4-DIGIT 7-SEGMENT

Whether you're building a digital clock, stopwatch, or simple counter, a 4-digit 7-segment display is often the perfect way to show numeric data in a clear and compact format. But if you've ever tried wiring up a bare 7-segment display to an Arduino, you know it can quickly eat up a dozen pins and turn your breadboard into a spaghetti mess of jumper wires. Thankfully, there's a smarter way to do it.

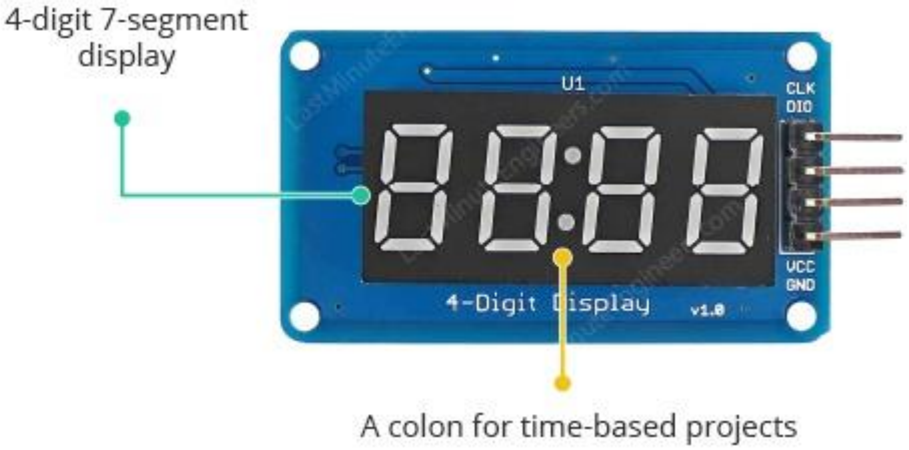
Enter the TM1637 module — a tiny yet powerful driver chip that makes it easy to control a 4-digit 7-segment display using just two Arduino pins. That's right: with only two data lines, you can control all four digits, along with colons and decimal points.

In this tutorial, we'll walk you through how to hook up the TM1637 to your Arduino, load the right library, and start displaying numbers in no time.

Let's dive in!

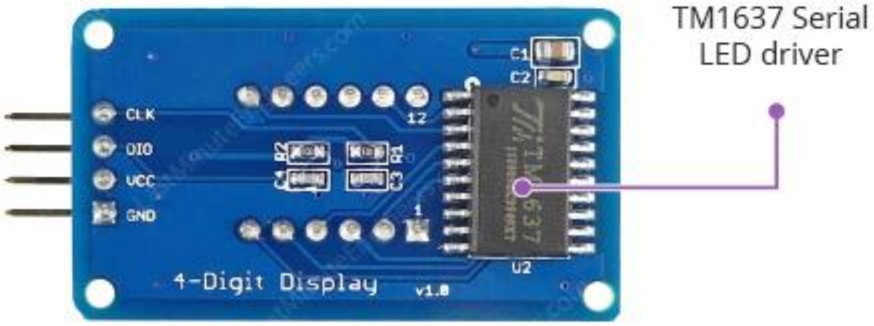
TM1637 MODULE HARDWARE OVERVIEW

The TM1637 module combines a 0.36-inch 4-digit 7-segment display with the TM1637 LED driver chip made by Titan MicroElectronics. This cool combination lets you control all four digits using just two pins from your microcontroller.



This module is perfect for projects where you need to display numbers, like timers, counters, sensor readings, or temperatures. It even has a colon in the middle, which makes it great for clocks and time-related projects.

One of the best things about the TM1637 is that it handles refreshing the display on its own after you update it. This means your microcontroller is free to work on other important tasks instead of constantly refreshing the display.



The amount of power the module uses depends on how many segments are lit and the brightness level you set. When all segments are lit at maximum brightness, the module uses about 80 mA of current. You can save power by adjusting the display's brightness to a lower level.

The TM1637 module works with a supply voltage between 3.3V and 5V. This makes it compatible with both 5V microcontrollers like the regular Arduino and 3.3V microcontrollers like the ESP32.

The module uses its own special two-wire communication protocol. While this isn't a standard protocol like I2C, there are easy-to-use Arduino libraries available that hide all the complicated stuff and make it simple to communicate with the display.

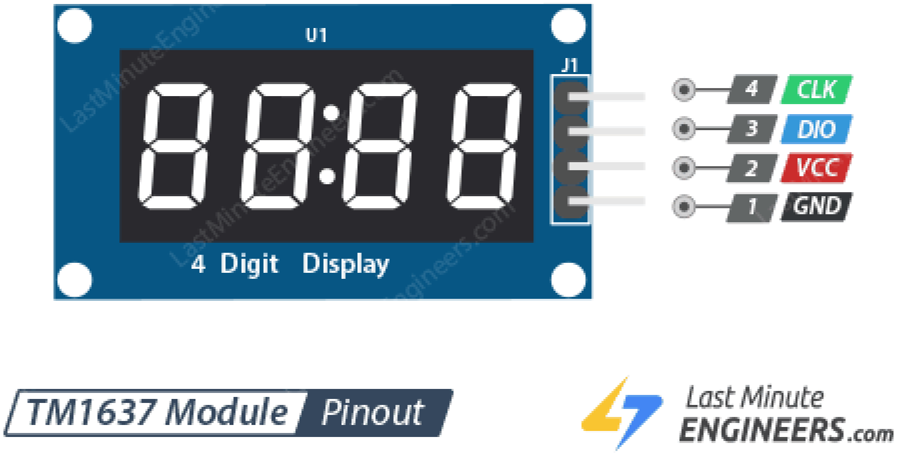
TECHNICAL SPECIFICATIONS

Here are the specifications:

Operating voltage	3.3 – 5 VDC
Maximum Current Draw	80mA
Reverse polarity protection	Yes
Display Size	42 mm x 24 mm (1.65" x .95")
Display Height	12 mm (0.47") (typical)
Character Height	9.14 mm (0.36")

TM1637 MODULE PINOUT

The TM1637 module has only four pins, making it very simple to use. Here is the pinout:



CLK is the clock input pin.

DIO is the data input/output (I/O) pin.

VCC is the power supply pin. Connect it to a 3.3V to 5V power source.

GND is the ground pin.

WIRING THE TM1637 MODULE WITH AN ARDUINO

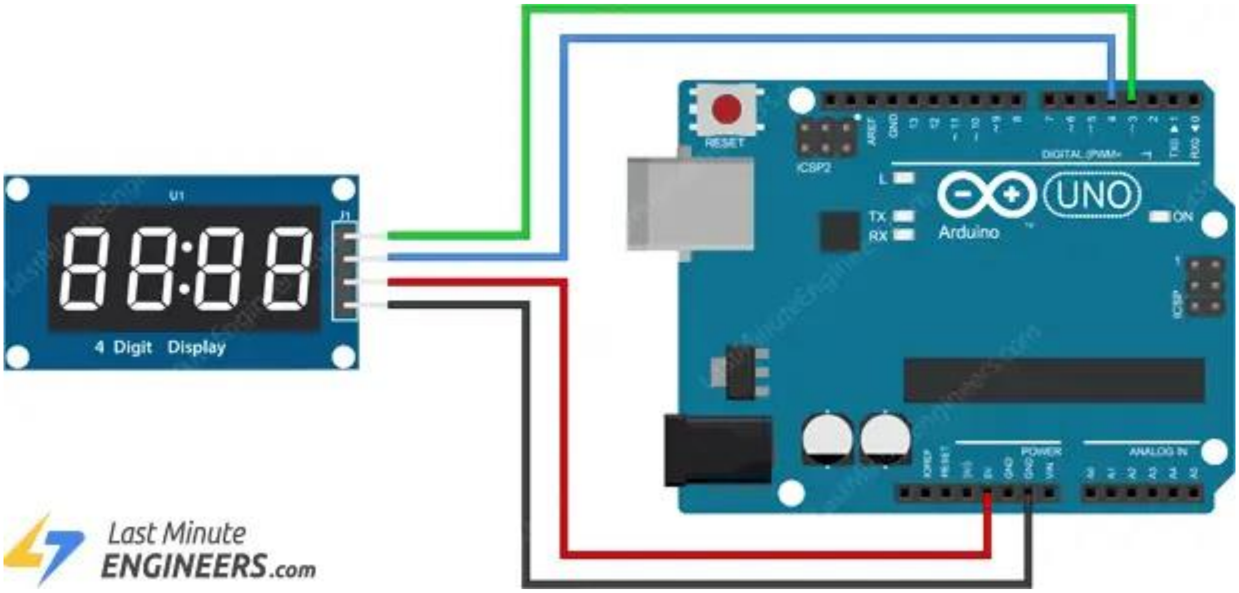
Connecting the TM1637 to an Arduino is very simple. Since the module only has four pins, you just need four wires: two for power and two for communication.

Start by connecting the VCC pin of the module to the 5V output on the Arduino, and the GND pin to one of the Arduino's ground (GND) pins. Then, connect the CLK pin to digital pin 3 on the Arduino, and the DIO pin to digital pin 4.

Here's a quick reference table for the pin connections:

TM1637 Module	Arduino
VCC	5V
GND	GND
DIO	4
CLK	3

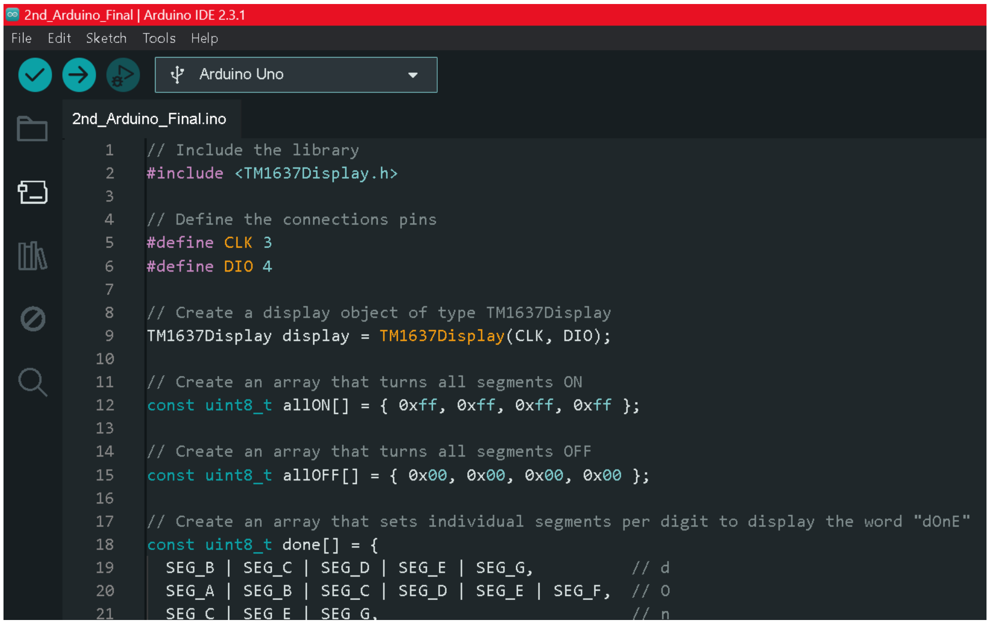
This diagram shows you exactly how to connect everything:



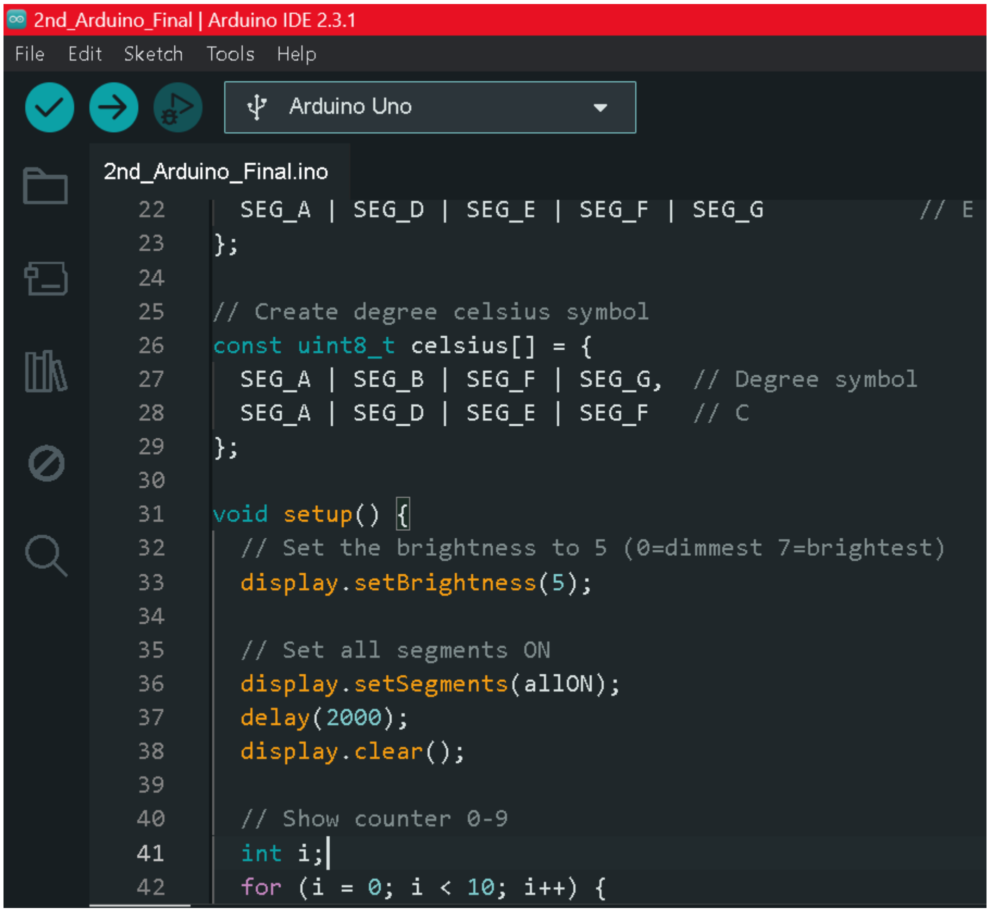
The great thing about the TM1637 module is that it doesn't need specific pins, so you can use different digital pins if you want. Just remember that if you use different pins, you'll need to update the pin numbers in your Arduino code to match your wiring.

CODE

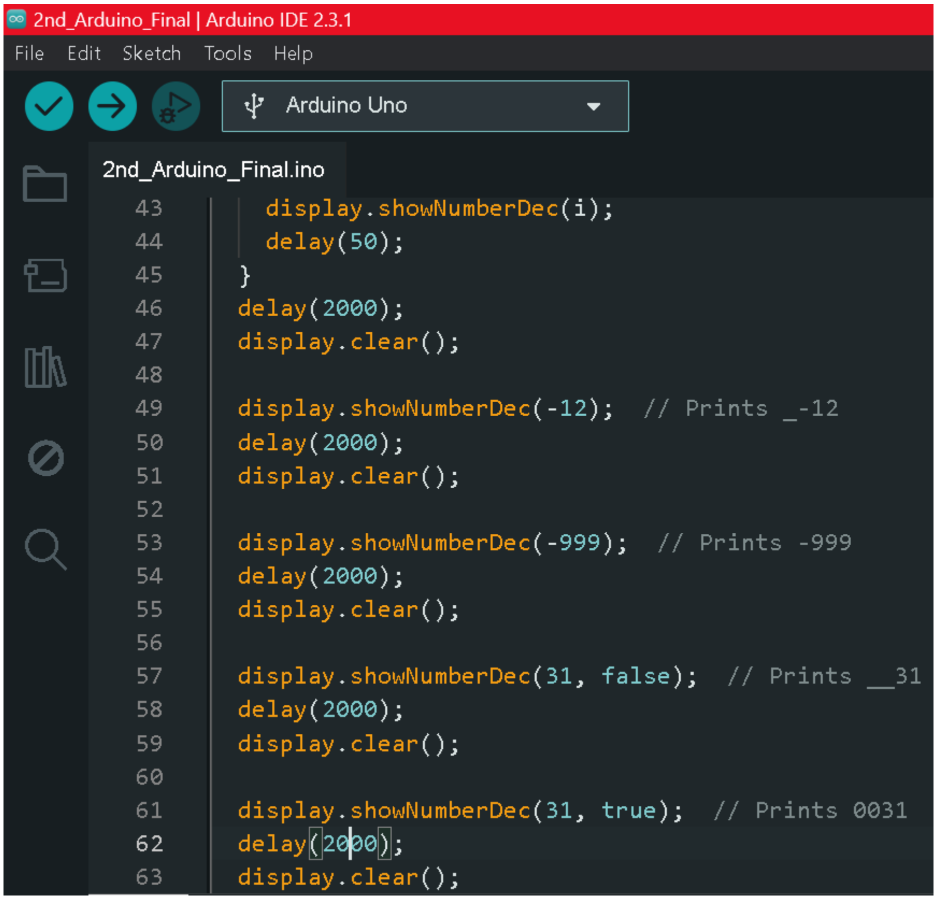
The following is a basic test program that goes through a bunch of different routines. Give it a try; we'll go over how the code works in a moment.



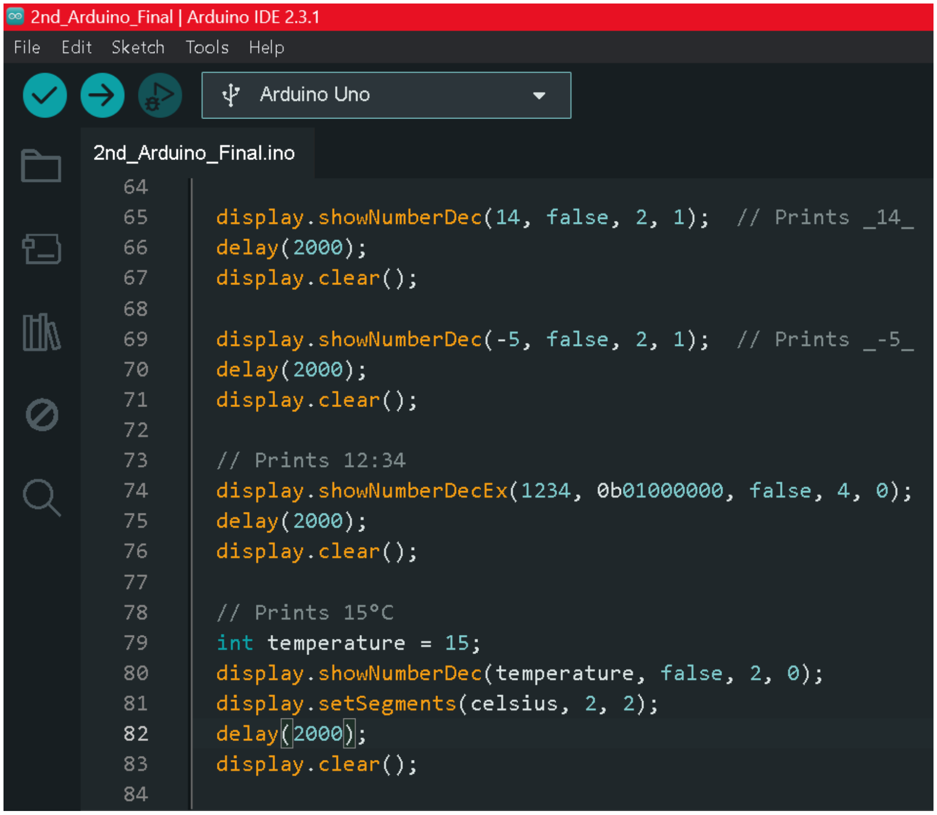
```
// Include the library
#include <TM1637Display.h>

// Define the connections pins
#define CLK 3
#define DIO 4

// Create a display object of type TM1637Display
TM1637Display display = TM1637Display(CLK, DIO);

// Create an array that turns all segments ON
const uint8_t allON[] = { 0xff, 0xff, 0xff, 0xff };

// Create an array that turns all segments OFF
const uint8_t allOFF[] = { 0x00, 0x00, 0x00, 0x00 };

// Create an array that sets individual segments per digit to display the word "dOnE"
const uint8_t done[] = {
  SEG_B | SEG_C | SEG_D | SEG_E | SEG_G,          // d
  SEG_A | SEG_B | SEG_C | SEG_D | SEG_E | SEG_F,  // O
  SEG_C | SEG_E | SEG_G,                          // n
```



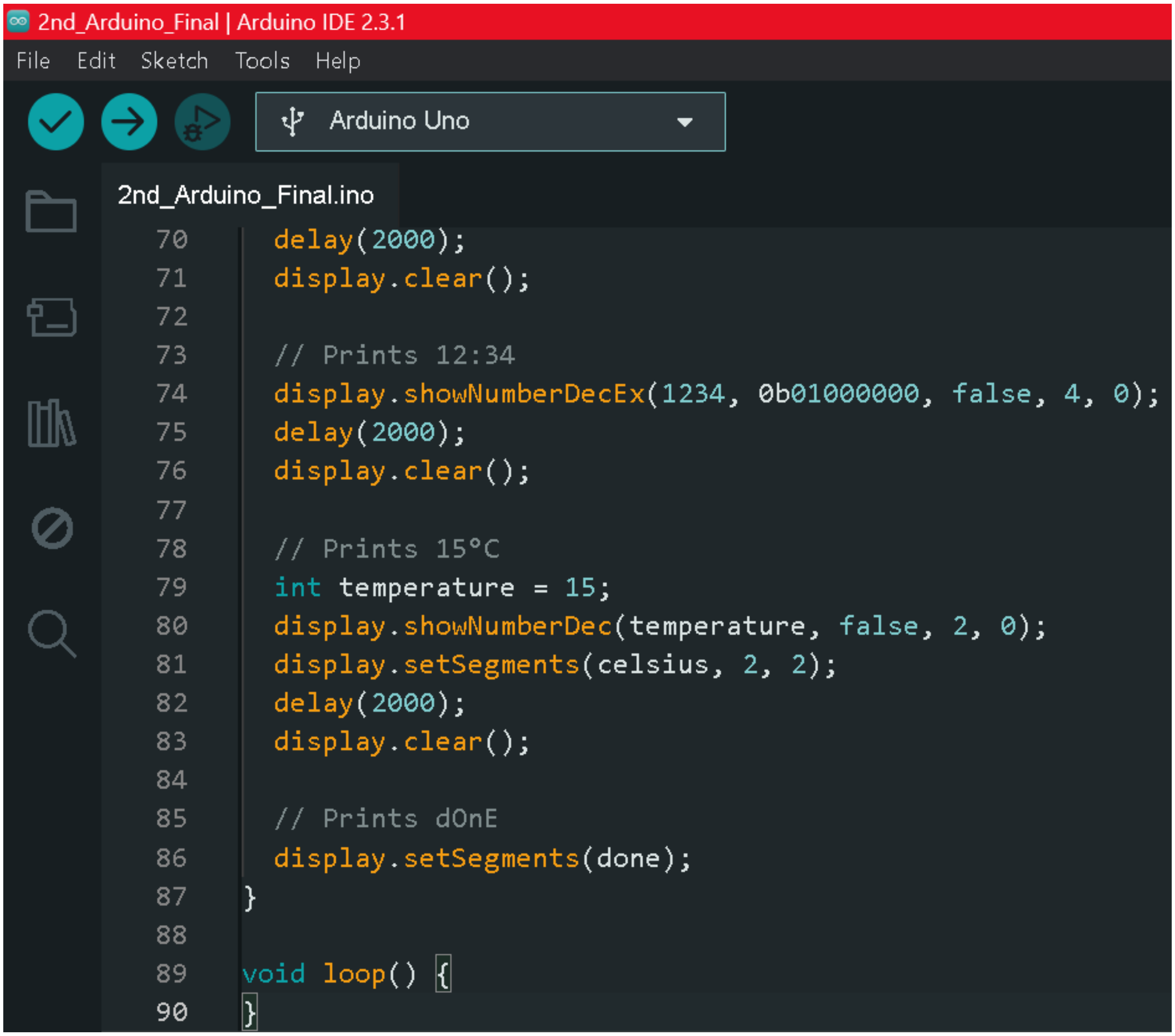
```
  SEG_A | SEG_D | SEG_E | SEG_F | SEG_G           // E
};

// Create degree celsius symbol
const uint8_t celsius[] = {
  SEG_A | SEG_B | SEG_F | SEG_G,  // Degree symbol
  SEG_A | SEG_D | SEG_E | SEG_F   // C
};

void setup() {
  // Set the brightness to 5 (0=dimmest 7=brightest)
  display.setBrightness(5);

  // Set all segments ON
  display.setSegments(allON);
  delay(2000);
  display.clear();

  // Show counter 0-9
  int i;
  for (i = 0; i < 10; i++) {
```



```
    display.showNumberDec(i);
    delay(50);
  }
  delay(2000);
  display.clear();

  display.showNumberDec(-12);  // Prints _-12
  delay(2000);
  display.clear();

  display.showNumberDec(-999);  // Prints -999
  delay(2000);
  display.clear();

  display.showNumberDec(31, false);  // Prints __31
  delay(2000);
  display.clear();

  display.showNumberDec(31, true);  // Prints 0031
  delay(2000);
  display.clear();
```



```

  display.showNumberDec(14, false, 2, 1);  // Prints _14_
  delay(2000);
  display.clear();

  display.showNumberDec(-5, false, 2, 1);  // Prints _-5_
  delay(2000);
  display.clear();

  // Prints 12:34
  display.showNumberDecEx(1234, 0b01000000, false, 4, 0);
  delay(2000);
  display.clear();

  // Prints 15°C
  int temperature = 15;
  display.showNumberDec(temperature, false, 2, 0);
  display.setSegments(celsius, 2, 2);
  delay(2000);
  display.clear();

```


2nd_Arduino_Final | Arduino IDE 2.3.1
File Edit Sketch Tools Help
Arduino Uno
2nd_Arduino_Final.ino

```
  delay(2000);
  display.clear();

  // Prints 12:34
  display.showNumberDecEx(1234, 0b01000000, false, 4, 0);
  delay(2000);
  display.clear();

  // Prints 15°C
  int temperature = 15;
  display.showNumberDec(temperature, false, 2, 0);
  display.setSegments(celsius, 2, 2);
  delay(2000);
  display.clear();

  // Prints dOnE
  display.setSegments(done);
}

void loop() {
}
```

REFERENCE:

https://lastminuteengineers.com/tm1637-arduino-tutorial/