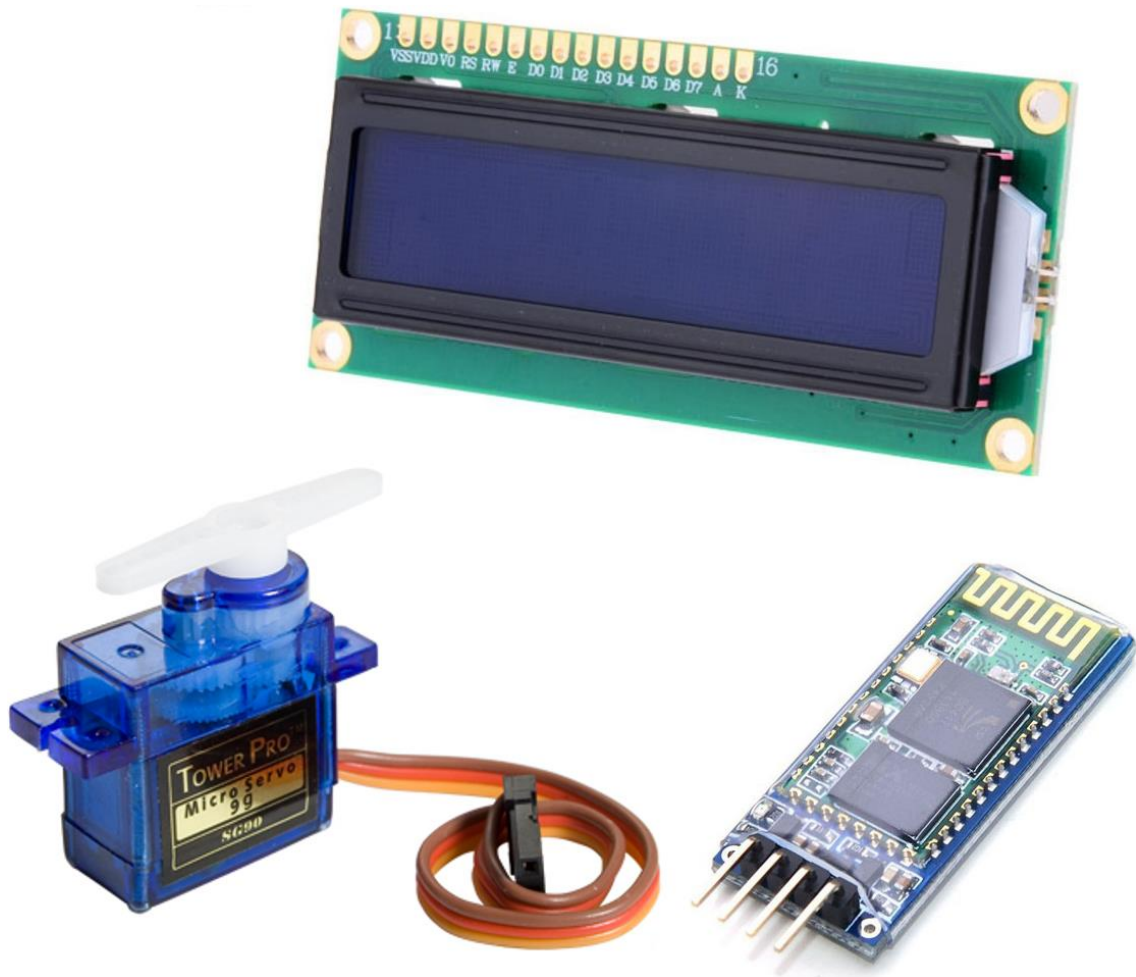


Programação em Linguagem C

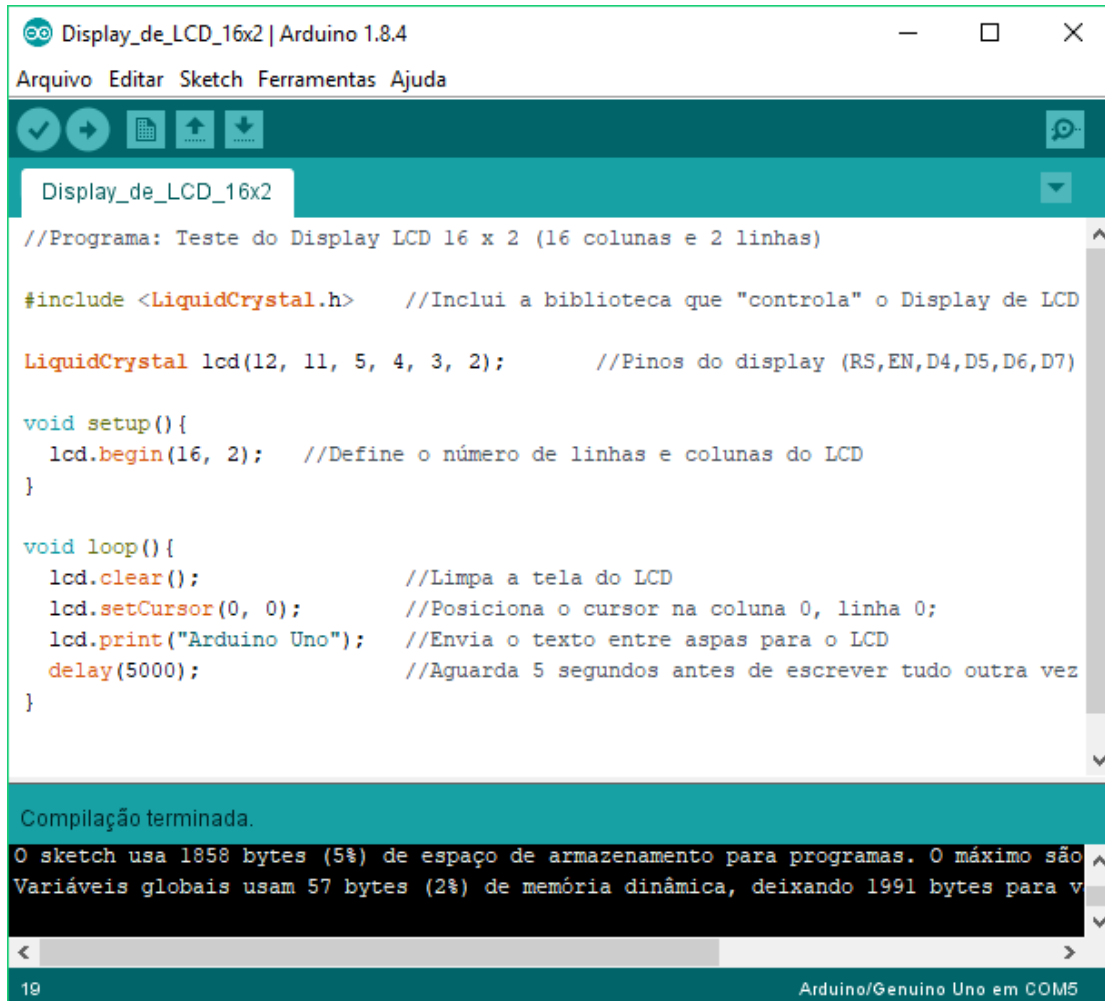
Unidade 3

Nesta unidade estudaremos:

- ✓ Display de LCD 16x2 (16 colunas x 2 linhas);
- ✓ Servo Motor;
- ✓ Comunicação Bluetooth.



Exemplo 1: Neste exemplo vamos apenas escrever Arduino Uno na tela do LCD.



```
Display_de_LCD_16x2 | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Display_de_LCD_16x2
//Programa: Teste do Display LCD 16 x 2 (16 colunas e 2 linhas)

#include <LiquidCrystal.h> //Inclui a biblioteca que "controla" o Display de LCD

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //Pinos do display (RS,EN,D4,D5,D6,D7)

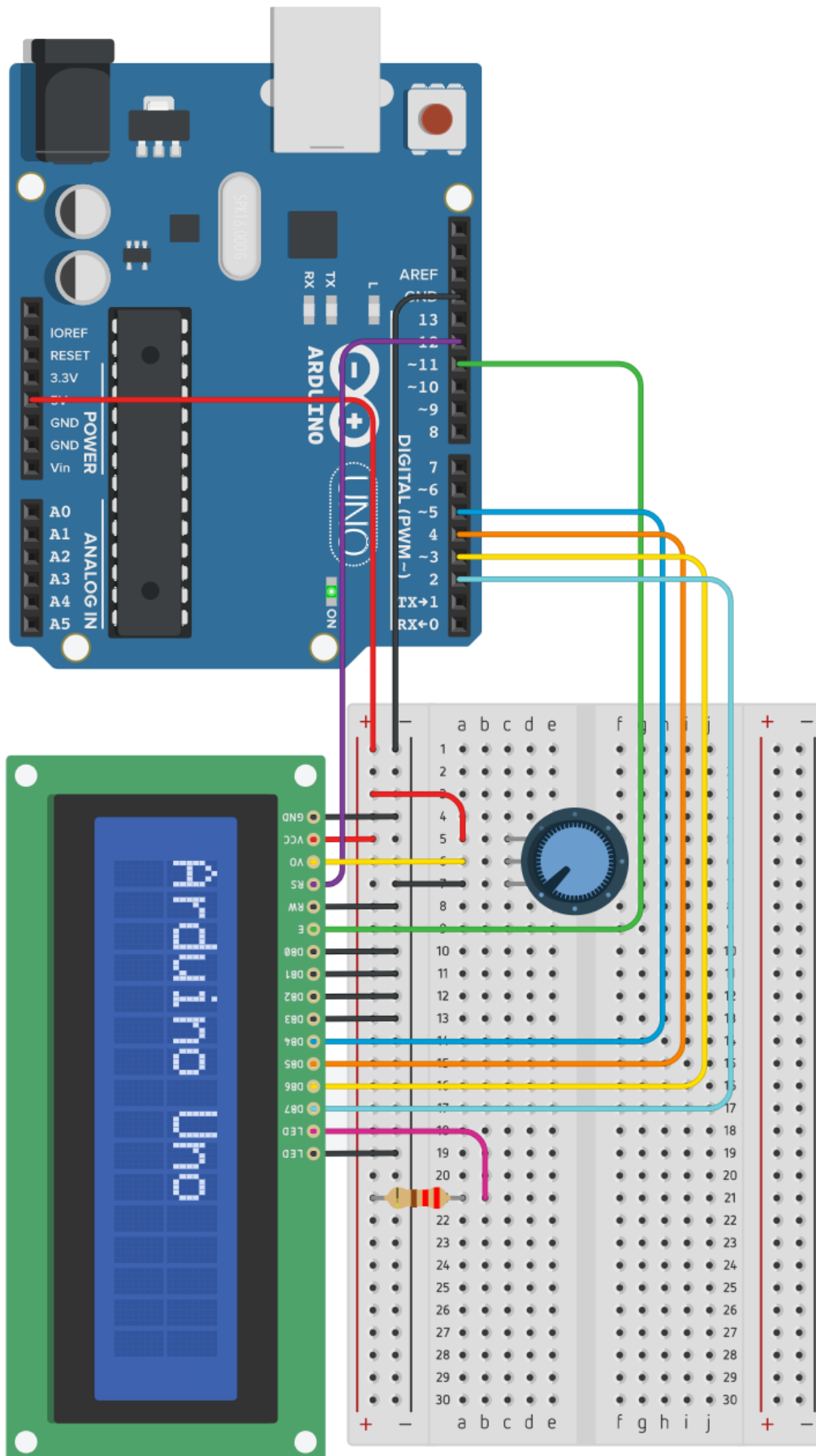
void setup() {
  lcd.begin(16, 2); //Define o número de linhas e colunas do LCD
}

void loop() {
  lcd.clear(); //Limpa a tela do LCD
  lcd.setCursor(0, 0); //Posiciona o cursor na coluna 0, linha 0;
  lcd.print("Arduino Uno"); //Envia o texto entre aspas para o LCD
  delay(5000); //Aguarda 5 segundos antes de escrever tudo outra vez
}


Compilação terminada.
O sketch usa 1858 bytes (5%) de espaço de armazenamento para programas. O máximo são
Variáveis globais usam 57 bytes (2%) de memória dinâmica, deixando 1991 bytes para v

19 Arduino/Genuino Uno em COM5
```

Montagem



Exemplo 2: Neste exemplo iremos escrever Arduino Uno na tela do LCD, porém, vamos centralizar as palavras e utilizar as duas linhas do display.



```
Display_de_LCD_16x2 | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Display_de_LCD_16x2
//Programa: Teste do Display LCD 16 x 2 (16 colunas e 2 linhas)

#include <LiquidCrystal.h> //Inclui a biblioteca que "controla" o Display de LCD

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //Pinos do display (RS,EN,D4,D5,D6,D7)

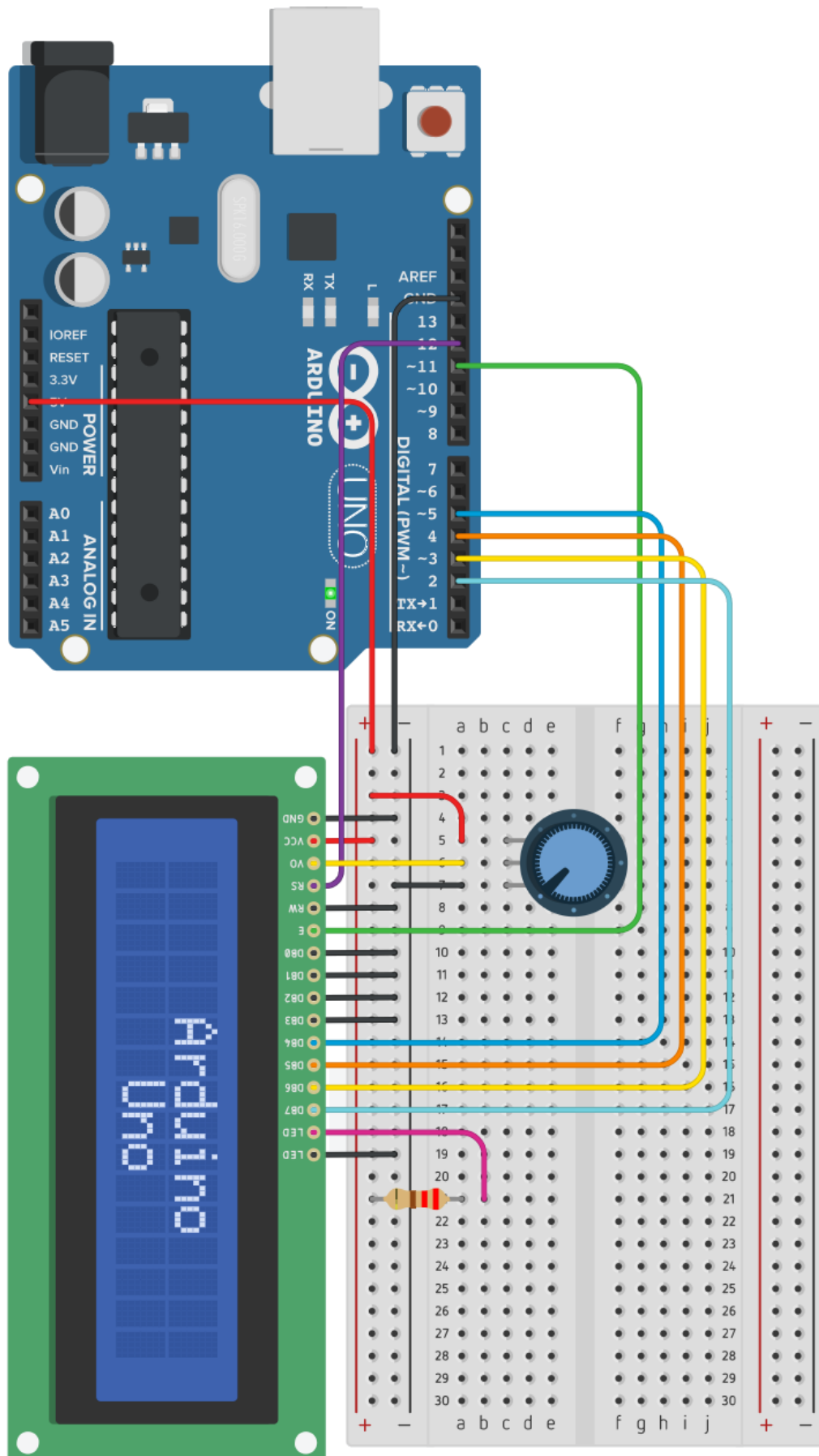
void setup() {
  lcd.begin(16, 2); //Define o número de linhas e colunas do LCD
}

void loop() {
  lcd.clear(); //Limpa a tela do LCD
  lcd.setCursor(5, 0); //Posiciona o cursor na coluna 5, linha 0;
  lcd.print("Arduino"); //Envia o texto entre aspas para o LCD
  lcd.setCursor(7, 1); //Posiciona o cursor na coluna 7, linha 1;
  lcd.print("Uno"); //Envia o texto entre aspas para o LCD
  delay(5000); //Aguarda 5 segundos antes de escrever tudo outra vez
}

Compilação terminada.
O sketch usa 1910 bytes (5%) de espaço de armazenamento para programas. O máximo são
Variáveis globais usam 57 bytes (2%) de memória dinâmica, deixando 1991 bytes para v

21 Arduino/Genuino Uno em COM5
```


Montagem



Exemplo 3: Neste exemplo vamos efetuar a média entre duas variáveis e escrever o resultado no display. Sendo que será escrito do seguinte modo:

Media:

valor da variável C.



```
Display_de_LCD_16x2 | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Display_de_LCD_16x2 §

#include <LiquidCrystal.h> //Inclui a biblioteca que "controla" o Display de LCD
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); //Pinos do display (RS,EN,D4,D5,D6,D7)

int a = 8;
int b = 6;
int c = (a+b)/2;

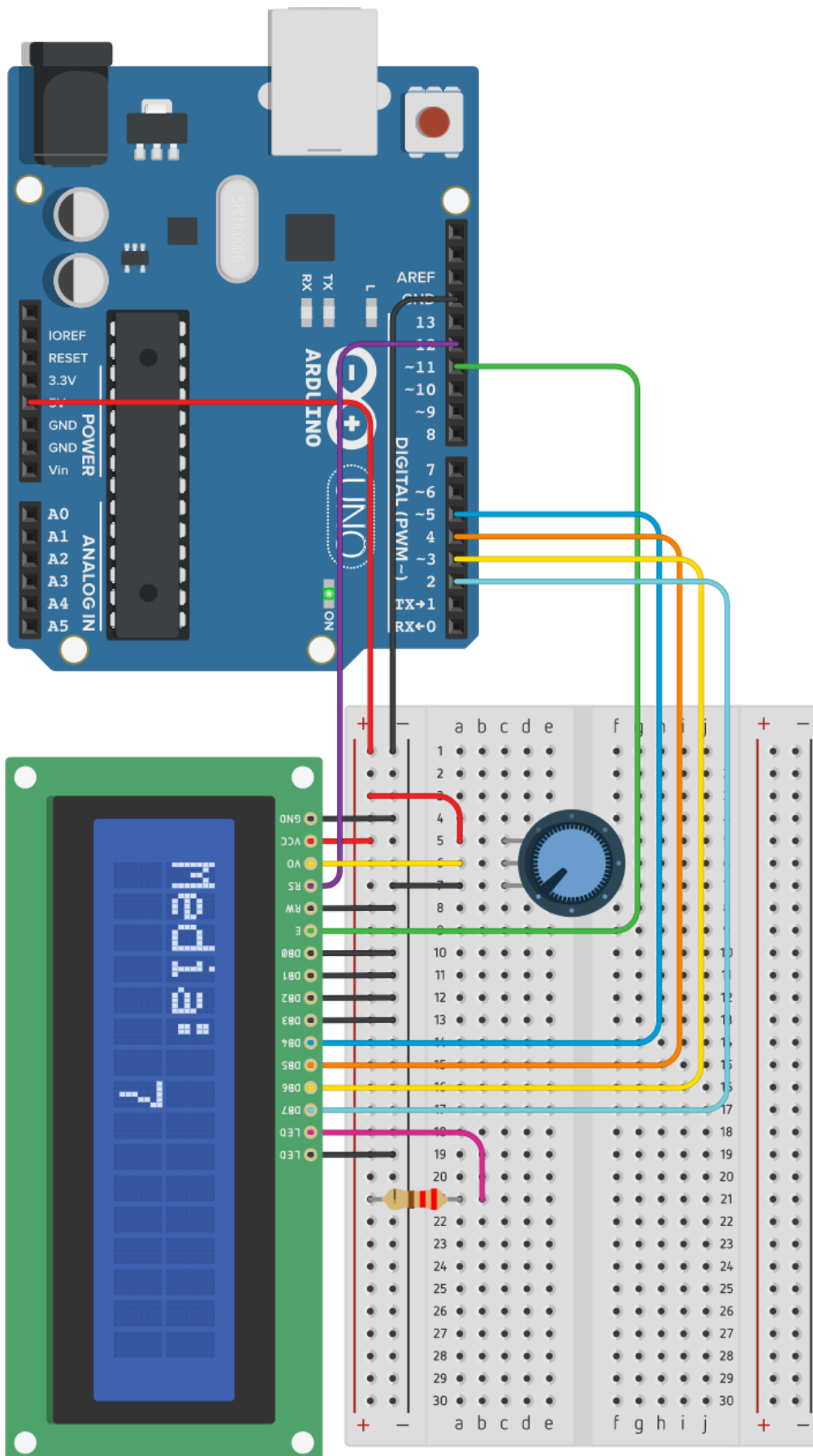
void setup() {
  lcd.begin(16, 2); //Define o número de linhas e colunas do LCD
}

void loop(){
  lcd.clear(); //Limpa a tela do LCD
  lcd.setCursor(0, 0); //Posiciona o cursor na coluna 0, linha 0;
  lcd.print("Media:"); //Escreve o texto entre aspas no LCD
  lcd.setCursor(7, 1); //Posiciona o cursor na coluna 7, linha 1;
  lcd.print(c); //Escreve no LCD o valor contido na variável c;
  delay(5000); //Aguarda 5 segundos antes de escrever tudo outra vez
}

Compilação terminada.
O sketch usa 1880 bytes (5%) de espaço de armazenamento para programas. O máximo são
Variáveis globais usam 67 bytes (3%) de memória dinâmica, deixando 1981 bytes para va

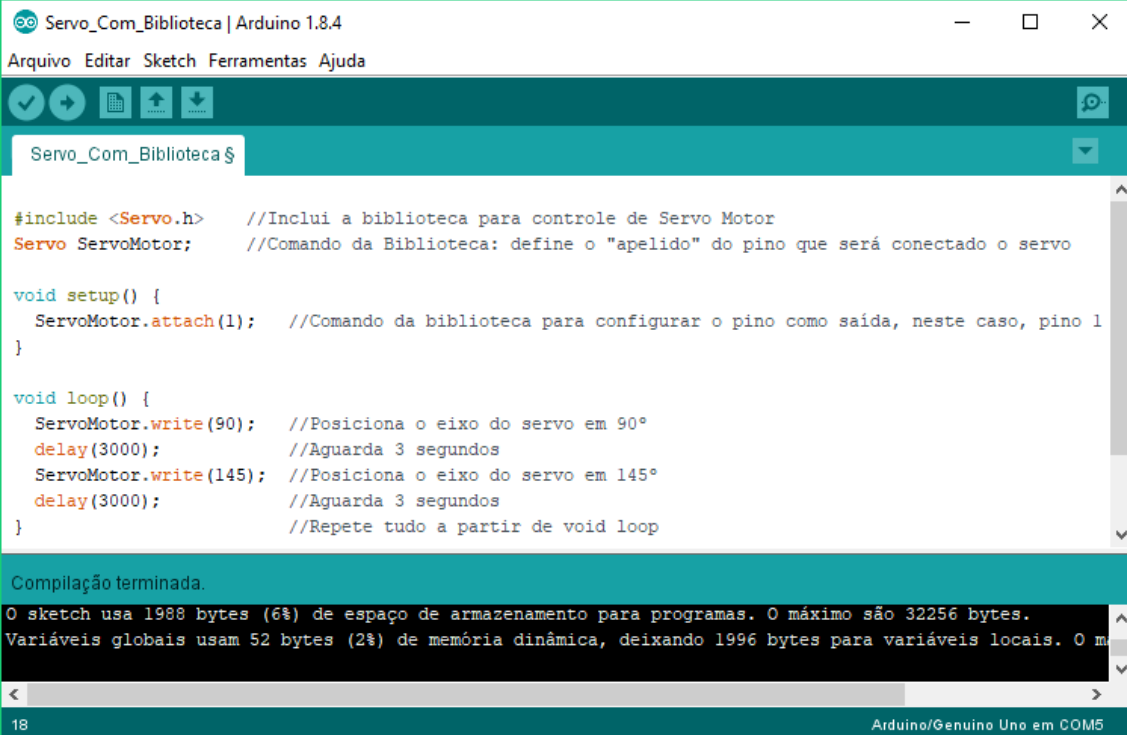
27 Arduino/Genuino Uno em COM5
```

Montagem



Exemplo 4: Controle de angulo de um servo motor.

Neste exemplo o eixo do servo motor será posicionado em 90°, após 3 segundos será posicionado em 145° e depois de mais 3 segundos o ciclo se repete.



```
Servo_Com_Biblioteca | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Servo_Com_Biblioteca $

#include <Servo.h> //Inclui a biblioteca para controle de Servo Motor
Servo ServoMotor; //Comando da Biblioteca: define o "apelido" do pino que será conectado o servo

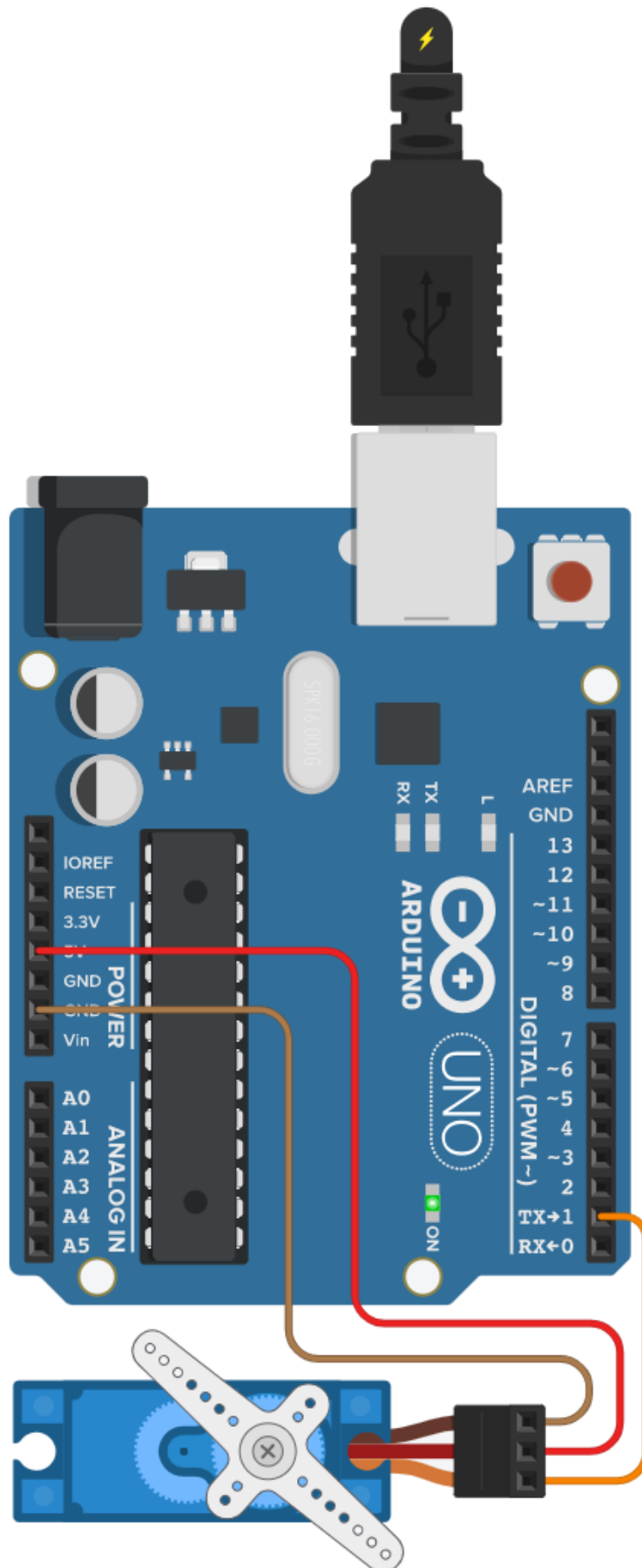
void setup() {
  ServoMotor.attach(1); //Comando da biblioteca para configurar o pino como saída, neste caso, pino 1
}

void loop() {
  ServoMotor.write(90); //Posiciona o eixo do servo em 90°
  delay(3000); //Aguarda 3 segundos
  ServoMotor.write(145); //Posiciona o eixo do servo em 145°
  delay(3000); //Aguarda 3 segundos
} //Repete tudo a partir de void loop

Compilação terminada.
O sketch usa 1988 bytes (6%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 52 bytes (2%) de memória dinâmica, deixando 1996 bytes para variáveis locais. O m

18 Arduino/Genuino Uno em COM5
```

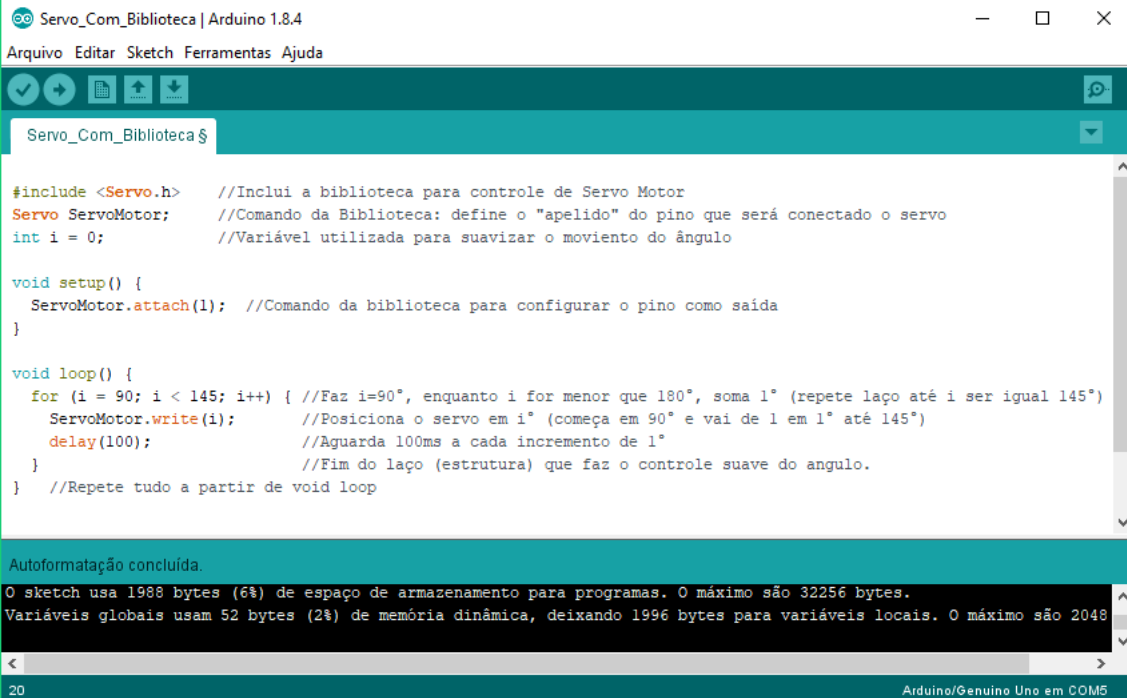

Montagem



Exemplo 5: “Controle suave do ângulo de um servo motor”.

No exemplo anterior, repare que o servo se move bruscamente de 90° para 145°, isto diminui drasticamente sua vida útil e se houve um peso amarrado em seu braço, há uma grande possibilidade deste braço quebrar.

Neste exemplo faremos o servo ir suavemente de 90° para 145°.



```
Servo_Com_Biblioteca | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Servo_Com_Biblioteca $

#include <Servo.h> //Inclui a biblioteca para controle de Servo Motor
Servo ServoMotor; //Comando da Biblioteca: define o "apelido" do pino que será conectado o servo
int i = 0; //Variável utilizada para suavizar o movimento do ângulo

void setup() {
  ServoMotor.attach(1); //Comando da biblioteca para configurar o pino como saída
}

void loop() {
  for (i = 90; i < 145; i++) { //Faz i=90°, enquanto i for menor que 180°, soma 1° (repete laço até i ser igual 145°)
    ServoMotor.write(i); //Posiciona o servo em i° (começa em 90° e vai de 1 em 1° até 145°)
    delay(100); //Aguarda 100ms a cada incremento de 1°
  } //Fim do laço (estrutura) que faz o controle suave do angulo.
} //Repete tudo a partir de void loop

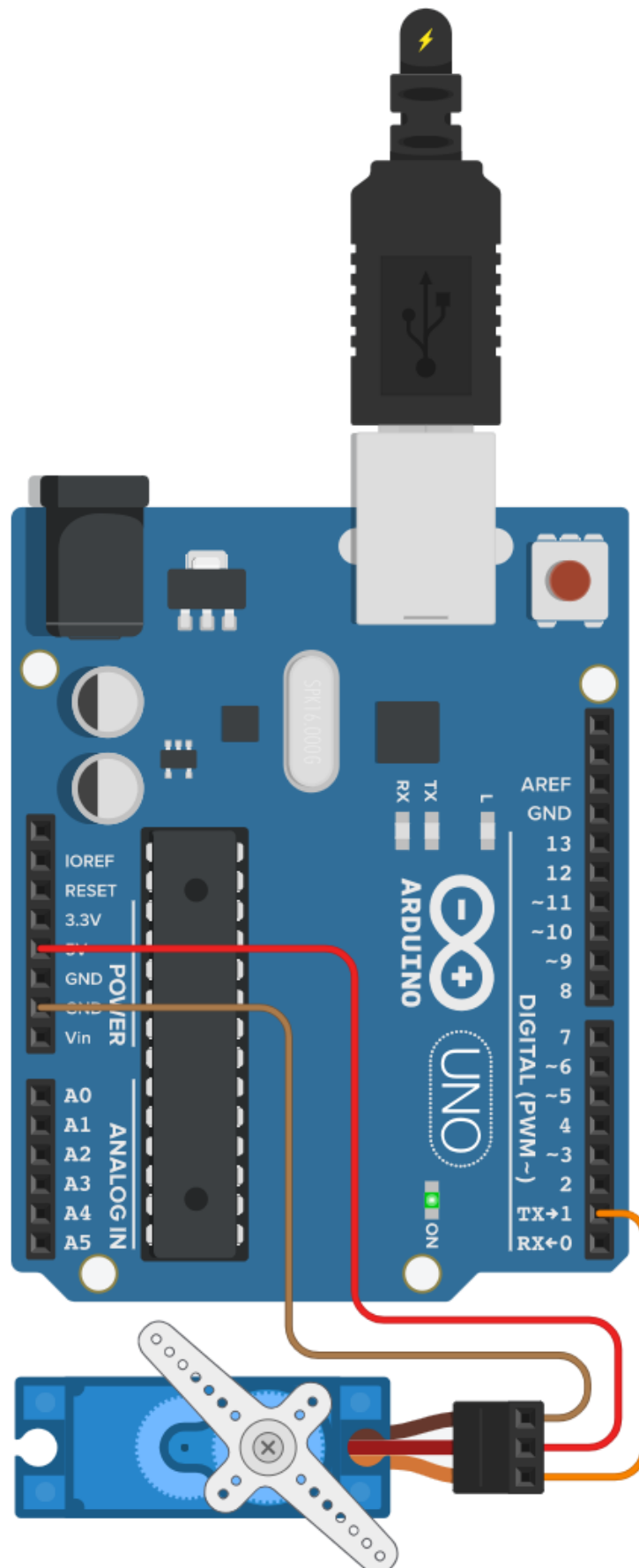
Autoformatação concluída.
O sketch usa 1988 bytes (6%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 52 bytes (2%) de memória dinâmica, deixando 1996 bytes para variáveis locais. O máximo são 2048

20 Arduino/Genuino Uno em COM5
```

Fica como exercício:

Faça o braço do servo retornar suavemente, do mesmo modo que ele avança.

Montagem



Exemplo 6: Comunicação Bluetooth.

Note que este programa é mesmo utilizado em uma comunicação serial!

```

Recebe_Dados | Arduino 1.8.4
Arquivo Editar Sketch Ferramentas Ajuda

Recebe_Dados
//Recepção de dados via Bluetooth utilizando o módulo HC-06
#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
char caractere; //variável utilizada para armazenar o caractere recebido via bluetooth.

void setup() {
  Serial.begin(9600); //Configura a taxa da comunicação serial para 9600Bps
  pinMode(LED_1, OUTPUT); //Configura LED_1 como saída
  pinMode(LED_2, OUTPUT); //Configura LED_2 como saída
}

void loop() {
  if (Serial.available()) { //Verifica se recebeu algum dado na porta serial. Se recebeu:
    caractere = Serial.read(); //Lê o dado recebido e armazena na variável caractere.
  } //Após recebido o dado, ou não recebido dado;

  switch (caractere) { //Variável caractere é tratada pela instrução switch case.
    case 'K': //Caso o dado armazenado em caractere seja a letra K;
      digitalWrite(LED_1, HIGH); break; //Liga LED_1 e sai do caso K quando encontra break.
    case 'W': //Caso o dado armazenado em caractere seja a letra W;
      digitalWrite(LED_1, LOW); break; //Desliga LED_1 e sai do caso W quando encontra break.

    case 'J': //Caso o dado armazenado em caractere seja a letra J;
      digitalWrite(LED_2, HIGH); break; //Liga LED_2 e sai do caso J quando encontra break.
    case 'Y': //Caso o dado armazenado em caractere seja a letra Y;
      digitalWrite(LED_2, LOW); break; //Desliga LED_2 e sai do caso Y quando encontra break.
  }
  caractere = 0; //Faz caractere = 0, se isso não for feito, caractere continua com o dado anterior
}

Compilação terminada.

O sketch usa 1810 bytes (5%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 185 bytes (9%) de memória dinâmica, deixando 1863 bytes para variáveis locais. O
32 Arduino/Genuino Uno em COM5

```

ALERTA:

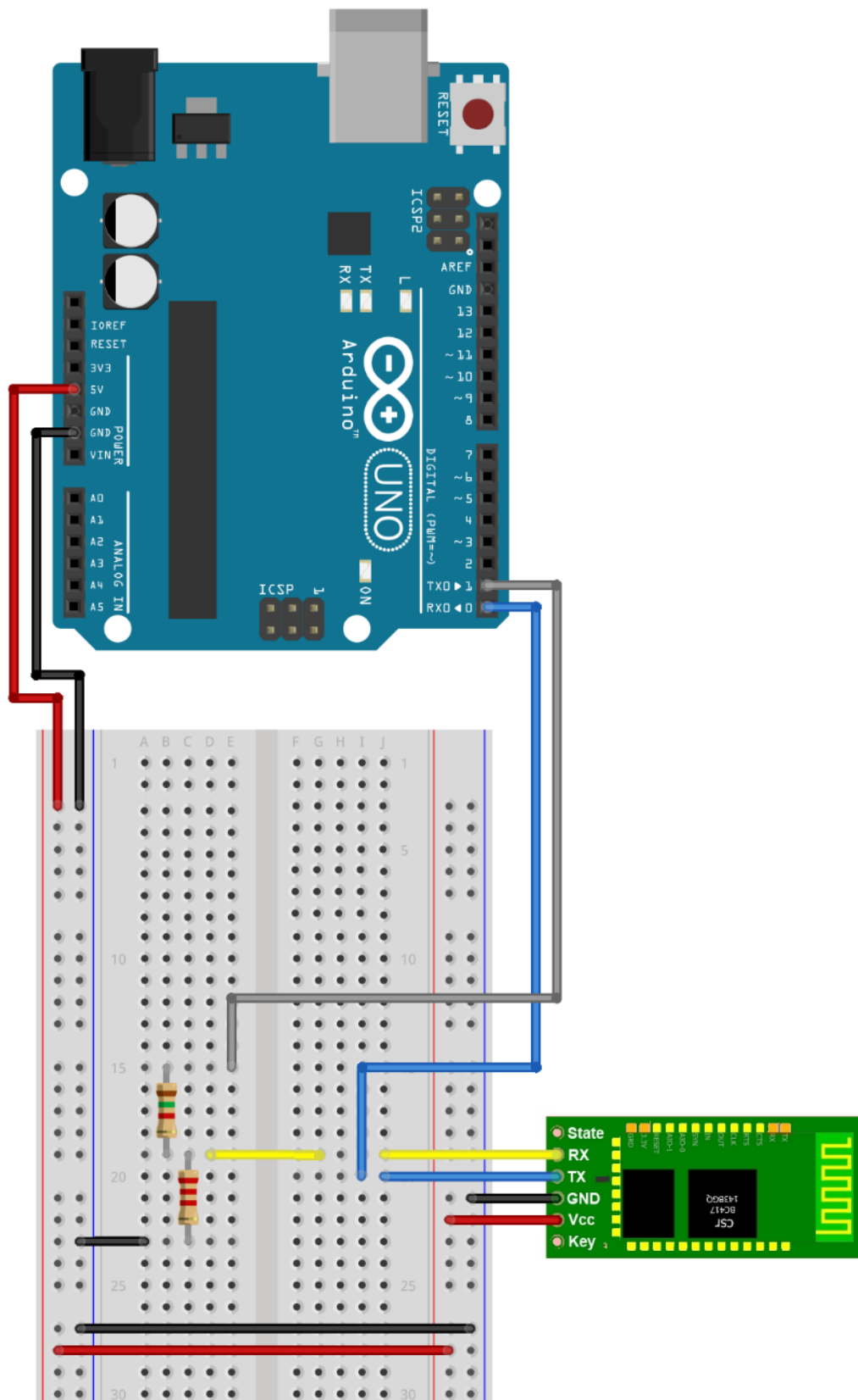
Para este método de comunicação, os pinos de comunicação TX e RX do ARDUINO devem estar desconectados do circuito responsável pela comunicação bluetooth durante o processo de comunicação do Arduino com o computador. Somente conecte-os ao circuito bluetooth se o Arduino estiver desconectado do computador.

Motivo:

O Arduino utiliza (internamente) os pinos TX/RX para fazer a comunicação com o PC. Se os mesmos estiverem conectados em outro circuito durante o processo de gravação, pode haver conflito entre as comunicações e até mesmo a queima de dispositivos!

Para este método, utilize uma fonte externa para alimentar o Arduino.

Montagem



Como este método necessita de uma fonte de alimentação externa (que não seja USB), estudaremos outro método de comunicação bluetooth na próxima unidade.