

**Ministerul Educației și Cercetării**

**Liceul “Regina Maria ” Dorohoi**

**PROIECT PENTRU OBTINEREA ATESTĂRII  
PROFESIONALE ÎN INFORMATICĂ**

**TITLUL LUCRĂRII:**

**Robot de tip Line Follower & Obstacle Aware**

Profesor coordonator:

**Amarie Bogdan**

Elev:

**Toma Anamaria-Ramona**

**Clasa a-XII-a A**

**Dorohoi**

Mai, 2021

# Cuprins

<b>1.INTRODUCERE.....</b>	<b>3</b>
<b>2.MOTIVAȚIA LUCRĂRII.....</b>	<b>5</b>
<b>3.LISTA DE COMPONENTE .....</b>	<b>6</b>
<b>4.Platforma OPEN ROBERTA LAB .....</b>	<b>9</b>
<b>5.Limbajul NEFO.....</b>	<b>14</b>
<b>6.Construirea robotului .....</b>	<b>20</b>
<b>7.Program – Obstacole.....</b>	<b>33</b>
<b>8.Program- Line Follower .....</b>	<b>34</b>
<b>9.CONCLUZII.....</b>	<b>37</b>
<b>10.Bibliografie.....</b>	<b>38</b>

# INTRODUCERE

Practic tot ceea ce numim produs de înaltă tehnicitate este produs robotic. Automobilul modern , roboții, tehnica de calcul, tehnica de telecomunicații, aparatura biomedicală, sistemele de transport inteligent, aparatura de cercetare, aparatura electrocasnică, aparatura cine-foto și audio-video, mașinile agricole moderne și așa mai departe, sunt exemple reprezentativ de produse robotice.

Robotica este știința care se ocupă cu tehnologia, proiectarea și fabricarea roboților. Robotica necesită cunoștințe de electronică, mecanică și programare, iar persoana care lucrează în acest domeniu a ajuns să fie cunoscută ca robotician. Denumirea de robot a fost introdusă pentru prima oară de către Karel Čapek în anul 1921 în lucrarea sa "Roboții universali ai lui Rossum", în 1921 plecând de la cuvântul ROBOTA, munca, activitate de rutină, preluat de către Isaac Asimov, în povestirea științifico-fantastică "Fuga în cerc" (1941).

Aflată la intersecția unor domenii ale științei cu performanțe de vârf în implementarea noilor tehnologii, robotica abordează concepte și sisteme noi în ingineria micro și nano senzorilor și sistemelor de acționare, materiale și compozite pretabile pentru implementări la scară celulară sau atomică, structuri celulare și rețele neuronale, sisteme ce prefigurează conceptele de nanoelectronică capabile să producă viitoarele nano-procesoare, noi concepte ale inteligenței artificiale privind adaptabilitatea, capacitatea de a raționa, capacitatea de instruire, noi sisteme de conducere axându-se în special pe controlul robust, tolerant la defecte, adaptiv, inteligent, sisteme expert și neuro-fuzzy și lista poate continua.

Sistemele robotice asigură:

- ➔ multifuncționalitate
- ➔ flexibilitate
- ➔ posibilitatea acționării la distanță
- ➔ evoluție continuă datorată dinamicii cerințelor pieții
- ➔ imitare a naturii-adaptabilitate

Automatizarea, ca precursor a unor roboți autonomi, vine și ea din Antichitate. Oamenii au dorit dintotdeauna să creeze lucruri care se mișcau automat, fie pentru amuzament, ca un semn al puterii sau în scop practic. Ceasurile sunt un asemenea exemplu, iar nevoia de a măsura timpul a existat mereu. Nu le-am încadrat acum la precursori ai roboților, dar, în fapt, nu poți crea un mecanism automatizat fără un soi de ceas intern. Ce-i drept, prea puține lucruri se știu despre automatele Antichității, sunt mai degrabă povești. Majoritatea erau mecanisme simple, acționate în general prin forța apei sau a aburului.

Era o perioadă în care se încerca imitarea vieții cu ajutorul unor dispozitive mecanice, motiv pentru care numeroase asemenea automatoane aveau trăsături umane. Acestea erau însă doar curiozități de carnaval, spectaculoase pentru vremea lor, dar primitive pentru ce așteptăm acum de la un robot. Adevărata revoluție se petrecea în fabrici, unde apăreau primele mașini care înlocuiau munca omului și care dădeau startul revoluției industriale.

Robotica începe cu Leonardo Da Vinci, un personaj atât de complex că nu-l poate ocoli niciun domeniu tehnologic, trece prin Čapek și Asimov, „inventatorii” roboților și roboticii, și îți va face cunoștință cu părinții teoretici ai domeniului, Alan Turing și Norman Wiener, creatorul ciberneticii, dar și cu primul inventator al unui robot „adevărat”, unul industrial, George Devol.

Termenul de robot descrie un domeniu destul de vast, cauză din care roboții sunt sortați în multe categorii. Iată câteva din acestea:

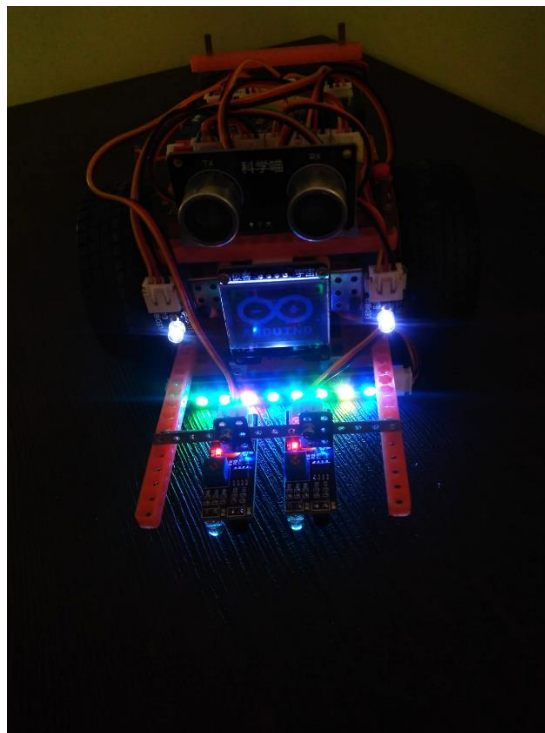
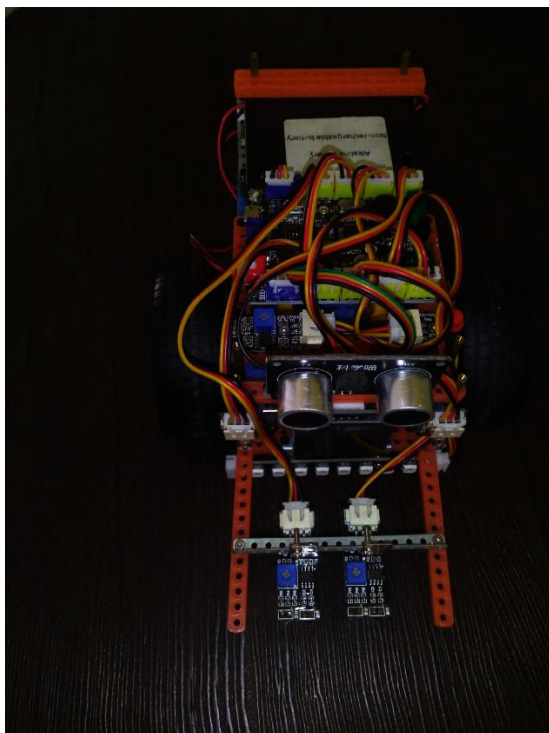
- ➔ Robot autonom mobil
- ➔ Robot umanoid

- Robot industrial
- Robot de servicii
- Robot jucărie
- Robot explorator
- Robot pășitor
- BEAM
- Robot militar

## MOTIVAȚIA LUCRĂRII

Robotica este în prezent o ramură a științelor tehnice, având ca obiect de studiu automatizarea operațiilor umanoide. În corelație cu definiția dată, am decis să realizez un robot ce urmărește o linie neagră pe un fundal alb, ce are ca scop parcurgerea întregului traseu determinat de linia neagră. Am văzut acest proiect la diverse prezentări de electronică și considerăm că este un prilej bun să acceptăm această provocare. Acest tip de robot (Line Follower Robot) are aplicații practice, punând în evidență tehnologia folosită în depozitele gigantice, sau în fabricile de asamblare, unde piesele trebuie să traverseze distanțe relativ mari. Cu ajutorul unor marcaje, roboții pot urmări aceste trasee pentru a transporta eficient și rapid cele necesare. Ideea robotului care ocolește obstacole (Obstacle Aware Robot) își are rădăcinile în aspiratoarele automate care se găsesc în magazine. Mi-am propus să realizez un astfel de robot, care să se poată plimba și să ocolească obstacolele, la fel ca un aspirator inteligent. Un robot care ocolește obstacolele poate fi folosit pentru a detecta suprafața unei camere sau pentru a-i desena harta.

Mi s-au părut interesante ambele idei, iar din motivul că nu m-am putut decide, am hotărât să le fac pe ambele. Am combinat senzori necesari pentru ambele tipuri de acțiuni și am creat două programe, separate, care vor determina robotul să-și atingă scopul.



## LISTA DE COMPONENTE

### Module:

- Placa Arduino;
- Bara de leduri;
- Ecran LCD;
- Modul drive;
- Power splitter;
- Senzor ultrasonic pentru distanță;
- Senzor de luminiozitate;
- Buton și leduri suplimentare(faruri);

### Accesorii:

- Două motoare;
- Două suporturi de prindere;
- O roată de stabilizare;
  
- Un cablu din 4 fire de 10 cm;
- Patru cabluri din 2 fire de 10 cm;
- Șase cabluri din 3 fire de 10 cm;
- Două cabluri din 3 fire de 20 cm;
  
- Patru suporturi cu 4 găurii;
- Două suporturi cu 6 găuri;
- Două suporturi cu 8 găuri;
  
- Patru piloni de 3,5 cm;
- Patru piloni de 2,5 cm;
- Paisprezece piloni de 1cm;
- Doi piloni de 1 cm cu dublă intrare;
  
- Două benzi glue 3M;
- Un buton roșu;
- Cinci distanțieri portocalii;
- Cinci distanțieri transparenți;
  
- 6 șuruburi 0,6 cm;
- 23 șuruburi 0,8 cm;
- 26 de șuruburi de 1 cm;
- 50 piulițe;



- Două roți;
- Suportul pentru baterii;
  
- Un bloc de 7,5 cm X 6 cm;
- Două blocuri de 4 cm X 1 cm;
- Două blocuri de 7,5 cm X 0,5 cm;
- Două blocuri de 12 cm X 7,5 cm;
  
- Un cablu de conectare USB;
- Două șurubelnițe cu cap hexagonal;
- Trei cabluri din 3 fire lipite;
  
- Două suporturi cu 8 găuri;
- Două suporturi cu 22 găuri;
- Două suporturi cu 30 găuri;
- Doi piloni de 1,5 cm;
- Opt șuruburi de 1 cm;
- 20 de piulițe;
  
- Un buton;
- Doi senzori de linie;
- Două blocuri de 7,5 cm X 1 cm;
- Două blocuri de 7,5 cm X 0,5 cm;
- Două blocuri în formă de V;
- Un suport mare de prindere;





# Platforma OPEN ROBERTA LAB

## ➔ Meniurile platformei OPEN ROBERTA LAB

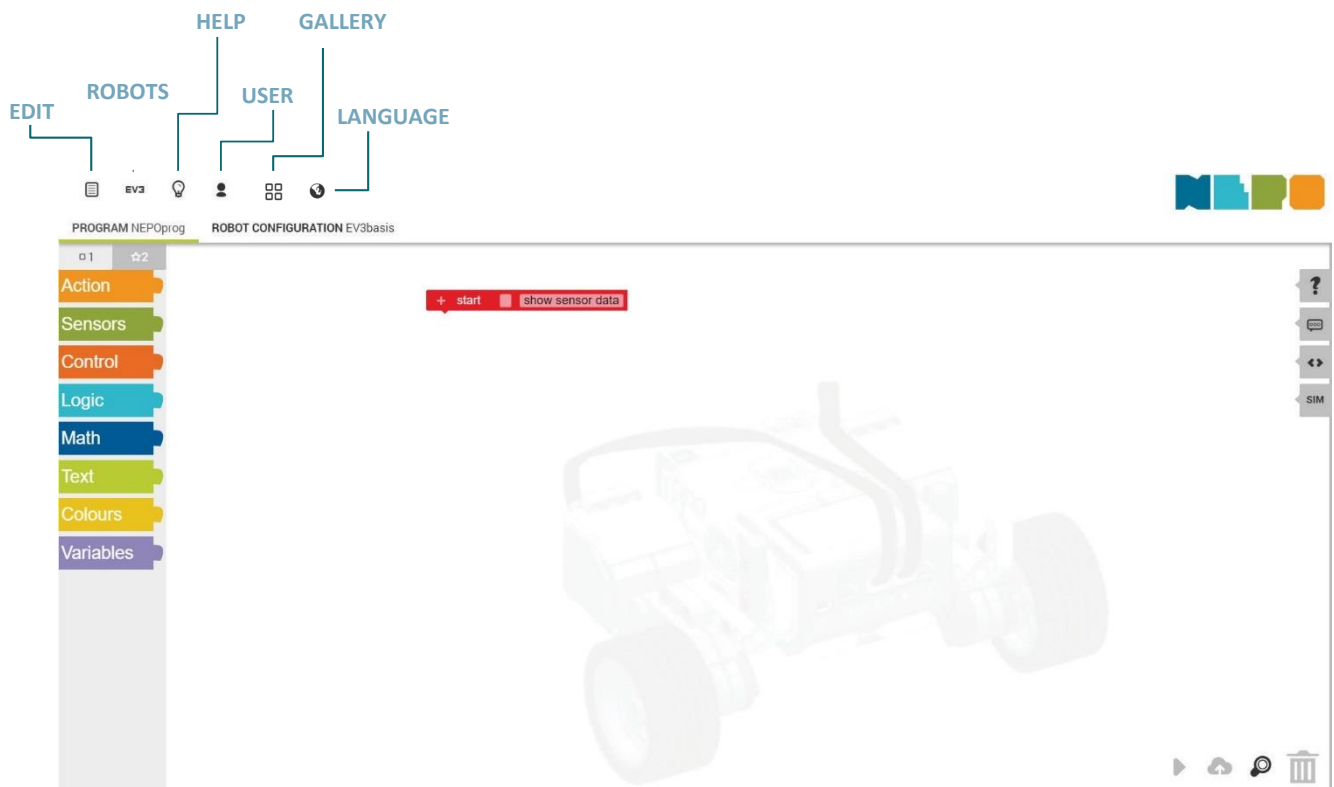


Utilizarea unui mediu de programare pentru implementarea algoritmilor

Explorarea elementelor de interfață ale unui mediu de programare în scopul identificării principalelor facilități ale acestuia

Fereastra principală conține în stânga sus 6 meniuri:

- Edit
- Help
- Gallery
- Robots
- User
- Language





Acest element cuprinde un meniu în care veți găsi tot ce puteți face cu programul OPEN ROBERTA LAB: creare proiect, export/import proiect, exemple, vizualizare cod sursă etc. În plus, puteți alege blocurile NEPO pe care doriți să le programați, de exemplu, blocuri pentru începători sau blocuri pentru experți.



### *RUN ON ROBOT*

Dacă faceți clic pe acest submeniu, robotul execută programul. Programul și configurația robotului vor fi trimise la serverele Open Roberta Lab unde este compilat și trimis robotului. Robotul rulează programul imediat ce este complet transferat. Dacă robotul real este conectat, va primi un mesaj direct.

### *RUN IN SIMULATION*

Această comandă pornește programul în mediul de simulare. Programul va fi trimis la simulator și poate fi executat cu butonul de pornire al simulării. Configurația robotului este o configurație standard.

### *NEW...*

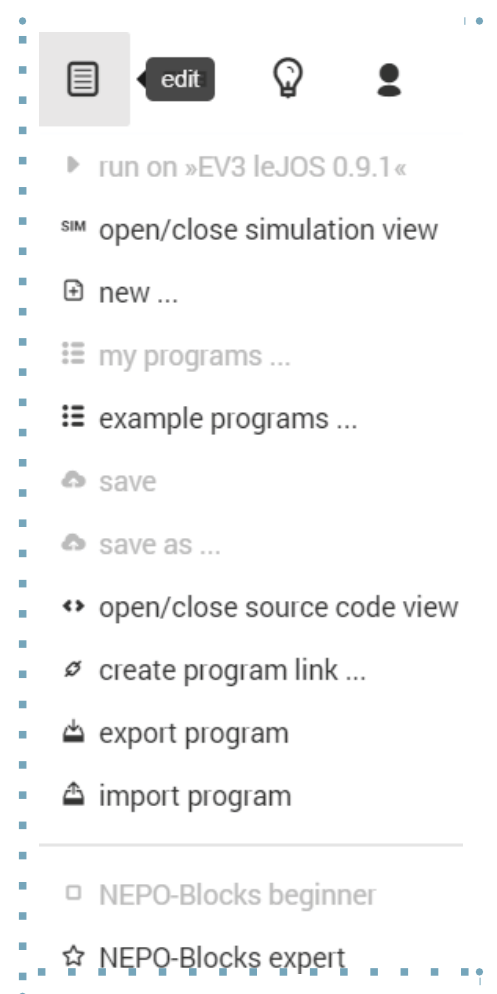
Creați un nou program. Dacă programul actual a fost modificat, va exista un avertisment că ar trebui să salvați modificările.

### *MY PROGRAMS ...*

Afișează toate programele salvate. Pentru aceasta, trebuie să vă conectați la meniul user sau să creați un nou cont de utilizator cu new.

### *EXAMPLE PROGRAMS ...*

Oferă exemple model pentru primele programe.



## <> *SHOW CODE*

O fereastră de coduri va fi rulată pentru a afișa codul Java carea fost generat din programul cu blocuri.

## *CREATE PROGRAM LINK*

Generează link-ul programului curent. Îl poți partaja sau include în alte pagini web.

## *SAVE*

Dacă programul a primit deja un nume cu save as, se vor salva modificările programului. Aceeași funcționalitate este disponibilă utilizând butonul din colțul dindreapta jos al ecranului.

## *SAVE AS ...*

Programul e salvat cu un nume nou.

## *EXPORT PROGRAM*

Utilizarea comenzii export este folosită pentru a stoca programul curent pe un dispozitiv de stocare local. Codul XML generat va fi stocat în folderul dorit. Acest lucru este întotdeauna util dacă cloudul Open Roberta nu este disponibil.

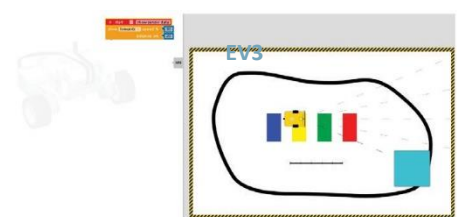
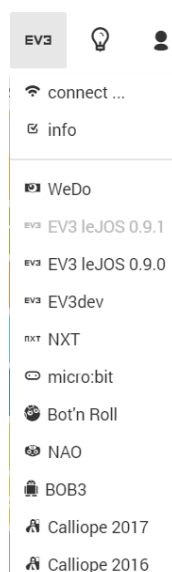
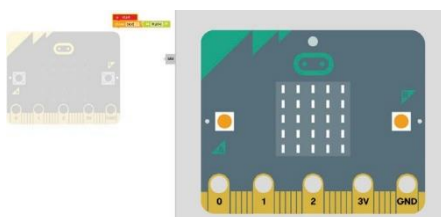
## *IMPORT PROGRAM*

Comanda import este utilă pentru restaurarea unui program stocat pe un dispozitiv de stocare local.

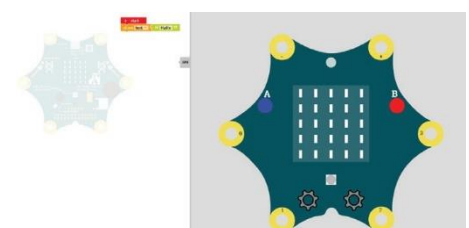
Meniul Robots comută între simulatoarele diferitelor tipuri de roboți.



MICROBIT



CALLIOPE



## *HELP MENU*

Meniul Ajutor trimite spre Open Roberta Wiki. În plus, sunt disponibile următoarele submeniuri suplimentare:

### *GENERAL HELP*

Aceasta este o legătură directă în wiki-ul Open Roberta, care oferă toată informația referitoare la proiect și programarea cu NEPO.

### *FAQ*

Întrebări frecvente și răspunsuri la acestea.

### *ABOUT*

Aici puteți găsi câteva informații despre proiectul Open Roberta și despre Institutul Fraunhofer IAIS.

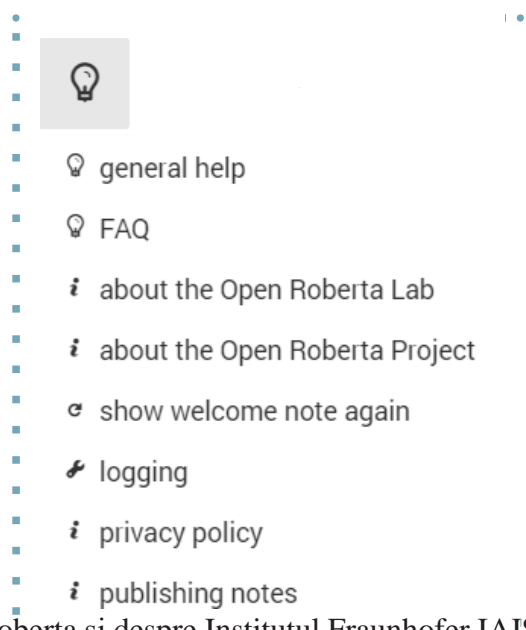
### *LOGGING*

Afișarea tuturor acțiunilor efectuate în cadrul platformei Open Roberta Lab.

Aici puteți găsi câteva informații despre proiectul Open Roberta și despre Institutul Fraunhofer IAIS.



Dacă faceți clic pe acest submeniu, se va afișa din nou mesajul de întâmpinare. Aici sunt prezentate cele mai recente informații despre ultima versiune a platformei Open Roberta Lab.



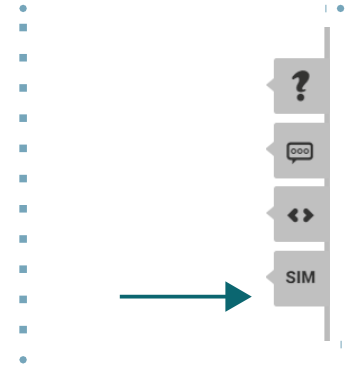
## Simulatorul OPEN ROBERTA SIM

Mediul de simulare Open Roberta permite testarea programelor. Acest lucru este foarte util dacă nu există un robot real la îndemână.

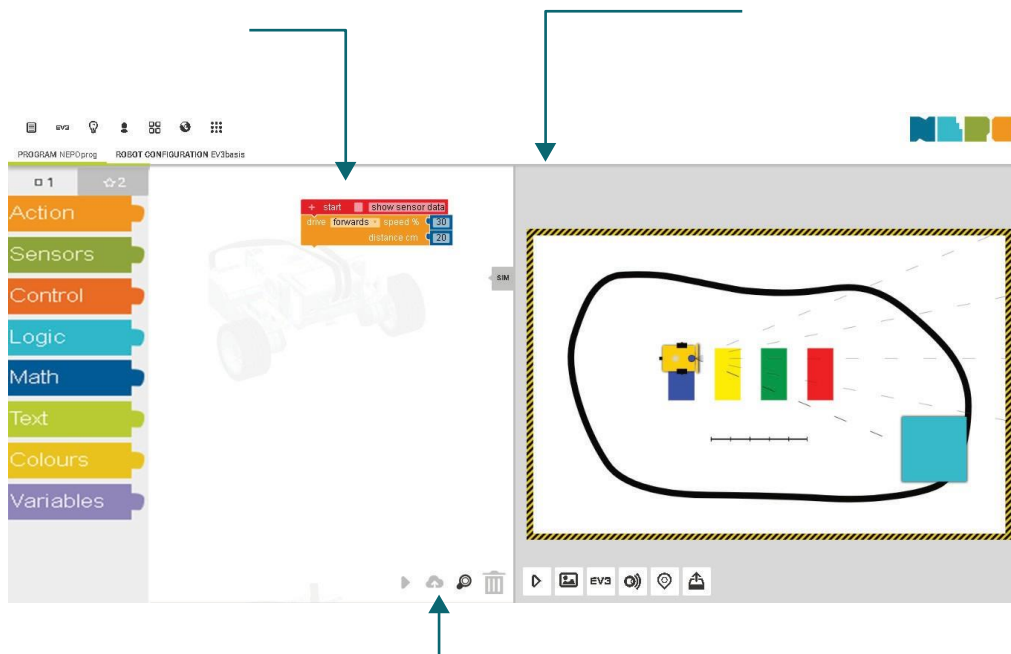
Programele OPEN Roberta pot fi rulate pe roboți reali sau în mediul de simulare. În simulare este disponibil un model simplu de robot 2D.

Mediul de simulare pentru robotul 2D va fi afișat făcând clic pe butonul SIM din dreapta mediului de programare. Același buton SIM poate fi utilizat pentru a închide mediul de simulare. În mediul de simulare există mai puține blocuri de programare executabile decât în mediul EV3. Acest lucru sedatorează modelului robot simplu disponibil în prezent în mediul de simulare.

Când dați clic pe butonul de pornire a simulării pentru a rula un program în mediul de simulare, fereastra de afișare se modifică ca în figura de mai jos.



În panoul din stânga este afișată o parte a programului.



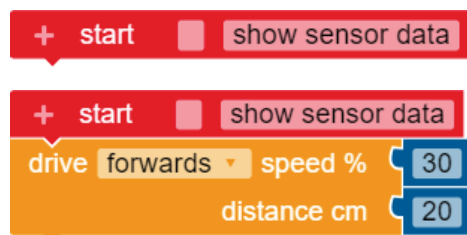
## Limbajul NEFO

NEPO (New Easy Programming Online) este un limbaj de programare din cadrul platformei Open Roberta Lab care utilizează biblioteca Blockly la care au fost adăugate funcționalități suplimentare și îmbunătățiri, adaptate pentru Open Roberta Lab. Paradigma de programare a NEPO este inspirată de Scratch de la Massachusetts Institute of Technology.

Un bloc NEPO reprezintă și încorporează o anumită funcționalitate a robotului. Blocuri sunt organizate pe categorii, în funcție de acțiunile pe care le pot programa, de exemplu senzori. Blocurile sunt interconectate și vor fi executate de robot conform ordinii lor, începând cu blocul start. Acest principiu se numește programare secvențială.

În programarea cu NEPO, fiecare program începe cu blocul start. Acest bloc este adăugat întotdeauna în mod implicit în spațiul de lucru Open Roberta Lab. Primul bloc pe care îl veți executa este conectat cu conectorul de secvență al blocului start.

Conectorul de secvență este un triunghi în partea de jos a blocului și acest triunghi va deveni galben imediat ce un bloc adecvat este în imediata apropiere, după cum puteți vedea în imaginea alăturată.



### ➔ BLOCURILE NEPO

Open Roberta Lab oferă două opțiuni atunci când selectează blocurile existente: Începător și Expert. Modul Începător este cel implicit. Această categorie include 8 blocuri care sunt cele mai importante pentru programarea unui robot. Modul Expert include în plus trei categorii de blocuri: “Liste”, “Funcții” și “Mesaje”.

CATEGORIE	DESCRIERE
ACTION	include blocuri pe care robotul poate să le execute direct
SENSORS	conține blocuri pentru toți senzorii standard ai sistemului EV3
CONTROL	include blocuri pentru controlul secvenței programului: dacă __ atunci __ dacă __ atunci __ altfel __ repetăți la infinit repetăți de n ori așteaptă așteaptă până când

CATEGORIE		DESCRIERE
	<b>LOGIC</b>	cu blocurile logice pot fi create condiții cu ajutorul cărora se pot interconecta între ele stări, valori și evenimente
	<b>MATH</b>	conține operatori matematici și blocuri de parametri
	<b>LIST</b>	include blocuri pentru a crea o listă și pentru a căuta sau sorta elemente din listă
	<b>TEXT</b>	include blocuri pentru a scrie texte pe afișajul robotului
	<b>COLOURS</b>	conține blocuri standard de culoare pentru a compara intrările senzorilor
	<b>VARIABLES</b>	conține blocuri cu ajutorul cărora pot fi definite variabile locale și globale
	<b>FUNCTIONS</b>	conține blocuri cu ajutorul cărora pot fi definite funcții cu parametri de intrare și ieșire
	<b>MESSAGES</b>	include blocuri pentru a trimite și a primi mesaje bluetooth
	<b>PROGRAM-START</b>	fiecare program începe cu acest bloc; acest bloc este întotdeauna disponibil în spațiul de lucru

## ➔ MENIUL CONTEXTUAL

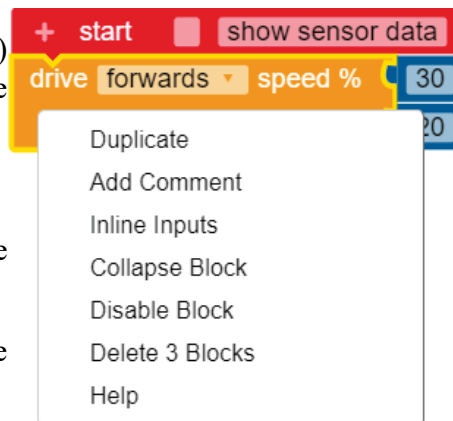
Prin clic dreapta pe un bloc, poate fi deschis un meniu contextual. Utilizând meniul, pot fi efectuate mai multe acțiuni:

Duplicare (copiază blocul și parametrii de intrare conectați)  
Adaugă comentariu (adaugă un comentariu la acest bloc) Ințrări inline  
(modifică aspectul blocului)

Restrângeți blocul (minimizează blocul)

Dezactivați blocul (dezactivează blocul astfel încât acesta să nu fie executat)

Ștergeți 3 Blocuri (șterge blocurile selectate și pe toate cele blocate)



## ➔ CONECTORI NEPO DE INTRARE/IEȘIRE

Conectorul permite blocurilor NEPO să fie inserate în alte blocuri. În total, există șase tipuri de intrări și ieșiri în cadrul NEPO. Aceste tipuri pot avea următoarele culori:

TIP	CULOARE
VALOAREA LOGICĂ	albastru deschis
NUMĂR	albastru închis
STRING / TEXT	verde
CULOARE	galben
LISTĂ	purpuriu
CONEXIUNE	roz



## ➔ PARAMETRII

Valorile pot fi transferate de la un bloc la altul. Tipul valorii transferate poate fi identificat de conectorul unui bloc. În capitolul Categoriilor de blocuri puteți vedea diferitele valori pe care un bloc le poate avea. Blocurile pot fi conectate numai când culorile conectorului de intrare și ieșire se potrivesc. Un bloc poate, de asemenea, să treacă opțional o valoare (numai una) unui alt bloc. Aceste blocuri au conectorii de ieșire colorați.

O caracteristică specială a blocurilor de senzori în comparație cu blocurile de acțiune este faptul că multe blocuri de senzori returnează o valoare. Tipul de valoare returnat de un bloc de senzor corespunde culorii conectorului.

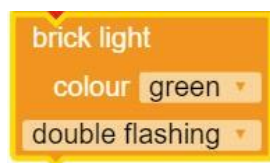


## Categoriile de blocuri

### ➔ CATEGORIA ACȚIUNE (ACTION)

Categoria Acțiune conține blocurile care efectuează o acțiune asupra robotului LEGO MINDSTORMS EV3:

- ➔ Blocuri pentru controlul motoarelor
- ➔ Blocuri pentru a controla afișajul
- ➔ Blocuri pentru a controla lumina de stare
- ➔ Blocuri pentru a controla difuzorul



### ➔ CATEGORIA MESAJE (MESSAGES)

Doi roboți EV3 pot schimba mesaje. Este necesar ca Bluetooth să fie activat pe roboții EV3 participanți. Unul dintre roboții implicați stabilește conexiunea cu celălalt robot (bloc expert: se conectează la numele robotului).

Celălalt robot așteaptă un apel de conectare (bloc expert: așteptați conexiunea). Aceasta funcționează similar cu apelurile telefonice: un apelant formează numărul unui partener de comunicare. Când este stabilită conexiunea, roboții pot schimba mesaje în ambele direcții.

## → CATEGORIA SENZORI (SENSORS)

OPEN ROBERTA LAB conține blocuri pentru următorii senzori:


- senzor de atingere
- senzor de culoare/lumina
- senzor ultrasonic
- senzor de infraroșu
- senzor de rotație (codificator motor)
- giroscopic
- butoane
- timer

## → CATEGORIA CULORI (COLORS)

Categoria culori oferă blocurile de culoare necesare senzorului de culoare.



### → Senzorul de culoare/lumină

Cu blocul  se poate transmite unui alt bloc care este culoarea senzorului. În plus, acest bloc oferă în meniul drop-down setările lumină (light), RGB și lumina ambientală (ambient light).



Aceste trei setări suplimentare transmit toate aceleași date de tip numeric. Valorile numerice sunt între 0 (negru) și 100 (alb). Cu aceste setări diferite, acest bloc poate fi configurat în funcție de cerințele specifice.

În modul color, senzorul emite lumină și detectează culoarea de bază de sub senzor. Recunoaște 7 culori (negru - BLACK, maro - BROWN, albastru

- BLUE, verde - GREEN, galben - YELLOW, roșu - RED, alb - WHITE) și fără culoare - GRAY.

În modul lumină, senzorul emite lumină prin LED-ul său roșu și măsoară intensitatea luminii reflectate pe o scară de la 0 la 100, (0 = foarte întunecată și 100 = foarte luminoasă).

În modul lumină ambientală se utilizează aceeași scală (0-100) ca în modul lumină. În imaginea alăturată se măsoară lumina ambientală care este percepută de senzor.

### → **Senzorul tactil/de atingere**

Cu blocul senzor tactil/de atingere puteți transmite unui alt bloc valori dacă senzorul tactil este apăsat sau nu. Acest bloc returnează valorile logice true dacă e în modul apăsat sau false dacă nu e apăsat. Acest bloc poate fi utilizat numai împreună cu un alt bloc care necesită o valoare logică ca parametru de intrare, de exemplu împreună cu blocul dacă da...atunci.

În mediul OPEN Roberta LAB senzorul tactil este apăsat dacă întâlnește un obstacol. În fiecare scenă, marginea este un obstacol fix, pătratul albastru din Simple și Drawing Scene este un obstacol mobil ce poate fi deplasat prin drag and drop.

### → **Senzorul ultrasonic**

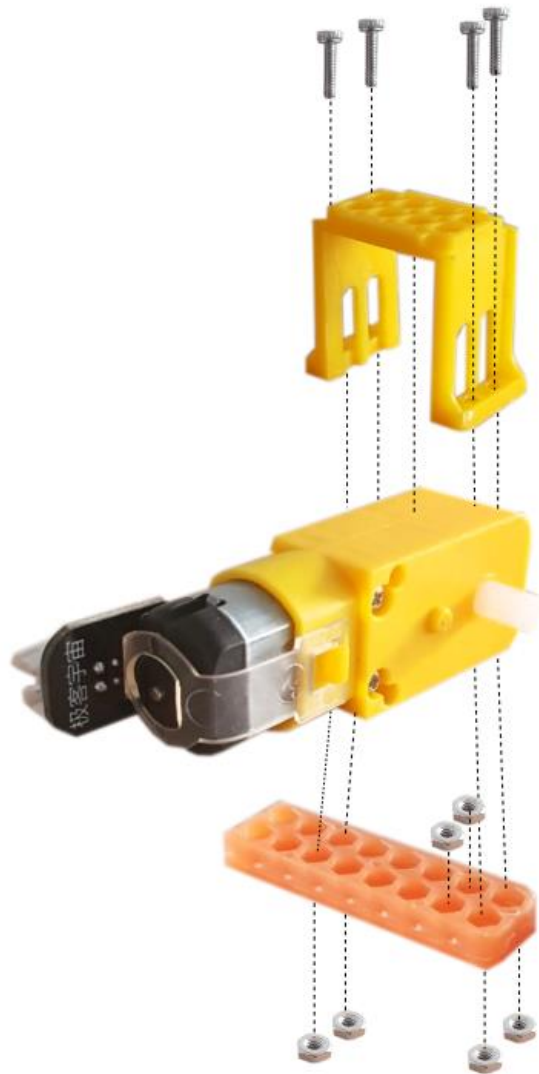
Cu blocul obține distanța/prezența se transmite unui alt bloc distanța măsurată de senzorul ultrasonic. Distanța este transmisă ca un număr, în cm.

În plus, acest bloc poate fi setat la “prezență” din meniul derulant. Această setare poate verifica dacă un alt senzor ultrasonic este activ. Această setare este specifică valorii logice true dacă un alt senzor ultrasonic este prezent sau false dacă niciun alt senzor ultrasunet nu este prezent.

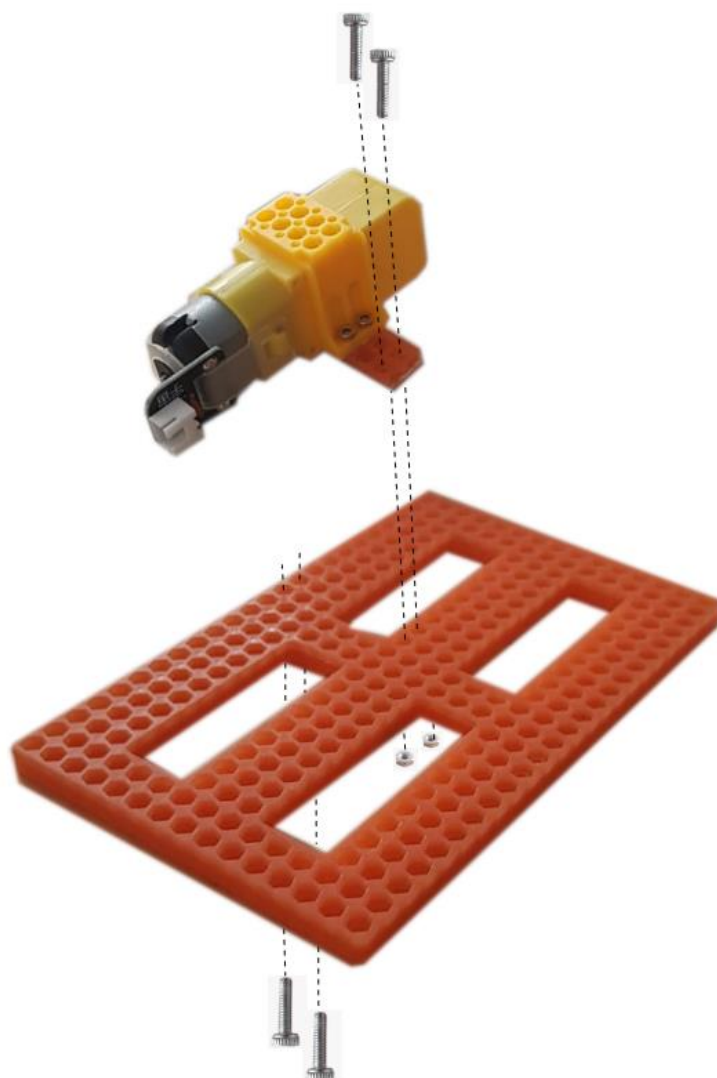
Dacă valoarea nu a fost trimisă de propriul senzor ultrasonic, poate duce la măsurători eronate. Dacă a fost selectat “distanța”, un număr care indică distanța în cm dintre senzorul ultrasonic și obstacol, se transmite unui alt bloc

## Construirea robotului

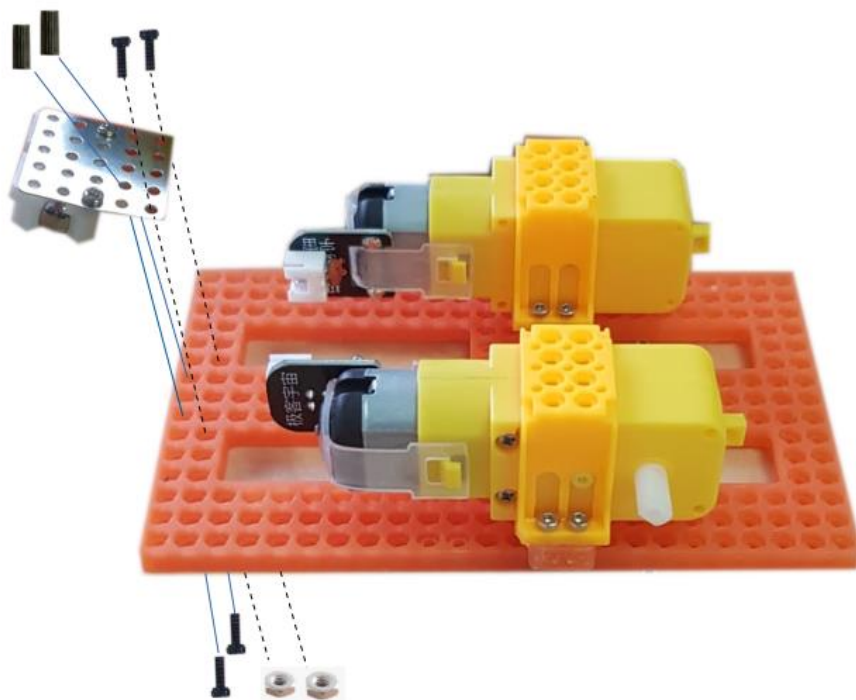
- ➔ **Pasul 1 - Atașarea motoarelor-** Montarea motoarelor pe suporturile portocalii mici cu ajutorul unei șurubelnițe hexagonale.



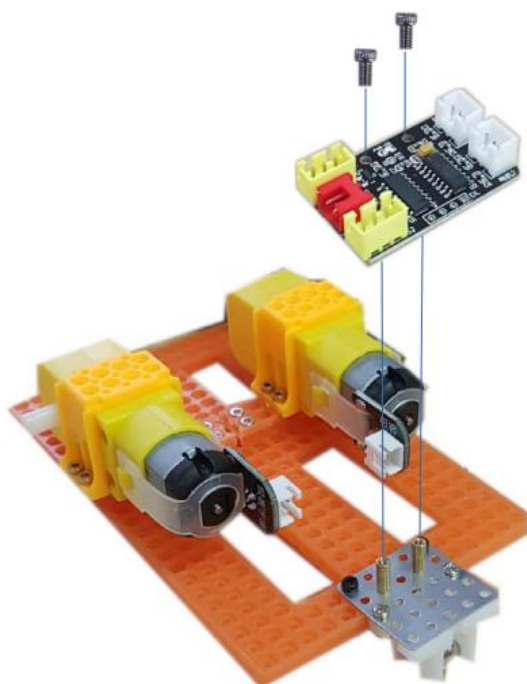
- ➔ **Pasul 2 – Montarea ansamblului motor**



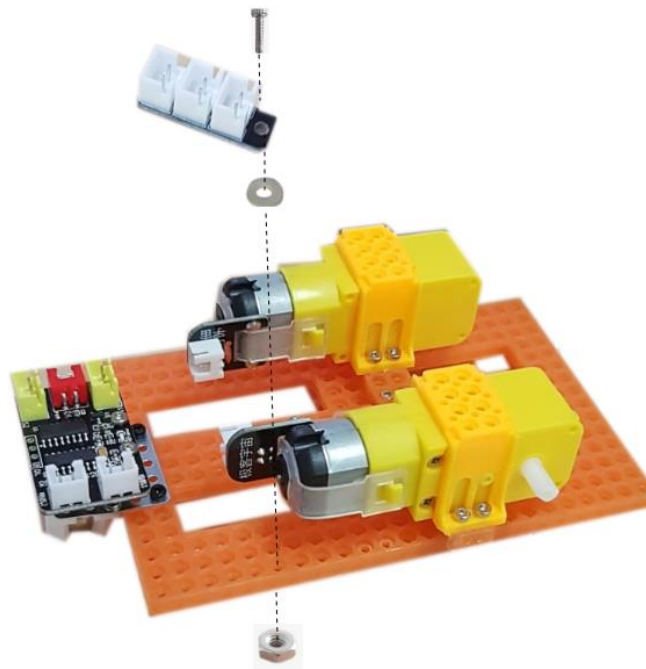
➔ **Pasul 3 – Montarea suportului cu bilă :** Suportul cu bilă va menține capătul posterior al robotului ridicat de la sol și vine cu avantajul suplimentar al unei mari mobilități.



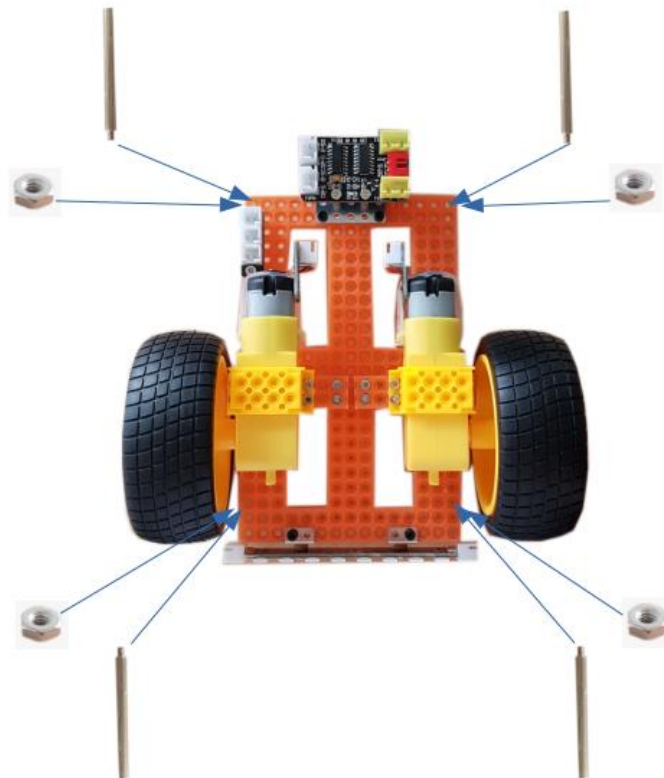
➔ **Pasul 4 - Montarea plăcuței de control a motoarelor :** Plăcuța de control a motoarelor ne permite să controlăm viteza acestora.



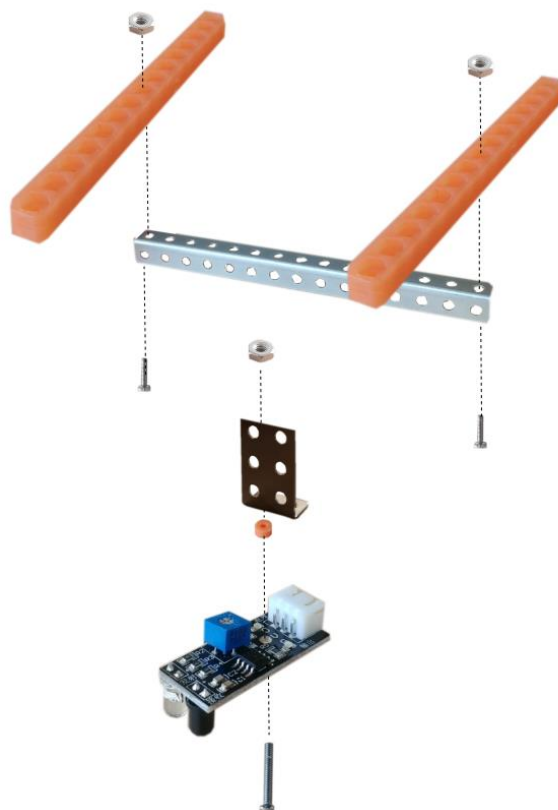
- ➔ **Pasul 5 - Montarea plăcuței de alimentare :** Plăcuța de alimentare ne permite să furnizăm curent electric atât plăcuței de control a motoarelor cât și plăcuței arduino în același timp .



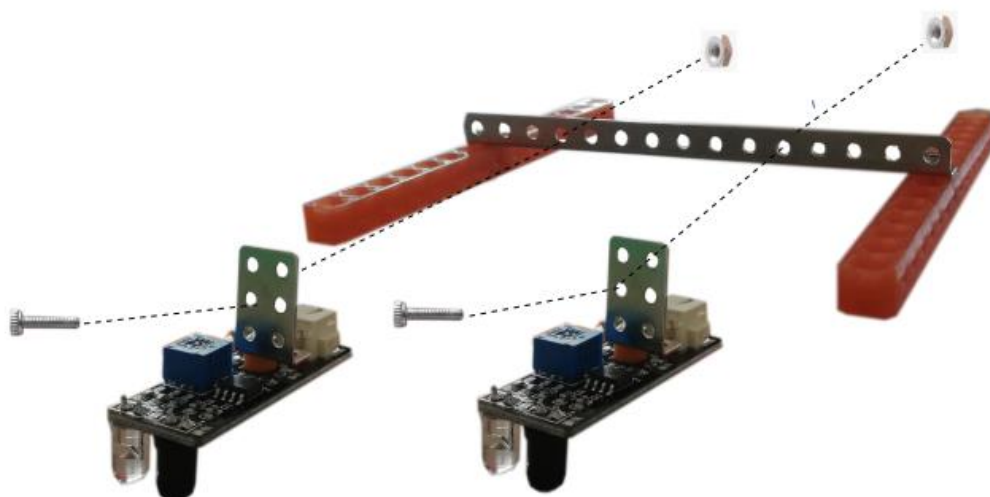
- ➔ **Pasul 6 - Atașarea pilonilor :**



➔ **Pasul 7 - Cadrul de susținere :**

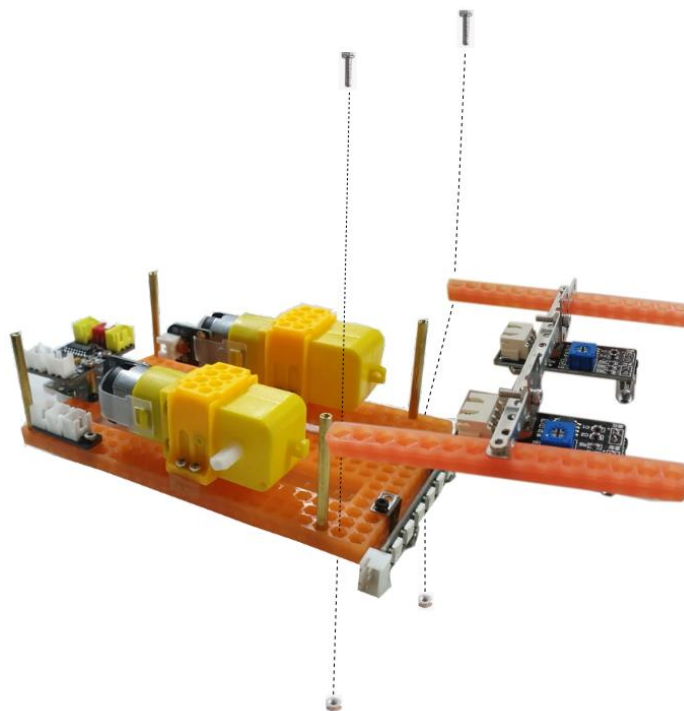


➔ **Pasul 8 - Montarea senzorilor :** Potențiometrul se folosește pentru a calibra senzorul de linie în funcție de traseu și condițiile de iluminare.

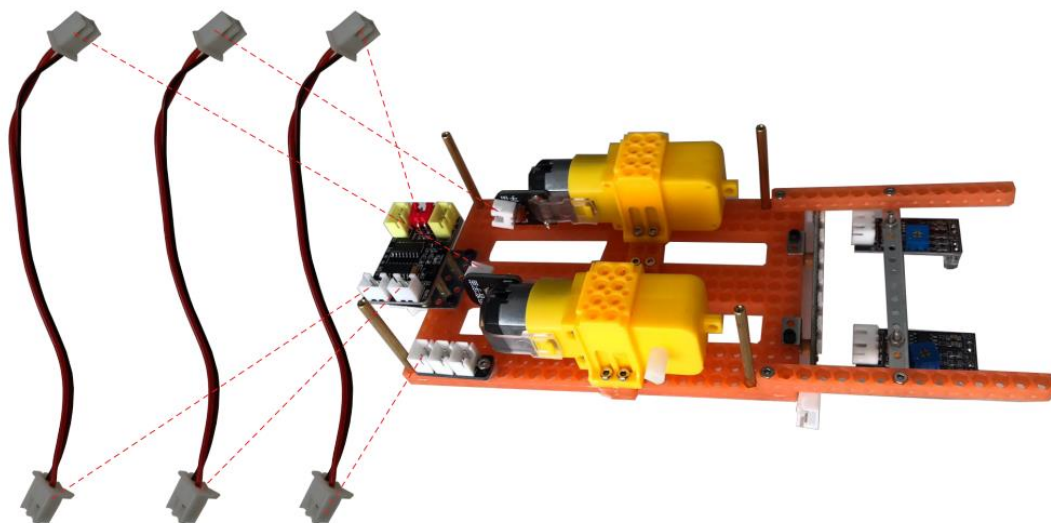




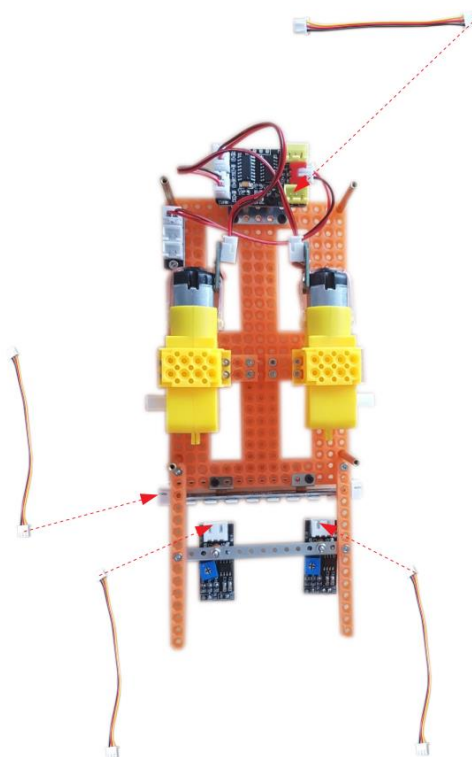
➔ **Pasul 9 - Montarea ansamblului :**



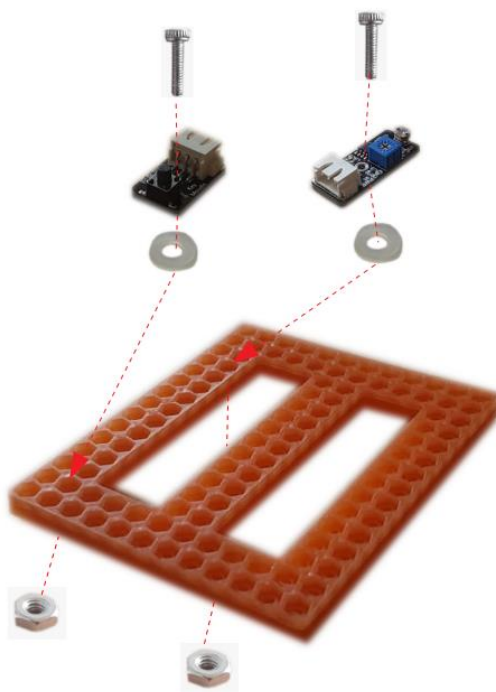
➔ **Pasul 10 - Cablare Motoare :**



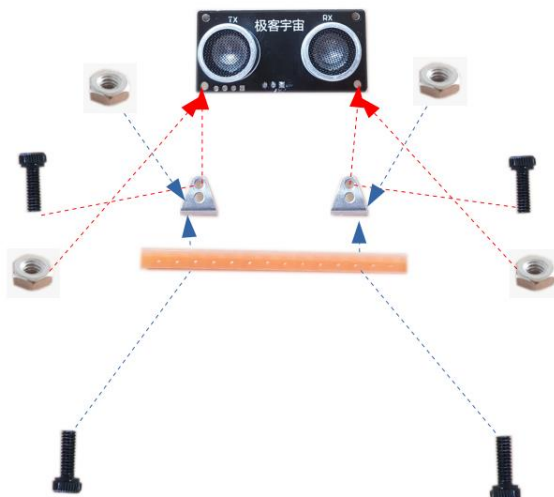
➔ **Pasul 12 - Cablare senzori :**



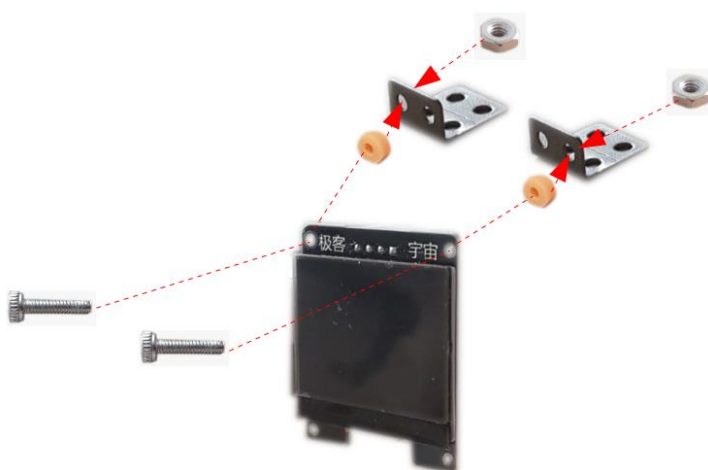
➔ **Pasul 13 - Atașare buton și potențiometru :**



➔ **Pasul 14 - Ansamblu senzor ultrasonic :**



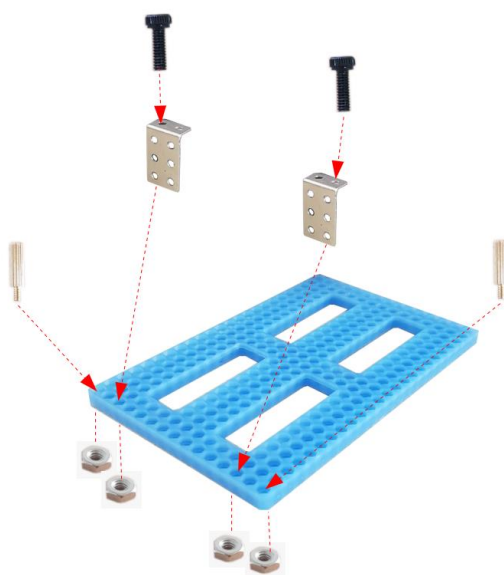
➔ **Pasul 15 - Ansamblu Ecran LCD :**



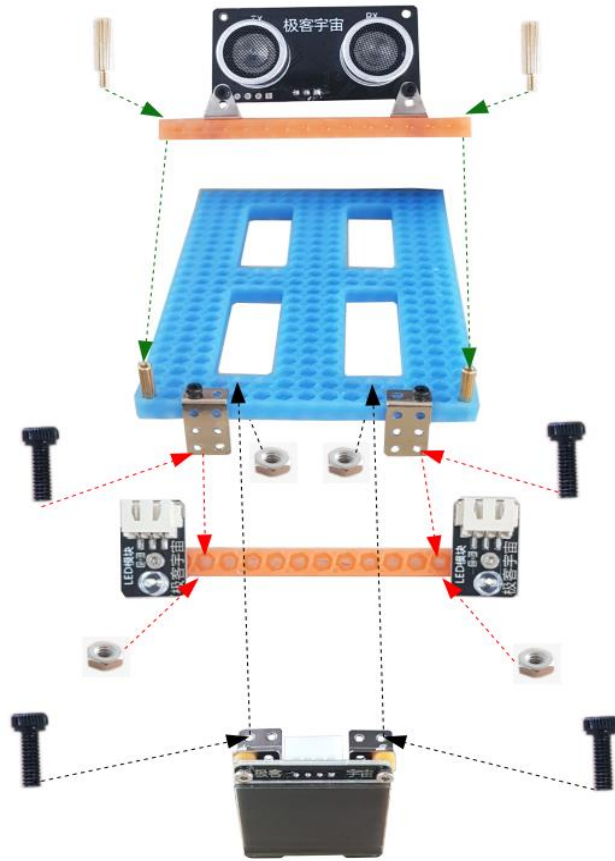
➔ **Pasul 16 - Ansamblu Faruri :**



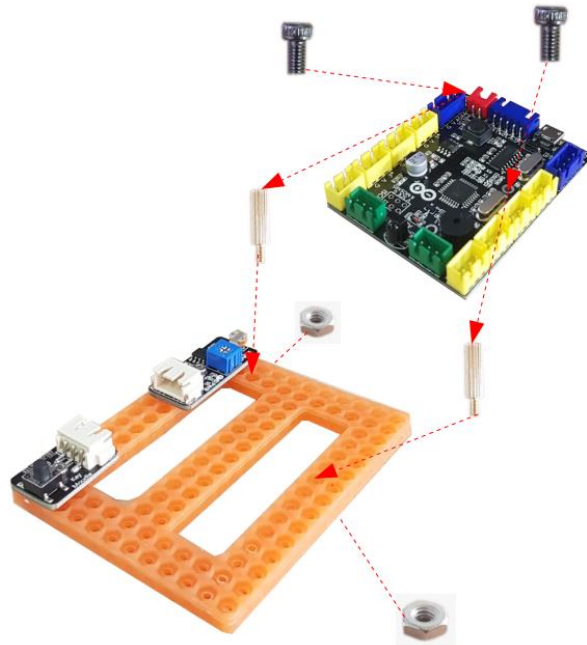
➔ **Pasul 17 - Suportul albastru :**



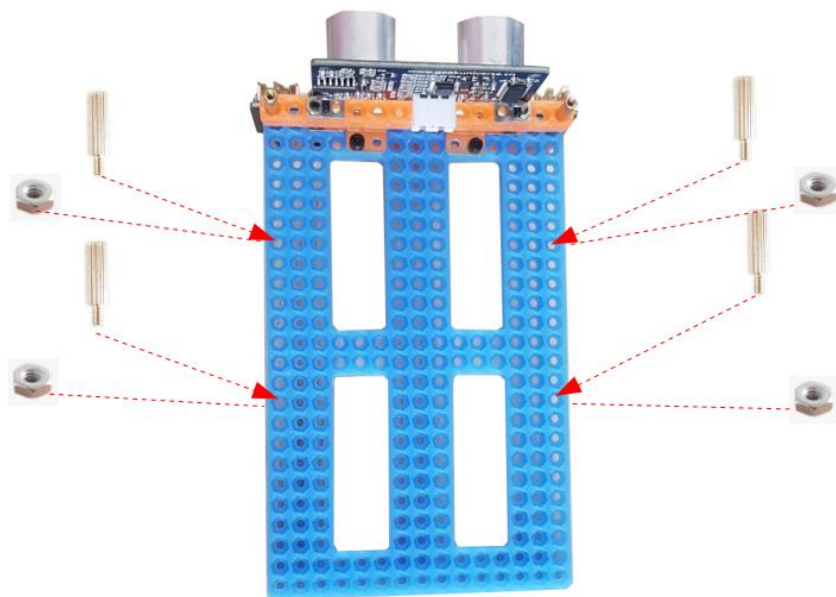
➔ **Pasul 18 - Atașare componente suport albastru :**



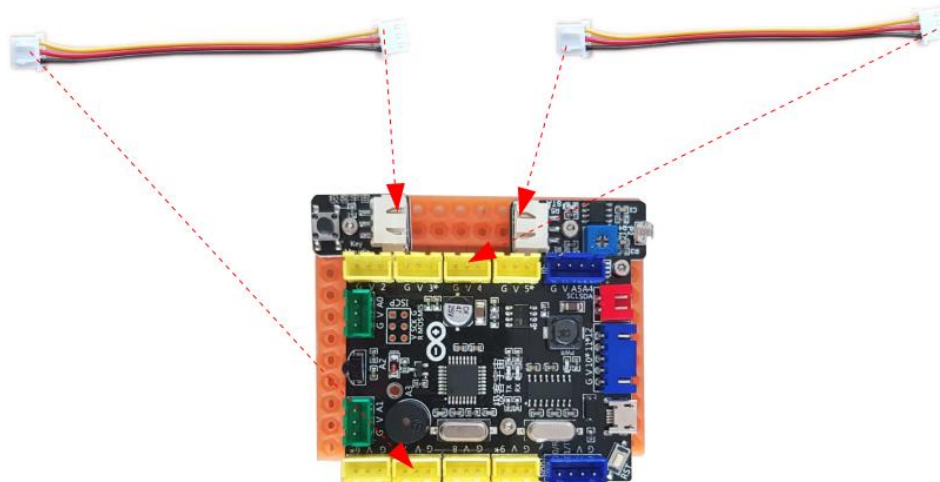
➔ Pasul 19 - Atașare Arduino support :



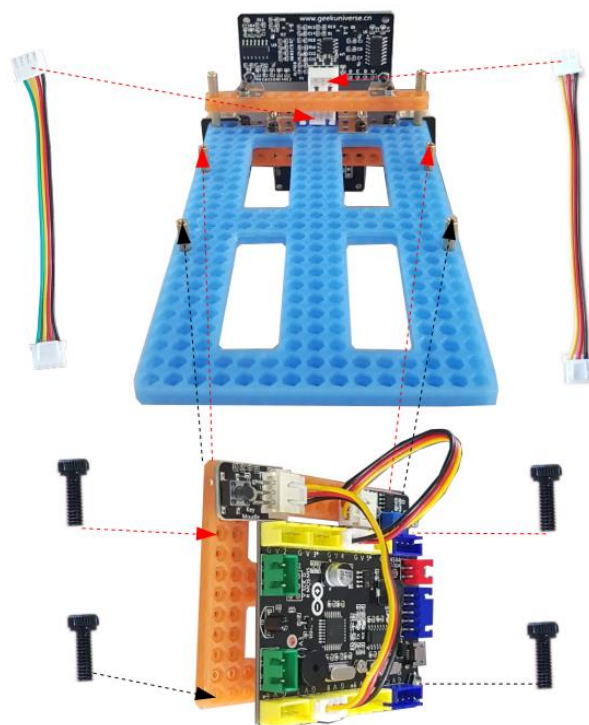
➔ **Pasul 20 - Piloni suport albastru :**



➔ **Pasul 21 - Cablare potențiomtru și buton :**

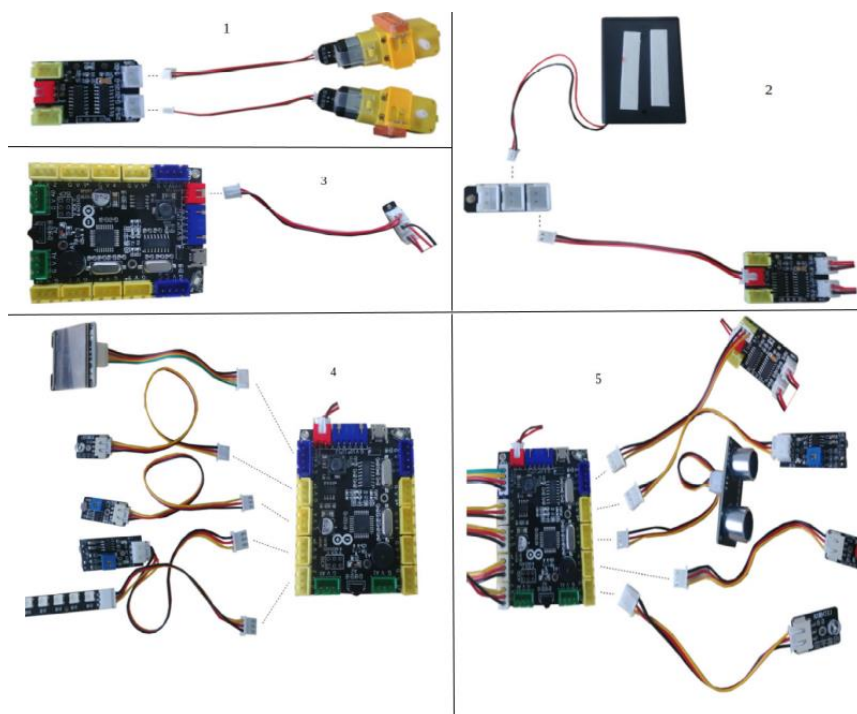


➔ **Pasul 22 - Montare ansamblu Arduino :**



➔ **Pasul 23 - Montare finală suporturi;**

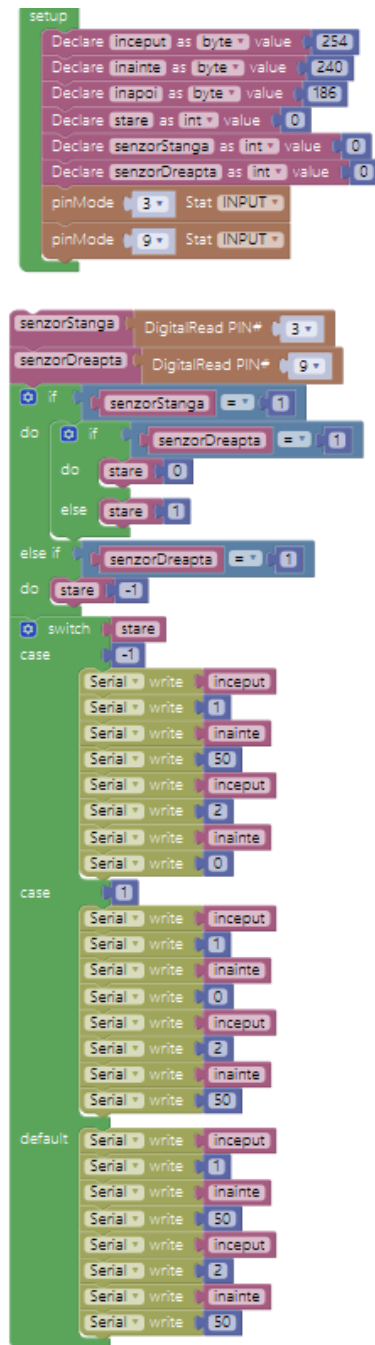
➔ **Pasul 24 - Finalizare cablare :**





## Program – Obstacole

- ➔ Robotul pornește încet și crește viteza treptat cât timp nu detectează niciun obstacol. La apropierea de un obstacol începe să reducă viteza și îl ocolește. Pentru a evita un obstacol robotul merge puțin în spate și virează stânga sau dreapta. Pentru a marca mersul înapoi va porni un led.



```

setup
  Declare incepat as byte value 254
  Declare inainte as byte value 240
  Declare inapoi as byte value 186
  Declare stare as int value 0
  Declare senzorStanga as int value 0
  Declare senzorDreapta as int value 0
  pinMode 3 Stat INPUT
  pinMode 9 Stat INPUT

senzorStanga DigitalRead PIN# 3
senzorDreapta DigitalRead PIN# 9

if senzorStanga = 1
do
  if senzorDreapta = 1
  do
    stare = 0
  else
    stare = 1
  else if senzorDreapta = 1
  do
    stare = -1
  switch stare
  case -1
  Serial write incepat
  Serial write 1
  Serial write inainte
  Serial write 50
  Serial write incepat
  Serial write 2
  Serial write inainte
  Serial write 0
  case 1
  Serial write incepat
  Serial write 1
  Serial write inainte
  Serial write 0
  Serial write incepat
  Serial write 2
  Serial write inainte
  Serial write 50
  default
  Serial write incepat
  Serial write 1
  Serial write inainte
  Serial write 50
  Serial write incepat
  Serial write 2
  Serial write inainte
  Serial write 50

```

## Program- Line Follower

### Principiul de funcționare:

- Se citește informația de la senzori: Se vor primi valorile analogice, câte una de la fiecare senzor, reprezentând intensitatea luminii reflectate, respective recepționată de senzor.
- Se verifică care senzor se află deasupra benzii: se compară fiecare valoare cu o limită prestabilită. În cazul în care valoarea este mai mică decât acea limită, programul înțelege că senzorul respectiv se află deasupra benzii de culoare neagră. În acest moment sunt aprinse și LED-urile care se găsesc deasupra liniei.
- Se ia o decizie: În funcție de poziția robotului deasupra liniei (mai în stânga sau în dreapta) microcontrollerul acționează motorul în mod corespunzător. Când robotul se află mai în stânga liniei, motorul drept încetinește, iar în cazul în care robotul nu mai „vede” deloc linia, acestuia (motorului) îi este schimbat sensul pentru o redresare cât mai rapidă. Pentru cazul în care robotul va fi mai în dreapta liniei, este exact pe dos.
- Se repetă: Ciclul este repetat.

```
#include < avr/pgmspace .h>
#include < LCD .h>
LCD = Lcd ;

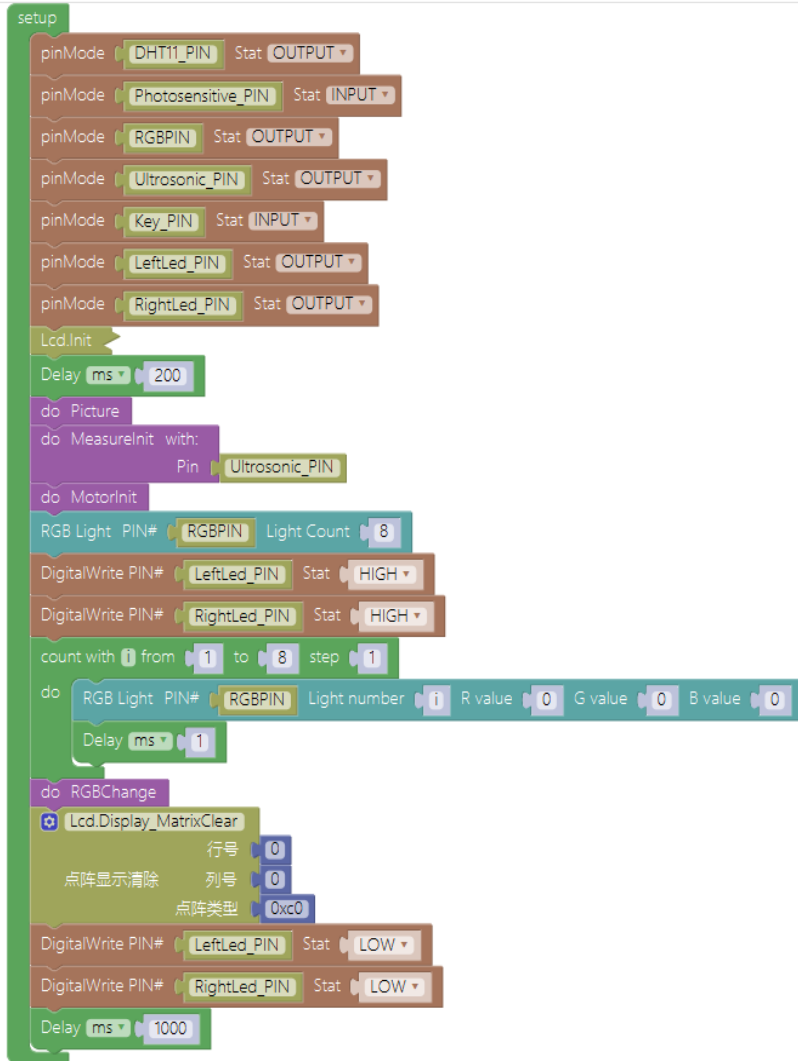
#define RGBPIN 2
#define LeftLed_PIN 6
#define RightLed_PIN 5
#define DHT11_PIN 3
#define Key_PIN 7
#define Ultrasonic_PIN 8
#define Photosensitive_PIN 4

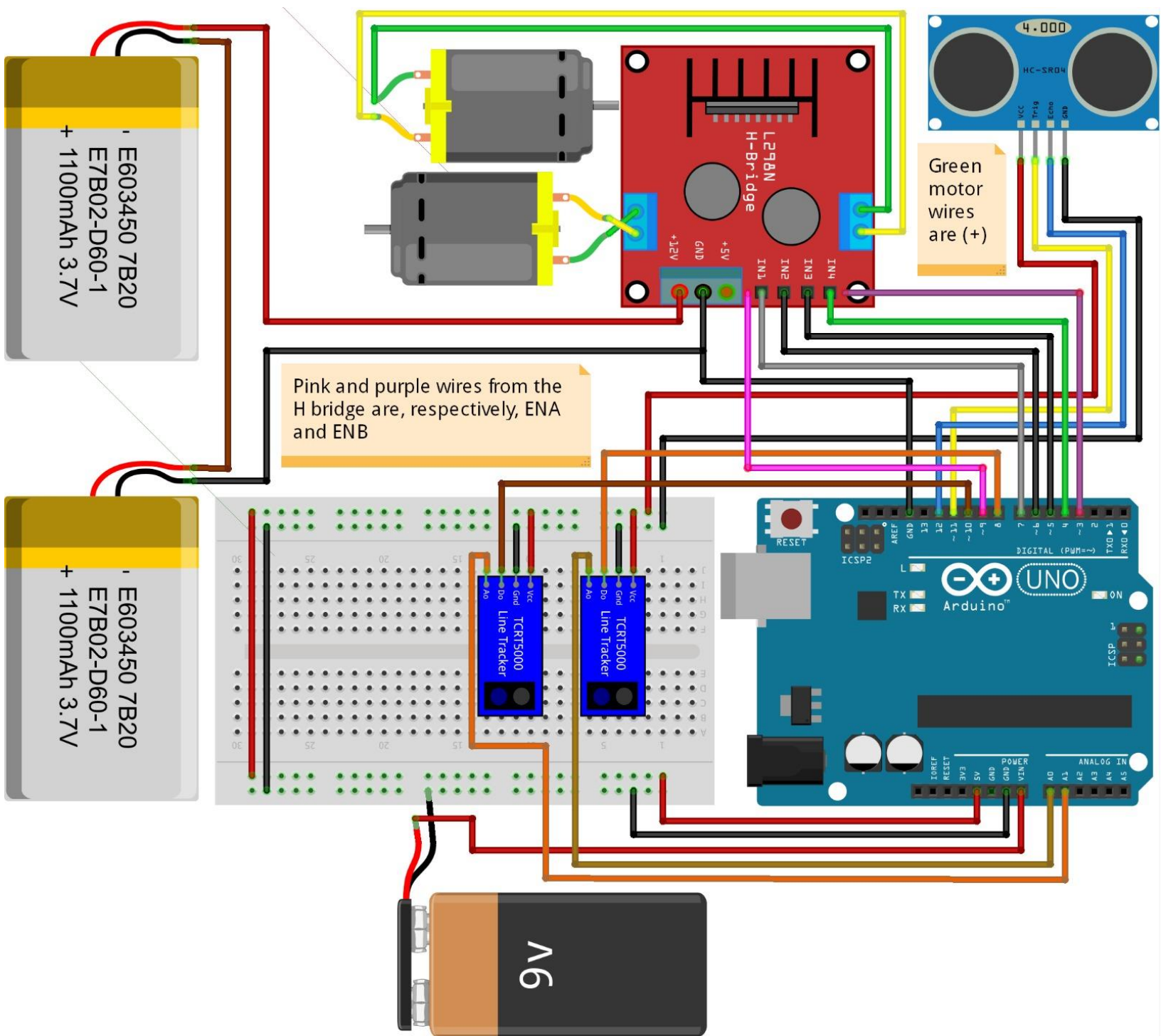
/*存儲圖形*/ const PROGMEM uint8_t arduino[]={0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x... ;

Declare PicDat as byte value 0
Declare Distance as long value 0
Declare Key8Val as byte value 0
Declare RandColodr as byte value 0

float tonelist [ 7 ] make list from text “ 1046.5,1174.7,1318.5,1396.9,1568,1760,1975.5 ”
int musiclist [ 32 ] make list from text “ 1,2,3,1,1,2,3,1,3,4,5,3,4,5,5,6,5,4,3,1,5,6,5,4,... ”
int highlist [ 32 ] make list from text “ 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,... ”
int rhythmlist [ 32 ] make list from text “ 4,4,4,4,4,4,4,4,4,2,4,4,2,8,8,8,8,4,4,8,8,8,... ”
int mylist [ ] make list from text “ 786,882,935,1049,1178,1322,1484 ”

repeat while true
do
do playmusic
do RandRGB
do Runing
do Monitor
do Greet
do Sing
```





fritzing

## CONCLUZII

Considerăm că tema abordată pentru atestat ne-a ajutat să ne dezvoltăm foarte mult. Am învățat atât să lucrăm în echipă, să ne spunem punctul de vedere și să aducem argumente pro și contra pentru fiecare idee, cât și să fim niște programatoare mai bune, să avem o gândire analitică, să verificăm fiecare detaliu și să căutăm cea mai eficientă soluție pentru a perfecționa robotul.

Cu această ocazie, am înțeles cum funcționează senzorul ultrasonic, ecranul LCD și placa Arduino. De asemenea, ne-am dat seama cât de mult influențează setarea unei valori analogice din codul programului în funcționarea senzorilor.

Dacă am mai avea ocazia să repetăm acest proiect, cu siguranță am fi luat un șasiu puțin mai mare, deoarece în momentul de față centrul de greutate nu este perfect centrat, mașina trăgând puțin stânga, însă acest lucru este corectat de senzorii IR care oferă turații corespunzătoare motoarelor pentru a urmări linia în mod corect. De asemenea, dacă am fi dispus de mai mult spațiu, am fi alimentat motoarele la o tensiune mai mare, poate chiar o alimentare separată pentru acestea. Calibrarea robotului a fost dificilă și nu a fost prea ușor să găsim o valoare perfectă. Am fi luat niște roți mai mari întrucât unele curbe nu sunt făcute eficient, conform așteptărilor.

Per total, a fost o experiență interesantă pe care am repeta-o cu siguranță.

## Bibliografie

- ➔ <http://www.scoalaargeseana.ro>
- ➔ <https://ro.scribd.com>
- ➔ <https://ro.wikipedia.org>
- ➔ <https://shop.nextlab.tech>
- ➔ <https://ro.scribd.com/document/340211628/Line-Follower-Robot-Presentation>
- ➔ [https://www.researchgate.net/publication/337945789\\_A\\_Line\\_Follower\\_Robot\\_with\\_Obstacle\\_Detection\\_by\\_Ultrasonic](https://www.researchgate.net/publication/337945789_A_Line_Follower_Robot_with_Obstacle_Detection_by_Ultrasonic)
- ➔ <https://ocw.cs.pub.ro/courses/pm/prj2013/dtudose/11>
- ➔ [https://ocw.cs.pub.ro/courses/pm/prj2012/avoinescu/line\\_follower2](https://ocw.cs.pub.ro/courses/pm/prj2012/avoinescu/line_follower2)
- ➔ <https://www.intorobotics.com/obstacle-avoidance-robot/>
- ➔ <https://www.optimusdigital.ro/en/kits/1716-line-follower-robot-kit.html>
- ➔ [https://en.wikipedia.org/wiki/Robot\\_calibration](https://en.wikipedia.org/wiki/Robot_calibration)