შეხვედრა 5: სენსორების სამყარო – როგორ „გრძნობს“ არდუინო

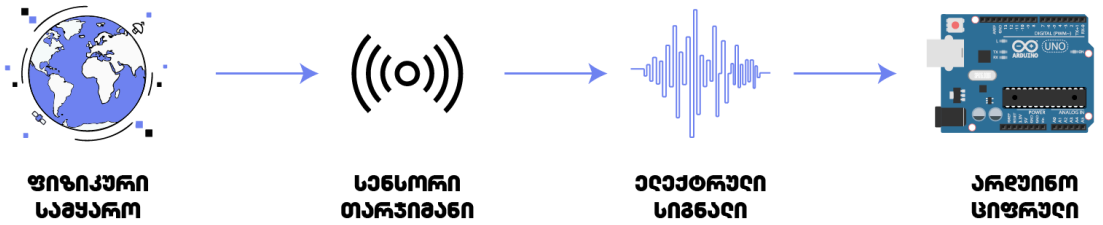
გამარჯობა! წინა შეხვედრაზე შენ შენი პირველი მოქმედი პროექტი შექმენი და შუქდიოდი ააციმციმე. შენ არდუინოს ასწავლე, როგორ შეასრულოს მოქმედება. დღეს კი ჩვენ მას ვასწავლით, როგორ „იგრძნოს“ მის გარშემო არსებული სამყარო. დღეს კი ჩვენ მას ვასწავლით, როგორ „იგრძნოს" მის გარშემო არსებული სამყარო..

1. სენსორების შესავალი

1.1. რა არის სენსორი და როგორ გარდაქმნის ის ფიზიკურ სიდიდეს ელექტრულ სიგნალად

სენსორები – არდუინოს „გრძნობის ორგანოები“: ჩვენ, ადამიანები, სამყაროს ხუთი ძირითადი გრძნობით აღვიქვამთ: მხედველობა, სმენა, ყნოსვა, გემო და შეხება. არდუინოსაც აქვს თავისი გრძნობის ორგანოები – ესენია **სენსორები**.

როგორ მუშაობს სენსორი? სენსორის მთავარი ამოცანაა, რაიმე ფიზიკური მოვლენა (მაგალითად, სინათლის სიკაშკაშე, ტემპერატურა ან მანძილი) გაზომოს და გადააქციოს ელექტრულ სიგნალად, რომლის „წაკითხვაც“ და გაგებაც არდუინოს შეუძლია. ფაქტობრივად, სენსორი არის თარჯიმანი ფიზიკურ სამყაროსა და არდუინოს ციფრულ სამყაროს შორის.



1.2. ციფრული და ანალოგური სენსორების მუშაობის პრინციპები

ისევე, როგორც ჩვენი გრძნობები, სენსორებიც სხვადასხვანაირად მუშაობენ. ძირითადად, ისინი ორ ტიპად იყოფიან:

ციფრული სენსორები (Digital Sensors):

- **მუშაობის პრინციპი:** ისინი მუშაობენ „კი - არა“ პრინციპით, როგორც შუქის ჩამრთველი. მათ შეუძლიათ, არდუინოს უთხრან მხოლოდ ორი რამ: „კი“ (HIGH) ან „არა“ (LOW).

- **მაგალითი:** უბრალო ღილაკი ციფრული სენსორია. ის ან დაჭერილია (HIGH), ან არა (LOW). ის ვერ გვეტყვის, *რა სიძლიერით* დააჭირეს ღილაკს.

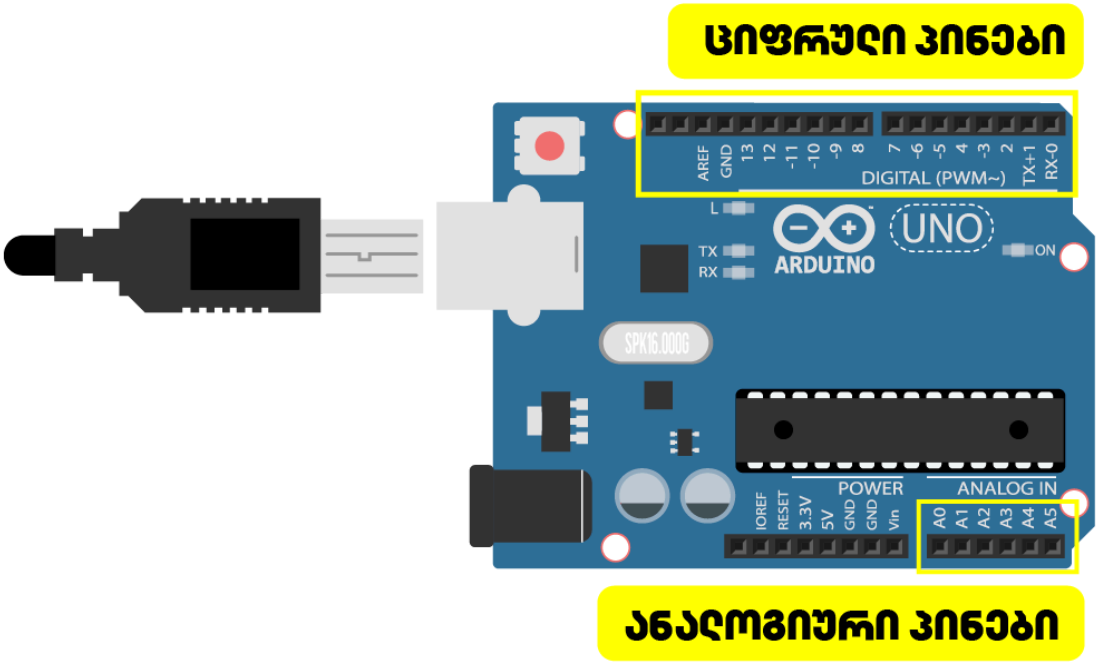
ანალოგური სენსორები (Analog Sensors):

- **მუშაობის პრინციპი:** ციფრულისგან განსხვავებით, ანალოგური სენსორები ბევრად მეტ ინფორმაციას გვაწვდიან. ისინი არ გვეუბნებიან უბრალოდ „კი/არა“-ს, არამედ გვაძლევენ მნიშვნელობების მთელ დიაპაზონს.

- **მაგალითი:** სინათლის სენსორიანალოგურია. ის არ გვეუბნება უბრალოდ „ნათელია“ გარემო თუ „ბნელი“. ის ზომავს, *რამდენად* ნათელია და გვაძლევს ზუსტ მნიშვნელობას – „ძალიან ბნელა“, „ცოტა ბნელა“, „საშუალოდ ნათელია“, „ძალიან ნათელია“.

Arduino-ში პინების დანაწილება:

ამ ორი ტიპის სენსორების სწორად დასაკავშირებლად Arduino-ში სპეციალურად გამოყოფილია პინები. ციფრული სენსორებისთვის გამოიყენება **digital პინები**, ხოლო ანალოგური სენსორებისთვის — **analog პინები**. ასეთი განაწილება საშუალებას გვაძლევს, სენსორების ტიპის მიხედვით სწორად დავუკავშიროთ ისინი მიკროკონტროლერს და მივიღოთ/გავცეთ საჭირო ინფორმაცია.



1.3. სენსორების როლი რეალური სამყაროს მონაცემების შეგროვებაში

სენსორებიაგროვებენ ინფორმაციას (მონაცემებს) გარემოდან, რის საფუძველზეც არდუინო იღებს გადაწყვეტილებებს. მაგალითად:

- თუ სინათლის სენსორი იტყვის, რომ „ბნელა“, არდუინომ შეიძლება აანთოს ნათურა.

- თუ ტემპერატურის სენსორი იტყვის, რომ „ცხელა“, არდუინომ შეიძლება ჩართოს ვენტილატორი.

- თუ მანძილის განმსაზღვრელი სენსორი იტყვის, რომ „წინ დაბრკოლებაა“, რობოტმა შეიძლება მიმართულება შეიცვალოს.

2. ანალოგური შემავალი სიგნალი (Analog Input)

დღეს ჩვენ ვიმუშავებთ ანალოგურ სენსორთან. ამისთვის უნდა ვისწავლოთ, როგორ წავიკითხოთ ანალოგური სიგნალი.

2.1. `analogRead(pin)` ფუნქცია: ანალოგური პინიდან ძაბვის დონის წაკითხვა (0-1023 დიაპაზონში)

ეს არის მთავარი ბრძანება, რომლითაც არდუინო კითხულობს ინფორმაციას ანალოგური პინებიდან (A0, A1, A2...).

როგორ მუშაობს? ანალოგური სენსორი პინზე აწვდის ძაბვას 0-დან 5 ვოლტამდე. `analogRead()` ფუნქცია ამ ძაბვას კითხულობს და გარდაქმნის რიცხვად **0-დან 1023-მდე** დიაპაზონში.

- **0** ნიშნავს, რომ პინზე ძაბვა 0 ვოლტია (სიგნალი არ არის).
- **1023** ნიშნავს, რომ პინზე ძაბვა 5 ვოლტია (მაქსიმალური სიგნალი).
- შუალედური მნიშვნელობები (მაგალითად, 512) ნიშნავს, რომ ძაბვა დაახლოებით 2.5 ვოლტია.

სინტაქსი: `int sensorValue = analogRead(A0);` (ეს ნიშნავს: „წაიკითხე მნიშვნელობა A0 პინიდან და შეინახე ის `sensorValue` ცვლადში“).

2.2. ვირტუალური ელექტრული სქემის აწყობა ანალოგური სენსორით (ფოტორეზისტორის მაგალითზე)

ფოტორეზისტორი: ეს არის სინათლის სენსორის ერთ-ერთი ტიპი. მისი მთავარი თვისება ისაა, რომ ის ელექტრულ დენს სხვადასხვანაირად ატარებს სინათლის ინტენსივობის მიხედვით. რაც უფრო მეტი სინათლე ხვდება მას, მით უფრო ნაკლებია მისი წინაღობა (და პირიქით).

ძაბვის გამყოფი: იმისთვის, რომ არდუინომ ეს ცვლილება გაიგოს, ფოტორეზისტორს სქემაში ვრთავთ ჩვეულებრივ რეზისტორთან ერთად. ეს ქმნის „ძაბვის გამყოფს“, რომელიც არდუინოს ანალოგურ პინზე ცვალებად ძაბვას ქმნის.

AnITa-ს პლატფორმაზე: შენ ამ სქემის აწყობაზე ფიქრი არ გჭირდება! ჩვენს სიმულატორში ფოტორეზისტორი უკვე დაკავშირებულია A0 პინთან. შენ მხოლოდ კოდის დაწერა გევალება.

2.3. წაკითხული მნიშვნელობების გამოტანა სერიულ მონიტორზე

როგორ დავინახოთ, რას კითხულობს სენსორი? ამისთვის ჩვენ უკვე ნაცნობ სერიულ მონიტორს გამოვიყენებთ.

loop() ფუნქციაში analogRead()-ით წავიკითხავთ სენსორის მნიშვნელობას, შევინახავთ ცვლადში და შემდეგ Serial.println() ბრძანებით დავბეჭდავთ ამ ცვლადის მნიშვნელობას. რადგან ეს კოდი loop()-შია, ჩვენ რეალურ დროში დავინახავთ, როგორ იცვლება სენსორის მონაცემები.

ფოტორეზისტორით აწყობილი წრედის ნიმუში tinkercad-ში:

https://drive.google.com/drive/folders/1pzRu7t3LTUaN4cxNjfQZ7rOiu0-UBz_S //5.1

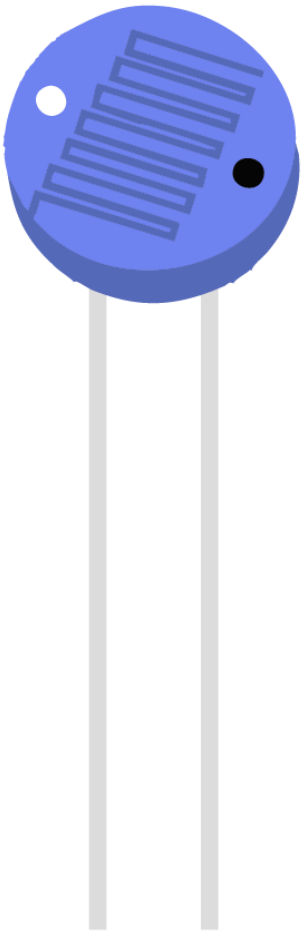
ვიდეოში ჩანს ფოტორეზისტორის მუშაობის პრინციპი. სინათლის ცვლილებასთან ერთად serial monitor−ზე იცვლება გამოტანილი ცვლადის მნიშვნელობაც.

3. სენსორების ტიპების მიმოხილვა

ამ კურსის განმავლობაში ჩვენ სამ ძირითად ანალოგურ სენსორს შევისწავლით:

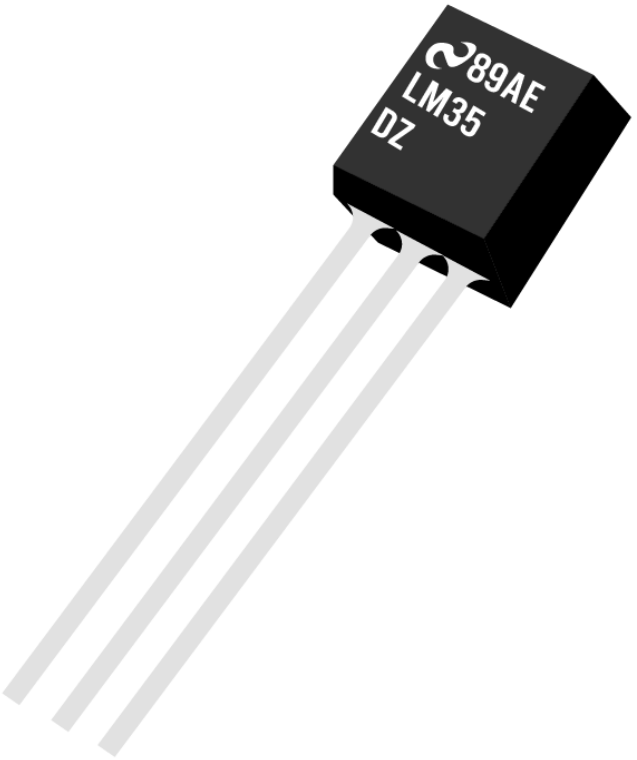
3.1. სინათლის სენსორი (ფოტორეზისტორი)

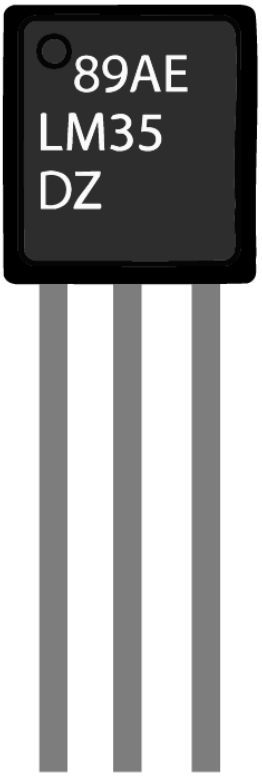
- რას ზომავს? სინათლის ინტენსივობას.
- რაში ვიყენებთ? ავტომატური განათების სისტემებში, რობოტებში, რომლებიც სინათლის მიმართულებას იმეორებენ და ა.შ.



3.2. ტემპერატურის სენსორი

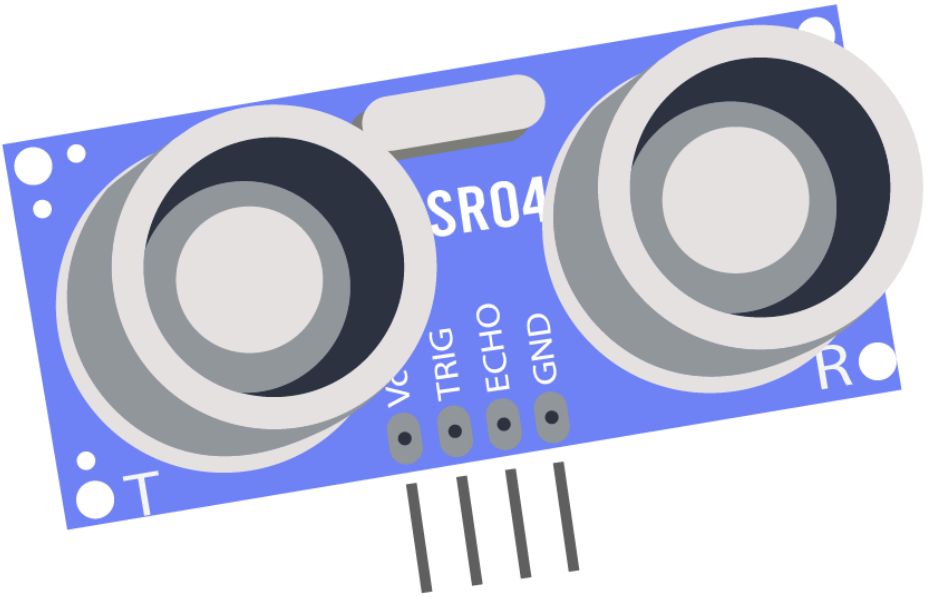
- რას ზომავს? გარემოს ტემპერატურას.
- რაში ვიყენებთ? ამინდის სადგურებში, ჭკვიან თერმოსტატებში, გადახურებისგან დამცავ სისტემებში.

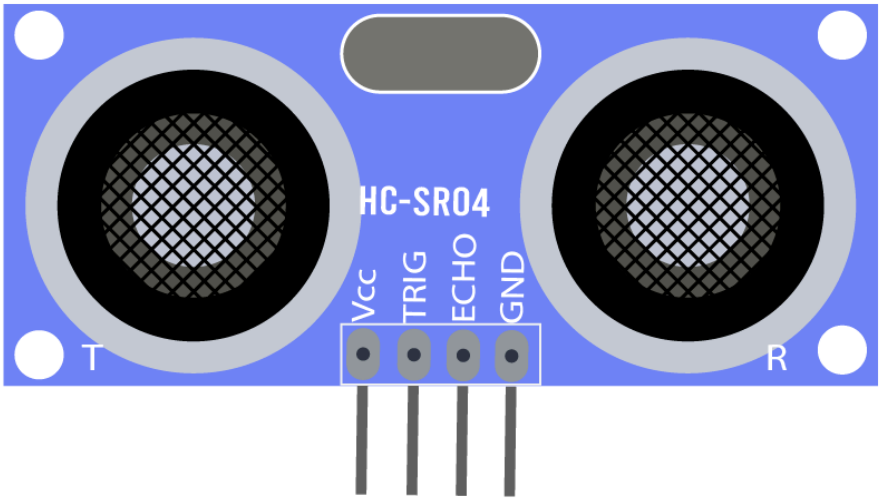

89AE
LM35
DZ



3.3. მანძილის განმსაზღვრელი სენსორი

- რას ზომავს? მანძილს უახლოეს ობიექტამდე.
- რაში ვიყენებთ? რობოტებში დაბრკოლებების თავიდან ასარიდებლად, მანქანის პარკირების სისტემებში, ავტომატური კარის მოდელში.


SR04
VC
TRIG
ECHO
GND
T
R



დავალება 5: სენსორთან მუშაობის პირველი გამოცდილება

შენი ამოცანაა, დაწერო პროგრამა, რომელიც მუდმივად წაიკითხავს სინათლის სენსორის (ჩვენს შემთხვევაში ფოტორეზისტორის) ნედლ მონაცემს (0-1023 დიაპაზონში) და დაბეჭდავს მას სერიულ მონიტორზე. პროგრამის გაშვების შემდეგ გამოიყენე არდუინოს პანელზე არსებული სლაიდერი, შეცვალე ვირტუალური განათების დონე და დააკვირდი, როგორ იცვლება რიცხვები სერიულ მონიტორზე.

დავალება 5.1: შეცდომის პოვნა და გასწორება

პროგრამისტს კოდის წერისას შეცდომა მოუვიდა. ანალოგური პინიდან მონაცემების წამკითხველი ფუნქცია არასწორად არის დაწერილი. შენი ამოცანაა, იყო კოდის დეტექტივი, იპოვო და გაასწორო შეცდომა, რათა პროგრამამ ჩვენი დავალება შეასრულოს და გამართულად იმუშაოს.

მინიშნება: ყურადღება მიაქციე, როგორ იწერება ფუნქციების სახელები C++ ენაში. დიდი და პატარა ასოები მნიშვნელოვანია!

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  // იპოვე შეცდომა ამ ხაზში
  int sensorValue = analogread(A0);

  Serial.println(sensorValue);
  delay(100);
}
```

დავალება 5.2: კოდის დასრულება

პროგრამის ძირითადი ნაწილი უკვე აგებულია, სერიული მონიტორი მზად არის სამუშაოდ. თუმცა, მთავარი ნაწილი — სენსორის მონაცემების წაკითხვა და მათი დაბეჭდვა — გამორჩენილია. შენი ჯერია! დაამატე ორი გამოტოვებული ბრძანება `loop` ფუნქციაში, რათა პროგრამა დასრულდეს.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  // --- ჩაამატე პირველი ბრძანება აქ ---
```

```
  // წაიკითხე მნიშვნელობა ანალოგური პინიდან A0 და შეინახე ცვლადში.


  // --- ჩაამატე მეორე ბრძანება აქ ---

  // დაბეჭდე ამ ცვლადის მნიშვნელობა სერიულ მონიტორზე.


  delay(100);

}
```

დავალება 5.3: შეიმუშავე პროგრამული კოდი

დაწერე პროგრამული კოდი, შექმენი პროგრამა, რომელიც შეასრულებს შემდეგ სამუშაოს:

1. `setup` ფუნქციაში მოამზადებს სერიულ მონიტორს სამუშაოდ;
2. `loop` ფუნქციაში მუდმივად წაიკითხავს მნიშვნელობას ანალოგური პინიდან A0;
3. დაბეჭდავს ამ მნიშვნელობას სერიულ მონიტორზე;
4. გააკეთებს 100 მილიწამიან პაუზას წაკითხვებს შორის.

```
void setup() {

  // დაწერე setup ფუნქციის კოდი აქ.

}


void loop() {

  // დაწერე loop ფუნქციის კოდი აქ.

}
```

სწორი პასუხი (პროგრამული კოდი სრულად):

```
void setup() {
  // 1. მოამზადე სერიული მონიტორი სამუშაოდ.
  Serial.begin(9600);
}

void loop() {
  // 2. წაიკითხე მნიშვნელობა ანალოგური პინიდან A0
  //    და შეინახე ის int ტიპის ცვლადში (მაგალითად, sensorValue).
  int sensorValue = analogRead(A0);

  // 3. დაბეჭდე ამ ცვლადის მნიშვნელობა სერიულ მონიტორზე.
  Serial.println(sensorValue);

  // 4. დაამატე მცირე პაუზა, რათა მონაცემები ძალიან სწრაფად არ გამოვიდეს
  //    და ადვილი წასაკითხი იყოს.
  delay(100);
}
```

```JSON
{
  "version": 1,
  "board": "arduino:avr:uno",
  "steps": [
    {
      "type": "compile"
    },
```

```
    {
      "type": "static",
      "rules": [
        {
          "id": "serial_begin",
          "name": "Serial initialization",
          "kind": "require_regex",
          "pattern":
"Serial\\.begin\\s*\\(\\s*9600\\s*\\)\\s*;?",
          "flags": "iu",
          "must_pass": true,
          "source": "stripped",
          "msg_fail": "სერიული მონიტორი უნდა ინიციალიზდეს
9600 სიჩქარით: Serial.begin(9600);",
          "msg_pass": "სერიული მონიტორი სწორად არის
ინიციალიზებული."
        },
        {
          "id": "analog_read_a0",
          "name": "Read from A0 pin",
          "kind": "require_regex",
          "pattern": "analogRead\\s*\\(\\s*A0\\s*\\)",
          "flags": "iu",
          "must_pass": true,
          "source": "stripped",
          "msg_fail": "სენსორის მნიშვნელობა უნდა
წაიკითხოს A0 პინიდან: analogRead(A0);",
          "msg_pass": "A0 პინიდან წაკითხვა სწორადაა."
        },
        {
          "id": "store_sensor_value",
          "name": "Store sensor value in variable",
          "kind": "require_regex",
          "pattern":
"int\\s+\\w+\\s*=\\s*analogRead\\s*\\(\\s*A0\\s*\\)",
          "flags": "iu",
```

```
          "must_pass": true,
          "source": "stripped",
          "msg_fail": "წაკითხული მნიშვნელობა უნდა
შეინახოთ int ტიპის ცვლადში.",
          "msg_pass": "სენსორის მნიშვნელობა ინახება
ცვლადში."
        },
        {
          "id": "serial_print",
          "name": "Print to Serial Monitor",
          "kind": "require_regex",
          "pattern":
"Serial\\.print(?:ln)?\\s*\\(\\s*\\w+\\s*\\)",
          "flags": "iu",
          "must_pass": true,
          "source": "stripped",
          "msg_fail": "სენსორის მნიშვნელობა უნდა
დაიბეჭდოს სერიულ მონიტორზე:
Serial.println(sensorValue);",
          "msg_pass": "მნიშვნელობა იბეჭდება სერიულ
მონიტორზე."
        },
        {
          "id": "has_delay",
          "name": "Delay for readability",
          "kind": "require_regex",
          "pattern": "delay\\s*\\(\\s*\\d+\\s*\\)\\s*;?",
          "flags": "iu",
          "must_pass": true,
          "source": "stripped",
          "msg_fail": "უნდა დაემატოს პაუზა, რათა
მონაცემები ადვილად წაიკითხოს: delay(100);",
          "msg_pass": "პაუზა დამატებულია."
        },
        {
          "id": "correct_sequence",
```

```
          "name": "Correct sequence in loop",
          "kind": "require_ordered_regex",
          "patterns": [

  "int\\s+\\w+\\s*=\\s*analogRead\\s*\\(\\s*A0\\s*\\)",

  "Serial\\.print(?:ln)?\\s*\\(\\s*\\w+\\s*\\)",
            "delay\\s*\\(\\s*\\d+\\s*\\)"
          ],
          "flags": "iu",
          "must_pass": true,
          "source": "stripped",
          "msg_fail": "loop()-ში თანმიმდევრობა უნდა იყოს:
  1) analogRead(A0), 2) Serial.println(), 3) delay().",
          "msg_pass": "ოპერაციების თანმიმდევრობა სწორია."
        }
      ]
    }
  ]
}
```

Sim Elements html

```
<div style="display: flex; flex-direction: column; align-items: center;
gap: 20px; margin-bottom:
 20px;">
    <wokwi-photoresistor-sensor id="light-sensor"
pin="A0"></wokwi-photoresistor-sensor>
    <label style="font-weight: bold;">Light Sensor (A0)</label>
```

```
</div>

<div class="distance-control">
    <label>A0 Sensor Value: <span
data-sensor-display="A0"></span></label>
    <input
        type="range"
        data-sensor="A0"
        min="0"
        max="1023"
        class="distance-slider"
    />
</div>
```

Sim elements j

```
return {
    onInit: function(runner, sensorValues) {
        if (sensorValues.value.A0 === undefined) {
            sensorValues.value.A0 = 512;
        }
    }
};
```