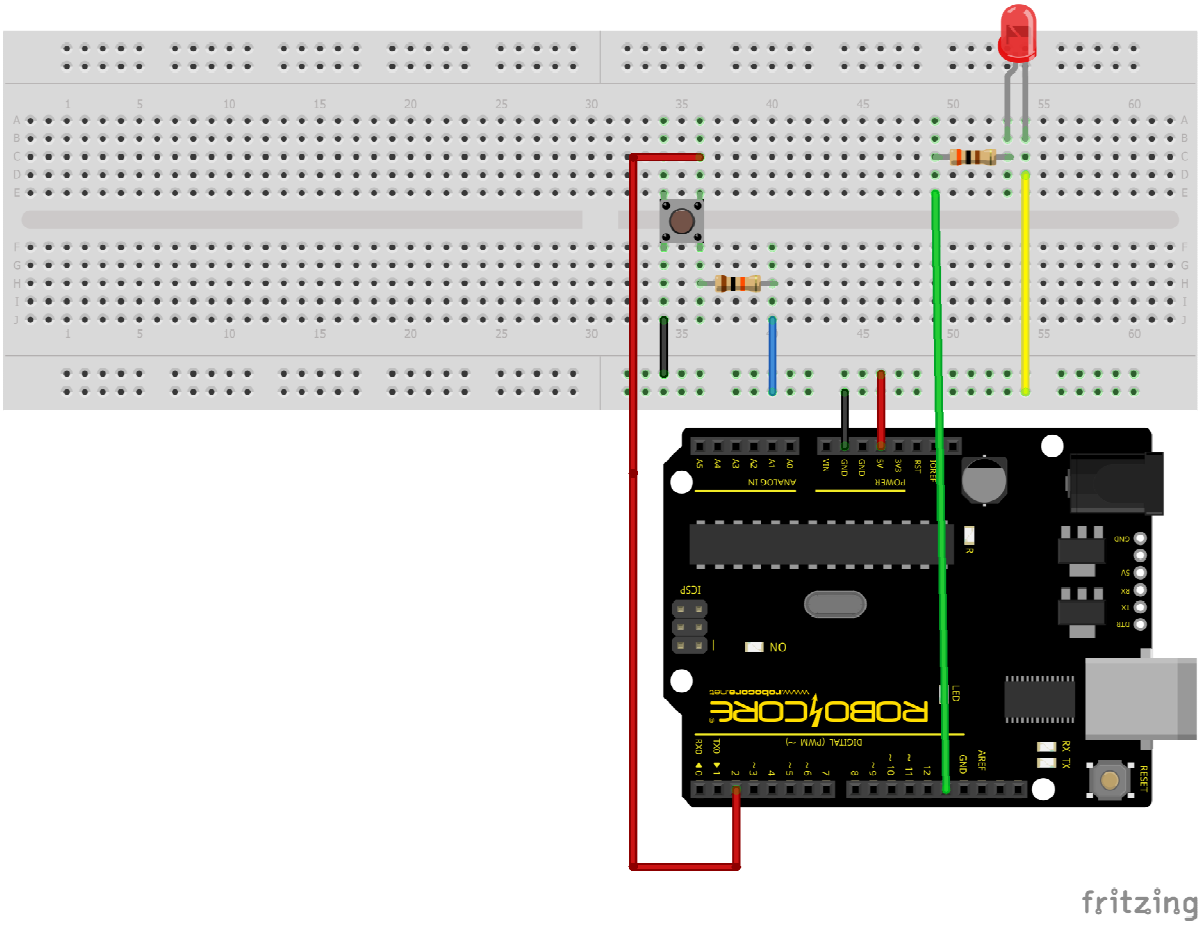
- **Módulo 1**

Componentes: 1 Botão + 1 Led
Descrição: Conforme você pressiona um pushbutton, um led é aceso
Dificuldade:

Trata-se de fazer um botão acender um led quando pressionado e, quando solto, o led deverá apagar. Coloque os componentes como está sendo mostrado na imagem abaixo, bem como suas ligações:

Componentes utilizados: 01x Led Vermelho / 01x Resistor de 300Ω / 01x PushButton / 01x Resistor de 10kΩ / cabos diversos.




***Dica 1:*** *Caso tenha dificuldades em montar o circuito, a imagem a cima (e todas as outras imagens de circuitos) estão disponíveisem alta definição na pasta "**Imagens dos Experimentos**" no CD que acompanha este kit.*

***Dica 2:*** *Não importam as cores dos fios na hora de ligação, e sim fazer a ligação correta. Preste bastante atenção ao fazer as ligações pois apenas 1 fio no lugar errado pode comprometer todo o experimento.*

***Dica 3:*** *Se divirta! Mais que estudar e aprender, você vai se divertir com esta placa :)*

Código:

```
/*****************************************\
**    ROBOCORE ARDUINO KIT INICIANTE    **
*                                         *
**              Módulo 1                 **
\*****************************************/

const int ledPin = 13; //led no pino 13
const int Botao = 2; //botao no pino 2
int EstadoBotao = 0; //Variavel para ler o status do pushbutton
void setup(){
 pinMode(ledPin, OUTPUT); //Pino do led será saída
 pinMode(Botao, INPUT); //Pino com botão será entrada
}
void loop(){
 EstadoBotao = digitalRead(Botao); /*novo estado do botão vai ser igual ao que
                                     Arduino ler no pino onde está o botão.
                                      Poderá ser ALTO (HIGH)se o botão estiver
                                     Pressionado, ou BAIXO (LOW),se o botão
                                     estiver solto */
 if (EstadoBotao == HIGH){   //Se botão estiver pressionado (HIGH)
  digitalWrite(ledPin, HIGH); // acende o led do pino 13.
  }
  else{                        //se não estiver pressionado
digitalWrite(ledPin, LOW); //deixa o led do pino 13 apagado
 }
}
```

Recomendamos que você tente entender passo a passo o programa anterior, para não ter problemas quando os códigos começarem a ficar mais complexos.

Após compilar o código e fazer o upload na sua placa, você já deve poder apertar o botão e o led da protoboard acender e, quando soltar, o led deve apagar.

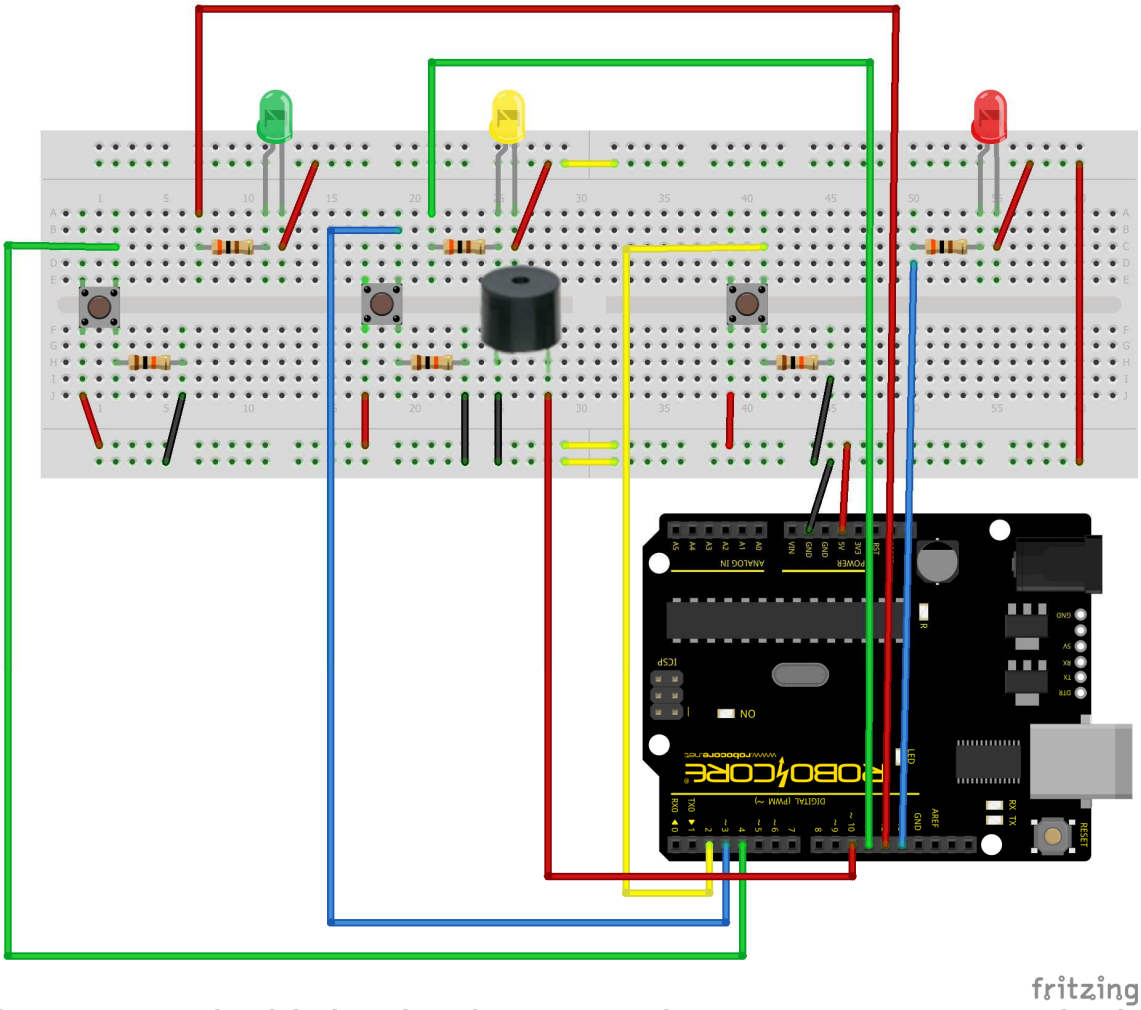
Se houve algum problema, procure seu erro e tente consertá-lo. Se não, parabéns! Você concluiu o primeiro módulo RoboCore Arduino Kit Iniciante. Agora você está pronto para começar o módulo de número 2.

- **Projeto Piano**

Componentes: 3 Botões + 3 Leds + Buzzer
Descrição: Cada botão toca uma nota musical diferente e acende um led. É expansível – por conta do usuário – para mais uma nota musical com o botão (e o led) reserva.
Dificuldade: 

Utilizando os conceitos aprendidos nos módulos 1 e 2, podemos agora montar o primeiro projeto: o Projeto Piano. Neste projeto cada um dos 3 botões tocará uma nota musical diferente. Para montar o projeto usaremos um novo componente: o Buzzer. Um Buzzer nada mais é do que um pequeno alto-falante. Obviamente que ele não consegue tocar músicas, mas consegue fazer apitos soarem, como sirenes ou alarmes. A maioria dos alarmes de pequenos equipamentos eletrônicos é feito através de um buzzer. Ele funciona da seguinte maneira: quando alimentado por uma fonte oscilante, componentes metálicos internos vibram, produzindo assim um som. Para este experimento, você também pode utilizar um pequeno alto-falante (o som sai mais puro e a diferença entre as notas musicais é mais nítida). Último detalhe sobre o Buzzer: ele tem polaridade. Olhando o componente de perto, você poderá ver um sinal de positivo (+). Este sinal mostra onde está o pino positivo do componente. Sempre ligue este a uma saída digital do Arduino e o outro em GND. Para fazer a montagem, o modelo a seguir pode ser seguido:

Componentes utilizados: 01x Led Verde / 01x Led Amarelo / 01x Led Vermelho / 03x Resistores de 300 Ω / 03x Resistores de 10k Ω / 03x Pushbutton / cabos diversos / 01x Buzzer 5V.




**Note que são usados dois tipos de resistores, por mais que pareçam ter o mesmo valor eles são diferentes!**

Veja que a única diferença entre este projeto e o módulo 2 é a inserção de um Buzzer.

Código:

```
/*****************************************\
**    ROBOCORE ARDUINO KIT INICIANTE    **
*                                         *
**            Projeto Piano              **
\*****************************************/


const int ledPin1 = 13;
const int ledPin2 = 12;
const int ledPin3 = 11;
const int Botao1 = 2;
const int Botao2 = 3;
const int Botao3 = 4;
const int Buzzer = 10;  //O buzzer está colocado no pino 10
int EstadoBotao1 = 0;
int EstadoBotao2 = 0;
int EstadoBotao3 = 0;

void setup() {
 pinMode(Buzzer, OUTPUT);
 pinMode(ledPin1, OUTPUT);
 pinMode(Botao1, INPUT);
 pinMode(ledPin2, OUTPUT);
 pinMode(Botao2, INPUT);
 pinMode(ledPin3, OUTPUT);
 pinMode(Botao3, INPUT);
}

void loop(){
 EstadoBotao1 = digitalRead(Botao1);
 EstadoBotao2 = digitalRead(Botao2);
 EstadoBotao3 = digitalRead(Botao3);
    if(EstadoBotao1 && !EstadoBotao2 && !EstadoBotao3) {
        digitalWrite(ledPin1, HIGH);
        tone(Buzzer, 100);
      }
    if(EstadoBotao2 && !EstadoBotao1 && !EstadoBotao3) {
        digitalWrite(ledPin2, HIGH);
        tone(Buzzer, 500);
      }
    if(EstadoBotao3 && !EstadoBotao2 && !EstadoBotao1) {
        digitalWrite(ledPin3, HIGH);
        tone(Buzzer, 1000);
      }
 digitalWrite(ledPin1, LOW);
 digitalWrite(ledPin2, LOW);
 digitalWrite(ledPin3, LOW);
 noTone(Buzzer);
}
```

O que estamos fazendo de diferente nesse código é que, dentro dos parentesis condicionais IF utilizamos símbolos como && e também !. Os símbolos de && significam que o código dentro da rotina IF só irá ser executado se todas as condições acontecerem. O símbolo de ! representa que queremos testar se a variável retorna o valor de falso. Assim, o código irá executar os comandos dentro do primeiro IF se o botão 1 estiver pressionado, e os botões 2 e 3 estiverem soltos. De forma análoga poderíamos escrever o código dentro do IF dessa forma:

```
if((EstadoBotao1 == true) && (EstadoBotao2 == false) && (EstadoBotao3 == false)){
```

Porém, para economizar linhas de código, podemos também colocar da forma que foi proposta.

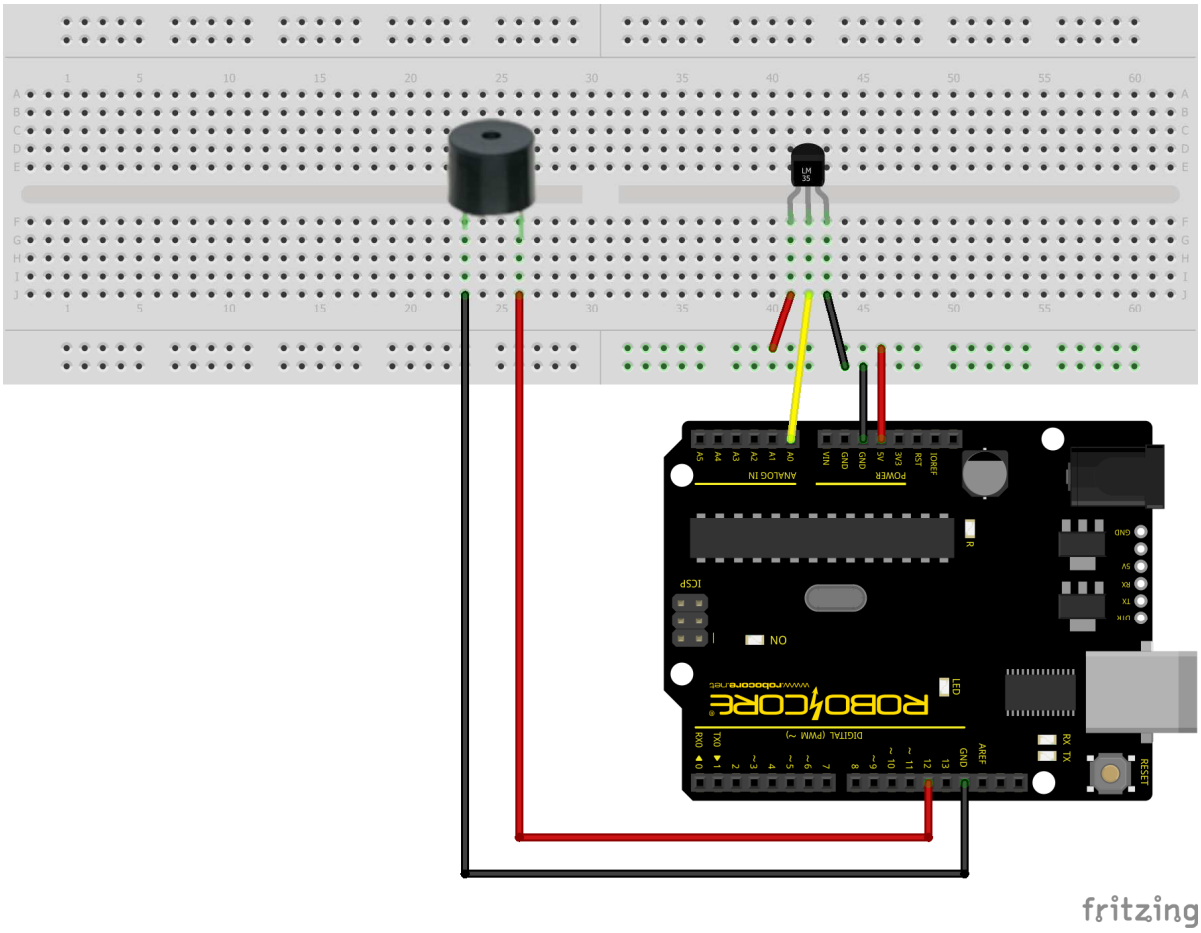
A função *tone()* serve justamente para fazer soar um tom, ou seja, uma nota musical. Como parâmetros, ou os valores entre parentesis, devemos inserir o pino que está o buzzer, e a frequência da nota musical a ser tocada. E sempre devemos usar o comando *noTone()* para que o buzzer pare de tocar. Mais sobre a função *tone()* será mostrado futuramente nesse material.

- **Projeto Alarme**

Componentes: 1 Sensor de Temperatura LM35 + 1 buzzer
Descrição: A partir dos valores lidos no módulo 3, o usuário poderá montar um alarme que, se a temperatura de onde o sensor estiver localizado for maior, ou menor, ele soará.
Dificuldade:

O intuito é muito simples: quando a temperatura for maior que um valor, escolhido por você, o buzzer começará a soar até que a temperatura volte ao estado perfeito. Este procedimento é muito usado em indústrias com processos e também em sensores de calor.
O circuito é o seguinte:

Componentes utilizados: 01x LM35 (sensor de temperatura) / cabos diversos / 01x Buzzer 5V.




***Veja que a única diferença desta montagem para a anterior é a inserção de um buzzer, com seu pino positivo (com um sinal de + no topo do componente) ligado no pino digital 12 e o outro pino ligado em GND.***

Código:

```
/****************************************\
 **     ROBOCORE ARDUINO KIT INICIANTE    **
 *                                        *
 **             Projeto Alarme           **
 \****************************************/

const int LM35 = 0;
float temperatura = 0;
int ADClido = 0;
const int Buzzer = 12;

void setup(){
  analogReference(INTERNAL); //No Mega, use INTERNAL1V1, no Leonardo remova essa linha
  pinMode(Buzzer, OUTPUT);
}

void loop(){
  ADClido = analogRead(LM35);
  temperatura = ADClido * 0.1075268817204301; //no Leonardo use 0.4887585532
  if(temperatura > 25){ // setei como 25ºC
     delay(1000);
     if(temperatura > 25){
        tone(Buzzer, 2000);
     }
   }
   else{
     noTone(Buzzer);
      }
   }
```

Deste código para o do módulo anterior tivemos poucas mudanças. A começar pela inserção de uma nova variável chamada **Buzzer**, que de fato não é uma variável por isso escrevemos o **const.** Esta constante irá informar ao Arduino onde nosso pino positivo do buzzer está ligado, neste caso no pino digital 12. A próxima mudança no código foi dizer, no bloco **setup** do código, que o pino onde está o buzzer é uma saída, ou seja, iremos usar o pino quando a temperatura for maior que alguma escolhida. Você conseguiu identificar no código onde estão estas duas mudanças?

A mudança no **loop** é um pouco mais significativa. Vamos dar uma olhada:

```
void loop(){
  ADClido = analogRead(LM35);
  temperatura = ADClido * 0.1075268817204301; //no Leonardo use 0.4887585532
  if(temperatura > 25){ // setei como 25ºC
     delay(1000);
     if(temperatura > 25){
        tone(Buzzer, 2000);
     }
   }
   else{
     noTone(Buzzer);
      }
   }
```

A começar que tiramos a escrita no monitor serial. Temos então duas rotinas if, uma dentro da outra. Entre elas existe um delay de 1 segundo. Fazemos isso para que o buzzer só seja ativado quando a temperatura realmente for maior que 25ºC. Sem o delay e o segundo if, o buzzer ficaria intermitente. Portanto, o segundo if serve apenas para garantir que a temperatura desejada foi atingida. As rotinas if neste caso funcionam assim:

**SE** a temperatura for **MAIOR** que **25** faça: `aguarde 1 segundo.`
**SE** a temperatura continuar sendo **MAIOR** que **25** faça: `tone(Buzzer, 2000);`
**SE NÃO** faça: `noTone(Buzzer);`

Veja que só coloquei 25ºC no teste pois onde o manual foi escrito a temperatura ambiente é de 23ºC. Colocando o dedo no sensor, a temperatura sobe acima dos 25ºC e assim consigo testar. Se você está em um lugar mais quente, altere este valor até encontrar um que dê resultados.

- **Projeto Iluminação Automatizada**

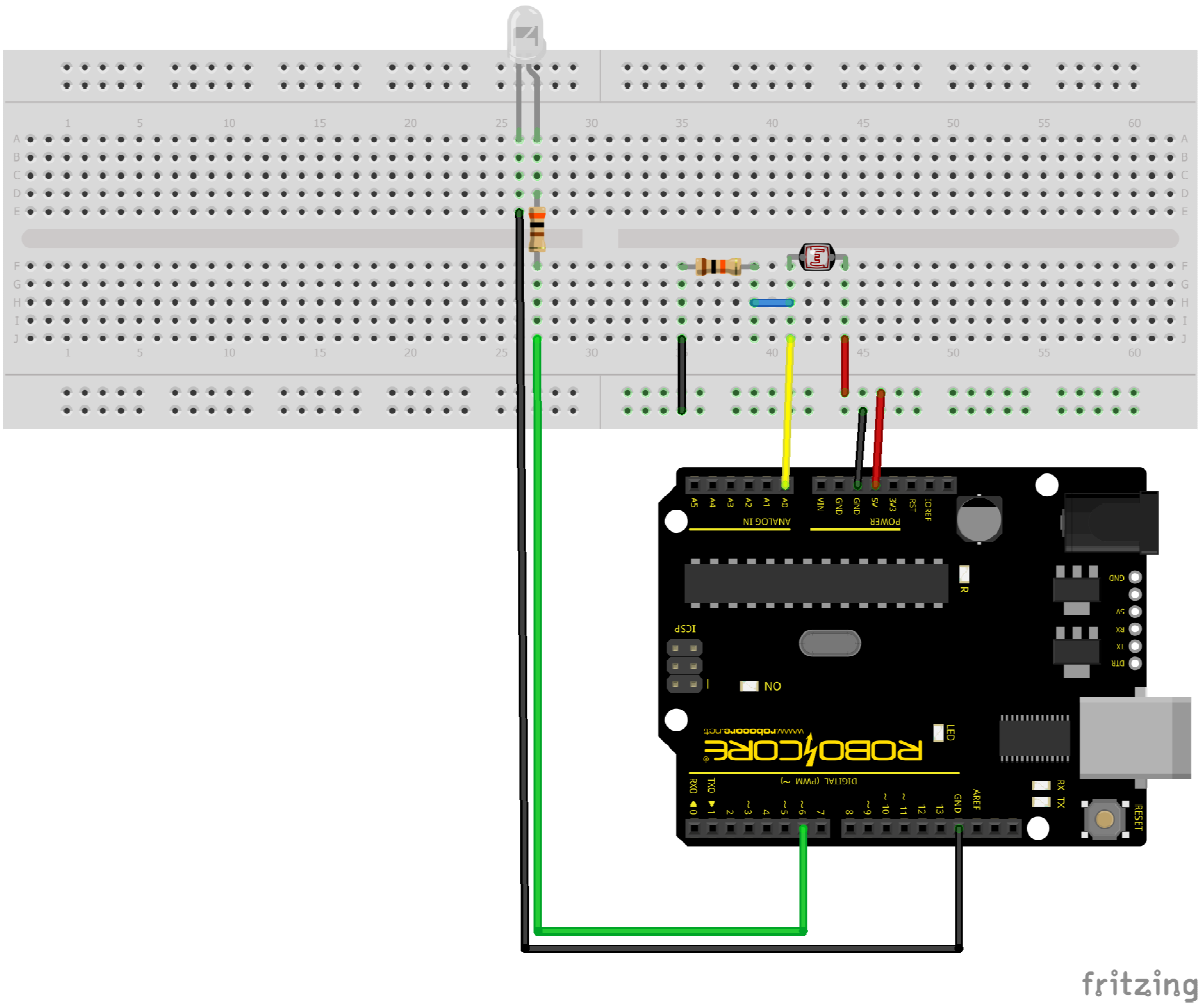
Componentes: 1 Led Alto Brilho + 1 Sensor de Luminosidade LDR
Descrição: Se a iluminação ambiente, por qualquer motivo, diminuir ou apagar completamente, um led de alto brilho acende gradativamente.
Dificuldade: 

Com conceitos de entradas analógicas, sensores e pwm, já podemos pensar em um projeto de automação. Projeto semelhante é utilizado em postes de luz, onde as lâmpadas acendem sozinhas, conforme a luminosidade do dia - ou você acha que todo dia uma pessoa responsável liga e desliga todas as luzes de todos os postes de todas as ruas?

Para este projeto, usaremos um LDR. LDR nada mais é do que uma resistência que varia conforme a luminosidade: é um sensor de luminosidade.

Monte o seguinte circuito:

Componentes utilizados: 01x LDR (sensor de luinosidade) / 01x Reistor de 10k / 01x Led de Alto Brilho / 01x Resistor de 300 / cabos diversos.

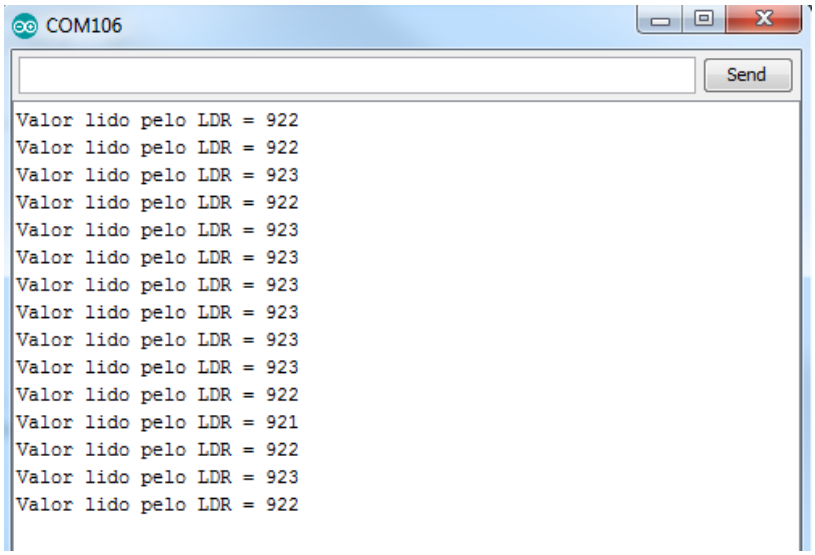



Veja que o led está colocado no pino 6 digital e o sensor de luminosidade no pino 0 analógico. Antes de qualquer coisa, temos que calibrar o sensor.

Código:

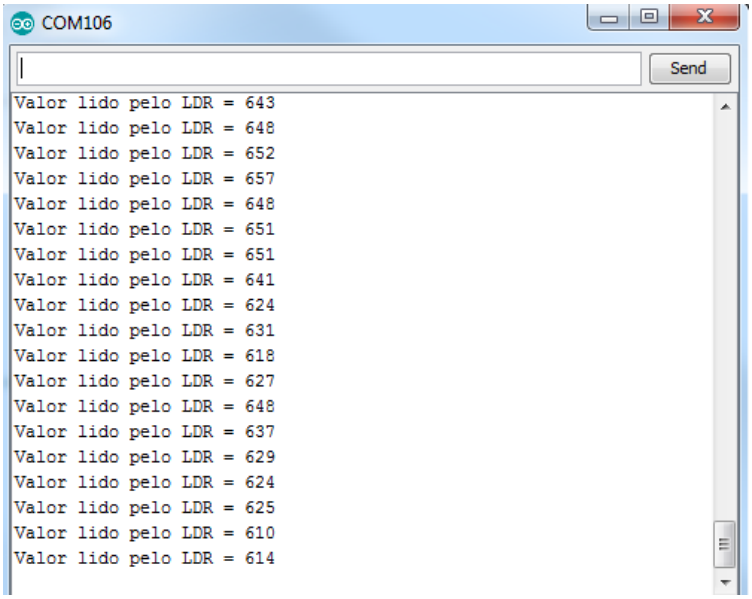
```
/*****************************************\
**     ROBOCORE ARDUINO KIT INICIANTE    **
*                                         *
**             Calibrar LDR              **
\*****************************************/

       const int LDR = 0;
       int valorLido = 0;

       void setup() {
         Serial.begin(9600);
       }

       void loop() {
         valorLido = analogRead(LDR);
         Serial.print("Valor lido pelo LDR = ");
         Serial.println(valorLido);
         delay(500);
       }
```

O código acima mostra no monitor serial os valores que o LDR está lendo. No caso da iluminação no local onde este material foi escrito, os valores lidos pelo LDR são os seguintes:




Você deve estar vendo em seu monitor algo parecido com esta figura acima. Se você está lendo um valor fixo de 1023 ou 0, certifique-se que os componentes estão bem colocados e na posição correta. Este é um erro muito comum neste tipo de experimento.

Coloque agora a palma da sua mão, ou qualquer outro material que tampe a luz ambiente, sobre o sensor tampando a luz e fazendo o sensor ficar na sombra. Você deve ler valores como os seguintes:




Como deve ter ficado subentendido:
*Quanto mais luz o LDR receber, mais alto será o valor lido.*
*Quanto menos luz o LDR receber, menor será o valor lido.*

Agora já temos os valores para calibrar nosso sensor. Vamos supor que você queira fazer com que um led acenda quando o valor lido é de 750 (uma sombra moderada sobre o LDR). Podemos então utilizar o seguinte código para fazer este projeto:

```
/*****************************************\
**     ROBOCORE ARDUINO KIT INICIANTE    **
*                                         *
**   Projeto Iluminação Automatizada     **
\*****************************************/

       const int LDR = 0;
       const int Led = 6;
       int ValorLido = 0;
       int pwm = 0;

       void setup() {
         pinMode(Led, OUTPUT);
       }

       void loop() {
         ValorLido = analogRead(LDR);

         if (ValorLido < 750){
          analogWrite(Led, pwm);
          pwm++;
          delay(100);
         }
         else{
          digitalWrite(Led, LOW);
          pwm = 0;
         }

          if(pwm > 255){
          pwm=255;
          }
       }
```

A maior parte dos elementos deste código já foi estudada. Vamos para a parte que merece nossa atenção no **loop**:

```
void loop() {
  ValorLido = analogRead(LDR);

  if (ValorLido < 750){
   analogWrite(Led, pwm);
   pwm++;
   delay(100);
  }
  else{
   digitalWrite(Led, LOW);
   pwm = 0;
  }

   if(pwm > 255){
   pwm=255;
   }
}
```

Primeiramente assimilamos o valor lido pelo LDR com a variável ValorLido. Depois disso fazemos as seguintes condições:

**SE** a variável **ValorLido** for **MENOR** que 750 (uma leve sombra), **FAÇA:**
**Escreva** de uma maneira **ANALÓGICA**, ou seja, PWM no **Led** e
Some **1** na variável **pwm** (na linguagem C, colocar uma variável seguida de dois sinais de positivo significa somar 1 a esta variável), aguarde 100 milisegundos para podermos ver o efeito ocorrer;

**SE NÃO** (ou seja, Se ValorLido for MAIOR que 750), **FAÇA:**
**Escreva** de uma maneira **DIGITAL**, ou seja, alto ou baixo no **Led** e o apague (**LOW**), e também zere a variável pwm para que o efeito possa acontecer sempre que o led apagar;

A próxima condição serve apenas para garantir que a variável pwm não ultrapasse 255, pois, como já visto, para fazer escritas analógicas com pwm podemos usar valores indo de 0 a 255.

**SE** a variável **pwm** for **MAIOR** que 255, **FAÇA:**
**pwm** é **IGUAL** a 255 (desta forma garantimos que pwm nunca passará dos 255).

Pronto. Compile e faça o upload deste código juntamente com o circuito montado e veja que circuito útil você tem agora em mãos.

Agora podemos ir para o próximo projeto, e sem dúvida um dos mais complexos de todos até agora.

Se você não teve dúvidas até agora está pronto para desenvolvê-lo. Se algo não saiu como os conformes, refaça quantas vezes for necessário o experimento.

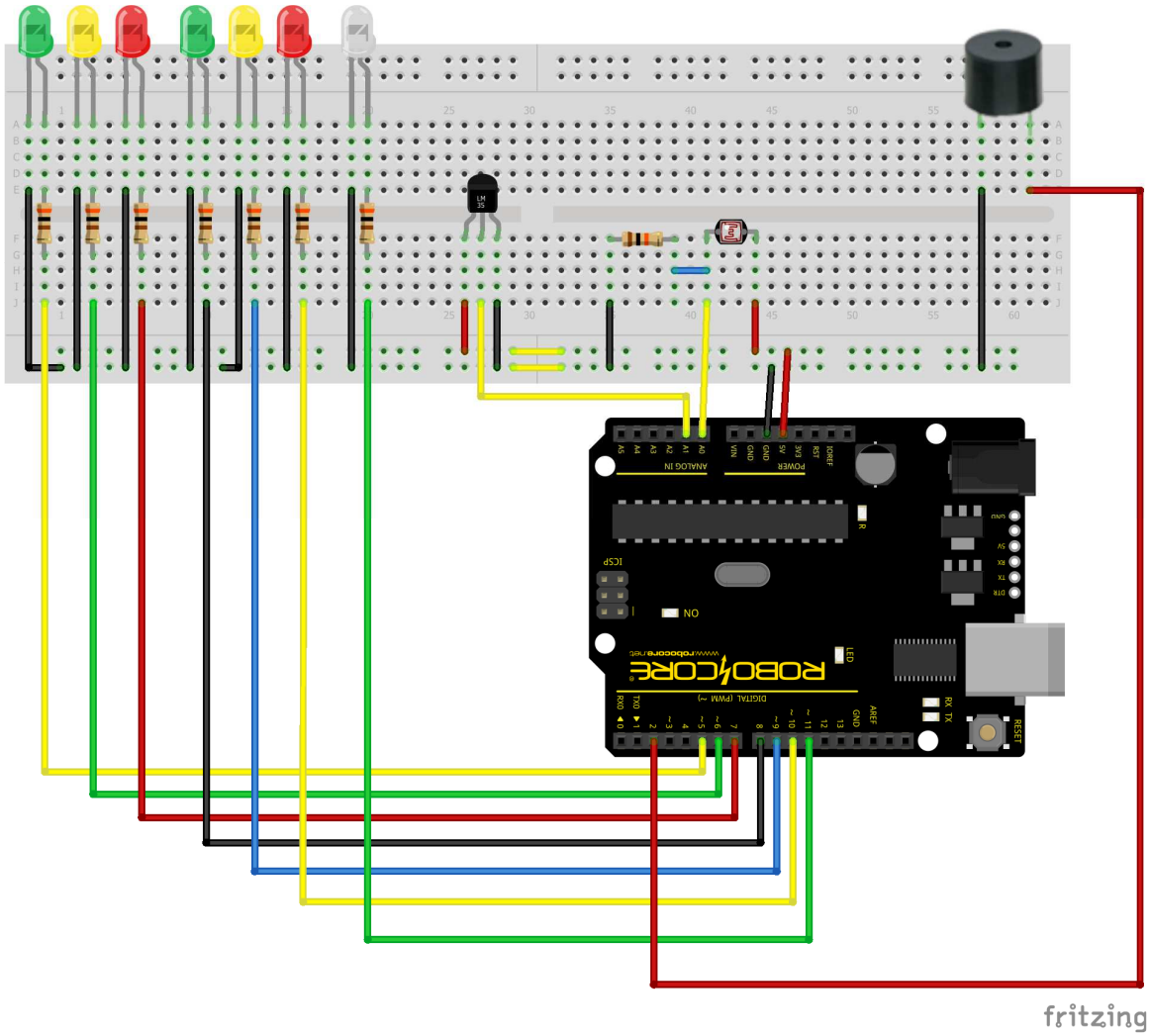
- **Projeto Alarme Multipropósito**

Componentes: 2 Leds Verdes + 2 Leds Amarelos + 2 Leds Vermelhos + 1 Sensor de Luminosidade LDR + 1 Sensor de Temperatura LM35 + 1 Led Alto Brilho + 1 buzzer
Descrição: Temos dois indicadores: um de luminosidade e outro de temperatura, através das cores dos leds. Se a temperatura estiver alta deverá acender os 3 Leds que a corresponde. Se os 3 Leds correspondentes à luminosidade estiverem apagados – indicando uma falta total de luminosidade no ambiente - um alarme deverá soar e um led de alto brilho irá acender.

Dificuldade: 

Para este complexo sistema, o circuito montado deve parecer com o seguinte:

Componentes utilizados: 02x Led Verde / 02x Led Amarelo / 02x Led Vermelho / 01x Led de Alto Brilho / 01x Buzzer 5V / 07x Resistor 300 Ω / 01x Resistor 10k Ω / 01x LM35 / 01x LDR / cabos diversos.




***Dica: tente usar as próprias "pernas" dos componentes para fazer as ligações, desse modo utilizando a menor quantidade de fios possível.***

***Cuidado: preste muita atenção em cada ligação para tudo dar certo ao final do experimento.***

Poderíamos aqui utilizar a barra de LEDs, porém vamos usar LEDs comuns para diversificar um pouco os experimentos.

Código:

```
/*****************************************\
 **    ROBOCORE ARDUINO KIT INICIANTE    **
 *                                        *
 **    Projeto Alarme Multipropósito     **
 \*****************************************/

const int LDR = 0;
const int LM35= 1;
const int Buzzer = 2;
const int led[] = {
  5,6,7,8,9,10,11};
int ValorLDR = 0;
int ADClido = 0;
float temperatura = 0;
int pwm = 0;

void setup(){
  for(int x = 0; x < 7 ; x++){
    pinMode(led[x], OUTPUT);
  }
  pinMode(Buzzer,OUTPUT);
}

void loop(){
  ValorLDR = analogRead(LDR);
  ADClido = analogRead(LM35);
  temperatura = ADClido * 0.4887585532;
  if (temperatura > 20.00){
    digitalWrite(led[0], HIGH);
  }
  else{
    digitalWrite(led[0], LOW);
  }

  if (temperatura > 22.00){
    digitalWrite(led[1], HIGH);
  }
  else{
    digitalWrite(led[1], LOW);
  }

  if (temperatura > 24.00){
    digitalWrite(led[2], HIGH);
  }
  else{
    digitalWrite(led[2], LOW);
  }

  if (ValorLDR > 950){
    digitalWrite(led[5], HIGH);
  }
  else{
    digitalWrite(led[5], LOW);
  }
  if (ValorLDR > 825){
    digitalWrite(led[4], HIGH);
  }
  else{
    digitalWrite(led[4], LOW);
  }
  if (ValorLDR > 750){
    digitalWrite(led[3], HIGH);
    digitalWrite(led[6], LOW);
    noTone(Buzzer);
  }
  else{
    digitalWrite(led[3], LOW);
    digitalWrite(led[6], HIGH);
    tone(Buzzer, 2000);
  }
}
```

Neste ponto você já é capaz de entender perfeitamente o que se passa neste projeto, mesmo porque ele é apenas uma junção de alguns módulos vistos nesta apostila com alguns projetos.

Existe apenas uma pequena diferença do que fizemos até agora, você consegue encontrar qual é?

Nos primeiros experimentos com LM35, usando um UNO, BlackBoard ou Mega, multiplicávamos o valor do ADClido por 0,1075268817204301 para encontrar a temperatura em graus Celsius após a leitura pelo conversor analógico digital, e agora estamos multiplicando por 0.4887585532.
Você sabe dizer por quê?

Se você está atento, vai lembrar que nos primeiros experimentos com LM35 utilizamos um recusro chamado ***analogReference(INTERNAL)*** para mudar a referência das portas analógicas para um valor máximo de 1,1V. Neste experimento do alarme multipropósito não podemos usar este artifício e é claro que você sabe o porque disso. Nós temos que usar a referência das portas analógicas no modo padrão, ou seja, 5V, pois o sensor de luminosidade LDR envia valores para a entrada analógica de 0 a 5V, e se usássemos a referência de 1,1V provavelmente iríamos danificar nossa placa Arduino ao ler os dados do LDR. Chegamos no valor de 0.4887585532746823, pois fizemos aquela mesma regra de três anteior, mas com o valor máximo de 5V ao invés de 1,1V, ou seja, máximo de 500ºC, portanto fica assim:

$$\begin{array}{cc} 500ºC & 1023 \\ \text{temperatura} & ADC_{LIDO} \end{array}$$

$$\downarrow$$

$$\text{temperatura} = \frac{500}{1023} \times ADC_{LIDO}$$

$$\downarrow$$

$$\text{temperatura} = 0.4887585532746823 \times ADC_{LIDO}$$

A diferença de se usar o sensor desta forma é que ele não fica tão preciso quanto ficaria se usássemos a mudança da referência analógica.

Quanto aos valores das condições ***IF****, se eles* não estão satisfatórios para seu projeto, sinta-se a vontade para alterá-los como quiser. Uma dica é sempre fazer a calibragem do sensor que você for utilizar, com o código proposto no Projeto Iluminação Automatizada.

## MINI GLOSSÁRIO

Abaixo você encontrará algumas estruturas usadas na progrmação do Arduino:

**Estrutura básica de código:**

```
void setup(){
    //setup do código
}
void loop(){
  //loop do código
}
```

**Estrutura condicional if:**

```
if(variável == valor){
 //faça alguma coisa se
 //valor da variável for
//igual ao valor testado
}
else{
  // senão, faça outra
  //coisa
}
```

**Estrutura for:**

```
for(int x = 0; x < 10; x++){
//contagem de 0 a 9
//seu código vai se repetir
//10 vezes dentro desse loop
//nessas condições
}
```

**Estrutura switch case:**

```
switch (variável){
   case 1:
   //faça algo se
variável =1
     break;
  case 2:
     //faça algo se
variável =2
     break;
  default:
 //se variável não for
 //nem 1 nem 2, faça o
 //que estiver no default
}
```

**Estrutura while:**

```
while( variável < 10 ){
  //faça algo durante 10
vezes
  //pois incrimentamos de 1
em
  // 1 na linha abaixo

  variavel = variavel + 1;

}
```

**Estrutura do-while:**

```
do{

  //faça algo enquanto....

} while( variavel < valor );
```

**Operadores de comparação:**

```
== (igual a)
!= (difrente de)
< (menor que)
> (maior que)
<= (menor ou igual a)
>= (maior ou igual a)
&& (operador lógico AND)
|| (operador lógico OR)
! (operador lógico NOT)
```

**Configuração de pinagem:**

```
pinMode(pino, INPUT);
//seta o pino como ENTRADA

pinMode(pino, OUTPUT);
//seta o pino como SAÍDA

pinMode(pino,INPUT_PULLUP);
//seta como pull-up o pino
de entrada
```

**Pino Digital - Leitura e Escrita:**

```
digitalWrite(pino, HIGH);
//seta o pino como nivel
logico alto

digitalWrite(pino, LOW);
//seta o pino como nível
logico baixo

digitalRead(pino);
//retorna se o pino está HIGH
ou LOW
```

**Pino Analógico - Leitura e Escrita:**

```
analogRead(pino);
//faz a leitura de um
pino de entrada analógica

analogWrite(pino);
//envia uma tensão
analógica para um pino de
saída usando PWM
```

**Comunicação Serial:**

```
Serial.begin(9600);
//ajusta o baudrate da
comunicação

Serial.print("olá");
//imprime na porta serial a
palavra olá
```

Conheça muitas outras funções e estruturas de código acessando:

www.Arduino.cc

Esperamos que esta apostila tenha-lhe sido válida e que você seja mais um *Arduinizador do Mundo.*

***Arduinize o Mundo*** ***RoboCore Kit Iniciante Para Arduino*** ***WWW.ROBOCORE.NET***