PROFESSOR DANILO *PROJETOS DE CIÊNCIAS – 09/03/2022*

AULA 06

Vamos deixar o transistor de lado por enquanto, embora, você possa entrar no tinkercad e montar o circuito da aula passada por lá. Vamos então à programação do Arduino, afinal, esperamos tanto por isso. Vamos começar ditero pelo circuito: Montar um pequeno semáforo para entender a saída digital.

MONTAGEM DO CIRCUITO

Nesta aula, vamos criar um semáforo simples, que mantém o sinal verde por 5 segundos, passando depois para amarela (desligando a verde), mantendo-se assim por 2 segundos, posteriormente desliga-se o amarelo e liga o vermelho por 5 segundos. Passado este tempo, o ciclo recomeça mantendo apenas o verde asceso.

Para isso, você vai usilizar um resistor de 200 Ohm, ou algum próximo: o professor irá te informar qual o disponível.

Siga os passos das imagens a seguir, porém note que a matriz de contato do professor é diferente da sua. Além disso, aproveito para colocar um link para o tinkercad com um circuito semelhante e uma imagem deste cicuito na última figura. Lembre-se também que você pode baixar este circuito pelo QR-code abaixo.

Figura 1:baixe este material se quiser uma versão digital e colorida

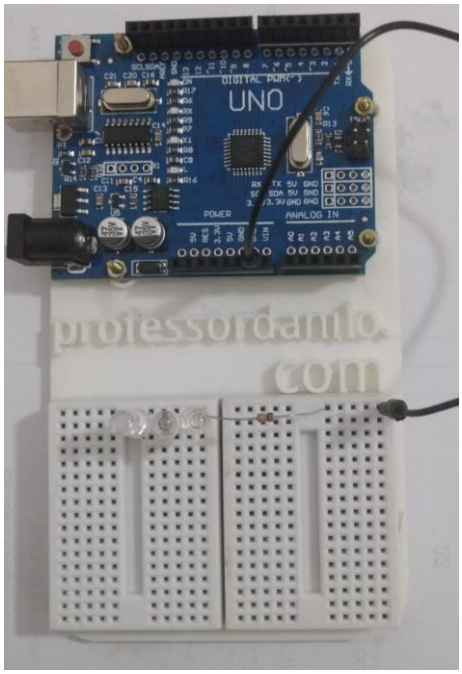

Figura 2:No GND o professor conectou três LEDS, nas cores verde, amarela e azul, olhando da direita para a esquerda. Lembre-se que o LED é polarizado e a perna mais cura deve ser conectada ao GND. O professor colocou, portanto, todos os LEDs com a perna menor do outro lado do resistor.

Figura 3: Veja detalhe da conexão dos resistores com o resistor.

Figura 4: agora conectamos um fio em cada um dos LEDs. O professor escolheu usar os fios nas cores correspondentes às cores do LED. Se quiser ver este material na sua versão colorida, leia o QR-code abaixo.

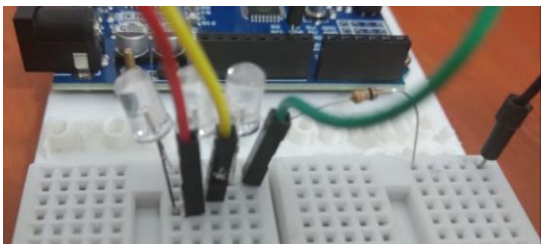

Figura 5: Veja de outro ângulo as conexões dos fios com os LEDs.

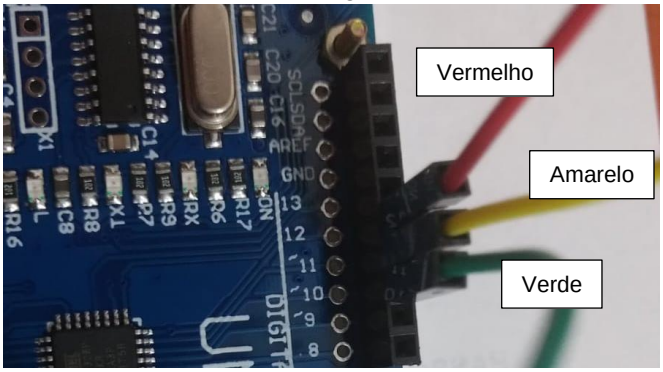

Figura 6: Vamos agora conectar os fios Verde, Amarelo e Vermelho, respectivamente, nas portas digitais 10, 11 e 12.

PROFESSOR DANILO *PROJETOS DE CIÊNCIAS – 09/03/2022*

Veja figura do circuito no tinkercad. Link no QR-Code.

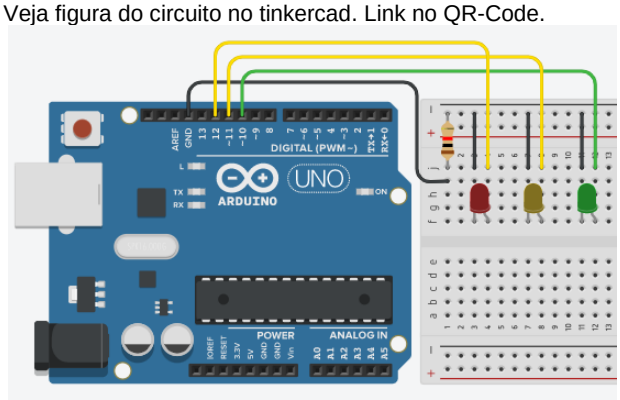

Figura 7: Circuito a ser montado na aula feito na plataforma tinkercad.

Figura 8: Circuito no Tinkercad, caso queira ver como deve funcionar seu circuito.

O PROGRAMA

Abaixo, você vê o programa. Procure digitar tudo, prestando atenção nas letras quando maiúscula e quando minúscula. É muito importante que você digite todo o código, exceto os comentários, pois isso te ajudará muito no aprendizado.

```
void setup(){
  //configurando as portas de saída
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
}
void loop(){
  //ligando o LED verde
  digitalWrite(10, HIGH);
  //aguardando 5 segundos
  delay(5000);
  //desligando o sinal verde
  digitalWrite(10,LOW);
  //ligando o sinal amarelo
  digitalWrite(11, HIGH);
  //aguardando dois segundos
  delay(2000);
  //desligando o sinal amarelo
  digitalWrite(11,LOW);
  //ligando o sinal vermelho
  digitalWrite(12,HIGH);
  //aguardadando 5 segundos
  delay(5000);
  //desligando o sinal vermelho
  digitalWrite(12,LOW);
  //agora o programa começara de novo
}
```

TERMINOU?

Se você terminou, vamos entender um pouco sobre imprimir dados via porta serial no computador, isto é, como se faz para que o Arduino envie dados para o PC.

Para isso devemos iniciar uma comunicação serial, que é apresentada no código a seguir. Note que é o mesmo que o código anterior, com alguns acréscimos.

```
void setup(){
  //configurando as portas de saída
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);

  //aqui dizemos para o arduino
  //iniciar a comunicação com o PC
  //usando uma taxa de 9600 bits por segundo
  Serial.begin(9600); //dizemos que a
comunicação serial foi iniciada

}
void loop(){
  //ligando o LED verde
  digitalWrite(10, HIGH);

  //para mostrar algo no monitor serial
usamos a função
  //print (para não dar um "enter" no final
da linha) ou
  //println (para dar um "enter" no final da
linha"
  Serial.print("Status atual do semáforo: ");
  Serial.println("Verde.");

  //aguardando 5 segundos
  delay(5000);
  //desligando o sinal verde
  digitalWrite(10,LOW);
  //ligando o sinal amarelo
  digitalWrite(11, HIGH);

  //Informamos que o sinal amarelo está
ligado
  Serial.print("Status atual do semáforo: ");
  Serial.println("Amarelo.");

  //aguardando dois segundos
  delay(2000);
  //desligando o sinal amarelo
  digitalWrite(11,LOW);
  //ligando o sinal vermelho
  digitalWrite(12,HIGH);

  //Informamos que o sinal vermelho está
ligado
  Serial.print("Status atual do semáforo: ");
  Serial.println("Vermelho.");

  //aguardadando 5 segundos
  delay(5000);
  //desligando o sinal vermelho
  digitalWrite(12,LOW);
  //agora o programa começara de novo
}
```

Se você conseguiu chegar aqui sem nenhum problema, acredito que chegou a hora de praticarmos um pouco mais a programação. Vamos começar entendendo o que é a tabela verdade.

TABELA VERDADE

Acredite, esse é um assunto bem intuitivo e você pode tentar entender isso brincando. Seu professor, e alguns colegas, criaram um site bem simples no qual você pode aprender um pouco de lógica de programação, mais precisamente, o que são conectivos lógicos, na forma de um jogo. Para ver como é, basta acessar o site abaixo ou ler o QR-code a seguir.

https://logicando.herokuapp.com/Apresentacao

Figura 9: Site onde você pode ter alguma ideia sobre lógica de programação na forma de um jogo.

Legal, mas o que é tabela verdade e o que são conectivos lógicos?

Basicamente, conectivos lógicos são símbolos ou palavras que conectam duas ou mais sentenças resultando em apenas dois possíveis resultados: **verdadeiro** ou **falso**.

Em linguagem de programação, temos os seguintes conectivos e seus respectivos símbolos na linguagem C++ (a que estamos usando para controlar o Arduino):

- E: &&
- OU: ||
- NÃO: !

Vamos ver então como usá-los de forma simples:

(O céu é azul) E (A lousa é verde) = VERDADEIRO
(Borboletas voam) OU (Gatos vivem no mar) = VERDADEIRO
(Borboletas voam) E (Cachorros vivem no mar) = FALSO

Agora podemos construir a tabela verdade. Faremos isso usando uma afirmação A, que pode ser verdadeira ou falsa (usaremos V ou F, respectivamente), e uma afirmação B, que também pode ser V ou F. Vamos utilizar os símbolos usados na programação do Arduino.

Tabela 1: tabela verdade preenchida junto com o professor durante a aula

A	B	A && B	A || B	!A	!B
V	V				
V	F				
F	V				
F	F				

Pegou a ideia?

Supondo que não haja nenhum problema aqui, vamos ver como usar isso no Arduino fazendo uso do comando *if*, que é traduzido simplesmente como “se”. Basicamente, se uma afirmação for verdadeira, ele executa uma ação.

Vamos adicionar o código abaixo no nosso programa e fazer alguns testes. Lembre-se de inseri-lo logo acima do último “}”:

```
//código inserido no final da aula 06
 if(3 > 2 && 5 > 1){
   Serial.println("Verdadeiro");
 }
```

Veja que no monitor serial, a cada 12 segundos, será impressa a palavra “Verdadeiro.”. Tente substituir o que está entre parêntesis por, incluindo os parêntesis, por:

(!(2 > 3))
(3 < 1 || 10 > 2)
(!(3 < 1 && 10 > 2))
(!(3 < 1 && 10 > 2) || (10 < 20))

Note que em todos os casos acima temos como resultado algo que é verdadeiro.

CONDICIONAIS

Em linguagem de programação, temos os conficionais *if* e *else* que são muito importantes em linguagem C++. A forma de trabalhar aqui é muito simples:

```
if(condição_a_ser_avaliada){
  faça_isso_se_verdadeiro
}
else{
  faça_isso_se_falso
}
```

Uma tradução direta seria:

```
se(condição_a_ser_avaliada){
  faça_isso_se_verdadeiro
}
se não{
  faça_isso_se_falso
}
```

Basicamente o programa executa uma ação se determinada condição for verdadeira e outra ação se tal condição for falsa. Veremos que esta condição pode ser o pressionar ou não um determinado botão.

Não se esqueça de entrar na página do seu professor para praticar um pouco o que você aprender e ter um pequeno spoiler da próxima aula.