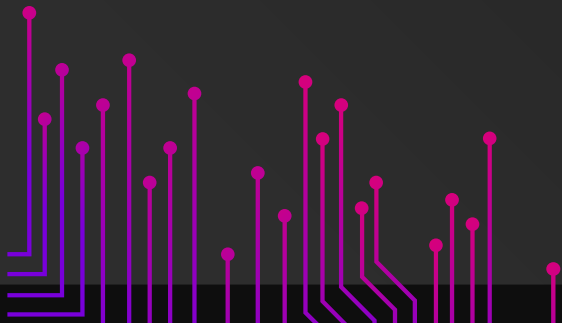
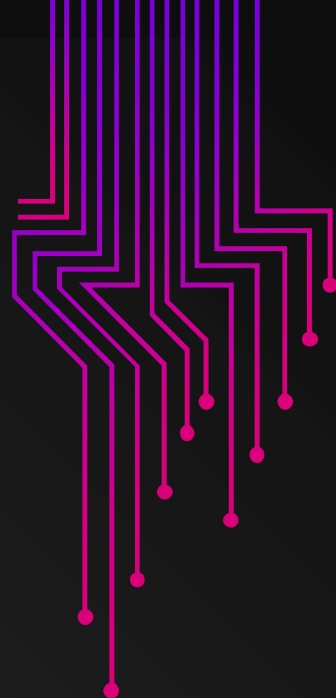


Projeto Cubetto-Uesc

Código Produzido no Ic anterior – *Comentado o passo a passo do código*



Inicialização dos pinos que serão utilizados

```
#include <SoftwareSerial.h>
```

```
SoftwareSerial BTSerial(10, 11); // RX do BT | TX do BT
```

```
// Inicia os pinos digitais 10(RX do BT) e 11(TX do BT) como as portas que irão se comunicar com o bluetooth.
```

```
const int IN1 = 4; // pino digital 4 do arduino
```

```
const int IN2 = 5; // pino digital 5 do arduino
```

```
const int IN3 = 6; // pino digital 6 do arduino
```

```
const int IN4 = 7; // pino digital 7 do arduino
```

```
int Count = 0; // inicia variável count
```

```
int Cmd[16] = {0}; // vetor declarado com 16 posições e iniciado com 0
```

```
/*
```

As variáveis IN são utilizadas para identificar os pinos digitais do Arduino, recebendo respectivamente os valores dos pinos utilizados, definindo qual pino digital será utilizado como uma saída, por meio da função pinMode(), para controlar o motor.

```
*/
```

Transmissão com o modulo Bluetooth


```
void setup()
{
  Serial.begin(4800); // o método begin inicia a porta serial, com uma taxa de transmissão de 4800 baud(o baud é uma medida de velocidade de
  sinalização que representa o número de mudanças na linha de transmissão, ou eventos por segundo), estabelecendo a comunicação entre o
  arduino e computador por meio da porta serial
  BTSerial.begin(9600); // Inicia a comunicação serial com uma taxa de transmissão de 9600 baud, configura a comunicação com o módulo
  Bluetooth conectado aos pinos RX e TX do Arduino(O BTSerial é iniciado com a biblioteca SoftwareSerial, que cria uma porta serial em
  qualquer par de pinos). O que permite a variável BTSerial enviar e receber dados através da conexão Bluetooth
  pinMode(IN1, OUTPUT); // configura como pino de saída, permitindo que eles controlem motores ou dispositivos conectados
  pinMode(IN2, OUTPUT); // configura como pino de saída, permitindo que eles controlem motores ou dispositivos conectados
  pinMode(IN3, OUTPUT); // configura como pino de saída, permitindo que eles controlem motores ou dispositivos conectados
  pinMode(IN4, OUTPUT); // configura como pino de saída, permitindo que eles controlem motores ou dispositivos conectados
}
```

Funções de movimento




```
void Right()
{
    // Gira o Motor A no sentido anti-horario
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    // Gira o Motor B no sentido horario
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);


    /*
    A sintaxe básica da função digitalWrite() é:
    digitalWrite(pino, valor);
    Onde "pin" é o número do pino digital a ser controlado e "value" é o valor a ser escrito no pino, que pode ser HIGH ou LOW.
    */
    delay(260);
    // Para o motor A
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, HIGH);
    // Para o motor B
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, HIGH);
    // Delay para teste
    delay(1000);
}
```




```
void Left()
{
    // Gira o Motor A no sentido horario
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    // Gira o Motor B no sentido anti-horario
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, HIGH);
    delay(260);
    // Para o motor A
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, HIGH);
    // Para o motor B
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, HIGH);
    // Delay para teste
    delay(1000);
}


void Forward()
{
```






```
void Forward()
{
    // Gira o Motor A no sentido horario
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    // Gira o Motor B no sentido horario
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    delay(500);
    // Para o motor A
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, HIGH);
    // Para o motor B
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, HIGH);
    // Delay para teste
    delay(1000);
}
```

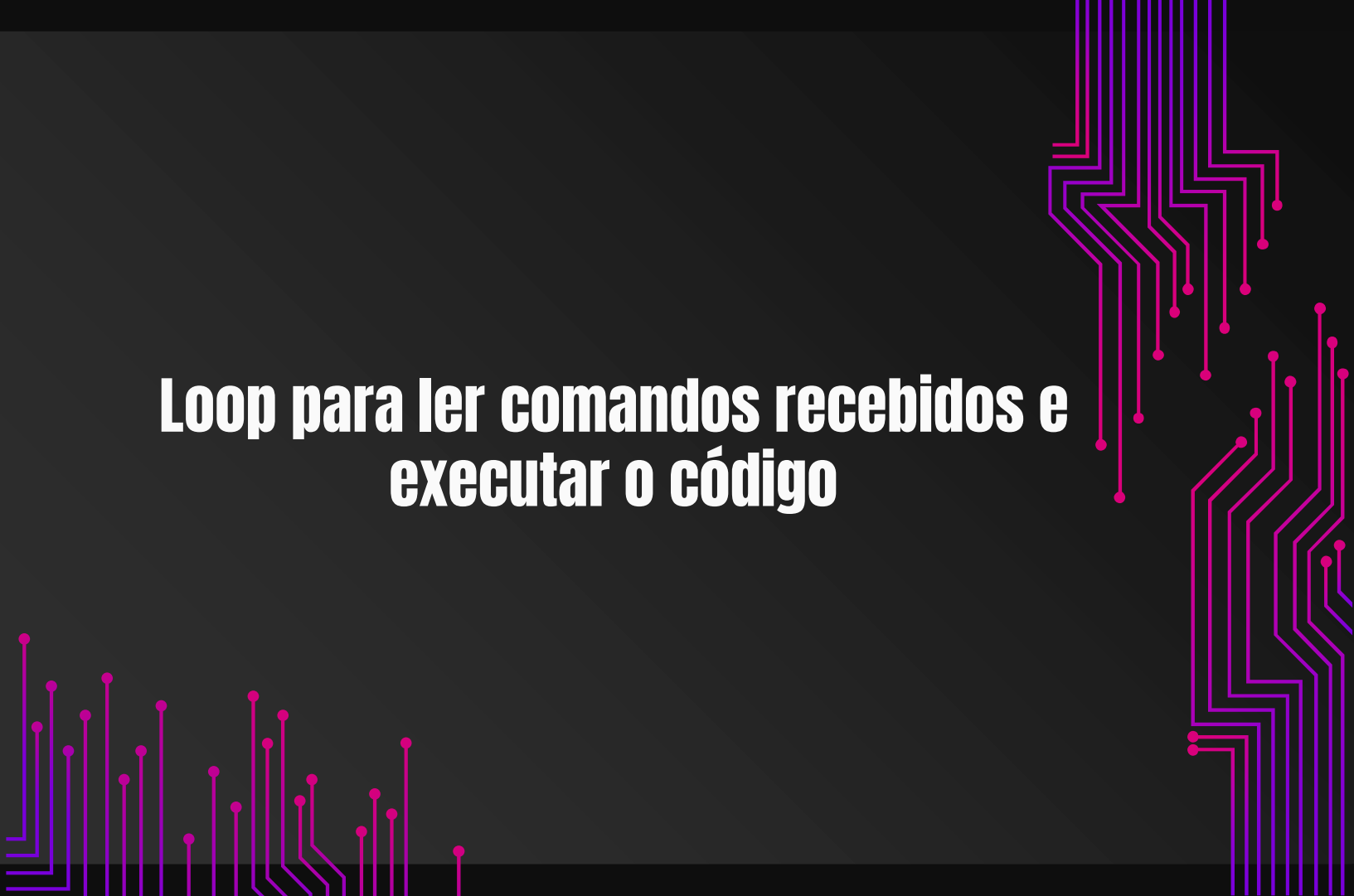




```
void Function()
{
    for (int i = 12; i < 16; i++) // o loop itera os primeiros 12 elementos do vetor
    {
        if (Cmd[i] == 1)           // caso a posição do cmd for 1 , chama a função forward (move o robo pra frente)
            Forward();
        else if (Cmd[i] == 2) // caso a posição do cmd for 2, chama a função left (move o robo pra esquerda)
            Left();
        else if (Cmd[i] == 3) // caso a posição do cmd for 3, chama a função right (move o robo pra direita)
            Right();
        else if (Cmd[i] == 4) // caso o comando for igual a 4, chama a função function
            Function();
    }
    // Delay para teste
    delay(1000);
}
```



**Loop para ler comandos recebidos e
executar o código**



```
void loop() // A função loop() é executada repetidamente no Arduino
{
  Serial.println("nada"); // retorna "nada" pela porta serial. Indica que nada está associado no momento
  if (BTSerial.available()) // verifica se há dados disponíveis para leitura na porta serial do módulo Bluetooth.
  {
    Cmd[Count] = BTSerial.read(); // caso entre na condição do if, o metodo read() vai ler um byte da porta serial do módulo
    Bluetooth e armazena na posição count do vetor cmd[]. O comando recebido é armazenado no vetor para futuro processamento
    Serial.println("LEU -----"); // retorna se o comando foi lido
    Count++; // incrementa o count
    if (Count == 16) // Se o count é igual a 16, indica que todos os comandos foram consumidos e entra na
    condicional if
    {
      for (int i = 0; i < 12; i++) // o loop itera os primeiros 12 elementos do vetor
      {
        // sequência de if, e if else
        if (Cmd[i] == 1) // caso a posição do cmd for 1 , chama a função forward (move o robo pra frente)
        {
          Forward();
        }
        else if (Cmd[i] == 2) // caso a posição do cmd for 2, chama a função left (move o robo pra esquerda)
        {
          Left();
        }
        else if (Cmd[i] == 3) // caso a posição do cmd for 3, chama a função right (move o robo pra direita)
        {
          Right();
        }
        else if (Cmd[i] == 4) // caso o comando for igual a 4, chama a função function
        {
          Function();
        }
      }
      Count = 0; // após o laço for, o contador é reiniciado, recebendo o valor 0, preparando para receber novos comandos no próximo loop()
    }
  }
} // Quando todos os comandos foram processados, o loop() continua sendo executado repetidamente, aguardando novos comandos sendo recebidos e
executando as ações correspondentes.
```