Minicurso de Arduino

Emanuelle e Heitor

Março 1, 2023

Conteúdos

1 Arduino **2**
1.1 Estrutura . . . 2
1.2 Sensores e Atuadores . . . 3
1.2.1 Instalando a IDE do Arduino . . . 3
1.2.2 Conhecendo a IDE do Arduino . . . 4
1.2.3 Estrutura de Código . . . 5
1.2.4 Hello, World - LED Blink . . . 5
1.2.5 Funções Bult In do Arduino . . . 6
1.2.6 O que é um sensor e um atuador? . . . 6
1.2.7 LDR + LED . . . 6
1.2.8 Código LDR comanda LED . . . 8
1.3 Projetos segundo dia . . . 9
1.3.1 Considerações Sobre o Minicurso . . . 11

Contatos

Instagram:
@giparvca
@manu_eg
@tor.rld
GitHub:
TorRLD

1 Arduino

Segundo o Laboratório de Tecnologia e Sistemas da Informação (Acesso em 02/2023), o Arduino é uma plataforma eletrônica open source, que tem como objetivo integrar hardware e software de maneira fácil, permitindo que pessoas com pouco conhecimento na área possam desenvolver as suas habilidades e aprendizado de maneira mais simples, aprendendo a eletrônica básica e programação.

1.1 Estrutura

Existem vários tipos de placas Arduino, mas a que vamos trabalhar é talvez a mais conhecida - Arduino Uno (Figura 1).

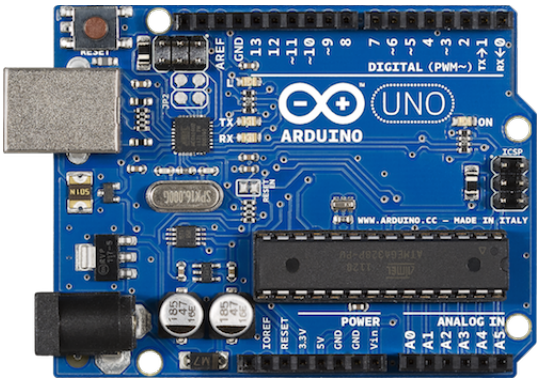

Figure 1: Arduino Uno

Essa placa contém portas digitais, analógicas, de 5 e 3.3 Volts, terra e de comunicação serial. Se observar a Figura, pode-se notar as características das portas.

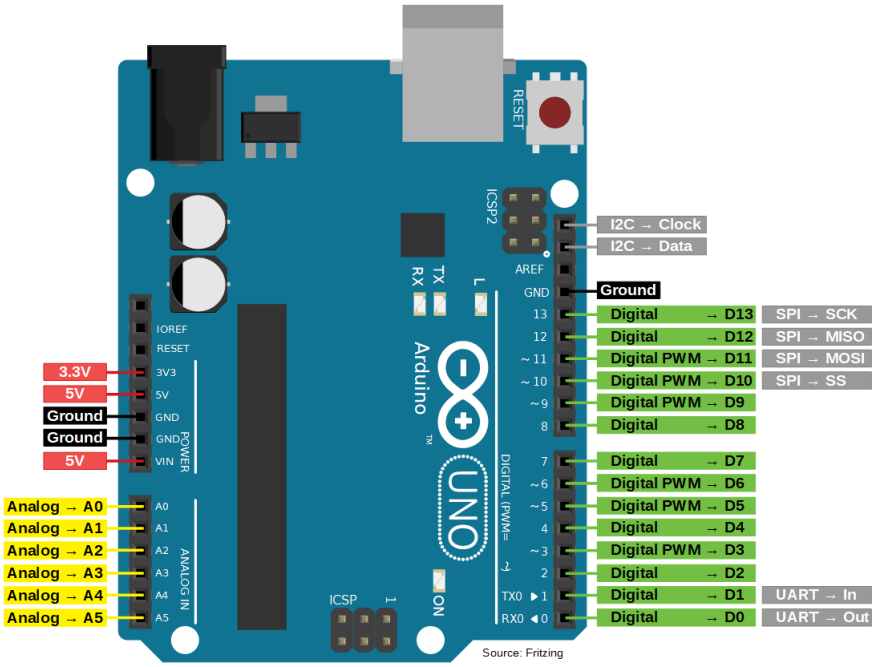

Figure 2: Pinagem do Arduino Uno

Mas o que são sinais analógicos e digitais?

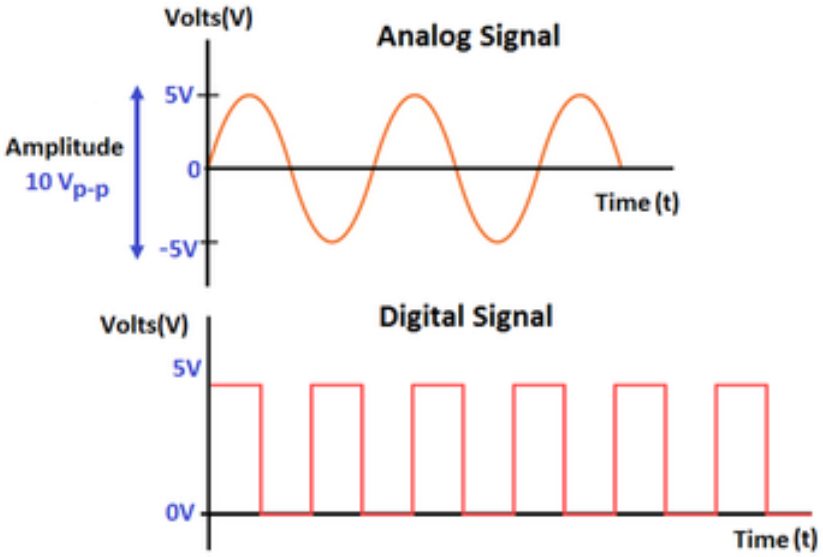

Figure 3: Sinal digital e analógico

Os sinais analógicos variam no tempo de forma contínua, enquanto o sinal digital é descontínuo (discreto) e possui apenas dois valores - HIGH ou LOW (1 ou 0).

Mas por que saber disso é importante?

No mundo real, os sentidos captam informações analógicas. "E o que isso quer dizer?". Vejamos alguns exemplos: Quando você aperta um botão para ligar um LED utilizando o arduino, você fez isso em um estante de tempo e a pressão sobre o botão variou desde o estado inicial (completamente aberto), passando para os estados intermediários e até chegar ao estado final (completamente fechado). Nota-se que neste exemplo as posições dos botões não foram binárias, mas isso apenas para o mundo fora do Arduino. Olhando da perspectiva do microcontrolador, ele recebeu dois estados - sinal baixo (botão não pressionado) e sinal alto (botão pressionado).

1.2 Sensores e Atuadores

Com o Arduino se pode fazer várias coisas, programar LED's para acender conforme a intensidade da luz recebida por um LDR, fazer com que motores girem em sentidos determinados pelo programador e muitas outras coisas. Mas você já parou para pensar como um LDR influencia na intensidade da luz emitida pelo LED?

Nesta seção, iremos entender o que são sensores e atuadores e como eles funcionam, começando aqui a utilizar as funções da biblioteca "Arduino.h" na IDE do arduino. Mas antes, vamos aprender como instalá-la.

1.2.1 Instalando a IDE do Arduino

Para se instalar a IDE do Arduino, primeiramente você terá que acessar o site oficial do arduino (disponível em: arduino.cc). Logos depois deverá clicar na aba "Software" e em seguida rolar a página e encontrar a sessão "Downloads" e clicar na versão que condiz com o seu sistema operacional. Vejam os passos nas figuras a seguir.

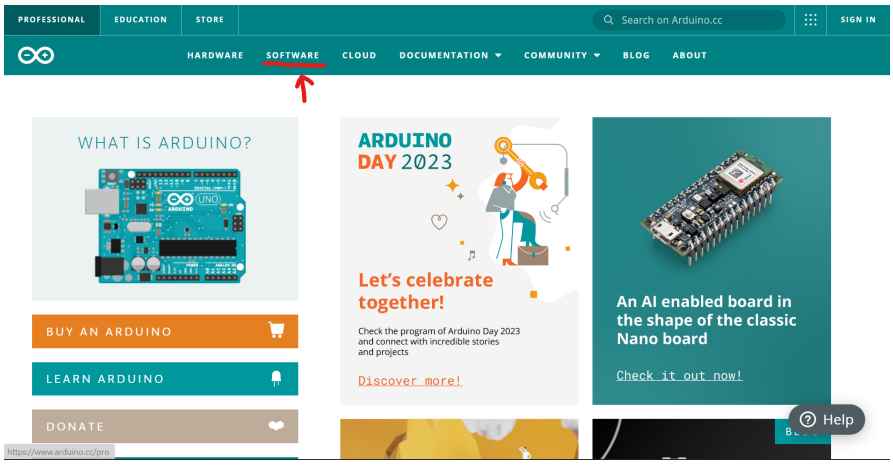

Figure 4: Passo 1: Clique na aba "Software"

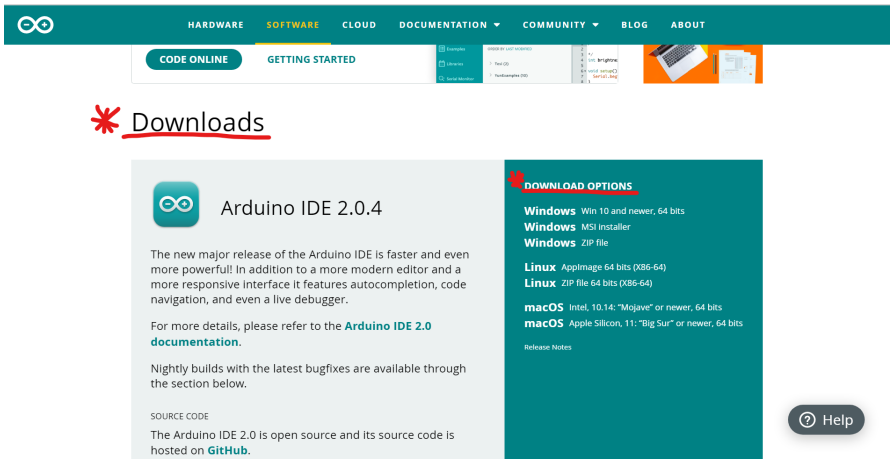

Figure 5: Passo 2: Vá até a sessão "Downloads" e selecione a opção adequada para você

1.2.2 Conhecendo a IDE do Arduino

Após o download, inicie a instalação e, por fim, abra a IDE do Arduino. Depois de abrir, você verá uma interface como mostrada na figura abaixo.

Figure 6: Arduino IDE 2.0

Na Figura 6, as principais funções e ambientes da IDE foram representadas, sendo que:

- Verify/Upload, é o botão que compila o código e faz o upload para a placa, respectivamente.
- Select Board Port, é onde você escolhe o modelo da sua placa Arduino e seleciona em qual porta ela está conectada (no Windows, acesso o gerenciamento de dispositivos e veja em qual COM a placa está conectada).
- Open Serial Monitor, é onde será mostrada as informações que o programador solicitar com um algoritmo específico.
- Open Serial Plotter, é onde será mostrada as informações em forma de gráficos.
- Library Manager, é a aba onde se pode adicionar bibliotecas para complementar um algoritmo, além de gerenciar as bibliotecas existentes.
- Board Manager, é a aba onde se pode adicionar um modelo de plataforma eletrônica que não veio por padrão na IDE, além de gerenciar as existentes.
- Search, é o campo onde se pode fazer a pesquisa por caracteres no algoritmo.

1.2.3 Estrutura de Código

A estrutura de código da IDE contém duas funções principais: void setup() e void loop(). Veja a figura a seguir.

Figure 7: Estrutura de código da IDE do Arduino

Onde a função void setup() executa os comandos de configurações iniciais da placa Arduino e a função void loop() executa o algoritmo criado pelo programador enquanto o arduino está ligado, até que haja uma interrupção.

1.2.4 Hello, World - LED Blink

Como em toda linguagem de programação que se aprende, com a plataforma eletrônica Arduino não é diferente. A linguagem utilizada no Arduino é C++, você talves já tenha feito o algoritmo "Hello, World!" nesta linguagem. Mas com a placa em mãos, você irá fazer a versão do "Hello, World!" para o Arduino.

```
1
2   Blink
3
4   Turns an LED on for one second, then off for one second, repeatedly.
5
6   Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
7   it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
8   the correct LED pin independent of which board is used.
9   If you want to know what pin the on-board LED is connected to on your Arduino
10  model, check the Technical Specs of your board at:
11  https://www.arduino.cc/en/Main/Products
12
13  modified 8 May 2014
14  by Scott Fitzgerald
15  modified 2 Sep 2016
16  by Arturo Guadalupi
17  modified 8 Sep 2016
18  by Colby Newman
19
20  This example code is in the public domain.
21
22  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
23 */
24
25 // the setup function runs once when you press reset or power the board
26 void setup() {
27   // initialize digital pin LED_BUILTIN as an output.
28   pinMode(LED_BUILTIN, OUTPUT);
29 }
30
31 // the loop function runs over and over again forever
32 void loop() {
```

```
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Você pode acessar a aba Arquivos -> Exemplos -> 01.Basics -> Blink. Irá abrir uma nova janela com o código acima.

Nesse algoritmo mostrado anteriormente, determina que um LED fique com valor digital alto (acenda) por 1 segundo e depois faz com que o LED fique com valor digital baixo (apague) por 1 segundo, fazendo-o, assim, o LED piscar. Você pode ver os comentários no código acima.

1.2.5 Funções Bult In do Arduino

Segundo a IBM, uma função bult in é uma extensão de codificação para C e C++ que permite que um programador use a sintaxe de chamadas de função C e variáveis C para acessar o conjunto de instruções do processador da máquina compiladora.

Alguns exemplos de funções bult in do Arduino são:

- Serial.begin() - Configura a taxa de comunicação, ou baud rate, em bits por segundo.
- pinMode() - Define se o pino do arduino vai ser um pino de entrada ou saída.
- analogWrite() - manda um valor analógico que varia de 0 a 255 (8 bits).
- analogRead() - ler valor de uma porta analógica.
- digitalWrite() - manda um valor digital que pode ser HIGH ou LOW.
- digitalRead() - ler valor de uma porta digital.

1.2.6 O que é um sensor e um atuador?

Voltando ao exemplo do LED que acende de acordo com a captação de luz do LDR, neste exemplo o LDR é o sensor e o LED é o atuador.

- Sensor: dispositivo de detecção.

Ainda no mesmo exemplo, o LED funciona como um atuador.

- Atuador: elemento que atua de acordo a um comando.

Dessa forma, pode-se ser programado, utilizando o arduino, o exemplo citado.

1.2.7 LDR + LED

Um LDR é um sensor de luminosidade que varia sua resistência conforme a incidência da luz (do inglês Light Dependent Resistor). Ou seja, ele é um resistor que pode variar sua resistência que, geralmente, diminui com o aumento da intensidade da luz.

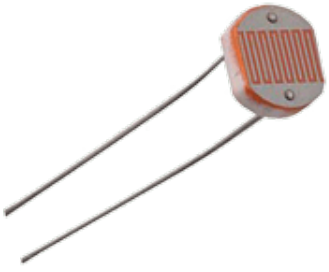

Figure 8: Sensor LDR 5mm

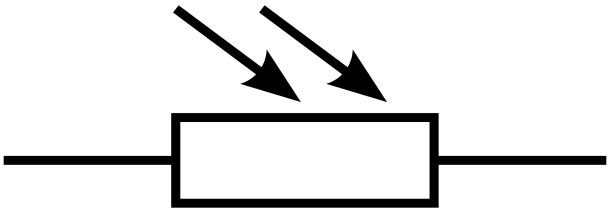

Figure 9: Símbolo do LDR em um diagrama de circuito elétrico.

Um LED é um dispositivo eletrônico que emite luz. Do inglês (Light Emissor Diode), ele é um diodo (dispositivo que conduz a corrente elétrica em uma direção) que emite luz.

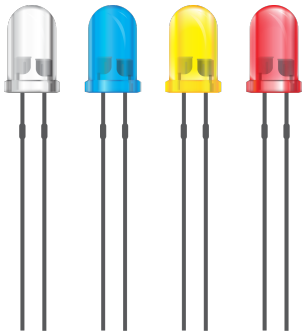

Figure 10: LED's

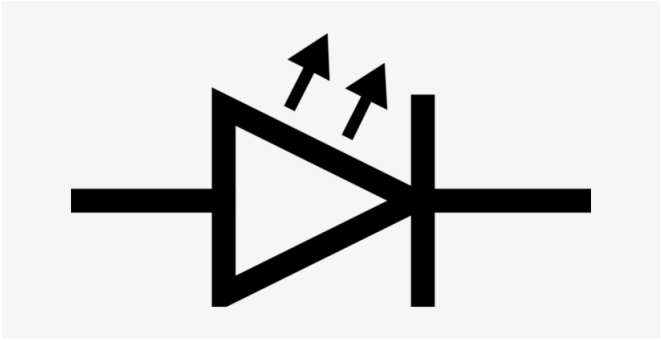

Figure 11: Símbolo do LED em um diagrama de circuito elétrico.

Na Figura 11, o lado esquerdo é o anodo (lado conectado ao positivo) e o direito é o catodo (lado conectado ao negativo).

Cada cor de LED tem uma tensão e corrente de funcionamento específica (Veja a Figura).

LEDs		
Cor do LED	Tensão em Volts (V)	Corrente em Miliamperes (mA)
Vermelho	1,8V – 2,0V	20 mA
Amarelo	1,8V – 2,0V	20 mA
Laranja	1,8V – 2,0V	20 mA
Verde	2,0V – 2,5V	20 mA
Azul	2,5V – 3,0V	20 mA
Branco	2,5V – 3,0V	20 mA

Figure 12: Tabela de tensão e corrente dos LED's de acordo com as cores.

Desta forma, precisa-se descobrir qual resistor será colocado em série com o LED, pois se não haver um, o LED receberá toda a tensão fornecida pelo arduino (vamos utilizar o pino de 5V) e ele irá queimar. Assim, pode-se calcular o valor da resistência utilizando a fórmula a seguir.

$$R = \frac{Tensão_{arduino} - Tensão_{LED}}{Corrente} \tag{1}$$

Onde a tensão do arduino é de 5 Volts, a do LED é de acordo com a tabela e a corrente tem valor de 20 miliAmperes.
Com o conhecimento visto até aqui, agora pode ser feito um algoritmo na IDE do Arduino para controlar a intensidade da luz do LED (ligado ou apagado) com um sensor LDR.
Deste modo, precisa-se fazer um diagrama de circuito elétrico. Para isso, vamos seguir os seguintes passos:

- Medir com um multímetro a variação de resistência do LDR de acordo com a faixa de luminosidade que ele será submetido.
- Escolher o LED e o resistor para ele de acordo com a Equação 1.
- Conectar o LED no negativo e no resistor, e este resistor ser conectado a um pino digital.

- De acordo com a variação do LDR obtida através das medições, utilizar um resistor que tenha um valor médio (o que recomendamos) entre o menor valor e o maior valor da resistência do LDR quando variou.
- Conectar o LDR no GND e o outro pino no resistor, sendo que este será conectado ao pino 5V.
- Assim, entre o resistor e o LDR, deverá haver uma conexão para um pino analógico.

O circuito construído se parecerá como o do diagrama mostrado a seguir.

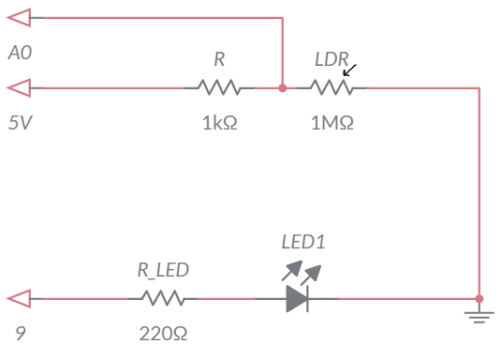

Figure 13: Diagrama do Circuito LDR controla LED

1.2.8 Código LDR comanda LED

```
1
2   /*
3  * Controle de LED dom LDR
4  * Autores: Emanuelle e Heitor (Gipar)
5  */
6
7  #define LED 9 //Pino do LED conectado a porta 9
8  #define LDR A0 //pino do LDR conectado a porta A0
9
10 void setup()
11 {
12   pinMode(LED, OUTPUT); //O LED vai emitir algo (luz), logo é saída
13 }
14
15 void loop()
16 {
17   int leitura = analogRead(LDR);  //O valor lido pelo LDR é armazenado na variável leitura
18
19
20   if (leitura > 500)  //Se o valor da leitura do LDR for maior que 500, o LED acende
21   {
22     digitalWrite(LED, HIGH);
23   }
24   else  //Se não, o LED apaga
25   {
26     digitalWrite(LED, LOW);
27   }
28
29   delay(100); // tem de espera de 100 milisegundos para cada medida
30 }
```

O funcionamento do algoritmo acima, está descrito nos comentários do próprio código. Lembrando que é muito importante ser feito todo o passo a passo mostrado anteriormente para se poder calcular os valores dos componentes, montar o circuito e programar o código.

Os valores captados na entrada A0 deverão ser medidos para se poder colocar nas condicionais do código (if e else). Pois aquele valor 500 em: "leitura > 500" é apenas demonstrativo.

1.3 Projetos segundo dia

Sensor Ultrassônico

A figura a seguir, exemplifica a pinagem do sensor ultrassônico.

Figure 14: Pinout Sensor Ultrassônico

Funções:

`PulseIn(pino, Valor)` = capta o valor em microssegundos.

Exemplo:

```
digitalWrite(5, HIGH);
delayMicrosseconds(10);
digitalWrite(5, LOW);
float dist = pulseIn(5, HIGH);
```

A função capta os valores de pulso alto do pino 5 por 10 microssegundos e armazena na variável `dist`.

A velocidade do som é de `340,29 m/s`, mas como a função capta os valores em microssegundos, precisa-se dessa velocidade em centímetros por microssegundos, que equivale a `0,0343 cm/us`

$$dist = dist * 0,0343 \tag{2}$$

Driver Ponte H (L298N)

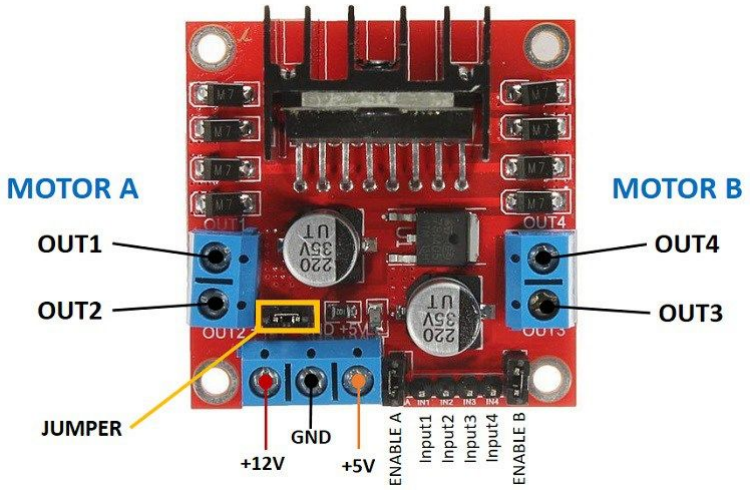

Figure 15: Driver de Motor L298N

Este driver possui seis pinos que vai interessar no desenvolvimento do algoritmo `ENA, IN1, IN2` (Para o motor 1) e `ENB, IN3, IN4` (Para o segundo motor).
Sendo que os pinos "IN" recebem valor 0 ou 1 e os pinos "EN" recebem valores analógicos de 0 a 255. E os pinos EN deverão ser conectados nos pinos PWM sendo que todos os pinos são definidos como saída Exemplo :

```
define EN1 9;
define EN2 10;
define IN1 5;
define IN2 6;
define IN3 7;
define IN4 8;
void setup(){
pinMode(EN1, OUTPUT);pinMode(EN2, OUTPUT);pinMode(
IN1, OUTPUT);pinMode(IN2, OUTPUT);pinMode(IN3, OUTPU
T);pin
Mode(IN2, OUTPUT);}
```

Os códigos dos projetos desenvolvidos vão estar disponíveis no github CLICANDO AQUI.

1.3.1 Considerações Sobre o Minicurso

Este minicurso teve o intuito de introduzir nos alunos conceitos básicos sobre programação com Arduino. Mas caso o curso tenha um bom feedback, poderá existir outras partes que abordem conceitos de hardware mais aprofundados, Arduino + ROS (Robot Operating System, framework de desenvolvimento robótico) e Arduino no VSCode + interface WEB (HTML, C++ e JavaScript).
E a você que leu esta apostila e participou do minicurso, eu agradeço muito pela experência que foi trocada e espero que continuem a aprender sobre robótica em geral.