

# Programação em Linguagem C

## Unidade 2

Nesta unidade estudaremos:

- ✓ If/else;
- ✓ for;
- ✓ while;
- ✓ switch case;
- ✓ Comunicação serial com PC;
- ✓ “apelidos” para os pinos do Arduino.

## Relembrando...

### Caso geral da instrução if-else:

```
if( expressão ){  
    instrução1;  
}else{  
    instrução2;  
}
```

O if-else funciona do seguinte modo:

1. O valor da **expressão** é calculado;
2. Se for verdadeiro, a **instrução1** será executada, mas a **instrução2** não;  
Se for falso, a **instrução2** será executada, mas a **instrução1** não.

**Caso geral da instrução for:**

```
for(expressão1; expressão2; expressão3 ){  
    instrução;  
}
```

A instrução for funciona do seguinte modo:

1. A **expressão1** é executada apenas na primeira vez que o programa passa pelo ciclo for.
2. O valor da **expressão2** é calculado e se for verdadeiro, o programa executa a **expressão3** e as **instruções** dentro das chaves.
3. O processo repete-se, isto é, o valor da **expressão2** é calculado novamente. Se for verdadeiro, o programa executa a **expressão3** e as **instruções** dentro das chaves.
4. O ciclo **for** termina quando a **expressão2** for falsa.

**Caso geral da instrução while:**

```
while(expressão) {  
    instrução;  
}
```

A instrução while funciona do seguinte modo:

A **expressão** é testada, se for verdadeira, o programa executa a **instrução**. Depois de executar a **instrução**, o programa testa novamente a **expressão**, executa a **instrução** (caso verdadeira) e continua assim indefinidamente até que a **expressão** se torne falsa.

### Caso geral da instrução switch case:

```
switch(expressão){  
    case 3:  
        instrução1;  
        break;  
    case 5:  
        instrução2;  
        break;  
    default:  
        instrução10; break;  
}
```

A instrução switch case funciona do seguinte modo:

1. Logo após a palavra **switch**, calcula-se o valor **expressão**.
2. Baseado nesse valor, o programa salta para o caso apropriado. Por exemplo, se o valor for 5: o programa salta para **case 5**, executa **instrução2** e prossegue com as instruções restantes até aparecer a instrução **break**. Esta instrução faz com que o computador salte para fora do switch.
3. O **default** é opcional e é executado se nenhum dos outros casos ocorrer.

**Obs.:** O switch case funciona também com caractere (letra).

**Exemplo 1:** Controlar o acionamento de um LED através de um botão de pulso.

Dados.: Botão de pulso conectado ao pino 7 do Arduino, envia nível lógico 0 quando pressionado. LED conectado ao pino 4 do Arduino, acende com nível lógico 1.

Funcionamento.:

- Inicialmente o LED deverá estar apagado;
- O LED deverá alterar seu estado (de ligado para desligado e vice-versa) toda vez que o botão for pressionado.

```

sketch_apr04a | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda
sketch_apr04a §
/*****
 * Neste exemplo, o botão envia nível lógico 0 quando for pressionado.
 * Portanto, enquanto o botão não for pressionado, ele envia nível lógico 1
 *****/

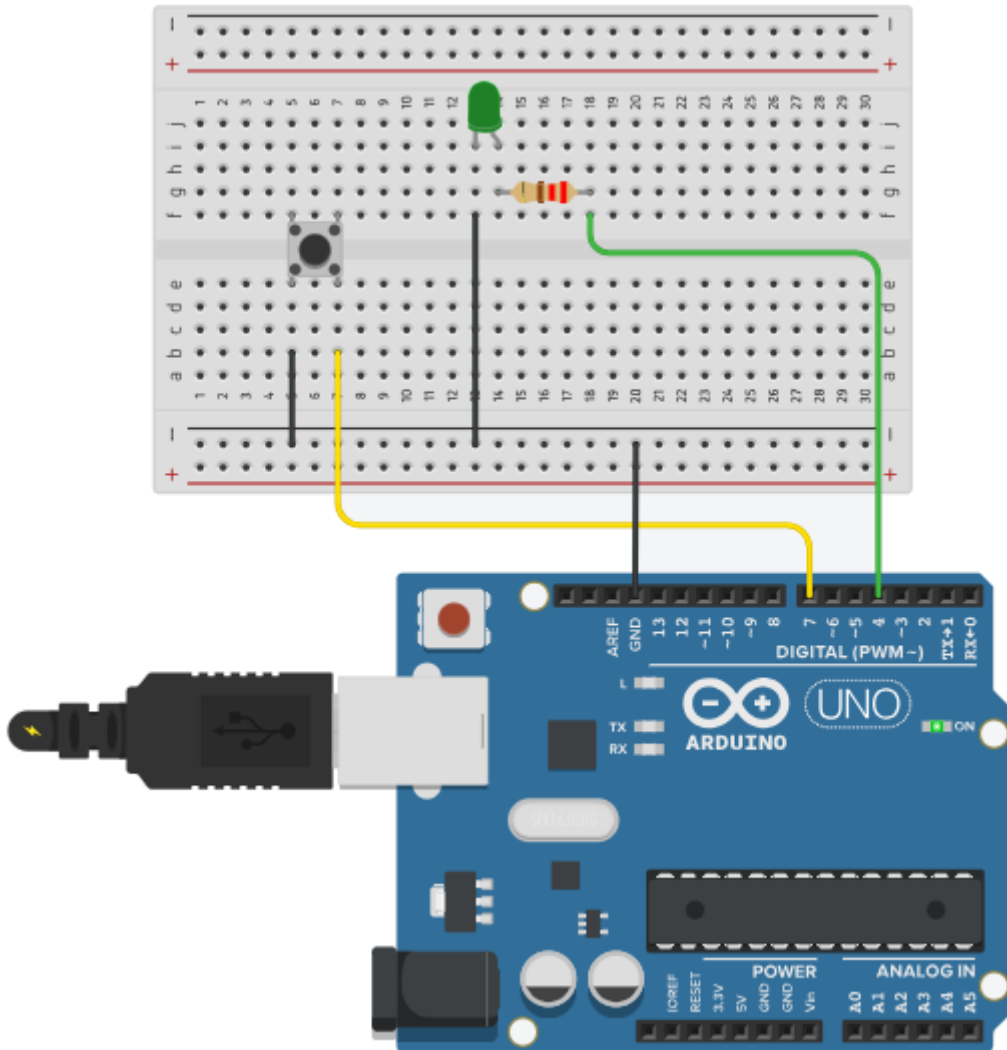
#define BOTA0 7 //Define o "apelido" de BOTA0 para o pino 7. Observe que não utilizamos o ;
#define LED 4 //Define o "apelido" de LED para o pino 4. Observe que não utilizamos o ;

void setup(){ //Configurações I/O (Entradas e Saídas)
  pinMode(BOTA0, INPUT_PULLUP); //Configura BOTA0 (pino 7) como entrada e ativa resistor de PULL UP
  pinMode(LED, OUTPUT); //Configura LED (pino 4) como saída
} //Fim do bloco de configuração I/O

void loop(){ //Repetição Infinita
  if(digitalRead(BOTA0)==1){ //Se o botão não for pressionado (pino 7 = 1);
    while(digitalRead(BOTA0)==1){ //fica em loop Enquanto o pino 7 permanecer em nível "1"
      digitalWrite(LED, LOW); //desliga o led (pino 4 em nível baixo)
    }
  } //Quando o botão for pressionado, (pino 7 = 0)
  delay(500); //Aguarda 500ms (tempo de soltar o botão)
  if(digitalRead(BOTA0)==1){ //Se o botão não for pressionado (pino 7 = 1);
    while(digitalRead(BOTA0)==1){ //fica em loop Enquanto o pino 7 permanecer em nível "1"
      digitalWrite(LED, HIGH); //desliga o led (pino 4 em nível baixo)
    }
  } //Quando o botão for pressionado novamente, (pino 7 = 0)
  delay(500); //Aguarda 500ms (tempo de soltar o botão) e volta p void loop().
}
27 Arduino/Genuino Uno em COM4

```

# Montagem



**Exemplo 2:** Faça um semáforo (com pedestre).

Dados:

Veicular: VERDE pino 0, AMARELO pino 1, VERMELHO pino 2;

Pedestre: VERDE\_P pino 4, VERMELHO\_P pino 5.

Funcionamento.:

- Os vermelhos devem permanecer ligados por 2s;
- Ligue o VERDE por 10s e depois ligue o AMARELO por 3s;
- Ligue o VERMELHO; Ligue o VERDE\_P por 2s;
- O VERMELHO\_P deve piscar 4x com intervalos de 1s.
- Retorne para o item b.



```

Semaforo | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Semaforo §
#define VERDE      0 //Define "apelido" VERDE para pino 0.  Observe que não utilizamos o ;
#define AMARELO    1 //Define "apelido" AMARELO para pino 1. Observe que não utilizamos o ;
#define VERMELHO   2 //Define "apelido" VERMELHO para pino 2. Observe que não utilizamos o ;
#define VERDE_P    4 //Define "apelido" VERDE_P para pino 4.  Observe que não utilizamos o ;
#define VERMELHO_P 5 //Define "apelido" VERMELHO_P para pino 5. Observe que não utilizamos o ;

void setup() {
    //Configurações I/O (Entradas e Saídas)
    pinMode(VERDE,      OUTPUT); //Configura VERDE      (pino 0) como saída
    pinMode(AMARELO,    OUTPUT); //Configura AMARELO    (pino 1) como saída
    pinMode(VERMELHO,   OUTPUT); //Configura VERMELHO   (pino 2) como saída
    pinMode(VERDE_P,    OUTPUT); //Configura VERDE_P    (pino 4) como saída
    pinMode(VERMELHO_P, OUTPUT); //Configura VERMELHO_P (pino 5) como saída
}

void loop() { //Inicio do bloco de loop (repetição)
    digitalWrite(VERMELHO, HIGH); //liga vermelho veicular (pino 2)
    digitalWrite(VERMELHO_P, HIGH); //liga vermelho pedestre
    delay(2000); //Aguarda 2 segundos

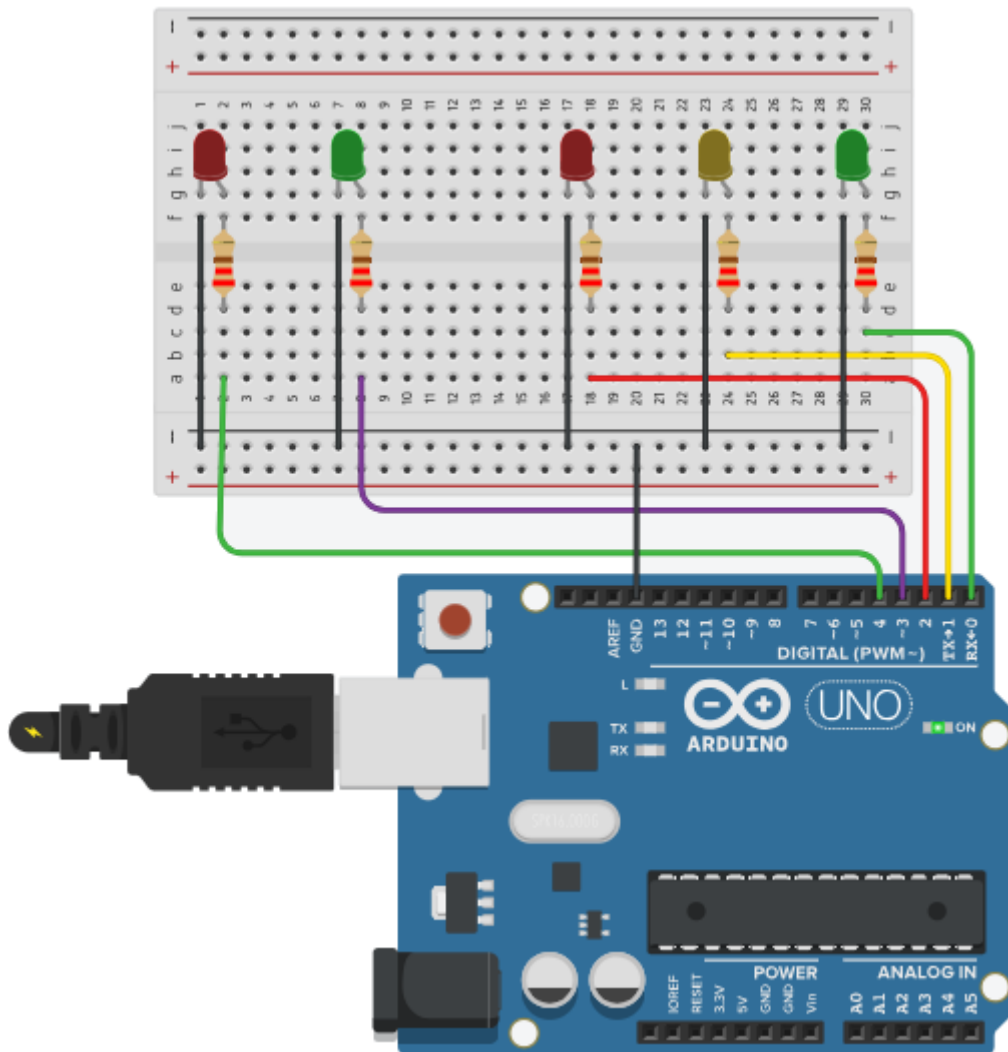
    while (true) { //while = Enquanto; true = Verdade; (loop infinito)
        digitalWrite(VERMELHO_P, HIGH); //liga vermelho pedestre (pino 5)
        digitalWrite(VERMELHO, LOW); //desliga vermelho veicular (pino 2)
        digitalWrite(VERDE, HIGH); //liga verde veicular (pino 0)
        delay(10000); //Aguarda 10 segundos
        digitalWrite(VERDE, LOW); //desliga verde veicular (pino 0)
        digitalWrite(AMARELO, HIGH); //liga amarelo veicular (pino 1)
        delay(3000); //Aguarda 3 segundos
        digitalWrite(AMARELO, LOW); //desliga amarelo veicular (pino 1)
        digitalWrite(VERMELHO, HIGH); //liga vermelho veicular (pino 2)
        digitalWrite(VERMELHO_P, LOW); //desliga vermelho pedestre (pino 5)
        digitalWrite(VERDE_P, HIGH); //liga verde pedestre (pino 4)
        delay(2000); //Aguarda 2 segundos
        digitalWrite(VERDE_P, LOW); //desliga verde pedestre (pino 4)

        for(int i=0; i<4; i++){ //Repete este laço 4x.
            digitalWrite(VERMELHO_P, HIGH); //liga vermelho pedestre (pino 5)
            delay(1000); //aguarda 1 segundos
            digitalWrite(VERMELHO_P, LOW); //desliga vermelho pedestre (pino 5)
            delay(1000); //aguarda 1 segundo
        } //fim do laço da repetição 4x
    }
}

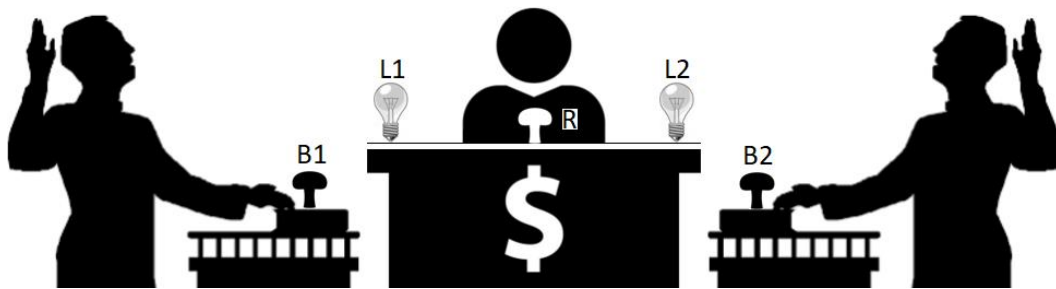
```

20 Arduino/Genuino Uno em COM3

Montagem



**Exemplo 3:** A imagem abaixo trata-se de um jogo de perguntas e respostas.



Deve ser feito o seguinte programa:

- Se o botão **B1** for pressionado primeiro que B2, a lâmpada **L1** deverá acender e somente deverá apagar quando o botão **R** for pressionado pelo apresentador;
- Se o botão **B2** for pressionado primeiro que B1, a lâmpada **L2** deverá acender e somente deverá apagar quando o botão **R** for pressionado pelo apresentador;
- Se uma lâmpada estiver acesa a outra não poderá acender.

```

Passa | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Passa §

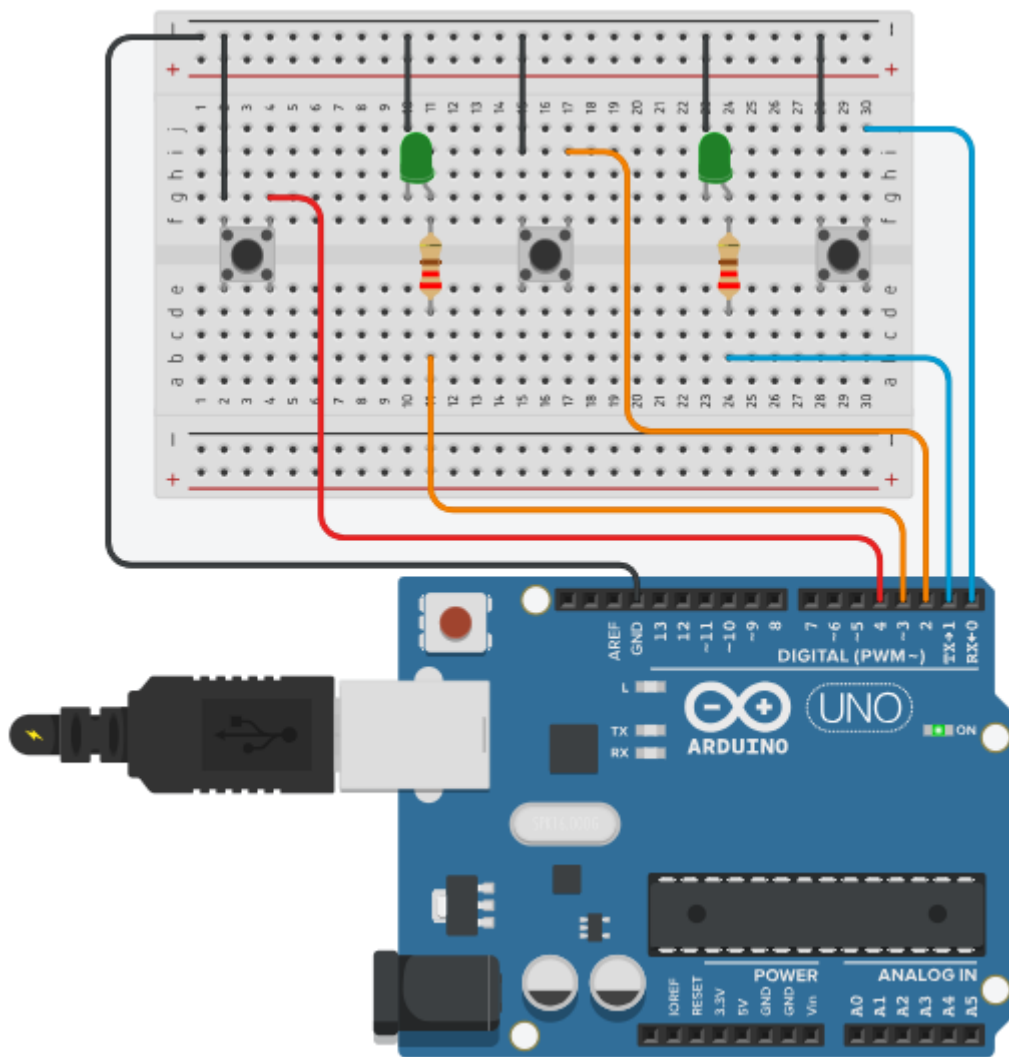
#define B1 0 //Define "apelido" B1 para pino 0. Observe que não utilizamos o ;
#define L1 1 //Define "apelido" L1 para pino 1. Observe que não utilizamos o ;
#define B2 2 //Define "apelido" B2 para pino 2. Observe que não utilizamos o ;
#define L2 3 //Define "apelido" L2 para pino 3. Observe que não utilizamos o ;
#define R 4 //Define "apelido" R para pino 4. Observe que não utilizamos o ;

void setup() { //Configurações I/O (Entradas e Saídas)
  pinMode(B1, INPUT_PULLUP); //Configura B1 (pino 0) como entrada
  pinMode(B2, INPUT_PULLUP); //Configura B2 (pino 2) como entrada
  pinMode(R, INPUT_PULLUP); //Configura R (pino 4) como entrada
  pinMode(L1, OUTPUT); //Configura L1 (pino 1) como saída
  pinMode(L2, OUTPUT); //Configura L2 (pino 3) como saída
} //Fim do bloco de configuração I/O

void loop() { //Inicio do loop infinito
  if (digitalRead(B1) == 0) { //lê B1 (pino 0) se for 0 (pressionado);
    digitalWrite(L1, HIGH); //Liga L1 (pino 1).
    while (digitalRead(R) == 1) { //Não faz nada enquanto R(pino 4) não for pressionado.
    } //Quando R for pressionado;
    digitalWrite(L1, LOW); //Desliga L1 (pino 1)
  } //Fim do "bloco" do botão B1
  if (digitalRead(B2) == 0) { //lê B2 (pino 2) se for 0 (pressionado);
    digitalWrite(L2, HIGH); //Liga L2 (pino 3).
    while (digitalRead(R) == 1) { //Não faz nada enquanto R(pino 4) não for pressionado.
    } //Quando R for pressionado;
    digitalWrite(L2, LOW); //Desliga L1 (pino 1)
  } //Fim do "bloco" do botão B2
} //Fim do loop infinito
  
```



Montagem



**Exemplo 4: Recebendo dados do PC e Tratando com o switch case**

Faça um programa que:

- Após receber a letra “T” enviada pelo computador, ligue o LED conectado ao pino 13 do Arduino;
- Após receber a letra “a”, pisque um LED conectado ao pino 7 três vezes;
- Após receber a letra “t”, ligue o LED conectado ao pino A0;
- Após receber a letra “A”, pisque um LED conectado ao pino A4 cinco vezes;
- Desligue todos os LED’s quando o Arduino receber a letra (P) enviada pelo computador;

```

Recebe_Dados | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Recebe_Dados $
//Enviando dados do PC para o Arduino
#define LED_1 13 //Define que o "apelido para o pino 13 do arduino será LED_1
#define LED_2 7 //Define que o "apelido para o pino 7 do arduino será LED_2
#define LED_3 A0 //Define que o "apelido para o pino A0 do arduino será LED_3
#define LED_4 A4 //Define que o "apelido para o pino A4 do arduino será LED_4
char caractere; //Cria um espaço na memória para armazenar os dados da variavel caractere.
int i = 0; //Cria um espaço na memória para armazenar um valor inteiro.
void setup() { //Configuração I/O
  Serial.begin(9600); //Configura a taxa da comunicação serial para 9600Bps
  pinMode(LED_1, OUTPUT); //Configura LED_1 (pino 13) como saída
  pinMode(LED_2, OUTPUT); //Configura LED_2 ( pino 7) como saída
  pinMode(LED_3, OUTPUT); //Configura LED_3 (pino A0) como saída
  pinMode(LED_4, OUTPUT); //Configura LED_4 (pino A4) como saída
} //Fim do bloco I/O
void loop() { //Inicio do Bloco de Repetição Infinita
  if (Serial.available()) { //Verificar se recebeu algum dado na porta serial. Se recebeu:
    caractere = Serial.read(); //Lê o dado recebido e armazena na variavel caractere.
  } //Após recebido o dado, ou não recebido dado;

  switch (caractere) { //Informação de caracter é tratada pela instrução switch case.

    case 'T': //Caso o dado armazenado em caractere seja a letra T;
      digitalWrite(LED_1, HIGH); break; //liga LED_1 (pino 13). Finaliza o bloco T com break

    case 'a': //Caso o dado armazenado em caractere seja a letra a;
      for (i = 0; i < 3; i++) { //faz i=0, enquanto i<3, faz as linhas abaixo e soma 1 em i
        digitalWrite(LED_2, HIGH); //Liga LED_2 (pino 7);
        delay(500); //Aguarda 500ms (meio segundo);
        digitalWrite(LED_2, LOW); //Desliga LED_2 (pino 7);
        delay(500); //Aguarda 500ms (meio segundo);
      } //Fim do laço de repetição for. (ESTUDE A ESTRUTURA FOR NO COMEÇO DESTA AQUIVO)
      break; //Finaliza o caso a e sai do switch.

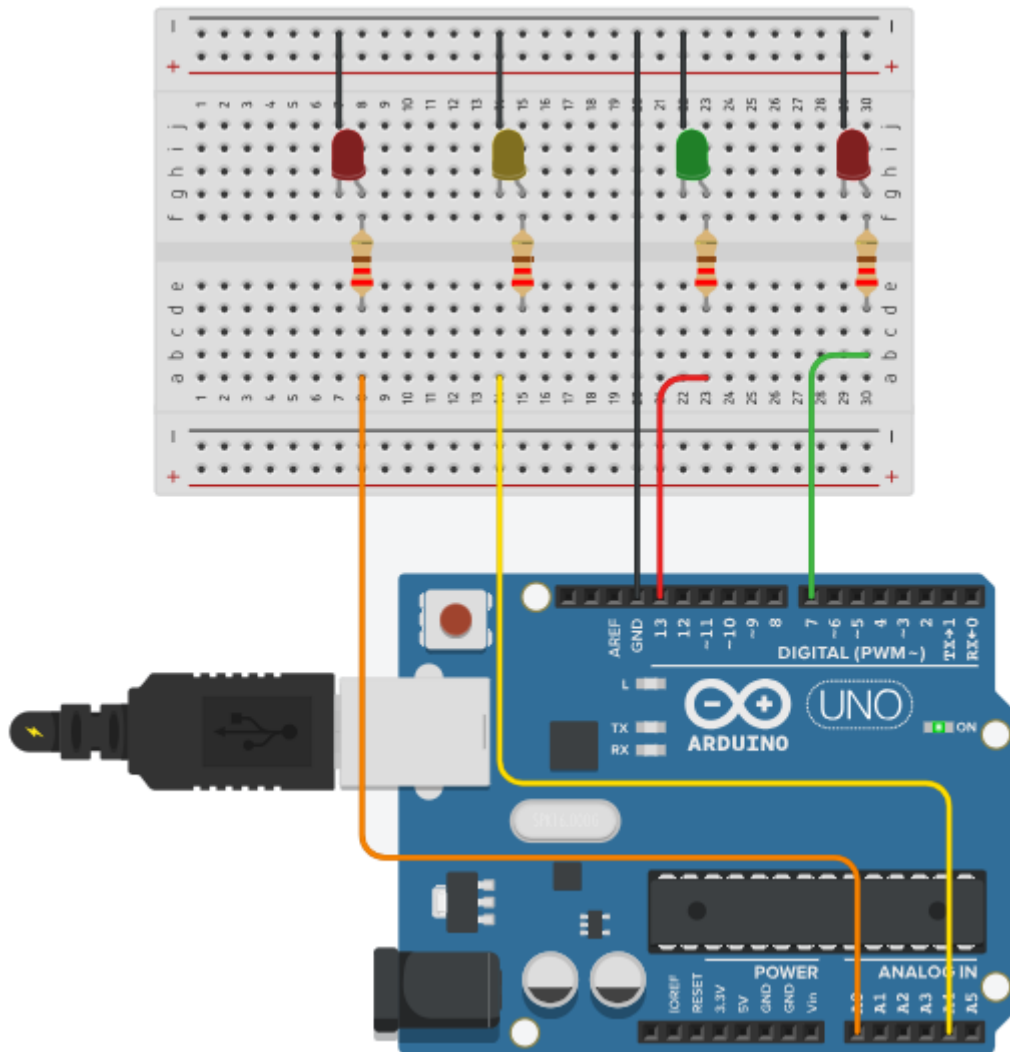
    case 't': //Caso o dado armazenado em caractere seja a letra t;
      digitalWrite(LED_3, HIGH); break; //Liga LED_3 (pino A0), Finaliza o bloco t com break

    case 'A': //Caso o dado armazenado em caractere seja a letra A;
      i = 0; //Antes de entrar no laço while, faz i=0;
      while (i < 5) { //Enquanto i<5, faz as instruções abaixo;
        digitalWrite(LED_4, HIGH); //Liga LED_4 (pino A4);
        delay(500); //Aguarda 500ms (meio segundo);
        digitalWrite(LED_4, LOW); //Desliga LED_4 (pino A4);
        delay(500); //Aguarda 500ms (meio segundo);
        i++; //Soma 1 na variavel i;
      } //Fim do laço de repetição while. (ESTUDE A ESTRUTURA WHILE NO COMEÇO DESTA AQUIVO)
      break; //Finaliza o caso A e sai do switch.

    case 'P': //Caso o dado armazenado em caractere seja a letra P;
      digitalWrite(LED_1, LOW); //Desliga LED_1 (pino 13)
      digitalWrite(LED_2, LOW); //Desliga LED_2 (pino 17)
      digitalWrite(LED_3, LOW); //Desliga LED_3 (pino A0)
      digitalWrite(LED_4, LOW); //Desliga LED_4 (pino A4)
      break; //Finaliza o bloco P com a instrução break (sai do switch)
  } //Fim do laço Switch case
  caractere = 0; //Faz caractere = 0, pois, se não, caractere continua com o dado anterior
} //Fim do laço de repetição
  
```

58 Arduino/Genuino Uno em COM3

## Montagem

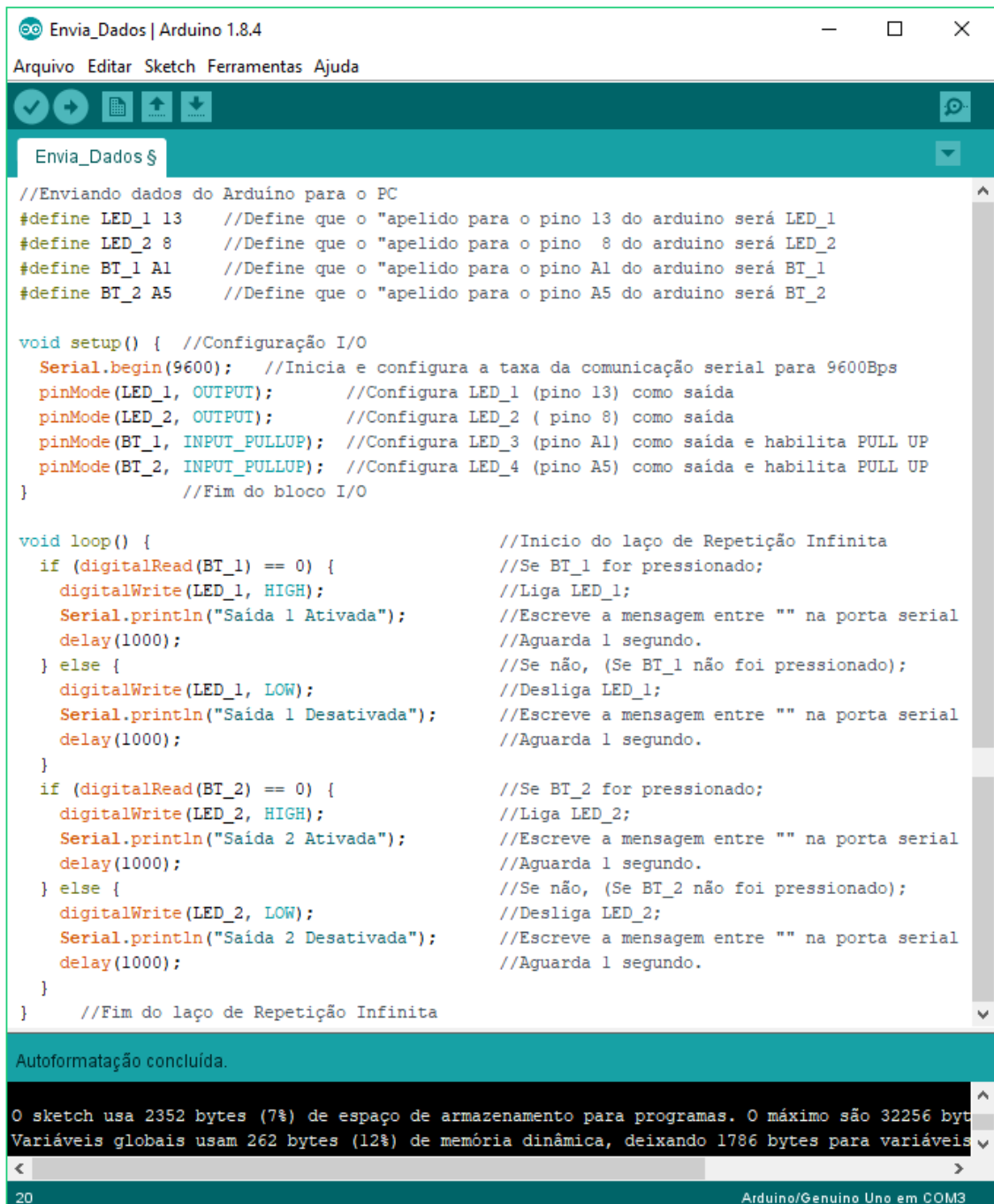


Para enviar dados do computador para o Arduino, utilize o monitor serial. No final desta unidade, é demonstrado como utilizar o monitor serial.

**Exemplo 5: Enviando dados para o PC**

Utilizando botões com retenção (trava), faça um programa que:

- Se o botão BT\_1 (pino A1) estiver fechado, ligue o LED\_1 (pino 13) e envie a mensagem: Saída 1 Ativada. Se BT\_1 aberto, desligue LED\_1 e envie a mensagem: Saída 1 Desativada.
- Se o botão BT\_2 (pino A5) estiver fechado, ligue o LED\_2 (pino 8) e envie a mensagem: Saída 2 Ativada. Se BT\_2 aberto, mensagem: Saída 2 Desativada.



```

Envia_Dados | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Envia_Dados $
//Enviando dados do Arduino para o PC
#define LED_1 13 //Define que o "apelido para o pino 13 do arduino será LED_1
#define LED_2 8 //Define que o "apelido para o pino 8 do arduino será LED_2
#define BT_1 A1 //Define que o "apelido para o pino A1 do arduino será BT_1
#define BT_2 A5 //Define que o "apelido para o pino A5 do arduino será BT_2

void setup() { //Configuração I/O
  Serial.begin(9600); //Inicia e configura a taxa da comunicação serial para 9600Bps
  pinMode(LED_1, OUTPUT); //Configura LED_1 (pino 13) como saída
  pinMode(LED_2, OUTPUT); //Configura LED_2 ( pino 8) como saída
  pinMode(BT_1, INPUT_PULLUP); //Configura LED_3 (pino A1) como saída e habilita PULL UP
  pinMode(BT_2, INPUT_PULLUP); //Configura LED_4 (pino A5) como saída e habilita PULL UP
} //Fim do bloco I/O

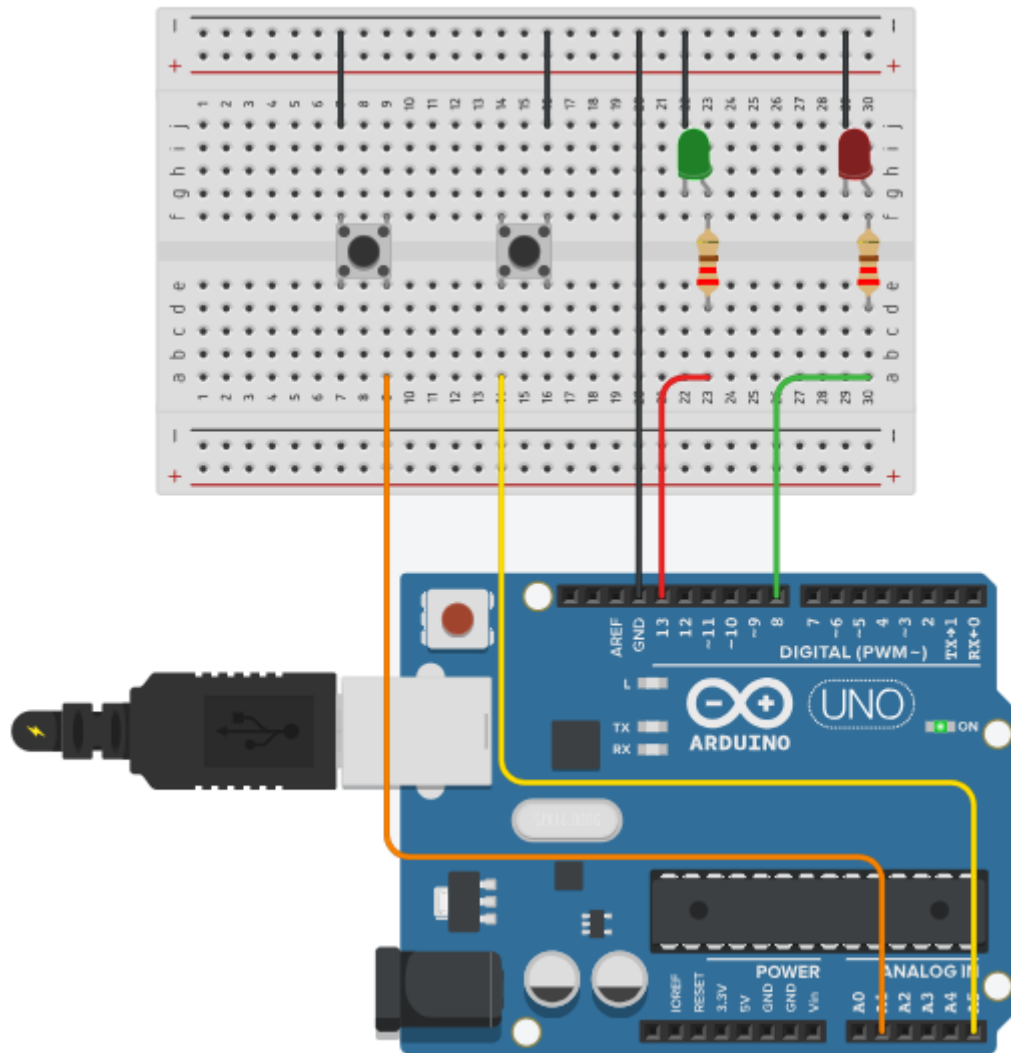
void loop() { //Inicio do laço de Repetição Infinita
  if (digitalRead(BT_1) == 0) { //Se BT_1 for pressionado;
    digitalWrite(LED_1, HIGH); //Liga LED_1;
    Serial.println("Saída 1 Ativada"); //Escreve a mensagem entre "" na porta serial
    delay(1000); //Aguarda 1 segundo.
  } else { //Se não, (Se BT_1 não foi pressionado);
    digitalWrite(LED_1, LOW); //Desliga LED_1;
    Serial.println("Saída 1 Desativada"); //Escreve a mensagem entre "" na porta serial
    delay(1000); //Aguarda 1 segundo.
  }
  if (digitalRead(BT_2) == 0) { //Se BT_2 for pressionado;
    digitalWrite(LED_2, HIGH); //Liga LED_2;
    Serial.println("Saída 2 Ativada"); //Escreve a mensagem entre "" na porta serial
    delay(1000); //Aguarda 1 segundo.
  } else { //Se não, (Se BT_2 não foi pressionado);
    digitalWrite(LED_2, LOW); //Desliga LED_2;
    Serial.println("Saída 2 Desativada"); //Escreve a mensagem entre "" na porta serial
    delay(1000); //Aguarda 1 segundo.
  }
} //Fim do laço de Repetição Infinita

Autoformatação concluída.
O sketch usa 2352 bytes (7%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 262 bytes (12%) de memória dinâmica, deixando 1786 bytes para variáveis locais.
20 Arduino/Genuino Uno em COM3

```

**NOTA.:** Neste exemplo, de 1 em 1 segundo será escrito o estado das saídas.

## Montagem



Para enviar dados do Arduino para o computador, utilize o monitor serial. No final desta unidade, é demonstrado como utilizar o monitor serial.

# Monitor Serial, comunicação entre Arduino e PC

## Introdução

O Arduino consegue se comunicar com o PC através da porta USB. Essa comunicação se dá através de um processo chamado “comunicação serial”, que podemos acessar pelo “Monitor Serial” no IDE do Arduino. Essa comunicação é feita em duas vias, ou seja, enviando e recebendo dados.

## Materiais utilizados neste tutorial

- ✓ 01 Arduino UNO;
- ✓ 01 Cabo USB.

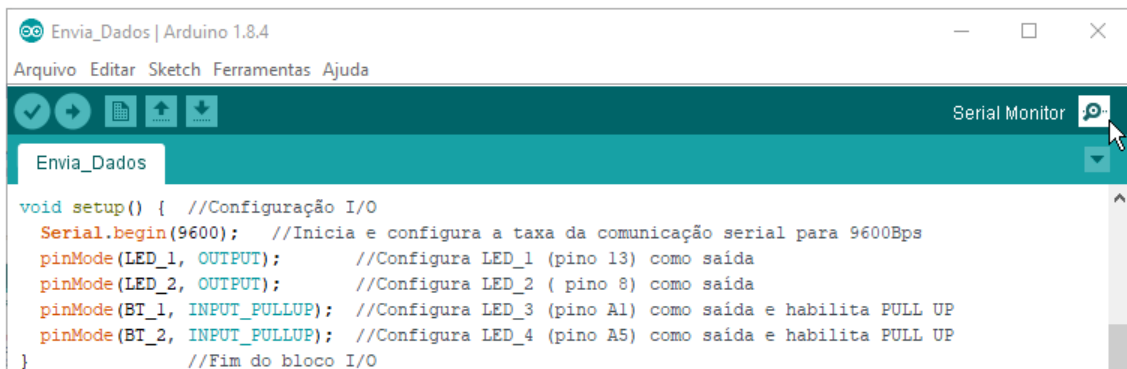
## Montagem

- 1) Conecte o Arduino ao PC por meio de um cabo USB;
- 2) Abra a IDE (software utilizado para digitar nossos programas do Arduino);
- 3) Selecione a placa e a COM. Caso não lembre como fazer, leia a unidade 1.

## Monitor Serial

Após fazer “Upload” do código no Arduino, abra o Monitor Serial.

(Para abri-lo clique em Serial Monitor assim como mostrado na imagem a baixo).



O Monitor Serial tem a seguinte aparência:



Selecione a mesma velocidade de comunicação inserida no programa. Para um melhor entendimento, observe as duas imagens.

Para enviar dados do PC ao Arduino, escreva na janela e clique em Send. Utilizando esta imagem como exemplo, se clicarmos em Send, será enviado a letra A.

Se o Arduino enviar dados, os mesmos serão exibidos no espaço em branco.