



02



Inatel

CURSO EAD



Introdução

E aí desbravador, o que você está achando? Conseguindo fazer as atividades propostas? Muito bem. Continue assim. Quanto mais exercícios você fizer, mas claro as informações ficarão em sua mente. Lembre-se sempre, a prática é o sistema que leva a perfeição. Quanto mais você praticar, quanto mais exercícios você fizer, melhor você se sairá na competição.

Mas já que estamos por aqui, que tal continuarmos com o conteúdo? É uma boa não é mesmo?

Para prosseguirmos então deixe-me lhe apresentar um novo componente que será bastante usado no Arduino Challenge. Estou falando com você sobre o sensor de refletância. Re... o quê? Isso mesmo refletância.

Relaxa, o seu processo de funcionamento é bem mais simples que o seu nome. Você por acaso já viu algum desse? Não? Então, preste bem atenção na ilustração a seguir.



Ilustração 1 - Sensor de refletância

Mas para que ele serve e como ele funciona? Ótimas perguntas desbravador. Vamos então às suas respostas:

Em suma, o objetivo de usarmos esse sensor é para perceber se a superfície que ele se encontra é clara ou escura. Por exemplo, ele pode ser usado em um carrinho seguidor de linhas e com isso detectar o trajeto correto a ser seguido.

O seu funcionamento é bem simples: esse sensor contém um led infravermelho e um fototransistor, ambos localizados na parte dianteira da placa. Assim que o led infravermelho enviar raios para uma superfície, esta pode refletir esses raios com uma intensidade de grau específica, variando com o tipo de superfície que estamos lidando.

Esses raios são recebidos de volta pelo fototransistor, e o arduino é o responsável por interpretar o valor recebido e fazer a identificação de qual superfície estamos tratando. Um exemplo disso pode ser visto na Ilustração 2.

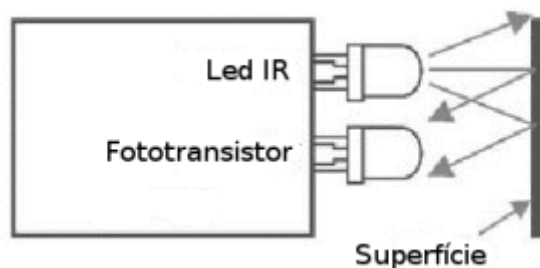


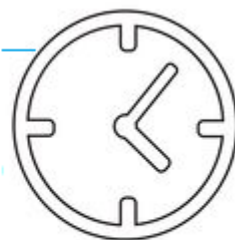
Ilustração 2 - Funcionamento do sensor de refletância

O modelo de sensor de refletância que estamos usando é o QRE1113. E vamos considerar um seguinte: Todas as vezes que este sensor ler um valor acima de 300, estamos nos referindo a uma superfície escura e sempre que for lido menor que 300, a superfície será clara. Os valores a serem lidos podem variar de 0 a 1023.

Uma vez entendido o funcionamento deste dispositivo e percebendo que sua função principal é informar ao arduino com qual tipo de superfície estamos lidando, quero que você me mostre que ainda não esqueceu o que foi visto na apostila passada e me responda:

Pense Rápido

O sensor de refletância é um dispositivo de entrada ou de saída? Por quê?



Resposta: É um dispositivo de entrada, pois o seu objetivo é fornecer informações ao arduino. Por conta disso ele usa a constante INPUT.

Está bem, entendi que o sensor pode ler um valor de 0 a 1023 e que todas as vezes que essa leitura for maior que 300 significa que estamos tratando uma superfície escura. Também entendi como esse dispositivo faz para ler este valor. Mas como que o arduino faz essa leitura? Porque uma coisa é o sensor ler e outra coisa é o arduino entender. Ou estou errado?

Você está certíssimo caro desbravador. Mas não entre em pânico, eu tenho a solução do seu problema. E o nome dessa solução é `analogRead`.

`analogRead()`: Faz a leitura de um pino analógico, que pode ter sua intensidade variada. A sua estrutura é:

analogRead(pin),

onde pin é a pinagem em que o dispositivo se encontra conectado.

Por exemplo, digamos que queremos ler a intensidade de resposta de um sensor e que este sensor se encontra no pino 3. Ficaríamos assim:


analogRead(3);

Todas as vezes que usarmos esse comando, ele ‘contará’ ao arduino qual o valor que o dispositivo que está conectado ao pino 3 está lendo.

Por exemplo, digamos que neste pino esteja ligado o nosso sensor de refletância...

Pense Rápido

O que é mesmo um sensor de refletância? E no nosso caso ele pode ler qual faixa de valores?



Resposta: É um sensor que tem como objetivo principal definir com qual tipo de superfície estou lidando. No nosso caso se é uma superfície clara ou escura. Ele pode ler uma faixa de valores que varia de 0 a 1023.

...e que depois de um tempo esse sensor capta o valor 500. Em palavras gerais, ele irá dizer ao arduino o seguinte: “Arduino, o valor da superfície aqui está dando 500” e com base nesse dado, o arduino tomará providências, ou seja, provocará determinadas manipulações.

Mas como ele faz isso?

Fique bem claro uma coisa em sua cabeça, desbravador: nós sabemos que um valor acima de 300 equivale a uma superfície escura e que abaixo, ela é clara. Repare no que eu disse, NÓS sabemos disso, o arduino não. Para ele saber essa informação, é necessário contar a ele. E podemos fazer isso de várias maneiras. O Arduino Challenge, neste momento, escolherá fazer isso usando uma estrutura condicional chamada de if.

Fique atento

O processo de ligação do sensor de refletância segue o mesmo procedimento do processo de ligação do led e buzzer visto na apostila anterior. Se ainda não compreendeu, não hesite, volte à apostila anterior e treine um pouco mais. Mas lembre-se sempre, ele tem que ser acoplado aos pinos destinados às entradas analógicas.



Faça a ligação do sensor de refletância ao pino
01. Você tem exatamente 5 minutos para
executar essa tarefa. Mãos à obra!!

Estrutura condicional (If)

Como o próprio nome sugere, o objetivo do if é dar uma condição ao arduino e baseada nesta, tomar as decisões que lhe for mais pertinente.

Deixe-me ser mais claro:

Uma estrutura condicional faz comparações entre informações. Essa comparação é denominada de **condição**. Por conta disso é necessário se ter em mãos os dados a serem conferidos. Por exemplo, digamos que você deseja comparar a sua nota da prova de matemática com a nota do seu amigo, para ver quem se saiu melhor. Para que essa comparação aconteça, é essencial que se conheça a sua pontuação, tal qual a dele. Isso significa, que para essa estrutura acontecer, é fundamental que se tenham dois elementos. Tendo esse dois elementos e feito esse contraste entre ambos, é possível tirar alguma conclusão. Diante dessa, você estabelecerá uma série de ações que lhe sejam convenientes.

Voltemos ao exemplo das notas. Após fazer a comparação, concluiu-se que a sua nota foi superior a nota de seu amigo, e por isso, você decidiu oferecer ajuda para ele estudar. Então ficamos assim: A comparação das notas (**condição**) levará a uma tomada de decisão (**ações**).

Ainda está um pouco confuso? Não se preocupe. Vamos detalhar um pouco mais. Verifique se a estrutura abaixo o ajuda na compreensão.

Temos nesse momento três informações:

- 1 – Minha nota;
- 2 – Nota do meu amigo;
- 3 – Oferecer ajuda;

A “**Minha nota**” e a “**Nota do meu amigo**” são dados. Ou seja, precisam ser coletados de algum lugar se ainda não o foram. **Oferecer ajuda** é uma ação.

A estrutura condicional exige que comparemos esses dois dados, e logo após executemos uma ação. Para fazer isso, use as palavras “Se” e “Então”. Veja como fica:

*Se “Minha nota” for maior que a “Nota do meu amigo”, **então**, eu vou “Oferecer ajuda”.*

OBS: Verificamos até o momento que as palavras “se” e “então” são poderosos aliados para se realizar uma comparação e promover ações. Por conta disso, é muito importante que sempre se faça uso dela.

Ajudou a clarear um pouco mais o que é a estrutura condicional? Pois bem, vamos prosseguir. Antes que possamos levá-la para o código em si, lhe mostrarei uma técnica que o ajudará bastante. Ela é chamada de pseudocódigo e serve para que rascunhemos o nosso programa em um pedaço de papel antes de irmos para o computador. Ele possibilita escrever um determinado programa em uma linguagem extremamente clara para qualquer ser humano.

Mais uma vez, vamos voltar ao exemplo das notas. Se fôssemos colocar o seu enunciado em pseudocódigo, ele ficaria da seguinte forma:

```
Se (MinhaNota>NotaDoMeuAmigo)  
{  
    Oferecer ajuda;  
}
```

Você consegue compreender o que foi escrito acima? Percebeu a entrada de alguns símbolos?

No pseudocódigo que escrevemos, continuamos usando a palavra “se”. A comparação, ou seja, a condição, foi colocada dentro dos parênteses e trocamos a palavra maior por seu respectivo símbolo. Logo após, a palavra “então” foi retirada, porém continua-se a usando para leitura, e a ação a ser tomada é finalizada por ponto-vírgula e finalizamos com outra chave. Repare um seguinte, sempre que se abrir um parêntese, chaves ou colchetes, é necessário fechá-lo.

Só para aumentar a sua compreensão, veja esse pseudocódigo comentado de como pode ser feito essa leitura.

```
Se (MinhaNota>NotaDoMeuAmigo)  
// Se a MinhaNota for maior que a NotaDoMeuAmigo, Então  
  
{  
    Oferecer ajuda; //Eu vou oferecer ajuda  
}
```

Perceba também que a ação a ser tomada, se encontra alinhada para a direita.

Até o momento, conseguiu entender como se dá a estrutura condicional simples?

Como pode perceber desbravador, em uma estrutura condicional, usamos alguns elementos de comparação. No exemplo da nota, ele se deu pela palavra maior, porém existem outros.

Veja abaixo a tabela com esses elementos e sua simbologia no mundo computacional que deve ser usado em seu pseudocódigo e posteriormente em seu código.

Operador de Comparação	Simbologia
Maior que	>
Menor que	<
Maior ou igual a	>=
Menor ou igual a	<=
Igual a	== (dois sinais de iguais seguidos)
Diferente de	!=

Tabela 1- Operadores de comparação e sua simbologia

Fique atento

A ação ‘Oferecer ajuda’ só acontecerá se a condição for verdadeira. Ou seja, se a minha nota for maior que a do meu amigo. Se por acaso a minha nota for menor, ‘Oferecer ajuda’ jamais entrará em ação. Isso significa que a estrutura condicional if, necessita que a sua condição seja atendida para que o arduino possa implementar a manipulação pretendida.

Mas já que aprendemos pseudocódigo...

Bora Praticar



Dado que x compreende um valor igual a 70, faça um pseudocódigo que:

- Se x for maior ou igual a 70, o aluno está aprovado;
- Se x for maior que 90, o aluno está aprovado com êxito;
- Se x for menor que 70 o aluno está de recuperação;
- Se x for menor ou igual a 30, o aluno está reprovado automaticamente.

Resposta:

Se (x>=70)

{

Aluno aprovado;

}

Se (x>90)

{

Aluno aprovado com êxito;

}

Se (x<70)

{

Aluno de recuperação;

}

Se (x<=30)


```
{  
    Aluno reprovado automaticamente;
```

Pense Rápido

Qual é a função da estrutura condicional if?



Resposta: A função dessa estrutura é condicionar ações. O que significa dizer que algumas manipulações só serão realizadas se a condição for verdadeira.

E aí desbravador, está tudo dentro da ‘caxola’? Está conseguindo compreender tudinho? Pois bem, já aprendemos como é o processo de uma estrutura condicional e como fazer o seu pseudocódigo, está na hora de colocarmos isso no programa. Com o conhecimento que temos em mãos, e se todos os exercícios propostos foram devidamente feitos, não encontraremos dificuldade em fazer essa transição.

Repare na comparação entre um pseudocódigo e um código na imagem abaixo:

1 - Faça um programa que Compare a sua nota da prova de matemática com a nota da do seu amigo e se a sua for maior, faça acender um LED.

Pseudocódigo

```
Se (MinhaNota>NotadoMeuAmigo)  
{  
    Acender LED;  
}
```

Código

```
if (MinhaNota>NotadoMeuAmigo)  
{  
    digitalWrite(LED,HIGH);  
}
```

Ilustração 2 - Comparação entre um pseudocódigo e um código

A palavra *se* do pseudocódigo, transforma-se em *if*, que é o seu significado em inglês. No mais, a estrutura segue quase que o mesmo processo, sendo feito apenas algumas modificações básicas. A ação deixou de ser descrita por definição e passou a se escrever como fazê-la.

Bem simples não é mesmo?

Vamos então retornar aquele exemplo do sensor de refletância que estávamos usando. Lembra que eu lhe falei que todas as vezes que o sensor ler um valor acima de 300 ele estará ‘vendo’ uma superfície escura e caso contrário a superfície será clara? Lembra que eu lhe disse também que essa é uma informação que nós sabemos, mas o arduino não? E que uma das formas de contar isso a ele é através da estrutura condicional? E que o sensor estava no nosso pino de entrada analógica 3? Então façamos um seguinte, para deixar o nosso exemplo mais lúdico, digamos que todas as vezes que eu o sensor ler uma faixa escura ele acende o led e todas as vezes que ele ler uma faixa clara ele o apagará.

Seguindo o modelo que estamos adotando até o momento, o nosso trecho do código que trata isso, ficaria assim:

```
void loop(){  
    if(analogRead(3)>300){  
        digitalWrite(led, HIGH);  
    }  
  
    if(analogRead(3)<300){  
        digitalWrite(led, LOW);  
    }  
}
```

Mas e se quiséssemos implementar neste código tudo o que vimos até o momento? Bem, para fazermos isso quero que saiba que o led está acoplado ao pino 12. Ficaríamos assim então:

```

#define led 12 // nomeação do dispositivo
#define sensor 3

void setup(){
    pinMode(led, OUTPUT); // configuração do dispositivo
    pinMode(sensor, INPUT);
}

void loop(){

    if(analogRead(sensor)>300){ // condição a ser atendida
        digitalWrite(led, HIGH); // ação a ser implementada
    }
    if(analogRead(sensor)<300){
        digitalWrite(led, LOW);
    }
}

```

Bem simples não é mesmo? Agora é só praticar um pouco. Pegue uma folha de papel e tente solucionar o exercício abaixo.

Bora Praticar



Faça um programa que todas as vezes que o sensor ler uma superfície escura, o led deverá piscar. Quando a superfície for clara um buzzer deverá aumentar gradualmente a sua intensidade de som, variando do mais baixo ao mais alto. Para isso, considere que o sensor se encontra no pino 4, o led no pino 12 e o buzzer no pino 11.

Resposta:

```

#define sensor 3
#define led 12
#define buzzer 11

void setup(){

    pinMode(sensor, INPUT);
    pinMode(led, OUTPUT);

```

```

    pinMode(buzzer,OUTPUT);
}

void loop(){

    if(analogRead(sensor)>300){

        analogWrite(buzzer,0);
        digitalWrite(led,HIGH);
        delay(1000);
        digitalWrite(led,LOW);
        delay(1000);
    }

    if(analogRead(sensor)<300){

        digitalWrite(led,LOW);
        analogWrite(buzzer,0);
        delay(1000);
        analogWrite(buzzer,50);
        delay(1000);
        analogWrite(buzzer,100);
        delay(1000);
        analogWrite(buzzer,150);
        delay(1000);
        analogWrite(buzzer,200);
        delay(1000);
        analogWrite(buzzer,255);
        delay(1000);
    }
}

```

Mas e se quiséssemos usar mais de um sensor. Como faríamos todo esse procedimento dentro de uma estrutura condicional? Você se arrisca a dizer?

Todas as vezes que quisermos utilizar mais de uma condição dentro do nosso if, devemos utilizar um recurso chamado de ‘Operadores Lógicos’.

Os operadores podem ser divididos entre “AND” (ou “E”) e “OR” (ou “OU”).

Operador AND

O operador “AND”, ou “E”, tem o objetivo principal de estabelecer uma relação de inclusão entre dois elementos. Por exemplo, quando dizemos: “Eu estou participando do torneio Arduino Challenge e estou gostando”, eu tenho uma relação de soma. Ambas coisas estão acontecendo. Quando isso é colocado em uma estrutura de condição, as ações provocadas por essas informações só acontecerão se ambas forem verdadeiras.

Na programação a sua simbologia é **&&**.

Por exemplo, veja o código abaixo:

```

if((sensor1)>300 && (sensor2)>300){
    digitalWrite(led,HIGH);
}

```

Neste caso, o led só será aceso se o sensor1 e o sensor2 lerem uma “faixa escura”. Se por acaso, o sensor1 ler uma “faixa clara” e o sensor2 ler uma “faixa escura”, ou se o sensor 1 ler uma “faixa escura” e o sensor2 ler uma “faixa clara”, o led não se acenderá.

Para que a ação aconteça, é necessário que **ambas as condições** sejam atendidas.

Com isso percebemos o seguinte:

Sensor 1	Sensor 2	Led
Faixa clara	Faixa clara	Não acende
Faixa clara	Faixa escura	Não acende
Faixa escura	Faixa clara	Não acende
Faixa escura	Faixa escura	Acende

Tabela 2 - Tabela da verdade do operador "AND"

É de fácil compreensão esse operador não é mesmo?

Pois bem, e se por acaso eu tiver dois sensores, mas se somente um deles ler “faixa escura”, já for suficiente para acender o led, em questão de programação, como faria isso acontecer?

Para responder essa pergunta, vamos conhecer o operador “OR”.

Operador OR

O operador “OR”, ou “OU”, tem por objetivo principal estabelecer uma relação de independência. Se **apenas um** dos elementos estiver atendendo a condição, então uma ação será realizada em cima disso.

Na programação a sua simbologia é ||.

Por exemplo, veja o código abaixo:

```

if((sensor1)>300 || (sensor2)>300){

  digitalWrite(LED, HIGH);

}

```

Neste caso, o led só não será acesso se o sensor1 e o sensor2 lerem “faixa clara”. Se por ventura, qualquer um dos dois lerem uma “faixa escura”, então o led deverá ser aceso.

Com isso percebemos o seguinte:

Sensor 1	Sensor 2	Led
Faixa clara	Faixa clara	Não acende
Faixa clara	Faixa escura	Acende
Faixa escura	Faixa clara	Acende
Faixa escura	Faixa escura	Acende

Tabela 3 - Tabela da verdade do operador "OR"

Agora perceba um seguinte: cada sensor tem separadamente duas opções de leitura de faixa: “faixa clara” ou “faixa escura”. A junção desses dois sensores me oferece quatro possibilidades de leitura: “faixa clara – faixa clara”, “faixa clara – faixa escura”, “faixa escura – faixa clara” e “faixa escura – faixa escura”.

Se por acaso, tivéssemos mais um sensor, teríamos 8 condições de leitura: “faixa clara – faixa-clara – faixa clara”, “faixa clara – faixa clara – faixa escura”, “faixa clara – faixa escura – faixa clara”, “faixa clara – faixa escura – faixa escura”, “faixa escura – faixa clara – faixa clara”, “faixa escura – faixa clara – faixa escura”, “faixa escura – faixa escura – faixa clara”, “faixa escura – faixa escura – faixa escura”. Isso acontece porque todas as possibilidades de um sensor, se junta com todas as possibilidades do outro sensor. Um pouco confuso não é mesmo? Veja a imagem abaixo usando apenas dois sensores e veja se clareia um pouco.

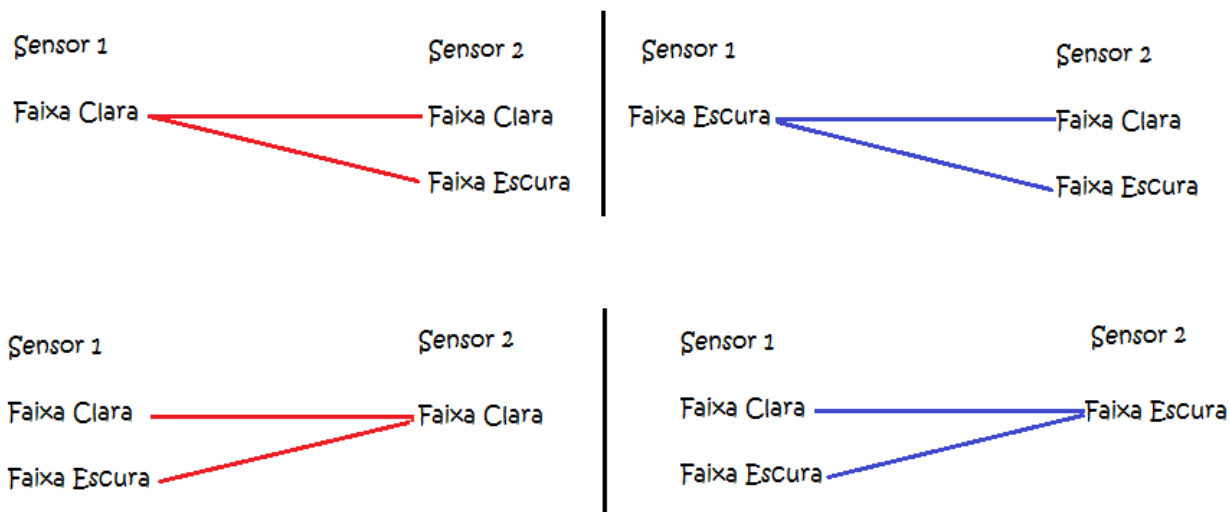


Ilustração 3 - Todas as combinações utilizando dois sensores

Se pegarmos todas as opções e fizermos um agrupamento de todas que se repetem, no fim, teremos apenas quatro opções válidas. Veja:

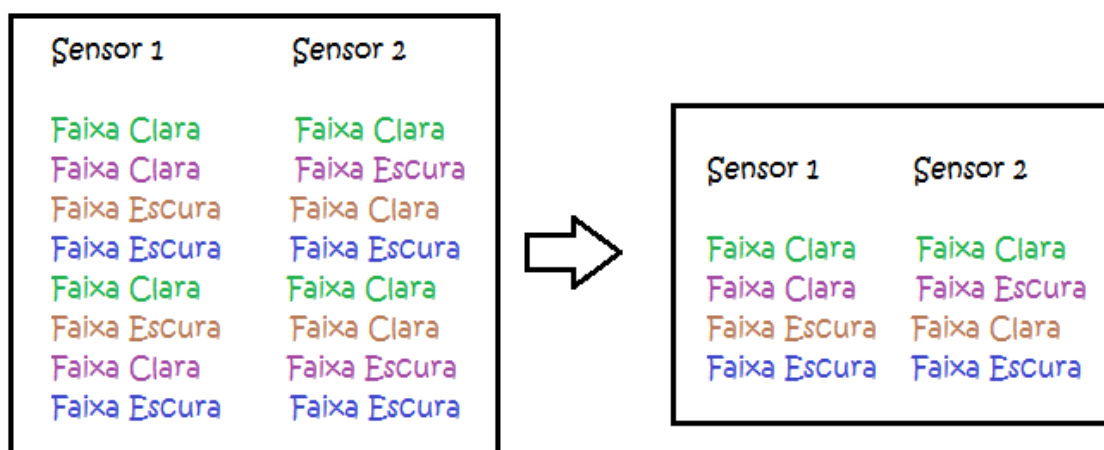


Ilustração 4 - Tabela da verdade de dois sensores

Agora imagine só ter que usar esse método para montar a tabela de 3 sensores! Iria ser bem trabalhoso não é mesmo? E se fossem 8 sensores? Seria uma “missão impossível”. Perderíamos muito tempo apenas descobrindo o número de opções possíveis de leitura para esses sensores juntos.

Fique atento

Por isso, é importante que você fique muito atento a uma regra que diz: “O número de linhas da tabela-verdade de uma proposição composta (número de opções de leitura dos sensores agrupados) depende do número de proposições simples que a integram, ou seja, a tabela-verdade de uma proposição composta com n proposições simples contém 2^n linhas.”

Com isso, se tivermos 03 sensores, basta que usemos 2^3 e assim sucessivamente. Se fossem quatro sensores, basta que colocasse 2 elevado a quarta potência. Como no caso, temos dois sensores, basta fazermos, 2^2 , resultando em 04 opções diferentes. Essa regra deixou as coisas mais claras não é mesmo?! Pois bem, não se esqueça dela, ela será indispensável durante a competição.

Mas agora que aprendemos a fazer estruturas condicionais compostas e já vimos como é a estrutura, usando os operadores lógicos “AND” e “OR”, está na hora de você mostrar que aprendeu os conceitos passados

Pense Rápido

O que são os operadores lógicos?
Quais são os existentes?



Resposta: Operadores Lógicos são ferramentas que a programação utiliza para interligar condições. Temos o operador lógico 'AND' ou 'E' e 'OR' ou 'OU'.

Calma, calma desbravador!! Já estamos praticamente finalizando esta apostila. Vamos ao nosso último comando que é o millis.

Millis()

O millis tem como função retornar a faixa de tempo desde que o arduino foi ligado até ele ser desligado. É válido ressaltar que o valor retornado é em milissegundos, ou seja, 5 segundos equivale a 5000, 1 segundo a 1000, $\frac{1}{2}$ segundo a 500. Este número chegará ao seu valor máximo e será reiniciado em um tempo de 50 dias. Em palavras gerais, ele contabiliza o tempo em que um programa começou a ser rodado, até 50 dias após

A sua estrutura é bem simples: *millis()*

Em suma, todas as vezes que eu quiser que algo aconteça, sendo condicionado a uma faixa de tempo, o millis é um bom pedido.

Pense Rápido

O que é o millis? Qual é sua diferença em relação ao delay?



Resposta: O millis é o responsável por contabilizar o tempo que o arduino foi ligado até ele ser desligado. A grande diferença entre o millis e o delay, é que o delay provoca um congelamento na varredura do código, enquanto que o millis executa uma contagem sem paralisação.