Atividades Práticas

Cabeçalho

Curso:	Ciência de Computação	**Turno:**	Diurno
Disciplina:	Microcontroladores	**Professor:**	Átila Carvalho Junior
Aluno:	João Vítor Fernandes Dias	**Matrícula:**	00119110377
Atividade:	Atividades Práticas	**Site usado:**	Tinkercad

Atividade Prática 1

Código Geral

```
int N_Atividade = 7;

void selectedSetup (int N_Atividade) {

 Serial.begin(9600);
 switch (N_Atividade) {
  case 3: {}
  case 4: {}
  case 5: {pinMode(13, OUTPUT);} break;
  case 6: {
   pinMode(12, OUTPUT);
   pinMode(13, OUTPUT);
   digitalWrite(12,  HIGH);
   digitalWrite(13,  HIGH);
  } break;
  case 7: {
   pinMode(12, OUTPUT);
   pinMode(13, OUTPUT);
   digitalWrite(12,  LOW);
   digitalWrite(13,  LOW);
  } break;
  default: {
   Serial.println("Opcao invalida");
  } break;
 }
}

 void selectedLoop (int N_Atividade) {
 switch (N_Atividade) {
  case 3: {
   digitalWrite(13, HIGH);
```

```
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
 } break;
 case 4: {
  int t = 1000;
  digitalWrite(13, HIGH);
  delay(t);
  digitalWrite(13, LOW);
  delay(t);
 } break;
 case 5: {
  digitalWrite(13, HIGH);
  delay(2000);
  digitalWrite(13, LOW);
  delay(500);
 } break;
 case 6: {
  /*
        É pedido para que
    "
     quando o primeiro LED se apagar, o novo LED
     imediatamente se acenda e vice-versa
    "
    A forma que eu fiz abaixo não faz exatamente isso por
    não ser imediato.
    Para que fosse mais imediato, precisaria que fosse
    utilizado um comparador lógico. Mas não lembro
    exatamente como usar.
    Até tentei, mas não consegui e desisti

    OPA, consegui pensar melhor. precisa de comparador não
    Tá resolvido
  */
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13,  LOW);
  delay(1000);
 }; break;
 case 7: {

  digitalWrite(13, HIGH);       //0/1/0
  delay( 300);                 //0/1/0 L  300
  digitalWrite(13,  LOW);       //0/0/0
  delay( 500);                 //0/0/0 XX 500
  digitalWrite(12,  HIGH);      //0/0/1
  delay(1000);                  //0/0/1 R 1000
  digitalWrite(12,  LOW);       //0/0/0
  delay(1000);                  //0/0/0 XX 500

 }; break;
 default: {
```

```
    while (1) {
      Serial.println("->Escolha uma atividade valida \
no codigo e tente novamente<-");
      delay(3000);
    }
  };break;
 }
}

void setup(){selectedSetup(N_Atividade);}
void loop(){selectedLoop (N_Atividade);}
```

1. Acenda um LED

Descrição

Faça o acendimento de um LED. Para isso:

a) conecte a placa Arduino na entrada USB do computador;

b) conecte o polo positivo do LED na saída +5V (Vcc) do Arduino;

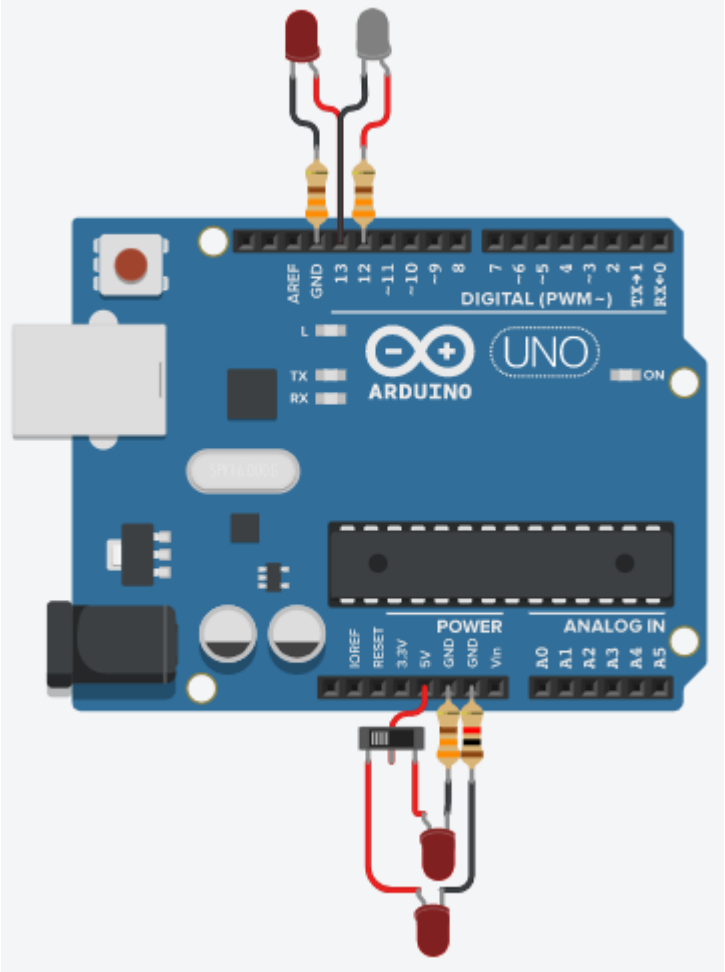
c) conecte o polo negativo do LED a um terminal do resistor de 220Ω (ou 330Ω);

d) conecte o outro terminal do resistor ao GND (Terra) do Arduino e observe o acendimento do LED;

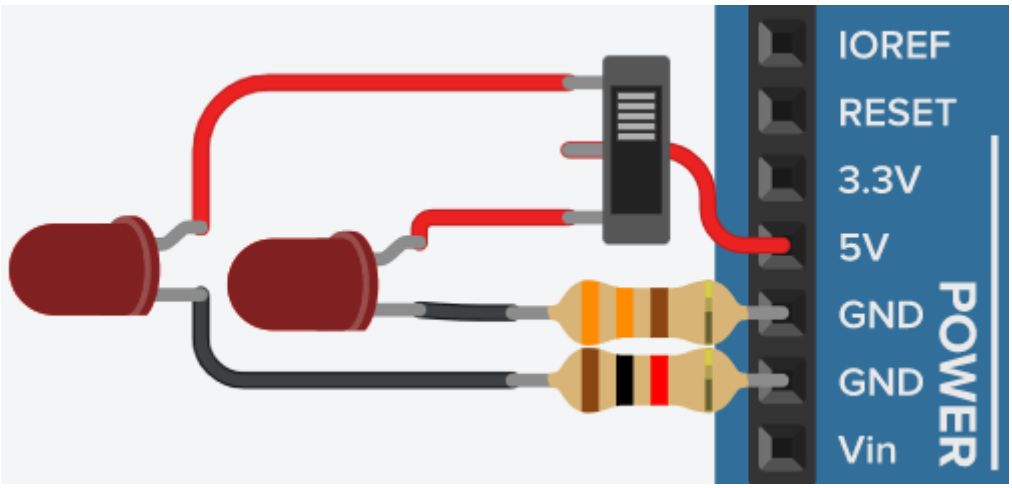
e) Repita os itens a) – d), mas usando um resistor de 1kΩ. Verifique em qual montagem o LED vai acender mais intensamente.

Imagem

Completo



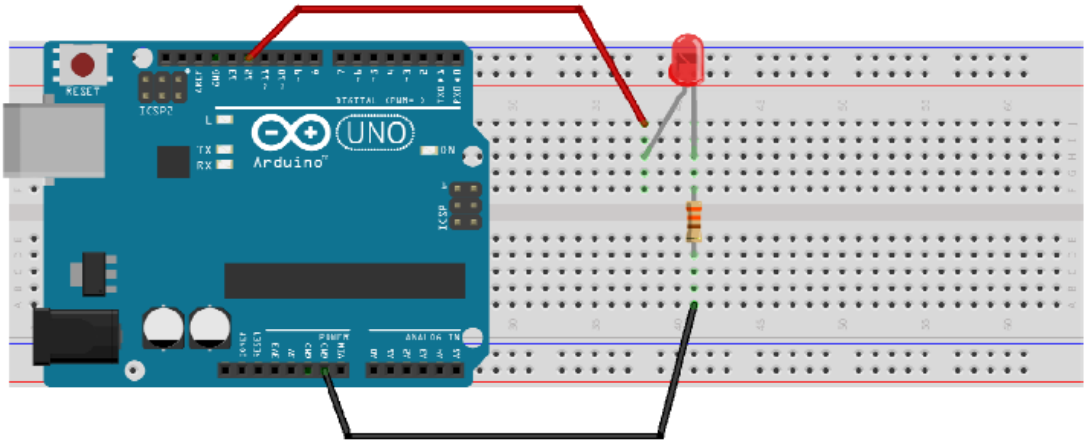
Específico



2. Monte o circuito

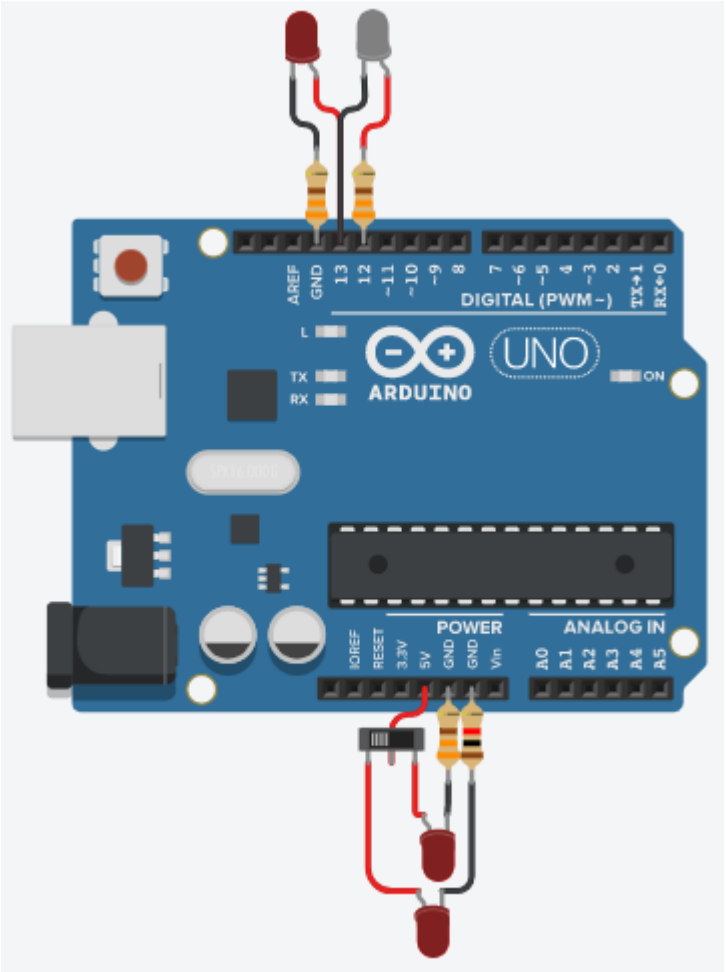
Descrição

Monte o circuito mostrado na figura a seguir. Lembre-se de verificar a correta polarização do diodo.

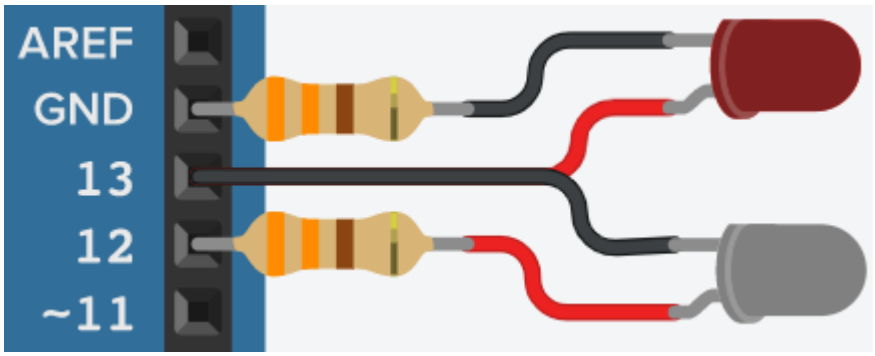


Imagem

Completo



Específico



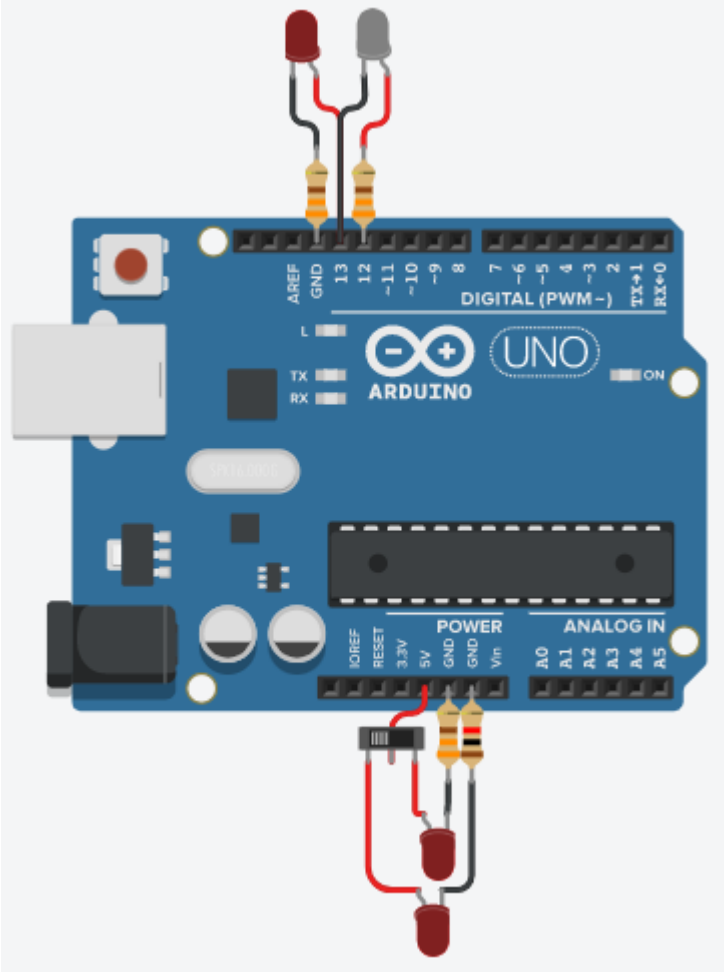
3. Pisque o LED a cada 1 segundo

Descrição

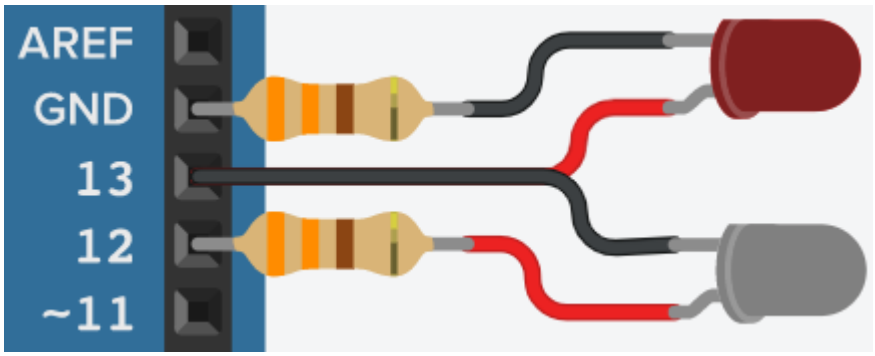
Na interface de programação de *Sketches* do Arduino, insira e compile um programa que faça o LED piscar a cada 1 segundo.

Imagem

Completo



Específico



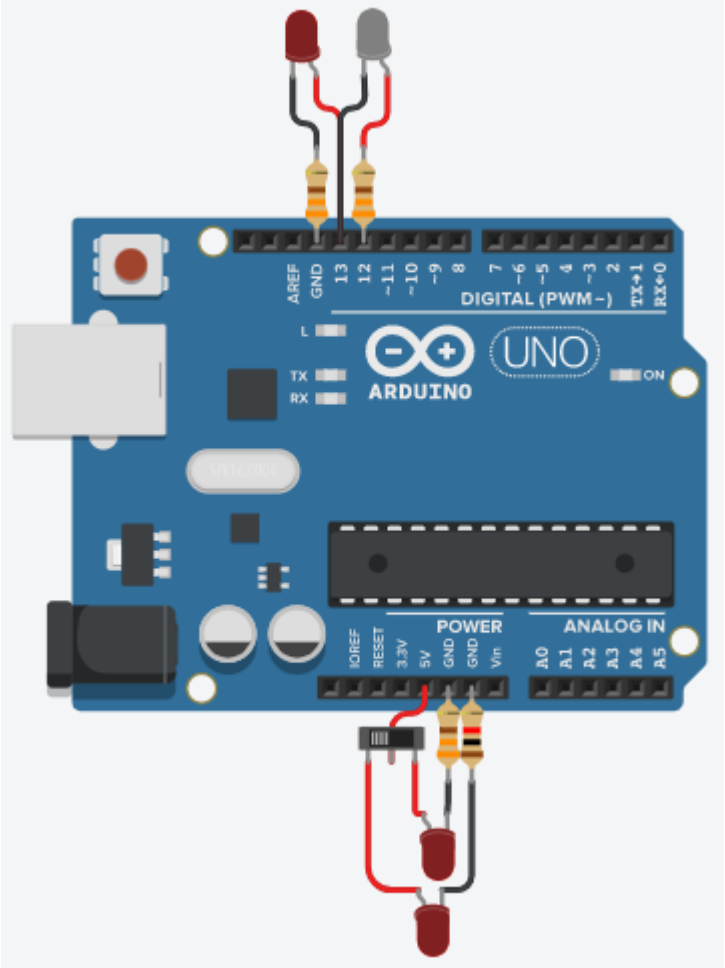
4. Pisque o LED por um tempo t

Descrição

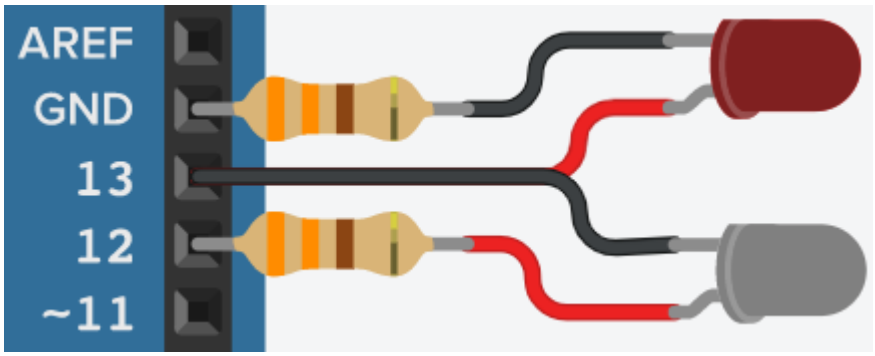
Altere o tempo de funcionamento do LED criando uma variável “t” para a temporização.

Imagem

Completo



Específico



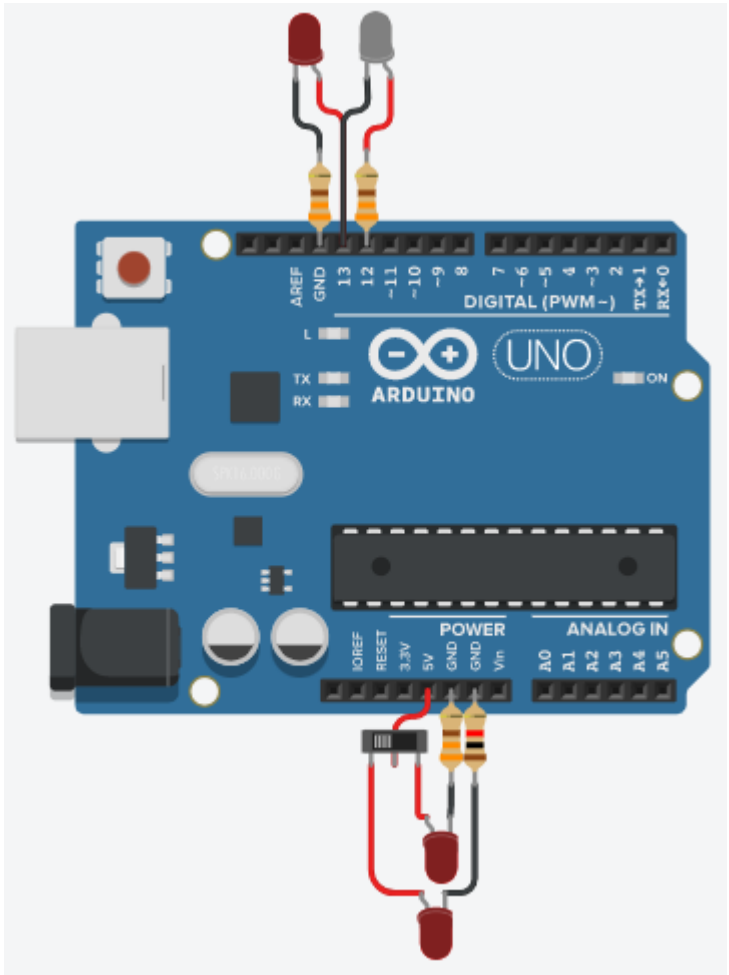
5. Ligado: 2 segundos; Desligado: 0,5 segundos

Descrição

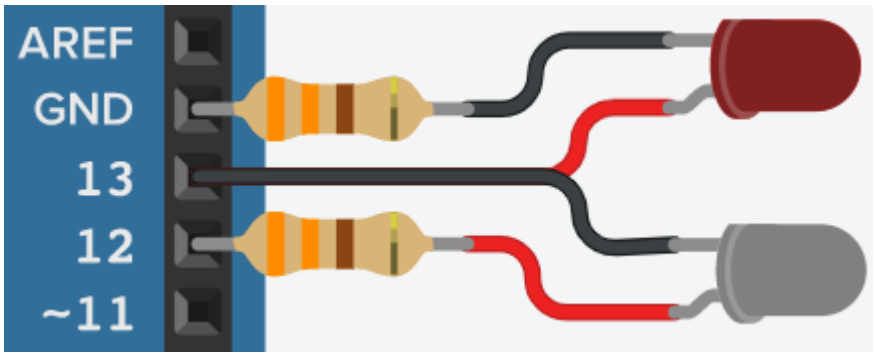
Altere o tempo de *delay* para 2 segundos com LED ligado e 0,5 segundo para LED desligado.

Imagem

Completo



Específico



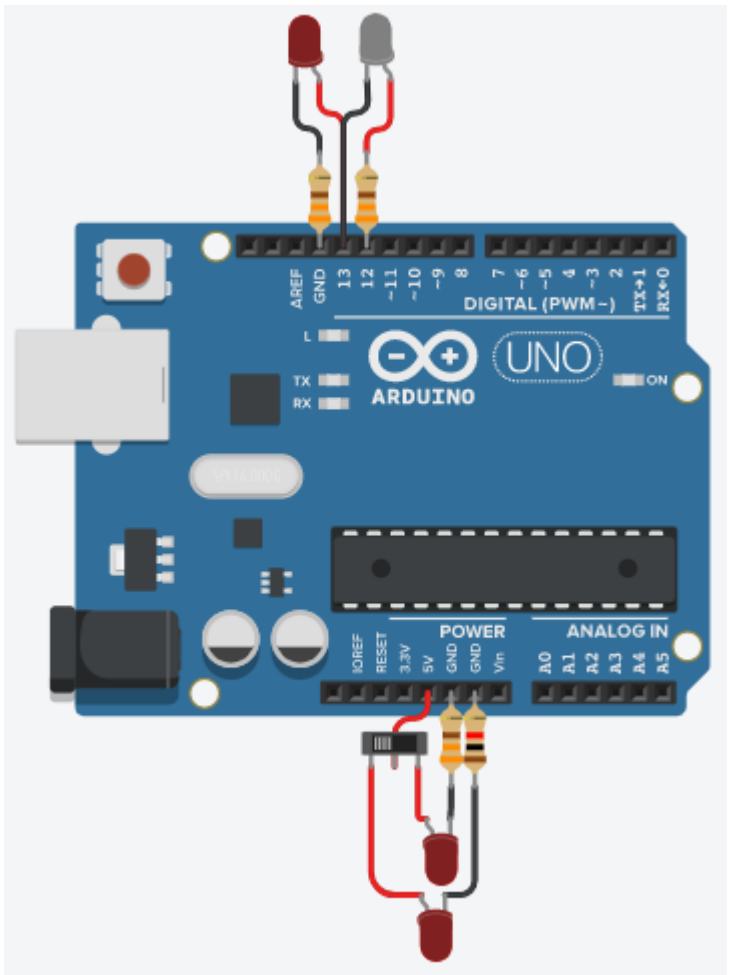
6. Alternar entre LEDs

Descrição

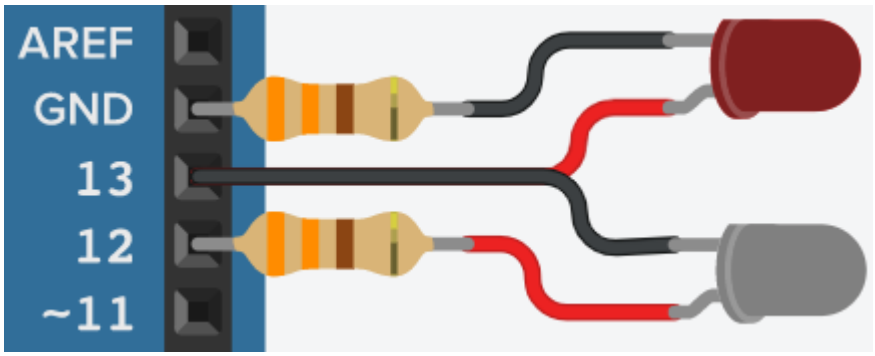
Inclua um novo LED que funcione na saída número 13 da placa do Arduino (sem retirar o anterior). Faça um novo programa em que os 2 LEDs funcionem alternadamente, de modo que quando o primeiro LED se apagar, o novo LED imediatamente se acenda e vice-versa. Cada um permanecerá ligado por 1 segundo.

Imagem

Completo



Específico



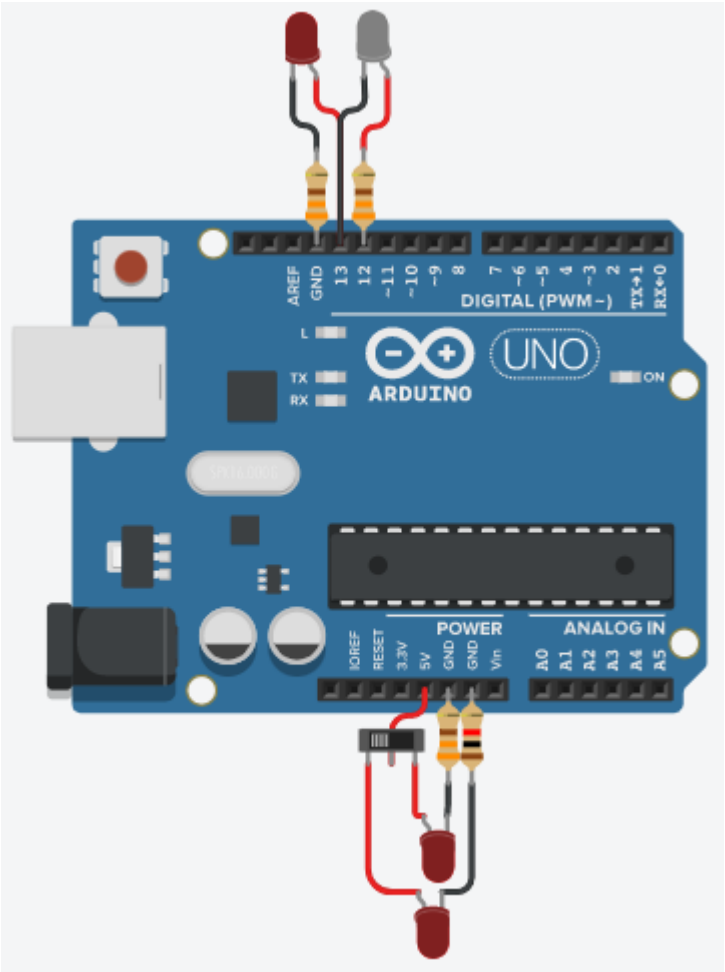
7. Alternar entre LEDs por tempos diferentes

Descrição

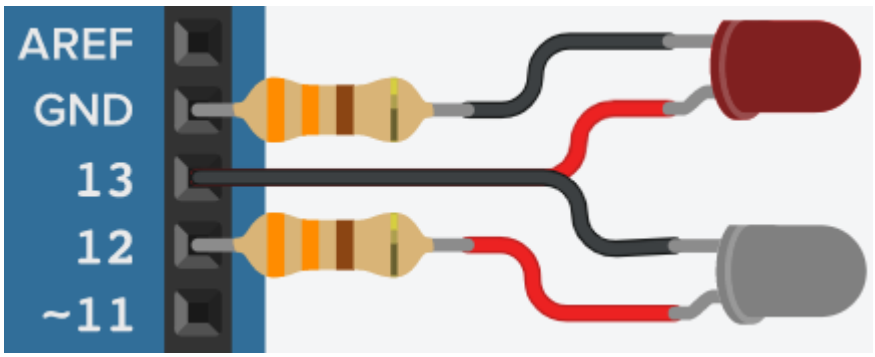
Faça um novo programa alterando o tempo de acendimento de cada LED, de tal modo que o primeiro esteja ligado durante 0,3 segundo; o segundo LED durante 1 segundo; e que haja um tempo de 0,5 segundo de *delay* antes de acender o LED seguinte.

Imagem

Completo



Específico



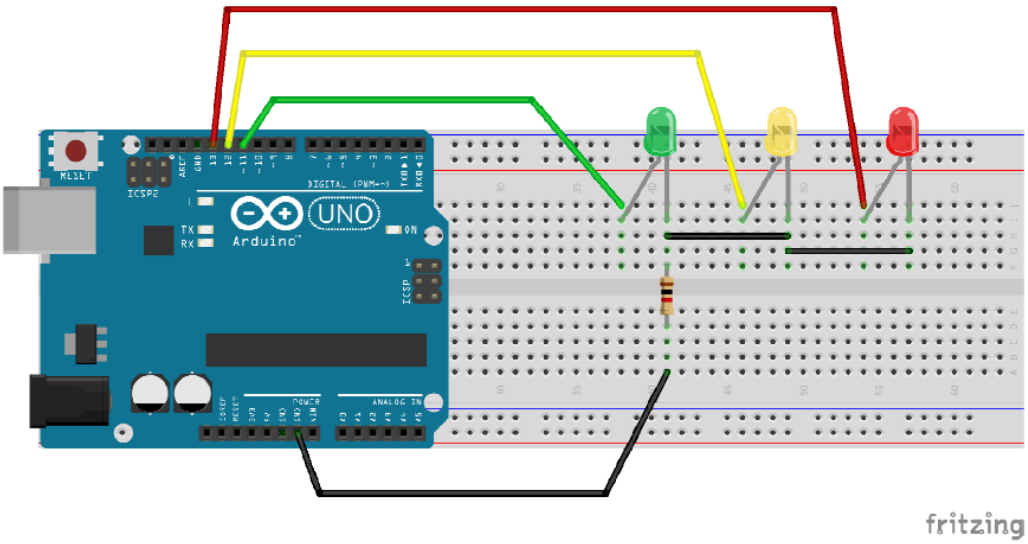
Atividade Prática 2

1) Circuito inicial proposto

Descrição

Monte o circuito mostrado na figura abaixo. Lembre-se de verificar a correta polarização dos diodos. O resistor de 1kΩ, como mostrado, atende aos três diodos alternadamente (um por vez). Por isso, não há problema de sobrecarga.

Imagem

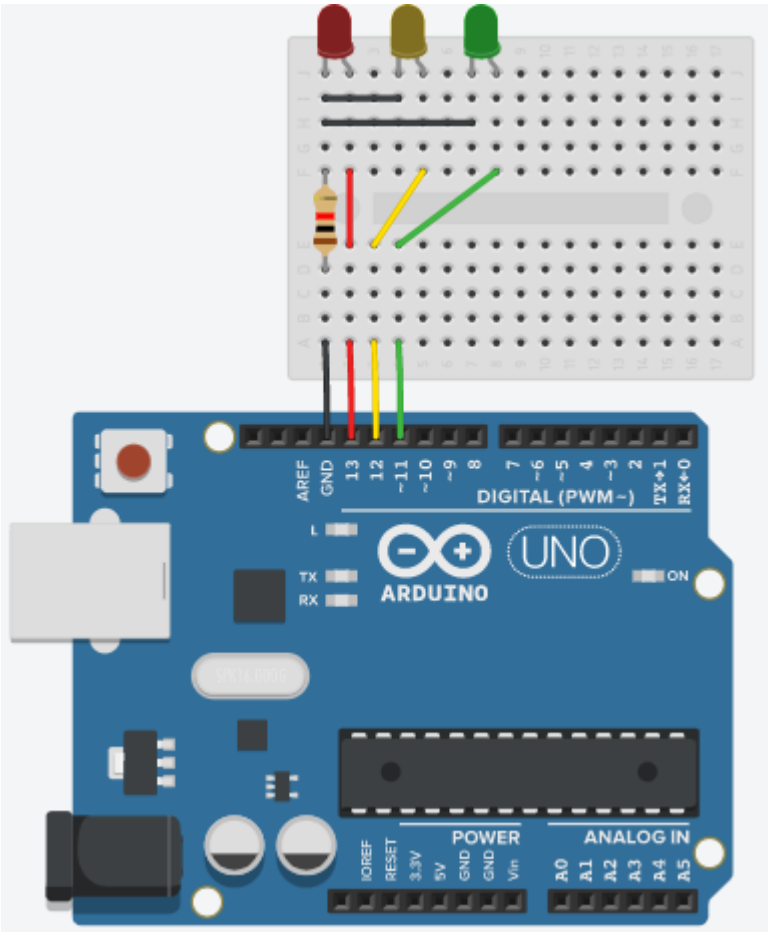


2) Semáforo

Descrição

Na interface de programação de *Sketches* do Arduino, insira e compile um programa que simule o funcionamento de um semáforo. Os LEDs vermelho e verde devem ficar acesos por 6 segundos e o amarelo por 2 segundos.

Imagem



Código

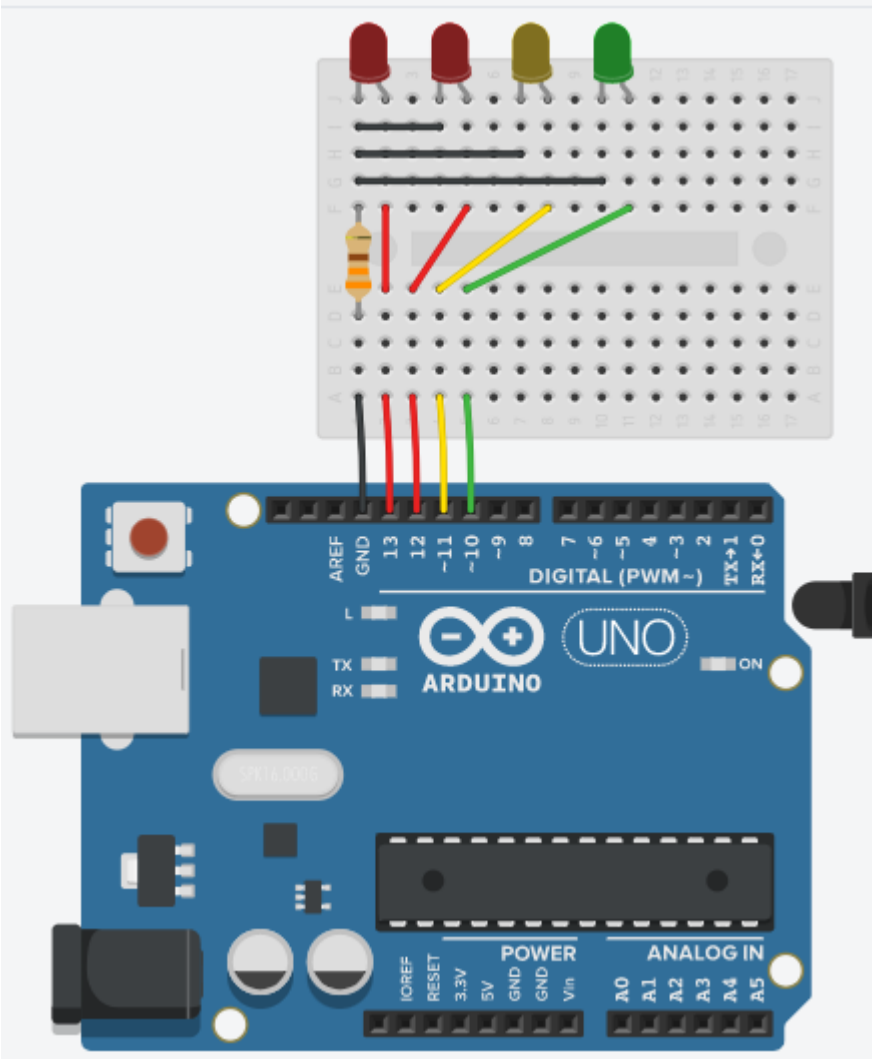
```
int red = 13, yellow = 12, green = 11 ;
void setup() {
  pinMode(red,    OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green,  OUTPUT);
}
void ligarLedporXSegundos (int led, int tempo) {
  digitalWrite(led, HIGH);
  delay(1000*tempo);
  digitalWrite(led, LOW);
}

void loop () {
  ligarLedporXSegundos(green,   6);
  ligarLedporXSegundos(yellow,  2);
  ligarLedporXSegundos(red,     6);
}
```

3) Semáforo escalonado

Descrição

Inclua agora mais 1 LED vermelho. Faça com que o sinal vermelho seja escalonado, ou seja, ao ser acionado o vermelho, os 2 LEDs se acendem simultaneamente, mas, antes de passar para o verde, apague um dos LEDs vermelhos e temporize 2 segundos antes de passar para o verde.

Imagem



Código

```
int red1 = 13, red2 = 12, yellow = 11, green = 10 ;

void setup() {
  pinMode(red1,    OUTPUT);
  pinMode(red2,    OUTPUT);
  pinMode(yellow, OUTPUT);
  pinMode(green,  OUTPUT);
}

void ligarLedporXSegundos (int led, int tempo) {
```

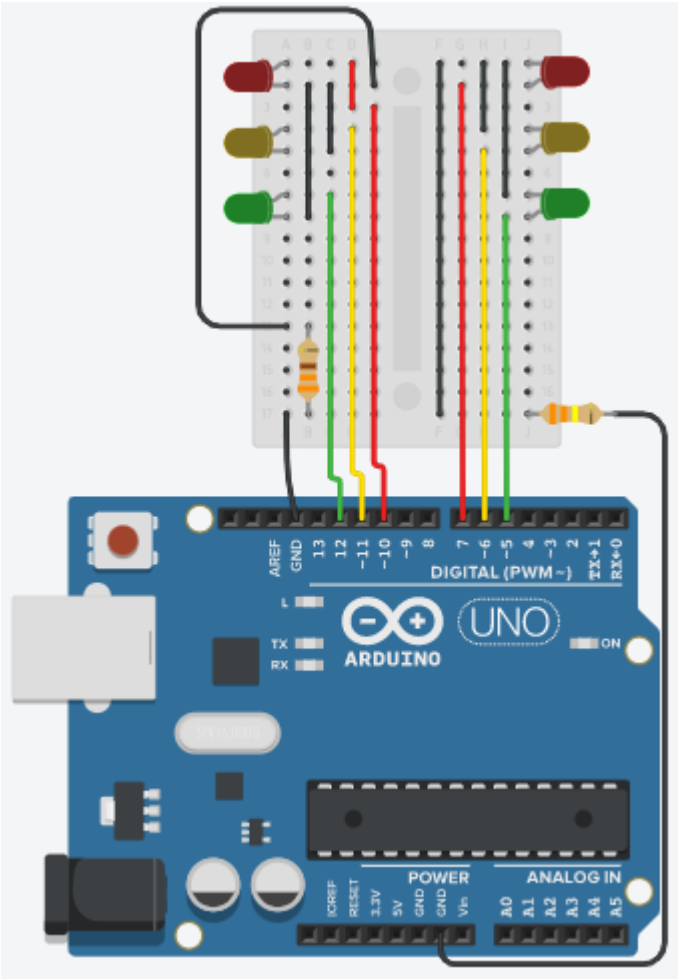
```
 digitalWrite(led, HIGH);
 delay(1000*tempo);
 digitalWrite(led, LOW);
}

void redEscalonado (int led1, int led2) {
 digitalWrite(led1, HIGH);
 digitalWrite(led2, HIGH);
 delay(4000);
 digitalWrite(led1, LOW);
 delay(2000);
 digitalWrite(led2, LOW);
}

void loop () {
 ligarLedporXSegundos(green,    6);
 ligarLedporXSegundos(yellow,   2);
 redEscalonado      (red1, red2  );
}
```

4) Semáforo intercalado

Descrição

Faça um projeto agora com 2 semáforos de 3 LEDs (vermelho, amarelo e verde) cada um. Simule a operação de um cruzamento com esses 2 semáforos, abrindo para uma rua e interrompendo a outra e vice-versa.

Imagem



Código

```
int red1 = 10, yellow1 = 11, green1 = 12 ;
int red2 =  7, yellow2 =  6, green2 =  5 ;

void setup() {
  pinMode(red1,    OUTPUT);
  pinMode(red2,    OUTPUT);
  pinMode(yellow1, OUTPUT);
  pinMode(yellow2, OUTPUT);
  pinMode(green1,  OUTPUT);
  pinMode(green2,  OUTPUT);
}

void liga    (int led) { digitalWrite(led, HIGH); }
void desliga (int led) { digitalWrite(led,  LOW); }

void loop () {
  liga    (green1 );
  liga    (red2   );
  delay   (4000   );
  desliga (green1 );
```

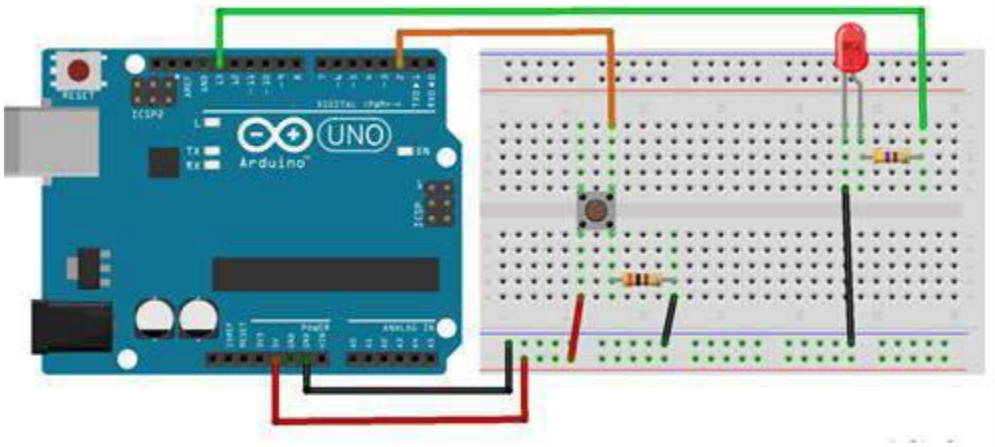
```
 liga    (yellow1);
 delay   (2000   );
 desliga (yellow1);
 desliga (red2   );
 liga    (red1   );
 liga    (green2 );
 delay   (4000   );
 liga    (yellow2);
 delay   (2000   );
 desliga (yellow2);
 desliga (red1   );
}
```

Atividade Prática 3

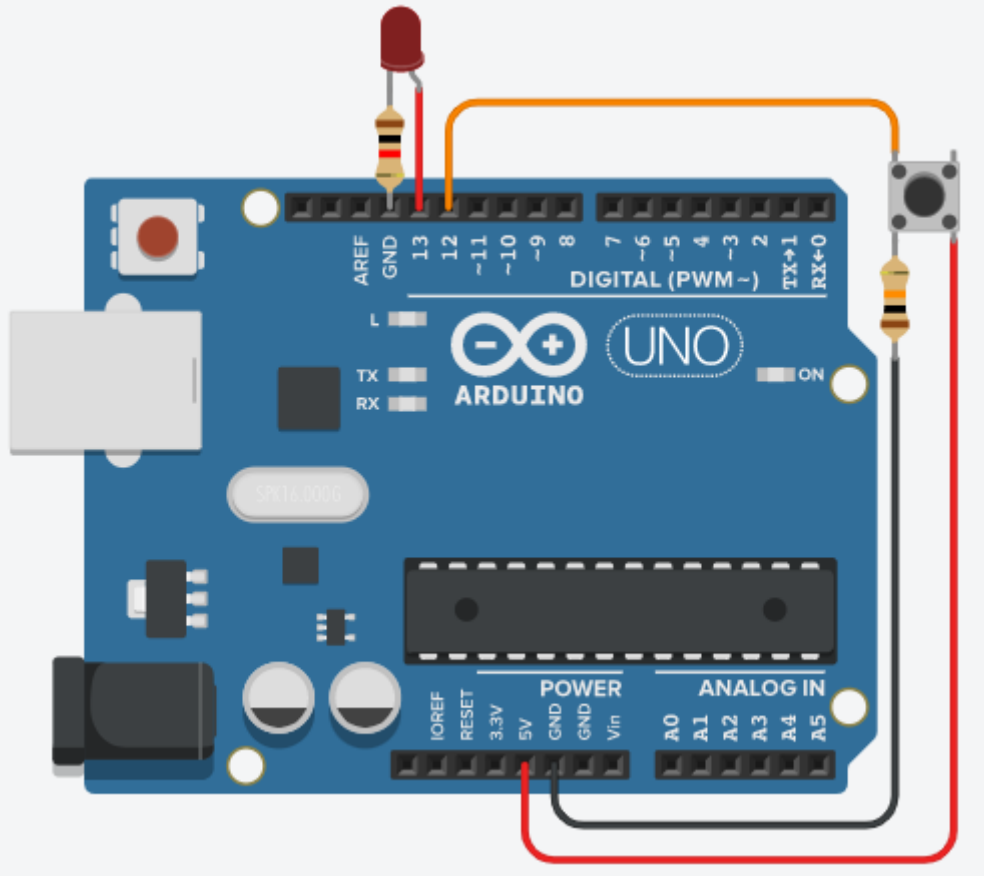
1- Monte o circuito

Descrição

Monte o circuito mostrado na figura abaixo. Lembre-se de verificar a correta polarização do diodo. Utilize um resistor em torno de 10kΩ para o botão e outro de até 1kΩ para o LED.



Imagem

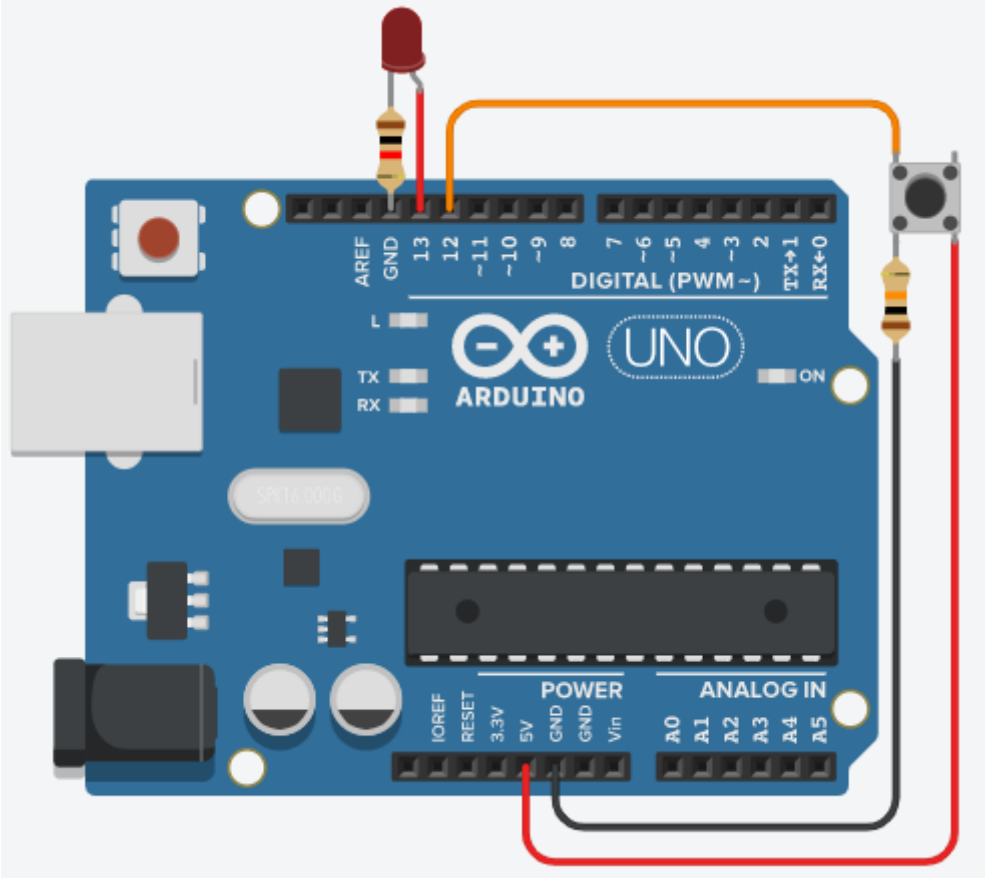


2- Acender LED com botão

Descrição

Na IDE do Arduino, insira e compile um programa que faça acender um LED ao apertar um botão. Lembrando que o botão será a entrada e o LED a saída.

Imagem



Programação

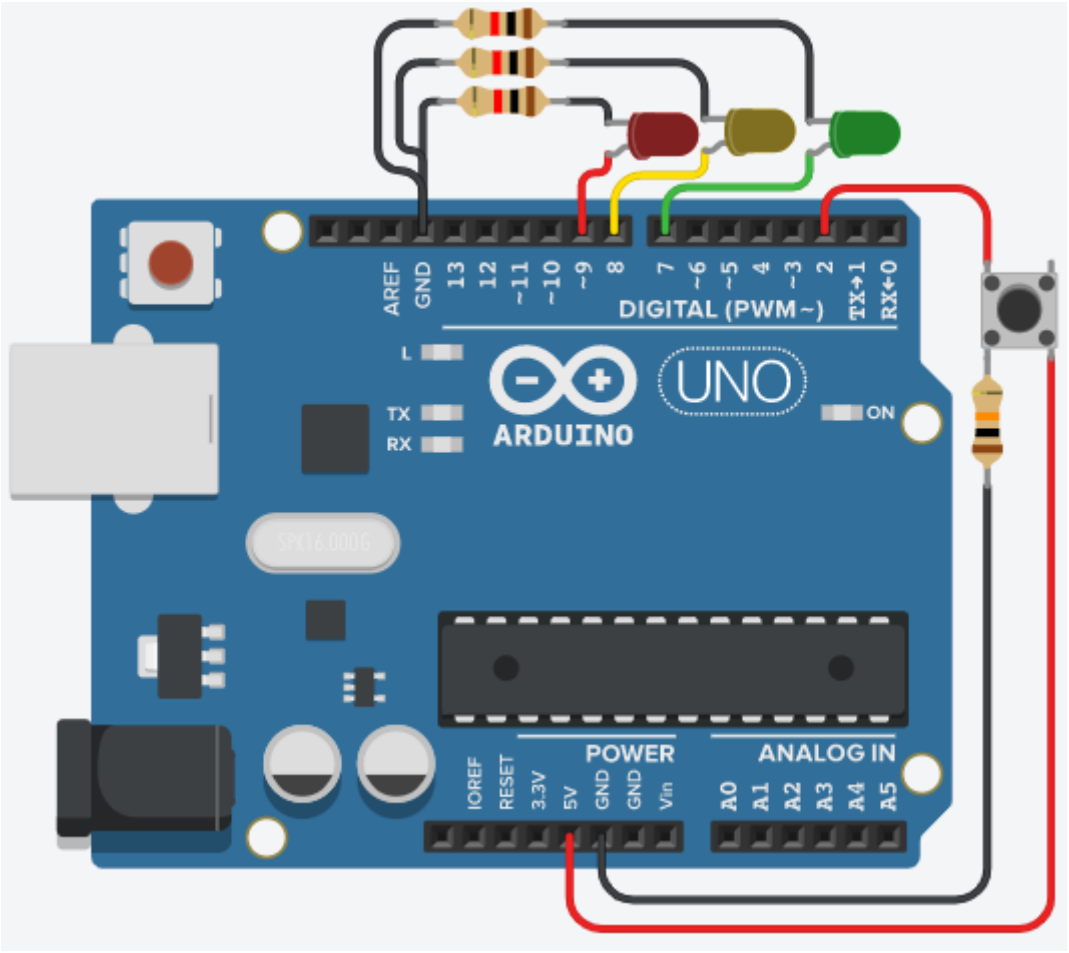
```
int led=13, input=12;
void setup () {
  pinMode (led, OUTPUT);
  pinMode (input,  INPUT);
}

void loop() {
  int val = digitalRead (input);
  if (val>0) {
        digitalWrite (led, HIGH);
  }
  digitalWrite (led, LOW);
}
```

3- Iniciar semáforo

Descrição

Utilize um botão para simular uma operação manual do semáforo da aula passada. Quando o botão for apertado, o semáforo temporiza 1 segundo antes de passar para o amarelo (onde permanece por 0,5 segundo) e depois para o vermelho (onde ficará 3 segundos antes de voltar para o verde).

Imagem



Programação

```
int red = 9, yellow = 8, green = 7 ;

int input=2;

void setup () {

  pinMode(red,    OUTPUT);

  pinMode(yellow, OUTPUT);

  pinMode(green,  OUTPUT);

  pinMode(input,  INPUT);

  Serial.begin (9600);

}
```

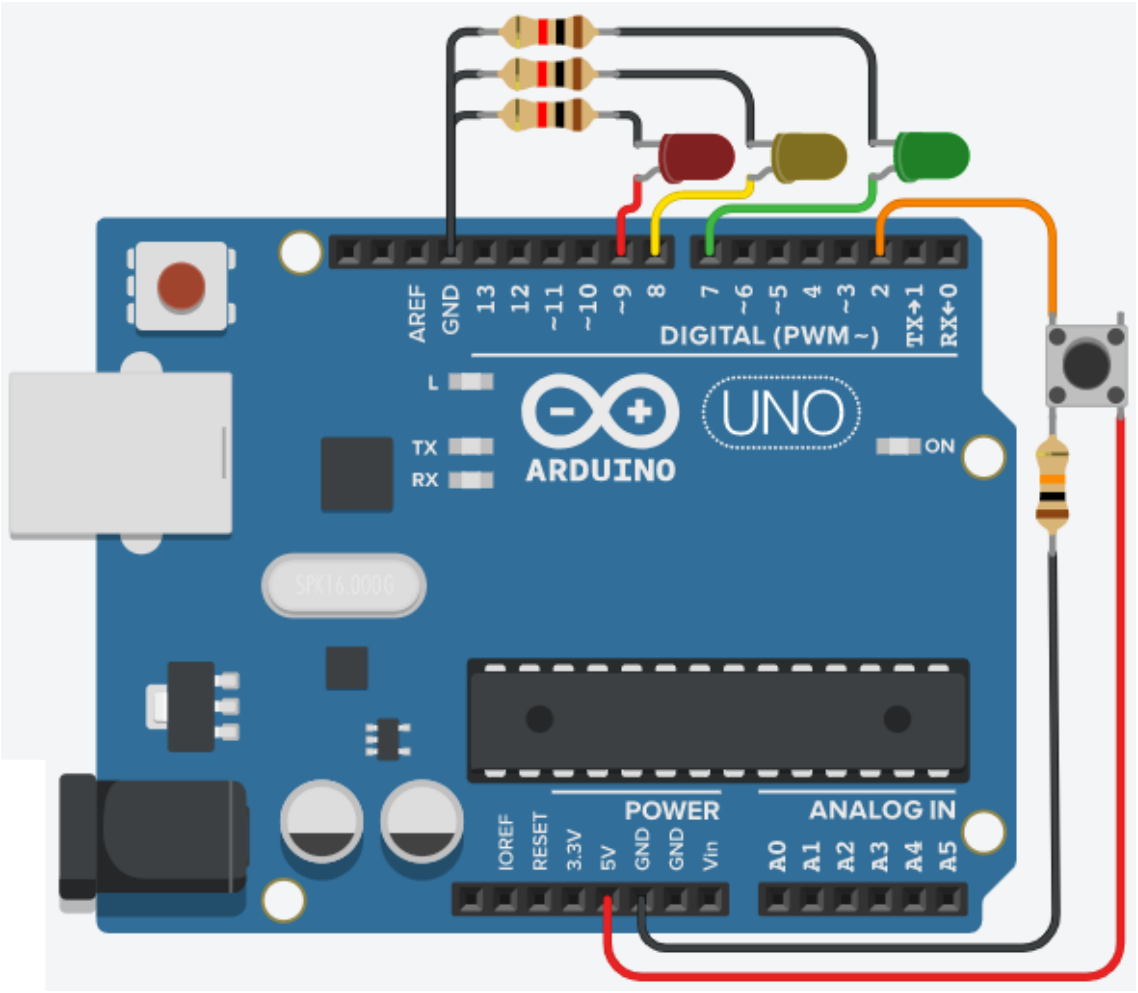
```
void ligarLedporXSegundos (int led, int tempo) {
  digitalWrite(led, HIGH);
  delay(1000*tempo);
  digitalWrite(led, LOW);
}

void ligarSemaforo () {
  ligarLedporXSegundos (green, 1);
  ligarLedporXSegundos (yellow, 0.5);
  ligarLedporXSegundos (red, 3);
}

void print (int val) {
  Serial.print(val);
  delay (1000);
}

void loop () {
  int val = 0;
  val = digitalRead(input);
  if (val>0) {
        ligarSemaforo ();
  }
}
```

4- Pausar semáforo

Descrição

Altere o projeto anterior da seguinte forma: O botão corresponde a um comando dado por um pedestre que deseja atravessar a rua. Ao aciona-lo, o semáforo, na próxima vez que passar ao vermelho, deve aumentar o tempo de vermelho em mais 4 segundos para permitir que o pedestre possa atravessar a rua.

Imagem



Programação

```
int red = 9, yellow = 8, green = 7 ;

int input=2;

void setup () {

 pinMode(red,    OUTPUT);
```

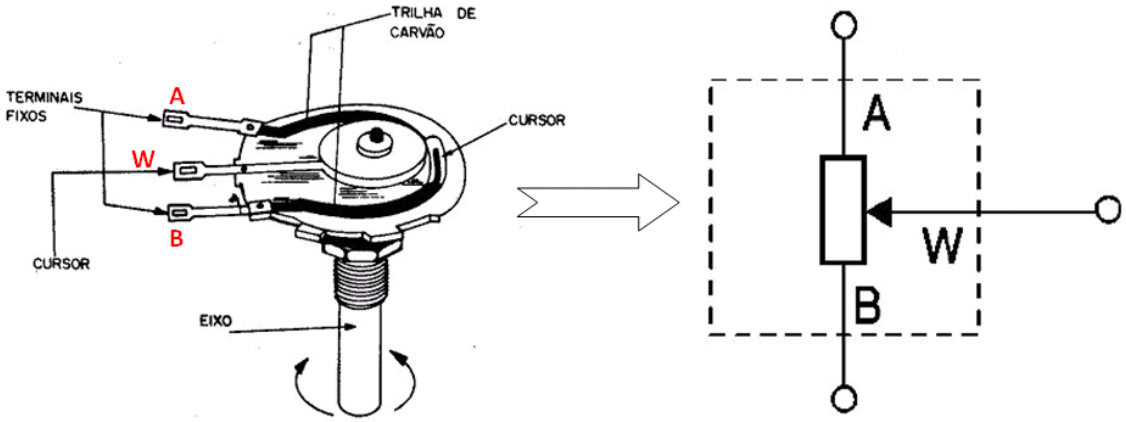
```
  pinMode(yellow, OUTPUT);
  pinMode(green,  OUTPUT);
  pinMode(input,  INPUT);
  Serial.begin (9600);
}

void ligarLedporXSegundos (int led, int tempo) {
  digitalWrite(led, HIGH);
  delay(1000*tempo);
  digitalWrite(led, LOW);
}

void ligarSemaforo (int delay) {
  ligarLedporXSegundos (green,    1);
  ligarLedporXSegundos (yellow, 0.5);
  ligarLedporXSegundos (red, 3+4*delay);
}

void loop () {
  ligarSemaforo(digitalRead(input));
}
```

Atividade Prática 4

Introdução teórica

Potenciômetro

Potenciômetros são resistores variáveis e ajustáveis, e por isso são usados para controle analógico de funcionalidades de alguns aparelhos eletrônicos, tal como o volume de um aparelho de som. Eles costumam possuir três terminais. Dois são ligados às extremidades da resistência (A e B) e a terceira a um cursor que anda de ponta a ponta da resistência (W).

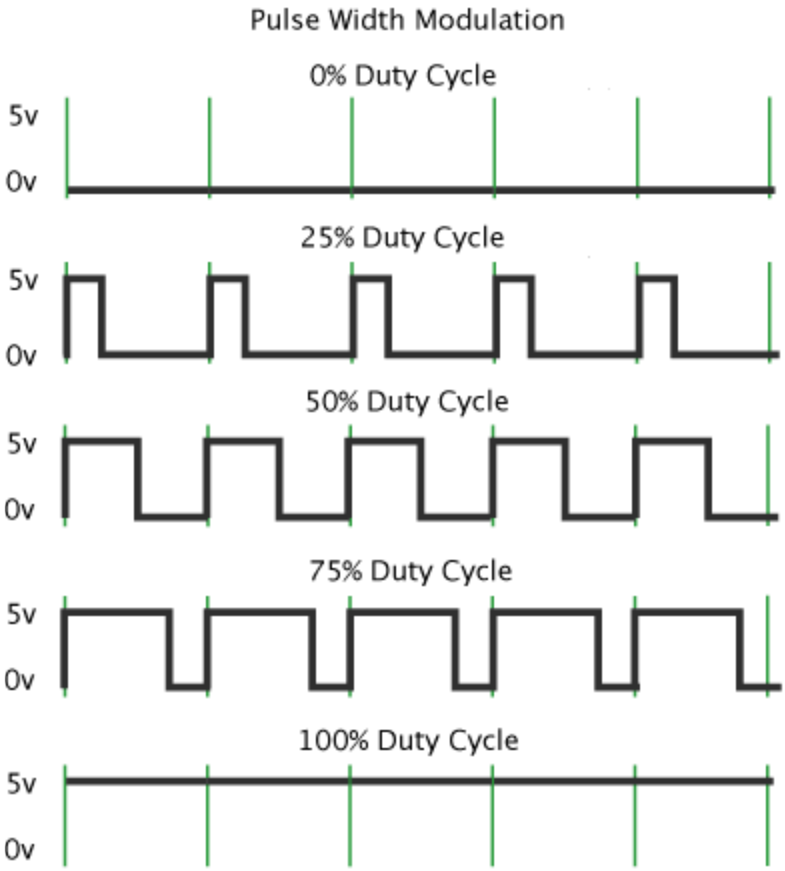


Leitura de Entrada Analógica

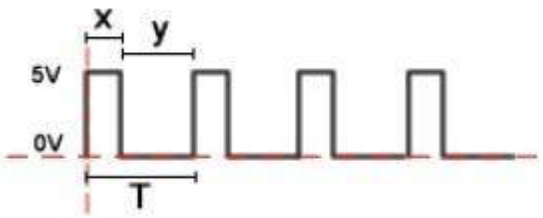
A leitura da entrada analógica é feita com a função analogRead, que recebe como parâmetro o pino analógico a ser lido e retorna o valor digital que representa a tensão no pino. Como o conversor analógico-digital do Arduino possui uma resolução de 10 bits, o intervalo de tensão de referência, que no nosso caso é 5 V, será dividido em 1024 pedaços (2^10) e o valor retornado pela função será o valor discreto mais próximo da tensão no pino.
Exemplo:
unsigned int valorLido = analogRead(A0);
O código acima lê o valor analógico de tensão no pino A0 e guarda o valor digital na variável valorLido. Supondo que o pino está com uma tensão de 2V, o valor retornado pela conversão será:
2 x 1024 / 5 = 409,6

PWM

PWM (Pulse Width Modulation – Modulação por Largura de Pulso) é uma técnica para obter resultados analógicos por meios digitais. Essa técnica consiste na geração de uma onda quadrada em uma frequência muita alta em que pode ser controlada a porcentagem do tempo em que a onda permanece em nível lógico alto. Esse tempo é chamado de Duty Cycle e sua alteração provoca mudança no valor médio da onda, indo desde 0V (0% de Duty Cycle) a 5V (100% de Duty Cycle) no caso do Arduino.



O Duty Cycle é a razão do tempo em que o sinal permanece na tensão máxima (5V no Arduino) sobre o tempo total de oscilação, como está ilustrado na figura abaixo:



O valor do Duty Cycle usado pelo Arduino é um inteiro armazenado em 8 bits, de forma que seu valor vai de 0 (0%) a 255 (100%).
Para escrever um sinal na saída PWM utiliza-se a função analogWrite, que recebe como parâmetros o pino PWM e o valor do duty cycle, respectivamente. Esse último parâmetro é guardado em 8 bits, de modo que esse valor deve estar entre 0 (0% de duty cycle) e 255 (100% de duty cycle).
analogWrite (9,127) ;//Escreve no pino 9 um sinal PWM com 50% de duty cycle // (50% de 255=127)
analogWrite (10,64) ;//Escreve no pino 10 um sinal PWM com 25% de Duty Cycle // (25% de 255=64)

Variando o duty cycle altera-se também o valor médio da onda, de modo que o efeito prático obtido com o PWM em algumas aplicações é um sinal com amplitude constante e de valor igual ao valor médio da onda. Isso permite que se possa, por exemplo,

controlar a intensidade do brilho de um LED, ou a velocidade de um motor de corrente contínua. Para variarmos o PWM, usamos o valor analógico lido no potenciômetro e armazenamos na variável valorLido, que armazena valores entre 0 e 1023. Porém, para que seja possível usar essa variável para controlar o PWM, devemos mudar sua escala para 0 a 255. Para isso usaremos uma função. Tal função recebe uma variável e muda sua escala.

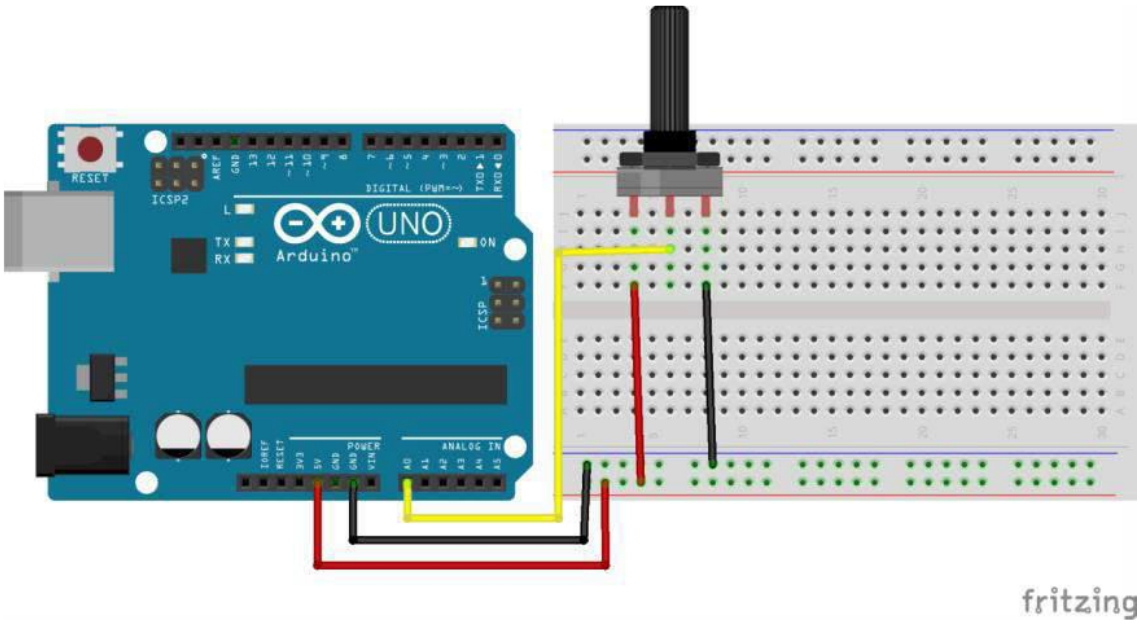
```
pwm = map (valorLido, 0, 1023, 0, 255);
// Mudança de escala
analogWrite (led, pwm);

//Escreve no led um sinal PWM proporcional ao valorLido
```

Roteiro

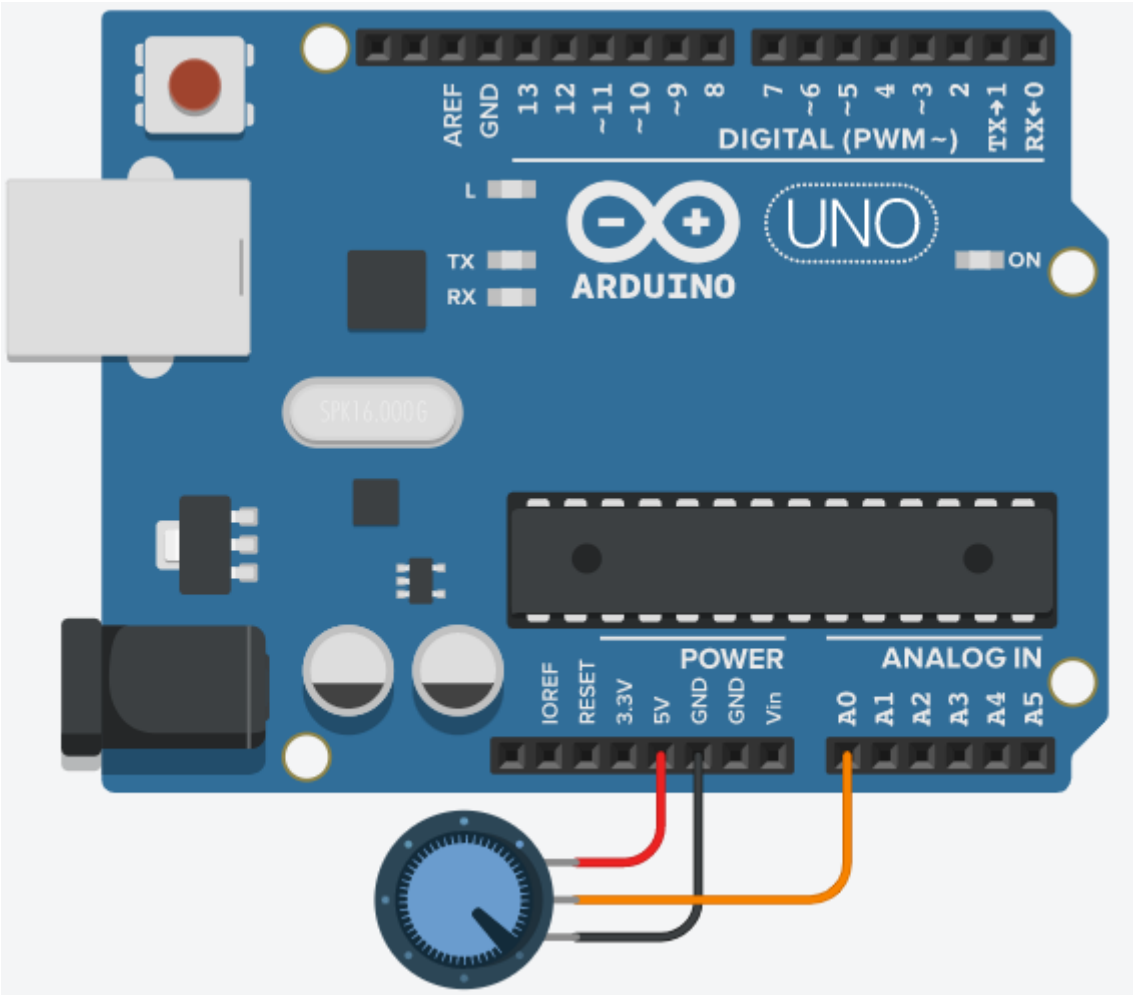
1- Montar circuito

Descrição

Monte o circuito conforme a figura a seguir:



Imagem

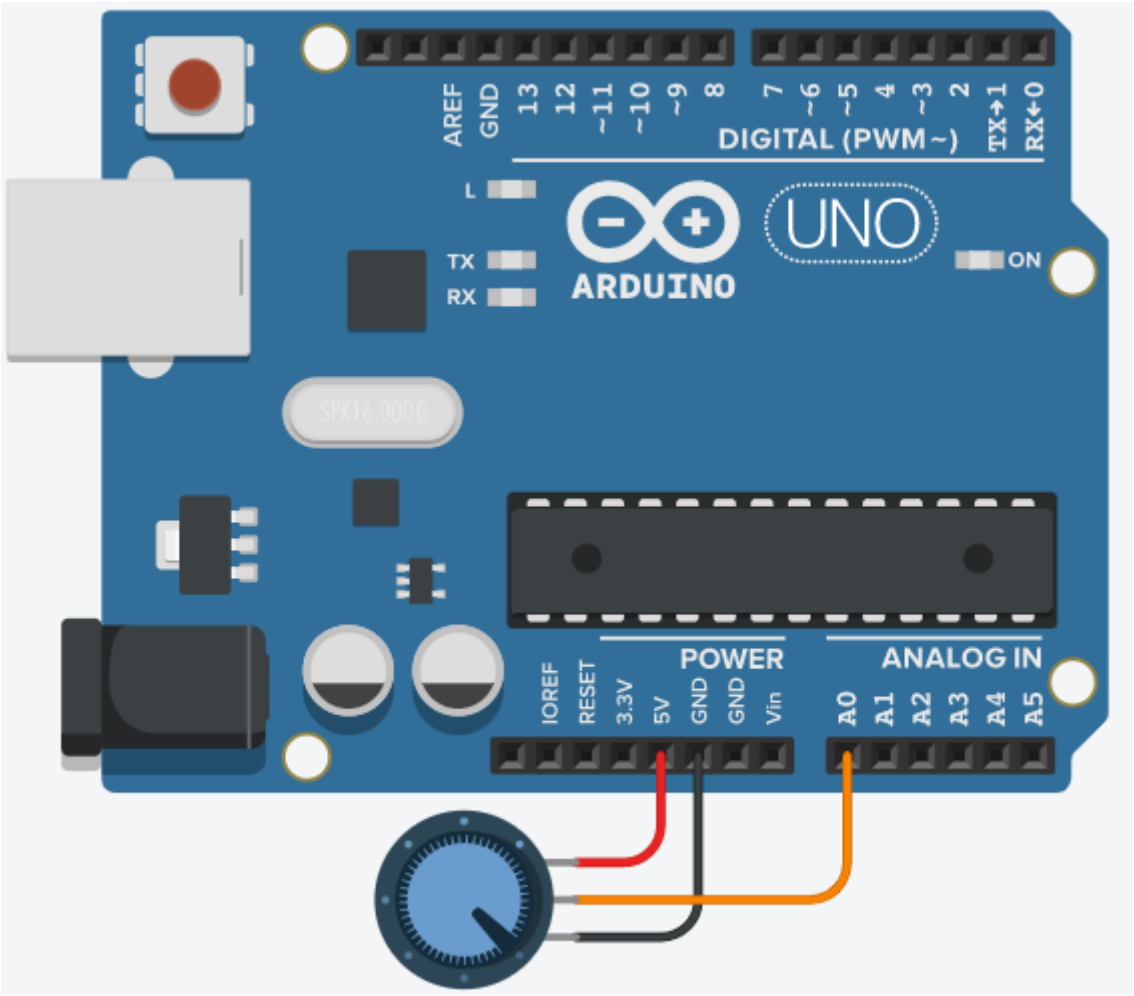


2- Mostrar tensão

Descrição

Crie um código para mostrar a cada 1 segundo (no monitor serial) o valor em bits da tensão de acordo com o ajuste do potenciômetro.

Imagem



Programação

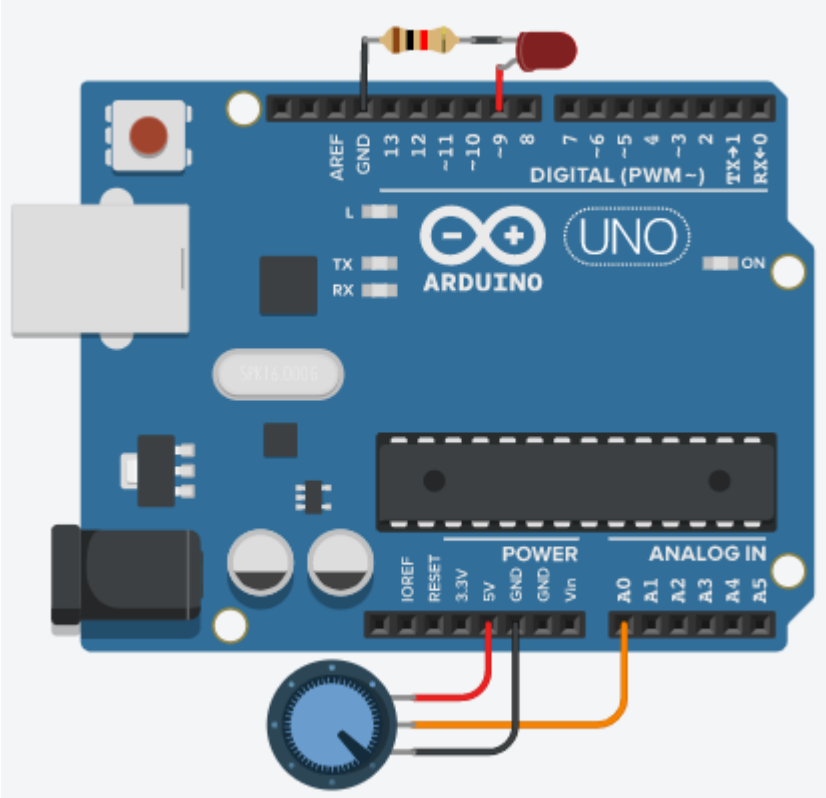
```
int input=A0;

void setup() {

        pinMode(input,  INPUT);

        Serial.begin (9600);

}


void loop() {

        int val = analogRead (input);

        delay (1000);

        Serial.println (val);

}
```

3- Variar tensão do LED

Descrição

Altere o código anterior e inclua um LED no circuito para variar a intensidade do LED através do potenciômetro.

Imagem



Programação

```
int input=A0, led=9;

void setup () {

        pinMode (led,   OUTPUT);

        pinMode (input,  INPUT);

}


void loop () {

        int val = analogRead (input);

        analogWrite (led, val);

}
```