1. Aula 18

Nesta aula, entenderemos parte de como funciona a programação do nosso carrinho robô. Já falamos sobre estas questões de programação na aula 7, mas é um assunto que vale a pena aprender mais.

Aprendemos o que são algoritmos, aqui vamos entender outros conceitos.

A programação da placa Arduino para criação de robôs é feita em uma linguagem de programação chamada C++ com pequenas alterações.

O C++ é uma linguagem de programação de nível médio, baseada na linguagem C. O desenvolvimento da linguagem começou na década de 80, por Bjarne Stroustrup. O objetivo do desenvolvimento desta linguagem era melhorar uma versão do núcleo Unix. Para desenvolvê-la, foram acrescentados elementos de outras linguagens de vários níveis, na tentativa de criar uma linguagem com elementos novos, sem trazer problemas para a programação.

O C++ tem uma enorme variedade de códigos, pois, além de seus códigos, pode contar com vários da linguagem C. Esta variedade possibilita a programação em alto e baixo nível. O C++ apresenta grande flexibilidade e, embora seja bom, este fato faz com que a programação seja muito mais cuidadosa para não acontecerem erros.

Alguns fatos sobre o C++:

O C++ é uma linguagem criada para ser tão eficiente quanto o C, porém com novas funções.

É uma linguagem que suporta múltiplos paradigmas.

A linguagem da liberdade para o programador escolher as opções, mesmo sendo a opção errada.

Muitos códigos podem ser transferidos para C facilmente, pois o C++ foi criado para ter compatibilidade com o C.

A linguagem não tem privilégios para alguns grupos de programadores, os comandos são feitos para todas as especialidades de programadores

Não é necessário um ambiente de desenvolvimento muito potente para o desenvolvimento de C++.

Alguns dos mais conhecidos programas são feitos em C++ ou parte dos seus códigos são nessa linguagem. Alguns deles são: Adobe Photoshop, MySQL, Mozilla Firefox, Internet Explorer, Microsoft Windows, entre vários outros.

Apenas para o nosso conhecimento saber que existe a linguagem “C++” e que podemos estar aprendendo sobre ela.

Agora vamos conhecer a estrutura do código.

```
int AIA = 9;  // (pwm) pino 9 conectado ao pino A-IA do Módulo
int AIB = 8;  // (pwm) pino 8 conectado ao pino A-IB do Módulo
int BIA = 7; // (pwm) pino 7 conectado ao pino B-IA do Módulo
int BIB = 6;  // (pwm) pino 6 conectado ao pino B-IB do Módulo
```

A palavra reservada “**int**” quer dizer que a nossa variável será do tipo inteiro, isso quer dizer que não vai aceitar valores do tipo 5,5 que possui casa decimal.

A sigla ao lado do “**int**”, em cada uma destas linhas, é o nome da variável, sendo seguida por um sinal de igualdade, que está atribuindo um valor.

Esses valores representam as conexões dos pinos, mas poderíamos criar a seguinte variável em outro exemplo.

Int idade = 12;

Neste exemplo, a nossa variável recebe a idade igual a 12 anos.

Na sequência, temos o símbolo “**//**”, as duas barras são comentários dentro da programação, é uma forma de escrita que não é lida e interpretada pelo sistema na hora de rodar o código. Tem a função de lembrar o que foi feito e de poder se organizar.

Nas quatro primeiras variáveis que criamos, programamos os pinos 9, 8, 7 e 6 da placa Arduino para serem os pinos que serão conectados ao módulo controlador dos motores.

```
byte frente = 255;
byte voltar = 200;
byte speed1 = 255;
byte speed2 = 255;// Mude este valor (0-255) para controlar a velocidade dos motores
byte speedv1 = 255;
byte speedv2 = 255;
```

A palavra reservada “**byte**” quer dizer que esta variável terá obrigatoriamente um valor entre “0” e “255”.

Logo em seguida, temos o nome da variável, o sinal de igualdade e os valores onde zero(0) representa nada e 255 o máximo. As variáveis citadas na imagem acima controlam a velocidade dos motores e também sua função “avançar e voltar”. Alterando esses valores, podemos alterar a velocidade que nosso carrinho irá se locomover ou mudar de direção.

Para finalizarmos, sabemos que os pinos analógicos da placa Arduino (9, 8, 7 e 6) serão utilizados para o controle de motores. Sabemos também que devemos ligar os pinos 9 e 8 a um módulo e os pinos 7 e 6 a outro. Os pinos dos sensores são ligados às portas digitais 11 e 12. Também temos os valores de velocidade que serão utilizados pelo robô para ajustar sua posição e acelerar os motores.

Estrutura completa.

```
  int AIA = 9;  // (pwm) pino 9 conectado ao pino A-IA do Módulo
  int AIB = 8;  // (pwm) pino 8 conectado ao pino A-IB do Módulo
  int BIA = 7; // (pwm) pino 7 conectado ao pino B-IA do Módulo
  int BIB = 6;  // (pwm) pino 6 conectado ao pino B-IB do Módulo
byte frente = 255;
byte voltar = 200;
byte speed1 = 255;
byte speed2 = 255;// Mude este valor (0-255) para controlar a velocidade dos motores
byte speedv1 = 255;
byte speedv2 = 255;

byte sensor2 = 12;//Pino que deve se conectar a saída (out) do sensor
byte sensor1 = 11; //Pino que deve se conectar a saída (out) do sensor

void setup() {
  pinMode(AIA, OUTPUT); // Colocando os pinos como saída
  pinMode(AIB, OUTPUT);
  pinMode(BIA, OUTPUT);
  pinMode(BIB, OUTPUT);
}

void loop() {

 if(!digitalRead(sensor2))
    {
 speed2=frente;
        speedv2=0;
    }else{
       speed2=0;
        speedv2=voltar;
      }

        if(!digitalRead(sensor1))
    {

speed1=frente;
speedv1=0;
    }else{
```