

Mapeando transição de estados

Transcrição

Vamos retornar ao diagrama e compreender como mapear a transição. Nesse esquema temos diversas informações e uma grande parte delas já escrevemos no código fonte! É preciso ver o que ainda falta.

Por exemplo, o tamanho da sequência já está pronta e, inclusive, foi escrita de maneira a ter uma definição bem clara e expressiva! Devemos ter uma quantidade de rodadas que é menor que a sequência, afinal se temos 4 luzes temos uma quantidade de rodadas que aumenta até chegar a 4. Isso significa que precisamos saber qual é a rodada em questão.

Dessa maneira, acrescentamos ao diagrama, embaixo do `SETUP`, um comentário mencionando que essa rodada é igual a 0. No `PRONTO_PARA_PROX_RODADA` acrescentamos o comentário `rodada++` e substituímos o `Número de Luzes Respondidas` por `Rodada < Tamanho Sequencia`. Agora, podemos retornar ao código fonte e aprender como a variável `rodada` funciona e se relaciona.

O primeiro passo é criar uma variável e como ela é importantíssima no jogo todo, então, a colocaremos junto ao escopo global. Isto é, junto ao `int sequenciaLuzes(TAMANHO_SEQUENCIA)`. Portanto, vamos escrever `int rodada = 0;`:

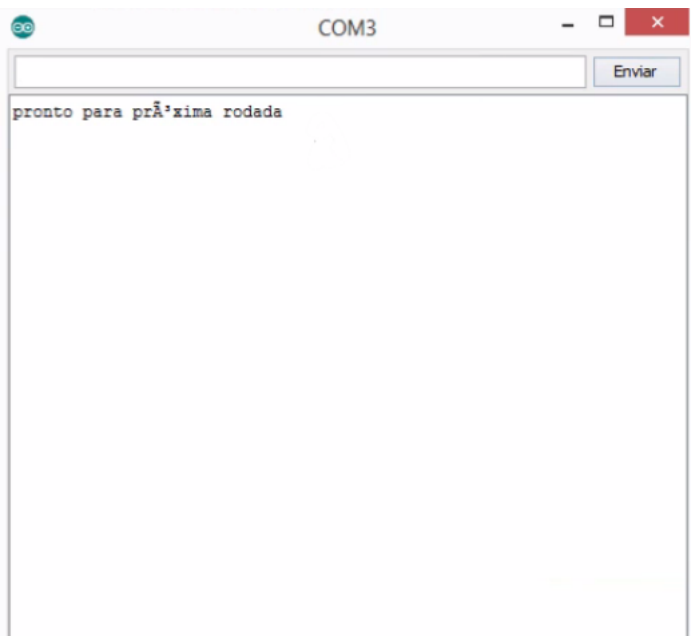
```
int sequenciaLuzes(TAMANHO_SEQUENCIA);

int rodada = 0;
```

Ao iniciar o jogo será perguntado ao `loop()` qual é o estado atual, no momento nós temos que é `USUARIO_RESPONDENDO`. Nós colocaremos que se a rodada é menor que o tamanho da sequência, assim, nós preenchemos `if(rodada < TAMANHO_SEQUENCIA)` e pedimos para retornar `PRONTO_PARA_PROX_RODADA`. E, ainda, acrescentamos um `else` seguido de um `return`. E apagamos o comentário `//Computar Estado Atual`:

```
int estadoAtual() {
    if(rodada < TAMANHO_SEQUENCIA) {
        return PRONTO_PARA_PROX_RODADA;
    }else{
        return JOGO_FINALIZADO_SUCESSO;
    }
}
```

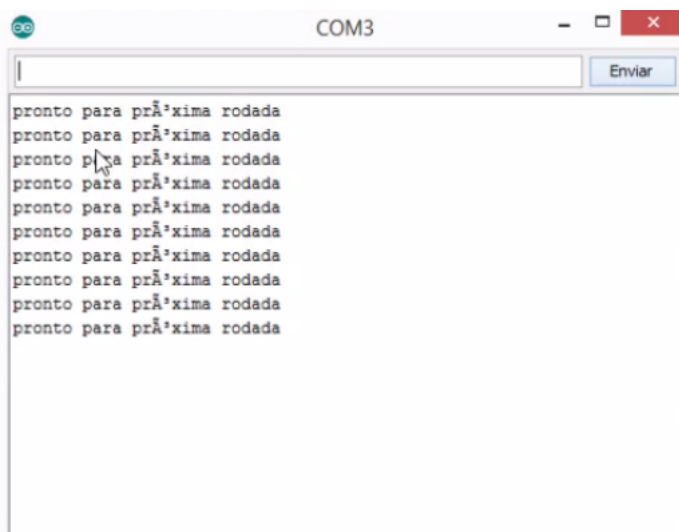
Vamos substituir o `for(;;)` por `delay` e ficaremos com o `delay(500)` para acompanhar melhor no `serial`. Podemos verificar se fica tudo certo e enviando para o **Arduino** teremos:



Ou seja, observando isso podemos concluir que ele nunca chega ao jogo finalizado com sucesso! Isso ocorre porque a rodada não está sendo incrementada, então, é preciso incrementar ela usando o `rodada++` :

```
void loop() {  
  switch(estadoAtual()) {  
    case PRONTO_PARA_PROX_RODADA;  
      Serial.println("pronto para próxima rodada");  
      rodada++;  
      break;  
    case USUARIO_RESPONDENDO;  
      Serial.println("usuário respondendo");  
      break;  
    case JOGO_FINALIZADO_SUCESSO;  
      Serial.println("jogo finalizado sucesso");  
      break;  
    case JOGO_FINALIZADO_FALHA;  
      Serial.println("jogo finalizado falha");  
      break;  
  }  
}
```

E acompanhando isso no monitor serial podemos verificar o que acontece:



Agora temos algo que é mais próximo do que queríamos!

Não vamos colocar o código de próxima rodada dentro disso, pois, ele ficará sujo e difícil de ler. Por isso, criamos uma função, a `preparaNovaRodada()` que acrescentamos junto de:

```
void loop() {  
  switch(estadoAtual()) {  
    case PRONTO_PARA_PROX_RODADA;  
      Serial.println("pronto para próxima rodada");  
      preparaNovaRodada();  
      rodada++;  
      break;  
  
      //...  
  
  }  
}
```

E, agora, podemos criar a função, `void preparaNovaRodada()` e dentro disso colocamos o `rodada++` que nós apagamos do `loop()`.

```
void loop() {  
  switch(estadoAtual()) {  
    case PRONTO_PARA_PROX_RODADA;  
      Serial.println("pronto para próxima rodada");  
      preparaNovaRodada();  
      break;  
  
      //...  
  
  }  
}
```

Por fim, adicionamos um `tocaLedsRodada`. Isso tudo deve ficar em baixo do `delay` e teremos:

```
void preparaNovaRodada(){  
  rodada++;  
  tocaLedsRodada();  
}
```

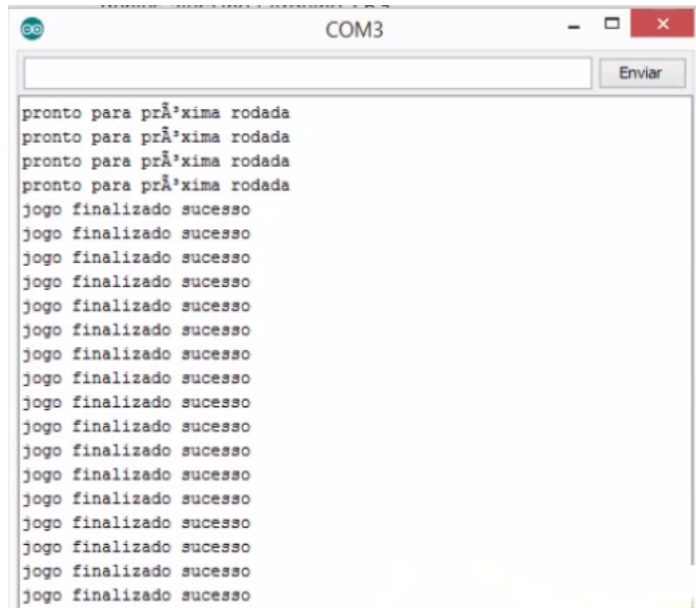
E testando isso verificamos que realmente funciona!

Toda vez que o `tocaLedsRodada()` for chamado, ele toca até o tamanho da sequência, mas, na verdade, queremos que o a sequência seja aumentada! Para arrumar isso basta modificar o `void tocaLedsRodada` para que o índice seja `< rodada`

```
void tocaLedsRodada() {  
  for(int indice = 0; indice < rodada; indice++){  
    piscaLed(sequenciaLuzes[indice]);  
  }  
}
```

Podemos enviar isso para o Arduino e observar o que acontece!

Se analisarmos o que acontece é que uma cor é repetida, entretanto, nós conseguimos que a cada rodada os LEDs fossem aumentados! A mesma cor está sempre sendo reproduzida, mas agora conseguimos perceber que a sequência está sendo aumentada a cada rodada. Isso é fácil de visualizar através do *console*!



```
COM3
pronto para próxima rodada
pronto para próxima rodada
pronto para próxima rodada
pronto para próxima rodada
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
jogo finalizado sucesso
```