Robotics Starter Course

1. Testing the equipment
2. Hello World in Ardublockly
3. How DC motor works
4. Controlling the DC motor with Digital Outputs
5. Geared reductor
6. Constructing the mobile robot
7. Controlling the robot
8. Programming functions
9. Programming loops - For
10. Writing program in C++
11. Digital sensors
12. Reading digital sensors
13. S-R-A loop
14. Hello World in Arduino IDE
15. Controlling the robot
16. Programming loop - FOR
17. Reading digital input
18. S-R-A loop
19. Reading analog input
20. Avoiding obstacles
21. Light sensor
22. Line follower
23. Variables
24. Barrier gate construction
25. Reference point.
26. Reed switch and pull-up resistors
27. On-module Buttons and pull-up resistors
28. Potentiometer as angle sensor
29. LCD(i2c)
30. Using variables to store data
31. Stop the robot at the end of line
32. Using LCD
33. Controlling the motors with PWM
34. Proportional control loop
35. Proportional-Integral control loop

36. Installing Aruino IDE
37. Installing Ardublockly
38. Installing RobDuino library

Testing the equipment {#1}

Basic testing in Arduino IDE

- Connect the Arduino Uno to PC with proper USB cable.
  `[Arduino Uno]` -> □ -> □
- Open Arduino IDE program and open program with:
  `Files □  Examples □ 01. Basics □ Blink.ino`
- Make sure that you will set the proper settings (follow the visual instructions). From the menu choose:
  `Tools`□
  1. `Board:` Arduino/Genuino Uno
  2. `Port:` COM3
- To upload the code you can click the icon `Upload`.
  If the uploading was successful you will be prompted with the text like:

```
Done uploading.
Sketch uses 970 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local var
```

Basic testing in Ardublockly

- Connect the Arduino Uno to PC with proper USB cable.
  `[Arduino Uno]` -> □ -> □
- Run Ardublockly program. Which will be running as localhost and you will be using internet browser as IDE. The addres will be:
  `http://localhost:8000/ardublockly/index.html`
- In the left corner of the program you can find `[=] menu icon`. From where you can choose (Slide 2 and 3)
  `[] Settings`:
  1. `Compiler Location:` C:\Program Files (x86)\Arduino\arduino_debug.exe
  2. `Arduino Board:` Uno

3. Com port: COM3
4. And press: [RETURN]

- Finaly you can press button PLAY And if uploading was successful you will be prompted with the text (Slide 4):

```
Successfully Uploaded Sketch
WARNING: Error loading hardware folder /home/david/Arduino/hardware/WAV8F
No valid hardware definitions found in folder WAV8F.
Sketch uses 444 bytes (1%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local var
```

> **Summary:**
>
> <++>

> **Issues:**
>
> **Ardublockly returns the Error id 55: Serial port Serial Port unavailable.**
> Try to reconect the Arduino board. Wait a moment, check the settings and choose the COM port again then try again.

Hello World in Ardublockly {#2}

Task:

- Make a very simple program like setting the digital output bit D3 to logical state 1 or **HIGH**.
- Send the program to Arduino controller.

Questions:

1. What is the voltage of the digital output pin D3?
2. Try to compare and understand the C++ programming code in aside window.

Summary:

<++>

⚠ Issues:

<++>

How DC motor works {#3}

Task:

Connect the DC motor to terminals of the battery. You can try different combinations like: + and -; - and +; - and -; + and +.

Questions:

1. In which direction the motor's shaft spins?
2. In which direction the electric current flow?
3. Why does motor not spinning when both connectors are connected to + terminal of the battery?

⚠ Summary:

The rotation of the DC motor depends on the direction of electric current.

⚠ Issues:

When I connect the DC motor to + and - terminals of the battery the motor's shaft does not spin.

Check the voltage of the battery... battery may be discharged.
Check the connectors of the motor... may be bad.

Controlling the DC motor with Digital Outputs {#4}

Task:

- Connect the DC motor to Digital Output D7 and D6.
- Write the program and check all the combinations of digital outputs; 00, 01, 10 and 11.

Questions:

1. For each combination of digital outputs mark the state of the motor.
2. Try to stop the shaft of the DC motor for a short time and try to remember how hard is it?

Summary:

The motor's shaft is spinning according to the direction of the electric current trough the motor. The torque is weak.

Issues:

Geared reductor {#5}

Task:

- Add geared reductor to DC motor.
- Try to stop the shaft of the geared reductor.

Questions:

1. How hard is to stop the shaft of the reductor in comparison to shaft of the motor.
2. How fast the shaft of the reductor is spinning in comparison to the shaft of the motor?
3. Are you able to freely rotate the shaft of the reductor by hand?
4. What happened with the produced mechanical power?
5. Try to calculate the geared ratio of the reductor.

Summary:

Gead ratio

The gear ratio describing the ratio between the angular velocity of input gear G1 and angular velocity of output gear G2.
$$ { i=\frac{\omega_1}{\omega_2} } $$
Because each gear moves tooth per tooth and if two touching gears have different numbers of teeths their's angular velocity will be different. In fact the anguar velocity will be inversely proportional.
$$ {\frac{\omega_1}{\omega_2}=\frac{N_2}{N_1}=i} $$

⚠ Issues:

The reductor's shaft is not spinning although the DC motor is working properly.

Check if the reductor is all the way connected on the motor. Check if the worm gear of the motor is in contact with first gear of the roductor.

Constructing the mobile robot {#6}

Tasks:

- Construct the mobile robot according to this video.
- Add the battery between the red cornered bricks. The connector shuld be pointing to the back of the robot.
- Add also the RobDuino controller. Clip the controller between the red bricks with the grove

Questions:

1. Where do you think is th front side of the robot?
2. Are you able to rotate the wheels freely by hand?

⚠ Summary:

<++>

<++>

⚠ Issues:

<++>

<++>

Controlling the robot {#7}

Tasks:

- Connect LEFT MOTOR to digital outputs:

  - D7 and D6
- Connect RIGHT MOTOR to digital outputs:
  - D5 and D4
- Now you can write the program to control both motors in order to move the robot FORWARD for 3 second and STOP.
- Next you can write the program which will move the robot in several different directions:
  - forward
  - backward
  - turn left
  - turn right

Questions:

1. How many digital outputs you have to set in order to control the robot for specific move?
2. How many different moves your robot can make?

Summary:

Controlling the robot in two degrees of freedom

To controlling the robot in two degrees of freedom we need to control two motors. Since we have to set two digital outputs for each motor we have to set four digital outputs for each move.

Issues:

When I change the direction of the robot the robot does not move as expected.

Probably you did not set all of the outputs correctly. Remember taht some outputs may have remained set in previous output state from taken action in previous task.

Programming functions {#8}

Tasks:

- Write a programming function which includes the certain programming steps in order to move the robot in specific direction.
- Write also other functions like:
  - robotForward()

  - robotStop()
  - robotLeft()
  - robotRight()
  - robotBackward()
- Write larger program to move the robot allover the classroom.

Questions:

1. What would happened if several robots would have the same program?
2. Can you change the program in a way that robot would repeat the program?
3. What happens if the mobile robot run into an obstacle?

Summary:

<++>

<++>

Issues:

<++>

<++>

Programming loop - FOR {#9}

Tasks:

- If we want to repeat some programming instructions for several times we can use For Loop.
- For example the next program repeats the functions robotLeft() and **robotRight()** for **10 times** and robot will do a funny ”dancing” move.
- Experiment a bit more with such programming techniques.

Questions:

1. <++>
2. <++>

Summary:

<++>

<++>

Issues:

<++>

<++>

Writing program in C++ {#10}

Tasks:

- Make a really basic program with easy task like is shown on the slide 1.
- Open the Arduino IDE program.
- Copy-Paste all the c++ code from Ardublockly into Arduino IDE.
- Experiment with the c++ code.

Questions:

1. <++>
2. <++>

Summary:

<++>

<++>

Issues:

<++>

<++>

Digital sensors {#11}

Tasks:

- In sake to detect the obstacles we have to equip robot with the ”touch sensor”. This sensor is basically a switch or key, which toggles it’s output between GND and +5 V voltage potentials.
- Follow video instructions to construct bumper in front of the robot.

Questions:

1. Do you hear ”clicking” sound when you push the bumper?
2. Name the mechanical mechanism where smaller force on one end can cause greater force on the other end of the mechanism.

Tasks:

The key has three connecting terminals. Each of one is marked with the number 1, 2 or 3. Connect them in right order. Connect the key terminals in order that are specified in presentation and listed as:

1. connect to RobDuino C0 terminal.
2. connect to RobDuino voltage terminal GND.
3. connect to RobDuino voltage terminal +5V.

<++>

Questions:

1. What is the output voltage of the sensor when the robot is (or is NOT) touching the obstacle?
2. How many different states are presented with such sensor?
3. Name several more examples where digital sensor can take place.

Summary:

Digital sensors

The output of a digital sensor can be just in two states:

- logical ”0” - presented in voltage as 0 V.
- logical ”1” - presented in voltage as +5V.

⚠ Issues:

Robot has no power since I connected the key as a sensor.

Probably the key or switch is connected wrong and there is short connection between the GND and +5V voltage terminals. Unconnect the key or switch and verify if the power is back.

Reading digital input {#12}

Tasks:

- Write the program shown in the presentation to test the readings of the digital sensor.
- Then... complete the program to turn OFF the LED when the bumper is not touching anything.
- Next... Change IF statements into single one IF-THEN-ELSE statement.
- Advanced... Solve the problem without IF statement.

Questions:

1. Check if the LED on the output terminal D3 is turend ON when the bummper is pressed.

⚠ Summary:

<++>

<++>

⚠ Issues:

<++>

<++>

S-R-A loop {#13}

Tasks:

- Write the program to drive the robot around the class and avoid the obstacles.
- Using the S-R-A loop technique you should write the program in particular order:

1. Check the sensor. IF the bummper ...
2. ... is pressed the robot has to stop/go back/turn.
3. ... is not pressed the robot can drive forward.

Questions:

1. Would this routine also work in Arduino run first function (check the program in Slide 2)?
2. <++>

⊠ Summary:

Senzoning-Reasoning-Acting Loop

S-R-A loop is the most important thing in robotics.

⊠ Issues:

It seems that the program is not working right ... like it would be ignoring the value of the sensor.

Probably the S-R-A loop is not actually a loop. Check the program if the input is read just onces or is read continuously.

Hello World in Arduino IDE {#14}

Tasks:

- Make a very simple program like setting the digital output bit D3 to logical state 1 or **HIGH**.
- Send the program to Arduino controller.

Questions:

1. What is the difference between:
    1. `void setup()`
    2. `void loop()`
2. What is the difference between:
    1. `pinMode(3, OUTPUT);`
    2. `digitalWrite(3, HIGH);`

```
void setup() {
  // put your setup code here, to run once:
  pinMode(3, OUTPUT);
  digitalWrite(3, HIGH);
}

void loop() {
  // put your main code here, to run repeatedly:

}
```

⊠ Summary:

Using curly braces -

Using curly braces in C++ is important part of writing the programming code. Imagine that you want to merge several members of programing code to a single pile. As we would separate pencils into one pile and markers to another - to be more organized. In real life we would do by elastic bundle or rope. If you have to choose single character from the keyboard to indicate that several members are combined to the same pile - which character would you choose? Probably curly braces {} are the best choice.

Function Declaration

```
void loop() {

}
```

Function Call

```
digitalWrite(3, HIGH);
```

Function Name

Function name should be stucked together from 2 - 5 short words that unicly describing the function-alaty of the function. The first word shoul start with lower case and all the others words following should start with upper case. Some examples should be:

```
badname();
```

```
goodFunctionName();
```

⊠ Issues:

Error: expected ';' before 'something'

Probably you forgot to put ; (semicolon) at the end of the command. Find the row starting with "**something**" and look the row above... probably missing "**;**".

Controlling the robot {#15}

Tasks:

- Declare your own functions to control the robot. Choose proper names like:
  - robotForward()
  - robotStop()
  - robotLeft()
  - robotRight()
  - robotBackward()
- Don't forget to specify the direction (INPUT or OUTPUT) of all used pins you need.

Questions:

1. <++>
2. <++>

```
void setup() {
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);

  //simple move...
  robotForward();
  delay(3000);
  robotStop();
}
```

```
void loop() {

}

void robotForward(){
  digitalWrite(7, HIGH);
  digitalWrite(6, LOW);
  digitalWrite(5, HIGH);
  digitalWrite(4, LOW);
}

void robotStop(){
  digitalWrite(7, LOW);
  digitalWrite(6, LOW);
  digitalWrite(5, LOW);
  digitalWrite(4, LOW);
}
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Programming loop - FOR {#16}

Tasks:

- If we want to repeat some programming instructions for several times we can use For Loop.
- For example the next program repeats the functions robotLeft() and **robotRight()** for **10 times** and robot will do a funny ”dancing” move.
- Experiment a bit more with such programming techniques.

Questions:

1. <++>
2. <++>

```
 void setup() {
   pinMode(4, OUTPUT);
   pinMode(5, OUTPUT);
   pinMode(6, OUTPUT);
   pinMode(7, OUTPUT);

   // Funny dancing move.
   int i = 0;
   for (i = 0; i < 10; i++) {
     robotLeft();
     delay(100);
     robotRight();
     delay(100);
   }
   robotStop();
 }

[+]void loop() {
[+]void robotForward() {
[+]void robotStop() {
[+]void robotLeft() {
[+]void robotRight() {
[+]void robotBackward() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Reading digital input {#17}

Tasks:

- Write the program shown aside of this text.
- Then... complete the program to turn OFF the LED when the bumper is not touching anything.
- Next... Change IF statements into single one IF-THEN-ELSE statement.
- Advanced... Solve the problem without IF statement.

Questions:

1. Check if the LED on the output terminal D3 is turend ON when the bummper is pressed.

```
void setup() {
  pinMode(A0, INPUT);
}

void loop() {
  if ( digitalRead(A0) == HIGH){
    digitalWrite(3, HIGH);
  }
}
```

```
[+]void robotForward() {
[+]void robotStop() {
[+]void robotLeft() {
[+]void robotRight() {
[+]void robotBackward() {
```

Summary:

IF Statement

If statement can be written in several forms. The easiest one is:

```
if ( value_one == value_two ){
  statement1;
  statement2;
```

```
}
```

It can be expanded into IF-ELSE form:

```
if ( value_one == value_two ){
  statement1;
  statement2;
}else{
  statement3;
}
```

⊠ Issues:

<++>

<++>

S-R-A loop {#18}

Tasks:

- Write the program to drive the robot around the class and avoid the obstacles.
- Using the S-R-A loop technique you should write the program in particular order:
    1. Check the sensor. IF the bummper ...
    2. ... is pressed the robot has to stop/go back/turn.
    3. ... is not pressed the robot can drive forward.

Questions:

1. <++>
2. <++>

```
[+]void setup() {

   void loop() {
     if ( digitalRead(A0) == HIGH){
       robotStop();
     }else{
       robotForward();
```

```
    }
  }

[+]void robotForward() {
[+]void robotStop() {
[+]void robotLeft() {
[+]void robotRight() {
[+]void robotBackward() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Reading analog input {#19}

Tasks:

- Unmount robot's bumper and all connections to the switch.
- Equip the robot with distance sensor according to video and scheme.
- Copy & Paste next program and test the serial output.

Questions:

1. What kind of values do you getting from the reading of the distance sensor?
2. Find the reasonable value vhere you shuld stop the robot.

Summary:

Analog to digital converter - ADC

ADC is an electronic sistem that converts analog signal (voltage) to a digitalized values. In our particular case the range of an analog voltage from 0V to 5V is converted to range of numbers from 0 to 1023.

⊠ Issues:

<++>

<++>

Avoiding obstacles {#20}

Tasks:

Write the program to drive the robot around the class and avoid the obstacles.

1. Check the value of distance sensor. If the distance if far away (smaller number) ...
2. ... the robot can drive forward.
3. ...else ... the robot must to stop/go back/turn.

Questions:

1. <++>
2. <++>

```
[+]void setup() {

   void loop() {
     if ( analogRead(A0) < 400 ){
       robotForward();
     } else {
       robotStop();
     }
    }

[+]void robotForward() {
[+]void robotStop() {
[+]void robotLeft() {
[+]void robotRight() {
```

```
[+]void robotBackward() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Light sensor {#21}

Tasks:

- Construct the light sensor according to video and scheme. Add also the light bulb which will help to lightning the area beneath the robot.
- To test the light sensor and light bulb copy&paste next program and check the reported serial data:

Questions:

1. What is the value of the sensor when the robot is over white/black area?
2. Calculate the average between those two values.

Summary:

Sensors

Sensors are electronic devices which convert physical quantity into electrical quantity (usually voltage). In simplest setup, sensor can be constructed as voltage divider with two resistors - R_1 and R_2. One of the resistors is resistor with fixed resistance value (eg. R_1 = 10 kΩ). The second one is a bit special and it's resistance depends on some physical quantity (e.g. light, temperature, humidity...). When combining those two resistors into such voltage divider the output of the voltage divider can be calculated as: $$ U_{Out} = \frac{R_1}{R_1 + R_2} U_0 $$

⚠ Issues:

<++>

<++>

Line follower {#22}

Tasks:

- Write the program to control the robot to follow the line (actually above the edge between black and white area).

Questions:

1. What is the program function to get the `light_sensor_value`?
2. Determine the movements of the robot if the robot is over the black area and if the robot is over the white area.

```
[+] void setup() {

    void loop() {
     if ( light_sensor_value < treshold_value ){
       //do this if robot is ower the blck line

     } else {
       // do this if robot is ower white area

     }
    }

[+] void robotForward() {
[+] void robotStop() {
[+] void robotLeft() {
[+] void robotRight() {
[+] void robotBackward() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Variables {#23}

Tasks:

- Stop the robot when it reaches the end of line.
- Detecting the end of line can be done by measuring the time that robot spend over the black and white area. E.g. if the robot is driving along the line - the time spent over black and time spent over white area will be quite the same. When line ends the robot will not detect the black area soon and the time spent over white area will increase significantly - and that is the trigger for detecting the end of line.
- Advanced: Make a function to align (move) the robot back to the line.

Questions:

1. How can we store a data to the controller's memory?
2. How can we measure time in programming loops?
3. <++>

```
[+] void setup() {

    int time_on_black = 0;
    int time_on_white = 0;

    void loop() {
     if ( analogRead(A1) < 400 ){
                          // BLACK area
       robotLeft();
```

```
      time_on_white = 0; // reset time on white
      time_on_black++;   // meas. time on black
      delay(100);
    } else {
                         // WHITE area
      robotRight();
                         // Do similar meas.
                         // of time on white

      delay(100);
                         // If time is signif.
                         // longer:
                         // robotStop();exit(0);
    }
   }
```

```
[+] void robotForward() {
[+] void robotStop() {
[+] void robotLeft() {
[+] void robotRight() {
[+] void robotBackward() {
```

⊠ Summary:

What is variable?

Variables are used in C++ where you will need to store any type of values within a program and whose value can be changed during the program execution. These variables can be declared in various ways each having different memory requirements and storing capability. Variables are the name of memory locations that are allocated by compilers, and the allocation is done based on the data type used for declaring the variable.

Variable definition and initialization in C++

A variable definition means that the programmer writes some instructions to tell the compiler to create the storage in a memory location. The syntax for defining variables is:

```
data_type variable_name;
```

Here `data_type` means the valid C++ data type which includes int, float, double, char, wchar_t, bool

and variable list is the lists of variable names to be declared which is separated by commas. Variables are declared in the above example, but none of them has been assigned any value. Variables can be initialized, and the initial value can be assigned along with their declaration.

```
data_type variable_name = value;
```

Examples:

```
bool is_raining = false;
int sensor_value = 23;
float pi_value = 3.14;
char letter = "A";
char sentence[] = "Some dummy text.";
```

Time measuring with programming loops

The easiest way to measure time is to simply count the number of loop's executions. And if we know how long is one execution of the loop - we can easily determine the time lapsed for the whole process.

Example:

```
int t = 0;
while (t<10){
  t++;
  delay(100);
}
```

In the previous example the `while` loop is executed 10 times (t = [0 .. 9]), since each execution of the loop last 100 ms (determined by `delay(100);`) the whole `while` loop last 1 s.

Time measuring with Timers

More proper way of measuring the time is by using the timer's values. More on that can be read here.

Example:

```
unsigned long start_time;
unsigned long stop_time;

start_time = millis();
// time measured process goes here
// ...
stop_time = millis();
```

```
  unsigned long duration = stop_time - start_time;
```

Where the `duration` is the time in milliseconds.

⚠ Issues:

<++>

<++>

Barrier gate construction {#24}

Tasks:

- Construct the barrier gate accordant to video instructions.
- Connect the motor to digital outputs D7 and D6.
- Write the program to rising and lowering the barrier. The main functionalities (e.g. Up, Down, Stop) should be written in program functions:
  - `gateUp();`
  - `gateDown();`
  - `gateStop();`

Questions:

1. What is time for rising and lowering the barrier? Compare it to your colleague's situation.
2. What is disadvantage of time controlled loop.

```
[-] void setup() {
      pinMode(7, OUTPUT); //declaration of I/O pins
      pinMode(6, OUTPUT);

      gateUp();           // Lift the barrier.

      delay(3000);        // Wait a bit...

      gateDown();     // Lower the barrier.
```

```
    }

[+] void loop() {
[+] void gateStop(){
[-] void gateUp() {
      digitalWrite(7, HIGH);
      digitalWrite(6, LOW);
      delay(1000);
      gateStop();
    }
[+] void gateDown() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Reference point. {#25}

Tasks:

- Add a switch to detect the reference point of the barrier gate. Let the reference point be the closed position of the barrier.
- Connect the switch to the controller according to schematics.
- Change the program to lower the barrier gate to reference switch position.

Questions:

1. Why is detection of reference point important?
2. <++>

Summary:

<++>

<++>

Issues:

<++>

<++>

Reed switch and pull-up resistors {#26}

Tasks:

- Add a reed switch to the front of the barrier gate to detect the car.
- Connect the reed switch to the input pin A1 an GND.

Questions:

1. What is pull-up resistor?
2. How can we turn on the internal pull-up resistor of the microcontroller?

```
 [-]void setup() {
      pinMode(7, OUTPUT); //declaration of I/O pins
      pinMode(6, OUTPUT);
      pinMode(A1, INPUT_PULLUP); // Enab. int. Pull-up.
    }

 [-]void loop() {
      if ( digitalRead(A1) == 0 ){
        gateUp();
        delay(3000);
        gateDown();
      }
    }
[+] void gateStop(){
[+] void gateUp() {
```

```
[+] void gateDown() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

On-module Buttons and pull-up resistors {#27}

Tasks:

- Module RobDuino includes two ”on-board” buttons which are connected from pin A4 and A5 to GND. This two buttons can allso be used but internal pull-up resistors must be turned on.
- Add manual functionality to the automated barrier gate. Add the possibility to manually lift (e.g. press A4 button) and lower (A5 button) the barrier gate.

Questions:

1. <++>
2. <++>

```
[+] void setup() {
[-] void loop() {
      if ( digitalRead(A1) == 0 ){
        gateUp();
        delay(3000);
        gateDown();
      }

      manualControll();
    }
[-] void manualControll(){
```

```
        if ( digitalRead(A4) == 0 ){ //Button A4 is pressed
          gateUp();
        }
                                      // add code for A5 ...
    }
[+] void gateStop(){
[+] void gateUp() {
[+] void gateDown() {
```

Summary:

<++>

<++>

Issues:

<++>

<++>

Potentiometer as angle sensor {#28}

Tasks:

- Add the potentiometer to the shaft of barrier gate.
- Test the potentiometer values with next program:

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

void loop() {
  Serial.println(analogRead(A3));
  delay(100);
}
```

- Change the functions for lifting and lowering the barrier gate to use potenciometer readings instead of switch and time controlled movement.

```
[+] void setup() {
[+] void loop() {
[+] void manualControll(){
[+] void gateStop(){
[-] void gateUp() {
      while (analogRead(A3) < 750){
        digitalWrite(7, HIGH);
        digitalWrite(6, LOW);
      }
      gateStop();
}
[+] void gateDown() {
```

Questions:

1. What is the value of the angle sensor when the barrier gate is in the upper orientation...
2. ... and in lower orientation.

Summary:

<++>

<++>

Issues:

<++>

<++>

LCD(i2c) {#29}

Tasks:

- <++>
- <++>

Questions:

1. <++>
2. <++>

[Visual instructions.]

Summary:

<++>

<++>

Issues:

<++>

<++>