Ministério da Educação – MEC
Secretaria de Educação Superior – SESu
Programa de Educação Tutorial – PET

Universidade Federal Fluminense – UFF
Escola de Engenharia – TCE
Grupo PET do Curso de Eng. de Telecomunicações – PET-Tele

Introdução ao *kit* de desenvolvimento Arduino

Planos de aulas, Questionários de aulas e Miniprojetos semanais

(Versão: A2023M01D19)

Manutenção: Pedro Henryque Barbosa da Silva e Kriz
Lúcio Folly Sanches Zebendo
Lucas Pontes Siqueira

Autores: Alexandre Santos de la Vega (Planos e unificação)
Lorraine de Miranda Paiva (Questionários e Miniprojetos)
Roberto Brauer Di Renna (Questionários e Miniprojetos)
Thiago Elias Bittencourt Cunha (Questionários e Miniprojetos)

Tutor: Alexandre Santos de la Vega

Niterói – RJ
Janeiro / 2023

Sumário

1 Introdução ao curso e ao *hardware* básico.

1.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: nenhuma/nenhum.
- Comunicação/elemento: nenhuma/nenhum.

1.2 Plano de aula

- Motivações:
  - Os alunos devem ter conhecimento da origem e da dinâmica do curso.
  - Os alunos devem ter conhecimento das regras definidas para o curso, a sua estrutura organizacional e os tópicos que nela serão abordados,
  - Os alunos devem ter conhecimento dos seus instrumentos de trabalho.
  - O Arduino é um *kit* de desenvolvimento, projetado tanto para uma prototipação rápida, quanto para uma solução final. Ele será empregado em todos os experimentos realizados no curso. O *kit* possui diversos componentes estruturais.
  - O *hardware* do Arduino é acompanhado por um *software* que é um ambiente de desenvolvimento integrado (*Integrated Development Environment* ou IDE), destinado ao desenvolvimento dos projetos.
  - Uma *protoboard* é um elemento auxiliar de prototipação. Ela é, praticamente, uma matriz de conexões. Nela, podem ser conectados fios e pequenos componentes eletro-eletrônicos.
  - Foi preparado um *kit* de aula, que é um organizador contendo diversos componentes, a serem usados nos experimentos com o Arduino.
  - Há uma ferramenta Web com recursos para o Arduino, onde é possível desenhar circuitos com o Arduino, inserir um código e realizar simulações do projeto.
- Objetivo:

  Realizar esclarecimentos iniciais sobre o curso e apresentar os principais instrumentos de trabalho.
- Conteúdo abordado:
  - Apresentação do curso: histórico, conteúdo, estrutura, metodologia e regras.
  - Breve apresentação sobre a área de aplicação do kit Arduino: microprocessadores, microcontroladores, sistemas embarcados e sistemas de controle.
  - Apresentação do *kit* Arduino: tipos e partes constituintes.
  - Apresentação do material auxiliar: *protoboard*, *kit* de aula e ferramenta Web.
  - Breve apresentação sobre o ambiente de desenvolvimento integrado (*Integrated Development Environment* ou IDE) do Arduino.
  - Breve apresentação sobre a estrutura de um código executado pelo *kit* Arduino.

- Resultados esperados:

  Ao final da experiência, o aluno deverá ser capaz de entender como o curso será conduzido e de dominar os conceitos básicos sobre o *kit* Arduino.

- Material utilizado:

  - *Hardware*: *kit* Arduino, *protoboard* e *kit* de aula.
  - *Software*: IDE do Arduino e ferramenta Web.
  - Material didático: *slides* e apostilas.

- Metodologia :

  1. Apresentar o curso
     - Histórico, conteúdo, estrutura e regras.
     - Apresentar os *websites* onde encontram-se os materiais didáticos e o simulador do Arduino para aulas *on-line*.
       Material: `http://www.telecom.uff.br/pet/petws/index.php`.
       Simulador: `https://www.tinkercad.com/circuits`).
     - Apresentar os métodos de avaliação do curso:
       Questionários de aulas (10%), Miniprojetos semanais (30%) e Projeto final (60%).
  2. Apresentar o *hardware*: *kit* Arduino, *protoboard* e *kit* de aula.
  3. Apresentar o *software*: visão geral do IDE e da linguagem de programação do Arduino.
  4. Apresentar um exemplo simples completo (*hardware* e *software*).
  5. Aplicar "Questionário de Aula".

1.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Desenhe, na Figura 1, a organização das trilhas da *protoboard*.

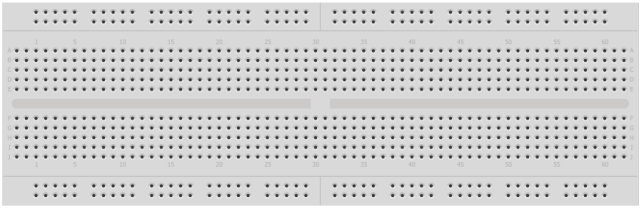

Figura 1: *Protoboard*.

2. Preencha a Tabela 1, identificando cada elemento presente na placa do Arduino UNO, ilustrado na Figura 2.

01	
02	
03	
04	
05	
06	
07	
08	
09	
10	
11	
12	
13	
14	
15	

Tabela 1: Elementos do Arduino.

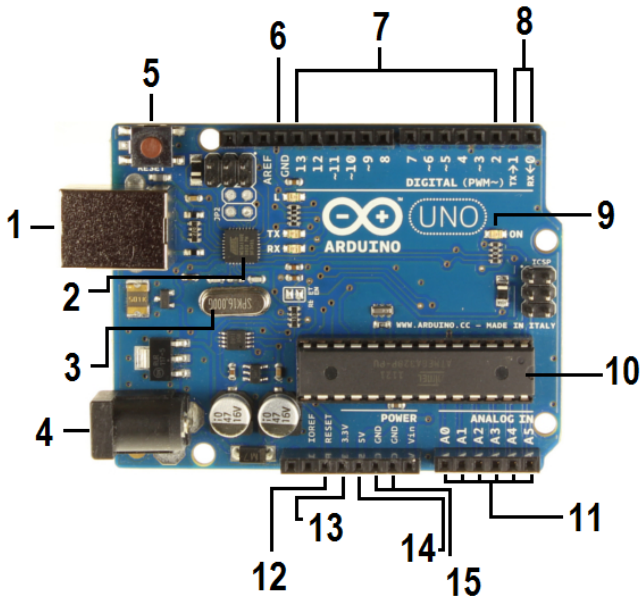

Figura 2: Arduino UNO.

1.4 Miniprojeto semanal

Empregando a ferramenta Web TinkerCAD, elabore um desenho contendo um *kit* Arduino e uma *protoboard*. Na *protoboard*, adicione um resistor e um LED, conectados em série. Conecte, com um fio, o resistor ao pino de GND do Arduino. Conecte, com um fio, o LED ao pino de número 6 do Arduino.

2 Introdução ao *software*: programação básica.

2.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: nenhuma/nenhum.
- Comunicação/elemento: nenhuma/nenhum.

2.2 Plano de aula

- Motivações:
  - O elemento principal do *kit* Arduino é um microcontrolador, o que possibilita que ele seja reconfigurável e programável.
  - A fim de facilitar a programação do *kit*, a programação pode ser realizada em uma linguagem similar à linguagem C++.
  - A tradução e o carregamento do código são realizados por um ambiente de desenvolvimento integrado (*Integrated Development Environment* ou IDE) que acompanha o *kit*.
  - Para um uso completo do *kit*, deve-se ter conhecimento sobre os seguintes tópicos: a linguagem de programação, as estruturas de um código, o fluxo de um código e o IDE.
- Objetivo: Introduzir os conceitos básicos sobre a programação do *kit* Arduino.
- Conteúdo abordado:

  Conceitos básicos sobre os seguintes tópicos:
  - Linguagem de programação do Arduino.
  - Estrutura geral de um código para o Arduino.
  - Fluxo de um código para o Arduino.
  - O IDE do Arduino.
- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender como o Arduino pode ser programado, quais são as principais estruturas da sua linguagem de programação e como utilizar o seu IDE.
- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard* e *kit* de aula.
  - *Software*: IDE do Arduino e ferramenta Web.
  - Material didático: *slides* e apostilas.

- Metodologia :

    1. Apresentar conceitos básicos sobre linguagens de programação.
    2. Apresentar conceitos básicos sobre as estruturas de controle de fluxo.
    3. Apresentar conceitos básicos sobre a estrutura de um código para o Arduino.
    4. Apresentar conceitos básicos sobre as estruturas de controle de fluxo para o Arduino.
    5. Apresentar conceitos básicos sobre o IDE do Arduino.
    6. Apresentar um exemplo simples completo (*hardware* e *software*).
    7. Propor expansão do exemplo apresentado.
    8. Aplicar "Questionário de Aula".

2.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. A linguagem de alto nível utilizada para programar o Arduino é similar à seguinte linguagem de programação: _____.

2. Os três elementos que compõem a estrutura geral de um código Arduino são os seguintes: _____, _____ e _____.

3. Descreva, brevemente, a função de cada um dos elementos que compõem a estrutura geral de um código Arduino.

4. No controle do fluxo da execução de um programa, as estruturas mais comumente utilizadas para alterar o fluxo linear são as seguintes: _____ e _____.

5. Apresente a sintaxe das estruturas mais comumente utilizadas para alterar o fluxo linear de execução de um programa.

2.4 Miniprojeto semanal

Elabore um código Arduino, utilizando os comandos `if` e `while`, que imprima no *Serial Monitor* os números pares de 1 a 100.

3 Acionamento de LED comum.

3.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: digital/LED comum.
- Comunicação/elemento: nenhuma/nenhum.

3.2 Plano de aula

- Motivações:
  - Uso de LED como elemento de atuação:
    - Como o próprio nome já indica, o LED (*Light-Emitting Diode*) é um dispositivo semicondutor capaz de emitir luz.
    - Tal característica faz dele um dispositivo naturalmente utilizado para simples iluminação, sinalização de alguma condição e até mesmo para comunicação à distância.
    - LEDs são capazes de irradiar em diversas faixas, tais como: infravermelho, ultravioleta e luz visível (vermelho, amarelo, verde, azul e violeta).
    - A radiação luminosa de um LED não é monocromática, como em um *laser*, mas a faixa irradiada é bem estreita, gerando a sensação de uma única cor.
    - Os LEDS comuns são baseados em único diodo, emitindo apenas em uma única faixa.
    - Os LEDS RGB (*Red*, *Green*, *Blue*) são compostos por três diodos, possibilitando a geração de cores compostas.
    - Uma vez que um LED é uma junção semicondutora, ele não é capaz de controlar a corrente elétrica que passa por ele. Para realizar tal função, pode-se empregar um resistor, em uma ligação série com o LED. O controle da corrente elétrica tem as seguintes funções: proteção contra danos e controle da intensidade da luz emitida.
  - O Arduino pode ser programado para gerar um sinal digital de saída, com dois níveis de tensão elétrica.
  - O sinal digital gerado pelo Arduino pode ser usado para o acionamento de um LED comum, acendendo-o ou apagando-o.
- Objetivo: Introduzir os conceitos básicos sobre o funcionamento de um LED e a forma de programar o Arduino para atuar sobre ele.
- Conteúdo abordado:
  - Funcionamento e operação de um LED.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de algumas funções básicas do Arduino, tais como: `pinMode()` e `digitalWrite()`.
- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender como funciona um LED e como programar o Arduino para atuar sobre ele.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, LEDs comuns e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Identificar alguns tipos de LED.
  2. Explicar o funcionamento de um LED.
  3. Explicar o cálculo dos resistores de proteção.
  4. Revisar o básico da programação do Arduino.
  5. Apresentar as funções usadas para geração de sinal digital de saída.
  6. Apresentar um código exemplo.
  7. Propor a expansão do exemplo apresentado.
  8. Aplicar "Questionário de Aula".

3.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Se, na função `pinMode()`, um pino for identificado como _____, podemos ler sua tensão como _____ para 5 V ou 3,3 V ou _____ para 0 V ou GND. Porém, se o pino for identificado como _____, podemos ler os valores do nosso componente.

2. A Figura 3 ilustra um circuito para acionamento de um LED. Analise o circuito e comente o seu funcionamento.

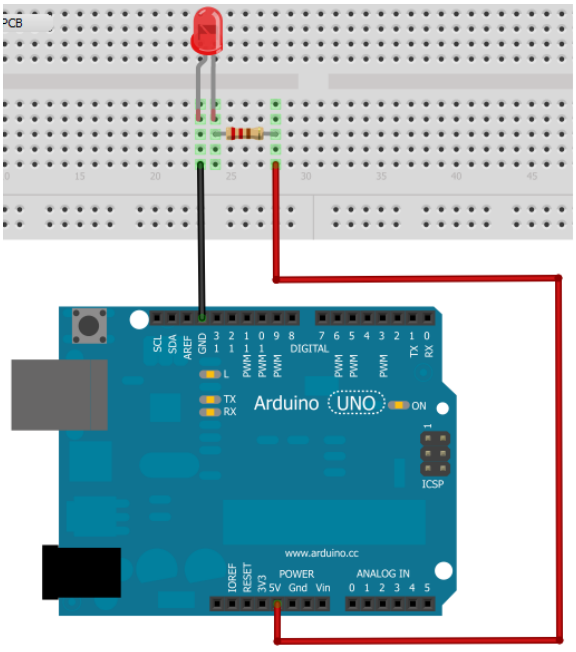

Figura 3: Circuito para acionamento de um LED.

3. A Figura 4 mostra um LED em detalhe. Complete a figura, indicando a sua polarização direta.

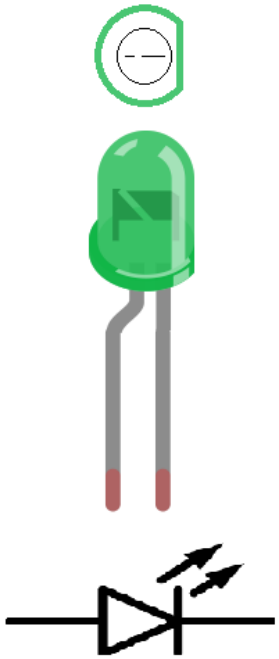

Figura 4: Polarização de um LED.

3.4 Miniprojeto semanal

Proponha um circuito com cinco LEDs, de cores variadas. Proponha uma sequência de acionamentos para o circuito. Elabore um *script* Arduino que implemente a sequência proposta.

4 Monitoramento de botões e acionamento de LED RGB.

4.1 Tipos de interação

- Entrada/sensor: digital/chave (botão).
- Saída/atuador: analógica/LED RGB.
- Comunicação/elemento: nenhuma/nenhum.

4.2 Plano de aula

- Motivações:
  - Uso de dispositivo mecânico de chaveamento como elemento de sensoriamento:
    - Dispositivos mecânicos de chaveamento geram oscilações (*bouncing*).
    - Uso de botão como dispositivo mecânico de chaveamento.
    - Uso de resistores (*pullup* e *pulldown*) para fixação de valores de tensão elétrica.
  - Uso de LED como elemento de atuação:
    - Como o próprio nome já indica, o LED (*Light-Emitting Diode*) é um dispositivo semicondutor capaz de emitir luz.
    - Tal característica faz dele um dispositivo naturalmente utilizado para simples iluminação, sinalização de alguma condição e até mesmo para comunicação à distância.
    - LEDs são capazes de irradiar em diversas faixas, tais como: infravermelho, ultravioleta e luz visível (vermelho, amarelo, verde, azul e violeta).
    - A radiação luminosa de um LED não é monocromática, como em um *laser*, mas a faixa irradiada é bem estreita, gerando a sensação de uma única cor.
    - Os LEDS comuns são baseados em único diodo, emitindo apenas em uma única faixa.
    - Os LEDS RGB (*Red*, *Green*, *Blue*) são compostos por três diodos, possibilitando a geração de cores compostas.
    - Uma vez que um LED é uma junção semicondutora, ele não é capaz de controlar a corrente elétrica que passa por ele. Para realizar tal função, pode-se empregar um resistor, em uma ligação série com o LED. O controle da corrente elétrica tem as seguintes funções: proteção contra danos e controle da intensidade da luz emitida.
  - Uso de sinal PWM para geração de valor analógico de saída a partir de sinal digital:
    - Basicamente, um sinal PWM (*Pulse-Width Modulation*) é uma onda quadrada, cujo valor da largura dos seus pulsos é controlado pelo valor de algum parâmetro de um outro sinal.
    - Em essência, um sinal PWM é um sinal analógico, mas que carrega uma informação discreta, na forma de dois valores constantes distintos.
    - Uma onda quadrada pode ser definida como a soma de sinais senoidais, com valores específicos para os seus parâmetros de definição (amplitude, frequência e ângulo de fase).

        * Um cosseno com frequência nula representa um valor constante ou valor DC. Por sua vez, cossenos com frequências não nulas, possuem valor médio (ou DC) nulo.
        * Logo, um sinal PWM com valor DC não nulo, aplicado a um atuador que não consiga responder rapidamente às variações das suas componentes senoidais, é aproximadamente equivalente a um sinal constante aplicado sobre o atuador.
        * O uso de sinal PWM é uma técnica muito eficaz, e muito utilizada, para gerar um sinal analógico constante, a partir de um sinal digital, sem a necessidade de um conversor digital-analógico (*Digital-to-Analog Converter* ou DAC).
    - Alguns microcontroladores possuem um resistor interno que pode ser habilitado por *software* para uso externo (resistor interno de *pullup* ou de *pulldown*).
    - O Arduino pode ser programado para gerar um sinal PWM de saída, que pode ser usado como um sinal analógico constante de saída.

- Objetivo: Introduzir os conceitos básicos sobre a interação com chaves e LEDs RGB, bem como sobre a forma de programar o Arduino para gerar um sinal PWM.

- Conteúdo abordado:

    - Funcionamento de chaves e botões.
    - Definição do efeito de *bouncing*.
    - Conceito de resistor de *pullup* e de *pulldown*.
    - Funcionamento e operação de um LED RGB.
    - Classificação de sinais: analógicos e digitais.
    - Modulação PWM e seu uso na geração de sinal constante.
    - Apresentação de algumas funções básicas do Arduino, tais como: `pinMode()` e `analogWrite()`.

- Resultados esperados: Ao final dessa experiência, o aluno deverá ser capaz de entender como monitorar uma chave, acionar um LED RGB e gerar um sinal PWM com o Arduino.

- Material utilizado:

    - *Hardware*: *kit* Arduino, *protoboard*, fios, chaves (botões) e LED RGB.
    - *Software*: IDE do Arduino.
    - Material didático: *slides* e apostilas.

- Metodologia:

    1. Identificar alguns tipos de botões.
    2. Explicar o efeito de *bouncing*.
    3. Explicar o conceito de resistor de *pullup* e de *pulldown*.
    4. Identificar alguns tipos de LED.
    5. Explicar o funcionamento de um LED RGB.
    6. Explicar o cálculo dos resistores de proteção.
    7. Explicar a classificação de um sinal como digital ou analógico.
    8. Explicar a modulação PWM.

9. Revisar o básico da programação do Arduino.
10. Apresentar as funções usadas para geração de sinal PWM de saída.
11. Apresentar um código exemplo.
12. Propor a expansão do exemplo apresentado.
13. Aplicar “Questionário de Aula”.

4.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Nesta aula, além da `digitalWrite()`, também usamos a função _____ para variar a intensidade luminosa dos LEDs _____. Essa função só opera corretamente quando usamos as saídas _____. A função _____ escreve um valor de _____ para o pino digital, que pode variar de _____ a _____. Quanto mais alto o valor escrito, maior a intensidade luminosa do LED.

2. A Figura 5 mostra a imagem de um LED RGB. Identifique a sua pinagem.

Figura 5: LED RGB.

3. No início da função `void loop()`, aparece o seguinte trecho de código:

```
estadoBotao_1 = digitalRead(botao_1);
estadoBotao_2 = digitalRead(botao_2);
estadoBotao_3 = digitalRead(botao_3);
```

Descreva o funcionamento da função `digitalRead()`, usando esse trecho como exemplo.

4.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito em que um LED RGB passe pelas três cores diferentes e que cada cor passe da intensidade mais alta (255) até a intensidade mais baixa (0).

Dica: Utilize o comando de repetição `for`.

5 Comunicação serial e interação com o *Serial Monitor*.

5.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: digital/LED comum.
- Comunicação/elemento: serial/*Serial Monitor*.

5.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Uso de LED como elemento de atuação:
    - Como o próprio nome já indica, o LED (*Light-Emitting Diode*) é um dispositivo semicondutor capaz de emitir luz.
    - Tal característica faz dele um dispositivo naturalmente utilizado para simples iluminação, sinalização de alguma condição e até mesmo para comunicação à distância.
    - LEDs são capazes de irradiar em diversas faixas, tais como: infravermelho, ultravioleta e luz visível (vermelho, amarelo, verde, azul e violeta).
    - A radiação luminosa de um LED não é monocromática, como em um *laser*, mas a faixa irradiada é bem estreita, gerando a sensação de uma única cor.
    - Os LEDS comuns são baseados em único diodo, emitindo apenas em uma única faixa.
    - Os LEDS RGB (*Red*, *Green*, *Blue*) são compostos por três diodos, possibilitando a geração de cores compostas.
    - Uma vez que um LED é uma junção semicondutora, ele não é capaz de controlar a corrente elétrica que passa por ele. Para realizar tal função, pode-se empregar um resistor, em uma ligação série com o LED. O controle da corrente elétrica

tem as seguintes funções: proteção contra danos e controle da intensidade da luz emitida.

  - Alguns microcontroladores possuem pinos (transmissor ou TX e receptor ou RX) por meio dos quais pode ser estabelecida uma comunicação serial no padrão EIA RS-232C.
  - Há circuitos integrados especializados em servir de interface de comunicação serial entre os padrões EIA RS-232C e USB.
  - O Arduino pode ser programado para estabelecer uma comunicação serial.
  - O Arduino pode ser programado para gerar um sinal digital de saída, com dois níveis de tensão elétrica.
  - O sinal digital gerado pelo Arduino pode ser usado para o acionamento de um LED comum, acendendo-o ou apagando-o.

- Objetivo: Introduzir os conceitos básicos sobre uma comunicação serial e o funcionamento do *software Serial Monitor*.

- Conteúdo abordado:

  - Conceitos básicos sobre comunicação digital.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Funcionamento e operação de um LED.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de algumas funções básicas da classe *Serial*, que controlam a comunicação serial no Arduino.

- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender como funciona uma comunicação serial e como utilizar o *Serial Monitor* para se comunicar com o Arduino.

- Material utilizado:

  - *Hardware*: *kit* Arduino, *protoboard*, fios, LEDs comuns e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.

- Metodologia:

  1. Explicar os conceitos básicos sobre comunicação digital.
  2. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  3. Identificar alguns tipos de LED.
  4. Explicar o funcionamento de um LED.
  5. Explicar o cálculo dos resistores de proteção.
  6. Revisar o básico da programação do Arduino.
  7. Apresentar as funções usadas para geração de sinal digital de saída.
  8. Apresentar um código exemplo.
  9. Propor a expansão do exemplo apresentado.
  10. Aplicar “Questionário de Aula”.

5.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

Parte I

Na prática de hoje, acendemos um LED a partir do teclado do computador. Proponha uma extensão, agora com uma tecla para apagar e uma outra para acender. Apresente o código criado para a extensão proposta.

Parte II

Em cada um dos casos abaixo, escolha uma única opção.

1. O *Serial Monitor* é muito usado para:

   a) Mostrar valores lidos.

   b) Mostrar o circuito montado.

   c) Interpretar sinais.

   d) Mostrar, automaticamente, valores nas escalas do Sistema Internacional.

2. Quando ajustamos a tensão para acender e apagar um LED, escolhemos um pino digital ao invés de um analógico, porque:

   a) O pino digital mantém a tensão constante em 5 V.

   b) O pino digital envia sinais de *HIGH* e *LOW*, quando for desejado.

   c) Há mais pinos digitais do que analógicos.

   d) Não faz diferença optar por um pino analógico.

3. `Serial.begin()`, `Serial.flush()`, `Serial.read()` e `Serial.print()`, são funções que servem, respectivamente, para:

   a) Iniciar o programa; esvaziar o *buffer*; ler o valor da porta serial; mostrar os dados na tela.

   b) Determinar a taxa de *bits* por segundo (bps); esvaziar o *buffer*; ler o valor da variável serial; escrever na tela.

   c) Iniciar o programa; controlar a transmissão de todos os dados do tipo serial e esvaziar o *buffer* da porta de entrada; ler o valor da variável serial; mostrar os dados na tela.

   d) Determinar a taxa de *bits* por segundo (bps); esvaziar o *buffer* da porta de entrada (nas versões 1.0.x) e esperar o fim da transmissão de todos os dados seriais; ler o valor da porta serial; mostra os dados na tela.

5.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito em que as teclas A, B e C, acendam e apaguem 3 LEDs distintos. Utilize também a tecla 1 para que apenas 1 dos LEDs possa permanecer aceso, a tecla 2 para que 2 LEDs possam permanecer acesos simultaneamente e a tecla 3 para que os 3 LEDs possam permanecer acesos simultaneamente. Ao alterar entre as teclas 1, 2 e 3, todos os LEDs que estiverem acesos devem ser apagados. Utilize a interface *Serial Monitor* para verificar as teclas pressionadas.

6 Acionamento de dispositivo sonorizador.

6.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: digital/sonorizador *buzzer* ativo e/ou passivo.
- Comunicação/elemento: serial/*Serial Monitor*.

6.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Som:
    - O som é uma percepção humana para vibrações do ar.
    - De uma forma aproximada, o ouvido humano é capaz de captar vibrações na seguinte faixa de frequências: $20 \leq f \leq 20.000$ Hz.
    - Para que a voz humana seja compreendida, é necessário que se utilize apenas a seguinte faixa: $20 \leq f \leq 4.000$ Hz.
    - Matematicamente, uma forma de onda periódica pode ser definida como a soma de sinais senoidais, com valores específicos para os seus parâmetros de definição (amplitude, frequência e ângulo de fase).
    - A composição de um sinal, formada por tal soma de sinais senoidais, é chamada de Composição Espectral do Sinal.
  - Música:
    - Em Teoria Musical, definem-se parâmetros que possuem correspondência com um ou mais parâmetros da Física do Som.
    - Um *musical tone* (tom musical) é definido como um som musical periódico. Ele é caracterizado pelos seguintes parâmetros: *duration* (duração temporal), *pitch* (frequência fundamental), *intensity* ou *loudness* (intensidade) e *timbre* ou *quality* (timbre ou composição sonora do tom).

    - O timbre traduz a composição sonora do tom musical, caracterizando os componentes senoidais envolvidos na composição.
    - O *pitch* de um tom musical é a menor frequência não nula da sua composição.
    - Um tom musical pode ser classificado como tom simples ou puro (apenas um componente) ou como tom complexo (vários componentes).
    - Os componentes que formam o tom são denominados de *partials*. O componente com a menor frequência não nula é chamado de *first partial*. Os componentes com frequências acima do *first partial* são denominados de *overtones*.
    - Quando as frequências da composição possuem valores que são múltiplos inteiros da menor frequência não nula, ela é dita uma composição harmônica. Nesse caso, a menor frequência não nula da composição é denominada de frequência fundamental. Por sua vez, o *first partial* é dito primeiro (componente) harmônico.
    - Os *overtones* que possuem ou não uma relação harmônica com o *pitch* são respectivamente chamados de *harmonic partials* ou *inharmonic partials*.
    - A representação de um som musical é denominada de nota.
    - Uma nota pode representar o *pitch* e a duração temporal da vibração.
    - Quando dois *pitches* se diferenciam por uma razão de dobro da frequência, isso caracteriza um intervalo de oitava (*octave*) ou de oitava perfeita (*perfect octave*) entre eles.
    - Tons musicais com *pitches* relacionados por 2^k, onde $k \in \mathbb{Z}$, são percebidas de forma muito similar. Portanto, também pode-se dizer que uma nota representa uma classe de *pitches*, onde uma determinada nota da classe é determinada por nota_k. Por exemplo, no piano, a representação Lá_k corresponde às suas diversas notas Lá, que são associadas a diferentes frequências e que são localizadas em diferentes posições do seu teclado.
- Mecanismo de produção do som em dispositivos sonoros:
    - Basicamente, os dispositivos sonoros geram seus sons por meio da movimentação de uma membrana.
    - Comumente, a membrana é acionada de uma forma eletromecânica. Um sinal elétrico senoidal, com uma dada frequência, é aplicado a um indutor (ou bobina), que possui um núcleo móvel. A variação do campo magnético, produzida pela variação da corrente elétrica no indutor, movimenta o núcleo, que, por sua vez, movimenta a membrana.
    - Uma alternativa para a movimentação de pequenas membranas é o uso de um cristal piezoelétrico. Tal elemento sofre deformações mecânicas quando se aplica uma diferença de potencial elétrico em pontos específicos sobre ele, e vice-versa.
    - Nos pequenos dispositivos sonoros denominados de *buzzers*, a membrana é uma fina lâmina metálica ou ainda uma final lâmina de cristal piezoelétrico.
- Uso de *buzzer* como elemento de atuação:
    - De acordo com o seu acionamento, um *buzzer* pode ser classificado nos seguintes tipos: ativo e passivo.
    - Um *buzzer* ativo possui circuito eletrônico de acionamento interno, contendo um oscilador senoidal, com frequência única e predefinida. Logo, para acioná-lo, basta aplicar sobre ele apenas um valor de tensão constante.
    - Um *buzzer* passivo não possui circuito eletrônico de acionamento interno. Logo, é necessário aplicar sobre ele um sinal senoidal com a frequência desejada.

  - O Arduino pode ser programado para gerar um sinal digital de saída, com dois níveis de tensão elétrica. Tal sinal pode ser usado para o acionamento de um *buzzer* ativo.
  - O Arduino pode ser programado para gerar uma onda quadrada na saída, com uma frequência desejada. Tal sinal pode ser usado para o acionamento de um *buzzer* passivo.

- Objetivo: Introduzir os conceitos básicos sobre o som, a música, o funcionamento de sonorizadores e a forma de programar o Arduino para atuar sobre sonorizadores do tipo *buzzer*, a partir do teclado computador, com o auxílio do *software Serial Monitor*.

- Conteúdo abordado:
  - Conceitos básicos sobre comunicação digital.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Conceitos básicos sobre som e música.
  - Funcionamento e operação de sonorizadores, em particular do tipo *buzzer*.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de algumas funções básicas que controlam a geração de tons: `tone()` e `noTone()`.

- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender como funciona uma comunicação serial, como utilizar o *Serial Monitor* para se comunicar com o Arduino, como usar o Arduino para atuar sobre um sonorizador do tipo *buzzer*.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, *buzzer* e resistor.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.

- Metodologia:
  1. Explicar os conceitos básicos sobre comunicação digital.
  2. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  3. Explicar os conceitos básicos sobre som e música.
  4. Identificar os tipos de *buzzer*.
  5. Explicar o funcionamento de um *buzzer*.
  6. Explicar o cálculo dos resistores de proteção.
  7. Revisar o básico da programação do Arduino.
  8. Apresentar as funções usadas para geração de sinal de atuação sobre o *buzzer*.
  9. Apresentar um código exemplo.
  10. Propor a expansão do exemplo apresentado.
  11. Aplicar "Questionário de Aula".

6.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. A função `tone()` gera, em um pino especificado, uma onda quadrada de frequência especificada, com ciclo de trabalho (*duty cycle*) de 50%. A duração também pode ser especificada. Caso contrário, a onda permanece até que haja uma chamada para a função _____.

   Apenas um tom pode ser gerado de cada vez. Se um tom já está tocando em um pino diferente, a chamada para a `tone()` não terá qualquer efeito. Se o tom está tocando no mesmo pino, a chamada irá redefinir a sua frequência.

   A ativação da função `tone()` irá interferir na saída _____ nos pinos 3 e 11 (em todas as placas, exceto no Arduino MEGA).

   Não é possível gerar tons inferiores a 31 Hz.

   Nota:

   Se você quiser atuar em diferentes pinos, você precisará chamar `noTone()` em um pino antes de chamar a `tone()` no próximo pino.

   Sintaxe:

   tone(_____,_____)

   ou

   tone(_____,_____,_____)

2. Complete a Tabela 2, para cada nota musical.

Nota	Frequência (Hz)	Tecla usada
DÓ		
RÉ		
MI		
FÁ		
SOL		
LÁ		
SI		

Tabela 2: Associação Nota-Frequência-Tecla.

6.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito em que um *buzzer* toque uma música de sua escolha.

7 Mecanismos de temporização.

7.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: nenhuma/nenhum.
- Comunicação/elemento: serial/*Serial Monitor*.

7.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Mecanismos de temporização:
    - De uma forma geral, um *loop* de controle é um processo contínuo de controle que envolve, nessa sequência, as seguintes ações, executadas pelo sistema de controle: sensoriamento (medição), processamento dos dados obtidos nas medições, tomadas de decisões e atuação sobre o sistema sob controle.
    - Por vezes, nesse processo, é necessário parar o processamento, em alguma de suas etapas, por uma determinada quantidade de tempo.
    - Em outras ocasiões, é necessário contabilizar uma determinada quantidade de tempo, existente entre dois eventos consecutivos.
  - O Arduino pode ser programado para realizar a retenção do seu processamento em um determinado ponto e para computar intervalos de tempo entre eventos.
- Objetivo: Introduzir os conceitos básicos sobre mecanismos de temporização (retenção e contabilidade temporal) e o funcionamento do *software Serial Monitor*.

- Conteúdo abordado:
  - Conceitos básicos sobre mecanismos de temporização.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de algumas funções básicas para temporização: `delay()`, `delayMicroseconds()`, `millis()` e `micros()`.
- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender os mecanismos de temporização e como utilizar o *Serial Monitor* para se comunicar com o Arduino.
- Material utilizado:
  - *Hardware*: *kit* Arduino.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Explicar os conceitos básicos sobre os mecanismos de temporização.
  2. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  3. Revisar o básico da programação do Arduino.
  4. Apresentar as funções usadas para temporização.
  5. Apresentar um código exemplo.
  6. Propor a expansão do exemplo apresentado.
  7. Aplicar "Questionário de Aula".

7.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Apresentamos, na aula de hoje, as seguintes funções de temporização: _____, _____ e _____.

   Vimos que a função `delay()` é responsável por pausar o programa por um tempo determinado em _____. Por sua vez, as funções _____ e `micros()` retornam em milissegundos e _____, respectivamente, o tempo que o programa está em execução.

2. Nesta aula, foi trabalhada a função `Serial.print()`. Escreva um pequeno trecho de código, utilizando-a, para escrever, no *Serial Monitor*, o conteúdo de uma variável que esteja armazenando a quantidade de tempo, em microssegundos, em que o programa está em execução.

7.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito, que implemente um semáforo com três cores (verde, amarelo e vermelho), simulado por LEDs, utilizando as funções de temporização `delay()`, `millis()` e `micros()`.

A luz verde (aberto) deve permanecer ligada por 3 segundos, a amarela (atenção) por 1 segundo e a luz vermelha (fechado) por 2 segundos, nessa sequência, em um ciclo ininterrupto.

Além disso, ao final de cada sinal fechado, o conteúdo de uma variável temporal deve ser impresso no *Serial Monitor*, indicando, em milissegundos, o tempo em que o programa esteve em execução.

8 Monitoramento de sensor de luminosidade.

8.1 Tipos de interação

- Entrada/sensor: analógica/sensor de luminosidade LDR.
- Saída/atuador: nenhuma/nenhum.
- Comunicação/elemento: serial/*Serial Monitor*.

8.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Grandezas analógicas, grandezas digitais e digitalização (amostragem e quantização):
    - Uma relação funcional $y = f(x)$ é considerada analógica quando as variáveis y e x são contínuas.
    - Por sua vez, uma relação funcional $z = g[n]$ é considerada digital quando as variáveis z e n são discretas e cada um dos seus valores pode ser representado por um número finito de símbolos.
    - A conversão de uma relação analógica em digital, denominada de digitalização, envolve um processo de discretização, aplicado sobre as variáveis y e x, a fim de gerar as variáveis z e n, respectivamente.
    - Basicamente, a discretização significa tomar alguns valores individuais de uma grandeza contínua, dentro de uma determinada faixa de valores.
    - A discretização da variável independente x é denominada amostragem (*sampling*). Ao conjunto de valores individuais de x, pode ser associado um conjunto de índices inteiros n.
    - A discretização da variável dependente y é denominada quantização. Ao conjunto de valores individuais de y, pode ser associado um conjunto de valores z, aproximados com um número finito de símbolos.

    * Logo, pode-se dizer que a digitalização converte um conjunto infinito de valores, não completamente representáveis, em uma sequência finita de valores aproximados, perfeitamente representáveis.
    * No caso de uma discretização uniforme, os valores individuais são tomados com base em um intervalo constante, denominado de resolução da discretização (r). Assim, podem ser definidas a resolução da amostragem (r_S) e a resolução da quantização (r_Q)
  - Uso de conversor analógico-digital:
    * Por vezes, em tarefas de controle, é necessário captar dados analógicos, realizar cálculos sobre eles e tomar decisões a partir disso.
    * Porém, os dados são armazenados e manipulados, bem como as decisões são tomadas, em ambiente digital.
    * Existem circuitos eletrônicos que, utilizando diferentes técnicas, são capazes de implementar a digitalização. Um tal circuito é denominado Conversor A/D (*Analog-to-Digital Converter* ou *ADC*).
    * Os microcontroladores costumam incorporar um ADC em seus circuitos internos.
  - Uso de sensor de luminosidade:
    * Um fotodetector ou fotosensor é um sensor de luz (ou de outra radiação eletromagnética não visível).
    * Os fotodetectores podem ser classificados de acordo com o mecanismo de detecção empregado na sua fabricação.
    * Alguns mecanismos de detecção utilizados são os seguintes: fotoemissão ou efeito fotoelétrico; efeito térmico; polarização ótica; efeito fotoquímico; efeitos sobre a interação fraca (*weak interaction*).
    * Entre os fotodetectores que se utilizam de material semicondutor, podem-se citar os fotoresistores.
    * Um fotoresistor ou *Light Dependent Resistors* (LDR) é um resistor que varia o valor da sua resistência de acordo com a intensidade da luz que incide sobre ele. Normalmente, a resistência diminui com o aumento da intensidade de luz incidente.
  - O Arduino pode ser programado para utilizar o ADC interno ao seu microcontrolador a fim de monitorar uma grandeza analógica.

- Objetivo: Introduzir os conceitos básicos sobre o processo de conversão analógico-digital, o monitoramento de um sensor analógico, o funcionamento de um sensor de luminosidade e o funcionamento do *software Serial Monitor*.

- Conteúdo abordado:
  - Conceitos básicos sobre o processo de conversão analógico-digital.
  - Conceitos básicos sobre o funcionamento de um sensor de luminosidade do tipo LDR.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de função básica para leitura de dados analógicos: `analogRead()`.

- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender o processo de conversão analógico-digital, o monitoramento de um sensor analógico, o funcionamento de um sensor de luminosidade e como utilizar o *Serial Monitor* para se comunicar com o Arduino.
- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, sensor de luminosidade LDR e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Explicar os conceitos básicos sobre o processo de conversão analógico-digital.
  2. Explicar os conceitos básicos sobre o funcionamento de um sensor de luminosidade do tipo LDR.
  3. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  4. Revisar o básico da programação do Arduino.
  5. Apresentar um código exemplo.
  6. Propor a expansão do exemplo apresentado.
  7. Aplicar "Questionário de Aula".

8.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Na aula de hoje, compreendemos melhor o uso da função `analog.Read()`. Vimos que esta função é utilizada para _____ de um dos pinos _____ específico.

   Estudamos também o conceito de quantização e descobrimos como funciona o conversor A/D do Arduino. Dado que este conversor apresenta uma resolução de _____, teremos uma escala com 1024 valores quantizados, variando de _____ a _____. Tendo, assim, uma resolução de leitura de _____ para cada unidade de medida em uma escala entre de 0 e 5 V.

2. Descreva, com suas palavras, o funcionamento do processo de quantização e a conversão analógico-digital (A/D) feita pelo Arduino.

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

3. Nesta aula, utilizamos um sensor de luminosidade LDR, ilustrado na Figura 6. Este sensor nada mais é do que uma resistência _____, que varia de acordo com a luminosidade incidida. Observamos que, quanto maior a luminosidade incidente, _____ era a tensão lida pelo Arduino, e que, quanto menor a luminosidade, _____ era a tensão lida.

Figura 6: Sensor de luminosidade LDR.

4. Desenhe, abaixo, o diagrama utilizado para a leitura do sensor LDR.

8.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito em que um LED verde acenda quando o valor lido de um sensor de luminosidade LDR seja menor que 75, um LED vermelho acenda quando o valor for maior que 200 e que um LED amarelo acenda quando o valor estiver entre 75 e 200.

9 Monitoramento de sensor de temperatura e funções definidas pelo usuário.

9.1 Tipos de interação

- Entrada/sensor: analógica/sensor de temperatura.
- Saída/atuador: nenhuma/nenhum.
- Comunicação/elemento: serial/*Serial Monitor*.

9.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Grandezas analógicas, grandezas digitais e digitalização (amostragem e quantização):
    - Uma relação funcional $y = f(x)$ é considerada analógica quando as variáveis y e x são contínuas.
    - Por sua vez, uma relação funcional $z = g[n]$ é considerada digital quando as variáveis z e n são discretas e cada um dos seus valores pode ser representado por um número finito de símbolos.
    - A conversão de uma relação analógica em digital, denominada de digitalização, envolve um processo de discretização, aplicado sobre as variáveis y e x, a fim de gerar as variáveis z e n, respectivamente.
    - Basicamente, a discretização significa tomar alguns valores individuais de uma grandeza contínua, dentro de uma determinada faixa de valores.
    - A discretização da variável independente x é denominada amostragem (*sampling*). Ao conjunto de valores individuais de x, pode ser associado um conjunto de índices inteiros n.
    - A discretização da variável dependente y é denominada quantização. Ao conjunto de valores individuais de y, pode ser associado um conjunto de valores z, aproximados com um número finito de símbolos.

    * Logo, pode-se dizer que a digitalização converte um conjunto infinito de valores, não completamente representáveis, em uma sequência finita de valores aproximados, perfeitamente representáveis.
    * No caso de uma discretização uniforme, os valores individuais são tomados com base em um intervalo constante, denominado de resolução da discretização (r). Assim, podem ser definidas a resolução da amostragem (r_S) e a resolução da quantização (r_Q)

- Uso de conversor analógico-digital:
    * Por vezes, em tarefas de controle, é necessário captar dados analógicos, realizar cálculos sobre eles e tomar decisões a partir disso.
    * Porém, os dados são armazenados e manipulados, bem como as decisões são tomadas, em ambiente digital.
    * Existem circuitos eletrônicos que, utilizando diferentes técnicas, são capazes de implementar a digitalização. Um tal circuito é denominado Conversor A/D (*Analog-to-Digital Converter* ou *ADC*).
    * Os microcontroladores costumam incorporar um ADC em seus circuitos internos.
- Uso de sensor de temperatura:
    * Na Física, a temperatura é uma grandeza associada à energia cinética média das partículas que compõem um sistema em equilíbrio térmico.
    * Por sua vez, calor é energia, que pode ser transferida entre porções de matéria que estão em contato.
    * Quando duas porções de matéria são colocadas em contato, há uma transferência de energia (calor) daquela que possui a temperatura mais elevada para aquela que possui a temperatura mais baixa.
    * Na percepção humana, a temperatura está relacionada às sensações definidas como frio (perda de calor) e quente (ganho de calor).
    * Os seguintes sensores de temperatura são comumente utilizados:
        · DHT11 e DHT22/AM2302: que possuem um sensor de umidade capacitivo e um sensor de temperatura termorresistivo (termistor) com característica NTC (*Negative Temperature Coefficient*).
        · LM35: que é um sensor baseado em semicondutor, com característica PTC (*Positive Temperature Coefficient*), dada por 10 mV/^{o}C.
        · TMP35/36/37: que é um sensor baseado em semicondutor, com característica PTC (*Positive Temperature Coefficient*), dada por [10;10;20] mV/^{o}C, *offset* de saída dado por [0;0,5;0] mV e saída para 25^{o}C dada por [250,750,500] mV.
    * A temperatura pode ser expressa em diversas unidades diferentes, tais como: Kelvin (K), grau centígrado ou Celsius (^{o}C) e grau Fahrenheit (^{o}F).
    * Conhecendo-se as relações entre duas unidades diferentes, funções podem ser definidas para realizar o mapeamento entre elas, tais como:
      K = C + 273,15 e F = (9/5) C + 32.
- O Arduino pode ser programado para utilizar o ADC interno ao seu microcontrolador a fim de monitorar uma grandeza analógica.
- Assim como em linguagens de programação de uso geral, podem-se empregar funções definidas pelo usuário em um código para o Arduino.

- Objetivo: Introduzir os conceitos básicos sobre o processo de conversão analógico-digital, o monitoramento de um sensor analógico, o funcionamento de um sensor de temperatura, o procedimento para criação e uso de funções e o funcionamento do *software Serial Monitor*.
- Conteúdo abordado:
  - Conceitos básicos sobre o processo de conversão analógico-digital.
  - Conceitos básicos sobre o funcionamento de um sensor de temperatura.
  - Conceitos básicos sobre o procedimento para criação e uso de funções
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de função básica para leitura de dados analógicos: `analogRead()`.
- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender o processo de conversão analógico-digital, o monitoramento de um sensor analógico, o funcionamento de um sensor de temperatura, o procedimento para criação e uso de funções e como utilizar o *Serial Monitor* para se comunicar com o Arduino.
- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, sensor de temperatura e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Explicar os conceitos básicos sobre o processo de conversão analógico-digital.
  2. Explicar os conceitos básicos sobre o funcionamento de um sensor de temperatura.
  3. Explicar os conceitos básicos sobre o procedimento para criação e uso de funções.
  4. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  5. Revisar o básico da programação do Arduino.
  6. Apresentar um código exemplo.
  7. Propor a expansão do exemplo apresentado.
  8. Aplicar “Questionário de Aula”.

9.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Nesta aula, utilizamos um sensor de temperatura e a função `analog.Read()` para lermos o valor de _____ e identificarmos a temperatura correspondente.

   A Figura 7 apresenta uma imagem do sensor. Identifique a sua pinagem.

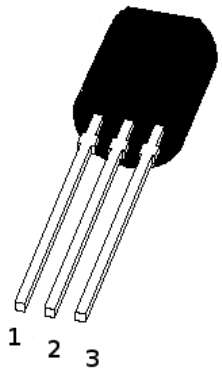

Figura 7: Sensor de temperatura.

   Pino / função :

   1 - .....

   2 - .....

   3 - .....

2. Na aula anterior, estudamos a função `analog.Read()` e vimos como funciona o conversor A/D do Arduino. Vimos que o conversor possui uma resolução de leitura de _____, que é calculada dividindo _____ por _____, que é a quantidade de valores quantizados.

   Utilizamos esta mesma função para determinar a temperatura lida pelo sensor. Para isso, vimos, em aula, que, para cada _____ lido pelo Arduino, tínhamos uma elevação de 1 K.

   Assim, para sabermos a temperatura em graus Celsius basta diminuir a temperatura lida em Kelvin por _____.

9.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito que leia a temperatura de um sensor de temperatura, após o acionamento de um botão.

Se a temperatura estiver acima de um determinado limite superior T_{high}, um LED vermelho deve ser ligado, por um tempo preestabelecido.

Se a temperatura estiver abaixo de um determinado limite inferior T_{low}, um LED azul deve ser ligado, por um tempo preestabelecido.

Se a temperatura estiver entre os dois limites anteriores, um LED amarelo deve ser ligado, por um tempo preestabelecido.

10 Monitoramento de sensor de movimento IR.

10.1 Tipos de interação

- <Entrada> / <sensor>: <analógica e/ou digital> / <sensor de movimento IR>.
- <Saída> / <atuador>: <digital> / <sonorizador *buzzer* ativo e/ou passivo>.
- <Comunicação> / <elemento>: <serial> / <*Serial Monitor*>.

10.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Uso de sensor de movimento IR:
    - A Física indica que objetos com temperatura acima do zero absoluto (0 K) emitem calor na forma de radiação, na faixa de frequências denominada IR (*InfraRed* ou infravermelho). Assim, é possível detectar a presença e/ou a movimentação de um corpo por meio da detecção da sua radiação IR.
    - Um sensor de movimento PIR (*Passive InfraRed*) é um dispositivo que possui dois detectores IR. Analisando os sinais presentes em cada detector, é possível detectar o movimento de um corpo radiante. Normalmente, o detector PIR apresenta uma área de detecção definida por uma distância linear e um ângulo.
    - Usando dispositivos semicondutores, é possível implementar um transmissor de sinal IR (LED IR) e um receptor de sinal IR (fototransistor). Assim, é possível detectar a presença e/ou a movimentação de um corpo por meio da interrupção de um feixe de radiação IR.
  - Sinalização da detecção de presença ou movimento:
    - Uma vez detectados uma presença e/ou um movimento, pode-se atuar sobre um dispositivo de sinalização, tal como um *buzzer*.

    - Além disso, pode-se computar e sinalizar a temporização associada à detecção da presença e/ou do movimento.
  - O Arduino pode ser programado para utilizar uma biblioteca de funções destinadas a interagir com um tipo de sensor de movimento PIR.
  - O Arduino pode ser programado para atuar sobre um LED IR e monitorar um foto-resistor.
  - O Arduino pode ser programado para atuar sobre um *buzzer* e para computar temporizações.

- Objetivo: Introduzir os conceitos básicos sobre o funcionamento de um sensor de movimento IR, as possibilidades de sinalização de um evento e o funcionamento do *software Serial Monitor*.

- Conteúdo abordado:
  - Conceitos básicos sobre o funcionamento de um sensor de movimento IR.
  - Conceitos básicos sobre algumas possibilidades de sinalização de um evento.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.

- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender o funcionamento de um sensor de movimento IR, algumas possibilidades de sinalização de um evento e como utilizar o *Serial Monitor* para se comunicar com o Arduino.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, sensor de movimento IR, *buzzer* e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.

- Metodologia:
  1. Explicar os conceitos básicos sobre o funcionamento de um sensor de movimento IR.
  2. Explicar os conceitos básicos sobre algumas possibilidades de sinalização de um evento.
  3. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  4. Revisar o básico da programação do Arduino.
  5. Apresentar um código exemplo.
  6. Propor a expansão do exemplo apresentado.
  7. Aplicar “Questionário de Aula”.

10.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Nesta aula, foi explicado o funcionamento do sensor de movimento PIR (*Passive InfraRed*). O circuito tem um funcionamento simples. Explique, com suas palavras, o funcionamento do circuito:

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

2. O sensor infravermelho pode nos ser útil em diversas aplicações. Entre elas, podemos fazer um alarme sonoro. Um sensor de presença pode ser instalado em um canto da parede, ao lado da porta. Quando uma pessoa atravessar a porta, o sensor captará movimento, acionando um alarme. Desenvolva um *script* para este pequeno projeto, de tal forma que ele acione um *buzzer* quando uma pessoa atravessar a porta.

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

10.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito que utilize a função *millis()* e imprima no *Serial Monitor* o tempo decorrido entre duas vezes sucessivas em que o sensor captou um movimento (tempo ocioso). Na primeira vez, o *buzzer* deve ser ativado. Na segunda vez, o *buzzer* deve ser desativado.

11 Comunicação óptica IR e controle remoto IR.

11.1 Tipos de interação

- <Entrada> / <sensor>: <nenhuma> / <nenhum>.
- <Saída> / <atuador>: <digital> / <LED>.
- <Comunicação> / <elemento>: <serial> / <*Serial Monitor* e receptor IR>.

11.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Comunicação óptica IR e controle remoto IR:
    - Usando dispositivos semicondutores, é possível implementar um transmissor de sinal IR (LED IR) e um receptor de sinal IR (fototransistor).
    - Para minimizar erros de transmissão, os dados a serem transmitidos devem ser codificados em símbolos adequadamente escolhidos.
    - Utilizando-se sinais IR para transmitir os símbolos desejados, pode-se estabelecer uma comunicação óptica simples.
    - Dispositivos de controle remoto IR enviam comandos codificados em símbolos, do transmissor para o receptor, empregando um sinal óptico IR.
  - O Arduino pode ser programado para utilizar uma biblioteca de funções destinadas a interagir com um tipo de receptor de controle remoto IR.
  - O Arduino pode ser programado para se comunicar com o *Serial Monitor* e mostrar os códigos enviados por um transmissor de controle remoto IR.
  - O Arduino pode ser programado para atuar sobre um LED comum e sinalizar o recebimento de um determinado comando por controle remoto.

- Objetivo: Introduzir os conceitos básicos sobre uma comunicação óptica IR simples, o funcionamento de um controle remoto IR, a possibilidade de sinalização de um evento e o funcionamento do *software Serial Monitor*.
- Conteúdo abordado:
  - Conceitos básicos sobre uma comunicação óptica IR simples.
  - Conceitos básicos sobre o funcionamento de um controle remoto IR.
  - Conceitos básicos sobre a possibilidade de sinalização de um evento.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.
- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender o mecanismo de uma comunicação óptica IR simples, o funcionamento de um controle remoto IR, a possibilidade de sinalização de um evento e como utilizar o *Serial Monitor* para se comunicar com o Arduino.
- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, transmissor e receptor de controle remoto, LEDs e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Explicar os conceitos básicos sobre uma comunicação óptica IR simples.
  2. Explicar os conceitos básicos sobre o funcionamento de um controle remoto IR.
  3. Explicar os conceitos básicos sobre a possibilidade de sinalização de um evento.
  4. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  5. Revisar o básico da programação do Arduino.
  6. Apresentar um código exemplo.
  7. Propor a expansão do exemplo apresentado.
  8. Aplicar “Questionário de Aula”.

11.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Na prática de hoje, utilizamos o controle remoto infravermelho. Para realizarmos o procedimento, utilizamos um *script* para identificarmos o código de cada tecla do teclado.

   Copie o *script* utilizado no espaço abaixo e identifique o código de 6 teclas do teclado.

   Tecla / código:

   1 - .....

   2 - .....

   3 - .....

   4 - .....

   5 - .....

   6 - .....

   Script:

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

2. Com as teclas identificadas na questão anterior, crie um *script* que acione o Arduino com o controle remoto, de forma que ele alterne entre 6 efeitos luminosos sobre 5 LEDs. Apresente o código proposto.

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

11.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito que leia os valores associados a duas teclas numéricas, pressionadas em um teclado de controle remoto IR, e adicione-os. Em seguida, imprima os valores e o resultado da adição no *Serial Monitor*.

12 Acionamento de *display* de cristal líquido.

12.1 Tipos de interação

- Entrada/sensor: analógica/sensor de temperatura.
- Saída/atuador: digital/*display* LCD 16x2.
- Comunicação/elemento: serial/*Serial Monitor*.

12.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Grandezas analógicas, grandezas digitais e digitalização (amostragem e quantização):
    - Uma relação funcional $y = f(x)$ é considerada analógica quando as variáveis y e x são contínuas.
    - Por sua vez, uma relação funcional $z = g[n]$ é considerada digital quando as variáveis z e n são discretas e cada um dos seus valores pode ser representado por um número finito de símbolos.
    - A conversão de uma relação analógica em digital, denominada de digitalização, envolve um processo de discretização, aplicado sobre as variáveis y e x, a fim de gerar as variáveis z e n, respectivamente.
    - Basicamente, a discretização significa tomar alguns valores individuais de uma grandeza contínua, dentro de uma determinada faixa de valores.
    - A discretização da variável independente x é denominada amostragem (*sampling*). Ao conjunto de valores individuais de x, pode ser associado um conjunto de índices inteiros n.
    - A discretização da variável dependente y é denominada quantização. Ao conjunto de valores individuais de y, pode ser associado um conjunto de valores z, aproximados com um número finito de símbolos.

    * Logo, pode-se dizer que a digitalização converte um conjunto infinito de valores, não completamente representáveis, em uma sequência finita de valores aproximados, perfeitamente representáveis.
    * No caso de uma discretização uniforme, os valores individuais são tomados com base em um intervalo constante, denominado de resolução da discretização (r). Assim, podem ser definidas a resolução da amostragem (r_S) e a resolução da quantização (r_Q)
- Uso de conversor analógico-digital:
    * Por vezes, em tarefas de controle, é necessário captar dados analógicos, realizar cálculos sobre eles e tomar decisões a partir disso.
    * Porém, os dados são armazenados e manipulados, bem como as decisões são tomadas, em ambiente digital.
    * Existem circuitos eletrônicos que, utilizando diferentes técnicas, são capazes de implementar a digitalização. Um tal circuito é denominado Conversor A/D (*Analog-to-Digital Converter* ou *ADC*).
    * Os microcontroladores costumam incorporar um ADC em seus circuitos internos.
- Uso de sensor de temperatura:
    * Na Física, a temperatura é uma grandeza associada à energia cinética média das partículas que compõem um sistema em equilíbrio térmico.
    * Por sua vez, calor é energia, que pode ser transferida entre porções de matéria que estão em contato.
    * Quando duas porções de matéria são colocadas em contato, há uma transferência de energia (calor) daquela que possui a temperatura mais elevada para aquela que possui a temperatura mais baixa.
    * Na percepção humana, a temperatura está relacionada às sensações definidas como frio (perda de calor) e quente (ganho de calor).
    * Os seguintes sensores de temperatura são comumente utilizados:
        · DHT11 e DHT22/AM2302: que possuem um sensor de umidade capacitivo e um sensor de temperatura termorresistivo (termistor) com característica NTC (*Negative Temperature Coefficient*).
        · LM35: que é um sensor baseado em semicondutor, com característica PTC (*Positive Temperature Coefficient*), dada por 10 mV/^{o}C.
        · TMP35/36/37: que é um sensor baseado em semicondutor, com característica PTC (*Positive Temperature Coefficient*), dada por [10;10;20] mV/^{o}C, *offset* de saída dado por [0;0,5;0] mV e saída para 25^{o}C dada por [250,750,500] mV.
    * A temperatura pode ser expressa em diversas unidades diferentes, tais como: Kelvin (K), grau centígrado ou Celsius (^{o}C) e grau Fahrenheit (^{o}F).
    * Conhecendo-se as relações entre duas unidades diferentes, funções podem ser definidas para realizar o mapeamento entre elas, tais como:
      K = C + 273,15 e F = (9/5) C + 32.
- Uso de *display* LCD:
    * Por vezes, em um sistema de controle, é necessário atuar sobre um visualizador simples, a fim de expor uma informação simples e específica.
    * Um *display* LCD (*Liquid-Cristal Display*) é um dispositivo adequado para este tipo de aplicação.

    - Um cristal líquido é um material que pode fluir como um líquido, mas as suas moléculas podem ser orientadas como em um cristal sólido.
    - Um *display* LCD gera ou oculta uma imagem a partir do uso de material polarizador de luz e da propriedade de modulação da luz por cristais líquidos.
    - Um *display* LCD possui a forma de um painel plano. Ele é composto por várias camadas, entre elas: polarizador de luz; vidro com eletrodos para formação dos caracteres; cristal líquido; fonte de luz, por reflexão e/ou por geração.
    - Os displays LCD podem ser monocromáticos ou coloridos e podem ser construídos em diversos tamanhos.
    - Alguns displays LCD simples, de uso geral, são organizados na forma matricial de caracteres genéricos e comumente denominados "LCD <colunas>x<linhas>". Por exemplo, LCD 16x2.
  - O Arduino pode ser programado para utilizar o ADC interno ao seu microcontrolador a fim de monitorar uma grandeza analógica.
  - Assim como em linguagens de programação de uso geral, podem-se empregar funções definidas pelo usuário em um código para o Arduino.
  - O Arduino pode ser programado para utilizar uma biblioteca de funções destinadas a interagir com um tipo de *display* LCD 16x2.

- Objetivo: Introduzir os conceitos básicos sobre o processo de conversão analógico-digital, o monitoramento de um sensor analógico, o funcionamento de um sensor de temperatura, o funcionamento de um *display* LCD, o procedimento para criação e uso de funções e o funcionamento do *software Serial Monitor*.

- Conteúdo abordado:
  - Conceitos básicos sobre o processo de conversão analógico-digital.
  - Conceitos básicos sobre o funcionamento de um sensor de temperatura.
  - Conceitos básicos sobre o funcionamento de um *display* LCD,
  - Conceitos básicos sobre o procedimento para criação e uso de funções.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de funções básicas, da biblioteca `LiquidCrystal.h`, para interação com o *display* LCD 16x2:
    - `LiquidCrystal lcd(RS, E, D4, D5, D6, D7)`
    - `lcd.begin(total de colunas, total de linhas)`
    - `lcd.setCursor(coluna, linha)`
    - `lcd.print(string)`
    - `lcd.clear()`
    - `lcd.scrollDisplayLeft()`
    - `lcd.scrollDisplayRight()`

- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender o processo de conversão analógico-digital, o monitoramento de um sensor analógico, o funcionamento de um sensor de temperatura, o funcionamento de um *display* LCD, o procedimento para criação e uso de funções e como utilizar o *Serial Monitor* para se comunicar com o Arduino.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, sensor de temperatura, resistores, *display* LCD 16x2.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Explicar os conceitos básicos sobre o processo de conversão analógico-digital.
  2. Explicar os conceitos básicos sobre o funcionamento de um sensor de temperatura.
  3. Explicar os conceitos básicos sobre o funcionamento de um *display* LCD.
  4. Explicar os conceitos básicos sobre o procedimento para criação e uso de funções.
  5. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  6. Revisar o básico da programação do Arduino.
  7. Apresentar um código exemplo.
  8. Propor a expansão do exemplo apresentado.
  9. Aplicar "Questionário de Aula".

12.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Na aula de hoje, aprendemos a trabalhar com o *display* de cristal líquido (LCD 16x2). Na Figura 8, temos a imagem do LCD usado em aula. Identifique a função de cada pino.

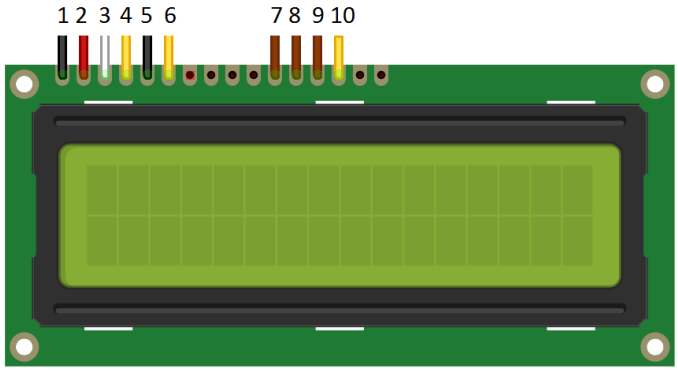

Figura 8: *Display* de cristal líquido LCD 16x2.

Pino / função:

1 - .....
2 - .....
3 - .....
4 - .....
5 - .....
6 - .....
7 - .....
8 - .....
9 - .....
10 - .....

2. Utilizando o LCD 16x2, crie um *script* que exiba o valor de temperatura lido pelo Arduino, a partir de um sensor de temperatura.

12.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito que leia um sensor de temperatura, escreva "Quente" no LCD quando o valor lido de temperatura for maior ou igual a um valor definido por você e escreva "Frio", caso contrário.

13 Interação com circuito integrado externo e acionamento de *display* de 7 segmentos.

13.1 Tipos de interação

- Entrada/sensor: nenhuma/nenhum.
- Saída/atuador: digital/CI conversor de códigos e *display* de 7 segmentos.
- Comunicação/elemento: nenhuma/nenhum.

13.2 Plano de aula

- Motivações:
  - Uso de *display* de 7 segmentos:
    - Por vezes, em um sistema de controle, é necessário atuar sobre um visualizador simples, a fim de expor uma informação simples e específica.
    - Um *display* de 7 segmentos é um dispositivo adequado para este tipo de aplicação.
    - Um *display* de 7 segmentos é composto por um conjunto de oito elementos, que são: sete segmentos lineares e um ponto.
    - Apesar de serem extremamente adequados à exibição de números decimais, também é possível, com a combinação adequada, exibir outros caracteres.
    - Com a intenção de exibir um conjunto maior de caracteres, bem como para compô-los mais facilmente, já foram construídos *displays* de 8, 9, 14 e 16 segmentos.
  - Conversão de códigos:
    - Por vezes, é necessário compatibilizar dois códigos diferentes, adotados por dois subsistemas diferentes.
    - Por vezes, para diminuir a quantidade de informação a ser transmitida, de forma serial ou paralela, é necessário modificar o código utilizado, gerando símbolos mais adequados.
    - A conversão de códigos pode ser realizada por *software*. Porém, uma solução de *hardware* tem a vantagem de facilitar a conversão de códigos e otimizar a transmissão dos dados.
    - Um sistema conversor de códigos também é genericamente conhecido como decodificador (*decoder*).
  - Conversão BCD-8421 para 7 segmentos:
    - Em alguns sistemas, os símbolos númericos são representados pelo código BCD-8421. Nesse código, os dez símbolos numéricos decimais são representados por seus equivalentes binários, com quatro dígitos binários (*bits*).
    - No caso do acionamento de um *display* de 7 segmentos a partir de um código BCD-8421, é necessário realizar a conversão entre os dois códigos.
    - Um CI que faz a conversão necessária é genericamente denominado *BCD-to-7-segment decoder*. O CI CD4511 é um exemplo de tal conversor.
  - O Arduino pode ser programado para interagir com um CI conversor de códigos, que, por sua vez, irá atuar sobre um dispositivo final.

- Objetivo: Introduzir os conceitos básicos sobre o funcionamento de um *display* de 7 segmentos, a conversão de códigos e o funcionamento de um CI conversor de códigos.
- Conteúdo abordado:
  - Conceitos básicos sobre o funcionamento de um *display* de 7 segmentos.
  - Conceitos básicos sobre a conversão de códigos.
  - Conceitos básicos sobre o funcionamento de um CI conversor de códigos.
  - Breve revisão sobre a programação do Arduino.
- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender o funcionamento de um *display* de 7 segmentos, a conversão de códigos e o funcionamento de um CI conversor de códigos.
- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, CI conversor de códigos.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Explicar os conceitos básicos sobre o funcionamento de um *display* de 7 segmentos.
  2. Explicar os conceitos básicos sobre a conversão de códigos.
  3. Explicar os conceitos básicos sobre o funcionamento de um CI conversor de códigos.
  4. Revisar o básico da programação do Arduino.
  5. Apresentar um código exemplo.
  6. Propor a expansão do exemplo apresentado.
  7. Aplicar "Questionário de Aula".

13.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

Neste projeto, é utilizado o circuito integrado CD4511, mostrado na Figura 9. Ele é um decodificador, do código BCD-8421 para o código de acionamento de um *display* de 7 segmentos, ilustrado na Figura 10. Isso possibilita a conexão de um Arduino com um *display* de 7 segmentos utilizando um número menor de pinos.

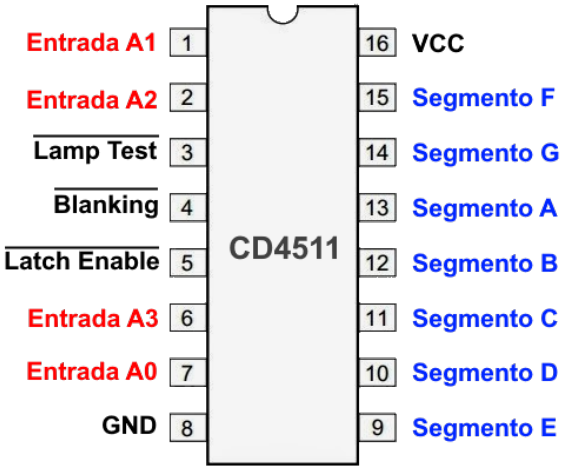

Figura 9: Decodificador CD4511, do código BCD-8421 para um *display* de 7 segmentos.

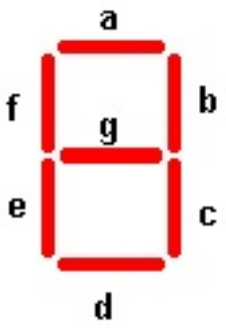

Figura 10: *Display* de 7 segmentos.

Atenda aos seguintes itens:

1. Apresente os valores binários do código BCD-8421 que geram os algarismos de 0 a 9 no *display* de 7 segmentos:

   Valor decimal / valor BCD-8421:

   0 - .....
   1 - .....
   2 - .....
   3 - .....
   4 - .....
   5 - .....
   6 - .....
   7 - .....
   8 - .....
   9 - .....

2. Elabore um código Arduino que realize a contagem de 0 a 9 e, posteriormente, de 9 a 0, com o *display* de 7 segmentos.

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

13.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito contendo 3 botões que, ao serem pressionados, provoquem o aparecimento de três dígitos numéricos, simples, diferentes e predefinidos, em um *display* de 7 segmentos.

14 Comunicação RF.

14.1 Tipos de interação

- <Entrada> / <sensor>: <nenhuma> / <nenhum>.
- <Saída> / <atuador>: <digital> / <LED comum>.
- <Comunicação> / <elemento>: <serial> / <*Serial Monitor*, transmissor e receptor RF>.

14.2 Plano de aula

- Motivações:
  - Uso de comunicação serial:
    - De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    - De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    - Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    - O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    - Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    - O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Comunicação, transmissor e receptor RF:
    - As radiações eletromagnéticas podem ser classificadas de acordo com as faixas de frequências consideradas. Uma classificação adotada é denominar a faixa 30 kHz $\leq f \leq$ 1 GHz de radiofrequência (RF).
    - Uma observação interessante sobre a faixa de RF é que ela está fora das faixas de frequências audíveis (20 Hz $\leq f \leq$ 20 kHz) e visíveis (390 THz $\leq f \leq$ 790 THz) ao ser humano.
    - Por não serem perceptíveis ao ser humano, podem-se empregar as radiações eletromagnéticas de RF para estabelecer comunicações à distância.
    - Para minimizar erros de transmissão, os dados a serem transmitidos devem ser codificados em símbolos adequadamente escolhidos.
    - Utilizando-se sinais RF para transmitir os símbolos desejados, pode-se estabelecer uma comunicação RF simples.
    - Por exemplo, dispositivos de controle remoto RF enviam comandos codificados em símbolos, do transmissor para o receptor, empregando um sinal RF.
  - O Arduino pode ser programado para utilizar uma biblioteca de funções destinadas a interagir com um tipo de transmissor e de receptor RF.

  - O Arduino pode ser programado para se comunicar com o *Serial Monitor* e mostrar os códigos enviados/recebidos por um transmissor/receptor RF.
  - O Arduino pode ser programado para atuar sobre um LED comum e sinalizar o recebimento de um determinado código.

- Objetivo: Introduzir os conceitos básicos sobre uma comunicação RF simples, o funcionamento de um transmissor/receptor RF, a possibilidade de sinalização de um evento e o funcionamento do *software Serial Monitor*.

- Conteúdo abordado:
  - Conceitos básicos sobre uma comunicação RF simples.
  - Conceitos básicos sobre o funcionamento de um transmissor/receptor RF.
  - Conceitos básicos sobre a possibilidade de sinalização de um evento.
  - Conceitos básicos sobre o *software Serial Monitor*.
  - Breve revisão sobre a programação do Arduino.
  - Apresentação de funções básicas, da biblioteca `VirtualWire.h`, para interação com um tipo de transmissor e de receptor RF.

- Resultados esperados: Ao final da experiência, o aluno deverá ser capaz de entender uma comunicação RF simples, o funcionamento de um transmissor/receptor RF, a possibilidade de sinalização de um evento e como utilizar o *Serial Monitor* para se comunicar com o Arduino.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, transmissor e receptor RF, LEDs e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.

- Metodologia:
  1. Explicar os conceitos básicos sobre uma comunicação RF simples.
  2. Explicar os conceitos básicos sobre o funcionamento de um transmissor/receptor RF.
  3. Explicar os conceitos básicos sobre a possibilidade de sinalização de um evento.
  4. Explicar os conceitos básicos sobre o *software Serial Monitor*.
  5. Revisar o básico da programação do Arduino.
  6. Apresentar um código exemplo.
  7. Propor a expansão do exemplo apresentado.
  8. Aplicar “Questionário de Aula”.

14.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

1. Identifique a pinagem do Receptor RF da Figura 11.

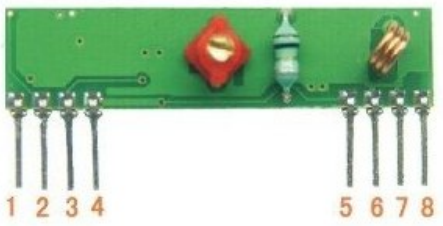

Figura 11: Receptor RF.

Pino / função:

1 - .....
2 - .....
3 - .....
4 - .....
5 - .....
6 - .....
7 - .....
8 - .....

2. Identifique a pinagem do Transmissor RF da Figura 12.

Figura 12: Transmissor RF.

Pino / função:

1 - .....
2 - .....
3 - .....
4 - .....

3. Elabore um *script* para ler um receptor RF e acender diferentes LEDs, um para cada mensagem distinta recebida.

14.4 Miniprojeto semanal

Elabore um *script* Arduino e um circuito que envie um valor de tensão, lido de um potenciômetro, de um Arduino transmissor a um Arduino receptor.

Ao receber um valor de tensão superior a 3,0 V, o Arduino receptor deverá acender um LED vermelho. Caso contrário, o LED deverá permanecer apagado.

15 Jogo Genius.

15.1 Tipos de interação

- <Entrada>/<sensor>: <digital>/<chave (botão)>.
- <Saída>/<atuador>: <digital>/<LED comum e sonorizador *buzzer* ativo e/ou passivo>.
- <Comunicação>/<elemento>: <serial>/<*Serial Monitor*>.

15.2 Plano de aula

- Motivações:
  - Uso de dispositivo mecânico de chaveamento como elemento de sensoriamento:
    - Dispositivos mecânicos de chaveamento geram oscilações (*bouncing*).
    - Uso de botão como dispositivo mecânico de chaveamento.
    - Uso de resistores (*pullup* e *pulldown*) para fixação de valores de tensão elétrica.
  - Uso de LED como elemento de atuação:
    - Como o próprio nome já indica, o LED (*Light-Emitting Diode*) é um dispositivo semicondutor capaz de emitir luz.
    - Tal característica faz dele um dispositivo naturalmente utilizado para simples iluminação, sinalização de alguma condição e até mesmo para comunicação à distância.
    - LEDs são capazes de irradiar em diversas faixas, tais como: infravermelho, ultravioleta e luz visível (vermelho, amarelo, verde, azul e violeta).
    - A radiação luminosa de um LED não é monocromática, como em um *laser*, mas a faixa irradiada é bem estreita, gerando a sensação de uma única cor.
    - Os LEDS comuns são baseados em único diodo, emitindo apenas em uma única faixa.
    - Os LEDS RGB (*Red*, *Green*, *Blue*) são compostos por três diodos, possibilitando a geração de cores compostas.
    - Uma vez que um LED é uma junção semicondutora, ele não é capaz de controlar a corrente elétrica que passa por ele. Para realizar tal função, pode-se empregar um resistor, em uma ligação série com o LED. O controle da corrente elétrica tem as seguintes funções: proteção contra danos e controle da intensidade da luz emitida.
  - Uso de *buzzer* como elemento de atuação:
    - De acordo com o seu acionamento, um *buzzer* pode ser classificado nos seguintes tipos: ativo e passivo.
    - Um *buzzer* ativo possui circuito eletrônico de acionamento interno, contendo um oscilador senoidal, com frequência única e predefinida. Logo, para acioná-lo, basta aplicar sobre ele apenas um valor de tensão constante.
    - Um *buzzer* passivo não possui circuito eletrônico de acionamento interno. Logo, é necessário aplicar sobre ele um sinal senoidal com a frequência desejada.

  - Uso de comunicação serial:
    * De acordo com a quantidade de canais de transmissão de dígitos binários (*bits*), uma interface de comunicação digital pode ser classificada como paralela (vários canais) ou serial (um único canal).
    * De acordo com a direcionalidade da comunicação, uma interface de comunicação pode ser classificada como *simplex* (um canal único e um sentido único), *half-duplex* (um canal único e dois sentidos) e *full-duplex* (dois canais e um sentido em cada canal).
    * Em uma interface de comunicação digital, serial e *full-duplex*, deve-se contar com um canal binário transmissor (TX) e com um canal binário receptor (RX).
  - Interação com o *software Serial Monitor*:
    * O *Serial Monitor* é um *software* que faz parte do IDE do Arduino.
    * Ele tem interação direta com periféricos do computador, tais como: o teclado, o monitor e a interface de comunicação serial USB.
    * O *Serial Monitor* pode ser usado para estabelecer uma comunicação entre um usuário humano e o Arduino.
  - Alguns microcontroladores possuem um resistor interno que pode ser habilitado por *software* para uso externo (resistor interno de *pullup* ou de *pulldown*).
  - O Arduino pode ser programado para gerar um sinal digital de saída, com dois níveis de tensão elétrica. Tal sinal pode ser usado para o acionamento de um *buzzer* ativo e um LED comum.
  - O Arduino pode ser programado para gerar uma onda quadrada na saída, com uma frequência desejada. Tal sinal pode ser usado para o acionamento de um *buzzer* passivo.
  - O Arduino pode ser programado para se comunicar com o *Serial Monitor*.

- Objetivo: Introduzir os conceitos básicos sobre a interação com chaves, LEDs comuns, *buzzers* e o *Serial Monitor*, bem como sobre a forma de programar o Arduino para criar um *loop* de controle: sensoriamento, processamento de dados e atuação.

- Conteúdo abordado:
  - Funcionamento de chaves e botões.
  - Definição do efeito de *bouncing*.
  - Conceito de resistor de *pullup* e de *pulldown*.
  - Funcionamento e operação de um LED comum.
  - Funcionamento e operação de um *buzzer*.
  - Revisão de algumas funções básicas do Arduino, para sensoriamento, processamento de dados e atuação.

- Resultados esperados: Ao final dessa experiência, o aluno deverá ser capaz de entender como interagir com chaves, LEDs comuns, *buzzers* e o *Serial Monitor*, bem como sobre a forma de programar o Arduino para criar um *loop* de controle: sensoriamento, processamento de dados e atuação.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *protoboard*, fios, chaves (botões), LEDs comuns, *buzzer*.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.
- Metodologia:
  1. Identificar alguns tipos de botões.
  2. Explicar o efeito de *bouncing*.
  3. Explicar o conceito de resistor de *pullup* e de *pulldown*.
  4. Identificar alguns tipos de LED.
  5. Identificar os tipos de *buzzer*.
  6. Explicar o funcionamento de um *buzzer*.
  7. Explicar o cálculo dos resistores de proteção.
  8. Definir o *loop* de controle a ser realizado.
  9. Revisar o básico da programação do Arduino.
  10. Apresentar as funções usadas para geração de sinal PWM de saída.
  11. Apresentar um código exemplo.
  12. Propor a expansão do exemplo apresentado.
  13. Aplicar "Questionário de Aula".

15.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

O Genius é um jogo que foi muito famoso na década de 1980 e que buscava estimular a memória do jogador por meio do uso de cores e de sons.

A prática de hoje trata de uma adaptação do Genius para o Arduino. Precisaremos dos seguintes componentes: 4 botões, 8 resistores de 330 Ω, 4 LEDs de cores diferentes e um *buzzer*.

A montagem do circuito é bem simples, sendo ilustrada na Figura 13. Porém, como temos uma quantidade maior de componentes do que nas práticas anteriores, é necessária uma maior atenção na montagem.

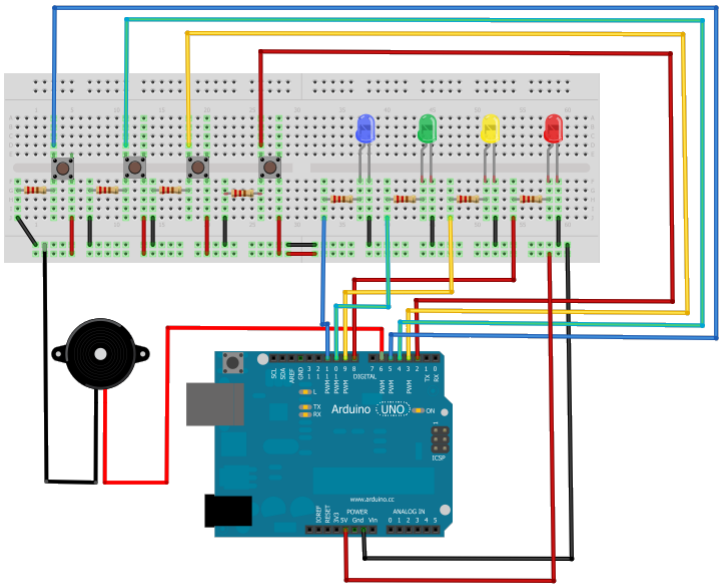

Figura 13: Circuito do jogo Genius.

15.4 Miniprojeto semanal

Adapte a prática de hoje, incorporando ao jogo um contador de pontos.

A cada rodada, o seu programa deverá mostrar, no *Serial Monitor*, quantos acertos o jogador possui. Quando acontecer um erro, ele deverá zerar a contagem.

Apresente o código proposto.

16 Atuação como *web server* auxiliada por *Ethernet shield*.

16.1 Tipos de interação

- Entrada/sensor: digital/chave (botão).
- Saída/atuador: digital/LED comum.
- Comunicação/elemento: diversos tipos/*Ethernet shield*.

16.2 Plano de aula

- Motivações:
  - Uso de dispositivo mecânico de chaveamento como elemento de sensoriamento:
    - Dispositivos mecânicos de chaveamento geram oscilações (*bouncing*).
    - Uso de botão como dispositivo mecânico de chaveamento.
    - Uso de resistores (*pullup* e *pulldown*) para fixação de valores de tensão elétrica.
  - Uso de LED como elemento de atuação:
    - Como o próprio nome já indica, o LED (*Light-Emitting Diode*) é um dispositivo semicondutor capaz de emitir luz.
    - Tal característica faz dele um dispositivo naturalmente utilizado para simples iluminação, sinalização de alguma condição e até mesmo para comunicação à distância.
    - LEDs são capazes de irradiar em diversas faixas, tais como: infravermelho, ultravioleta e luz visível (vermelho, amarelo, verde, azul e violeta).
    - A radiação luminosa de um LED não é monocromática, como em um *laser*, mas a faixa irradiada é bem estreita, gerando a sensação de uma única cor.
    - Os LEDS comuns são baseados em único diodo, emitindo apenas em uma única faixa.
    - Os LEDS RGB (*Red*, *Green*, *Blue*) são compostos por três diodos, possibilitando a geração de cores compostas.
    - Uma vez que um LED é uma junção semicondutora, ele não é capaz de controlar a corrente elétrica que passa por ele. Para realizar tal função, pode-se empregar um resistor, em uma ligação série com o LED. O controle da corrente elétrica tem as seguintes funções: proteção contra danos e controle da intensidade da luz emitida.
  - Uso de *Ethernet shield* como *web server*:
    - No âmbito do Arduino, *shields* são circuitos adicionais que podem ser acoplados a ele, para ampliar suas capacidades e/ou adicionar funcionalidades especializadas.
    - Um *Ethernet shield* insere o Arduino na Internet ou em uma rede local de forma simples. Fisicamente, basta conectar o *shield* no Arduino e ligar um cabo de rede ao *shield*.
    - Um *web server* pode ser definido como uma máquina computacional que abriga uma *webpage* ou um *website* na Internet.
    - O *Ethernet shield* é capaz de operar como um *web server*, interagindo com um usuário humano por meio de uma *webpage*, abrigada internamente por ele.

    - Utilizando os elementos de composição de uma *webpage*, comandos podem ser enviados para o Arduino e informações podem ser recebidas a partir dele, com a intermediação do *Ethernet shield*.
    - Portanto, com o auxílio de um *Ethernet shield*, pode-se estabelecer um *loop* de controle (sensoriamento, processamento e atuação), via Web, com participação humana.
  - Alguns microcontroladores possuem um resistor interno que pode ser habilitado por *software* para uso externo (resistor interno de *pullup* ou de *pulldown*).
  - O Arduino pode ser programado para gerar um sinal digital de saída, com dois níveis de tensão elétrica. Tal sinal pode ser usado para o acionamento de um LED comum.
  - O Arduino pode ser programado para interagir com um *Ethernet shield*, que, por sua vez, poderá atuar como um *web server*.

- Objetivo: Introduzir os conceitos básicos sobre a interação com chaves e LEDs comuns, bem como sobre a forma de programar o Arduino para interagir com um *Ethernet shield* e criar um *web server* associado a um *loop* de controle.

- Conteúdo abordado:
  - Funcionamento de chaves e botões.
  - Definição do efeito de *bouncing*.
  - Conceito de resistor de *pullup* e de *pulldown*.
  - Funcionamento e operação de um LED comum.
  - Funcionamento e operação de um *Ethernet shield*.
  - Revisão de algumas funções básicas do Arduino, para sensoriamento, processamento de dados e atuação.
  - Apresentação de funções básicas, das bibliotecas `SPI.h` e `Ethernet.h`, para interação com um tipo de *Ethernet shield*.

- Resultados esperados: Ao final dessa experiência, o aluno deverá ser capaz de entender como monitorar uma chave, acionar um LED RGB e programar o Arduino para interagir com um *Ethernet shield* e criar um *web server* associado a um *loop* de controle.

- Material utilizado:
  - *Hardware*: *kit* Arduino, *Ethernet shield*, *protoboard*, fios, chaves (botões), LEDs comuns e resistores.
  - *Software*: IDE do Arduino.
  - Material didático: *slides* e apostilas.

- Metodologia:
  1. Identificar alguns tipos de botões.
  2. Explicar o efeito de *bouncing*.
  3. Explicar o conceito de resistor de *pullup* e de *pulldown*.
  4. Identificar alguns tipos de LED.

5. Explicar o cálculo dos resistores de proteção.
6. Revisar o básico da programação do Arduino.
7. Apresentar as funções usadas para geração e controle da *webpage* no *Ethernet shield*.
8. Apresentar um código exemplo.
9. Propor a expansão do exemplo apresentado.
10. Aplicar "Questionário de Aula".

16.3 Questionário de aula

Data: .....
Aluno: .....
Aluno: .....

Na aula de hoje, trabalhamos um projeto utilizando o Arduino e um *Ethernet shield*. Vimos que, para acionarmos diferentes cargas, possuímos diferentes circuitos, um para cada finalidade.

1. Suponha que você possua uma sirene que deve ser alimentada por uma fonte de 12 V (DC). Apresente o circuito que você utilizaria para acioná-la com o Arduino.

2. Suponha que você possua um ventilador que deve ser alimentado por uma fonte de 110 V (AC). Apresente o circuito que você utilizaria para acioná-lo com o Arduino.

16.4 Miniprojeto semanal

Imagine que você possua uma casa de praia e que, nela, exista um poço de água potável. Recentemente, você esteve lá e deixou a caixa d'água vazia. No momento, você está em sua moradia, mas pretende voltar lá, em breve. Para não faltar água logo que você chegar na casa, você resolve ligar a bomba da sua casa de praia a partir de um projeto com o Arduino.

Elabore um *script* que acione a bomba e que, ao identificar a caixa cheia, desligue-a, avisando a você que a operação foi executada com sucesso.