

**FIRST
LEGO
LEAGUE**
DISCOVER

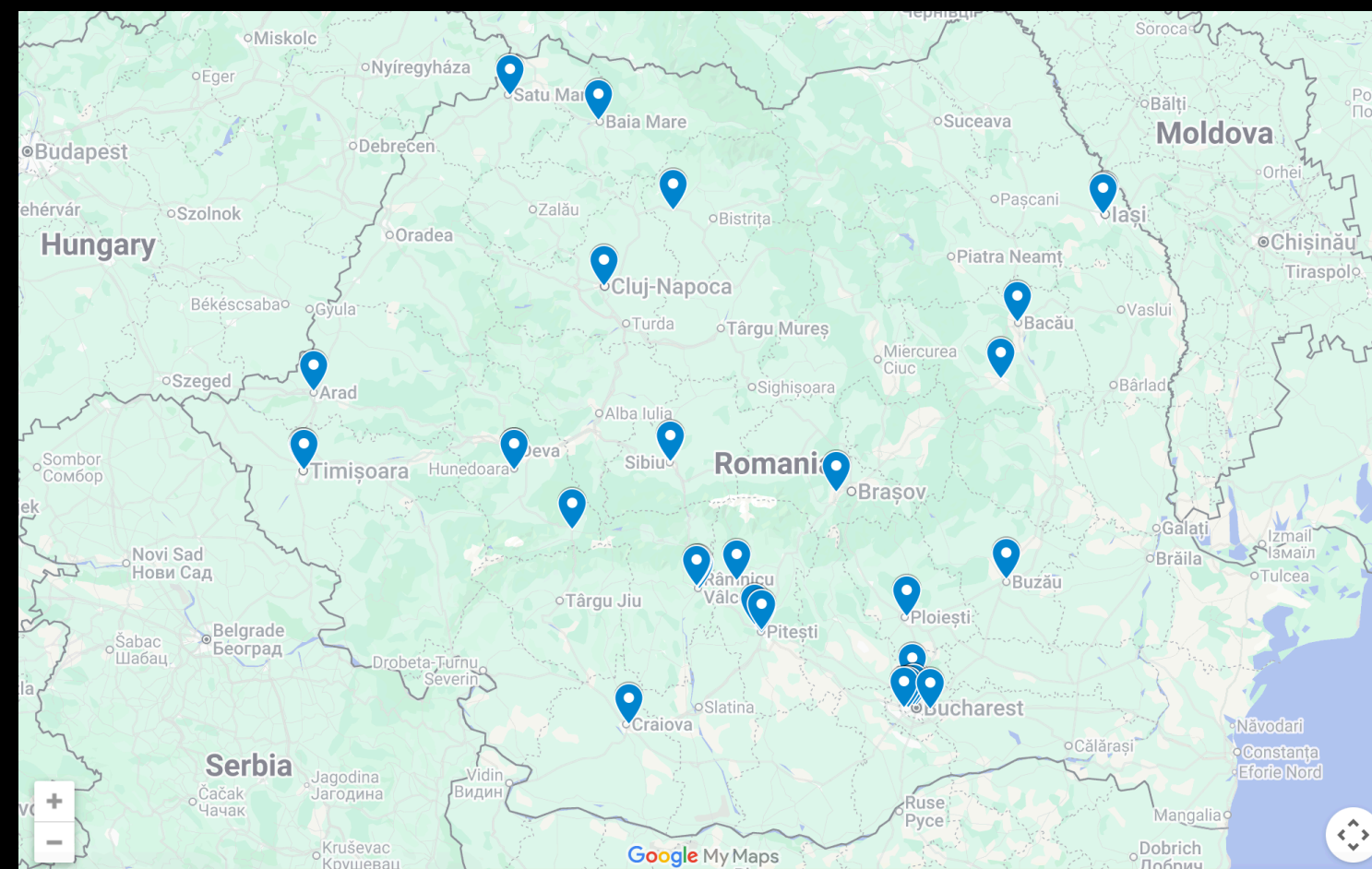
AGES
4-6

**FIRST
LEGO
LEAGUE**
EXPLORE

AGES
6-10

**FIRST
LEGO
LEAGUE**
CHALLENGE

AGES
9-16*
*ages vary by
country



Despre First LEGO League

FLL — Joc, Artă, Stiință

- Lectii complete **complet gratuite**– senzori, programare, mecanică -
 - <https://education.lego.com/en-us/lessons/>
 - Scratch, Python



FLL - Mai mult decat roboti

- In fiecare sezon FLL (FIRST LEGO League) copiii de 9-16 ani
 - propun și finalizează un proiect in care își ajută comunitatea
 - dezvoltă competențe STEM aplicate, o gândire critică și spirit de cercetare
 - întâlnesc profesioniști locali din **lumea reală a științei și tehnologiei**
 - învață să lucreze in echipe multidisciplinare
 - îmbină competiția și colaborarea : un termen inventat “coopertition”
cooperation+competition
 - programează roboți



Joc de robotică



Proiect Inovator



Valori fundamentale



Designul Robotului

FLL — De la joc la vocație

Fiecare misiune — o fereastră spre o lume profesională nouă

- FLL oferă un mediu unde copiii **aplică tehnologii moderne** – senzori, programare, mecanică – pentru a rezolva provocări inspirate din industrii reale:
 - *HYDRO DYNAMICS* (2017–2018) – inginerie hidraulică și protejarea resurselor de apă curată.
 - *CITY SHAPER* (2019–2020) – arhitectura, inginerie civilă și planificare urbană.
 - *CARGO CONNECT* (2021–2022) – logistica, descoperirea lanțurilor de aprovizionare globale.
 - *SUPERPOWERED* (2022–2023) – inginerie energetică, energie verde, infrastructuri inteligente.
 - *MASTERPIECE* (2023–2024) – tehnologii creative: design digital, animație, realitate augmentată.
 - *SUBMERGED* (2024–2025) – tehnologii marine, robotică subacvatică, știința mediului.

 *Rezultat:*

Cunoștințele STEM sunt aplicate în contexte reale și dezvoltă **interesul pentru cariere tehnice**

FLL — Tehnologia în acțiune.

- FLL nu înseamnă doar roboți – misiunea principală este de responsabilitate socială și învățare prin și în comunitate.
 - În fiecare an proiectele au teme cu impact social și specialiști au sesiuni de Q&A
 - Expert Interviews : <https://community.firstinspires.org/uncover-inspiration-for-your-unearthed-season-with-upcoming-webinar>
 - Career Connections: <https://www.youtube.com/playlist?list=PLpaPRqT711tjlOEveikQG1IKM2AdcA7-d>
 -  *Impact:* Copiii devin **cetățeni activi**, conștienți de provocările lumii și capabili să aducă schimbare pozitivă prin tehnologie și empatie.
-

Modelele de pe tabla de joc

FIRST LEGO LEAGUE - Empatie și Curiositate

- Fiecare misiune de pe tabla de joc îi îndeamnă copiii să exploreze lumea cu empatie și să vadă lumea și prin ochii altora
 - Organizatorii pornesc de la proiecte reale: unele mici și altele mari
 - Propun copiilor să creeze și ei un proiect în comunitatea lor
-

Project Spark 2

Site: Northeast United States

Conditions: Flat ground with a few rolling hills. Easy access for people, material and equipment

Client: Townspeople

Needs: Redesign and update an old playground



Intr-un oras s-a refacut un leagan pentru a ajuta o fetiță aflata in scaunul cu rotile sa se poata da in leagan chiar daca ea nu-si susține singura spatele.

Misiunea „**Leagănul**” (Swing) din provocarea FLL City Shaper presupune ca robotul să elibereze suportul unei fete aflate într-un scaun cu rotile, pentru ca aceasta să se poată legăna.

Pentru a îndeplini această misiune, robotul trebuie să:

- Să se deplaseze până la **leagăn**, evitând prima grindă de susținere,
- Și apoi să **elibereze corect suportul**, permițând legănarea.



Project Spark 3

Site: Valparaíso Region on the coast of Chile.

Conditions: Steep, beachfront property with difficult access for builders. Prone to earthquakes.

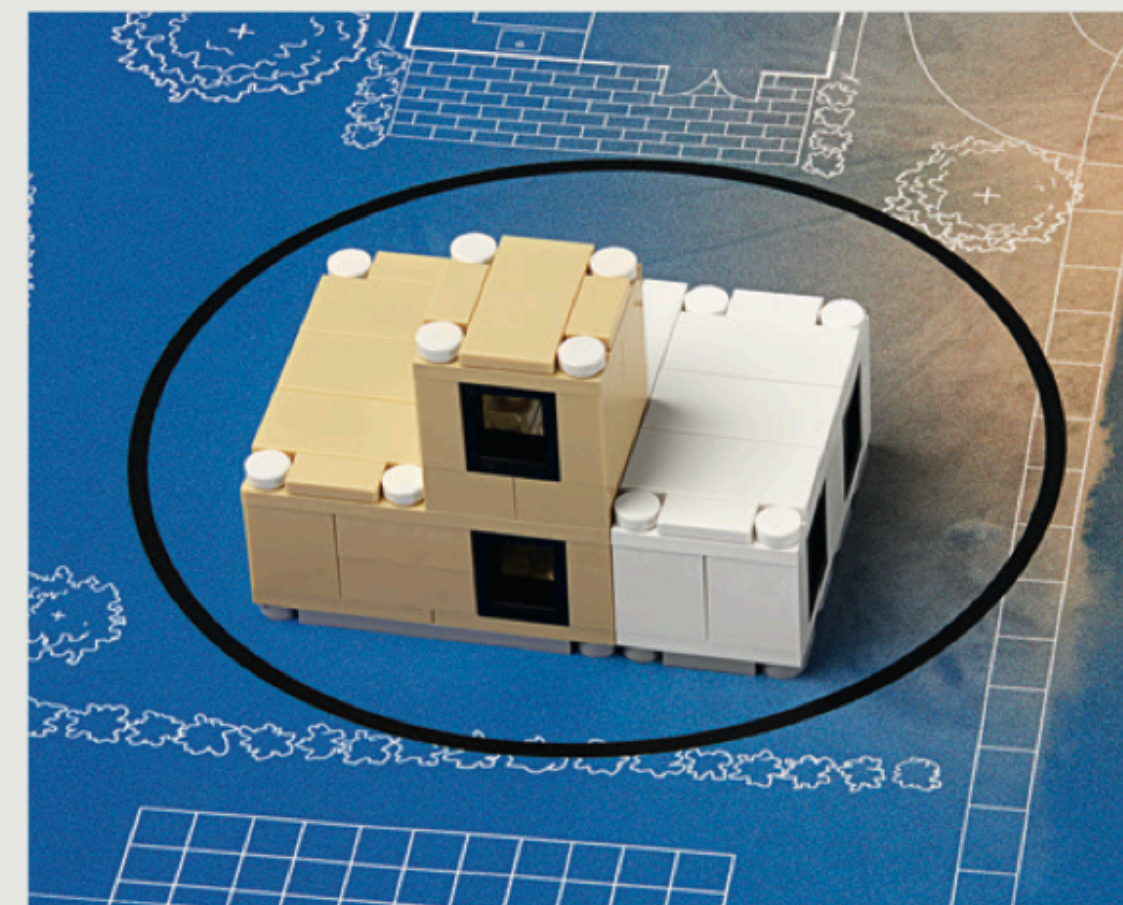
Client: Homebuyers in Coastal Chile

Intr-un oras s-au construit locuinte ieftine compacte modulare de Forma cubica care in caz de cutremur tind sa se separe si sa se rostogoleasca fara a se prabusi una peste alta.



The game

The “Building Units” Mission shows a great way to build homes swiftly. Modular construction is a way to create sections of a building in a factory that can be assembled quickly at the building site.



Proiectul premiat: FLL Challenge 2025 World Festival

- FLL SESI HEROES, Brazilia
 - au identificat o specie invazivă de coral : Tubastraea — “sun coral”
 - s-au consultat cu biologii marini => imediat după îndepărtarea mecanică a coloniilor din substratul natural (în prezent, cea mai aplicată metodă pentru a reduce răspândirea a lungul coastei braziliene) coralul se apara si eliberareaza larvele permitand
 - au propus și testat o metodă pentru scoaterea coralilor
- Solutia:
 - Scafandrul 1 poziționează un sac de colectare pentru plancton și acoperă cu el corali.
 - Scafandrul 2 răzuie coralul de pe substrat și îl direcționează direct în sac.
 - reduce riscul de raspândire



FLL Challenge

- In fiecare an un proiect nou e lansat in august.
 - *UNEARTHED* (2025–2026) – arheologie și analiza științifică a civilizațiilor trecute
 - <https://www.firstlegoleague.org/season#resources>

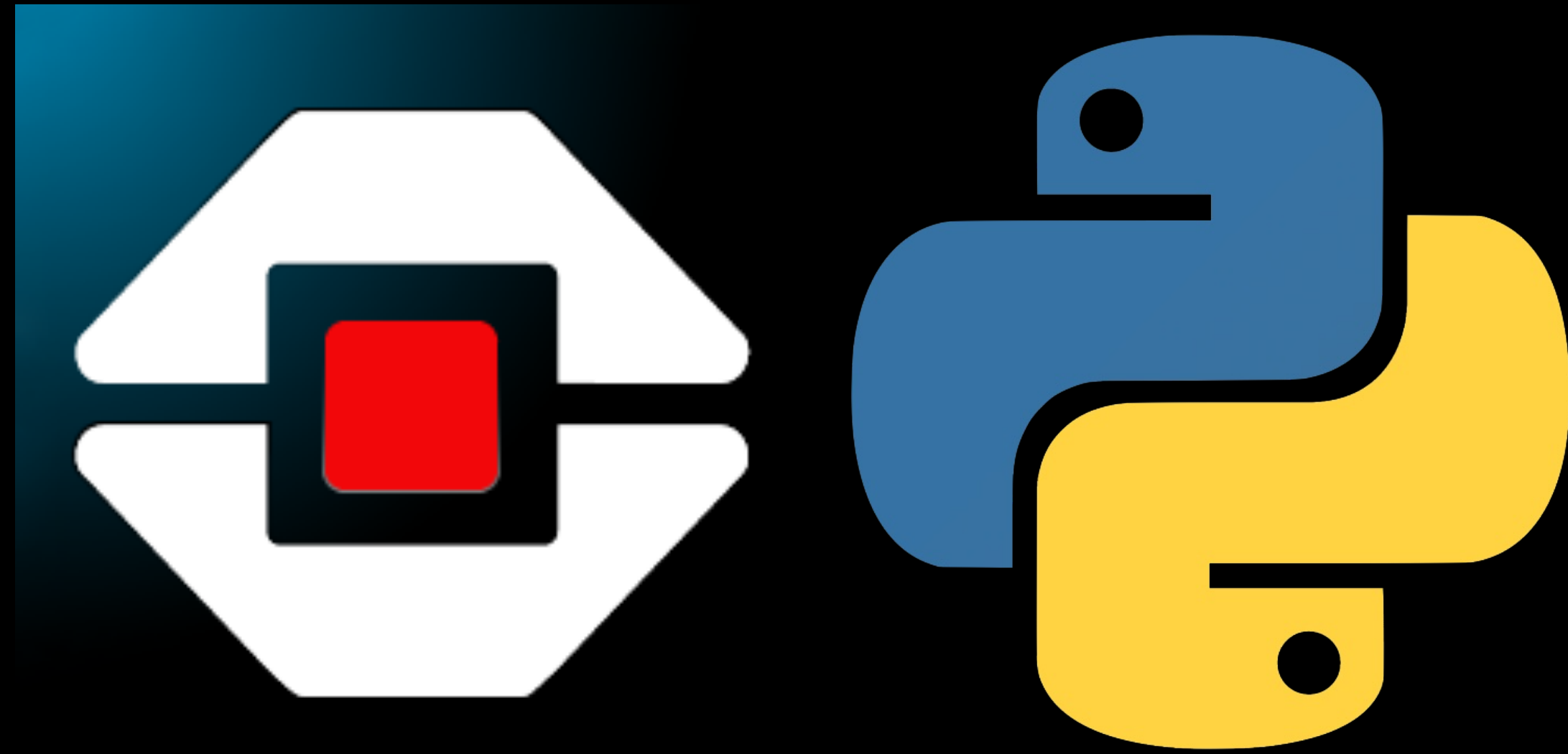


Comunitatea FLL este săritoare

Surse gratuite pentru hardware si programare (asigurați-vă că citați sursele):

- Robotii folosiți : un modul de control NXT/EV3/SPIKE, 4 motoare, 4 sensori
 - Robotul de bază : Drive Base
 - <https://firstinspires.blob.core.windows.net/fl/challenge/2019-20/crane-mission-ev3-soluton.pdf>
 - Roboti pentru învățat: EV3 <https://ev3lessons.com/en/RobotDesigns.html>
 - Roboti pentru competiții: SPIKE <https://primelessons.org/en/RobotDesigns.html>
 - Python EV3: Pybricks : <https://pybricks.com/ev3-micropython/index.html>
 - Python SPIKE: <https://spike.legoeducation.com/essential/help/lhs-help-python>
 - Lectii de stiinta cu roboti: <https://education.lego.com/en-us/lessons/ev3-robot-trainer/>
-

Lectii First LEGO League



Lecția 0 - Instalări

2024 - M. NICHITIU

LEGO MINDSTORMS Education - Surse

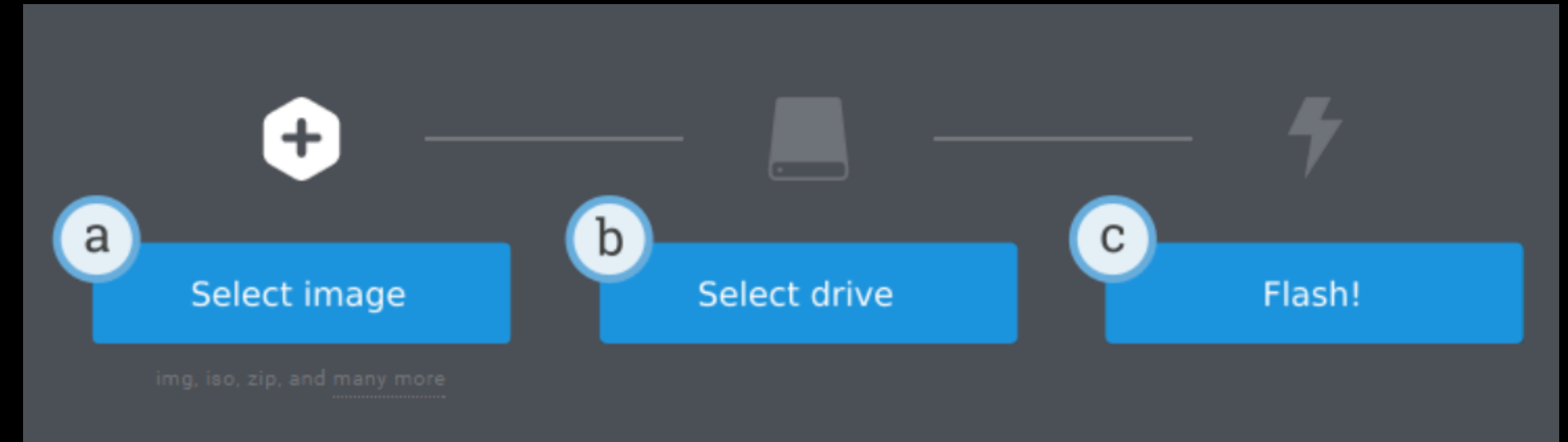
- Robotul pentru lecții este o variantă de “Driving Base” cu senzori

<https://education.lego.com/en-us/product-resources/mindstorms-ev3/downloads/building-instructions/#robot>

- Documentație microPython și instalări:

<https://pybricks.com/ev3-micropython/>

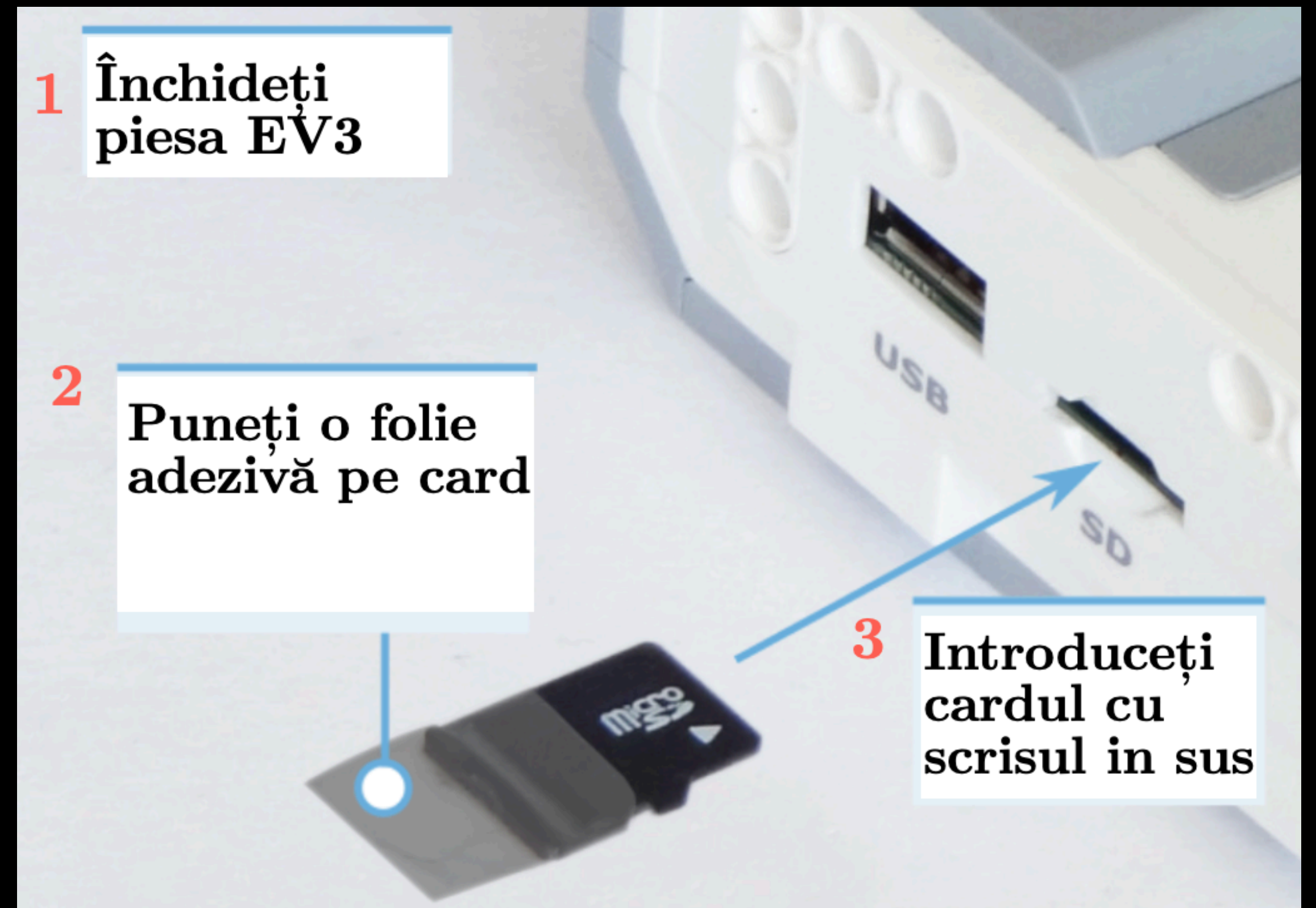
MicroPython



- Un card microSD (min 4GB, max. 32GB)
- Descărcați imaginea microSD EV3 MicroPython
<https://education.lego.com/en-us/product-resources/mindstorms-ev3/teacher-resources/python-for-ev3/>
- Descărcați, instalați, lansați Etcher - gravare simplă a cardurilor microSD
<https://etcher.balena.io/>
- Salvați imaginea microSD EV3 MicroPython pe card

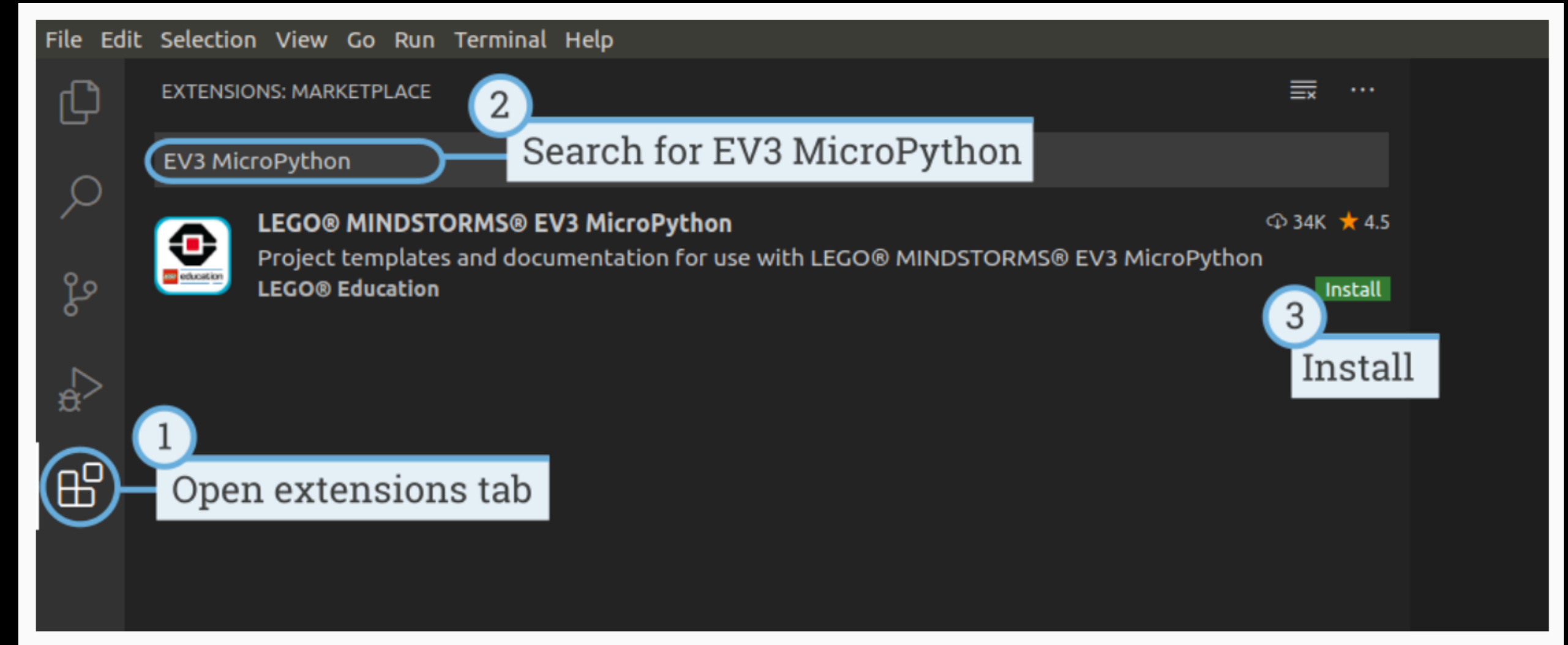
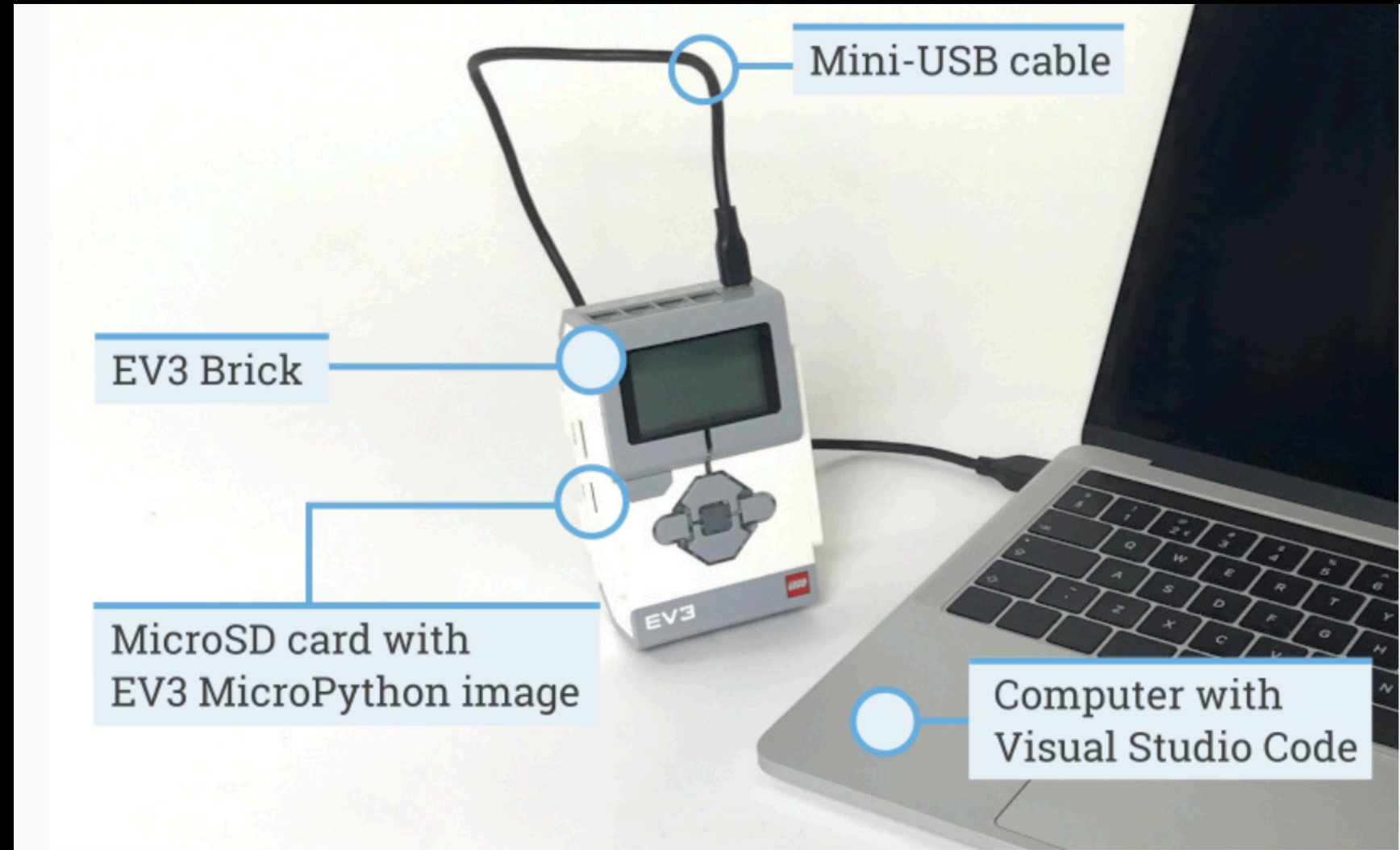
LEGO MINDSTORMS Education EV3 MicroPython

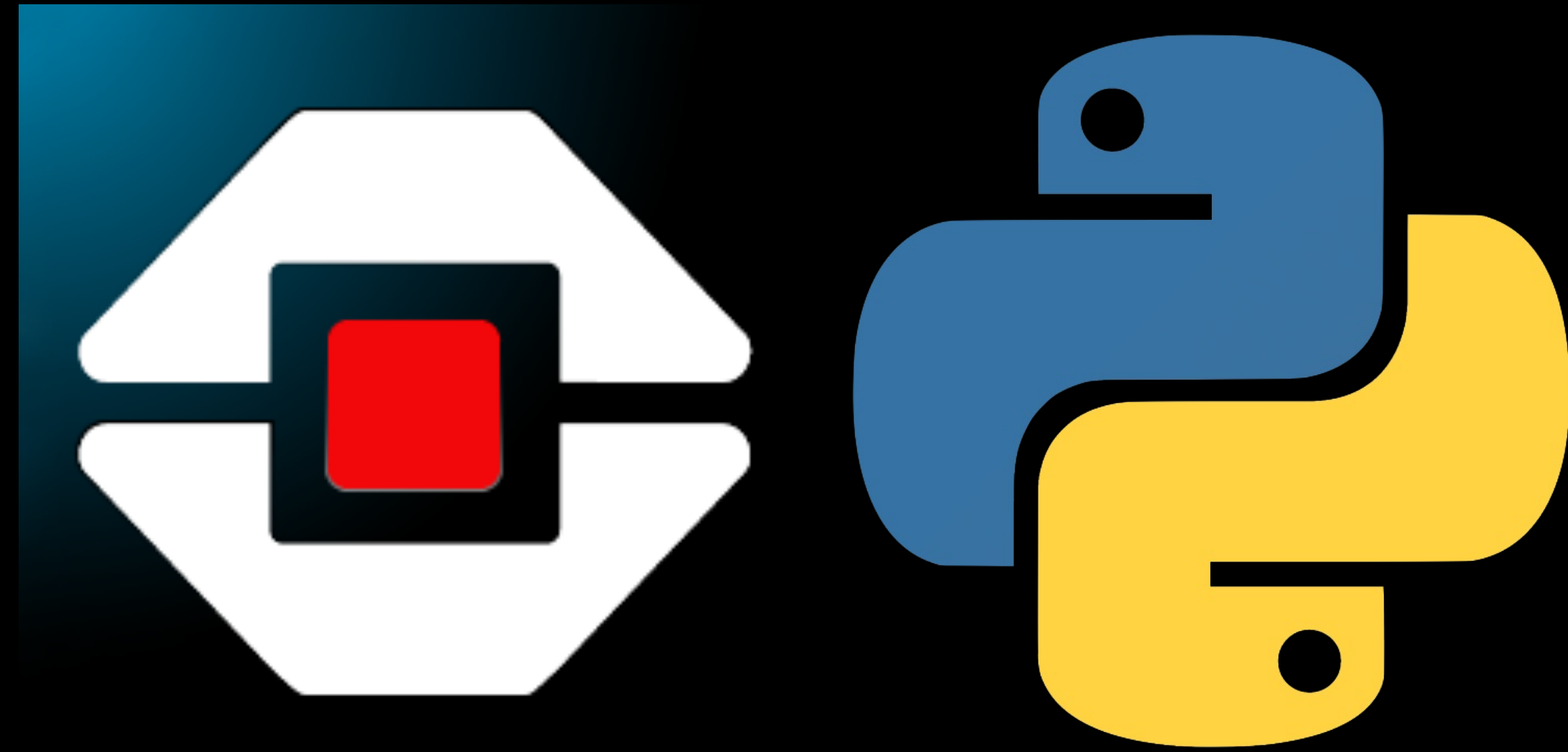
- Procesul de pornire poate dura câteva minute.
- Piesa EV3 este gata de utilizare atunci când lumina de stare devine verde.



LEGO MINDSTORMS Education EV3 MicroPython

- Descărcați, instalați, lansați Visual Studio Code.
- Deschideți extensiile si instalați EV3 MicroPython





Lecția 1 - Introducere în EV3

Curriculum

- Bazat pe: „Introduction to Programming EV3 Curriculum” dezvoltat de Carnegie-Mellon Robotics Academy <https://www.cmu.edu/roboticsacademy/roboticscurriculum/index.html>
 - FLL City Shaper Challenge
<https://www.firstinspires.org/resource-library/fl/challenge/challenge-and-resources>
-

FLL Challenge CityShaper

Pentru programare: table de joc din sezonul **CITY SHAPER** - modelarea unui viitor urban, abordând probleme legate de transport, accesibilitate și dezastre naturale.

- modelele sunt ușor de înțeles pentru copiii de orice vârstă
 - o problemă legată de o clădire sau un spațiu public din comunitatea lor
 - unde se petrec **inundațiile rutiere**, să înțeleaga de ce și să se gandeasca la modalități simple de a le reduce.
 - De duce **poluarea apei** in comunitatea lor și cum pot orașele să o diminueze.
 - <https://www.firstlegoleague.org/past-challenges> CityShaper
 - [Kickoff Video](#)
-



Objective

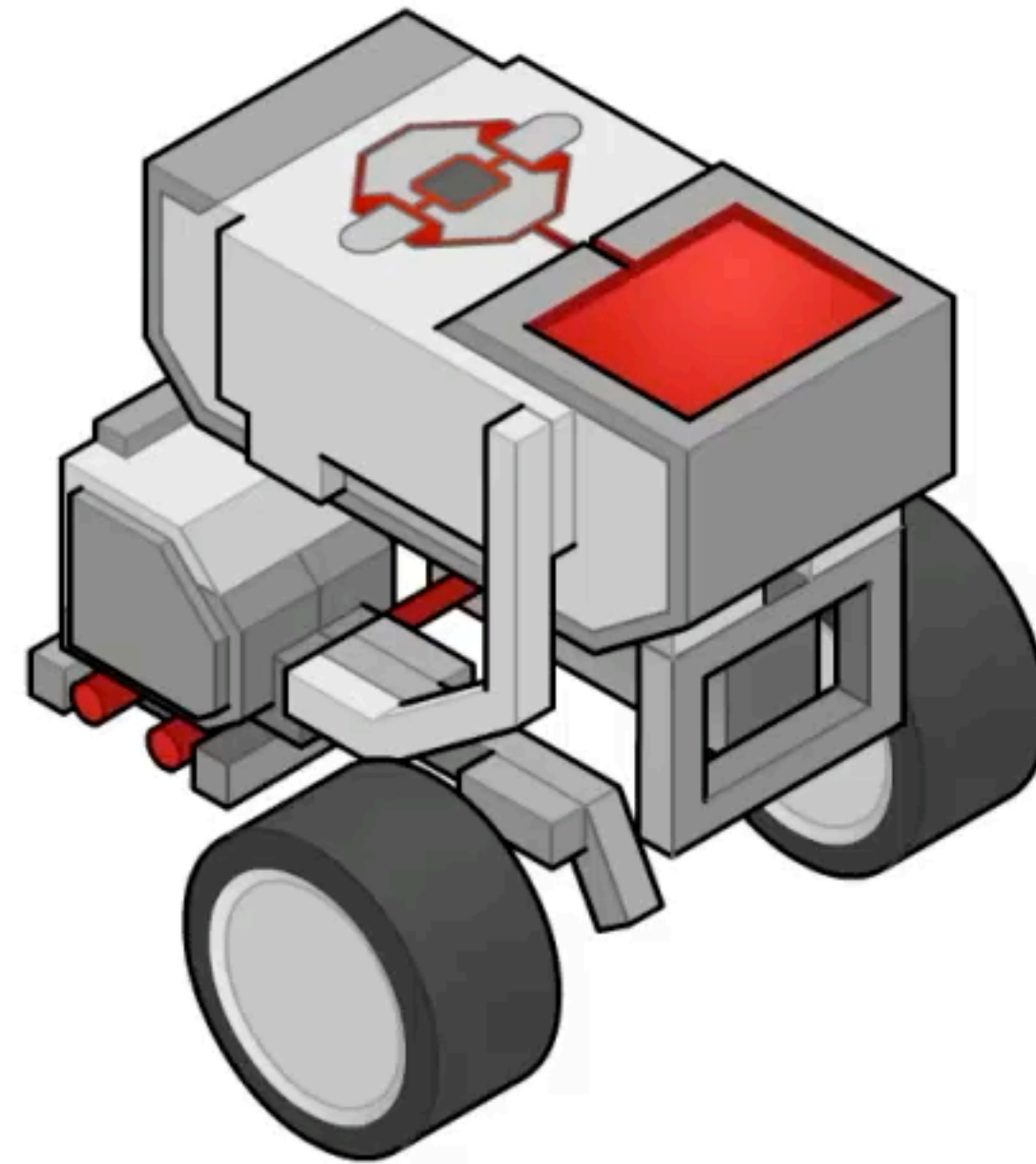
- ce este EV3 și cum funcționează.
- observarea valorilor motoarelor și senzorilor folosind sistemul EV3.
- Primele elemente de programare iterativă și EV3

Ideile Principale - Video



Big Ideas 1 & 2

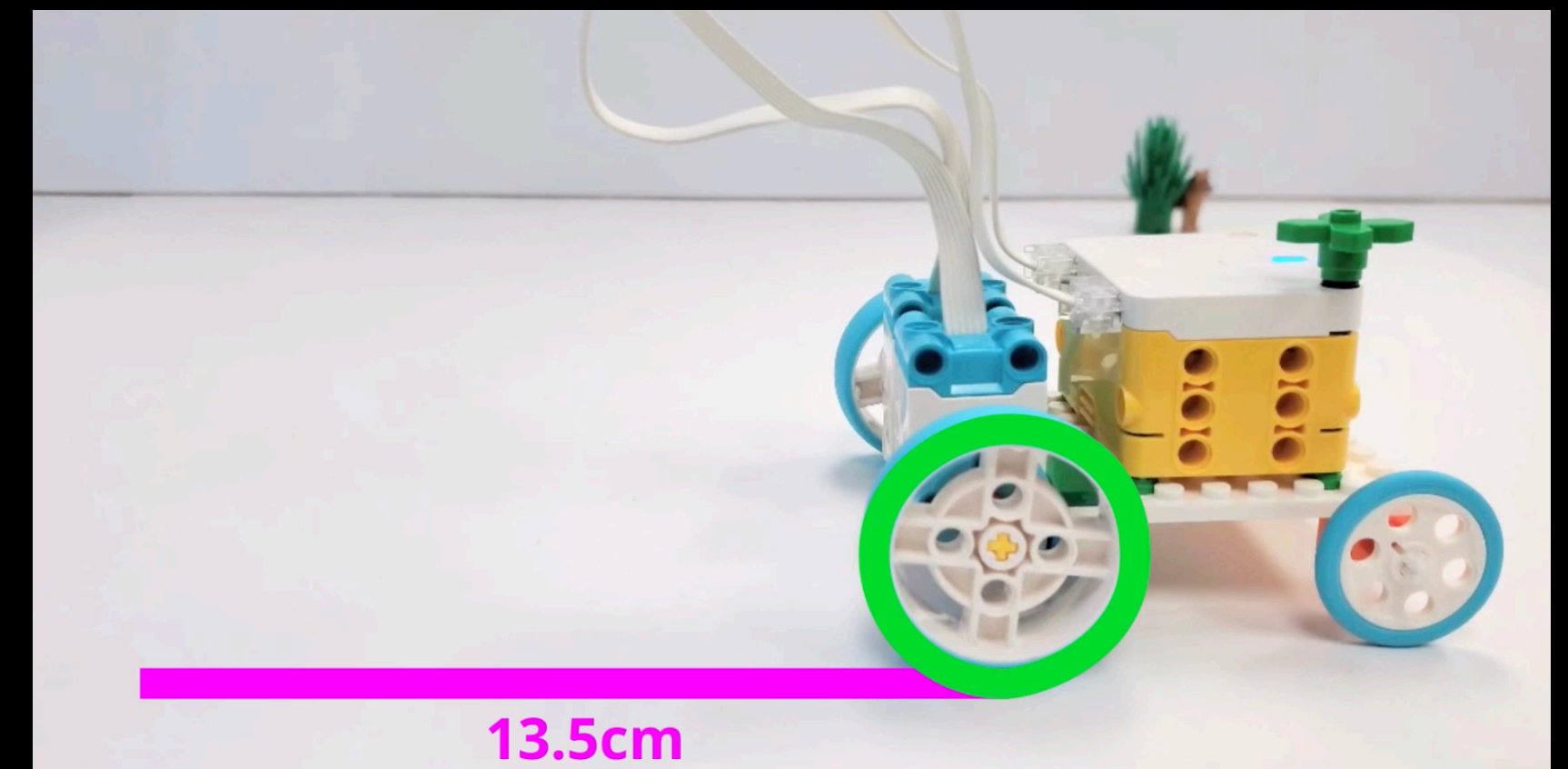
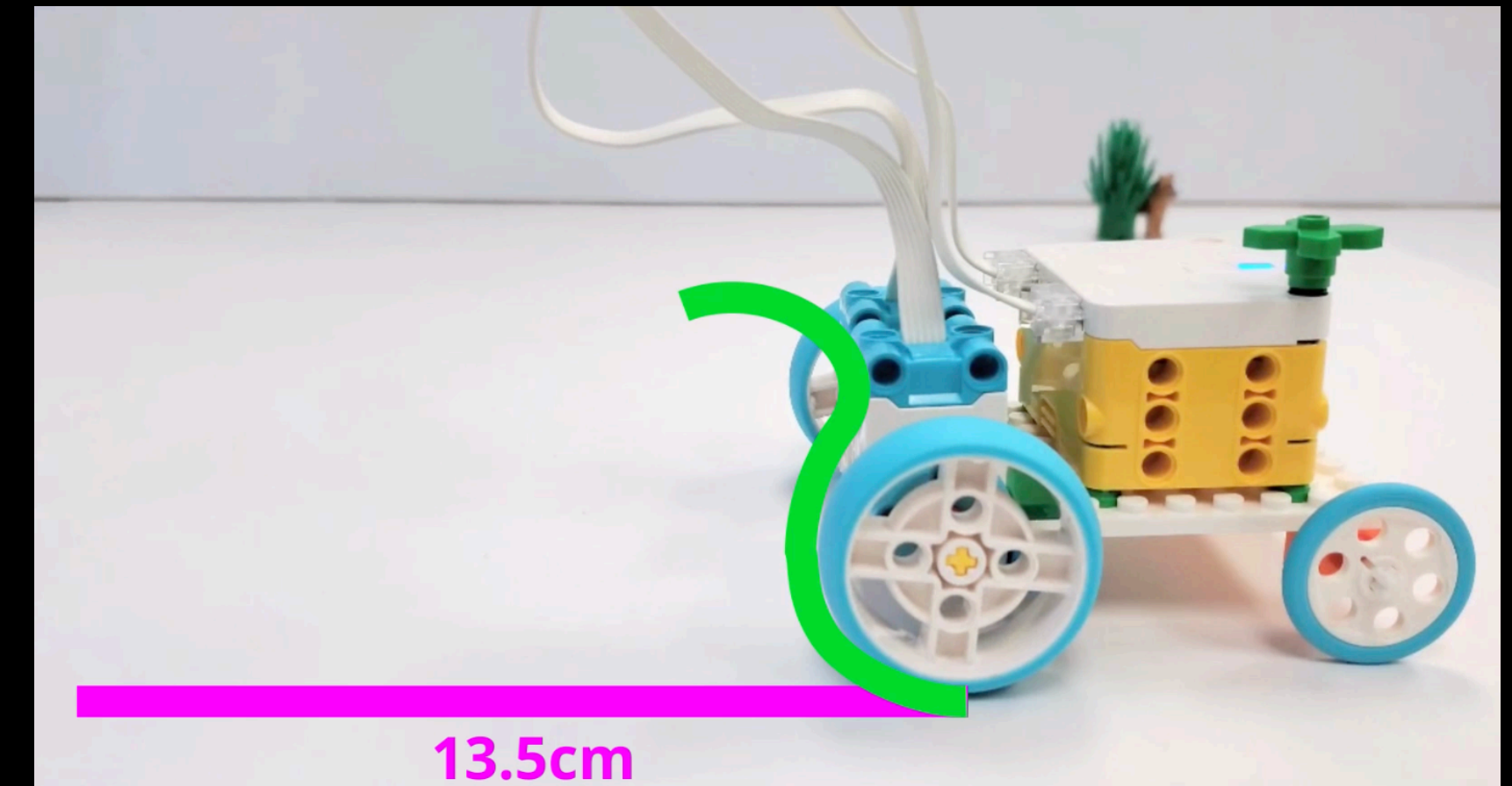
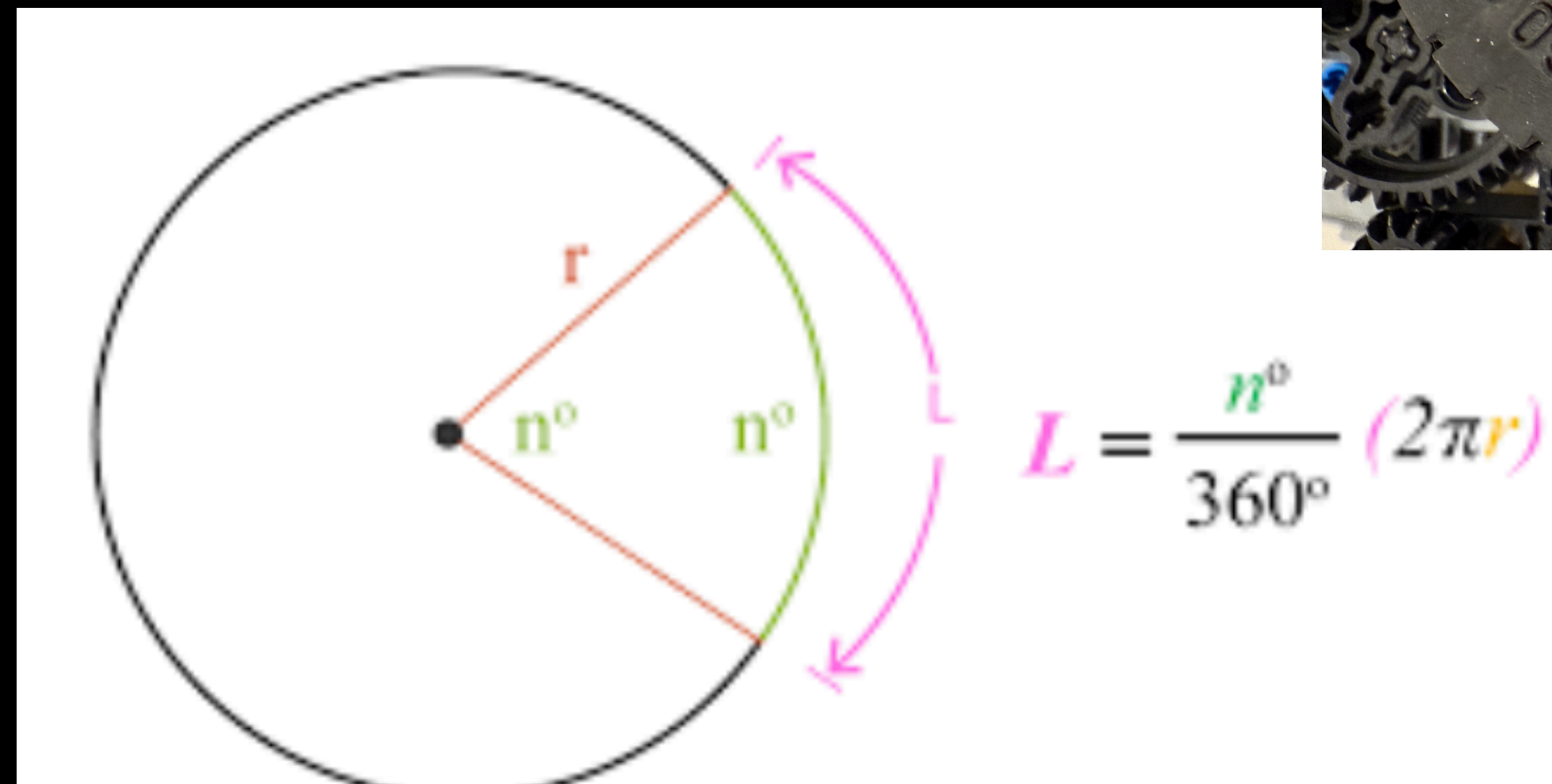
Porturi - Video Introductiv



Ports

Cum se măsoară distanța parcursă

- Circumferința roții reprezintă o rotație completă
- Diametrul roților LEGO, este primul număr găsit pe janta anvelopei: 62.4 mm

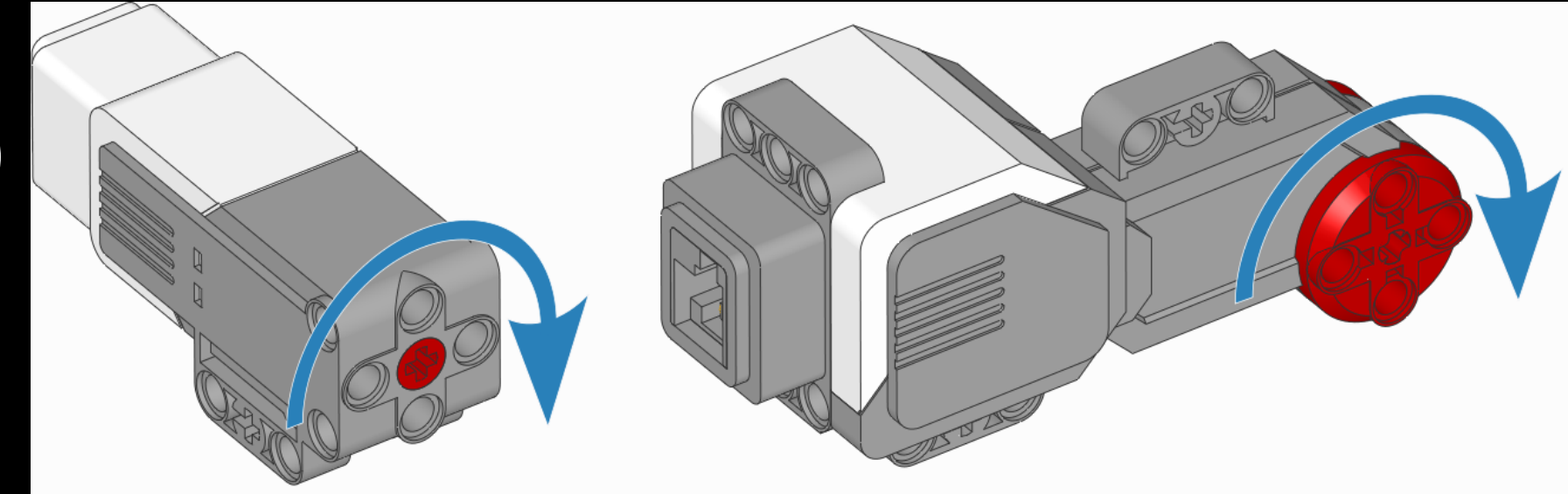


Controlul motoarelor cu senzori

```
class Motor(port, positive_direction=Direction.CLOCKWISE, gears=None)
```

Parametri:

- port(**Port**) – portul la care este conectat motorul
- positive_direction(**Direction**) – direcția de rotație pentru o valoare pozitivă de viteză sau unghi
- Metode relevante:
 - pentru măsurători: speed(), angle(), reset_angle(angle) - unghiul motorului se măsoară în grade (deg-degrees)
 - stalled() verifică dacă motorul este blocat, i.e. dacă nu se poate mișca la cuplu maxim (Intoarce : True/False)
 - run_target(speed, target_angle, stop_type=Stop.COAST, wait=True)- rotește motorul cu viteza de maxim speed grade pe secundă pentru un unghi total dat în grade
 - stop() Oprește motorul și îl lasă să se rotească liber. Motorul se oprește treptat din cauza frecării.
 - break() Frânează pasiv motorul.

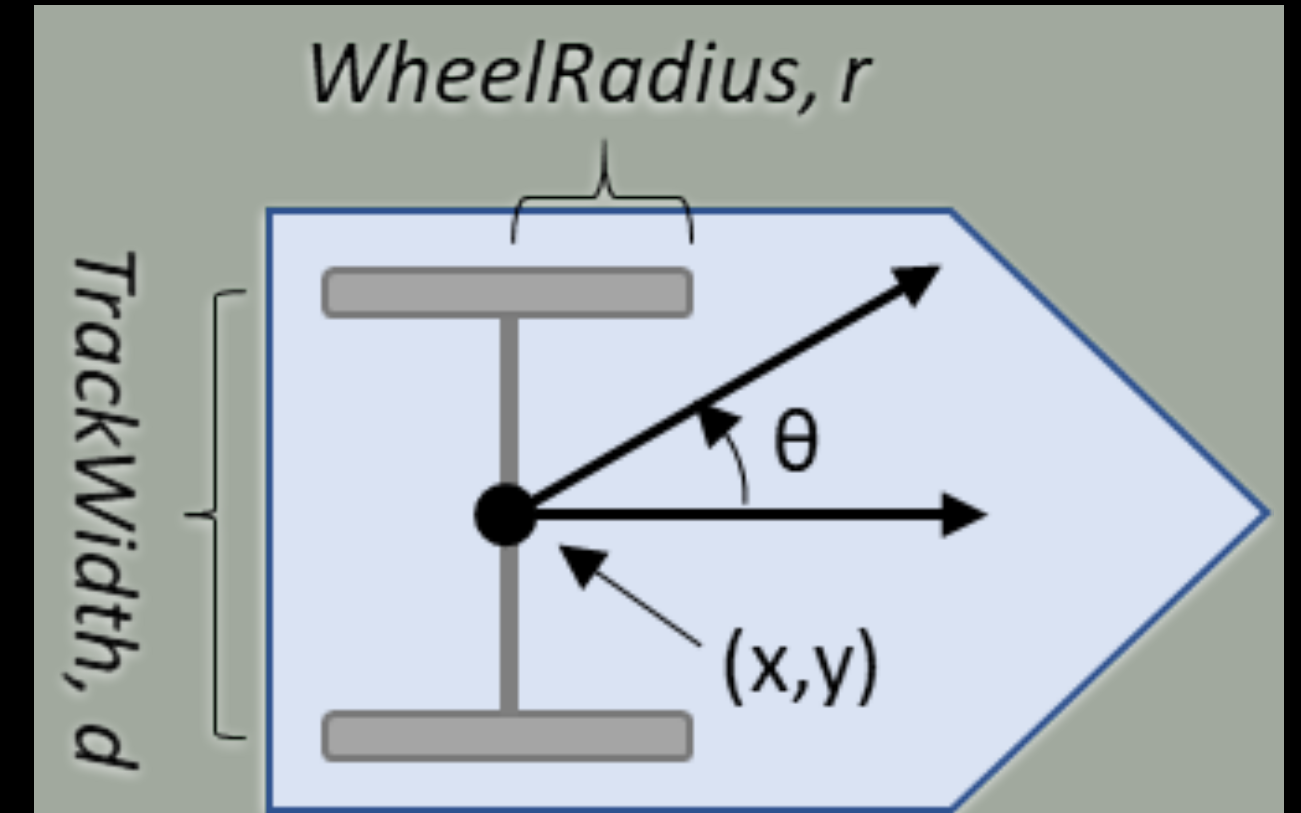


Motoare EV3 mari

```
# Inițializează un motor conectat la Portul B  
left_motor = Motor(Port.B)
```

```
# Inițializează un motor conectat la Portul C  
right_motor = Motor(Port.C)
```

Python poate face calculele
pentru deplasarea robotului



$\dot{\phi}_L$: Viteza roții stângi (m/s)

$\dot{\phi}_R$: Viteza roții drepte (m/s)

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2} \cos(\theta) & \frac{r}{2} \cos(\theta) \\ \frac{r}{2} \sin(\theta) & \frac{r}{2} \sin(\theta) \\ -r/d & r/d \end{bmatrix} \begin{bmatrix} \dot{\phi}_L \\ \dot{\phi}_R \end{bmatrix}$$

Clasa DriveBase(șasiu)

```
class DriveBase(left_motor, right_motor, wheel_diameter, axle_track)
```

Parametri:

- **left_motor** (*Motor*) – portul motorului roții stângi
- **right_motor** (*Motor*) – portul motorului roții drepte
- **wheel_diameter** (dimension: mm) – diametrul roților in mm,
- **axle_track** (dimension: mm) – distanța dintre roți

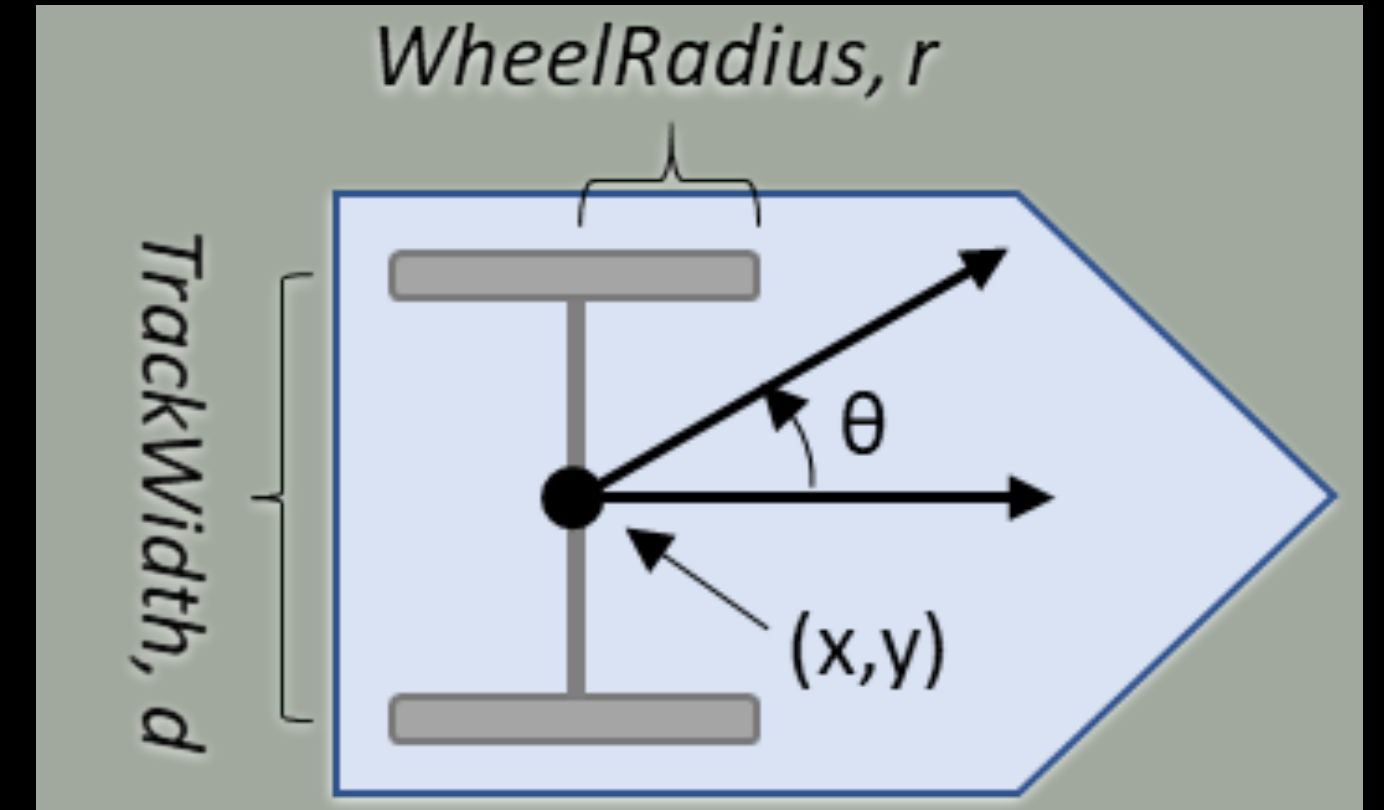
Clasa pentru șasiului robotului si controlează avansarea pe o distanță dată sau cotirea cu un unghi dat

Important:

- roțile pot aluneca în timpul deplasării, deci distanța parcursă și unghiul sunt doar estimări.
-

Clasa DriveBase(șasiu)

```
class DriveBase(left_motor, right_motor, wheel_diameter, axle_track)
```



Metodele clasei:

- `straight(distance)` – se deplasează în linie dreaptă pe o distanță specificată (mm), apoi se oprește.
- `turn(angle)` – se rotește pe loc cu un unghi specificat, apoi se oprește
- `drive(speed, steering)`
 - șasiul se deplasează la viteza liniară și viteza de rotație specificate. Ambele valori sunt măsurate din punctul central dintre roțile robotului.
 - `speed` (viteză: mm/s) – Viteza liniară a robotului.
 - `steering=turn_rate` (viteză rotațională: deg/s) – viteza de rotație a robotului.
- `stop()` – oprește robotul.

DriveBase - Dimensiunile robotului

Estimarea inițială:

- măsoară diametrul roții (`wheel_diameter`) cu o riglă
- estimează `axle_track` ca distanța dintre **mijloacele roților**

Roțile/Puntea se pot **tasa** sub greutatea robotului => Testează și măsoară distanța/unghiul real parcurs

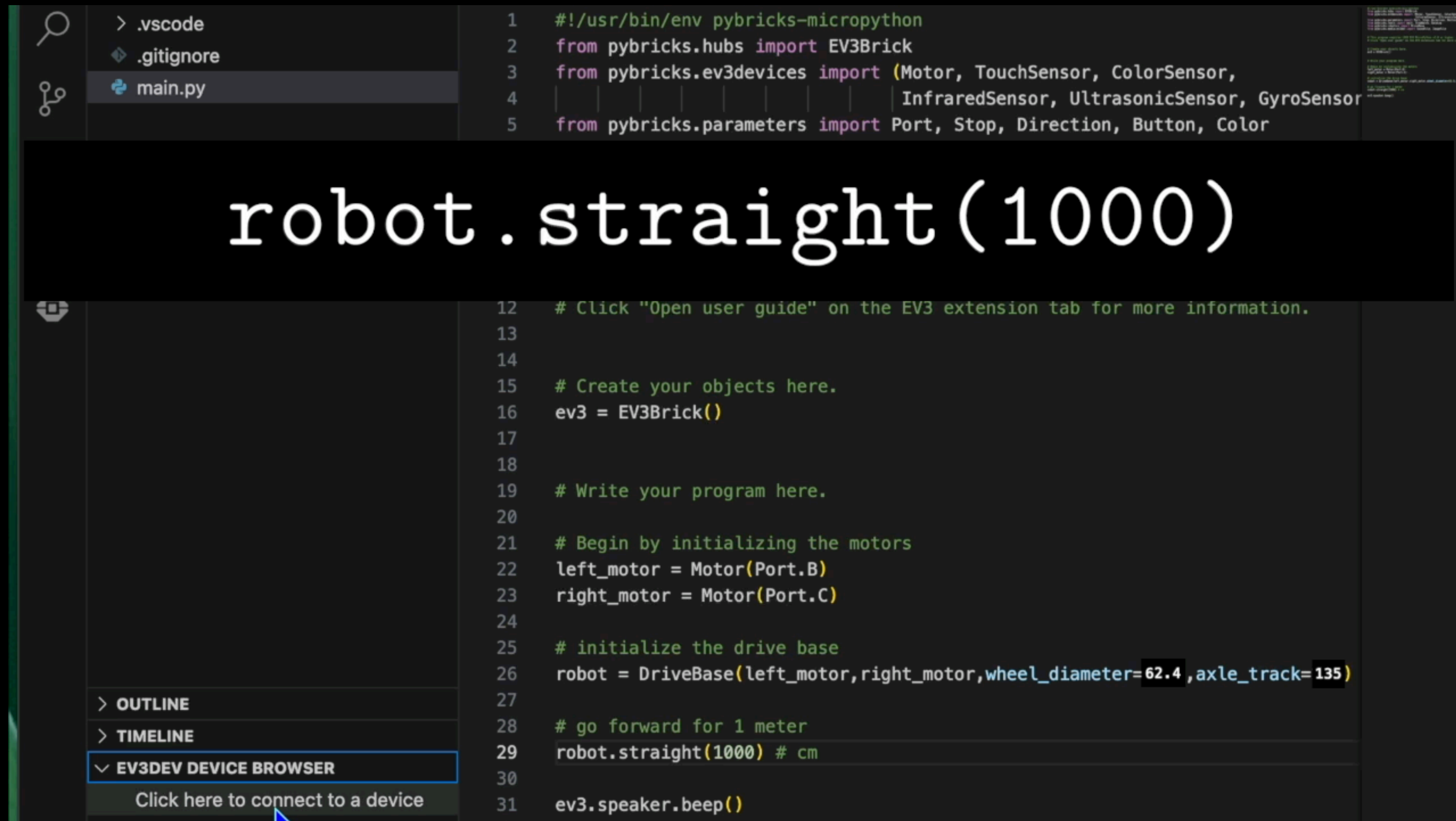
Ajustare `wheel_diameter`:

- Dacă robotul nu ajunge/rotește destul de departe, redu valoarea *wheel_diameter*/ crește valoarea *axle_track*.
- Dacă robotul merge/rotește prea departe, crește valoarea *wheel_diameter*/ redu valoarea *axle_track*.

```
my_robot.reset()  
my_robot.straight(1000)  
  
...  
my_robot.distance()
```

```
my_robot.reset()  
my_robot.angle(360)  
  
...  
my_robot.angle()
```

Programare 1 - Video: Mișcare în Linie Dreaptă



```
1  #!/usr/bin/env pybricks-micropython
2  from pybricks.hubs import EV3Brick
3  from pybricks.ev3devices import (Motor, TouchSensor, ColorSensor,
4  | | | | | | | | InfraredSensor, UltrasonicSensor, GyroSensor)
5  from pybricks.parameters import Port, Stop, Direction, Button, Color

robot.straight(1000)

12  # Click "Open user guide" on the EV3 extension tab for more information.
13
14
15  # Create your objects here.
16  ev3 = EV3Brick()
17
18
19  # Write your program here.
20
21  # Begin by initializing the motors
22  left_motor = Motor(Port.B)
23  right_motor = Motor(Port.C)
24
25  # initialize the drive base
26  robot = DriveBase(left_motor, right_motor, wheel_diameter=62.4, axle_track=135)
27
28  # go forward for 1 meter
29  robot.straight(1000) # cm
30
31  ev3.speaker.beep()
```

> .vscode
◆ .gitignore
🔗 main.py

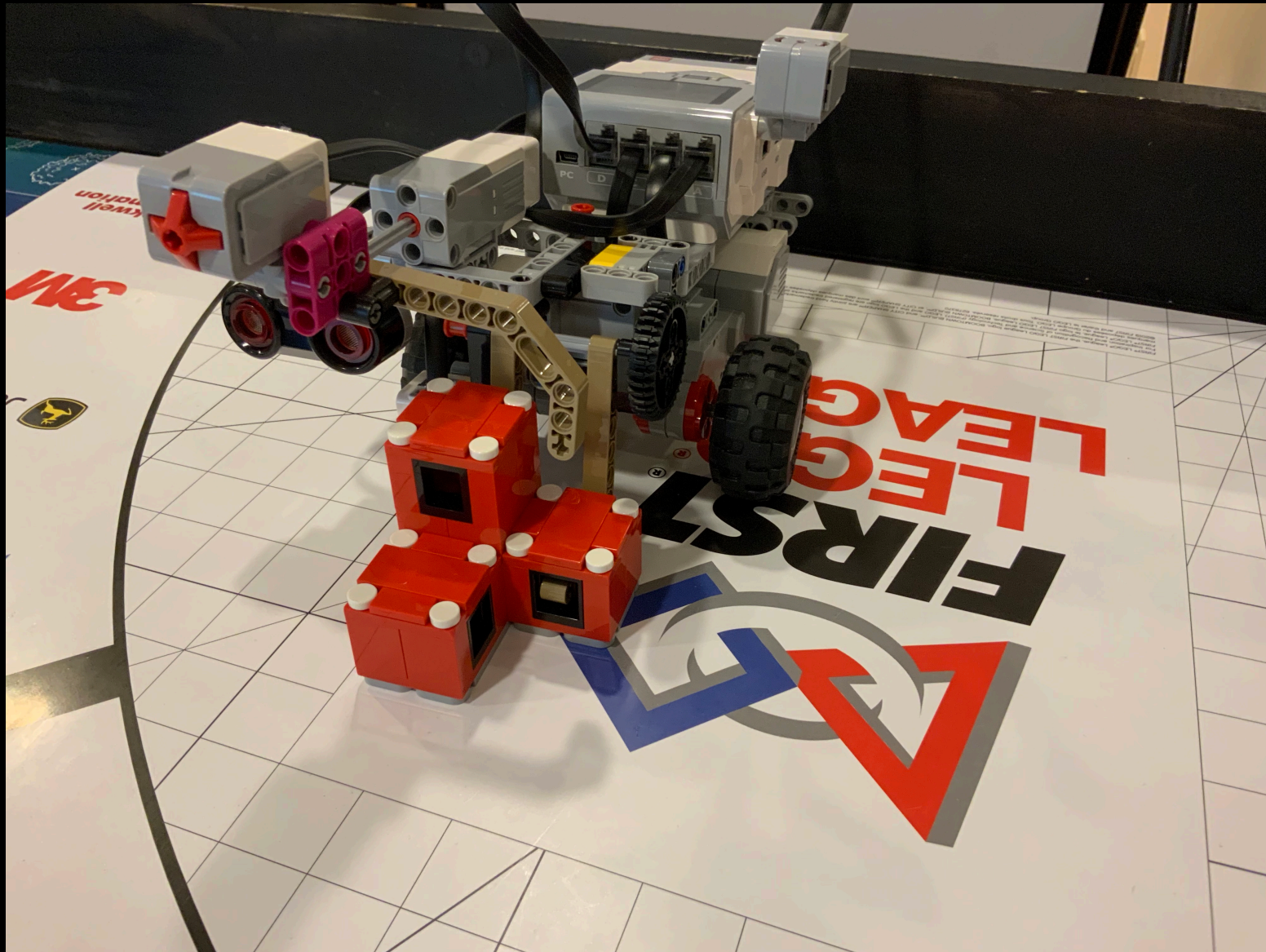
EV3DEV DEVICE BROWSER
Click here to connect to a device

Exercițiu: Deplasare de Blocuri

Programați robotul vostru să plece din baza spre cercul cel mai din stânga !

Robotul începe în sfertul de cerc (baza); plasați un bloc albastru în fața robotului, și încercați să împingeți blocul spre cercul, să se oprească în interior.

Încercați cu un bloc roșu.



Proiectează și
Construiește



Misiunea 12: Proiectează și Construiește

→ Loc: Dacă există cercuri cu cel puțin un bloc de culoarea respectivă, complet în interior și plat în jos:

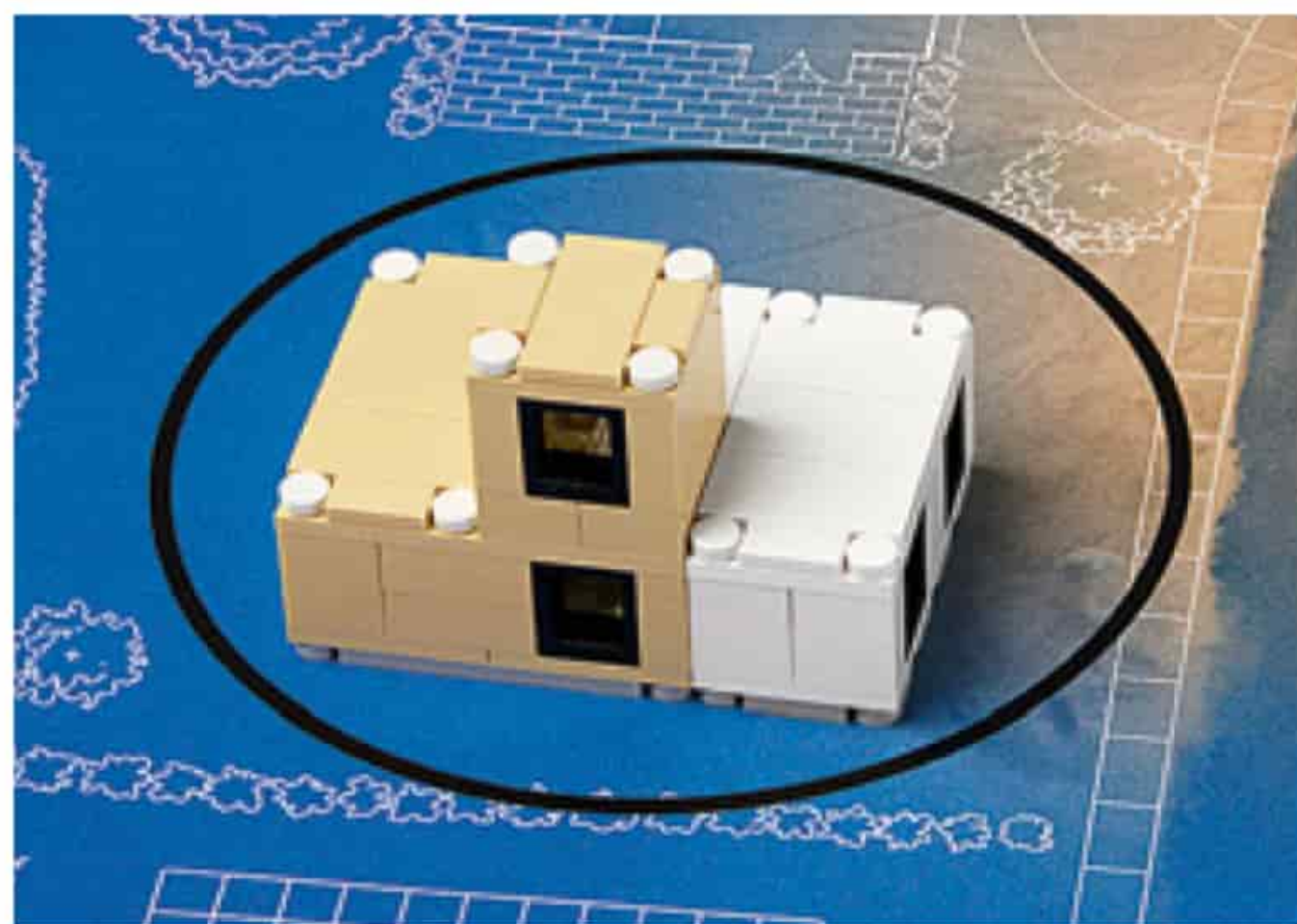
10 PUNCTE PENTRU FIECARE CERC

Notă: Cercul albastru nu face parte de Misiunea 12.

→ Înălțime: Dacă există stive independente în orice cerc, adaugă toate înălțimile împreună:

5 PUNCTE PENTRU FIECARE NIVEL

Notă: O stivă este una sau mai multe unități de construcție cu nivelul 1 care ating covorașul și orice niveluri superioare care ating planul la nivelul de dedesubt.



Culoare: Diferită

Stiva bej = 2 nivele

Stiva albă = 1 nivel

15 Puncte



Culoare: Diferită

Stiva cu pod: 4 nivele

20 Puncte



Culoare: Roșu (La fel)

Stiva roșie: 2 nivele

Ailaltă stivă: 4 nivele

40 Puncte

PseudoCode și Grafice

Roboții au nevoie de etape clare și precise!

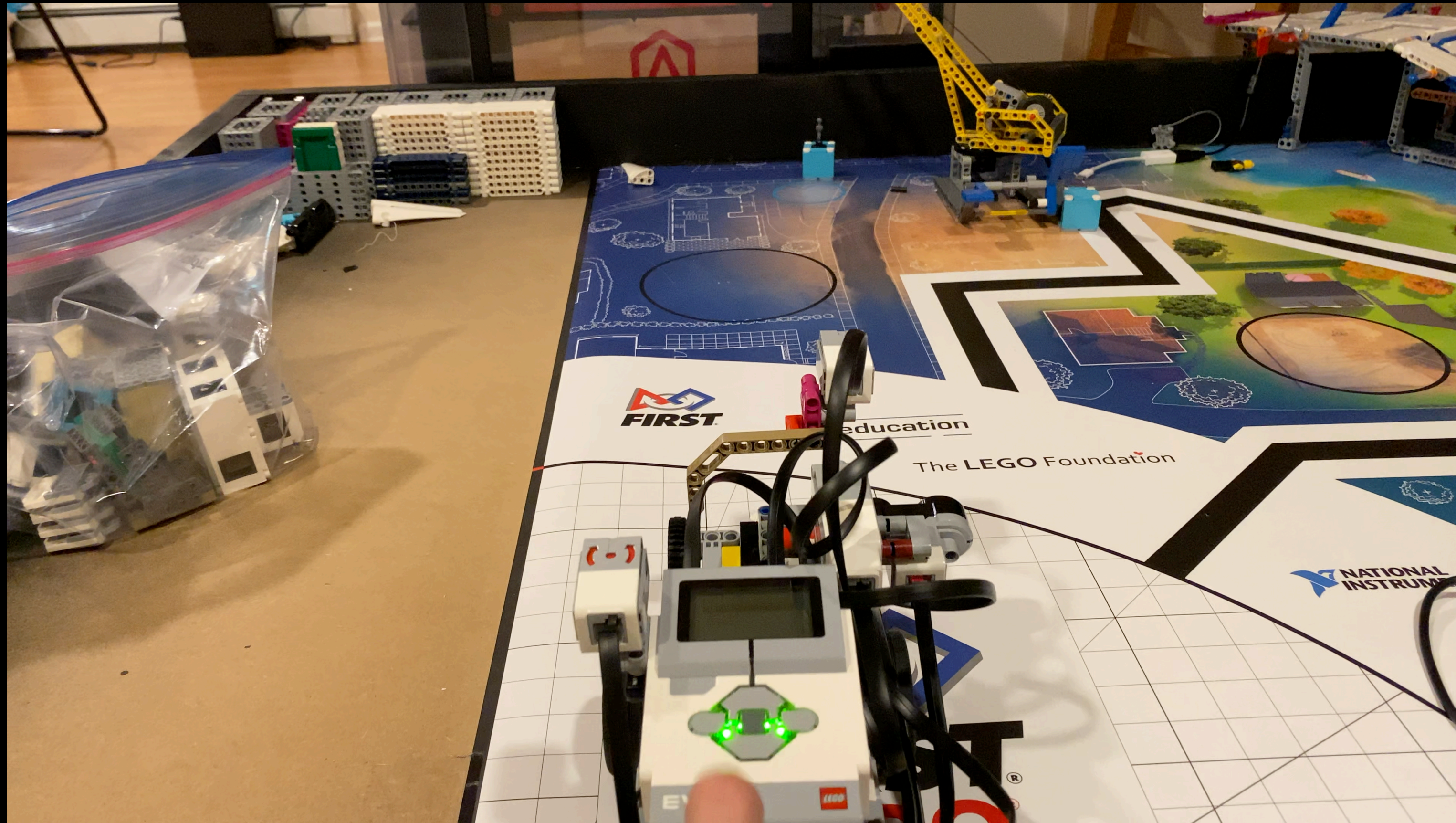
Pseudo-code: un ansamblu de notițe detaliate cu care un informatician poate scrie codul pentru robotul.

- Desenăm traseul pe tabla de joc
- Măsurăm distanta
- Folosim metodele Clasei Drive:
 - `drive(speed, steering)`
 - `straight()`



Fixați poziția inițială robotului (mereu aceeași)
Distanța de parcurs = ?

Rezolvat



Inainte

Create your objects here.

```
ev3 = EV3Brick()
```

Write your program here.

```
ev3.speaker.beep()
```

```
left_motor=Motor(Port.B)
```

```
right_motor=Motor(Port.C)
```

```
robot= DriveBase(left_motor,right_motor,wheel_diameter=62.4,axle_track=80)
```

```
ev3.speaker.beep()
```

```
speed =100
```

```
robot.drive(speed,0)
```

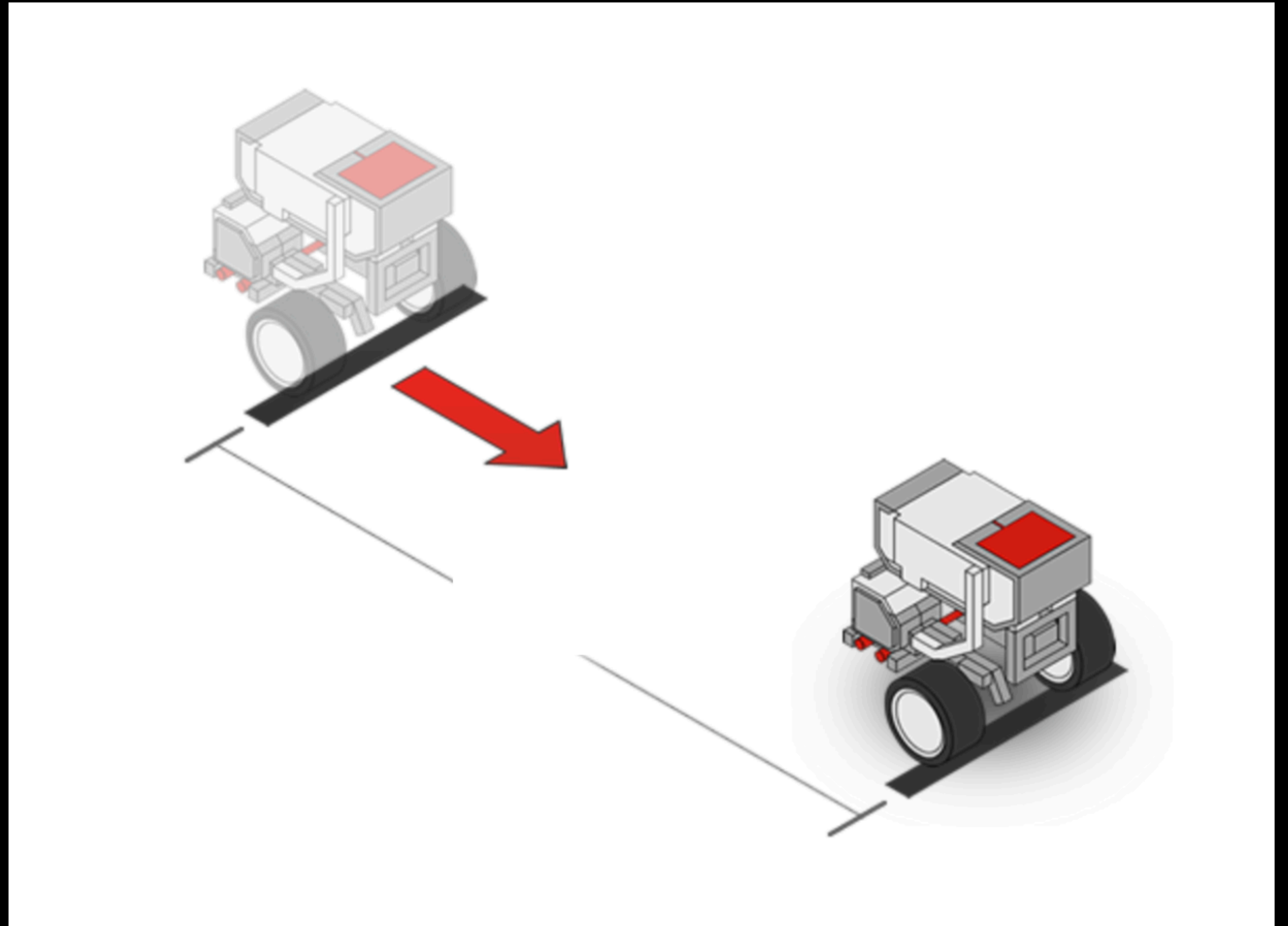
```
while (robot.distance())<390):
```

```
    time.sleep(0.0001)
```

Si inapoi ?

Metode relevante ale Clasei Drive:

- `drive(speed, steering)`
- `straight()`



Si inapoi ?

adaugam la programul de dinainte

```
robot.drive(-speed,0)
while (robot.distance()>0):
    time.sleep(0.001)
robot.drive(0,0)
```

Sfârșit !

- Momentul pentru întrebări.
- Ne vedem data viitoare !

