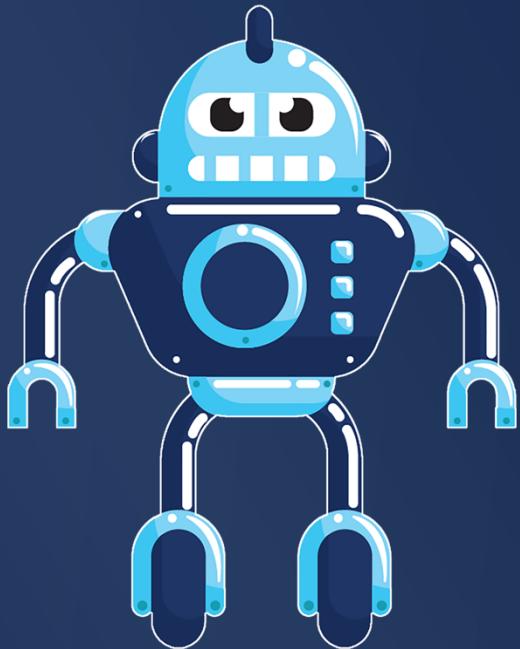


# CURS ROBOTICĂ



## Începători

[www.robotica.md](http://www.robotica.md) Ediția 2018

## Prefață

Te salut, eu sunt Roboțelul care te va ghida în procesul de studiere a roboticii, în primul rând vreau să-ți spun că îți apreciez interesul față de robotică, ești unul dintre puținii din Moldova care încearcă să cunoască domeniul roboticii. În acest curs îți-am pregătit o introducere în mediul roboticii, sunt sigur că deja cunoști cât de importantă este robotica în zilele noastre, deoarece foarte multe procese au devenit robotizate.

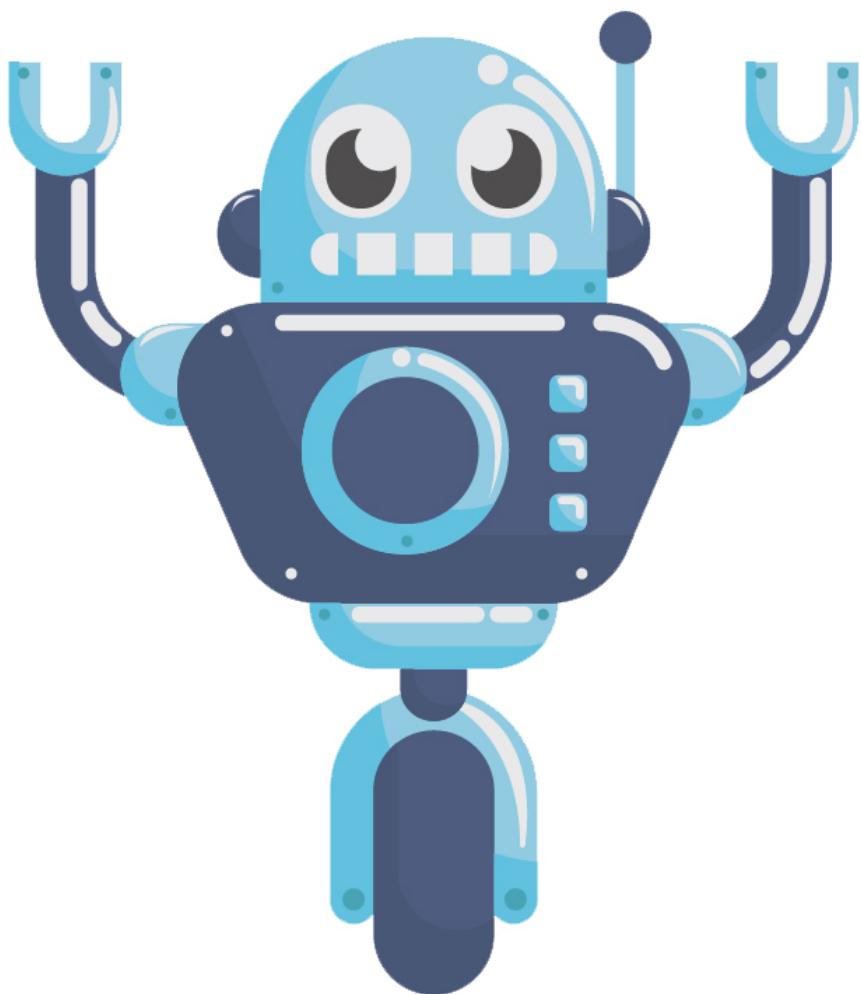
În viitorul apropiat în sistemul educațional va fi introdusă disciplina robotica, dar de ce să așteptă? Dacă poți studia acum această materie, ca urmare vei fi avanțat față de semenii tăi cunoscând mult mai multe în aceast domeniu.

Cum deja ai găsit în Kit, vom lucra cu platforma Arduino, în cazul Arduino, totul este foarte simplu de utilizat. Ai nevoie de câteva minute ca să îți instalezi mediul de dezvoltare și să scrii primul tău program pe Arduino, în primele capitole vom atinge aceste subiecte.

Avantajul major al platformei Arduino - ai totul integrat pe o placă ca urmare nu este necesar să lipești fire, tot ce ai nevoie este un port USB liber pentru a programa platforma.

Pentru orice întrebare sau nelămurire ai la dispoziție platforma noastră - <https://blog.robotica.md/>

Spor la treabă și mult succes!



## Cuprins

Introducere.....	1
Ce este Arduino?.....	5
Tipuri Arduino.....	6
Structura placii Arduino.....	10
Așezarea pinilor pe Arduino.....	13
Breadboard.....	14
Jump wires.....	15
Descrierea componentelor din KIT.....	17
Setare program Arduino IDE.....	19
Conectare Arduino.....	21
Structura unui program.....	23
Proiect 1 - Testarea Arduino.....	25
Proiect 2 - Clichește un LED.....	27
Proiect 3 - Buton de comandă.....	29
Proiect 4 - Potențiometru.....	31
Proiect 5 - Luminozitatea LEDurilor.....	33

Proiect 6 – Mișcarea LEDurilor.....	35
Proiect 7 – Histogramă cu LEDuri.....	37
Proiect 8 - LEDuri multiple.....	39
Proiect 9 - RGB LED.....	41
Proiect 10 - Foterezistor.....	43
Proiect 11 – Senzor temperatură.....	45
Anexe.....	47
Raspunsuri la întrebari.....	52
Inchiere.....	56

## Ce este Arduino ?

Arduino este una dintre cele mai simplu de utilizat platforme cu microcontroler. Această plăcuță pe care o ai în cutie este un minicalculator, care poate fi comparat cu un calculator obișnuit de acum 15 ani, minicalculatorul este capabil să culeagă informații, să le proceseze și să îndeplinească sarcinile setate.

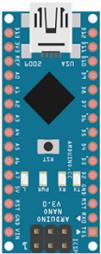
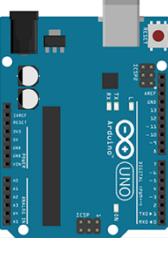
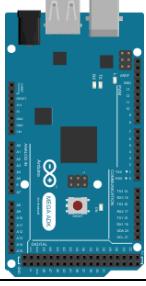
Arduino a fost creat în 2005 și a devenit foarte răspândit în toată lumea, multe școli și universități la lecțiile de informatică, robotică, fizică folosesc aceste platforme pentru a face învățământul mai interactiv și practic.

În jurul platformei Arduino s-a format o întreagă comunitate, ca urmare au apărut foarte multe dispozitive compatibile (senzori, traductori, afișaje), practic orice informație poate fi culeasă din exterior și prelucrată de microcontroler și afișată în mod setat.

Dacă ne referim la preluarea de informații din exterior, mai jos sunt doar câteva exemple de senzori : senzor de distanță, temperatură, umiditate, presiune atmosferică, forță de apăsare, grad de rotire, de incendiu și multe, multe altele, dar pentru prelucrarea informațiilor culese de senzor este nevoie de a avea o îndemânare practică în mediul Arduino, după parcurgerea acestui curs în totalmente cu siguranță vei putea trece la următorul nivel în care vei folosi o mare parte din acești senzori și vei crea aplicații complexe. În cursul pentru începători am pregătit o parte de explicare cum funcționează platforma Arduino, cum se fac conectările cu module, cum se culeg datele, cum datele pot fi afișate și o serie de 11 proiecte practice.

## Tipuri Arduino

Există multe tipuri de plăci Arduino, fiecare placă are propriile performanțe și medii în care este utilizată. Mai jos găsiți un tabel cu 3 cele mai răspândite tipuri de platforme Arduino.

Arduino	Nano	Uno	Mega
			
Frecvență de Ceas	16 MHz	16 MHz	16 MHz
Microcontroler	ATmega328	ATmega328	ATmega 2560
Număr Pini I/O digitali	22 (6 sunt PWM)	14 (6 sunt PWM)	54 (6 sunt PWM)
Număr Pini Input Analogici	8	6	16
Număr Pini PWM	6	6	14
Memorie Flash	32 Kb	32 Kb	256 Kb
Memorie SRAM	2 Kb	2 Kb	8 Kb
Tensiunea de operare	5V	5V	5V

**Frecvența de ceas** a procesorului este măsurată în cicluri pe secundă (unitatea de măsură Hz) este timpul în care procesorul realizează cea mai de bază operație, cum ar fi adunarea a două numere sau transferul de valori între doi registri. În mod uzual, frecvența procesorului este utilizată pentru a determina viteza unui procesor. Cu cât este mai mare această frecvență, cu atât procesorul execută mai rapid operațiile.

**Microcontrolerul** este un chip de dimensiune mică care poate prelucra și executa diverse operațiuni, comparabil cu procesorul doar ca are performanțe mai reduse.

**Pini Input/Output digitali**- prin intermediul acestor pini se pot introduce și prelua doar informații digitale (care pot avea doar una din două valori "0" sau "1" altfel spus "On" sau "Off"), mai jos în figură vedeti reprezentarea unui semnal dreptunghiular.



fig.1 Semnal digital

**Pini Input analogic** - se pot introduce doar semnale analogice. Un semnal analogic poate fi ușor perturbat de factori externi, în figură de mai jos intenționat semnalul sinusoidal nu este perfect. Inițial toate comunicațiile telefonice și radio, cât și televiziunea au funcționat cu tehnici analogice, acum practic toate au fost înlocuite cu un sistem digital, deoarece nu poate fi perturbat.

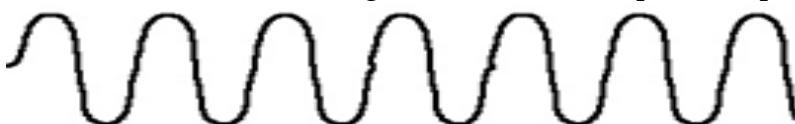


fig.2 Semnal analogic

**Pini PWM** sunt pinii la ce se pot aplica semnale PWM, PWM (Pulse Width Modulation) este o tehnică folosită pentru a varia în mod controlat tensiunea dată unui dispozitiv electronic. Această metodă schimbă foarte rapid tensiunea oferită dispozitivului respectiv din ON în OFF și invers (treceri rapide din HIGH (5V de exemplu) în LOW (0V)). Perioada de timp corespunzătoare valorii ON dintr-un ciclu ON-OFF se numește factor de umplere (duty cycle) și reprezintă, în medie, ce tensiune va primi dispozitivul electronic. Astfel, se pot controla circuite analogice din domeniul digital. Practic, asta înseamnă că un LED acționat astfel se va putea aprinde / stinge gradual, iar în cazul unui motor acesta se va învârti mai repede sau mai încet.

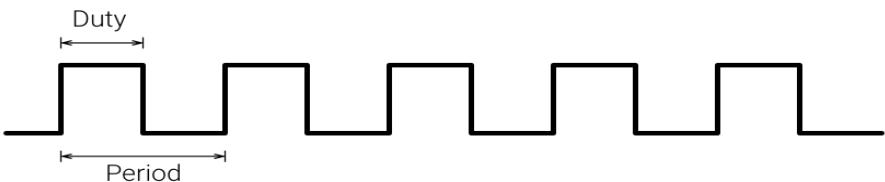


fig.3 Semnal PWM

**Memorie Flash** este o memorie nevolatilă (în care datele nu sunt șterse după fiecare resetare sau pierdere de alimentare cu energie electrică) și care la nevoie poate fi stearsă și reprogramată (reincărcată cu date).

**Memorie SRAM** este o memorie volatilă (datele stocate se pierd dacă alimentarea cu energie electrică se întrerupe) mai este și o memorie cu acces aleator. Această implică faptul că orice cuvânt din memorie poate fi accesat în același timp, datorită paralelismului în accesare se poate obține o viteza mare de prelucrare.

**Tensiunea de operare** este tensiunea cu care operează platforma, pentru microprocesoare de obicei este 5 Volti.

**Arduino Nano** cel mai des este folosit la proiecte de complexitate mică sau medie în care spațiul este limitat, spre exemplu la elaborarea unei stații meteo care va măsură temperatură și umiditatea aerului ar putea conta dimensiunea întregului sistem.

*Avantaje* - dimensiune mică și performanțe suficiente pentru proiecte mici și medii

*Dezavantaje* - firele de conexiune a modulelor nu pot fi legate direct, se pot lipi direct firele pe placă Arduino Nano sau conecta prin intermediul unui breadboard.

**Arduino Uno** cel mai răspândit tip, deoarece are performanțe suficiente pentru proiecte de complexitate medie și ușor în operare.

*Avantaje* - comod în utilizare, are un port separat pentru alimentare din exterior, firele se pot legă direct pe placă fără lipire.

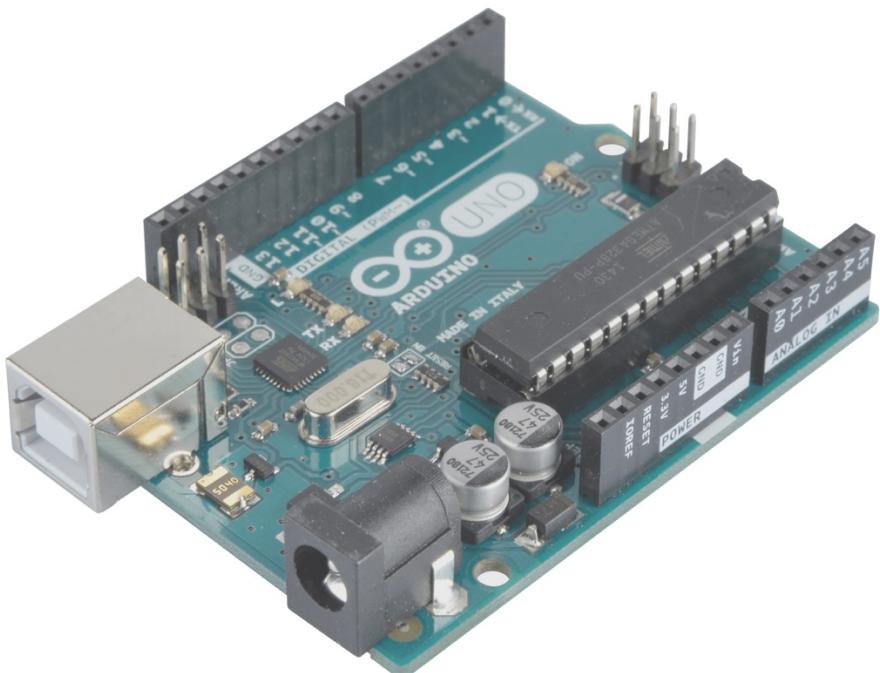
*Dezavantaje* - performanțe mai reduse față de Arduino Mega.

**Arduino Mega** este o versiune mai complexă, care conține un procesor cu performanțe mai bune și memorie flash mai mare, este folosit la proiecte complexe ce necesită prelucrarea rapidă a datelor și/sau necesită mai mulți pini de intrare/ieșire.

*Avantaje* - performanțe mari, număr mare de pini.

*Dezavantaje* - dimensiunea mare a plăcii.

## Plataforma Arduino Uno



## Structura plăcii Arduino

Aveți în față plăcuța Arduino de tip uno, înainte de a începe lucru pe ea, trebuie să lămurim fiecare portiune a plăcii pentru ce ne servește și ce putem face cu ea.

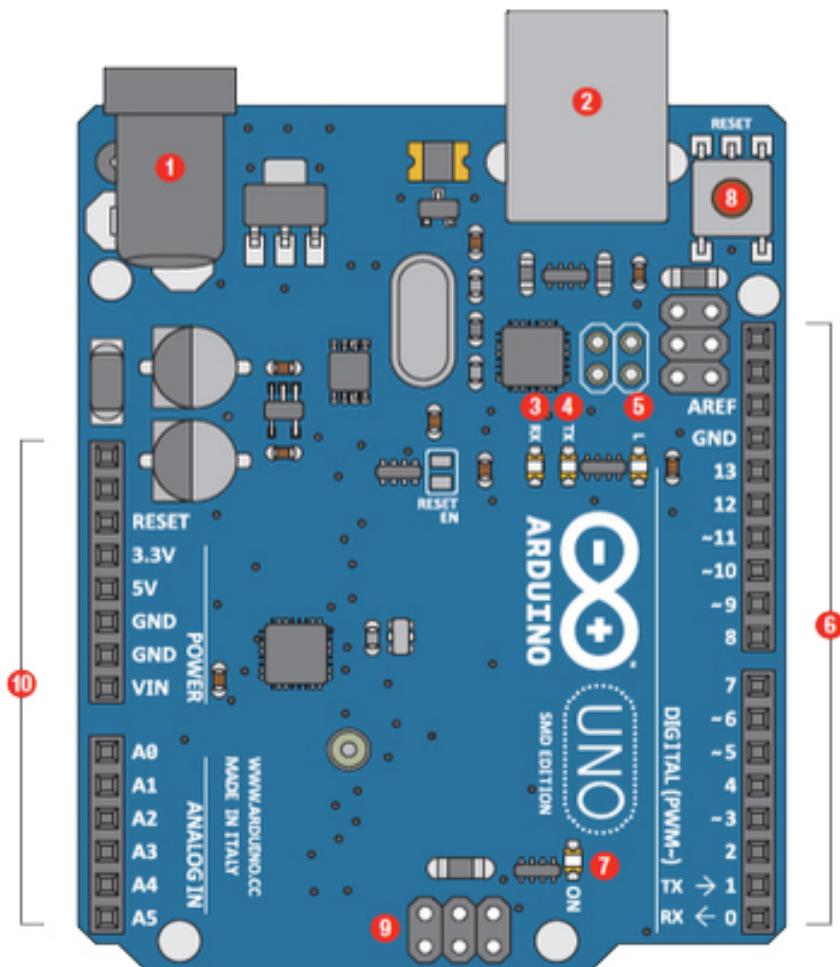


fig.4 Structura Arduino

- 1. Barrel Jack** - conector de alimentare a plăcii Arduino cu tensiune între 9 și 12V, se poate alimenta fi de la o baterie, o sursă reglabilă de curent continuu sau un alimentator.
- 2. USB Port** - se poate alimenta placă cu o tensiune de 5V și se folosește pentru comunicarea plăcii cu calculatorul, pentru a încărca programul pe Arduino.
- 3. LED (RX: Receptor)** - indică când Arduino primește date (exemplu: moment în care este încărcat programul pe placă LEDul clipește).
- 4. LED (RX: Emițător)** - indică când Arduino transmite date (exemplu: când rulăm un program – LEDul clipește).
- 5. LED (Pin 13: Alertă)** - este încorporat în programul care urmează a fi rulat pe Arduino, el semnalează dacă apare vreo eroare în program.
- 6. Pini (ARef, Nul, Digital, Rx, Tx)** - pot fi folosiți pentru introducere de date, pentru recepționarea datelor, pentru alimentare și împământare (pe următoarea pagină găsești reprezentarea schematică a pinilor).
- 7. LED (Indicator regim de lucru)** - indică dacă Arduino este alimentat și funcționează corect.
- 8. Buton de resetare** - permite resetarea manuală a programului (rularea de la început a programului).
- 9. ICSP Pini (Încărcarea codului fără Bootloader)** - sunt folosiți pentru "In-Circuit Serial Programming", se folosește dacă se vrea încărcarea unui program fără bootloader.

**10. Pini (Intrări Analogice, Alimentare Arduino, Nul, Alimentare Exterior, Resetare)** - folosiți pentru a introduce date analogice, alimentarea platformei Arduino, împământare, alimentarea modulelor din exterior și resetare.

## Așezarea pinilor pe Arduino

Deja cunoști că fiecare arduino are un număr diferit de pini, mai jos vezi reprezentat schematic tipul fiecărui pin.

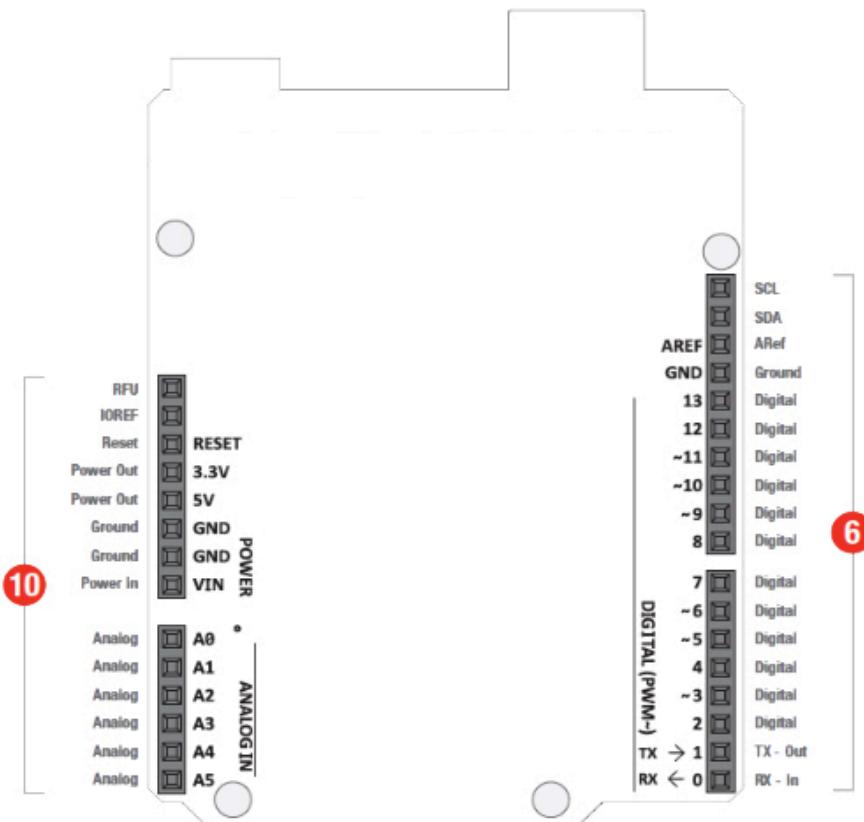


fig.5 Reprezentare pinilor Arduino UNO

## Ce este un Breadboard

**Breadboard** (solderless breadboard) este o placă pe care se pot pune elemente de circuit (module). Pe această placă nu se lipește nimic ceea ce permite modificarea ușoară a proiectului și remedierea erorilor.

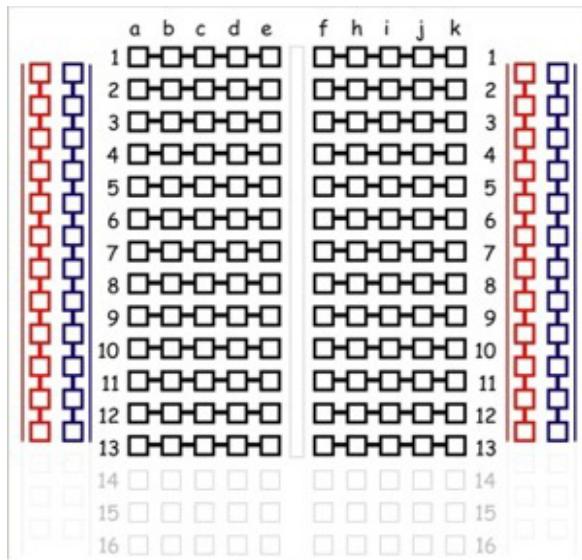


fig.6 Structura BreadBoard

Este foarte important să înțelegi structura unei plăci breadboard - vei opera foarte mult cu ea. Pentru o alimentare comodă se utilizează cele 2 perechi de rânduri + (roșu) și - (albastru), ca urmare fiecare intrare din acest rând va fi tensiune și masă. Observați că breadboard-ul este împărțit în 2 părți egale, ( $1a=1b=1c=1d=1e$ ) și ( $1f=1h=1i=1j=1k$ ) legăturile sunt făcute pe rând a câte 5 intrări, adică ce este introdus pe linia 1 intrarea "a" se regăsește și pe  $b=c=d=e$  linia 1.

## *Jump wires*

**Fire Jump** (cunoscut ca fir jumper sau simplu - jumper) este un fir (conductor) electric, el poate fi de 3 tipuri (*male-male*, *male-female* / *female-male*, *female-female*). Aceste fire sunt folosite pentru conectarea modulelor între ele, avantajul acestora - nu este nevoie de a lipi fire, în kit ai primit fire de diferite culori, intenționat am făcut acest lucru, pentru a diferenția cu ușurință legăturile din proiectul tău.

În fig.7 găsești reprezentarea schematică a firelor jumper

female-female (mama-mama) - se conectează 2 componente care la capăt au câte un pin de tip conector (nu se poate conecta pe un breadboard)

male-female/female-male (mama-tata sau tata-mama) - se pot conecta componente, la care o componentă are pin de intrare și cealaltă componentă are un conector (se poate folosi cu breadboard)

male-male (tata-tata)-se conectează componente care au pini de intrare (se poate folosi cu breadboard)



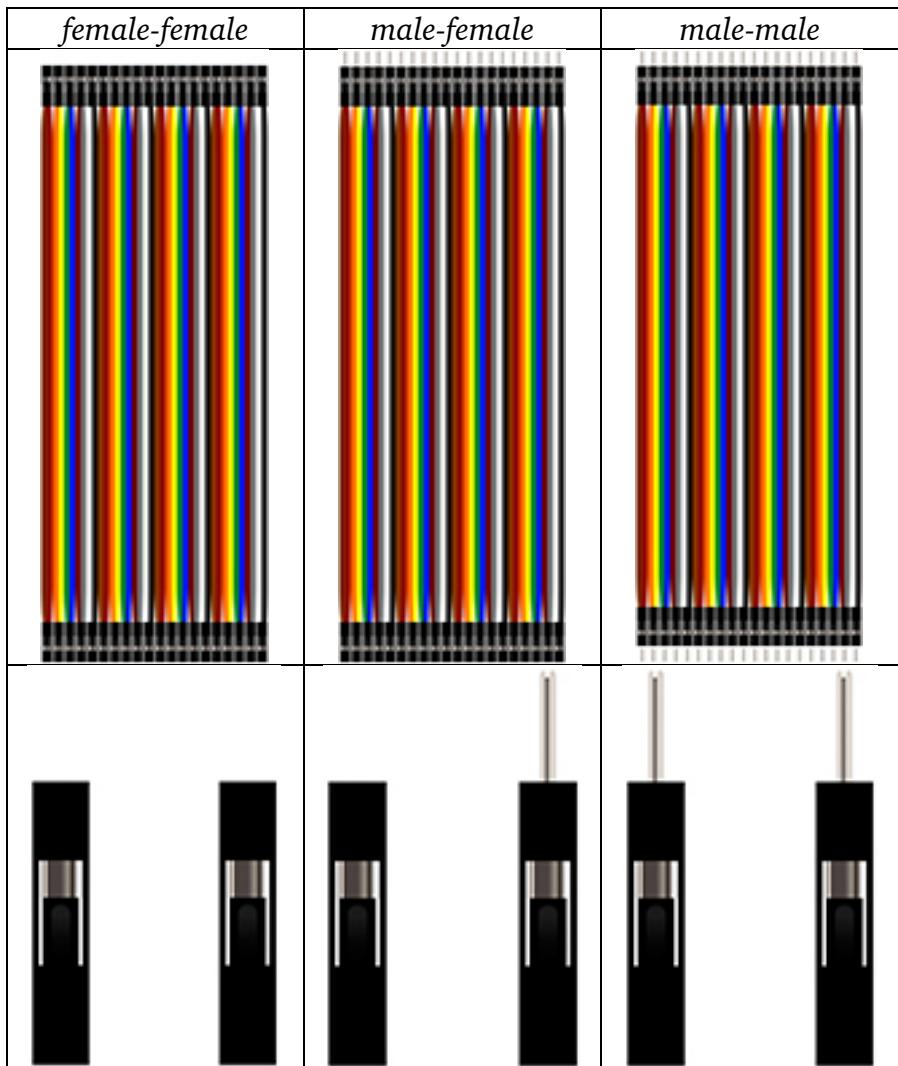


fig.7 Tip de Jumpere

## *Descrierea componentelor din KIT*

**LED** - (Dioda emițătoare de lumină) este o diodă ce conține materiale semiconductoare care convertesc energia electrică direct în lumină, efectul fiind o formă de electroluminiscență. Electroluminiscență este un fenomen optic și electric, în care un material emite lumină atunci când curentul electric trece prin el. Materialele semiconductoare (galiu, arsen sau fosfor) sunt stimulate de mișcarea electronilor. În acest fel, acești semiconductori generează fotoni, care sunt percepți sub formă de lumină.

LED-urile nu se ard, nu devin fierbinți și nu utilizează nicio substanță nocivă. În comparație cu becurile clasice cu filament care transformă 90% din energia electrică consumată în căldură care se pierde, LED-urile disipă căldura prin bază LED-ului, astfel încât aproape toată energia este transformată în lumină.

**LED RGB** - Sunt sigur că în copilărie te-ai jucat cu acuarele, iar în momentul în care nu ai avut culoarea necesară ai combinat culorile existente obținând în final ceea ce doreai. Un led RGB funcționează după același principiu. Combinând roșu, verde și albastru putem obține aproape orice culoare dorim, în rest funcționează după același principiu ca un LED simplu.

**Rezistor** - este o piesă componentă din circuitele electrice și electronice a cărei principală proprietate este rezistența electrică. Rezistorul are rolul de a proteja alte componente din circuit, spre exemplu alimentezi un LED de la sursă de alimentare și într-un moment îți crește curentul, caz în care ai prevăzut un rezistor el va arde și se va întrerupe circuitul protejând LEDul de la ardere, în caz contrar va arde LEDul. Rezistorul obișnuit are două terminale; conform legii lui Ohm, curentul electric care curge prin rezistor este proporțional cu

$$I = \frac{U}{R}$$
tensiunea aplicată pe terminalele rezistorului.

Cel mai important parametru al unui rezistor este rezistența sa electrică, exprimată în ohmi.

**Fotorezistor** - acționează ca un rezistor, cu excepția faptului că valoarea rezistenței depinde de câtă lumină strălucește pe el. Odată ce intensitatea lumii aplicate pe fotorezistor crește, scade rezistența lui și invers cu scăderea intensității luminii pe fotorezistor, crește rezistența lui. Cu ajutorul fotorezistorului se pot face aplicații de automatizare, odată ce se întunecă în cameră (adică cantitatea de lumina aplicată pe fotorezistor scade) să dea comandă la întrerupătorul de lumină și să aprindă lumină.

**Potențiometru** - este un rezistor variabil, îndeplinește aceeași funcție ca un rezistor, doar că se poate ajusta valoarea rezistenței manual, spre exemplu prin rotire.

## *Setare program Arduino IDE*

Înainte de a începe o aplicație, vom discuta fiecare secțiune din programul Arduino IDE.

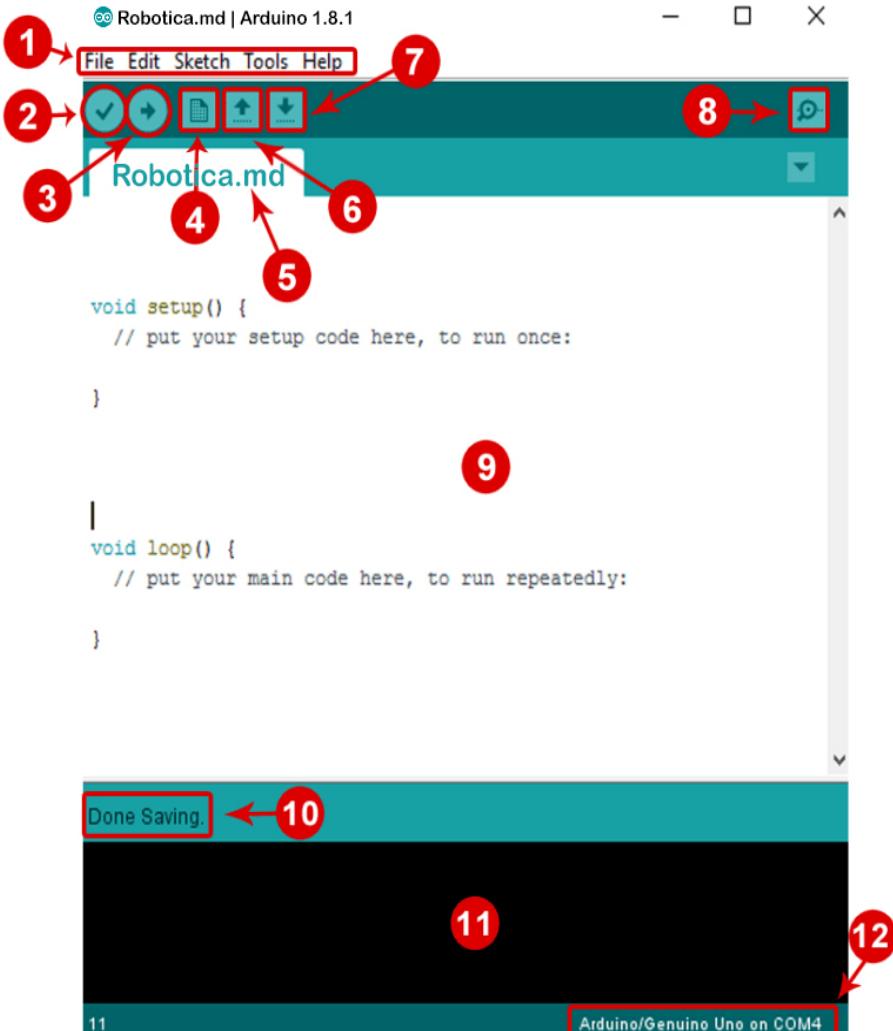


fig.8 Software Arduino IDE

- 1. Menu Bar:** Oferă acces la instrumentele necesare creării și salvării proiectelor Arduino.
- 2. Verify Button:** Compilează codul și verifică erorile în ortografie sau sintaxă.
- 3. Upload Button:** Trimit codul la placa conectată, în cazul tău e Arduino Uno. Luminile de pe placă vor clipi rapid când se încarcă un program.
- 4. New Sketch:** Deschide o fereastră nouă care conține o schiță de proiect.
- 5. Sketch Name:** Când proiectul este salvat, numele acestuia este afișat aici.
- 6. Open Existing Sketch:** Îți permite să deschizi un proiect salvat sau unul dintre exemplele stocate.
- 7. Save Sketch:** Îți salvează proiectul deschis în prezent.
- 8. Serial Monitor:** Atunci când placa este conectată, aceasta va afișa informațiile seriale ale plăcii Arduino.
- 9. Code Area:** Această zonă este locul în care compuneți codul proiectului care va fi rulat pe Arduino.
- 10. Message Area:** Această zonă vă indică starea salvării, compilarea codurilor, erorile și multe altele.
- 11. Text Console:** Afisează detaliile unui mesaj de eroare, mărimea programului care a fost compilat și informații suplimentare.
- 12. Board and Serial Port:** Îți spune ce placă este utilizată și la ce port serial este conectat.

## Conectare Arduino

În acest moment, ești pregătit să conectezi dispozitivul Arduino la computer. Conectați un capăt al cablului USB la arduino Uno și apoi celălalt capăt al portului USB la portul USB al computerului. Ai nevoie de un driver special, îl găsești pe platforma noastră – [www.robotica.md/drivere-arduino](http://www.robotica.md/drivere-arduino)

Odată ce bordul este conectat, va trebui să mergi la "Tools", apoi "Board" apoi selectezi Arduino Uno.

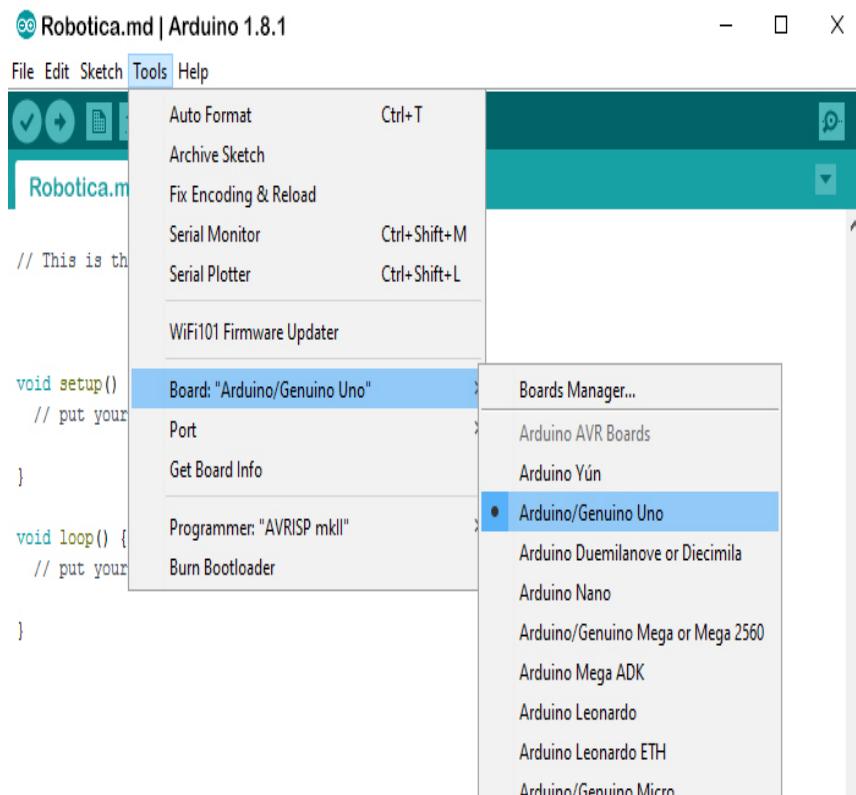


fig.9 Software Arduine IDE Board

Apoi, trebuie să-i setezi portul pe care este conectat Arduino la computer.

Pentru a selecta portul, mergi la "Tools", apoi la "Port", și selectezi portul care este indicat. Acum ești gata să rulezi programe pe Arduino.

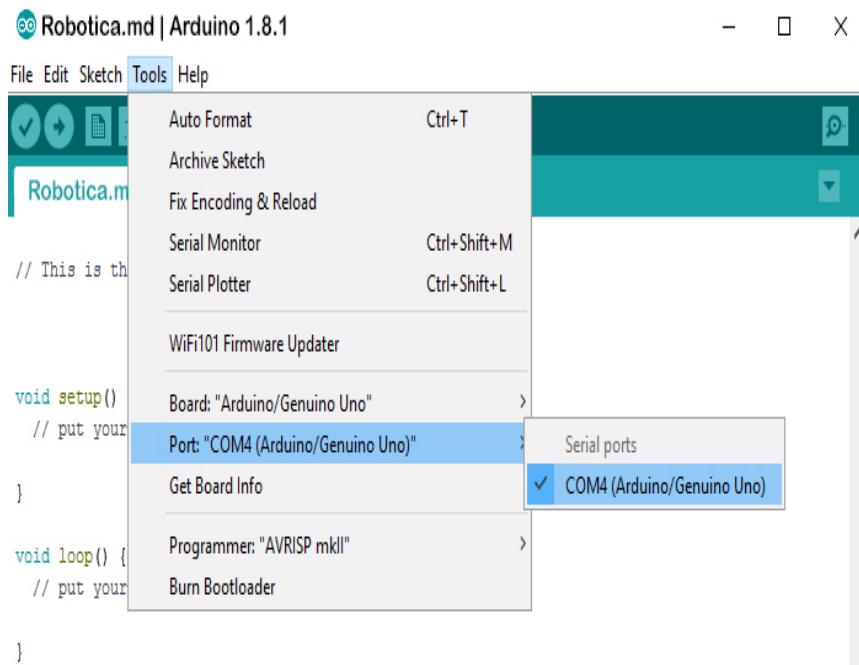


fig.10 Software Arduino IDE Port

## Structura unui program

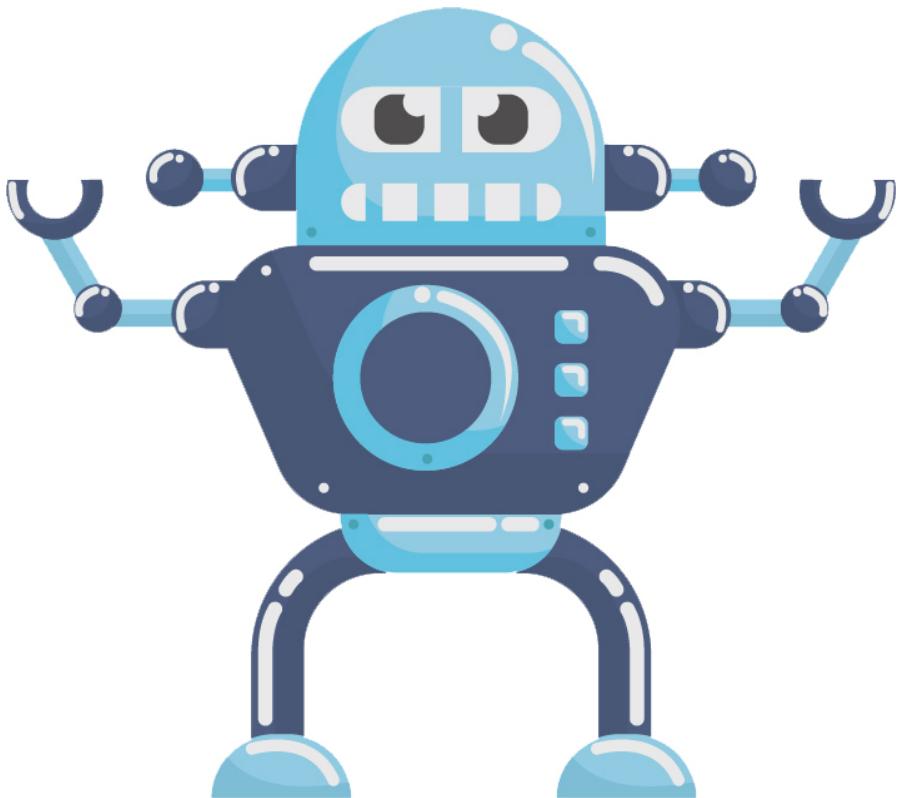
Orice program Arduino are două secțiuni. Secțiunea "setup", care este rulată doar o singură dată, atunci când placa este alimentată (sau este apăsat butonul "Reset"), și secțiunea "loop", care este rulată în ciclu, atât timp cât este alimentată placa. Să luăm un exemplu.

```
void setup() {  
    //codul scris aici ruleaza o singura data  
}  
void loop() {  
    //codul scris aici ruleaza tot timpul  
}
```

Astfel, în rutina "setup" vom pune de obicei cod de inițializare, iar în rutina "loop" vom scrie partea principală a programului nostru. Pe parcurs te vei obișnui când vei rula câteva programe.

Iată și s-a terminat partea de introducere, sper că nu te-ai plăcuit, dar la sigur îți vor fi de folos cunoștințele accumulate, fără nici o problemă poți reveni la oricare capitol de mai sus. Fiecare proiect prezentat mai jos va conține 6 părți, descriere proiectului, componentele necesare pentru implementare, diagrama proiectului - te ajuta să vizualizezi cum implementezi proiectul, etapele de implementare, codul proiectului și întrebările sau exercițiile care trebuie rezolvate.

## APLICAȚII PRACTICE



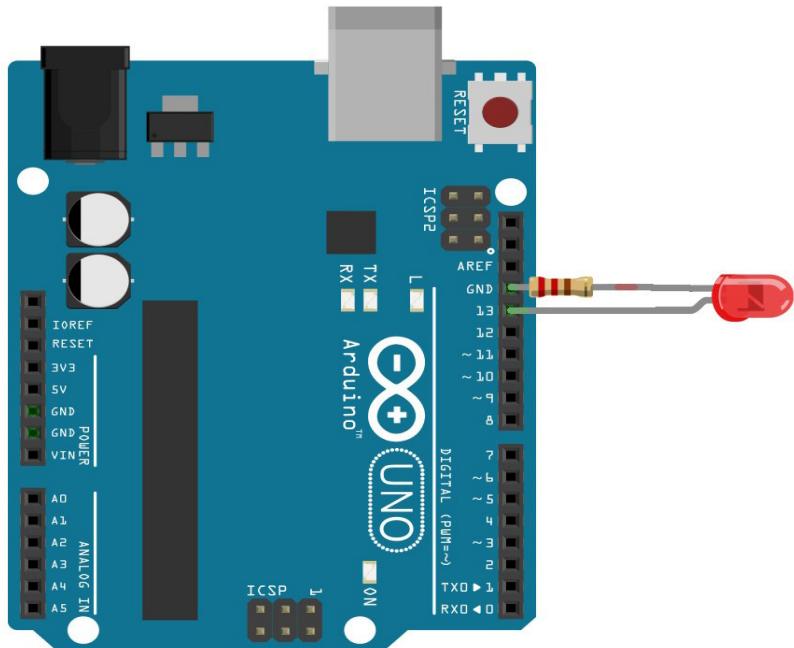
## #1 - Testarea Arduino

Primul proiect este unul dintre cele mai simple circuite pe care le poți crea cu Arduino. Acest proiect va testa placă Arduino pe care o ai prin clipirea unui LED care este conectat direct la bord.

### *Componete necesare*

- Arduino Uno
- USB A-to-B cablu
- LED 5mm
- Rezistor 220  $\Omega$

### *Diagrama proiectului*



## *Etapele proiectului*

1. Răsucesți o rezistență de  $220\ \Omega$  la piciorul scurt (-) al LED-ului.
2. Întrodu piciorul lung al LED-ului în pinul #13
3. Întrodu piciorul rezistorului care este conectat la LED-ul în pinul de masă (GND) de pe placă.

## *Codul proiectului*

```
void setup()
{
    pinMode(13, OUTPUT);
}

void loop()
{
    digitalWrite(13, HIGH);      // Aprinde LEDul
    delay(1000);                // Pauza 1 secunda
    digitalWrite(13, LOW);       // Stinge LEDul
    delay(1000);                // Pauza 1 secunda
}
```

## *Întrebări/Exerciții*

Încearcă să schimbi valoarea 1000 cu diferite valori și vezi ce se întâmplă. Cu cât valoarea este mai mică cu atât mai repede se aprinde și stinge Ledul. - De ce?

Scade timpul de pauză până la 10 ms. Mai poți observa aprinderea și stingerea LEDului?

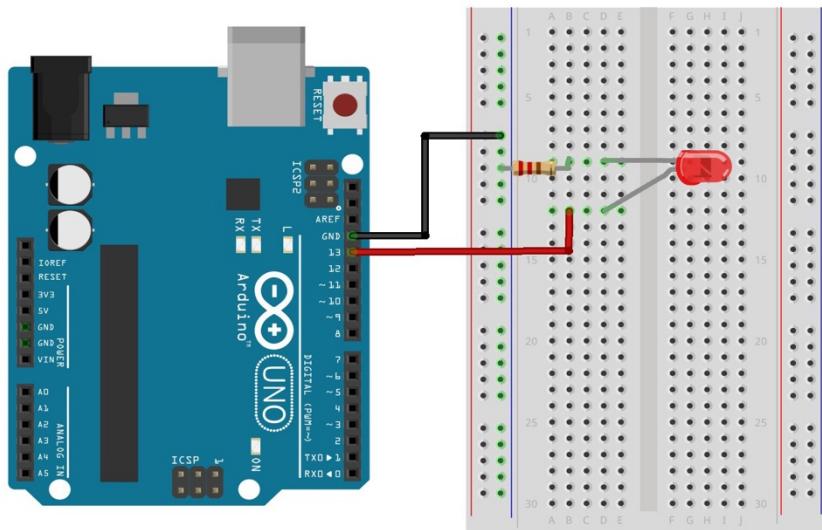
## #2 Clipește un LED

Acest proiect este identic cu proiectul nr. 1, cu excepția faptului că îl vom construi pe un breadboard. Odată terminat, LED-ul ar trebui să se aprindă pentru o secundă și apoi să fie oprit pentru o secundă într-o buclă.

### *Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- LED 5mm
- Rezistor 220  $\Omega$
- 2 Fire Jump

### *Diagrama proiectului*



## *Etapele proiectului*

1. Un capăt al jumper-ului (tata-tata) introduci în pinul GND de pe Arduino și celălalt capăt pe orice intrare de pe linia albastră.
2. Al 2-lea jumper (tata-tata) introduci în pinul #13 și cealalt capăt pe orice punct din breadboard (a-b-c-d-e).
3. Conectezi piciorușul lung al LEDului pe același rând unde este conectat jumperul care vine de la pinul #13.
4. În orice intrare de pe linia albastră introduci un capăt al rezistorului și celălalt capăt introduci pe o intrare (a-b-c-d-e).
5. Pe același rând unde este conectat piciorușul rezistorului conectezi al 2-lea picioruș al LED-ului .

## *Codul proiectului*

```
void setup()
{
    pinMode(13, OUTPUT);
}

void loop()
{
    digitalWrite(13, HIGH);      // Aprinde LEDul
    delay(1000);                // Pauza 1 secunda
    digitalWrite(13, LOW);       // Stinge LEDul
    delay(1000);                // Pauza 1 secunda
}
```

## *Întrebări/Exerciții*

Găsești un timp minim la care observi aprinderea și stingerea LEDului. Care este frecvența de aprindere a LED-ului?

Modifică codul ca să emită frecvența bătăilor de inimă.

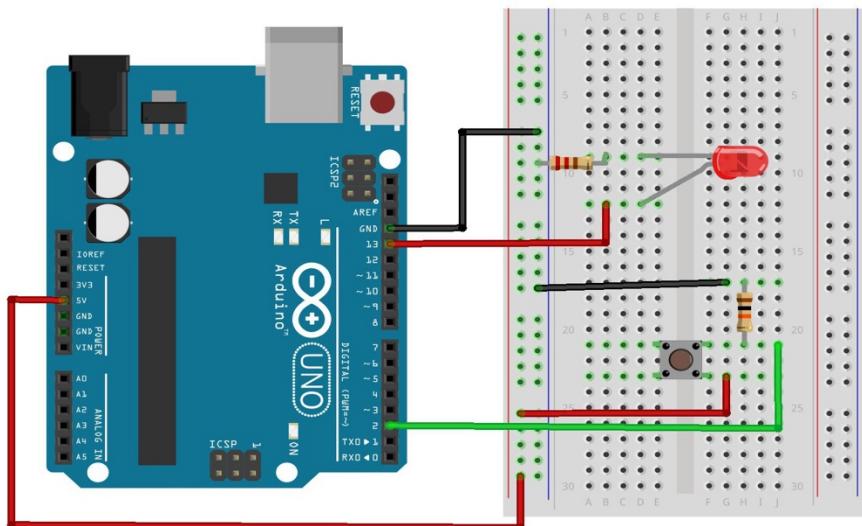
## #3 Buton de comandă

Utilizând un buton de comandă, vei putea activa și dezactiva un LED.

*Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- LED 5mm
- Rezistor 220 Ω
- Rezistor 10K Ω
- Buton de comandă
- 6 Fire Jumper

*Diagrama proiectului*



## *Etapele proiectului*

1. LED-ul conectat la pinul #13 și masă (GND).
2. Butonul de acționare conectat la pinul 2 și la tensiune (+5V)
3. Rezistorul 10K conectat la pinul 2 și masă (GND).

## *Codul proiectului*

```
const int buttonPin = 2;           // numărul pinului buton
const int ledPin = 13;            // numărul pinului LED

int buttonState = 0;              // variabilă pentru
citirea stării butonului

void setup() {
    // inițializarea pin LED ca fiind OUTput (ieșire)
    pinMode(ledPin, OUTPUT);

    // inițializarea pin buton ca fiind INput (intrare)
    pinMode(buttonPin, INPUT);
}

void loop() {
    // citirea valorii butonului:
    buttonState = digitalRead(buttonPin);

    // verificare dacă butonul este apăsat.
    if (buttonState == HIGH) {
        // aprindere LED
        digitalWrite(ledPin, HIGH);
    } else {
        // stingere LED
        digitalWrite(ledPin, LOW);
    }
}
```

## *Întrebări/Exerciții*

Modifică proiectul ca la apăsarea butonului LED-ul să se stingă.

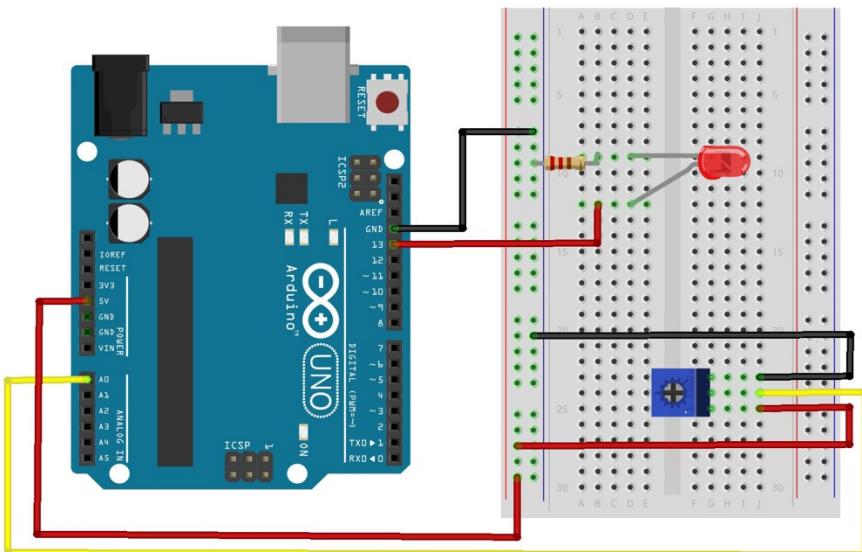
## #4 Potențiometru

Folosind un potențiometru, vei putea controla rezistența unui LED. Rotirea butonului va mări și micșora frecvența clipirii LED-ului.

### Componete necesare

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- LED 5mm
- Rezistor 220 Ω
- Potențiometru 10K
- 6 Fire Jumper

### Diagrama proiectului



## *Etapele proiectului*

1. Pinul central al potențiometrului conectat la pinul A0, o parte conectata la masă (GND) si cealaltă parte la tensiune (+5V).
2. Anodul LEDului (picior lung) conectat la ieșirea digitală #13.
3. Catodul LEDului (picior scurt) conectat la masă (GND).

## *Codul proiectului*

```
int sensorPin = A0;      // pin intrare pentru
potentiometru (neaparat pin analog!)
int ledPin = 13;          // pin intrare LED
int sensorValue = 0;      // variabilă pentru a stoca
valoarea provenită de la senzor

void setup() {
    // inițializarea pin LED ca fiind OUTput (ieșire)
pinMode(ledPin, OUTPUT);
}

void loop() {
    // citirea valorii de la senzor
sensorValue = analogRead(sensorPin);
    // aprindere LED
digitalWrite(ledPin, HIGH);
    // oprire program pt <sensorValue> milisecunde:
delay(sensorValue);
    // turn the ledPin off:
digitalWrite(ledPin, LOW);
    // oprire program pt <sensorValue> milisecunde:
delay(sensorValue);
}
```

## *Întrebări/Exerciții*

Modifică codul întrucât după actionarea potențiometru să acționeze LEDul doar după 3 secunde.

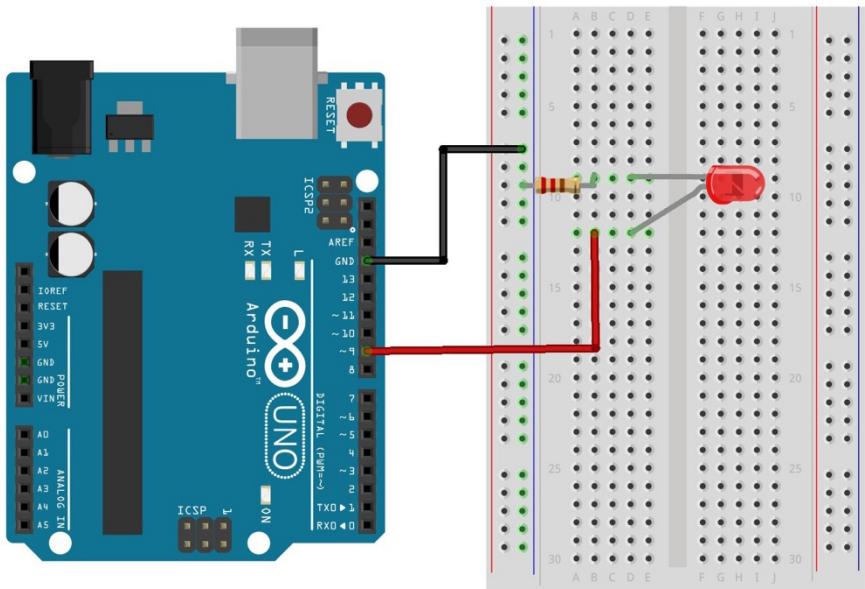
## #5 Luminozitatea LEDurilor

Folosind un pin PWM pe Arduino, vei putea crește și reduce intensitatea luminozității unui LED.

*Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- LED 5mm
- Rezistor 220 Ω
- 2 Fire Jumper

*Diagrama proiectului*



## *Etapele proiectului*

Funcția analogWrite () utilizează PWM, deci dacă dorești să modifici Pinul utilizat, asigurate că utilizezi un alt pin PWM. Pinii PWM sunt identificați cu un semn "~" (de regulă ~ 3, ~ 5, ~ 6, ~ 9, ~ 10 și ~ 11), în rest implementarea fizică este aceeași ca la proiectul 2.

## *Codul proiectului*

```
int led = 9;           // pin PWM conectat LED
int brightness = 0;    // nivel luminozitate LED
int fadeAmount = 5;    // nr. devizărilor luminozității
void setup() {
    // declare pin 9 to be an output:
    pinMode(led, OUTPUT);
}
void loop() {
    // setarea luminozității LED
    analogWrite(led, brightness);

    // modificarea luminozității pentru următorul ciclu
    brightness = brightness + fadeAmount;

    // creșterea/descrescerea luminozității
    if (brightness <= 0 || brightness >= 255) {
        fadeAmount = -fadeAmount;
    }
    // pauză 30 milisecunde între cicluri
    delay(30);
}
```

## *Întrebări/Exerciții*

Modifică programul ca să ai 12 puncte diferite de luminozitate și o pauză de 45 milisecunde între cicluri.

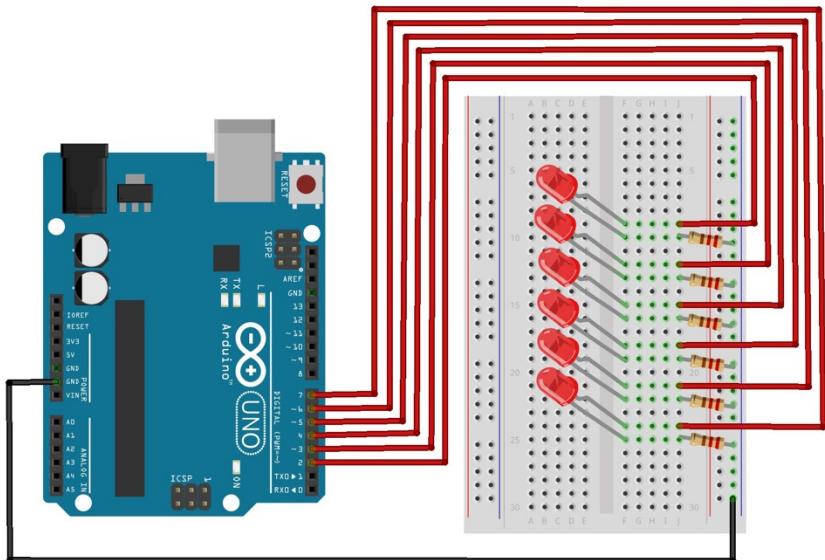
## #6 Mișcarea LEDurilor

In acest proiect vom clipi 6 LED-uri, unul câte unul consecutiv. Acest tip de circuit a fost făcut renumit de show-ul Knight Rider, care include o mașină cu LED-uri în buclă.

### Componetele necesare

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- 6 LEDuri 5mm
- 6 Rezistoare 220 Ω
- 7 Fire Jumper

### Diagrama proiectului



## *Etapele proiectului*

Conectăm anodul (picior lung) LEDurile la pinii de la 2 la 7 și catodul (picior scurt) la masă (GND) prin rezistoare de  $220\ \Omega$ .

## *Codul proiectului*

```
int timer = 100; // Timpul de pauza

void setup() {
    // bucla pentru a initializarea iesirilor
    for (int thisPin = 2; thisPin < 8; thisPin++) {
        pinMode(thisPin, OUTPUT);
    }
}

void loop() {
    // bucla de la cel mai mic pana la cel mai mare
    for (int thisPin = 2; thisPin < 8; thisPin++) {
        digitalWrite(thisPin, HIGH);
        delay(timer);
    }
    digitalWrite(thisPin, LOW);

    // bucla de la cel mai mare pana la cel mai mic
    for (int thisPin = 7; thisPin >= 2; thisPin--) {
        digitalWrite(thisPin, HIGH);
        delay(timer);
    }
    digitalWrite(thisPin, LOW);
}
```

## *Întrebări/Exerciții*

Ce se va întâmpla dacă valoarea lui "timer" va fi 1000ms?

Modifică codul ca ledurile să se aprinde în sens invers.

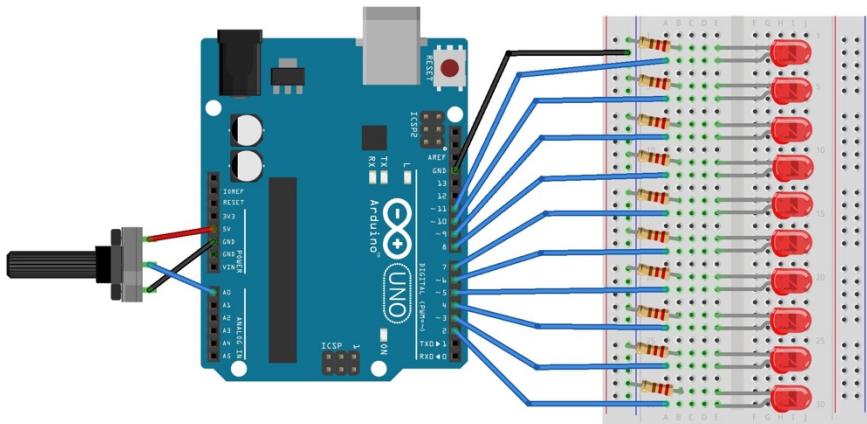
## #7 Histogramă cu LEDuri

Folosind un potențiometru, puteți controla o serie de LED-uri la rând. Prin acționarea potențiometrului vei activa sau dezactiva mai multe LED-uri în același timp. Deși în cazul dat utilizăm 10 LEDuri, poți utiliza orice număr de Leduri.

### *Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- Potențiometru rotativ
- 10 LEDuri 5mm
- 10 Rezistoare 220 Ω
- 11 Fire Jumper

### *Diagrama proiectului*



## *Etapele proiectului*

Conectăm anodul (picior lung) LEDurile la pinii de la 2 la 11 și catodul (picior scurt) la masă (GND) prin rezistoare de  $220\ \Omega$  și conectăm pinul din mijloc al potențiometrului la pinul A0, și deja ceilalți pini la tensiune și masă.

## *Codul proiectului*

```
const int analogPin = A0;      //  
const int ledCount = 10;        // Nr. LEDuri  
int ledPins[] = {  
    2, 3, 4, 5, 6, 7, 8, 9, 10, 11  
};    // pinii la care sunt conectati LEDurile  
  
void setup() {  
    // setam toti pinii ca fiind iesiri  
    for (int thisLed = 0; thisLed < ledCount; thisLed++)  
    {  
        pinMode(ledPins[thisLed], OUTPUT);  
    }  
}  
void loop() {  
    // citirea datelor de la potentiometru  
    int sensorReading = analogRead(analogPin);  
    // intervalul de la 0 la nr. de LEDuri  
    int ledLevel = map(sensorReading, 0, 1023, 0,  
ledCount);  
    for (int thisLed = 0; thisLed < ledCount; thisLed++)  
    {  
        if (thisLed < ledLevel) {  
            digitalWrite(ledPins[thisLed], HIGH);  
        }  
        else {  
            digitalWrite(ledPins[thisLed], LOW);  
        }  
    }  
}
```

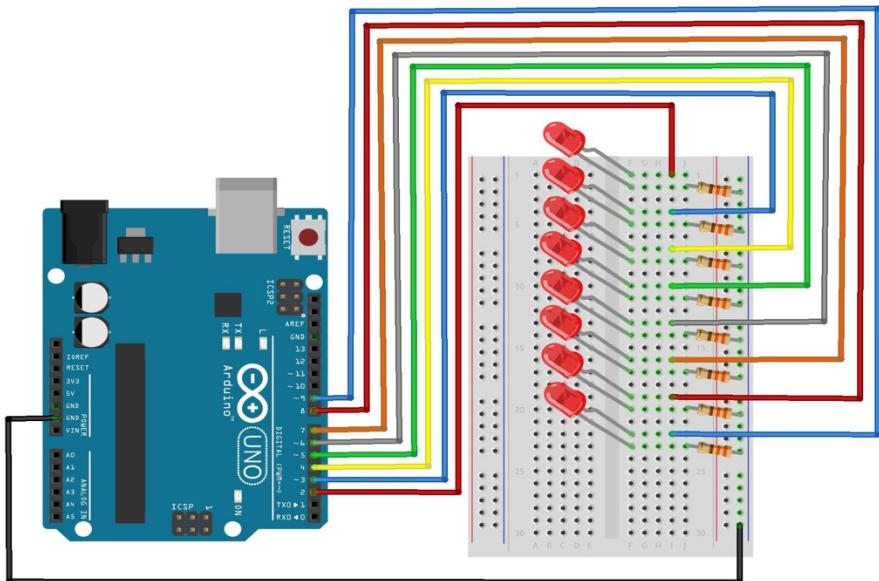
## #8 Multiplicarea LEDurilor

În acest proiect folosesc 8 pini de pe platforma Arduino pentru a clipi 8 LEDuri în același timp. Se pot face mai multe joculețe cu LEDuri, în anexă le vei găsi pe toate.

### *Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- Potențiometru rotativ
- 8 LEDuri 5mm
- 8 Rezistoare 330 Ω
- 9 Fire Jumper

### *Diagrama proiectului*



## *Etapele proiectului*

Conectăm anodul (picior lung) LEDurile la pinii de la 2 la 9 și catodul (picior scurt) la masă (GND) prin rezistoare de  $330\ \Omega$ .

## *Codul proiectului*

```
int ledPins[] = {2,3,4,5,6,7,8,9};      // pinii la care
sunt conectați LEDurile
void setup()
{
    // configurarea pinilor ca fiind de ieșire
    (numeratoarea începe de la 0)

    pinMode(ledPins[0],OUTPUT);    // ledPins[0] = 2
    pinMode(ledPins[1],OUTPUT);    // ledPins[1] = 3
    pinMode(ledPins[2],OUTPUT);    // ledPins[2] = 4
    pinMode(ledPins[3],OUTPUT);    // ledPins[3] = 5
    pinMode(ledPins[4],OUTPUT);    // ledPins[4] = 6
    pinMode(ledPins[5],OUTPUT);    // ledPins[5] = 7
    pinMode(ledPins[6],OUTPUT);    // ledPins[6] = 8
    pinMode(ledPins[7],OUTPUT);    // ledPins[7] = 9
}
void loop()
{
    oneAfterAnother();    // se aprind LEDurile în
    consecutivitate
    //oneOnAtATime();        // se aprind toate odata
    //pingPong();            // fix ca oneAfterAnother,
    doar ca după ce ajunge la ultimul led se schimbă
    direcția
}
```

## *Întrebări/Exerciții*

Modifică codul ca la fiecare 100ms să se aprinde un LED aleatoriu.

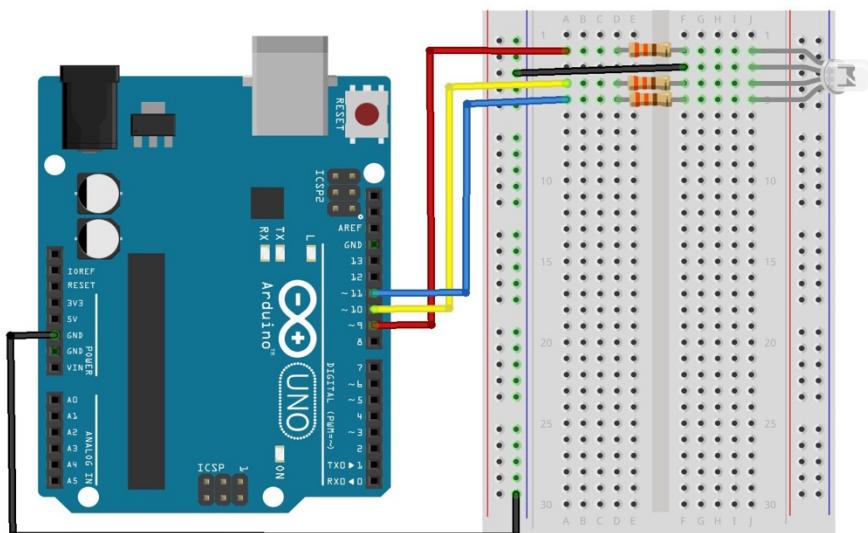
## #9 RGB LED

În acest proiect vom folosi un LED RGB pentru a derula o varietate de culori. RGB înseamnă Red (Roșu), Green (Verde) și Blue (Albastru) și acest LED are capacitatea de a crea combinații de culori aproape nelimitate. Mai multe coduri găsești în anexă.

*Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- RGB LED
- 3 Rezistoare 330 Ω
- 5 Fire Jumper

*Diagrama proiectului*



## Codul proiectului

```
const int RED_PIN = 9;
const int GREEN_PIN = 10;
const int BLUE_PIN = 11;

const int DISPLAY_TIME = 1000; // timp rulare culoarea
void setup() //setarea pinilor ca fiind OUTput
{
    pinMode(RED_PIN, OUTPUT);
    pinMode(GREEN_PIN, OUTPUT);
    pinMode(BLUE_PIN, OUTPUT);
}

void loop()
{
    mainColors(); // combinații de culori
    // showSpectrum(); // gradienți găsești în anexă
}
void mainColors()
{
    digitalWrite(RED_PIN, LOW); // toate LEDurile stinse
    digitalWrite(GREEN_PIN, LOW);
    digitalWrite(BLUE_PIN, LOW);
    delay(DISPLAY_TIME);

    digitalWrite(RED_PIN, HIGH); // culoarea roșie
    digitalWrite(GREEN_PIN, LOW);
    digitalWrite(BLUE_PIN, LOW);
    delay(DISPLAY_TIME);

    digitalWrite(RED_PIN, LOW);
    digitalWrite(GREEN_PIN, HIGH); // culoarea verde
    digitalWrite(BLUE_PIN, LOW);
    delay(DISPLAY_TIME);
//... continuarea găsești în anexă
}
```

## Întrebări/Exerciții

Modifică codul ca toate culorile să fie aprinse, ce culoare e?

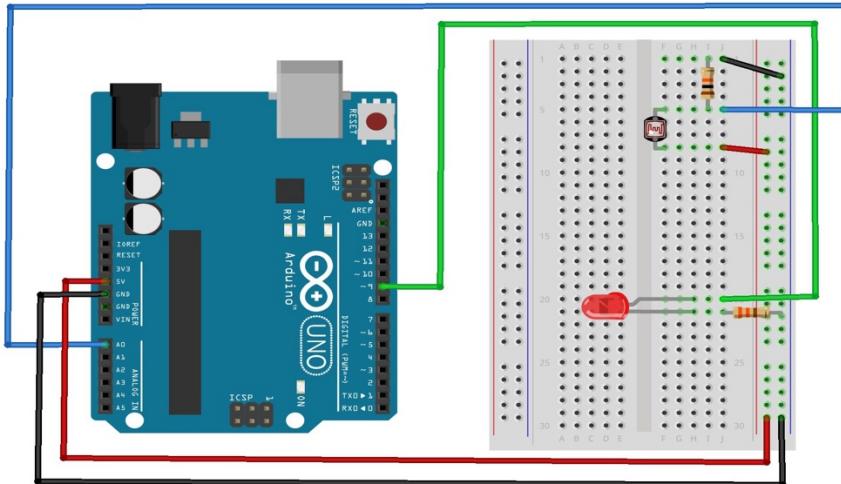
## #10 Foterezistor

Un fotorezistor modifică rezistență pe care un circuit o obține pe baza cantității de lumină la care este supus senzorul. În acest proiect, luminozitatea LED-ului va crește și scădea în funcție de cantitatea de lumină.

### *Componente necesare*

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- LED 5mm
- 1 Rezistor 330 Ω
- 1 Rezistor 10K Ω
- Fotorezistor
- 6 Fire Jumper

### *Diagrama proiectului*



## *Etapele proiectului*

Conecțăm anodul (picior lung) LEDului la pinul 9 și catodul (picior scurt) la masă (GND) prin rezistoare de  $330\ \Omega$ , iar fotorezistorul conectat la tensiune +5V și cealalt capăt conectat la pinul A0 și dus la masă prin rezistor de  $10K\ \Omega$ .

## *Codul proiectului*

```
const int sensorPin = 0;
const int ledPin = 9;
// setam variabile globale pentru nivel de lumină:
int lightLevel;
int calibratedLightLevel; // calibrăm nivel lumină
int maxThreshold = 0;      // nivel maxim lumină
int minThreshold = 1023;    // nivel minim lumină

void setup()
{
    pinMode(ledPin, OUTPUT);      // LED setat OUTput
    Serial.begin(9600);
}
void loop()
{
    lightLevel = analogRead(sensorPin); // citește
    tensiunea de pe senzor
    Serial.print(lightLevel);
    calibratedLightLevel = map(lightLevel, 0, 1023, 0,
    255); // scalare nivel de la 0 la 255
    Serial.print("\t");           // caracter tab
    Serial.print(calibratedLightLevel); // o linie după
    analogWrite(ledPin, calibratedLightLevel); // setează luminozitate LED
}
```

## *Întrebări/Exerciții*

Modifică codul ca LEDul sa se aprinda la intensitate maxima atunci cand pe fotorezistor nu este lumina.

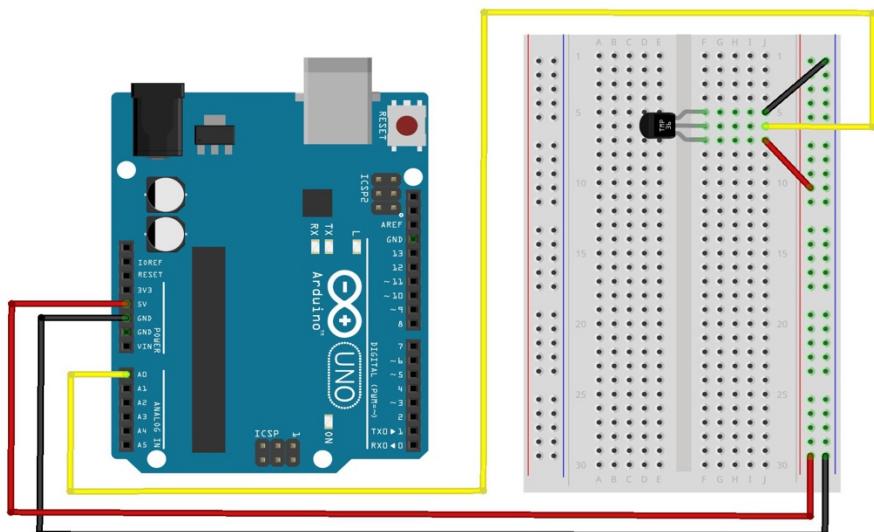
## #11 Senzor de temperatură

Un senzor de temperatură măsoară temperatura ambientă. În acest proiect, vom afișa temperatura în monitorul serial al softului Arduino IDE. TMP36 este un senzor analogic de temperatură, tensiunea este proporțională cu temperatura ambientă. Acest senzor se poate utiliza pentru orice tip de automatizare unde este nevoie de controlat temperatura.

### Componete necesare

- Arduino Uno
- USB A-to-B cablu
- Breadboard
- Senzor temperatură - TMP36
- 5 Fire Jumper

### Diagrama proiectului



## *Etapele proiectului*

Conecțăm pinul din mijloc (pin de semnal) cu pinul A0 de pe Arduino și pinul de alimentare conectăm la tensiune +5V și cealalt pin la masă. Mai mulți parametri ale senzorului TMP36 găsiți în Datasheet - [www.robotica.md/cataloage/tmp36](http://www.robotica.md/cataloage/tmp36)

## *Codul proiectului*

```
const int temperaturePin = A0;
void setup()
{
    Serial.begin(9600); // inițializăm portul serial la
9600 biți pe secundă (bps)
}
void loop()
{
    float voltage, degreesC, degreesF; // declarăm 3
variabile, pentru tensiune, grade C și grade F
    voltage = getVoltage(temperaturePin); // măsoară
tensiunea pe senzor
    degreesC = (voltage - 0.5) * 100.0; // formula de
convertire a tensiunei în grade Celsius
    degreesF = degreesC * (9.0 / 5.0) + 32.0; // formula
de convertire a gradelor Celsius la Fahrenheit

    //Printare date, seteaza Seriala la 9600
    Serial.print("voltage: ");
    Serial.print(voltage);
    Serial.print(" deg C: ");
    Serial.print(degreesC);
    Serial.print(" deg F: ");
    Serial.println(degreesF);
    delay(1000); // reînoire de date fiecare 1 s
}
float getVoltage(int pin) // citire și întoarcere
{
    return (analogRead(pin) * 0.004882814); // convertirea
valorii analogice în tensiune (0 - 5V)
}
```

## Anexa

### Cod jocurile proiect #8

```
int ledPins[] = {2,3,4,5,6,7,8,9};      // pinii la care
sunt conectați LEDurile
void setup()
{
    // configurarea pinilor ca fiind de ieșire
    // (numeratoarea începe de la 0)

    pinMode(ledPins[0],OUTPUT);    // ledPins[0] = 2
    pinMode(ledPins[1],OUTPUT);    // ledPins[1] = 3
    pinMode(ledPins[2],OUTPUT);    // ledPins[2] = 4
    pinMode(ledPins[3],OUTPUT);    // ledPins[3] = 5
    pinMode(ledPins[4],OUTPUT);    // ledPins[4] = 6
    pinMode(ledPins[5],OUTPUT);    // ledPins[5] = 7
    pinMode(ledPins[6],OUTPUT);    // ledPins[6] = 8
    pinMode(ledPins[7],OUTPUT);    // ledPins[7] = 9
}
void loop()
{
    oneAfterAnother();    // se aprind LEDurile în
    consecutivitate
    //oneOnAtATime();           // se aprind toate odata
    //pingPong();                // fix ca oneAfterAnother,
doar ca după ce ajunge la ultimul led se schimbă
direcția
}

// #program 1 LEDuri aprinse consecutiv
void oneAfterAnother()
{
    int index;
    int delayTime = 100; // 100ms pauză între Leduri

    for(index = 0; index <= 7; index = ++index)
    {
        digitalWrite(ledPins[index], HIGH);
```

```
    delay(delayTime);
}
for(index = 7; index >= 0; index = --index)
{
    digitalWrite(ledPins[index], LOW);
    delay(delayTime);
}
}

// #program2 toate LEDurile aprinse odate

void oneOnAtATime()
{
    int index;
    int delayTime = 100; // 100ms pauză între Leduri

    for(index = 0; index <= 7; index = ++index)
    {
        digitalWrite(ledPins[index], HIGH); // aprinde LEDuri
        delay(delayTime); // pauză
        digitalWrite(ledPins[index], LOW); // stinge LEDuri
    }
}
// #program3 joc ping pong
void pingPong()
{
    int index;
    int delayTime = 100; // 100ms pauză între Leduri
    for(index = 0; index <= 7; index = ++index)
    {
        digitalWrite(ledPins[index], HIGH); // aprinde LEDuri
        delay(delayTime); // pauză
        digitalWrite(ledPins[index], LOW); // stinge LEDuri
    }

    for(index = 7; index >= 0; index = --index)
    {
        digitalWrite(ledPins[index], HIGH); // aprinde LEDuri
        delay(delayTime); // pauză
        digitalWrite(ledPins[index], LOW); // stinge LEDuri
    }
}
```

## Cod proiect #9 RGB

```
const int RED_PIN = 9;
const int GREEN_PIN = 10;
const int BLUE_PIN = 11;

const int DISPLAY_TIME = 1000; // timp rulare culoare
void setup() //setarea pinilor ca fiind OUTput
{
    pinMode(RED_PIN, OUTPUT);
    pinMode(GREEN_PIN, OUTPUT);
    pinMode(BLUE_PIN, OUTPUT);
}

void loop()
{
    mainColors(); // combinații de culori
    // showSpectrum(); // gradienți
}
void mainColors()
{
    digitalWrite(RED_PIN, LOW); // toate LEDurile stinse
    digitalWrite(GREEN_PIN, LOW);
    digitalWrite(BLUE_PIN, LOW);
    delay(DISPLAY_TIME);

    digitalWrite(RED_PIN, HIGH); // culoarea roșie
    digitalWrite(GREEN_PIN, LOW);
    digitalWrite(BLUE_PIN, LOW);
    delay(DISPLAY_TIME);

    digitalWrite(RED_PIN, LOW);
    digitalWrite(GREEN_PIN, HIGH); // culoarea verde
    digitalWrite(BLUE_PIN, LOW);
    delay(DISPLAY_TIME);

    digitalWrite(RED_PIN, LOW);
    digitalWrite(GREEN_PIN, LOW);
    digitalWrite(BLUE_PIN, HIGH); // culoarea albastră
    delay(DISPLAY_TIME);
```

```
// culoarea galbena (roșu și verde)
digitalWrite(RED_PIN, HIGH);
digitalWrite(GREEN_PIN, HIGH);
digitalWrite(BLUE_PIN, LOW);
delay(DISPLAY_TIME);

// Cyan (verde și albastru)
digitalWrite(RED_PIN, LOW);
digitalWrite(GREEN_PIN, HIGH);
digitalWrite(BLUE_PIN, HIGH);
delay(DISPLAY_TIME);

// culoarea violetă (roșu și albastru)
digitalWrite(RED_PIN, HIGH);
digitalWrite(GREEN_PIN, LOW);
digitalWrite(BLUE_PIN, HIGH);
delay(DISPLAY_TIME);

}

void showSpectrum()
{
    for (int x = 0; x <= 767; x++)
    {
        RGB(x);           // se incrementează valoarea lui x și
        se modifică nuanța culorii
        delay(10);        // întârziere de timp 10 ms (1/100 din
        secundă) - pentru a ajuta la "netezirea"
    }
}

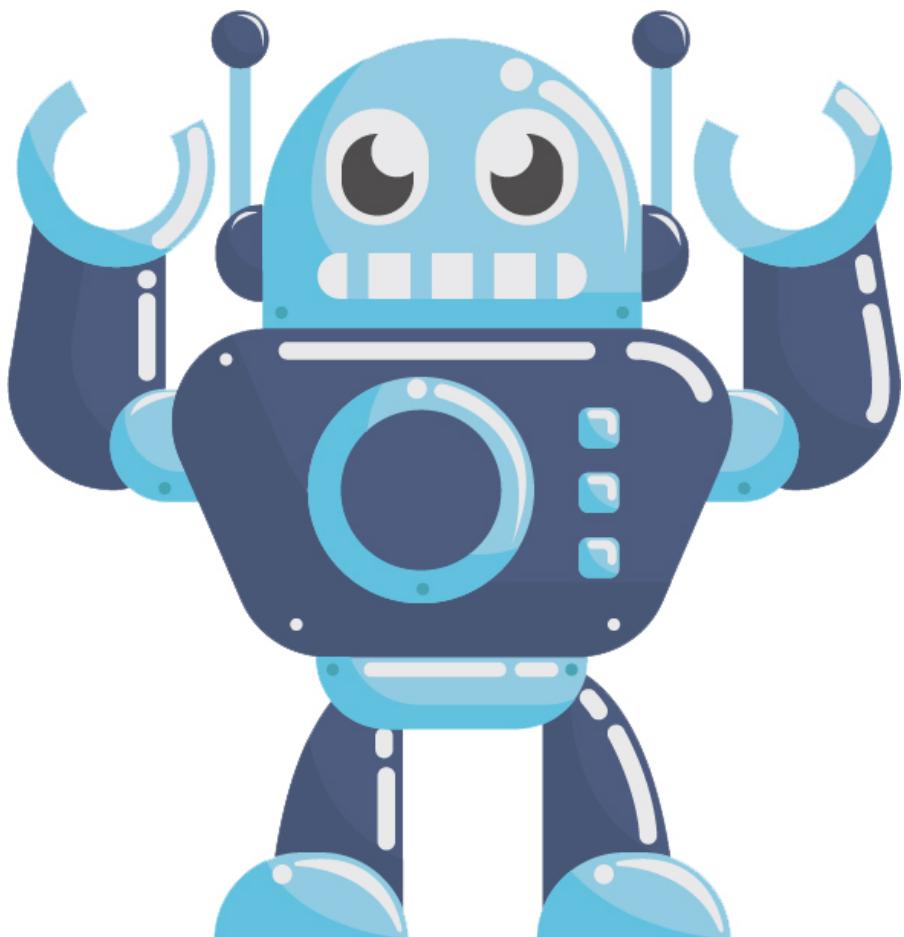
void RGB(int color)
{
    int redIntensity;
    int greenIntensity;
    int blueIntensity;

    color = constrain(color, 0, 767); // constrângerea
    valoarilor de intrare la un interval de la 0 la 767
```

```
if (color <= 255)          // interval 1 de la roșu la
verde
{
    redIntensity = 255 - color;    // roșu merge de la
pornit la oprit
    greenIntensity = color;        // verde merge de la
oprit la pornit
    blueIntensity = 0;            // alabstru este
mereu oprit
}
else if (color <= 511) // (256 - 511) interval 2 de
la verde la albastru
{
    redIntensity = 0;            // roșu mereu
oprit
    greenIntensity = 511 - color; // verde la
pornit la oprit
    blueIntensity = color - 256; // albastru
de la oprit la pornit
}
else                      // interval 3 ( >= 512) de la
albastru la roșu
{
    redIntensity = color - 512; // roșu de la
oprit la pornit
    greenIntensity = 0;        // verde mereu
oprit
    blueIntensity = 767 - color; // albastru de
la pornit la oprit
}

// Se "trimite" valorile de intensitate (roșu, verde,
albastru) folosind pin analogWrite ()
analogWrite(RED_PIN, redIntensity);
analogWrite(GREEN_PIN, greenIntensity);
analogWrite(BLUE_PIN, blueIntensity);
}
```

## Răspunsuri la întrebări



### ***Proiect #1 - Testare Arduino***

Valorea 1000 (milisecunde) reprezintă timpul de pauză între aprinderea și stingerea LEDului, micșorând valoarea dată se micșorează pauza între aprinderea și stingerea LEDului.

Micșorând foarte mult timpul de pauză, spre exemplu la 10 milisecunde, nu se mai observă cum efectiv se stinge Ledul, întradevăr el clipește doar că ochiul uman nu reușește să observe intreruperea lui și ai impresia că este aprins mereu.

### ***Proiect #2 - Clipește un LED***

După ce ai găsit timpul, spre exemplu 100ms, va trebui să calculezi și frecvență, frecvență este inversul perioadei, perioadă în cazul dat este 100ms sau 0,1 secunde =>  
 $1/0,1\text{s}=10\text{ Hz}$

Presupunem că frecvența bătăilor de inimă este 90 bătăi/minut  
=>  $90\text{ bătăi/minut} / 60\text{ secunde} = 1,5\text{ bătăi/secundă} =>$   
 $1000\text{ ms} / 1,5 = 667\text{ ms} =>$  timpul îl setăm la 667 ms.

### ***Proiect #3 - Buton de comandă***

În cod trebuie de modificat doar valoarea lui buttonState din HIGH în LOW, în aşa fel LEDul va fi aprins când butonul nu este acționat și invers.

### ***Proiect #4 - Potențiometru***

Se modifică valoarea lui sensorValue de la 0 la 3000.

## *Proiect #5 - Luminozitatea LEDurilor*

Pentru a mări numărul devizelor de luminozitate la 12, trebuie schimbată valoarea lui fadeAmount de la 5 la 12 și pentru a mări pauza între cicluri trebuie modificată valoarea lui delay de la 30 la 45.

## *Proiect #6 Clipirea LEDurilor*

Dacă timpul de la timer va fi marit de la 100 la 1000 (milisecunde), atunci LEDurile se vor aprinde consecutiv cu o întârziere de 1 secundă în loc de 0,1 secunde.

Pentru a porni proiectul în sens invers va trebui de modificat bucla de initializare, de înlocuit:

```
int thisPin = 2; thisPin < 8; thisPin++ cu  
int thisPin = 7; thisPin >= 2; thisPin--
```

## *Proiect #8 Multiplicarea LEDurilor*

Pentru a afișa un LED în mod aleatoriu, trebuie folosită funcția random, codul final arată așa:

```
void randomLED()  
{  
    int index;  
    int delayTime;  
  
    index = random(8); // numarul de LEDuri  
    delayTime = 100;  
  
    digitalWrite(ledPins[index], HIGH); // aprindere LED  
    delay(delayTime); // pauza  
    digitalWrite(ledPins[index], LOW); // stingere LED  
}
```

### *Proiect #9 RGB LED*

Pentru a aprinde în același timp toate culorile din LEDul RGB, trebuie fiecare culoare în parte să fie aprinsă (HIGH):

```
digitalWrite(RED_PIN, HIGH);  
digitalWrite(GREEN_PIN, HIGH);  
digitalWrite(BLUE_PIN, HIGH);  
delay(DISPLAY_TIME);
```

în aşa fel formăm culoarea albă .

### *Proiect #10 Foterezistor*

Pentru ca Ledul să se aprindă atunci când nu este lumină pe fotorezistor, trebuie modificate valorile de la maxThreshold și minThreshold, în sens invers, adică maxThreshold = 1023 și minThreshold = 0

## **Încheiere**

Felicitări, ai parcurs tot cursul de robotică pentru începători, acum deja ești familiarizat cu mediul Arduino și chiar ești în stare să faci un proiect desinestătător, sunt sigur că ți-am trezit interesul față de domeniul roboticii, ai văzut că poate fi ușor înțeleasă această materie pas cu pas și se pot face multe lucruri interesante și utile.

Deja ai o bază bine formată și poți trece la următoorele aplicații mai complexe și totodată mai interesante, pe [www.kit.robotica.md/](http://www.kit.robotica.md/) găsești și următoarele kit-uri și cursuri care deja îți sunt accesibile, vei opera cu mai mulți senzori (de distanță-vei putea măsura distanța corpuriilor între ele sau identifica prezența obiectelor, senzor de temperatură și umiditate-vei putea crea propria meteo stație, senzor de umiditate-vei putea măsura umiditatea solului la plante și altele), în acest curs ai învățat cum se afișează datele în monitor serial al programului Arduino IDE, în următorul curs vom trece la afișarea externă (display cu 7 segmente și LCD). La fel poți să-ți testezi cunoștințele după parcurgerea acestui kit, accesând linkul - [www.robotica.md/test-kit](http://www.robotica.md/test-kit) și vei primi o reducere pentru achiziționarea următorului kit. Te așteptăm pe platforma noastră [www.robotica.md](http://www.robotica.md)

Îți doresc mult succes!

Cu drag,  
Alexandru Oleinic