

Χαρακτηριστικά Arduino UNO R3

Μικροελεγκτής	ATmega328P
Συχνότητα λειτουργίας	16 MHz
Τάση τροφοδοσίας (του UNO)	7-12 V
Τάση λειτουργίας (του MCU)	5 V
Ψηφιακές είσοδοι/έξοδοι	14 (6 PWM)
Αναλογικές είσοδοι	6
Παροχή DC ρεύματος	40 mA (ανά pin) ή 200 mA (συνολικά)
Μνήμη Flash	32 KB
Μνήμη SRAM	2 KB
Μνήμη EEPROM	1 KB



- Αποθήκευση του κώδικα
 Διατήρηση των τιμών των μεταβλητών κατά τη διάρκεια της τροφοδοσίας του Arduino
 Διατήρηση των τιμών των μεταβλητών μετά τη διακοπή της τροφοδοσίας του Arduino

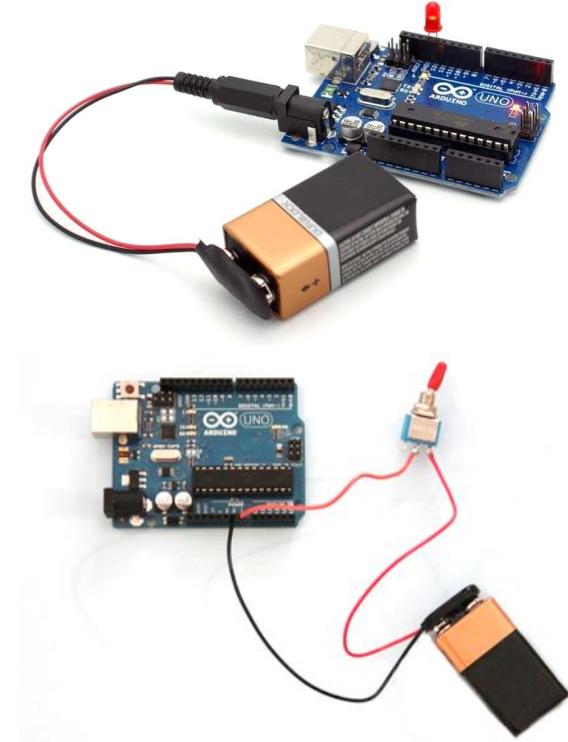
Τρόποι τροφοδοσίας Arduino UNO



Καλώδιο USB
(type A-B)

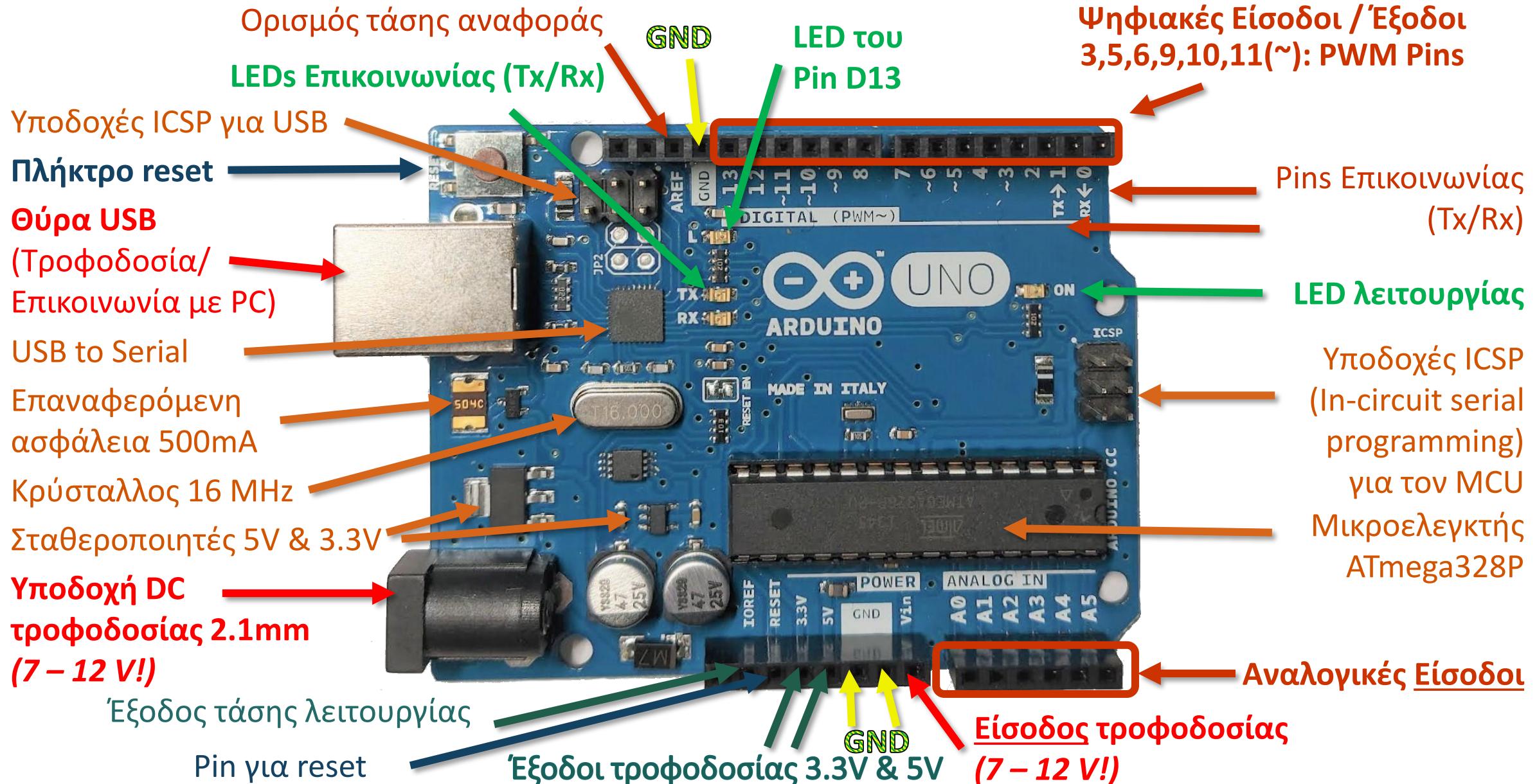


Τροφοδοτικό με
υποδοχή 2.1 mm
(9-12 V)



Μπαταρία 9 V

Στοιχεία πλακέτας Arduino UNO R3





Arduino IDE

Arduino IDE

- Είναι ένα **open-source** λογισμικό που επιτρέπει:
 - Τη συγγραφή του κώδικα
 - Την επικοινωνία με το Arduino
- Από την έκδοση **2.0** και μετά, βασίζεται στο Eclipse Theia IDE framework.
- Ο προγραμματισμός του Arduino γίνεται ουσιαστικά στη γλώσσα C++.
- Είναι συμβατό με **Windows**, **Linux** και **macOS**.
- Υπάρχει και **Online** έκδοση.

```
1 void setup() {  
2     // put your setup code here, to run once:  
3 }  
4  
5 void loop() {  
6     // put your main code here, to run repeatedly:  
7 }  
8  
9 }  
10
```

Ln 10, Col 1 UTF-8 C++ Arduino Uno [not connected]

Στοιχεία γραφικού περιβάλλοντος

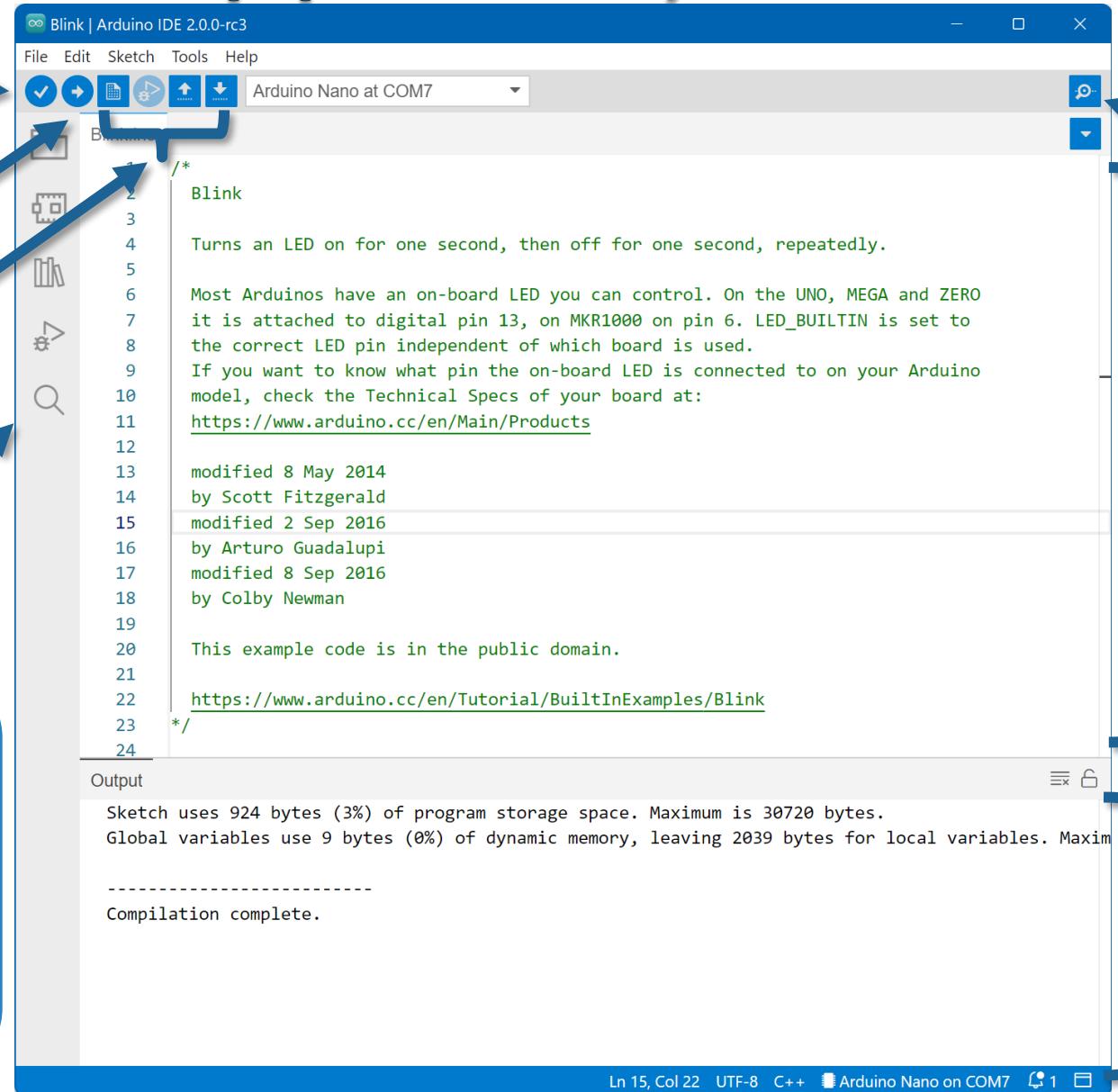
Έλεγχος κώδικα

Φόρτωση στο Arduino

Δημιουργία/Άνοιγμα/
Αποθήκευση αρχείου
και Αποσφαλμάτωση

Γραμμή συντομεύσεων

Το κάθε πρόγραμμα
γραμμένο για Arduino
ονομάζεται **Sketch**
(Σχέδιο) και έχει
κατάληξη **.ino**



Εμφάνιση
παραθύρου
σειριακής
επικοινωνίας
(Serial Monitor)

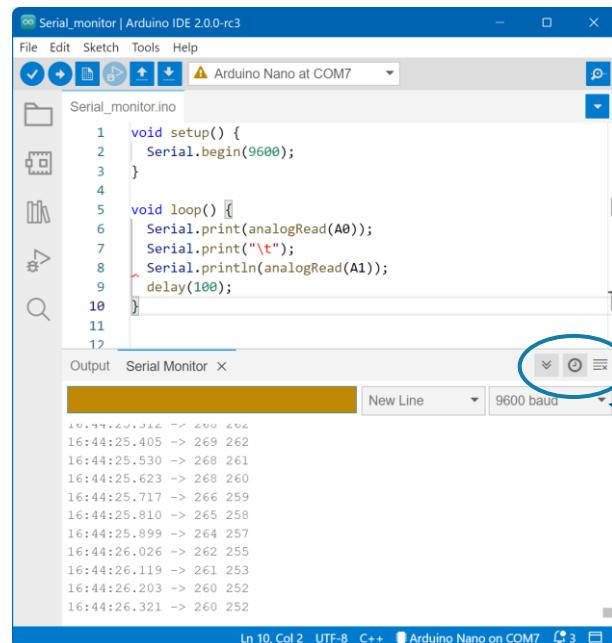
Περιοχή
συγγραφής κώδικα

Περιοχή
ειδοποίησεων/
πληροφοριών
σφαλμάτων

Σειριακή επικοινωνία

Επιτρέπει την ανταλλαγή δεδομένων από και προς το Arduino.

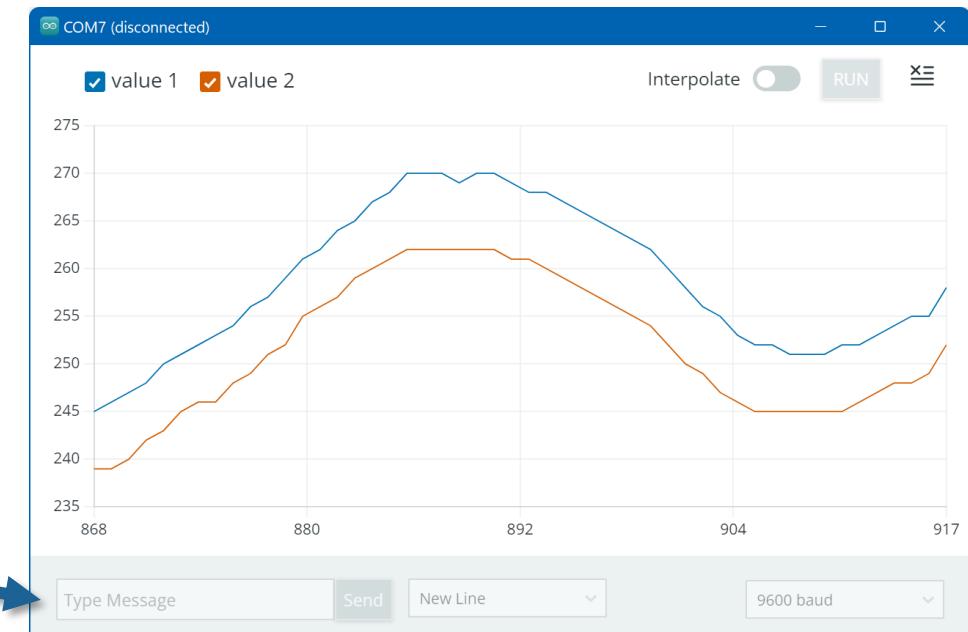
Ο ρυθμός μετάδοσης (baud rate) που θα ορίσουμε στον κώδικα μέσω της `Serial.begin()` πρέπει να είναι ο ίδιος με αυτόν που θα ορίσουμε στην εφαρμογή επικοινωνίας (π.χ. στο *Serial Monitor* του *Arduino IDE*). Διαφορετικά, το *Serial Monitor* θα εμφανίζει τυχαία σύμβολα ή θα είναι κενό. Συνήθεις τιμές είναι τα **9600** και **115200 bps**.



Αυτόματη κύλιση/
Χρονική σήμανση/
Εκκαθάριση

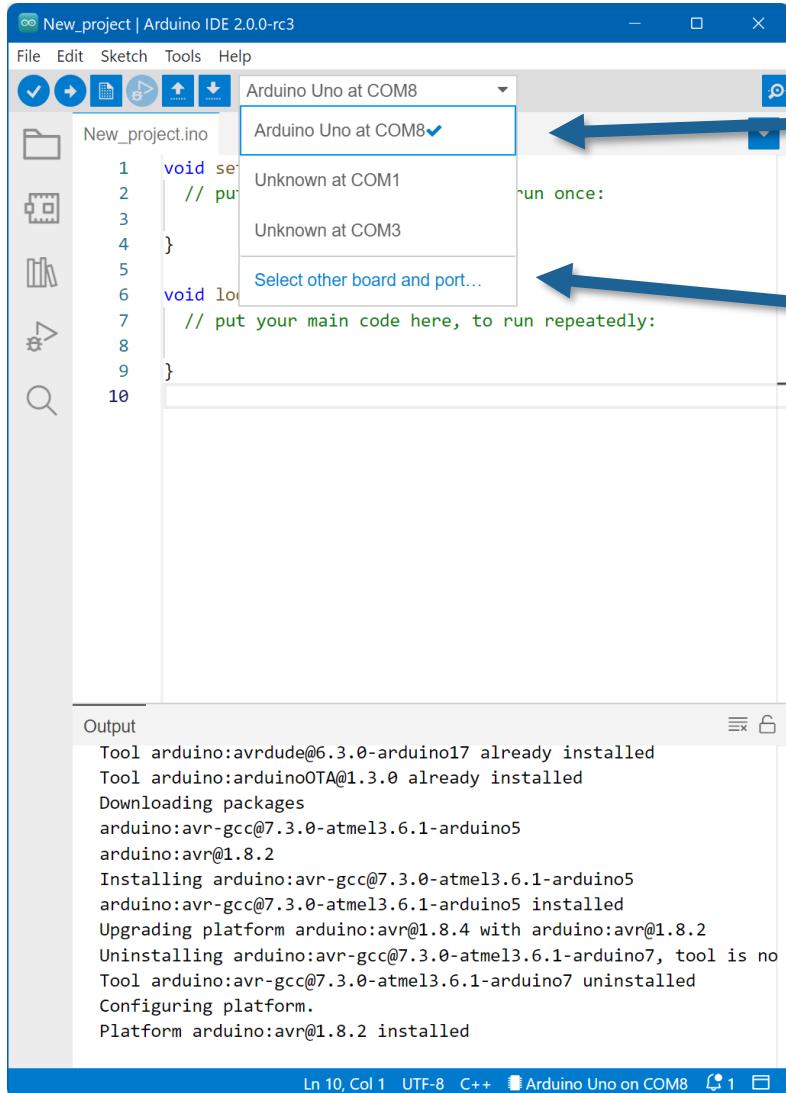
Αποστολή εντολών/
Ρυθμίσεις επικοινωνίας

Serial Monitor (Ctrl+Shift+M)



Serial Plotter

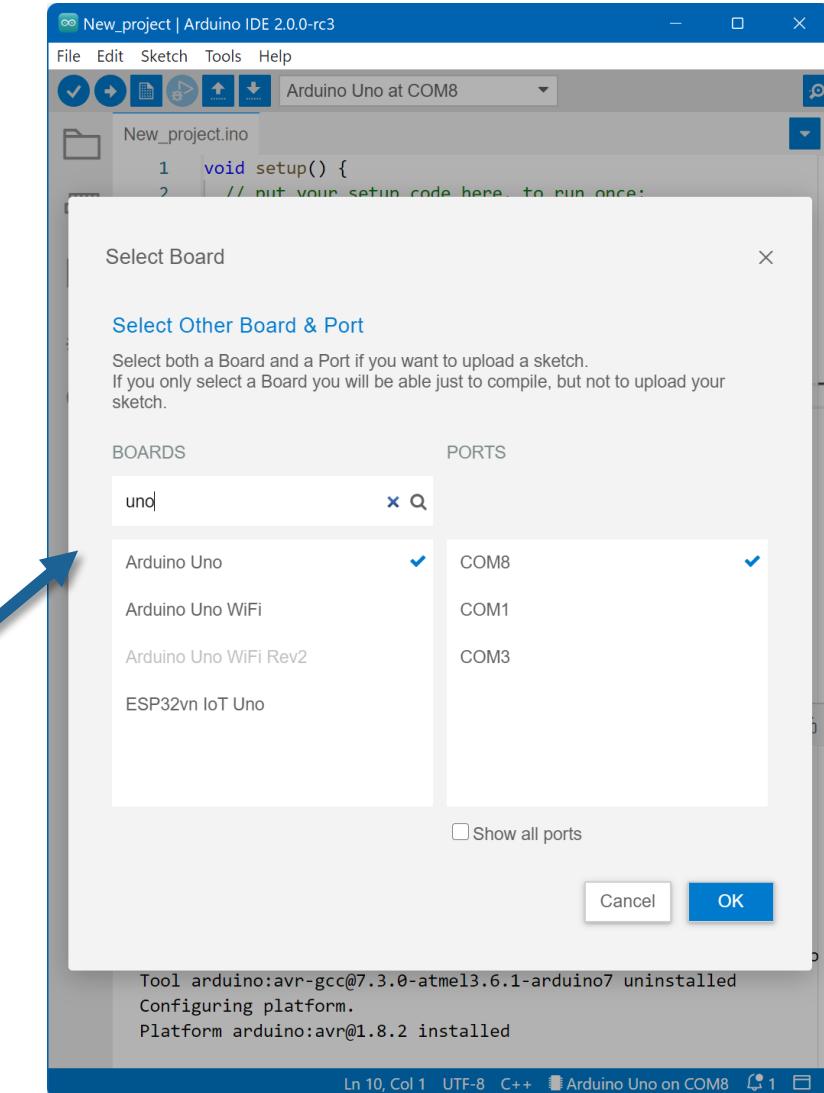
Επιλογή πλακέτας & θύρας



Απευθείας επιλογή
(αν υπάρχει)

ή
*Select other board
and port...*

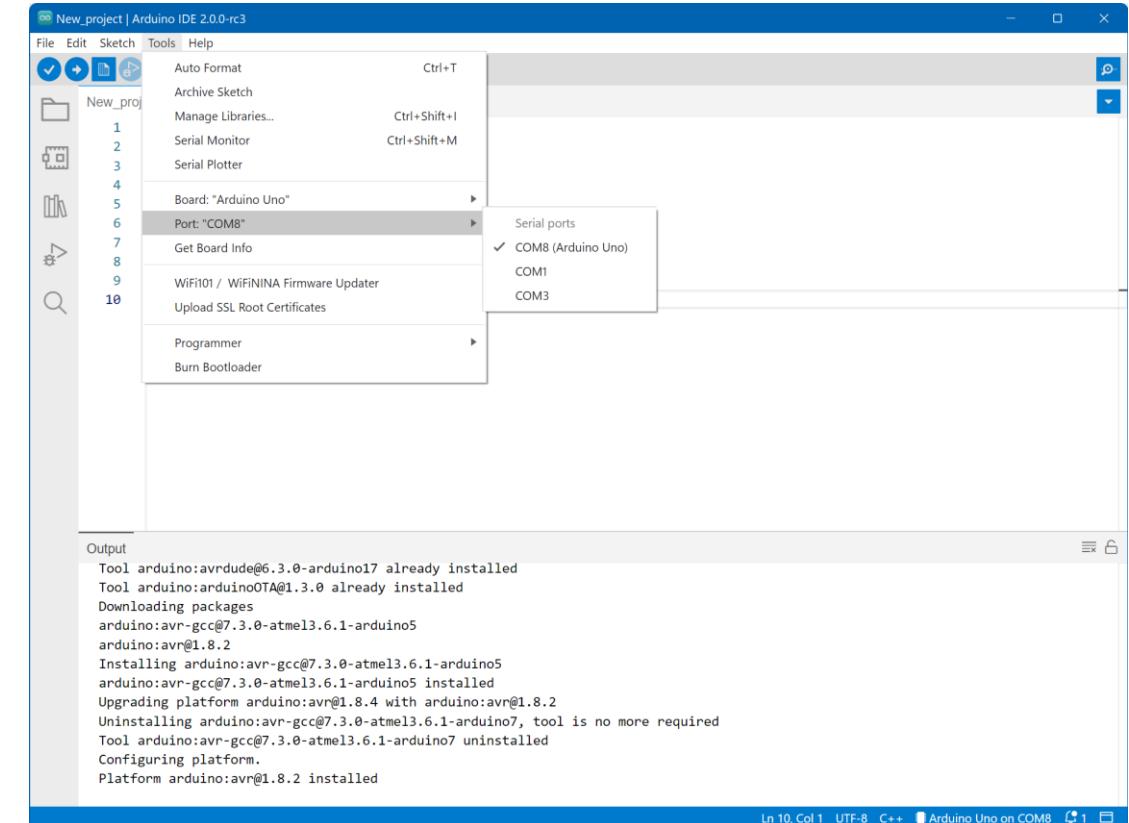
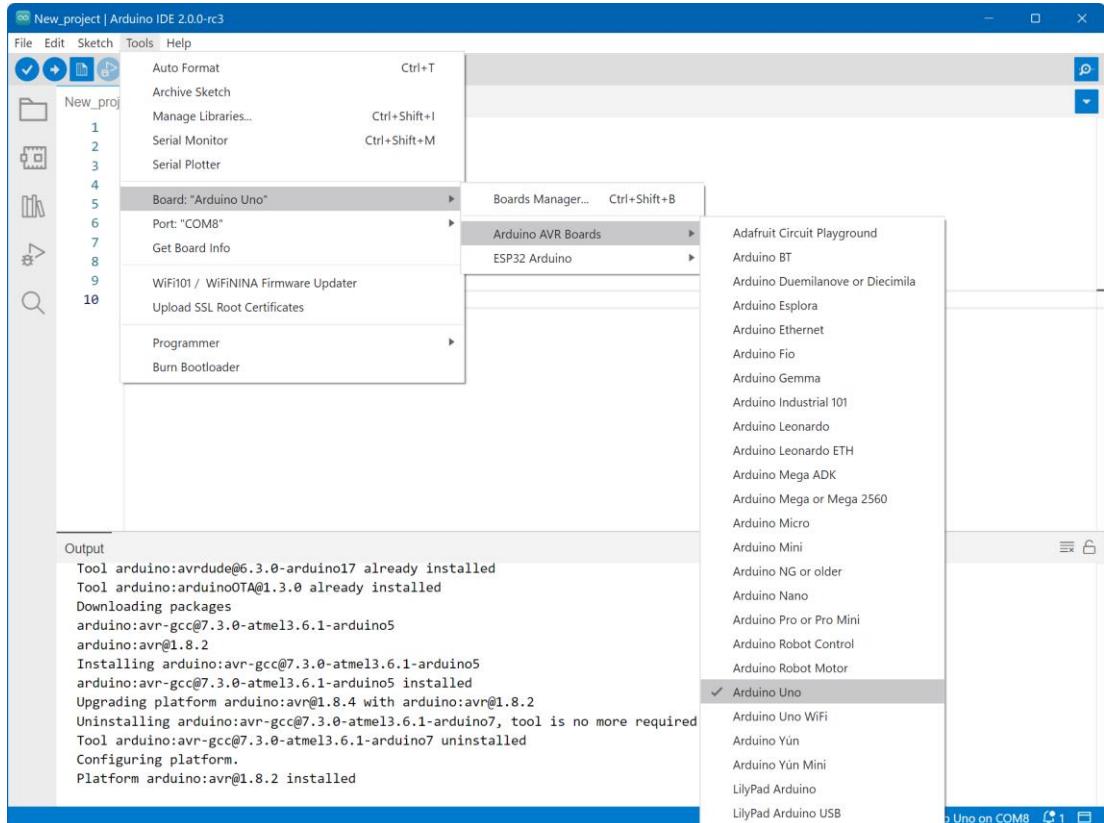
Αναζήτηση με
όνομα ή και
επιλογή από τη
λίστα



Επιλογή πλακέτας & θύρας (εναλλακτικά)

Tools → Board → Arduino AVR Boards → Arduino Uno

Tools → Port → COM...



Επιπρόσθετο βήμα για Linux (εντολές στο Terminal):

```
sudo usermod -a -G dialout <username>  
sudo chmod a+rwx /dev/ttyACM0
```

Βιβλιοθήκες

Συλλογές κώδικα και συναρτήσεων που δεν περιλαμβάνονται στη βασική γλώσσα του Arduino.

Εισαγωγή βιβλιοθήκης:

- Από τον **Library Manager**:

Arduino IDE → Sketch → Include Library → Manage Libraries...

ή *Ctrl + Shift + L* ή  στην γραμμή συντομεύσεων

- Από αρχείο **.ZIP**:

Arduino IDE → Sketch → Include Library → Add .ZIP Library...

Συμπερίληψη βιβλιοθήκης:

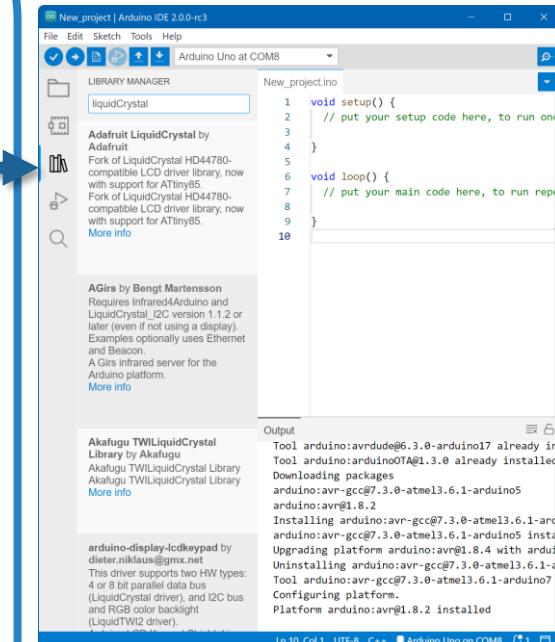
- Από το μενού:

Arduino IDE → Sketch → Include Library → Όνομα βιβλιοθήκης

- Απευθείας:

`#include <Όνομα βιβλιοθήκης.h>`

π.χ. `#include <LiquidCrystal.h>` (χωρίς ερωτηματικό στο τέλος)





Στοιχεία της γλώσσας

Δομή κώδικα (1/2)

Βασικές συναρτήσεις:

void setup ():

Κώδικας που εκτελείται μόνο μια φορά, κατά την εκκίνηση.

void loop ():

Κώδικας που εκτελείται συνεχώς, όσο τροφοδοτείται το Arduino.

```

New_project | Arduino IDE 2.0.0-rc3
File Edit Sketch Tools Help
Arduino Uno at COM8
New_project.ino
void setup() {
  // put your setup code here, to run once:
}
void loop() {
  // put your main code here, to run repeatedly:
}

```

- Για να απενεργοποιηθεί το Arduino, θα πρέπει να διακοπεί η τροφοδοσία του.
- Η επανεκκίνηση της εκτέλεσης του κώδικα μπορεί να πραγματοποιηθεί με το πλήκτρο Reset.
- Ο κώδικας παραμένει στη μνήμη flash του Arduino, ακόμη και μετά την απενεργοποίησή του.
- Οι τιμές των μεταβλητών χάνονται με την απενεργοποίηση ή επανεκκίνηση, εκτός εάν αποθηκευτούν στη μνήμη ROM ή σε κάποια εξωτερική μνήμη.

Δομή κώδικα (2/2)

Σχόλια:

Περιοχή του κώδικα που αγνοείται κατά την εκτέλεση.

Ορίζονται με:

- `/* & */` (για πολλές γραμμές)
- `//` (για 1 γραμμή)

Δήλωση μεταβλητής:

- **Τύπος μεταβλητής**
- **Όνομα μεταβλητής**
- **Αρχική τιμή μεταβλητής**

Κλήση συναρτήσεων:

- **Όνομα συνάρτησης**
- **Παράμετροι συνάρτησης**

Προσοχή στο ";" στο τέλος των γραμμών!

```

/*
Blink

Turns an LED on for one second, then off for one second, repeatedly.

modified 8 May 2014 by Scott Fitzgerald
modified 2 Sep 2016 by Arturo Guadalupi
modified 8 Sep 2016 by Colby Newman

This example code is in the public domain
http://www.arduino.cc/en/Tutorial/Blink

*/
// Define LED pin as pin D13.
int led = 13;
// the setup function runs once when you press reset or power the board
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(led, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(led, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(led, LOW);       // turn the LED off by making the voltage LOW
    delay(1000);                // wait for a second
}

```

Βασικοί τύποι μεταβλητών

Όνομα	Τύπος	Μέγεθος (bits)	Τιμές
boolean	λογική σχέση	8	HIGH / LOW (5 V / 0 V)
char	χαρακτήρας	8	-128 έως 127 (ASCII)
byte	απόλυτος ακέραιος	8	0 έως 255
int	ακέραιος	16	-32,768 έως 32,767
word	απόλυτος ακέραιος	16	0 έως 65,535
long	ακέραιος	32	-2,147,483,648 έως 2,147,483,647
unsigned long	απόλυτος ακέραιος	32	0 έως 4,294,967,295
float	δεκαδικός	32	-3.4028235E38 έως 3.4028235E38

Βασικές συναρτήσεις

Συνάρτηση	Παράδειγμα	Λειτουργία
pinMode(pin, mode)	pinMode(13,OUTPUT)	Ορίζει τη λειτουργία ενός ψηφιακού pin σε Είσοδο ή Έξοδο
digitalRead(pin)	digitalRead(6)	Διαβάζει την τιμή (HIGH ή LOW) ενός ψηφιακού pin
digitalWrite(pin, value)	digitalWrite(13, HIGH)	Εκχωρεί τιμή (HIGH ή LOW) σε ένα ψηφιακό pin
analogRead(pin)	analogRead(A0)	Διαβάζει την τιμή (0-1023 για το UNO) ενός αναλογικού pin
analogWrite(pin, value)	analogWrite(9, 127)	Εκχωρεί τιμή (0-255 για το UNO) σε ένα ψηφιακό PWM pin
delay(ms)	delay(1000)	Παύει την εκτέλεση του κώδικα για x milliseconds
millis() micros()	millis() micros()	Εμφανίζει το χρονικό διάστημα από την εκκίνηση της εκτέλεσης του προγράμματος, σε ms ή μs, αντιστοίχως
Serial.begin(bps)	Serial.begin(9600)	Ορίζει τον ρυθμό μετάδοσης (σε bits per second) των δεδομένων κατά τη σειριακή επικοινωνία
Serial.print(variable) ή Serial.println(variable)	Serial.print(temp) ή Serial.println(temp)	Εμφανίζει την τιμή της μεταβλητής μέσω της σειριακής επικοινωνίας

if, else if, else

```
if (συνθήκη1) {  
    Εκτέλεση κώδικα  
}  
else if (συνθήκη2) {  
    Εκτέλεση κώδικα  
}  
else {  
    Εκτέλεση κώδικα  
}
```



```
if (voltage >= 9.0) {  
    Serial.println("Good");  
}  
else if (voltage >= 8.0 && voltage < 9.0) {  
    Serial.println("Low battery");  
}  
else {  
    Serial.println("Replace battery!");  
}
```

while, for

while (συνθήκη) {
 Εκτέλεση κώδικα
}

π.χ.

```
int i = 0;  
while (i < 101) {  
  Serial.println(i);  
  i++;  
}
```

for (αρχική τιμή; συνθήκη; αύξηση/μείωση) {
 Εκτέλεση κώδικα
}

π.χ.

```
for (int i=0; i <= 255; i++) {  
  analogWrite(PWMpin, i);  
  delay(10);  
}
```

millis() αντί της delay()

Με την παρακάτω χρήση της συνάρτησης **millis()** αποφεύγεται η παύση του κώδικα που προκαλεί η συνάρτηση **delay()**.

```
unsigned long previousMillis = 0;
unsigned long interval = 1000;
```

```
void loop() {
...
    unsigned long currentMillis = millis();
    if (currentMillis - previousMillis >= interval) {
        Εκτέλεση κώδικα
        previousMillis = currentMillis;
    }
...
}
```

Την πρώτη φορά, έστω:

currentMillis = 2 ms

$2 - 0 < 1000$ **Άρα:**

Το τμήμα παρακάμπτεται

Μετά από μερικούς κύκλους:

currentMillis = 1002 ms

$1002 - 0 > 1000$ **Άρα:**

Εκτέλεση κώδικα και:

previousMillis = 1002 ms

π.χ.

Επόμενος κύκλος:

currentMillis = 1004 ms

$1004 - 1002 < 1000$ **Άρα:**

Το τμήμα παρακάμπτεται

...



Παραδείγματα

Παράδειγμα 1: Blink

Arduino IDE → File → Examples → 01.Basics → Blink

Πρόγραμμα που ανάβει και σθήνει το ενσωματωμένο LED του Arduino, ανά 1 δευτερόλεπτο

Ορισμός του pin που αντιστοιχεί στο ενσωματωμένο LED (pin D13), ως pin εξόδου

Ενεργοποίηση του pin D13

Παραμονή στην ίδια κατάσταση για 1 s (1000 ms)

Απενεργοποίηση του pin D13

Παραμονή στην ίδια κατάσταση για 1 s

Blink

modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016
by Arturo Guadalupi
modified 8 Sep 2016
by Colby Newman

This example code is in the public domain.

<http://www.arduino.cc/en/Tutorial/Blink>

*/

// the setup function runs once when you press reset or power the board

```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}
```

// the loop function runs over and over again forever

```
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on (HIGH is the voltage level)
  delay(1000);                         // wait for a second
  digitalWrite(LED_BUILTIN, LOW);        // turn the LED off by making the voltage LOW
  delay(1000);                         // wait for a second
}
```

Σχόλια, περιγραφή της λειτουργίας κ.λπ.

setup(): Εκτέλεση μόνο κατά την έναρξη

loop(): Επανάληψη του βρόχου συνεχώς

Παράδειγμα 1: Blink

1. Τροποποιήστε τον κώδικα ώστε το LED να μένει αναμμένο για 1.5 s και σβηστό για 0.5 s.
2. Τροποποιήστε τον κώδικα ώστε το LED να εκπέμπει SOS σε σήματα Morse βάσει των παρακάτω:

S	...
O	---
Παύση μεταξύ των συμβόλων (, ή -)	.
Παύση μεταξύ των γραμμάτων	-

όπου η τελεία (.) αντιστοιχεί σε 0.5 s και η παύλα (-) σε 1.5 s.

Παράδειγμα 2: Έλεγχος LED μέσω σειριακής επικοινωνίας

Arduino IDE → File → Examples → 04.Communication → PhysicalPixel

Πρόγραμμα που ανάβει ένα LED
μέσω σειριακής εντολής

Ορισμός μιας σταθεράς που αντιστοιχεί στο LED (pin D13)

const: Ορισμός σταθεράς (η τιμή που εκχωρείται δεν μπορεί να αλλάξει μέσα στο πρόγραμμα)

Ορισμός μιας μεταβλητής τύπου int

Έναρξη σειριακής επικοινωνίας στα 9600 bps

Ορισμός του pin D13 ως pin εξόδου

Εάν υπάρχουν δεδομένα από τη σειριακή επικοινωνία...

...να εκχωρηθούν στη μεταβλητή incomingByte και...

...εάν η τιμή είναι H, να ανάψει το LED

...εάν η τιμή είναι L, να σβήσει το LED

```
const int ledPin = 13; // the pin that the LED is connected to
int incomingByte; // a variable to read incoming bytes

void setup() {
  // initialize serial communication:
  Serial.begin(9600);
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // see if there's incoming serial data:
  if (Serial.available() > 0) {
    // read the oldest byte in the serial buffer:
    incomingByte = Serial.read();
    // if it's a capital H (ASCII 72), turn on the LED:
    if (incomingByte == 'H') {
      digitalWrite(ledPin, HIGH);
    }
    // if it's an L (ASCII 76) turn off the LED:
    if (incomingByte == 'L') {
      digitalWrite(ledPin, LOW);
    }
  }
}
```

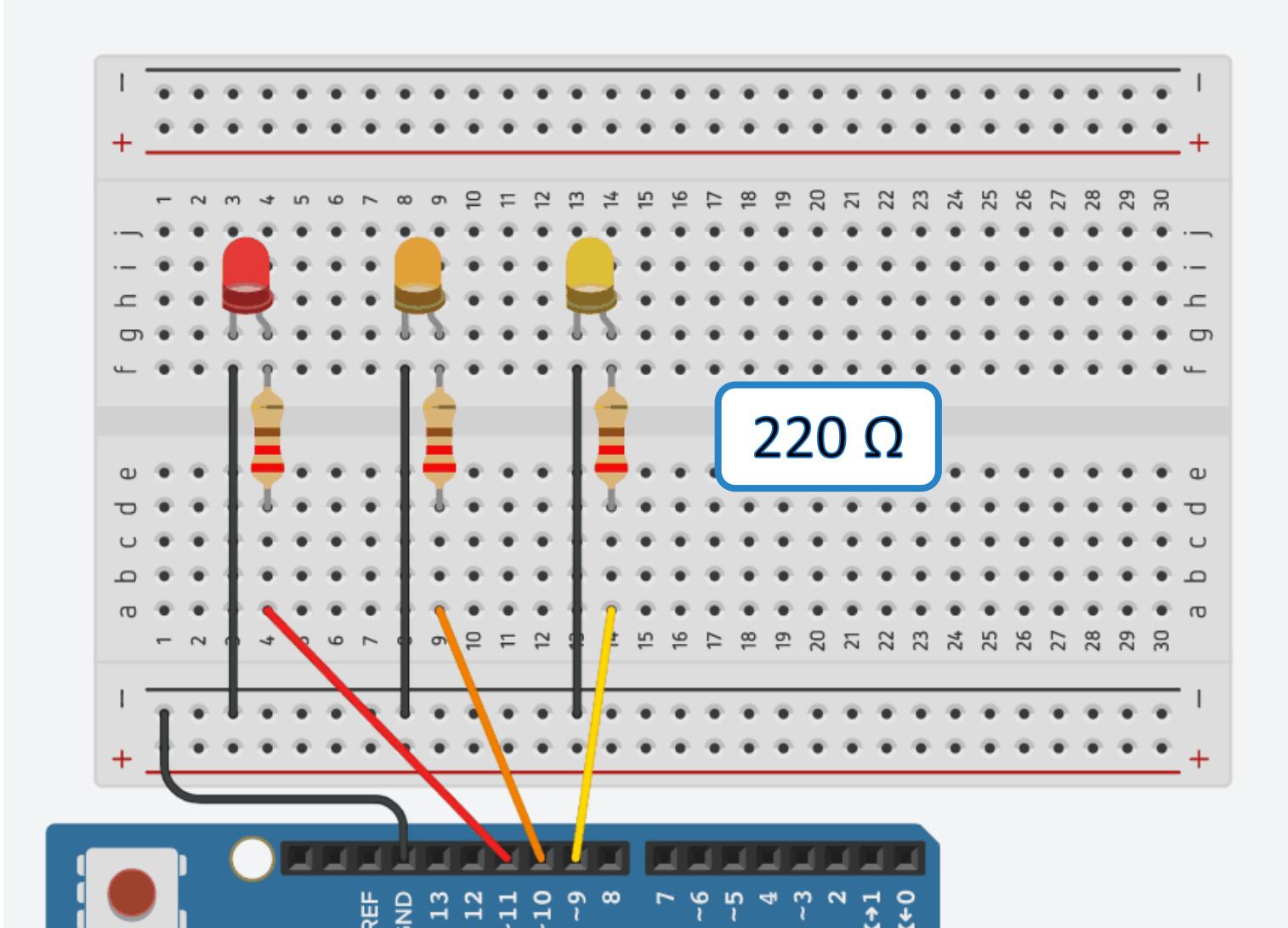
Παράδειγμα 2: Έλεγχος LED μέσω σειριακής επικοινωνίας

Τροποποιήστε τον κώδικα ώστε:

- Το LED να ενεργοποιείται με τους χαρακτήρες **H** ή **h** και να απενεργοποιείται με τους χαρακτήρες **L** ή **l**.
- Να εμφανίζεται ένα μήνυμα σφάλματος εάν πληκτρολογηθεί οποιοσδήποτε άλλος χαρακτήρας.

Παράδειγμα 3: Προσομοίωση φλόγας

Πρόγραμμα που προσομοιώνει το φως μιας φωτιάς, χρησιμοποιώντας LEDs



Παράδειγμα 3: Προσομοίωση φλόγας

Ορισμός σταθερών που αντιστοιχούν στα pins τριών LEDs (προσοχή: πρέπει να είναι PWM pins)

Ορισμός των pins ως pins εξόδου

randomSeed(είσοδος): Ορισμός πηγής τυχαίων αριθμών (π.χ. από μία αναλογική είσοδο).
random(min, max): Παραγωγή τυχαίας τιμής στο εύρος (min, max).

Λήψη τυχαίων αριθμών βάσει της εισόδου A0

Εκχώρηση τυχαίου duty cycle (δηλ. τυχαίας φωτεινότητας) σε κάθε LED

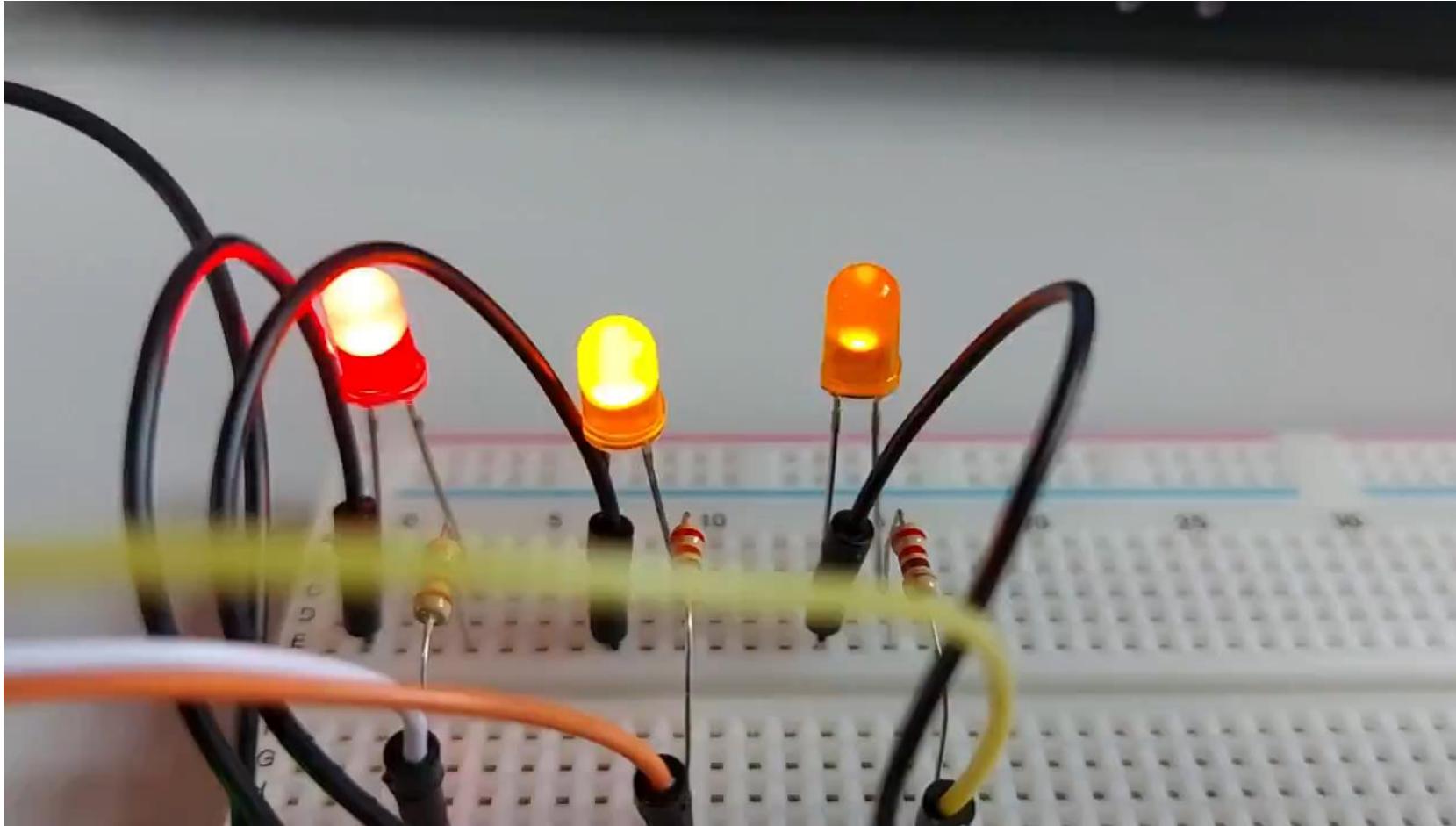
Ορισμός τυχαίας τιμής καθυστέρησης

```
int ledPin1 = 9;      //Pin 1ou LED
int ledPin2 = 10;     //Pin 2ou LED
int ledPin3 = 11;     //Pin 3ou LED

void setup()
{
    pinMode(ledPin1, OUTPUT);          //Ορισμός ως Ρ
    pinMode(ledPin2, OUTPUT);          //Ορισμός ως Ρ
    pinMode(ledPin3, OUTPUT);          //Ορισμός ως Ρ
    randomSeed(analogRead(A0));       //Λήψη τυχαίων
}

void loop()
{
    analogWrite(ledPin1, random(0, 130) + 125);
    analogWrite(ledPin2, random(0, 130) + 125);
    analogWrite(ledPin3, random(0, 130) + 125);
    delay(random(0, 120));
}
```

Παράδειγμα 3: Προσομοίωση φλόγας

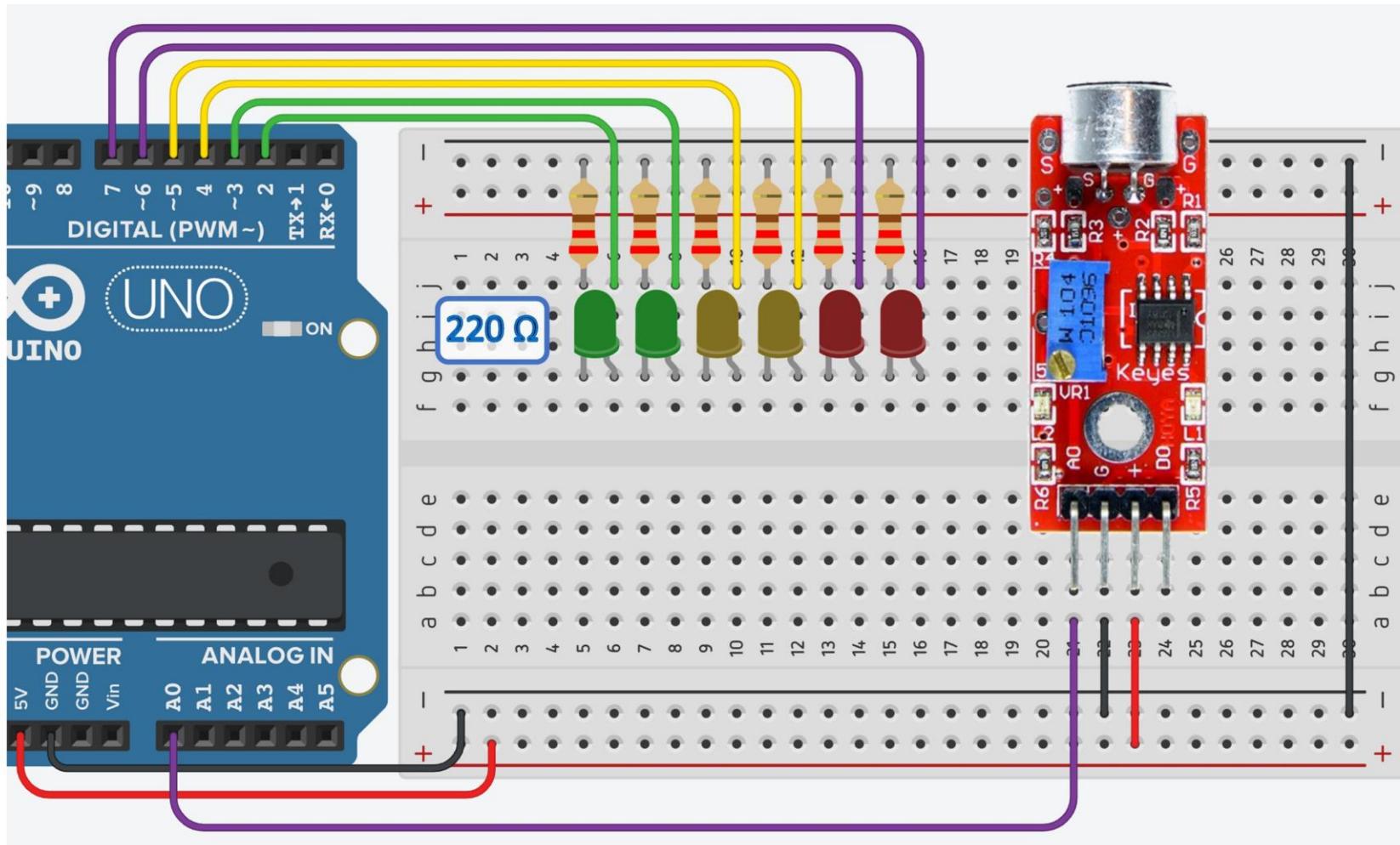


Παράδειγμα 3: Προσομοίωση φλόγας

1. Τροποποιήστε τον κώδικα ώστε να μην αναγράφονται 3 φορές οι συναρτήσεις `pinMode()` και `analogWrite()`.
2. Τροποποιήστε τον κώδικα ώστε να διαχειρίζεται 6 LEDs.

Παράδειγμα 4: Μικρόφωνο και LEDs

Πρόγραμμα που λαμβάνει το σήμα ενός μικροφώνου και εμφανίζει την ένταση μέσω LEDs.



Παράδειγμα 4: Μικρόφωνο και LEDs

```

const byte micPin = A0;
int led_index;
int mic_offset = 50;
int mic;

int led[6] = {2, 3, 4, 5, 6, 7};
int led_number = sizeof(led) / sizeof(led[0]);

void setup() {
    for (int i = 0; i < led_number; i++) {
        pinMode(led[i], OUTPUT);
    }
    Serial.begin(9600);
}
  
```

Ορισμός ψηφιακών εξόδων
με χρήση for και έναρξη
σειριακής επικοινωνίας

Ενεργοποίηση των LEDs που
αντιστοιχούν στην τρέχουσα ένταση
Απενεργοποίηση των υπολοίπων

Απενεργοποίηση όλων των LEDs όταν η
ένταση είναι χαμηλή

} Ορισμός μεταβλητών

← Ορισμός διανύσματος εξόδων των LEDs
← Υπολογισμός μεγέθους διανύσματος

Ανάγνωση τιμής
μικροφώνου

Έλεγχος για αγνόηση
του θορύβου

Αντιστοίχιση
έντασης με
θέση LED

```

void loop() {
    mic = analogRead(micPin);
    Serial.println(mic);
    if (mic > mic_offset) {
        led_index = map(mic, mic_offset, 1023, 0, led_number - 1);
    }
  
```

```

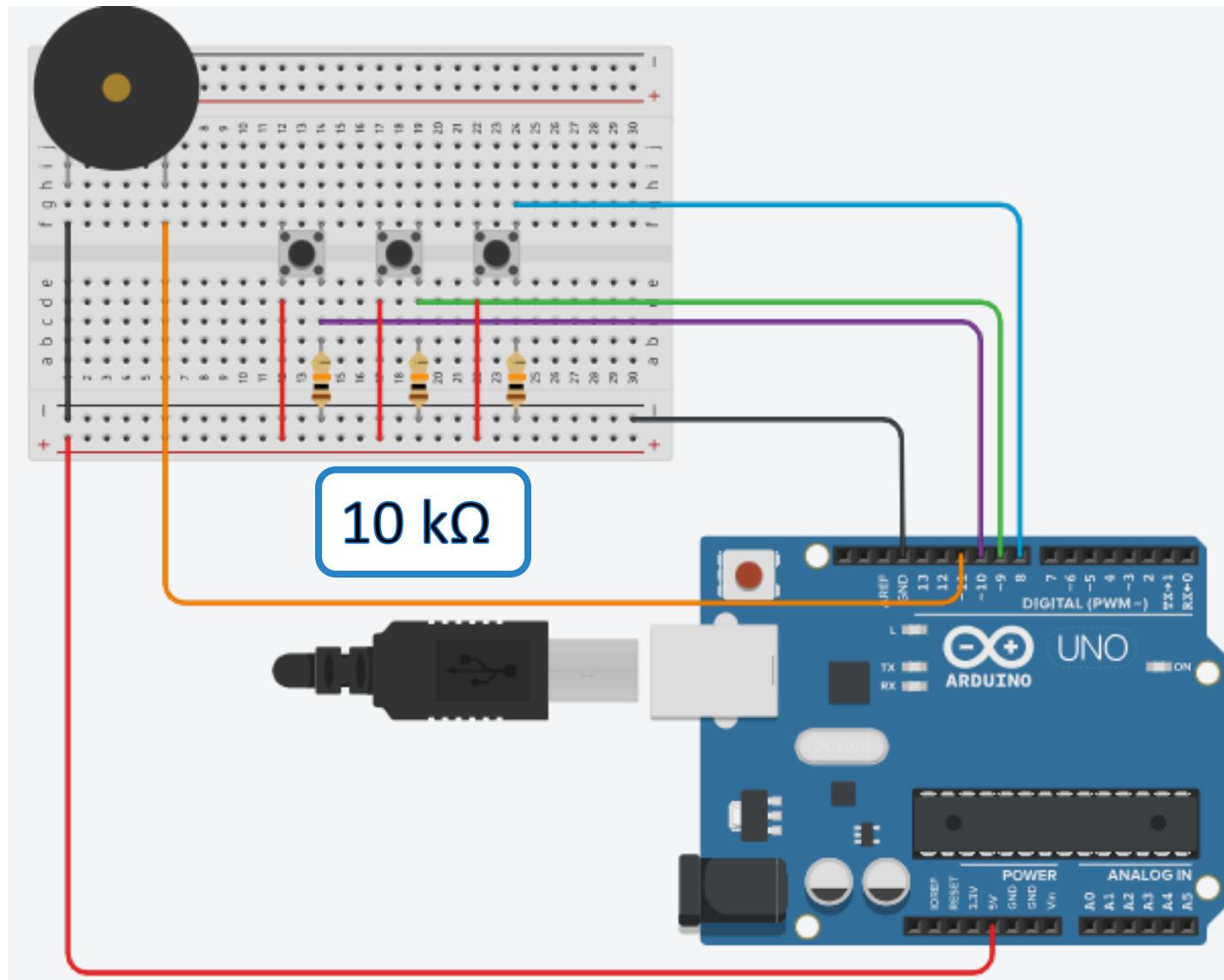
    for (int i = 0; i <= led_index; i++) {
        digitalWrite(led[i], HIGH);
    }
    for (int i = led_index + 1; i < led_number; i++) {
        digitalWrite(led[i], LOW);
    }
} else {
    for (int i = 0; i < led_number; i++) {
        digitalWrite(led[i], LOW);
    }
}
  
```

Παράδειγμα 4: Μικρόφωνο και LEDs

1. Προσαρμόστε το **trimpot** ώστε οριακά να μην ανάβει το 1^ο LED.
2. Εξηγήστε τον τρόπο λειτουργίας του κώδικα.

Παράδειγμα 5: Piano

Πρόγραμμα που αναπαράγει 3 νότες με το πάτημα αντίστοιχων πλήκτρων





Παράδειγμα 5: Piano

```

int speaker_pin = 11;
int Re_pin = 10;
int Fa_pin = 9;
int Sol_pin = 8;

int notes[] = {294, 349, 392}; ← Ορισμός διανύσματος με στοιχεία 3 νότες (συχνότητες)

void setup() {
    pinMode(Re_pin, INPUT);
    pinMode(Fa_pin, INPUT);
    pinMode(Sol_pin, INPUT);
    pinMode(speaker_pin, OUTPUT);
}
  
```

Όσο είναι πατημένο το αντίστοιχο κουμπί...

...να αναπαράγεται η αντίστοιχη νότα από το ηχείο, για 200 ms

tone(pin, συχνότητα, διάρκεια): Συνάρτηση για αναπαραγωγή ήχου

Ορισμός των pins

Ορισμός διανύσματος με στοιχεία 3 νότες (συχνότητες)

Ορισμός των pins ως εισόδων και εξόδου

```

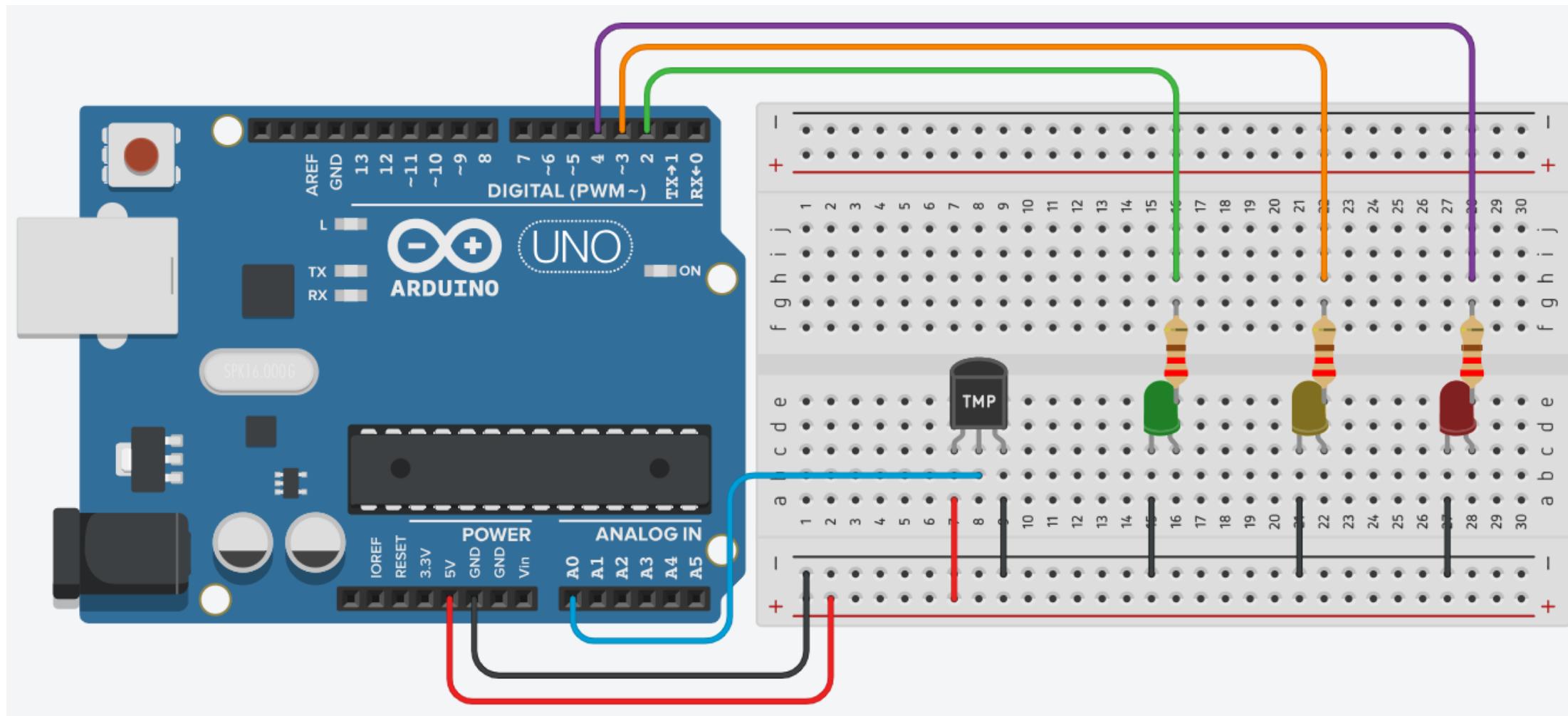
void loop() {
    while (digitalRead(Re_pin) == HIGH)
    {
        tone(speaker_pin, notes[0], 200);
    }
    while (digitalRead(Fa_pin) == HIGH)
    {
        tone(speaker_pin, notes[1], 200);
    }
    while (digitalRead(Sol_pin) == HIGH)
    {
        tone(speaker_pin, notes[2], 200);
    }
}
  
```

Παράδειγμα 5: Piano

1. Αφαιρέστε τις αντιστάσεις των $10\text{ k}\Omega$.
2. Τι πρόβλημα προκύπτει; Πώς το αντιμετωπίζουν οι συγκεκριμένες αντιστάσεις;

Παράδειγμα 6: Αισθητήρας Θερμοκρασίας TMP36

Πρόγραμμα που διαβάζει την έξοδο ενός αισθητήρα θερμοκρασίας TMP36 και ανάλογα με τη θερμοκρασία, ανάβει διαφορετικό LED



Παράδειγμα 6: Αισθητήρας Θερμοκρασίας TMP36

- Στο datasheet του αισθητήρα Θερμοκρασίας **TMP36** αναγράφεται ότι η σχέση τάσης εξόδου και θερμοκρασίας είναι **10 mV/°C**, με ένα offset **500 mV**.
- Η ανάλυση του Analog-to-Digital Converter (ADC) του Arduino UNO είναι **10 bits**, δηλ. $2^{10} = 1024$ samples.
- Επομένως, η μετρούμενη τάση σε **mV** θα είναι:
 - Για $V_{cc} = 5 \text{ V} = 5000 \text{ mV}$: $V_{out} = \text{τιμή_εισόδου} \cdot \frac{5000}{1024}$
 - Για $V_{cc} = 3.3 \text{ V} = 3300 \text{ mV}$: $V_{out} = \text{τιμή_εισόδου} \cdot \frac{3300}{1024}$
- Τελικά, βάσει των παραπάνω σχέσεων, η **Θερμοκρασία σε °C** για τον συγκεκριμένο αισθητήρα θα είναι:

$$\text{temp_C} = \frac{V_{out} - 500}{10}$$

Παράδειγμα 6: Αισθητήρας Θερμοκρασίας TMP36

```

int green_pin = 2;
int orange_pin = 3;
int red_pin = 4;
int tempPin = A0;

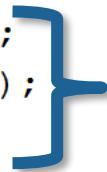
void setup()
{
    pinMode(green_pin, OUTPUT);
    pinMode(orange_pin, OUTPUT);
    pinMode(red_pin, OUTPUT);
    Serial.begin(9600);
}

```



Ορισμός των pins

Ανάγνωση, υπολογισμός και εμφάνιση τιμής θερμοκρασίας



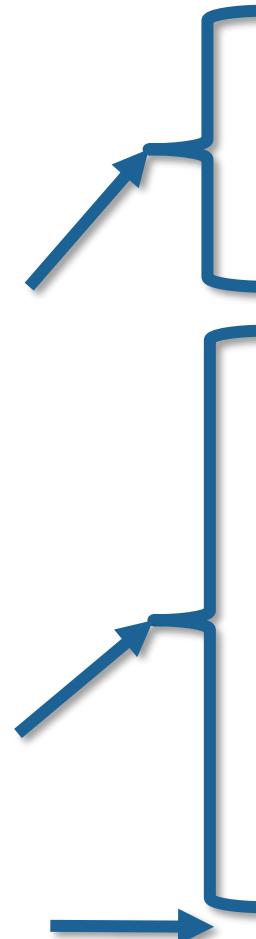
Ορισμός των pins ως εξόδου και εισόδου



Έναρξη σειριακής επικοινωνίας στα 9600 bps

Ενεργοποίηση αντίστοιχου LED, αναλόγως της θερμοκρασίας

Λήψη νέας μέτρησης κάθε 1 s



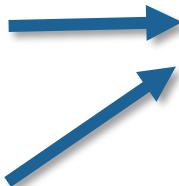
```

void loop()
{
    int tempValue = analogRead(tempPin);
    float temp = tempValue * 5000.0 / 1024.0;
    float temp_C = (temp - 500) / 10;
    Serial.println(tempValue);
    Serial.print("Temp = ");
    Serial.print(temp_C);
    Serial.println("C");
    if (temp_C <= 25) {
        digitalWrite(green_pin, HIGH);
        digitalWrite(orange_pin, LOW);
        digitalWrite(red_pin, LOW);
    }
    else if (temp_C > 25 && temp_C <= 35) {
        digitalWrite(green_pin, LOW);
        digitalWrite(orange_pin, HIGH);
        digitalWrite(red_pin, LOW);
    }
    else {
        digitalWrite(green_pin, LOW);
        digitalWrite(orange_pin, LOW);
        digitalWrite(red_pin, HIGH);
    }
    delay(1000);
}

```

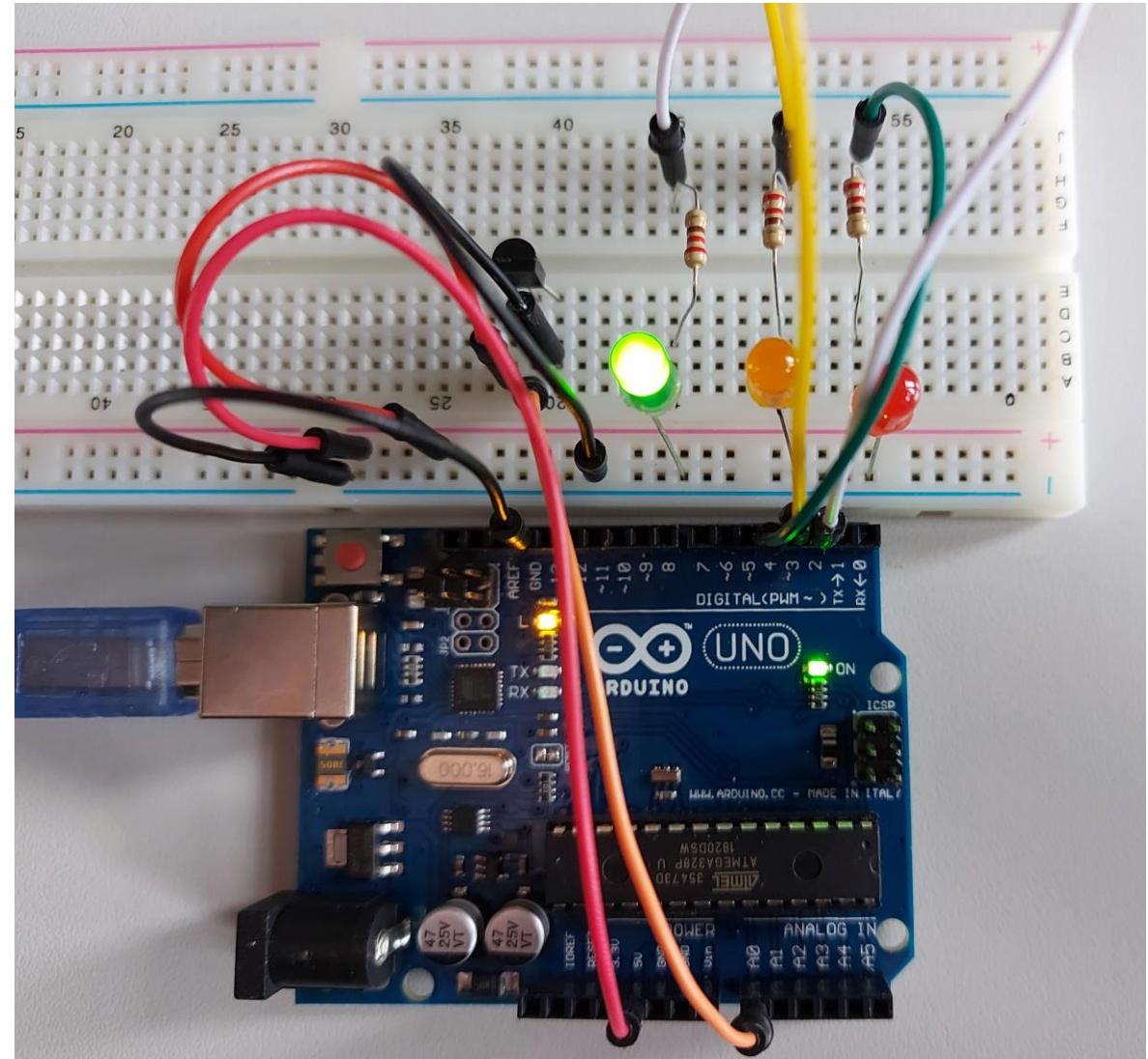
Παράδειγμα 6: Αισθητήρας Θερμοκρασίας TMP36

Ληφθείσα τιμή
Υπολογισμός
τιμής σε °C



```
Temp = 19.82C  
143  
Temp = 19.82C  
143
```

Serial Monitor



Παράδειγμα 6: Αισθητήρας Θερμοκρασίας TMP36

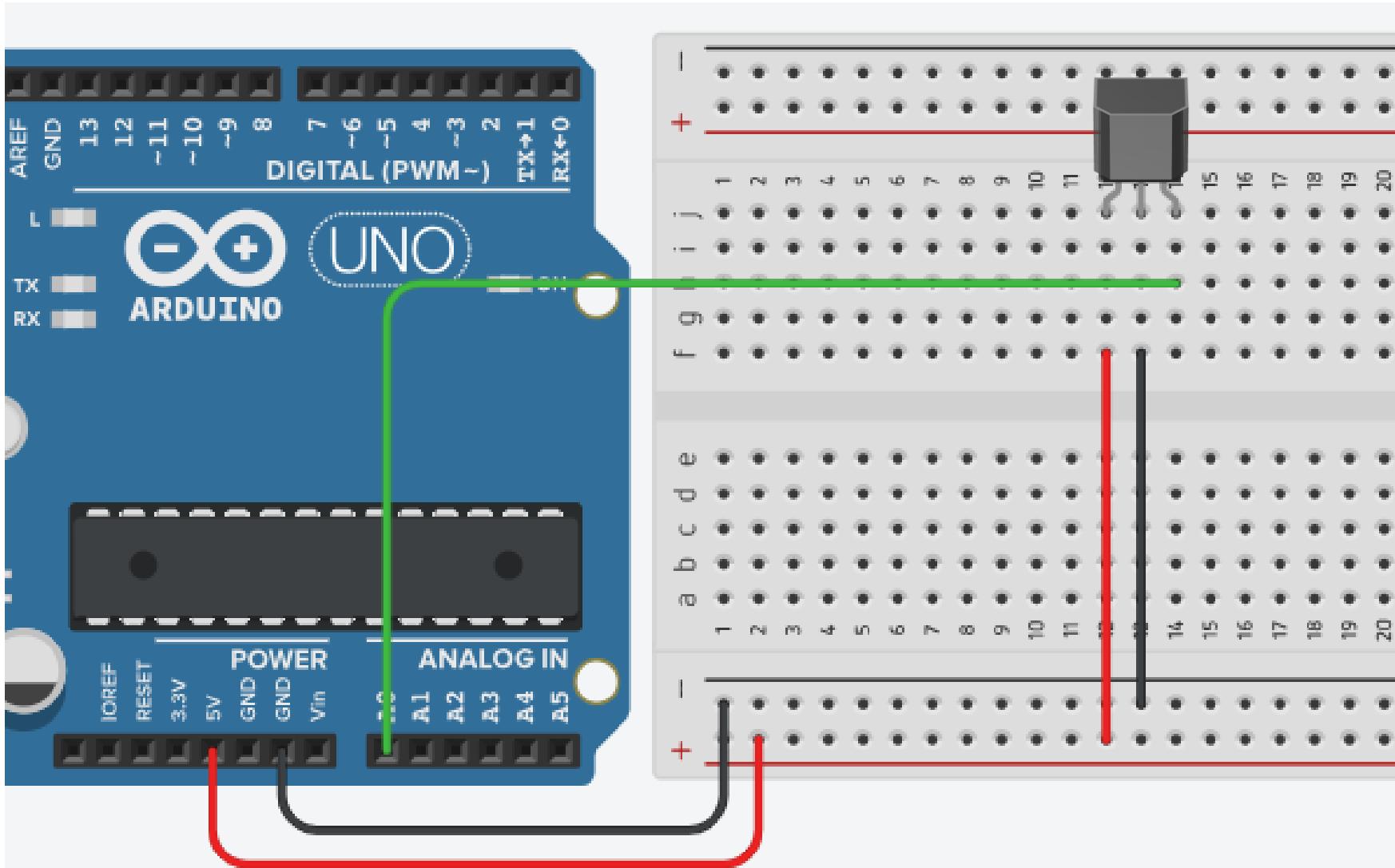
- Τροποποιήστε τον κώδικα ώστε να ανάβουν διαδοχικά όλα τα LEDs αναλόγως της θερμοκρασίας, δηλαδή:

Θερμοκρασία	LEDs
$\theta \leq 25$	Πράσινο
$25 < \theta \leq 35$	Πράσινο & Πορτοκαλί
$\theta > 35$	Πράσινο & Πορτοκαλί & Κόκκινο

- Έστω ότι χρησιμοποιούμε τάση τροφοδοσίας 3.3 V και 12-bit ADC. Ποιες τροποποιήσεις πρέπει να γίνουν στον κώδικα;

Παράδειγμα 7: Hall SS49

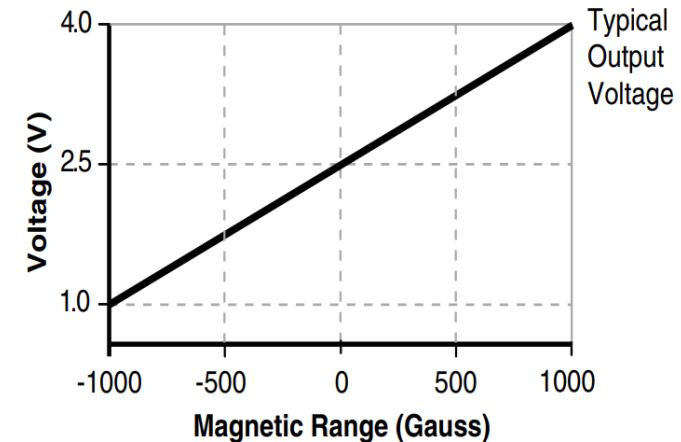
Πρόγραμμα που διαβάζει την έξοδο ενός μαγνητικού αισθητήρα Hall



Παράδειγμα 7: Hall SS49

- Στο datasheet του αισθητήρα Hall SS49 παρουσιάζεται διάγραμμα, βάσει του οποίου τα **1000 Gauss** αντιστοιχούν στα **4 V** και τα **-1000 Gauss** στο **1 V**.
- Χρησιμοποιούμε τη συνάρτηση **map()**:

map(μεταβλητή, αρχ. min, αρχ. max, τελ. min, τελ. max):
Συνάρτηση για αντιστοίχιση τιμών από ένα εύρος σε άλλο



Ορίζουμε την μεταβλητή (έστω **voltage**) από την οποία θα λαμβάνονται οι τιμές προς μετατροπή. Άρα τελικά:

map(voltage, 1, 4, -1000, 1000)

Η συνάρτηση **map() λειτουργεί μόνο με ακεραίους.**

Παράδειγμα 7: Hall SS49

```
int pin = A0;  
int vcc = 5000;  
  
int value;  
int voltage;  
int field;  
  
void setup() {  
    Serial.begin(9600);  
}
```

Έναρξη σειριακής
επικοινωνίας
στα 9600 bps

Ορισμός pin εισόδου και
τάσης τροφοδοσίας σε mV

Ορισμός μεταβλητών

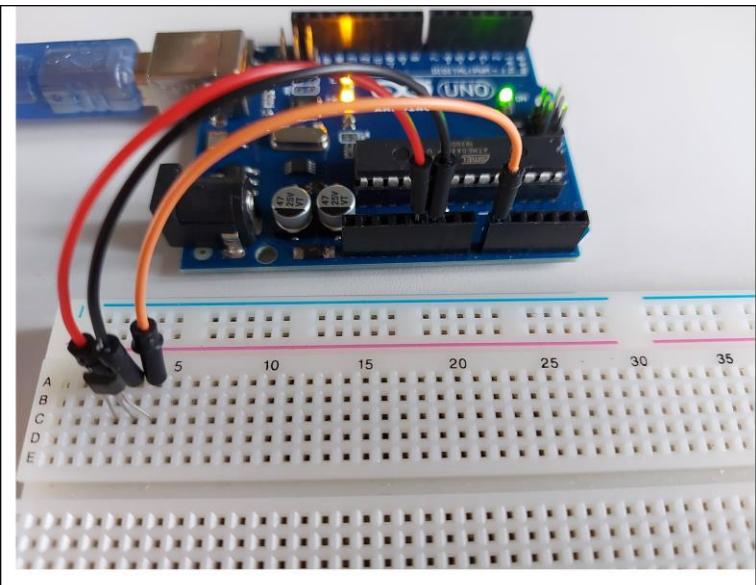
Υπολογισμός τιμής
τάσης & πεδίου

Εμφάνιση αποτελεσμάτων

Λήψη νέας μέτρησης κάθε 1 s

```
void loop() {  
    value = analogRead(pin); //Ανάγνωση τιμής του  
    voltage = value * (vcc / 1024); //Μετατροπή σ  
    field = map(voltage, 1000, 4000, -100, 100);  
  
    //Εμφάνιση αποτελεσμάτων  
    Serial.print("Value: ");  
    Serial.print(value);  
    Serial.print("\t");  
    Serial.print(voltage);  
    Serial.println(" mV");  
    Serial.print("Magnetic Field: ");  
    Serial.print(field);  
    Serial.println(" mT");  
    Serial.println("=====");  
    delay(1000); //Λήψη νέας τιμής κάθε 1sec  
}
```

Παράδειγμα 7: Hall SS49



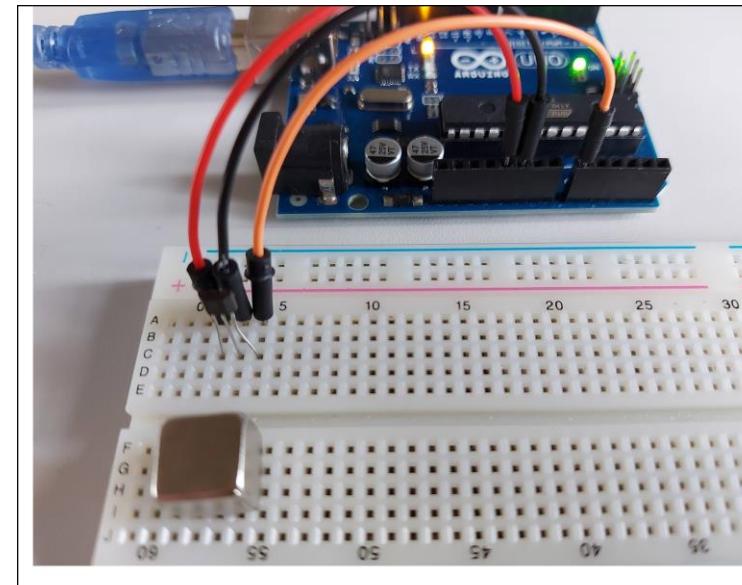
```
Value: 518      2072 mV  
Magnetic Field: -29 mT  
=====
```

```
Value: 518      2072 mV  
Magnetic Field: -29 mT  
=====
```

```
Value: 518      2072 mV  
Magnetic Field: -29 mT  
=====
```

```
Value: 517      2068 mV  
Magnetic Field: -29 mT  
=====
```

```
Value: 518      2072 mV  
Magnetic Field: -29 mT  
=====
```



```
Value: 442      1768 mV  
Magnetic Field: -49 mT  
=====
```

```
Value: 414      1656 mV  
Magnetic Field: -57 mT  
=====
```

```
Value: 412      1648 mV  
Magnetic Field: -57 mT  
=====
```

```
Value: 412      1648 mV  
Magnetic Field: -57 mT  
=====
```

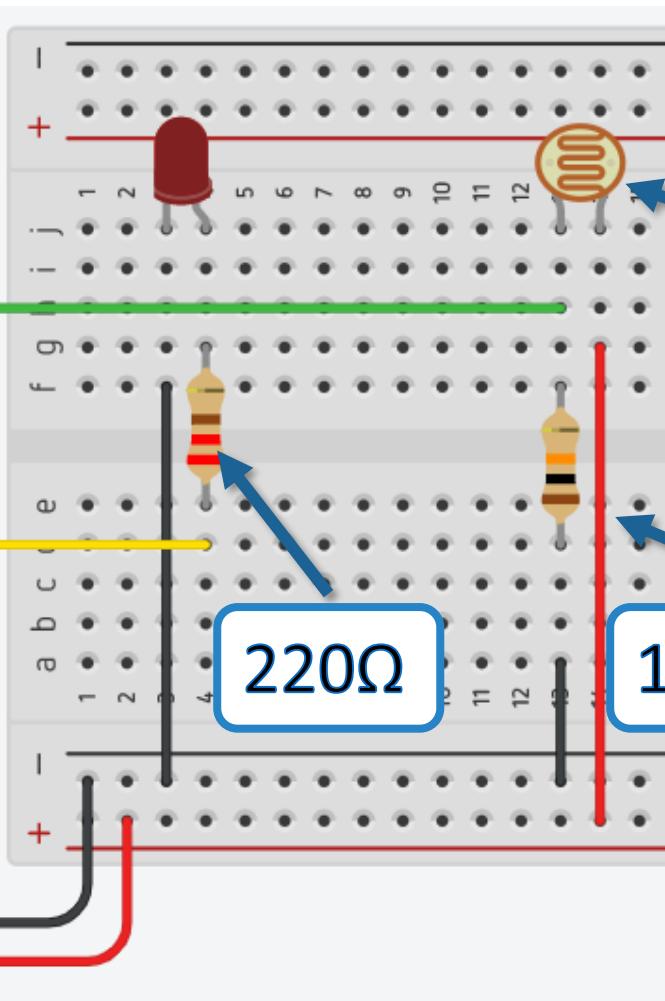
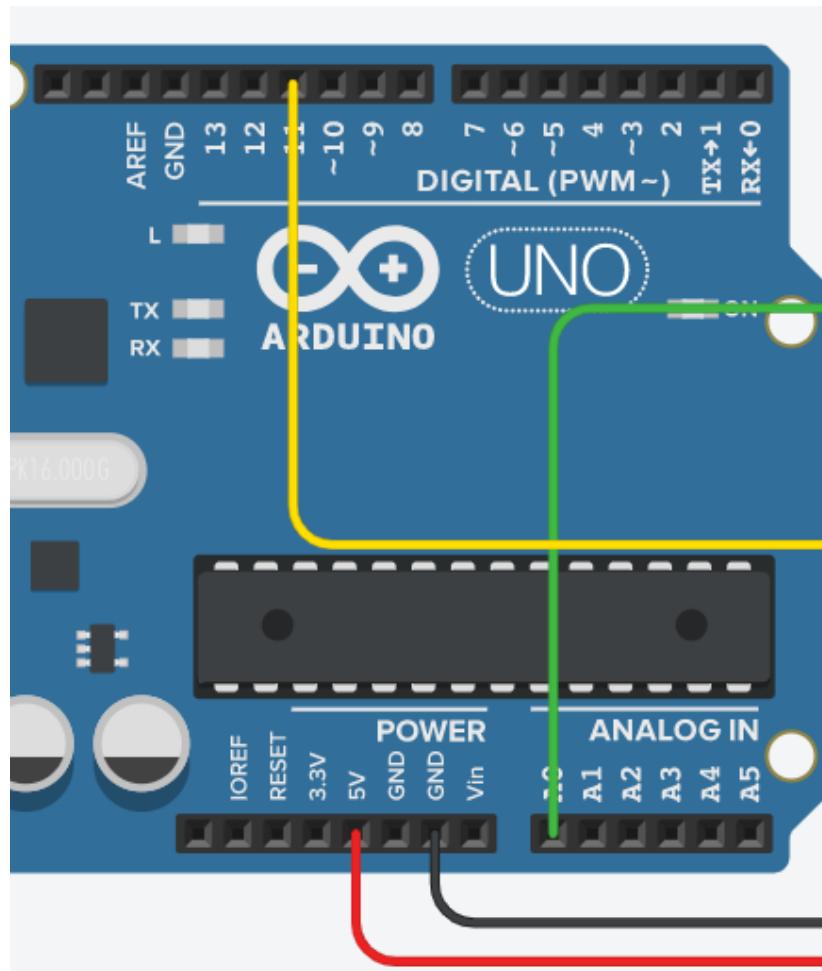
```
Value: 412      1648 mV  
Magnetic Field: -57 mT  
=====
```

Παράδειγμα 7: Hall SS49

1. Τροποποιήστε τον κώδικα ώστε κατά την απουσία του μαγνήτη να εμφανίζεται η τιμή 0 mT.
2. Τροποποιήστε το κύκλωμα και τον παραπάνω κώδικα ώστε να ενεργοποιείται ένα LED όταν η τιμή του μαγνητικού πεδίου που ανιχνεύει ο αισθητήρας Hall ξεπερνά κατ' απόλυτη τιμή την τιμή των 5 mT.
3. Αλλάξτε τη διάρκεια παύσης μεταξύ των μετρήσεων σε 100 ms. Στη συνέχεια, τροποποιήστε τον κώδικα ώστε η φωτεινότητα του LED να είναι ανάλογη του μαγνητικού πεδίου που ανιχνεύει ο αισθητήρας Hall.

Παράδειγμα 8: Αισθητήρας φωτεινότητας

Πρόγραμμα που διαβάζει ψηφιακά και αναλογικά την έξοδο μιας φωτοαντίστασης και αναλόγως ανάβει ή σβήνει ένα LED



Συνδεσμολογία
διαιρέτη τάσης



$$V_{out} = V_{CC} \frac{R_2}{R_1 + R_2}$$

Παράδειγμα 8: Αισθητήρας φωτεινότητας

```
int incomingByte;
int value;
}

void setup() {
  pinMode(11, OUTPUT);
  Serial.begin(9600);
  Serial.println("Please enter A for Analog or D for
}
```

Ορισμός του pin11 ως
pin εξόδου, έναρξη
σειριακής επικοινωνίας
στα 9600 bps και
εμφάνιση μηνύματος για
επιλογή λειτουργίας

Δήλωση
μεταβλητών

Εάν υπάρχουν δεδομένα από τη
σειριακή επικοινωνία...

enter A for Analog or D for

Εάν αποσταλεί A, να εκτελείται η
analog() μέχρι να αποσταλούν
ξανά δεδομένα

Εάν αποσταλεί D, να εκτελείται η
digital() μέχρι να αποσταλούν
ξανά δεδομένα

Άλλιως, να εμφανιστεί μήνυμα
σφάλματος

```
void loop() {
  if (Serial.available() > 0) {
    incomingByte = Serial.read();
    if (incomingByte == 'A') {
      while (Serial.available() == 0) {
        analog();
      }
    } else if (incomingByte == 'D') {
      while (Serial.available() == 0) {
        digital();
      }
    } else {
      Serial.println("Unknown command");
    }
  }
}
```

Παράδειγμα 8: Αισθητήρας φωτεινότητας

Ανάγνωση και εμφάνιση τιμής φωτοαντίστασης

Αντιστοίχιση της αναλογικής τιμής στο εύρος (0, 255) και εκχώρηση της τιμής ως duty cycle του LED

Ανάγνωση και εμφάνιση τιμής φωτοαντίστασης

Αντιστοίχιση της αναλογικής τιμής στις τιμές LOW ή HIGH και εκχώρηση της τιμής ως κατάσταση του LED

}

}

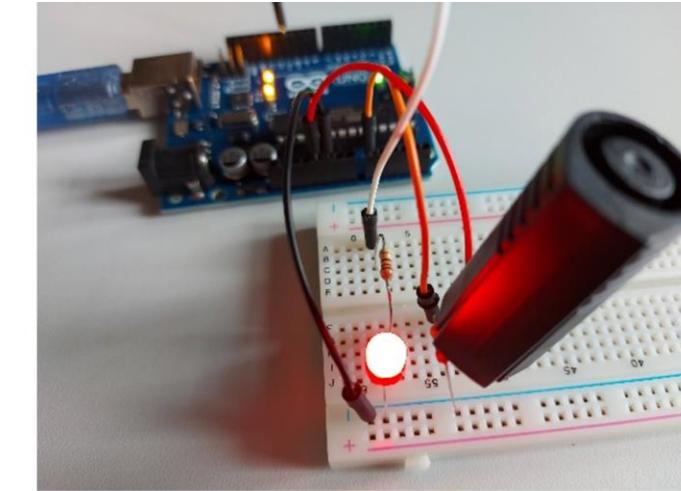
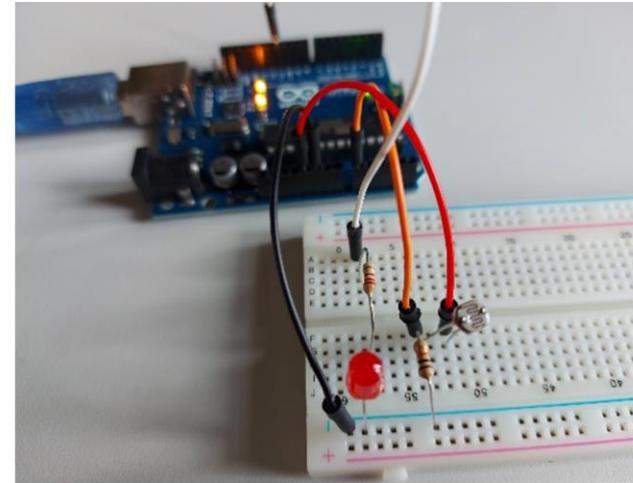
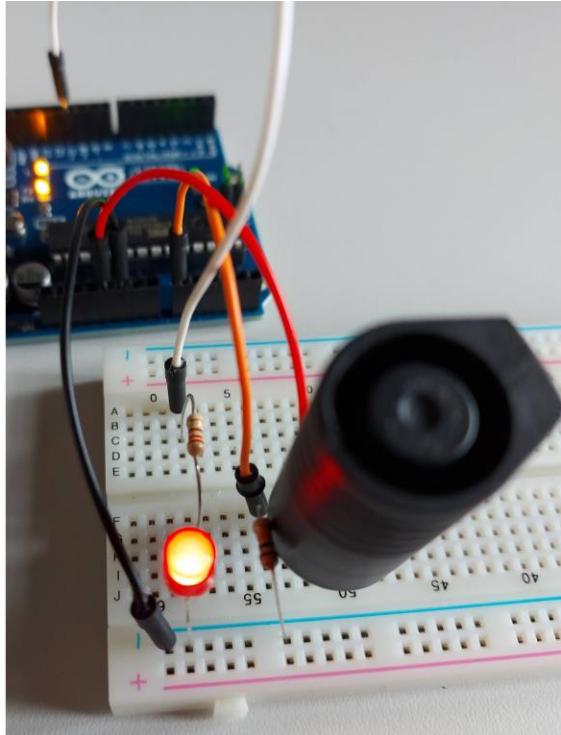
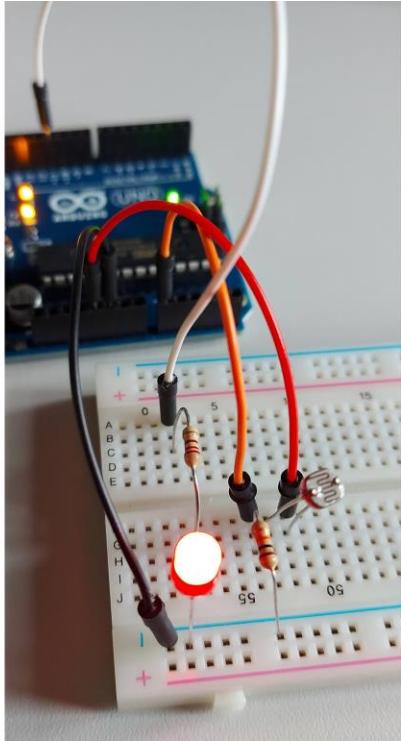
```
void analog() {
    value = analogRead(A0);
    Serial.print(value);
    Serial.print("\t");
    int analogValue = map(value, 512, 1023, 0, 255);
    analogWrite(11, analogValue);
    Serial.println(analogValue);
}
```

```
void digital() {
    value = analogRead(A0);
    Serial.print(value);
    Serial.print("\t");
    boolean digitalValue = map(value, 0, 600, HIGH, LOW);
    digitalWrite(11, digitalValue);
    Serial.println(digitalValue);
}
```

Προσοχή στις ρυθμίσεις του Serial Monitor:

No Line Ending ▾ 9600 baud ▾

Παράδειγμα 8: Αισθητήρας φωτεινότητας



Αναλογική λειτουργία

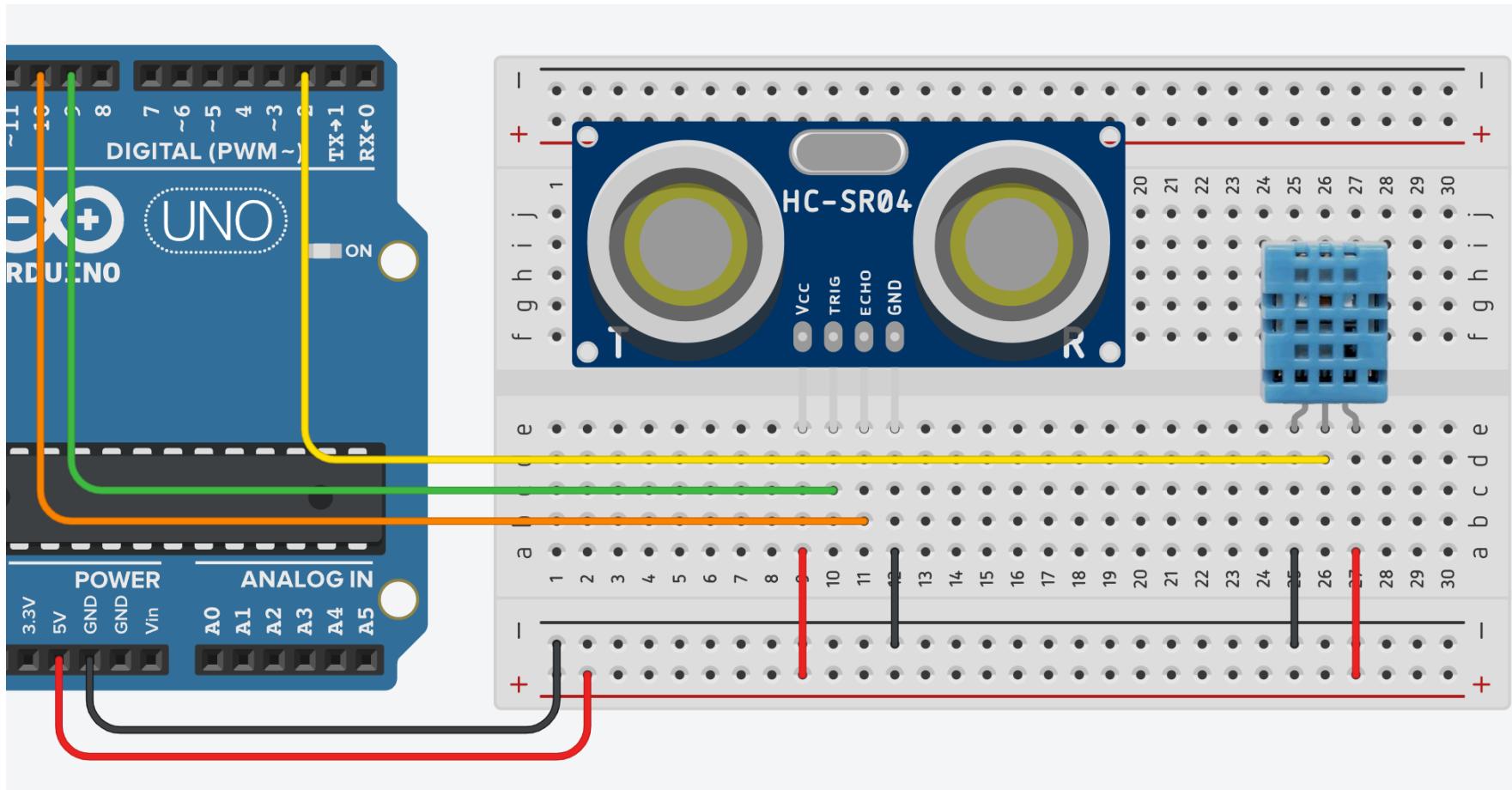
Ψηφιακή λειτουργία

Παράδειγμα 8: Αισθητήρας φωτεινότητας

1. Εξηγήστε τον ρόλο του διαιρέτη τάσης στο κύκλωμα.
2. Πραγματοποιήστε τις απαραίτητες αλλαγές ώστε η φωτοαντίσταση να μεταβάλλει τη συχνότητα που αναπαράγεται από ένα **buzzer**.

Παράδειγμα 9: SR04 + DHT11

Πρόγραμμα που διαβάζει θερμοκρασία και υγρασία και βάσει αυτών βελτιώνει την ακρίβεια των μετρήσεων απόστασης ενός αισθητήρα υπερήχων



Παράδειγμα 9: SR04 + DHT11

Arduino IDE → Sketch → Include Library → Add .ZIP Library...

```
#include <dht.h>
#include <SR04.h>

#define trigPin 9
#define echoPin 10
#define DHT11_PIN 2

float duration, distance;
float sound_speed;
float temp, hmdt;

dht DHT;
```

```
void setup() {
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    pinMode(DHT11_PIN, INPUT);
    Serial.begin(9600);
}
```

Ορισμός μεταβλητών και pins και έναρξη σειριακής επικοινωνίας

Συμπερίληψη βιβλιοθήκης για τον DHT11

Ανάγνωση τιμών θερμοκρασίας και υγρασίας
Αποστολή παλμού από το Trig για 10 μs

Μέτρηση διάρκειας παλμού
Διόρθωση τιμής
Υπολογισμός απόστασης

Εμφάνιση αποτελεσμάτων ή μηνυμάτων σφάλματος

Λήψη νέας μέτρησης σε 2 s

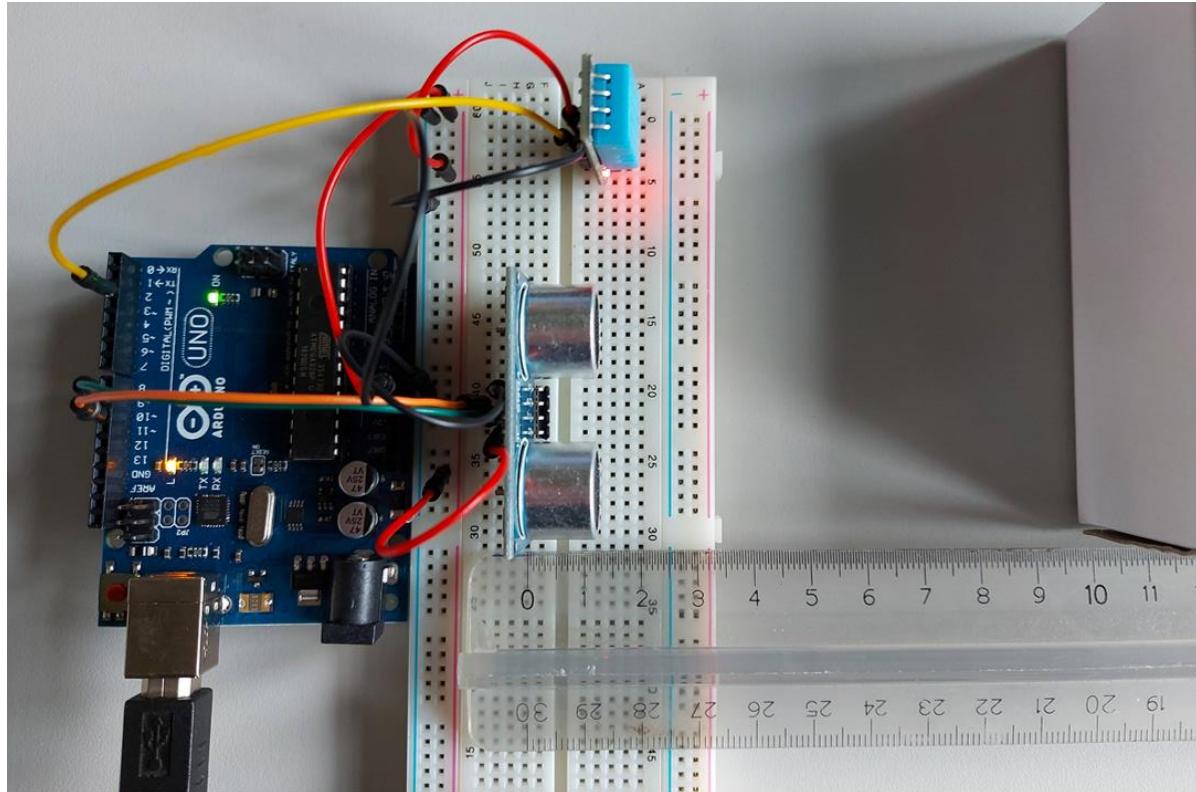
```
void loop() {
    int chk = DHT.read11(DHT11_PIN);
    temp = DHT.temperature;
    hmdt = DHT.humidity;

    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);

    duration = pulseIn(echoPin, HIGH);
    sound_speed = 331.4 + (0.606 * temp) + (0.0124 * hmdt);
    distance = (duration / 2) * (sound_speed / 10000);

    if (distance < 400 || distance > 2) {
        Serial.print(distance);
        Serial.println(" cm");
    } else {
        Serial.println("Out of range");
    }
    Serial.print(temp);
    Serial.println(" C");
    Serial.print(hmdt);
    Serial.println(" %");
    Serial.println("=====");
    delay(2000);
}
```

Παράδειγμα 9: SR04 + DHT11



Serial Monitor X

Message (Ctrl + Enter to send message to 'Arduino')

=====

10.25 cm

19.00 C

46.00 %

=====

10.36 cm

19.00 C

46.00 %

=====

Παράδειγμα 9: SR04 + DHT11

1. Τροποποιήστε το κύκλωμα και τον κώδικα, ώστε να αναπαράγεται ένας ήχος από ένα **buzzer** εάν κάτι πλησιάσει σε απόσταση <10 cm.
2. Τι πρόβλημα δημιουργεί η συνάρτηση **delay(2000)** στο τέλος του κώδικα;
3. Τροποποιήστε τον κώδικα ώστε να αποφευχθεί το παραπάνω πρόβλημα.