

Experimento de reação química alinhado à tecnologia: junção de áreas no currículo do ensino fundamental

Manual de Eletrônica

Caroline Passolongo Coelho - Unesp/FC - caroline.passolongo@unesp.br
Eliza Dionisio Moreno - Unesp/FC - eliza.moreno@unesp.br
Giovanna da Silva Pedro - Unesp/FC - g.pedro@unesp.br
Kleper de Oliveira Rocha - Unesp/FC - kleper.rocha@unesp.br
Mariana Rocha Côrrea - Unesp/FC - mr.correa@unesp.br
Pedro Augusto Otero Sabino - Unesp/FEB - pedro.sabino@unesp.br

Conteúdo

Parte Teórica

Recomendada para quem não tem conhecimento prévio na área e se interessa ou tem dúvidas com relação aos conceitos relacionados com o dispositivo utilizado.

Introdução

Conceitos Básicos de Programação

Parte Prática

Explicação do procedimento e realização dos códigos com análise dos resultados.

Configuração

Montagem

Programação

Análise dos Dados

Introdução

O que é Arduino?

Arduino é uma plataforma de eletrônicos e softwares abertos para o público. Comumente, quando se ouve falar sobre uso de Arduino, refere-se ao uso dos microcontroladores feitos sobre essa plataforma, por exemplo um Arduino Uno.

Ser aberto para o público significa que todo indivíduo tem acesso tanto ao código da interface de desenvolvimento integrado Arduino (IDE), que vamos falar mais a frente, quanto aos esquemáticos dos circuitos componentes de um microcontrolador dessa família. Sendo assim, todos podem utilizar dos esquemáticos e montar sua própria placa em casa, sem correr riscos de infringir direitos autorais. Esse é um dos motivos para o baixo custo desses microcontroladores.

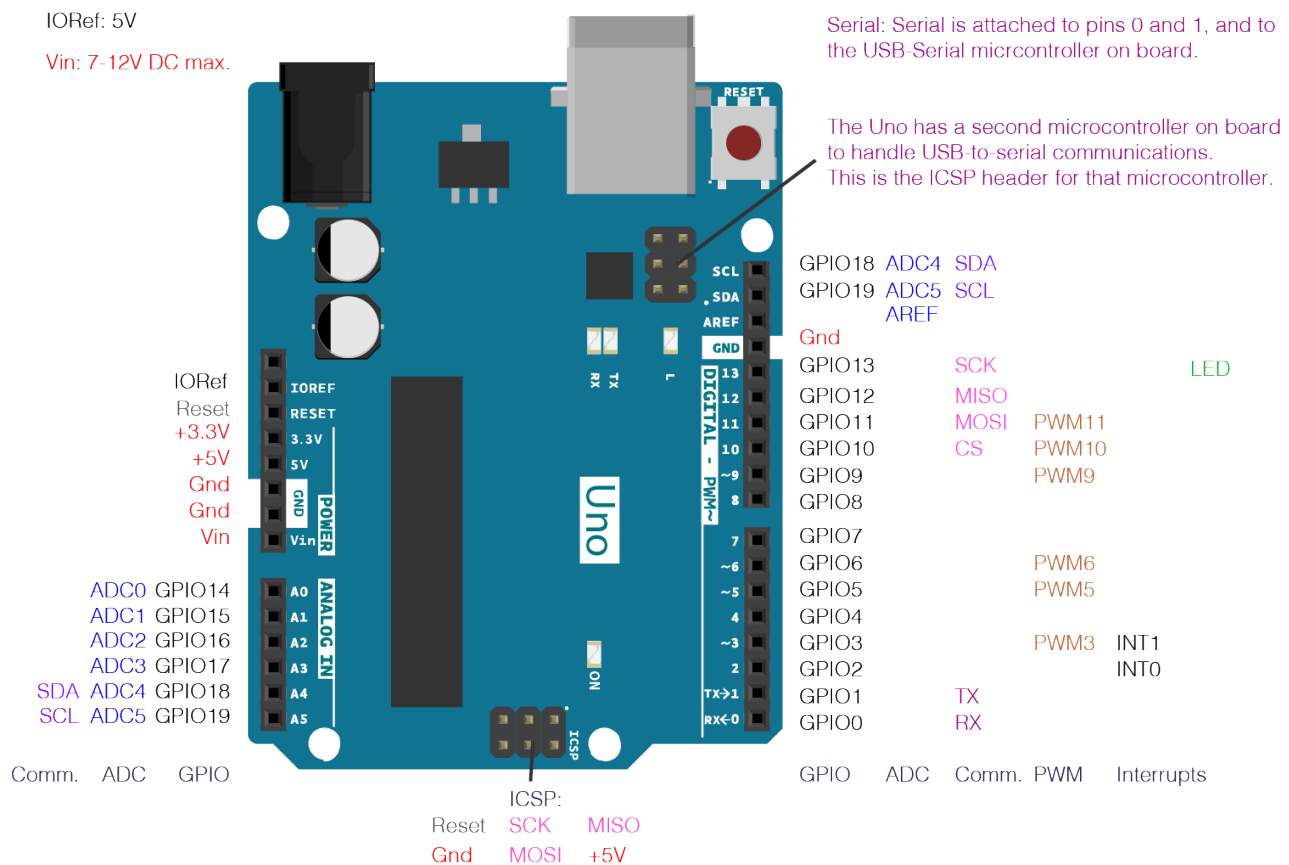
O que é um microcontrolador?

Um microcontrolador é o casamento de uma unidade de processamento (CPU), memórias (RAM e ROM) e dispositivos de entrada e saída (I/O) em um mesmo chip. Microcontroladores são utilizados principalmente em sistemas de eletrônica embarcada, que são os casos de um processador controlar um circuito de maneira autônoma.

Todos os microcontroladores Arduino são iguais?

Sim e não. Existem vários modelos de microcontrolador dentro dessa mesma família Arduino e todos utilizam da mesma programação, interface e pinos. A diferença está nas capacidades de cada modelo. No geral, todas as placas tem os mesmos pinos: Digitais, Analógicos, Alimentação e Comunicação Serial. Porém, alguns modelos tem mais ou menos pinos de cada tipo. Além disso, a unidade de processamento e memórias podem mudar de um modelo para outro, isso significa que: existem placas que são capazes de tarefas mais complexas e tem maior espaço para seu código, enquanto outras são mais limitadas com relação ao código e complexidade.

Para o caso do experimento explicado por esse manual, qualquer modelo disponível é suficiente.



O que são pinos Analógicos/Digitais? Como identifica-los?

Estes pinos são para ler sinais que levam o mesmo nome. De maneira simplificada: sinais digitais são sinais com dois estados, situação chamada de binária (0 ou 1), enquanto sinais analógicos assumem qualquer valor real dentro da faixa de medição. Por exemplo: a lógica transistor-transistor (TTL) é para sinais digitais de 0 a 5V, ou seja, a possibilidade 0 corresponde a 0V e a possibilidade 1 corresponde a 5V. Nessa mesma faixa, um sinal analógico seria por exemplo 2,5V, que não se encaixa nem em 0 ou em 1. A tensão ideal para se utilizar nos pinos do Arduino é entre 3,3 e 5V.

Você pode identificar se um sinal é analógico ou digital na placa observando o nome que vem na sua frente. Se o nome do pino for A2, ele é analógico, se for somente 2, ele é digital.

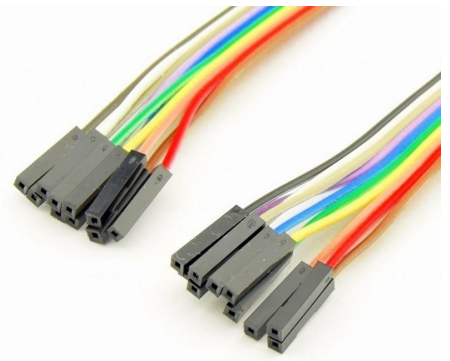
Como eu utilizo esses pinos?

Você programa eles pelo código e então liga eles por meio de cabos chamados *jumpers*. Existem duas terminações para esses cabos, que geram três variações: macho e fêmea; gerando macho-macho, macho-fêmea e fêmea-fêmea. Eles são representados pelas figuras a seguir:



Essa é uma ponta macho. Perceba que ela tem um pedaço exposto de metal, que é utilizado para dar contato com uma entrada.

Essa é uma ponta fêmea. Perceba que ao invés de uma ponteira de metal há uma entrada, que é onde a ponteira do cabo macho vai.



Como as ponteiros podem se encaixar, um cabo pode ser estendido utilizando de outros com ponteiros opostas. Então por exemplo, se você tiver um cabo macho-macho que precisa ser estendido, pode usar um cabo macho-fêmea para fazê-lo.

Para fazer conexões com as placas Arduino, você vai precisar de uma ponta do tipo macho, pois os pinos são do tipo fêmea.

Preciso só encaixar os pinos?

Basicamente sim. Porém, a utilização de cabos, ainda mais com conexões móveis, pode gerar mal contato entre os fios. A ocorrência de mal contato é muito comum e você deve estar atento a isso.

Para minimizar essa possibilidade, encaixe os pinos por completo e evite deixar partes de metal expostas.

Nunca faça conexões com o circuito alimentado e evite tocar o máximo possível nas extremidades de metal tanto da placa quanto dos pinos.

Como eu ligo o Arduino?

Você precisa de uma fonte de energia para alimentar o circuito. Existem dois jeitos de se fazer isso, ou pela conexão do cabo USB que depende de um computador, que vai ser explicada no próximo tópico, ou pela entrada de alimentação. Essa entrada de alimentação pode ser feita por uma bateria com tensão entre 7 e 12V para máxima segurança ou com uma fonte que pode ser conectada à uma tomada comum.

Para garantir a preservação da placa, é importante fazer a alimentação nos parâmetros impostos pelo fabricante. Sendo assim, nossa recomendação é usar ou um computador ou uma fonte externa própria para Arduino.

Conceitos Básicos de Programação

Programar é como dar ordens para as máquinas, mas a linguagem que cada máquina fala pode variar de uma para outra, assim como o idioma pode mudar de um país para o outro. Um dos problemas de fazer essa comunicação é que as máquinas usam um “idioma” muito diferente do que estamos acostumados e que requer muito esforço para ser decorado. Essa é a linguagem de máquinas: Assembly. Para facilitar a vida do programador, é muito comum serem utilizados passos intermediários entre a pessoa e a máquina, como se fosse um tradutor. Isso permite que o programador converse com a máquina utilizando uma linguagem de mais alto nível, fácil de compreender, enquanto o tradutor compila as informações e passa para a difícil linguagem de máquinas.

Os microcontroladores da família Arduino utilizam, de maneira geral, a linguagem de alto nível de nome C++, que é basicamente C com alguns adicionais.

Existem alguns conceitos básicos de programação que não se resumem apenas a essa linguagem, mas são recorrentes na maioria delas. Essas são os assuntos a serem tratados nesse tópico.

Variáveis

Essas são pedaços de informações. Da matemática, conhecemos variáveis como x ou y . Aqui é quase a mesma coisa. O que acontece é que essas variáveis podem ser vários tipos de informação. Abaixo há exemplos para alguns casos que podem ser utilizados num Arduino.

```
int x = 10;
```

```
float y = 2.5;
```

```
String z = “banana”;
```

x , y e z são as variáveis. Perceba que elas tem um termo que vem na frente delas, que muda de uma para a outra. Esse termo define o tipo de variável que ela é. O tipo “int” se refere a valores inteiros, como 10. O tipo “float” se refere a valores quebrados, como 2,5. E o tipo “String” se refere a palavras e frases, como banana.

ØBS: Quando precisar colocar um valor que não seja inteiro, lembre que devem ser utilizados pontos no lugar de vírgula. Assim: 2,5 deve ser 2.5. Note também que toda linha termina com um ponto e vírgula, é importante não se esquecer desse detalhe, pois esse carácter indica o fim de um comando.

Essas variáveis podem ser utilizadas no seu código para transmitir informações ou somente armazená-la.

Uma vez que você tenha definido ela, não há necessidade de representar seu tipo novamente. Então por exemplo:

```
int x = 10;
```

```
x = x + 1;
```

O que está acontecendo acima é: foi definido x como um inteiro que vale 10. Na linha de baixo, está sendo dito que x é igual a x + 1, ou seja, 11. Note que x só foi definido uma vez.

Funções

Funções são um conjunto de comandos que funcionam juntos, e que tem um nome atribuído. Abaixo está um exemplo:

```
int soma(x, y){  
    int valor = x + y;  
    return valor;  
}
```

Essa é uma função bem simples que faz o que o nome diz: soma. Perceba que assim como as variáveis, uma função também tem tipo definido, no caso inteiro. Esse tipo está relacionado com o que a função retorna para quem a chamou, no caso ela retorna um valor inteiro.

Na frente da definição está seu nome, soma. Na frente do nome está um parênteses e dentro as variáveis x e y. Essas variáveis dentro do parênteses são chamadas de argumentos de uma função. São as variáveis que você envia para a função na hora de chamar ela.

Dentro da função, está a variável valor, que é igual à soma de x com y.

Em seguida está o comando para retornar o valor resultante.

Então para utilizar essa função, você deve chamá-la, enviando os valores que desejar. Exemplo:

```
int soma = soma(2, 3);
```

O que acontecerá então? Depois de executar essa função, a variável soma valerá 5, que é a soma de 2 com 3.

Perceba que a função soma foi definida como inteira. Se você entendeu o processo, sabe que ela deve retornar um valor inteiro. Isso significa que essa função soma apenas funciona com argumentos inteiros. Ou seja, não consegue somar 2,5 com 3. Apenas valores inteiros.

Um código básico de Arduino tem duas funções obrigatórias: setup e loop. As duas funções são definidas como do tipo void, ou seja, não retornam nada.

Dentro da função setup, você faz as definições essenciais do programa, como pinos e variáveis. Essa função é executada apenas uma vez, quando o programa é carregado na placa.

Já a função loop é uma que se repete todo ciclo da placa, exatamente como o nome sugere. Dentro dela você quer colocar as coisas que vão se repetir várias vezes num programa.

A IDE do Arduino traz consigo algumas funções básicas, relacionadas com a utilização da placa e que são muito importantes de serem conhecidas.

Como usar um pino

Para usar um pino, você precisa dizer para a placa que quer utilizar ele e como pretende fazer isso. Para tal, usamos a função `pinMode(pino, tipo)`. Como dito anteriormente, o que vai dentro dos parênteses são os argumentos dessa função, aquilo que ela precisa para funcionar. Essa função precisa que você diga duas coisas para ela: qual pino você quer e como quer usá-lo. Com pino, entenda o nome do pino, por exemplo: você quer usar um pino analógico, então use o pino A0; A0 é o nome do pino. E o tipo é se ele é de entrada ou saída. Assim, se você quer utilizar o pino analógico A0 como entrada, vai fazer assim: `pinMode(A0, INPUT)`. Perceba que para entrada usamos INPUT (entrada em inglês) e para saída OUTPUT (saída em inglês).



Configuração

Essa etapa é para quem nunca utilizou um dispositivo Arduino em seu computador. Portanto, se você já o fez, pode pular.

Ambiente de Desenvolvimento Integrado (IDE)

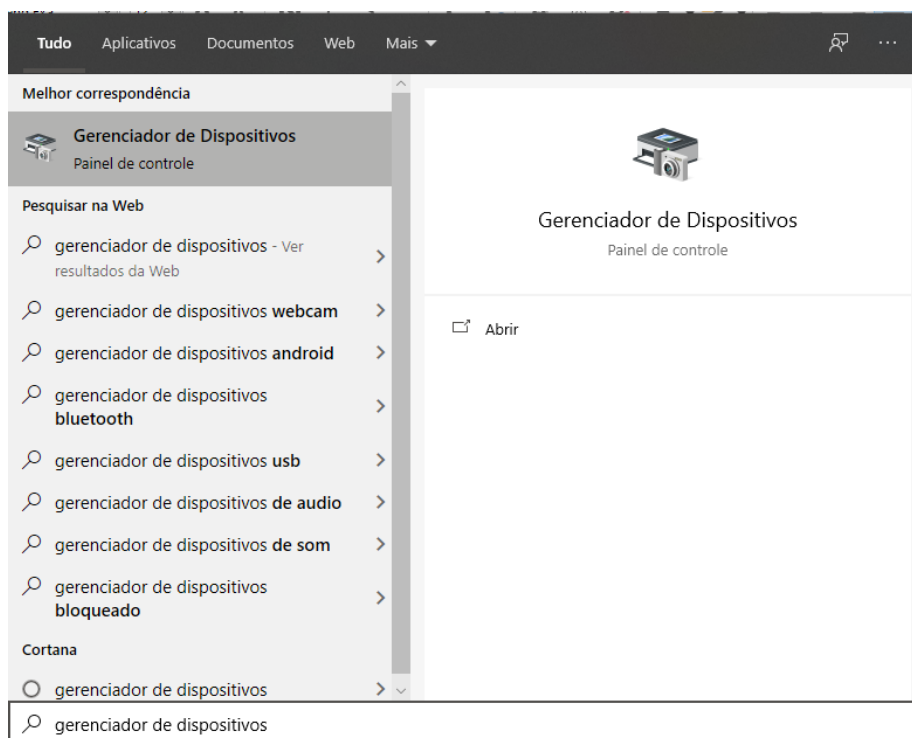
Esse é o nome dado ao programa de computador onde são feitos os códigos que colocamos nas placas Arduino. Você pode obtê-lo diretamente do site oficial da marca: <https://www.arduino.cc/en/software>. Como dito antes, esse é um software aberto e, portanto, gratuito para todos.

Depois de ter instalado, você terá o programa Arduino em seu computador.

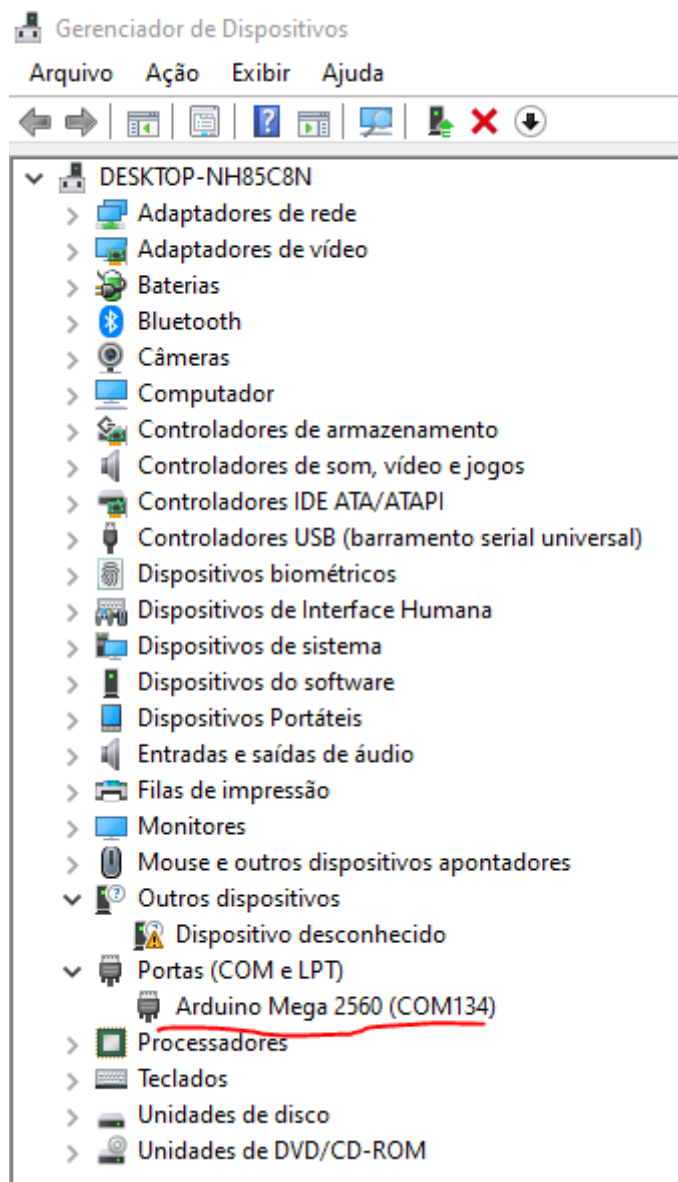
Conectando a placa pela primeira vez

Ligue a placa em seu computador pelo cabo USB que veio com ela. Como é a primeira vez, o sistema operacional provavelmente tentará instalar os drivers do dispositivo de maneira automática. Caso isso aconteça, você deverá esperar esse processo falhar para continuar.

Vá no seu menu de aplicativos e digite **Gerenciador de Dispositivos**, como mostrado abaixo:



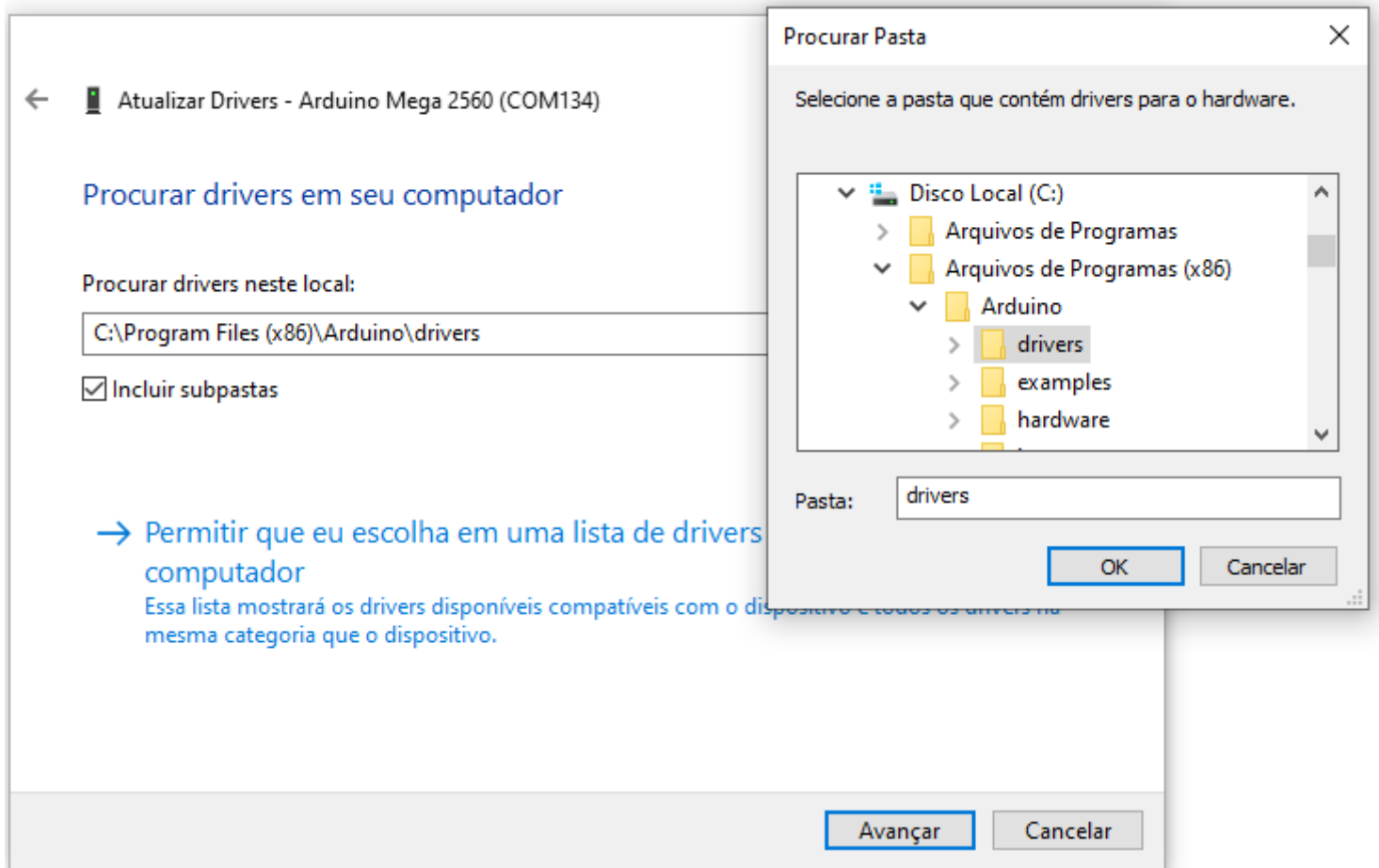
Entrando nesse painel, você verá todos os dispositivos conectados ao seu computador. Procure pela área **Portas (COM e LPT)**. Ao clicar, verá o nome da placa que conectou ao computador. Clique com o botão direito e vá em Atualizar Driver, como abaixo:



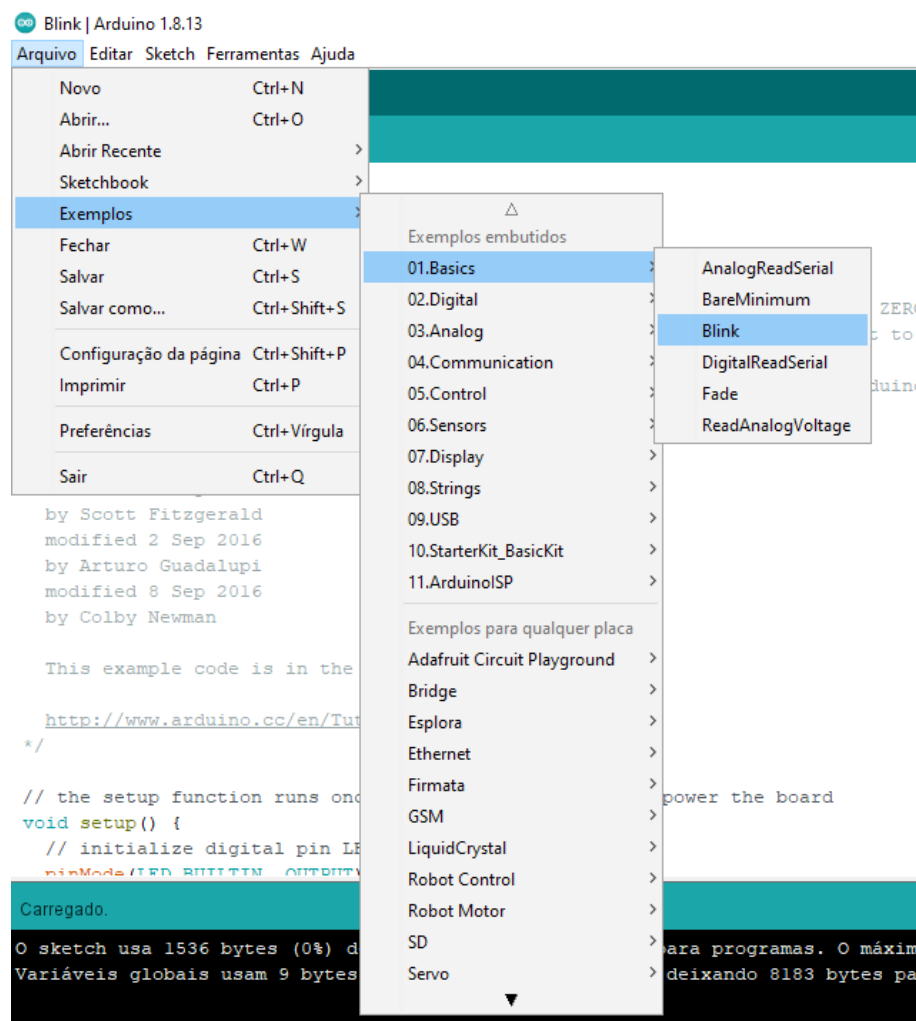
Vão aparecer duas opções: Pesquisa Automática ou Procurar no Computador. Clique em Procurar no Computador.

Agora você precisa colocar selecionar a pasta **drivers** dentro da pasta Arduino no seu disco local. O caminho varia de computador para computador. O nosso, por exemplo, é esse: **Disco Local > Arquivos de Programas (x86) > Arduino > drivers**. Uma foto mostrando o caminho se encontra na próxima página.

Tendo selecionado essa pasta, basta clicar em Avançar e aguardar o computador terminar a instalação.



Agora com os drivers instalados, abra a IDE Arduino e carregue o exemplo Blink. Como abaixo:



Uma nova janela abrirá, com várias coisas escritas. Esses textos são um código.

Basicamente, o microcontrolador precisa ser ensinado o que deve fazer e, para ensiná-lo, nós o programamos com uma linguagem de código.

Os textos escritos com um “//” na frente representam comentários do autor do código para o usuário. Ou seja, eles não interferem no programa em si.

Os texto que não vem como comentário são os comandos sendo dados para o processador. São como ordens.

Você verá uma linha escrito: **digitalWrite(LED_BUILTIN, HIGH);**. Essa linha é uma ordem que pode ser interpretada como: escreva um sinal digital no led que vem da placa, esse sinal deve ser alto (como foi dito, sinais digitais são 0 ou 1, assim, falar alto significa falar 1). Quando o processador recebe esse comando, ele escreve no pino do led o sinal que você pediu. Ao escrever 1 (alto), significa acender o led, e 0 (baixo) desligar o led.

Entre as linhas de acender e apagar, verá uma linha: **delay(1000);**. Essa linha é interpretada como: Espere 1000 milissegundos (1 segundo). Ao receber esse comando, o processador espera o tempo que você mandou.

De uma maneira bem simplificada, programar é então dar comandos às máquinas.

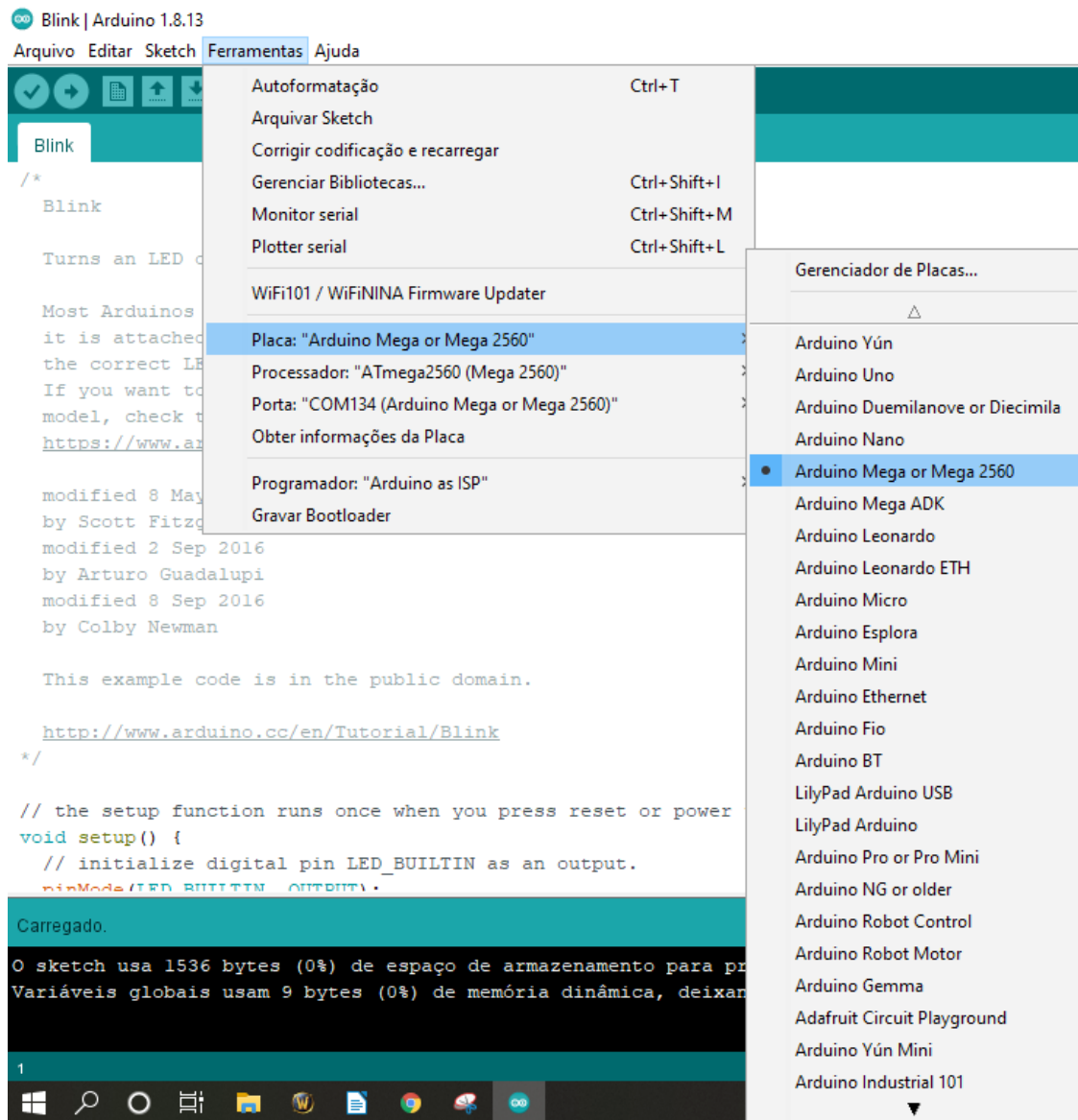
No canto superior esquerdo da sua tela haverão dois botões. Como na figura abaixo:



Clicando no que está a esquerda (V), você pede para a IDE corrigir seu código e ver se não há nenhum erro de sintaxe, mais ou menos como corrigir sintaxe de frases na língua portuguesa, mas agora na linguagem de programação.

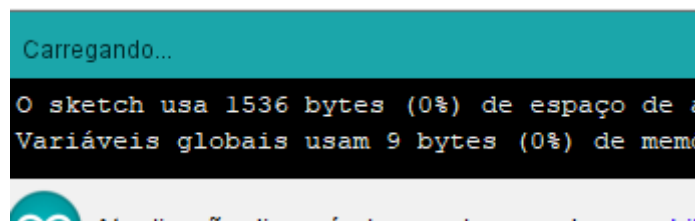
Clicando no que está a direita (setinha), você manda o programa para o Arduino e ele executa.

Agora você precisa dizer para a IDE qual modelo de placa você está usando. Para fazer isso, siga a figura abaixo:



Aí estarão todos os modelos existentes, você clica no que corresponder à sua placa. Para o caso do Arduino Mega, é necessário também especificar o Processador dela. Na figura acima, você pode conferir como deve ficar.

Se você clicar na setinha, para mandar o código, verá um texto na parte inferior esquerda da tela com carregando.

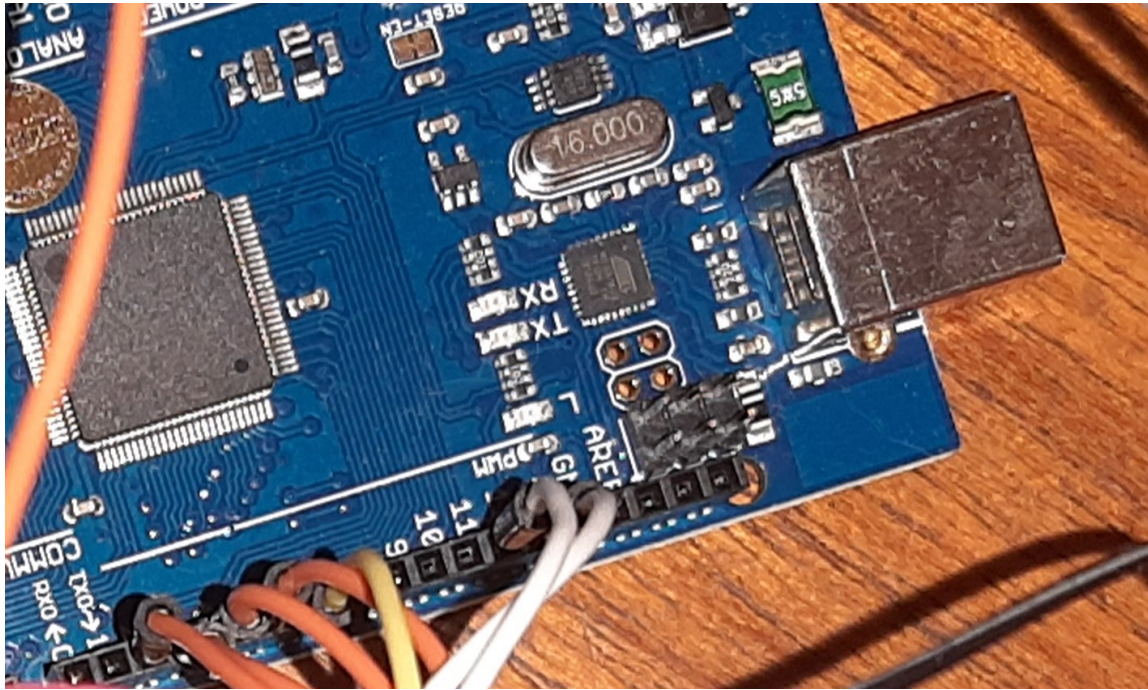


Isso significa que a mensagem está sendo enviada.

Depois que a mensagem for enviada por completo, no lugar de Carregando estará Carregado. Isso significa que o programa já está rodando no microcontrolador.

Esse programa que ensinamos a colocar agora é um programa de teste, para demonstrar os conceitos básicos, não o programa real que será usado no experimento. O procedimento para o experimento estará mais abaixo.

Para conferir se o programa de teste funcionou corretamente, procure pelo led sobre a placa que carrega a letra L ao lado.



Se ele estiver piscando em intervalos regulares de 1 segundo, tudo deu certo e você já pode seguir para o próximo tópico.

Caso o led não esteja piscando, é provável que a configuração de modelo ou processador da placa está errada. Nesse caso, tente repetir os passos descritos anteriormente.

Montagem

Aqui nós oferecemos duas opções para o leitor: Utilizar um display de LCD para fazer as medições ou fazer elas por meio de um computador, que obrigatoriamente deverá estar conectado à placa durante o experimento.

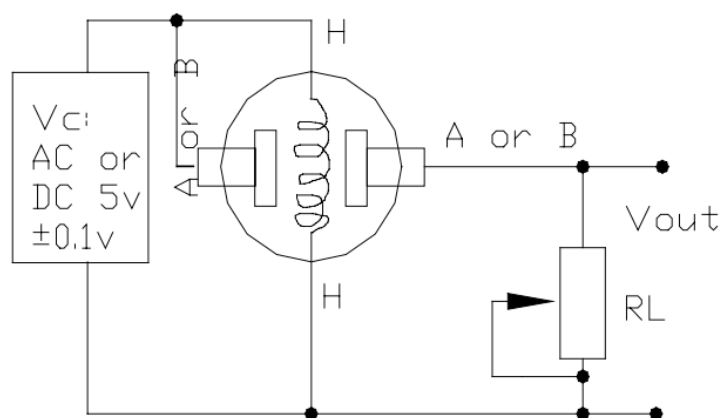
Os materiais básicos são:

- 1 - Microcontrolador da família Arduino (Uno, Mega etc);
- 1 - Sensor de gás carbônico, CO₂, podendo ser o Mq-135 ou MQ-2;
- 3* - Jumper macho-fêmea.

Caso decida utilizar um display, precisará de mais:

- 1 - Display LCD 16x2 (com backlight, independe da cor);
- 1 - Protoboard de qualquer tamanho (as menores chamam Breadboard);
- 14* - Jumper macho-macho;
- 1** - Fonte de alimentação para Arduino.

Independente do método que tenha escolhido, o sensor é o mesmo, e ele precisa passar por um processo de inicialização antes de podermos utilizá-lo. Abaixo está uma representação do circuito interno desse sensor.



* A quantidade de jumpers depende muito do comprimento deles.

** A fonte de alimentação é para que não seja necessário alimentar a placa pelo computador. Se não for possível usar uma, aconselhamos não usar o display, já que será necessário o computador de qualquer jeito.

No centro da figura está um círculo. Esse círculo representa tanto o aquecedor do sensor quanto a resistência que varia com a concentração do gás. No processo de fabricação, essa resistência é mantida o mais puro possível, para evitar falhas indevidas. Porém, nós precisamos que ela faça justamente o contrário, absorva impurezas do ar. Sendo assim, precisamos deixar o sensor ligado por um certo período, em que ela vai absorver essas impurezas. Esse processo é chamado de *burn-in*. O fabricante especifica que esse tempo deve ser de, no mínimo, 24h corridas.

Como fazer o burn-in

Você sobre precisa alimentar a placa Arduino, seja com o cabo USB ou pela fonte de alimentação, e ligar o sensor.

Os pinos do sensor são como na figura abaixo. Note que alguns modelos são diferentes, mas os pinos são sempre os mesmos.



Vcc - Alimentação positiva.

Gnd - Alimentação negativa.

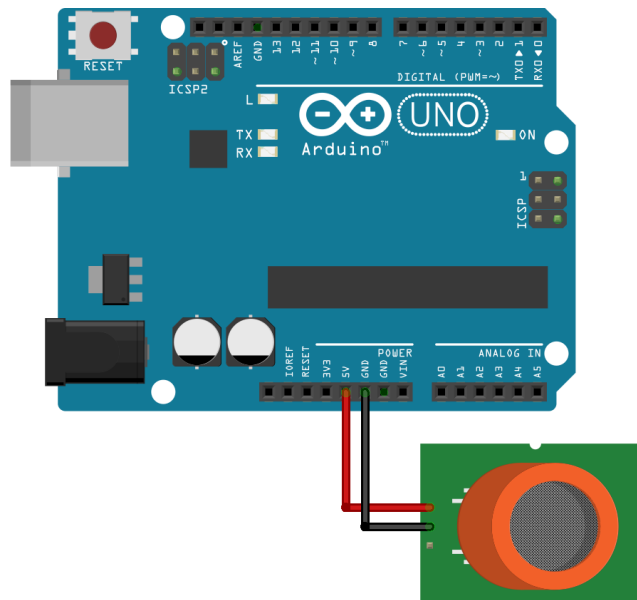
A0 - Sinal analógico de saída.

D0 - Sinal digital de saída.

Para ligar o sensor então, sem se preocupar o que ele marca, basta ligar os pinos de alimentação. Para isso:

- I) Ligue o pino Vcc do sensor no pino 5V do Arduino.
- II) Ligue o pino Gnd do sensor no pino Gnd do Arduino.

Essas ligações são representadas como abaixo:



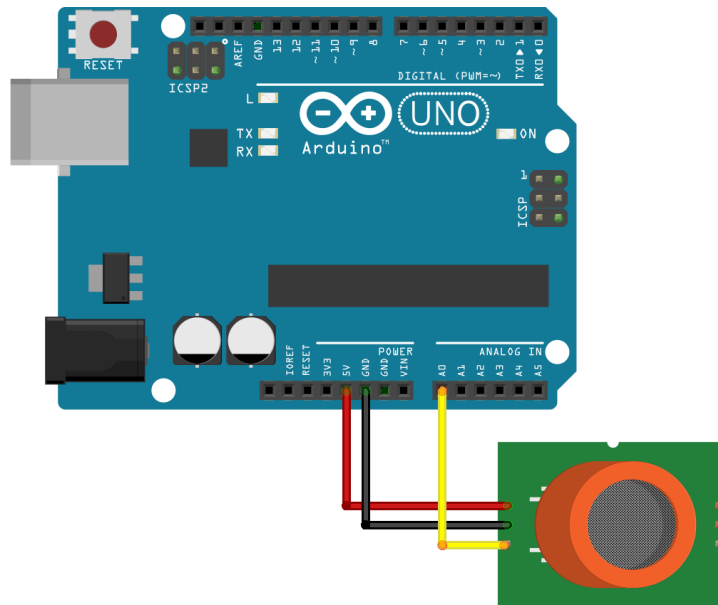
Depois de ter ligado tudo, alimente o circuito. Como dito, é necessário deixar essa ligação feita por pelo menos 24h corridas.

Montagem sem display

Para a montagem sem usar o display, você deve seguir os passos:

- I) Ligue o pino Vcc do sensor no pino 5V do Arduino;
- II) Ligue o pino Gnd do sensor no pino Gnd do Arduino;
- II) Ligue o pino A0 do sensor no pino A0 do Arduino.

Tendo feitas essas ligações, você terá algo como a figura abaixo:



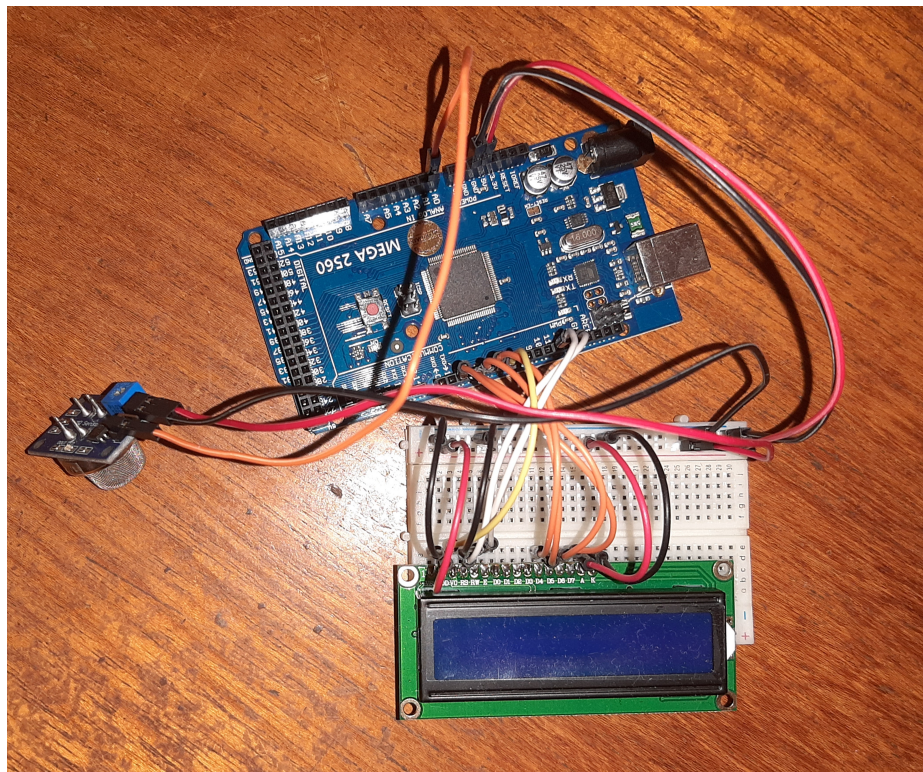
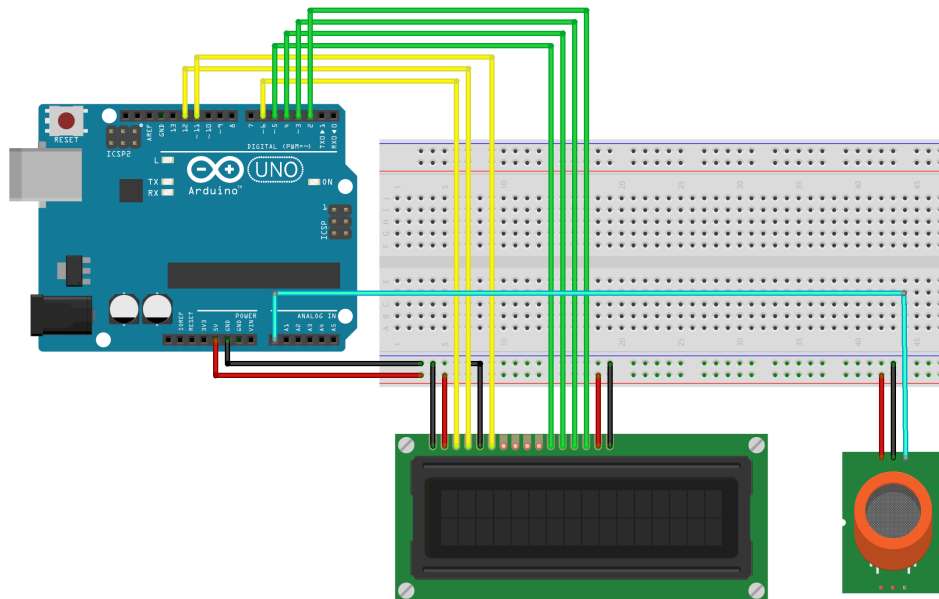
Com isso, você já pode passar para a área de programação.

Montagem com display

Para a montagem usando o display, você deve seguir os passos:

- I) Ligue o pino 5V do Arduino na trilha com + da protoboard;
- II) Ligue o pino Gnd do Arduino na trilha com - da protoboard;
- II) Ligue o pino Vcc do sensor na trilha + da protoboard;
- III) Ligue o pino VSS do display na trilha - da protoboard;
- IV) Ligue o pino VDD do display na trilha + da protoboard;
- V) Ligue o pino V0 do display no pino 6 do Arduino;
- VI) Ligue o pino RS do display no pino 12 do Arduino;
- VII) Ligue o pino RW do display na trilha - da protoboard;
- VIII) Ligue o pino E do display no pino 11 do Arduino;
- IX) Ligue o pino D4 do display no pino 5 do Arduino;
- X) Ligue o pino D5 do display no pino 4 do Arduino;
- XI) Ligue o pino D6 do display no pino 3 do Arduino;
- XII) Ligue o pino D7 do display no pino 2 do Arduino;
- XIII) Ligue o pino A (15) do display na trilha + da protoboard;
- XIV) Ligue o pino K (16) do display na trilha - da protoboard;
- XV) Ligue o pino Gnd do sensor na trilha - da protoboard;
- XVI) Ligue o pino A0 do sensor no pino A0 do Arduino.

Seguindo estes passos, você terminará com algo similar ao representado pelo esquemático abaixo ou a imagem de base:



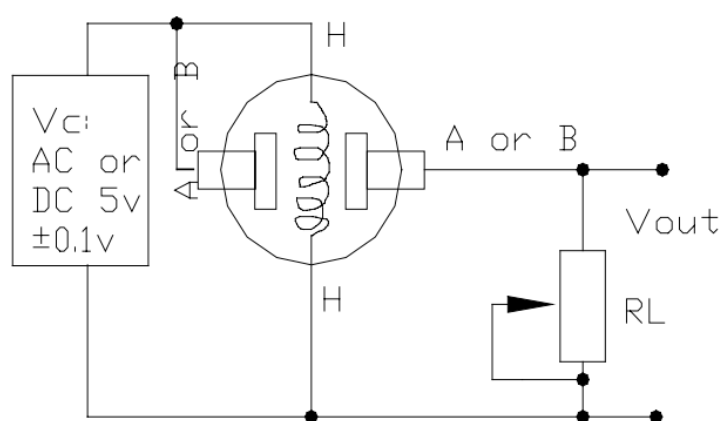
Com isso feito, você já pode passar para a área de programação.

Programação

Para seguir os passos daqui para a frente, é importante que você tenha passado pelo tópico de **Configuração** primeiro.

O código que você deverá utilizar depende de como você escolheu montar o circuito. Atente-se ao que você escolheu.

O que ambos os códigos fazem é basicamente pegar o valor fornecido pelo sensor no pino A0. Esse valor é a tensão sobre o resistor de carga R_L da figura abaixo:



Pego este valor, nós convertemos ele para o equivalente real, uma vez que o Arduino converte o valor analógico para um equivalente digital de 10 bits. Um sinal digital de 10 bits pode valer entre 0 e $2^{10} - 1$ (1023). Isso quer dizer que, se o sinal analógico valer 2,5V, o sinal que o Arduino armazena nele é 612, metade do valor máximo. Para obter o valor analógico, nós fazemos a conversão teórica do sinal.

Com o valor real da tensão, nós calculamos o valor do resistor do sensor, aquele que varia conforme a concentração de poluentes no ar. Analisando o valor desse sensor, nós podemos observar variações na concentração de CO_2 .

Código sem display

Se você escolheu fazer o experimento sem o display, copie e cole o código a seguir na IDE do Arduino.

O texto que estiver após os “//” é um comentário e não tem função real no programa além de explicar o que cada linha faz.

Código para se usar sem display:

```
#define RI 1000 //resistência interna do sensor (esse valor vem de fábrica)

void setup(){
  Serial.begin(9600); //inicia a comunicação serial com o computador

  pinMode(A0, INPUT); //pino A2 é definido como entrada analógica, no caso o valor do sensor
}

void loop(){
  float Vo = analogRead(A0) * 5; //cálculo da tensão sobre o resistor
  Vo = Vo / 1024;

  float Rs = (5 - Vo) * RI; //valor da resistência Rs

  Serial.println("Resistencia: " + String(round(Rs))); //mostra o valor medido
}
```

Código com display

Se você escolheu fazer o experimento com o display, copie e cole o código a seguir na IDE do Arduino.

O texto que estiver após os “//” é um comentário e não tem função real no programa além de explicar o que cada linha faz.

```
#include "LiquidCrystal.h" //biblioteca do display

#define RI 1000 //resistência interna do sensor

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //conexões que fizemos no display

void setup(){
  lcd.begin(16, 2); //tamanho do display, que no caso é 16x2

  pinMode(6, OUTPUT); //pino 6 é definido como saída
  analogWrite(6, 90); //pino 6 é uma saída pwm com tamanho 90

  pinMode(A0, INPUT); //pino A0 é definido como entrada analógica
}

void loop(){
  float Vo = analogRead(A2) * 5; //cálculo da tensão sobre o resistor
  Vo = Vo / 1023;

  float Rs = (5 - Vo) * RI; //valor da resistência Rs

  lcd.setCursor(0,0); //posição inicial para escrever
  lcd.print("Resistencia"); //escreve "Resistencia"(linha de cima)

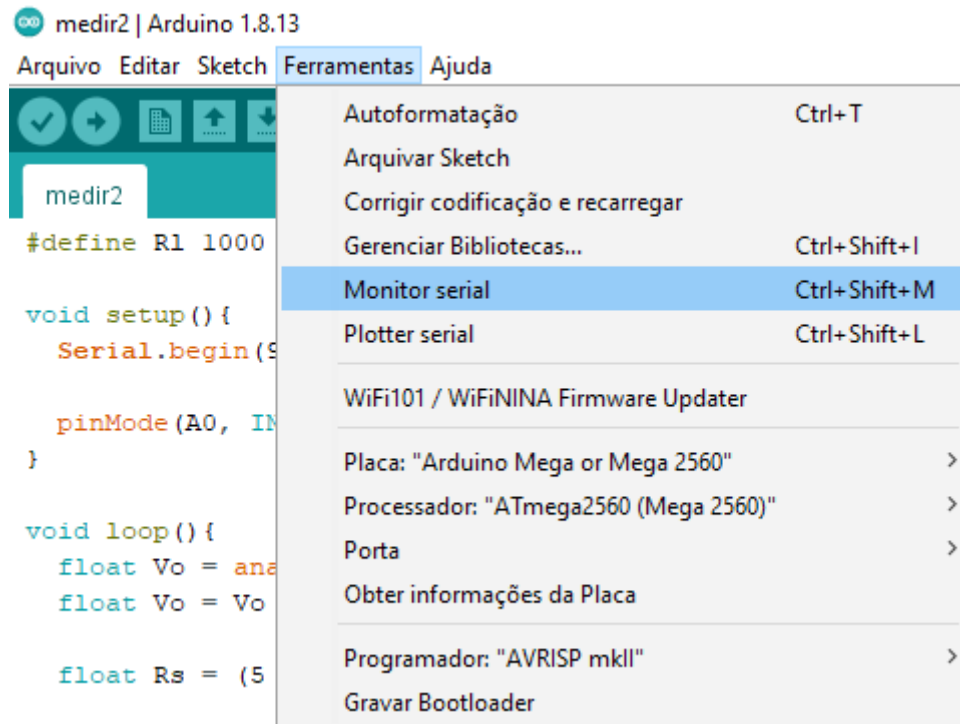
  lcd.setCursor(0, 1); //pula a posição para uma linha abaixo
  lcd.println(Rs); //escreve o valor de Rs (linha de baixo)
}
```

Em seguida, independente de qual modelo tiver escolhido, certifique-se que o circuito está montado corretamente, ligue a placa no computador com o cabo USB e carregue o programa.

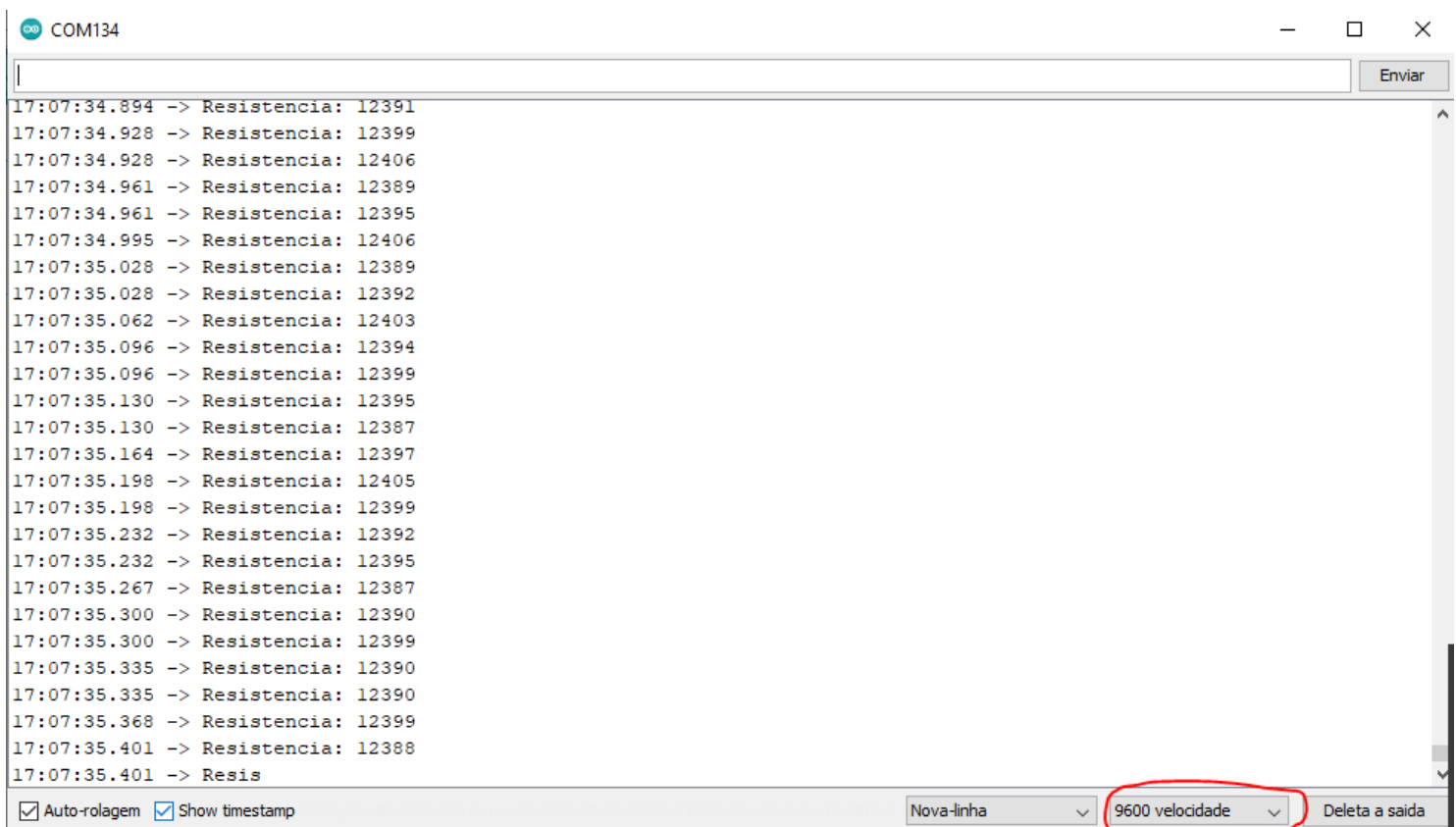
Se a montagem estiver certa, você deverá obter algo de acordo com o que está abaixo. O valor medido pode ser diferente, isso depende da concentração de poluentes no local em que for inicializado o sensor.

Sem display

Para conferir seu resultado, clique em: Ferramentas > Monitor Serial, como abaixo:



No canto inferior direito da janela, altere a Velocidade para 9600. Você obterá algo como abaixo:



Com display

Utilizando o display, mesmo sem ter carregado o código o backlight se acenderá ao alimentar a placa (provavelmente na cor azul). Com o código carregado, aparecerá nele textos assim:



Mostrando na linha de baixo o valor da resistência medida do sensor.

Caso você obtenha textos estranhos, piscando ou letras aleatórias, retire a alimentação da placa e confira a montagem do circuito.

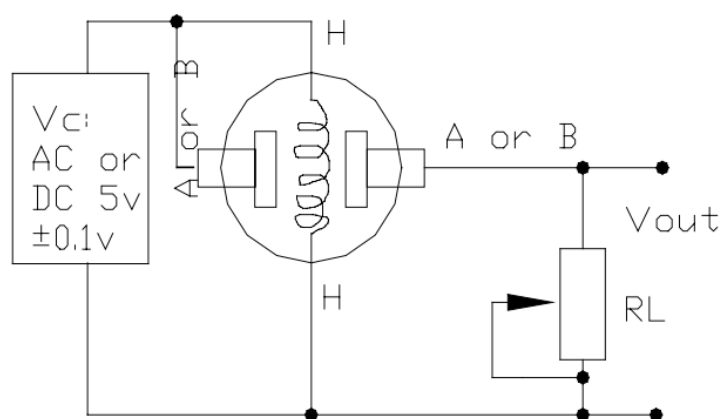
Tendo conferido e estando tudo correto, ligue novamente. Se o problema persistir, você provavelmente está com algum mal contato entre o display e a placa.

A ligação dos pinos D4 até D7 do display é especialmente sensível a mal contatos e requer atenção redobrada.

Caso o texto apresente falhas após um período funcionando corretamente, você pode tentar utilizar o botão de **Reset** manual presente na placa. Normalmente é um botão de cor vermelha. Clique apenas uma vez sobre ele e solte rapidamente.

Análise dos Dados

Como foi dito anteriormente, o sensor é composto de duas “resistência” ligadas em série. Uma dessas resistências é física e tem o valor fixado pelo fabricante, R_L . A outra resistência, R_S , varia de acordo com a concentração de poluentes presentes no ar, é a resistência do sensor.



A tensão de saída que o sensor fornece para o controlador é medida sobre R_L e, como dito, nós trabalhamos cálculos sobre esses valores conhecidos de R_L e V_{out} para encontrar R_S .

As variações na concentração de poluentes são observadas então com a variação de R_S . Uma variação positiva, R_S aumentar, significa diminuição na concentração de poluentes. Uma variação negativa, R_S diminuir, significa aumento na concentração de poluentes.

Assim, para fazer um teste rápido da sua montagem, perturbe o sensor com algum tipo de poluente, por exemplo acender um isqueiro perto da capsula prateada. Você deve observar uma diminuição de R_S .

Montagem do Gráfico

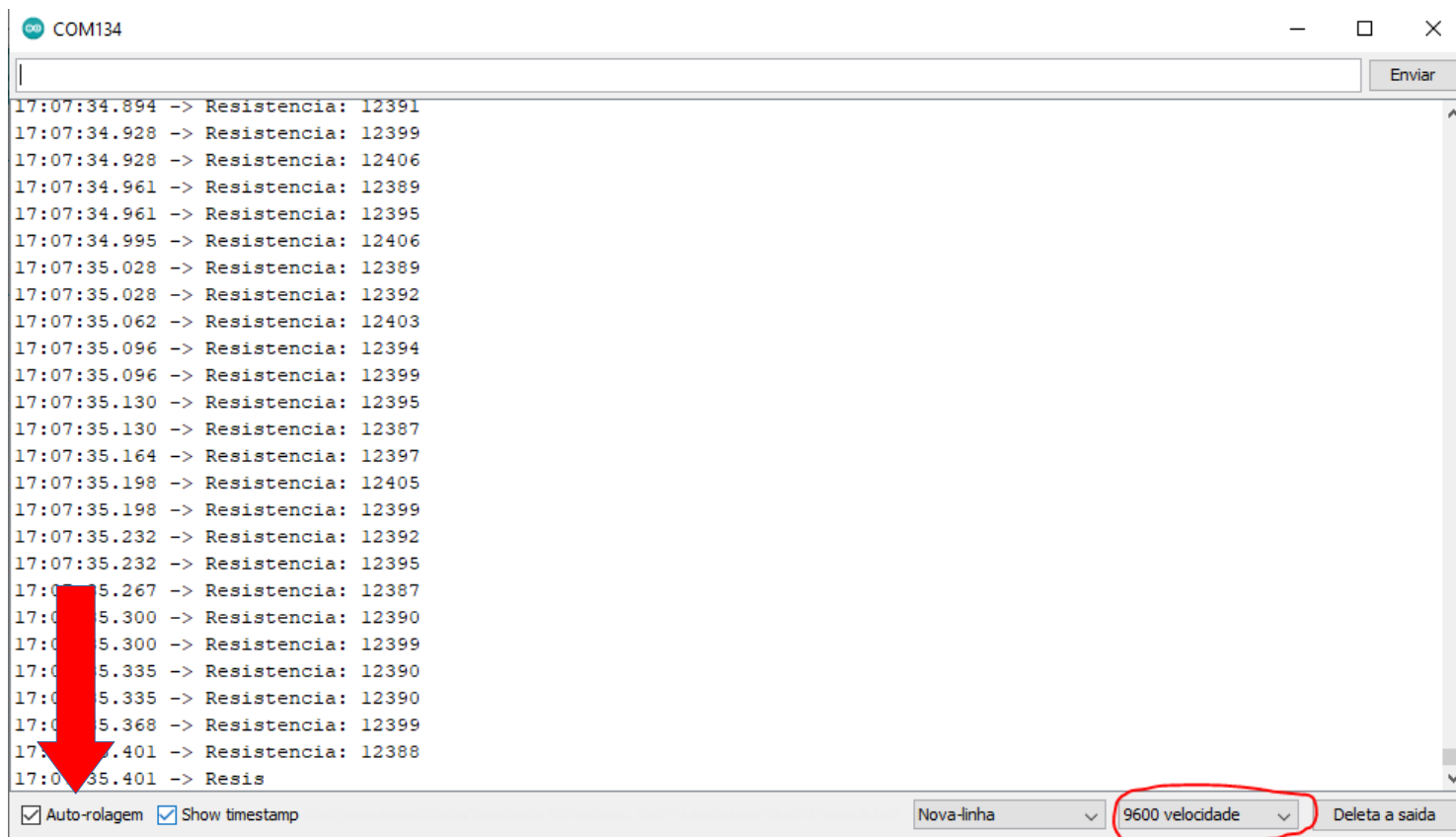
O experimento feito com o uso do display LCD requer que os valores mostrados, juntamente com o instante em que cada ponto aconteceu, sejam anotados manualmente e o gráfico seja construído com base nesses pontos.

Caso o experimento seja feito usando o monitor serial, você pode fazer o gráfico manualmente também, mas é possível utilizar os dados gerados e montar um gráfico em algum software externo (Excel e etc.).

Para tal, siga os seguintes passos:

Realize a montagem do experimento e execute o programa normalmente, observando o monitor serial.

Quando estiver satisfeito com o tempo de aquisição dos dados, desmarcar a caixa de “Auto-rolagem” na tela do monitor.

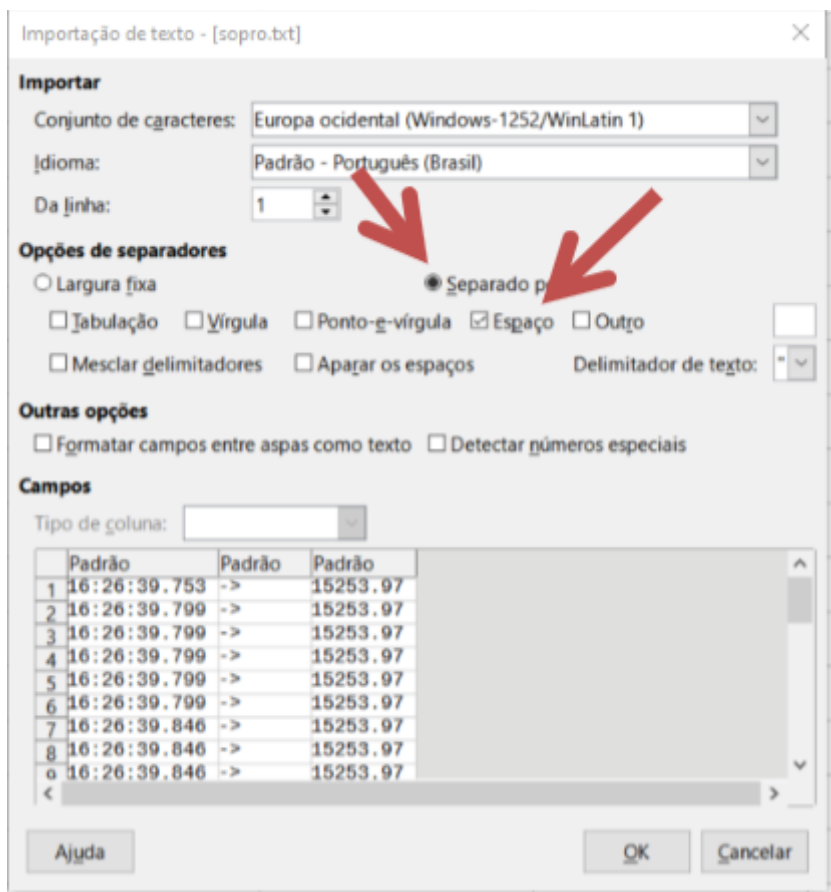


Em seguida, posicionar o cursor ao final valor na última linha da tela e clicar uma vez com o mouse. Observe que a tela para de rolar para baixo. Em seguida, pressione (no teclado) as teclas “Ctrl+Shift+Home” e todos os valores vão ficar marcados de azul. Retire a seleção da primeira linha pressionando as teclas “Shift+↓”. Copie os dados pressionando as teclas “Ctrl+C”.

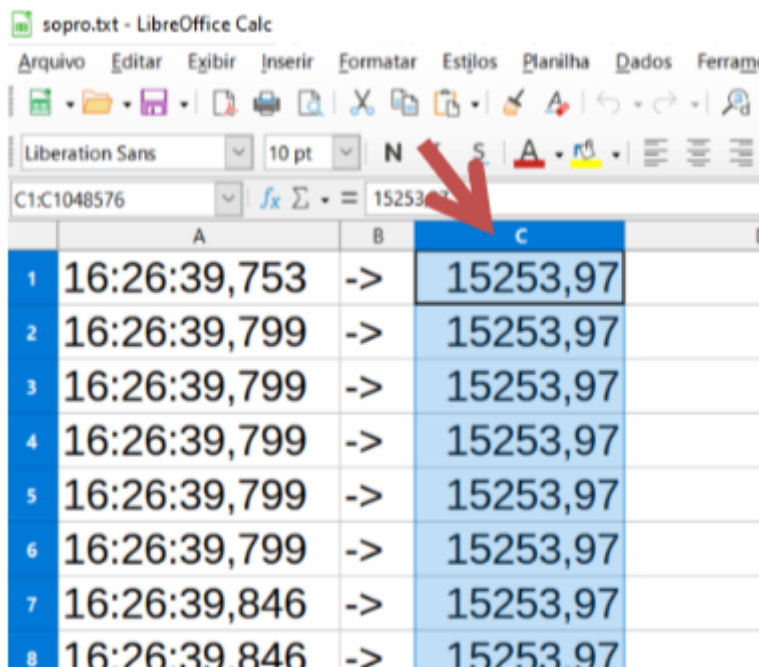
Abra um editor de textos (preferencialmente o Bloco de Notas) e cole os dados copiados anteriormente. Se tiver dado certo, você verá que os dados vistos no monitor agora estão nesse arquivo do bloco de notas. Salve esse arquivo em algum lugar do computador.

Agora abra o programa que tiver mais facilidade para gerar gráfico, comumente o Excel ou o Calc.

Vá em Arquivo→ Abrir e abra o arquivo. Como é um arquivo de texto sendo aberto em uma planilha, você vai precisar definir o limitador “Espaço” para os dados abrirem corretamente.

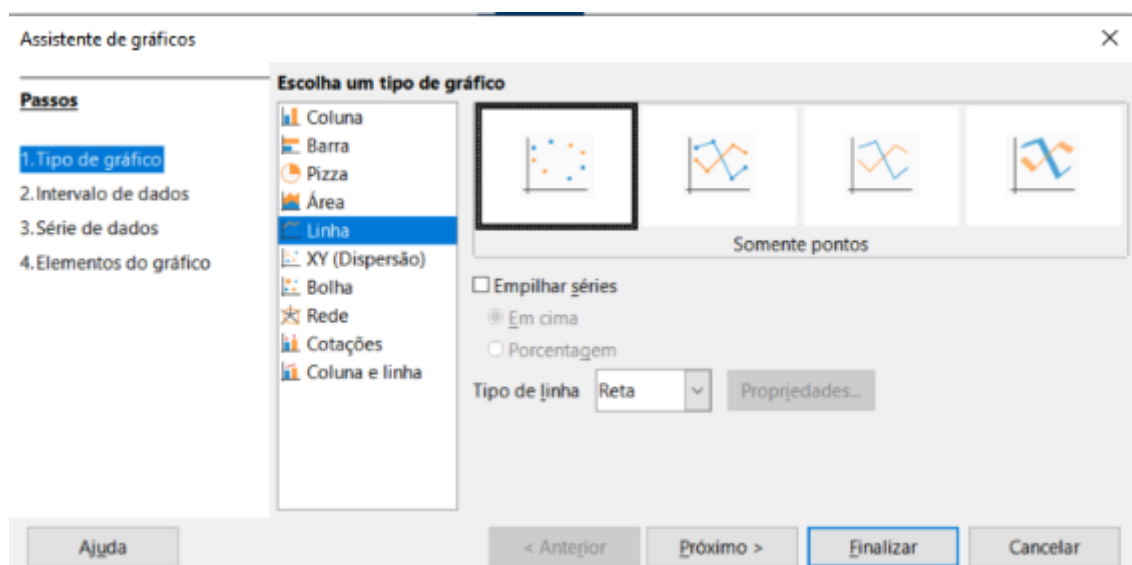


Clique em “OK”. Em seguida, selecione a coluna C clicando sobre ela.



Para fazer o gráfico vá em “Inserir→Gráfico”. Na janela que se abre, selecione “Linha” e clique em “Finalizar”.

Como mostrado na figura abaixo:



Tendo feito isso, o gráfico aparecerá na planilha. Você pode clicar sobre ele e arrastá-lo para onde achar mais conveniente.

Imagens Utilizadas

As imagens que não forem referenciadas são de autoria nossa.

1) Pinos Arduino:

<https://www.pngwing.com/pt/free-png-taajl>

2) Jumper macho:

<https://proesi.com.br/jumper-macho-x-macho-kit-com-10-unidades-20cm.html>

3) Jumper fêmea:

<https://www.recicomp.com.br/produtos/jumper-femea-femea-x10-unidades-20cm/>

4) Circuito Equivalente:

<https://www.olimex.com/Products/Components/Sensors/Gas/SNS-MQ135/resources/SNS-MQ135.pdf>

5) Sensor MQ-135:

<https://www.botnroll.com/pt/biometricos/2195-sensor-de-gases-mq-135.html>