Arduino Tutorial Common Syntax Serial Monitor Output

Serial.begin (9600); : Connects the program to the serial monitor

Serial.print: Prints text in the dialogue box. Used for output readings from the Arduino board.

Serial.println(): Drops cursor down to the next line on the serial monitor

Serial.parseint() or Serial.parsefloat(): Allows user to read either integer or real numbers for input

Serial.flush (); : Waits for transmission of outgoing data to be complete

Serial.available (); : Waits until the input buffer has a character **Digital/Analog Commands**

digitalWrite or analogWrite: Write code to the Arduino board (I.E digitalWrite (13, HIGH) = Turns

LED light linked to Pin 13 On) digitalRead or analogRead: Reads information from the Arduino

Board. Typically used with a switch or sensor to determine if contact is being made (I.E

digitalRead (5, Low) = Switch/Sensor

linked to Pin 5 on the Arduino board is not making contact)

Serial.read: Reads and Output information from a specified pin on the Arduino Board or keyboard (NOTE Keyboard input will always read as character variable (even numbers) **Power Transfer/Wait Time**

HIGH: Power transferred or switch will read in contact (On)

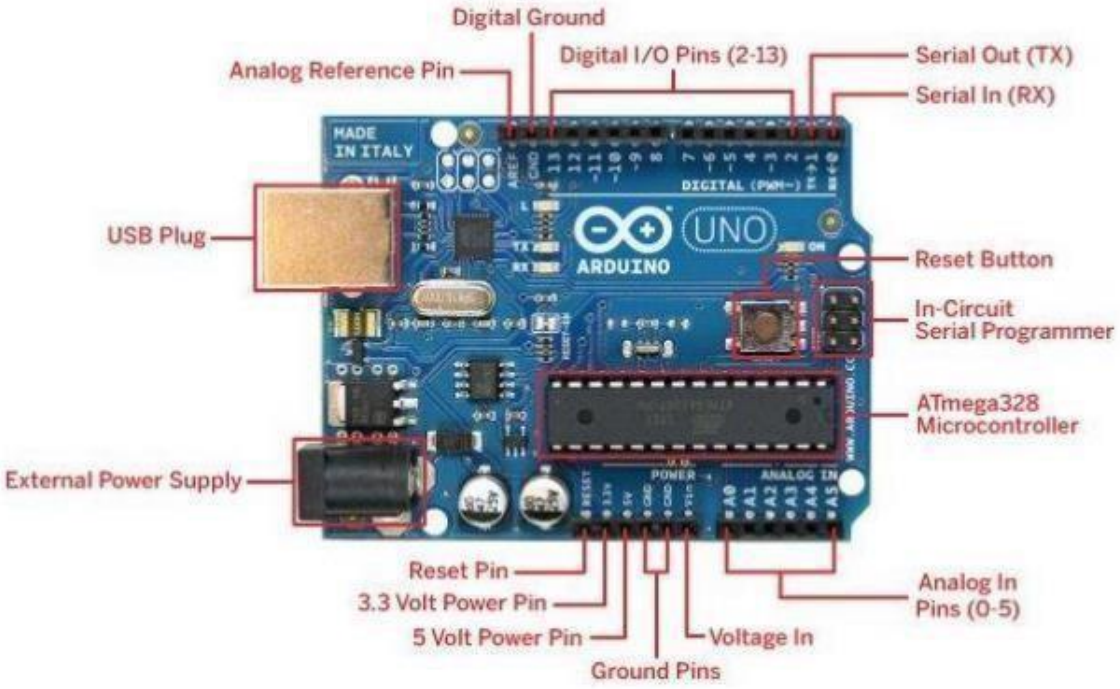
LOW: Power is not transferred or switch will read not in contact (Off)

delay(): Will pause the program for x number of milliseconds (NOTE: 1000 milliseconds = 1 second) **Pin Declaration**

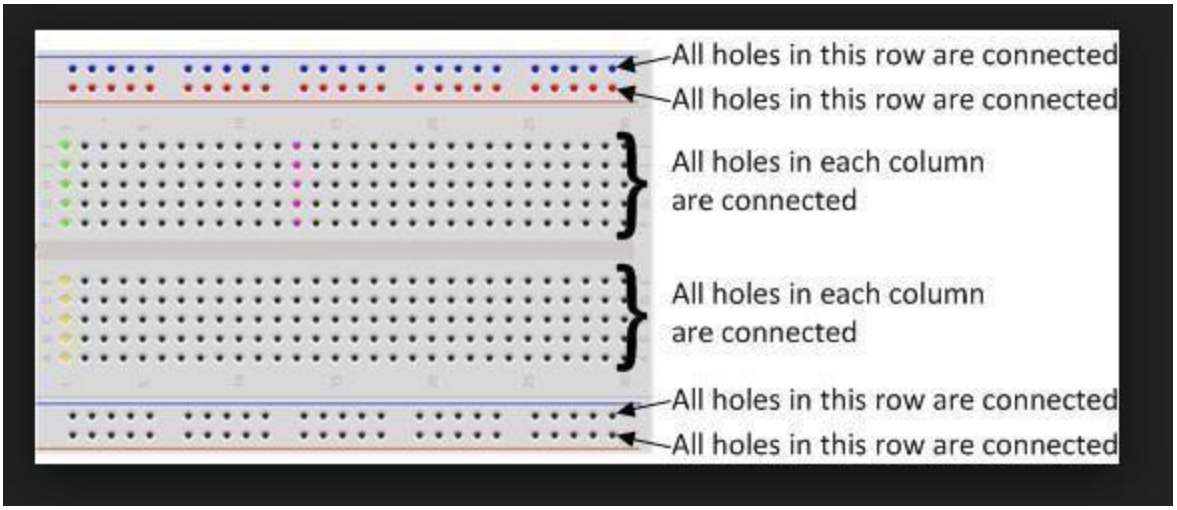
pinMode (): Declares the numbered pin to be either input or output (I.E pinMode (13, INPUT); or pinMode (13, OUTPUT)

NOTE: All Loops (for, while, do/while), if/then, if/else syntax is the same as learned in previous lessons

Basic Anatomy of the Arduino Board



Breadboard Setup

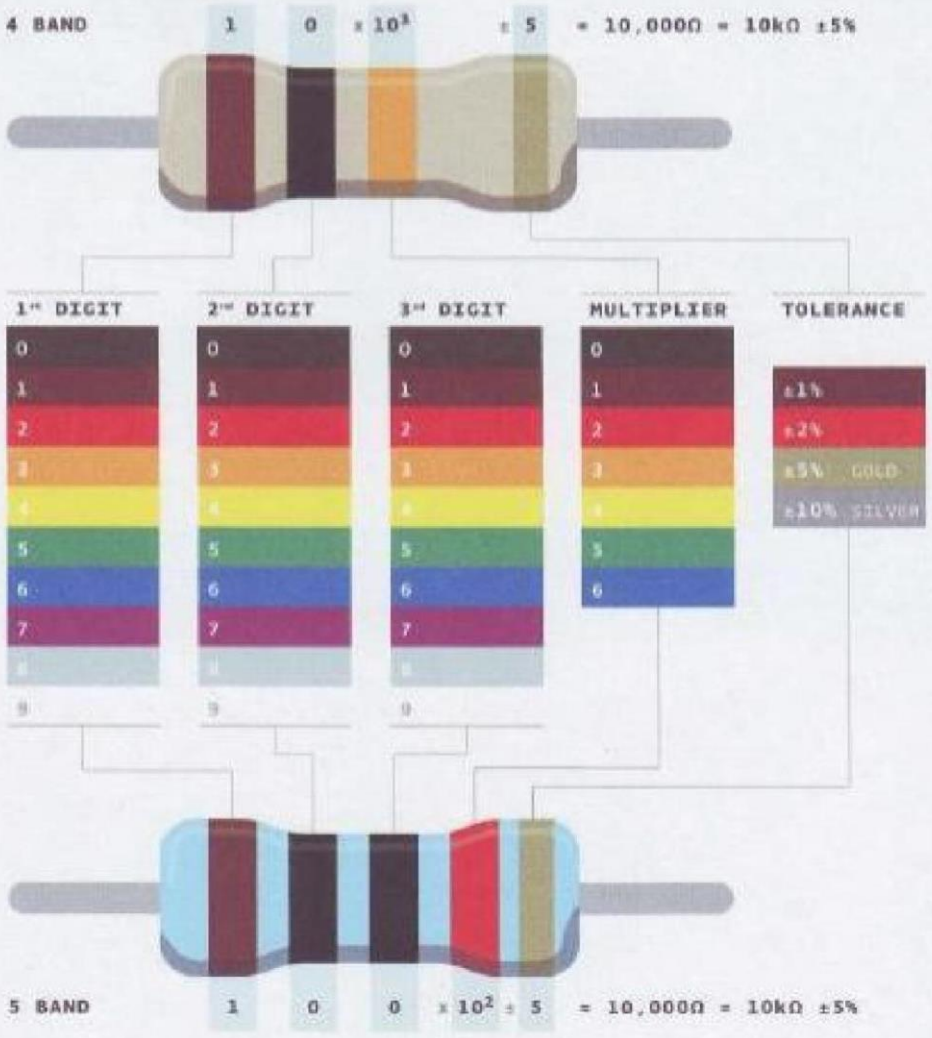


Reading a Resistor

HOW TO READ RESISTOR COLOR CODES

Resistor values are marked using colored bands, according to a code developed in the 1920s, when it was too difficult to write numbers on such tiny objects.

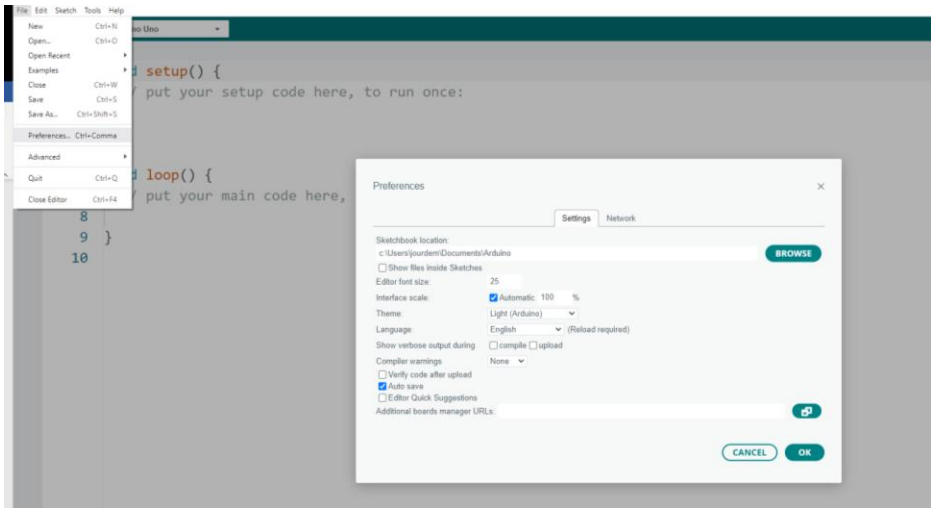
Each color corresponds to a number, like you see in the table below. Each resistor has either 4 or 5 bands. In the 4-band type, the first two bands indicate the first two digits of the value while the third one indicates the number of zeroes that follow (technically it reprents the power of ten). The last band specifies the tolerance: in the example below, gold indicates that the resistor value can be 10k ohm plus or minus 5%.



Workspace Setup

NOTE: It is not necessary to declare the #include <iostream> library or int main() in your program. Those codes are prebuilt into the structure of the Arduino program.

NOTE: Change Font Height of Text: Pull Down Menu File > Preferences > Font > Change to desired text height



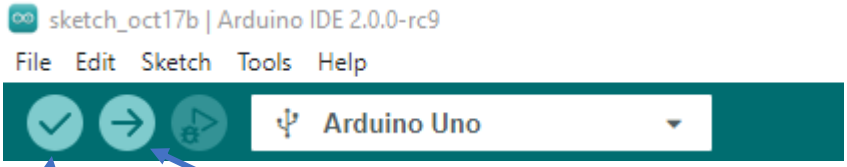
sketch_oct17b.ino

```
1  void setup() {
2    // put your setup code here, to run once:
3
4  }
5
6  void loop() {
7    // put your main code here, to run repeatedly:
8
9  }
10
```

Software provides 2 default functions:

void setup(){}: only runs the code between braces one time

void loop(){}: runs code between the braces infinitely. There is no way to end it. However program can be designed in a way to trap the program in an infinite loop that will not execute any commands within the loop



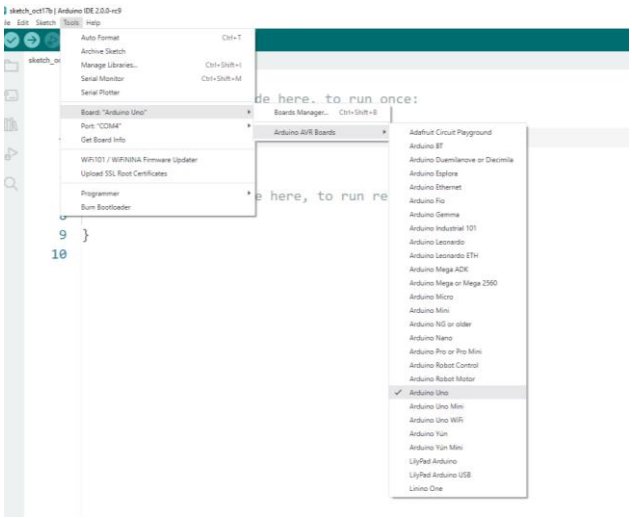
Verify: Checks to see if there are any errors in the program

Upload: Uploads the program to the Arduino

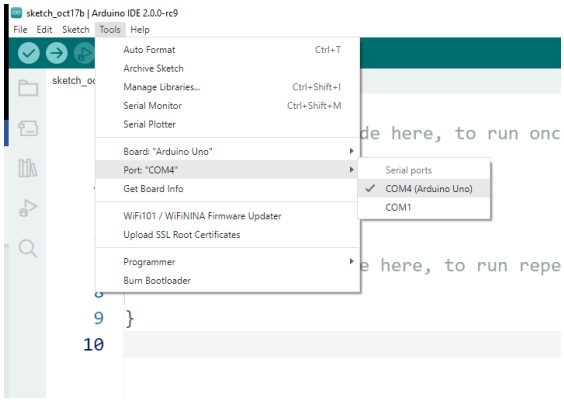
Connecting Arduino Uno to the Computer

NOTE: These steps should been done each time Arduino Software is used.

1. Using the USB-Fire Wire connector, connect the Arduino Board to the computer
2. Open Arduino Software from the desktop
3. Setting up Arduino Board Set the following a Tools > Board > Arduino AVR Boards > Select Arduino Uno



4. Tools > Port > Select the Com# Port that has (Arduino Uno) in parentheses (NOTE: if Arduino Uno does not show up; look at the list of ports > unplug the wire see which port disappears (remember the number) > replug the wire in and choose the port that had disappeared from the list in the previous step)



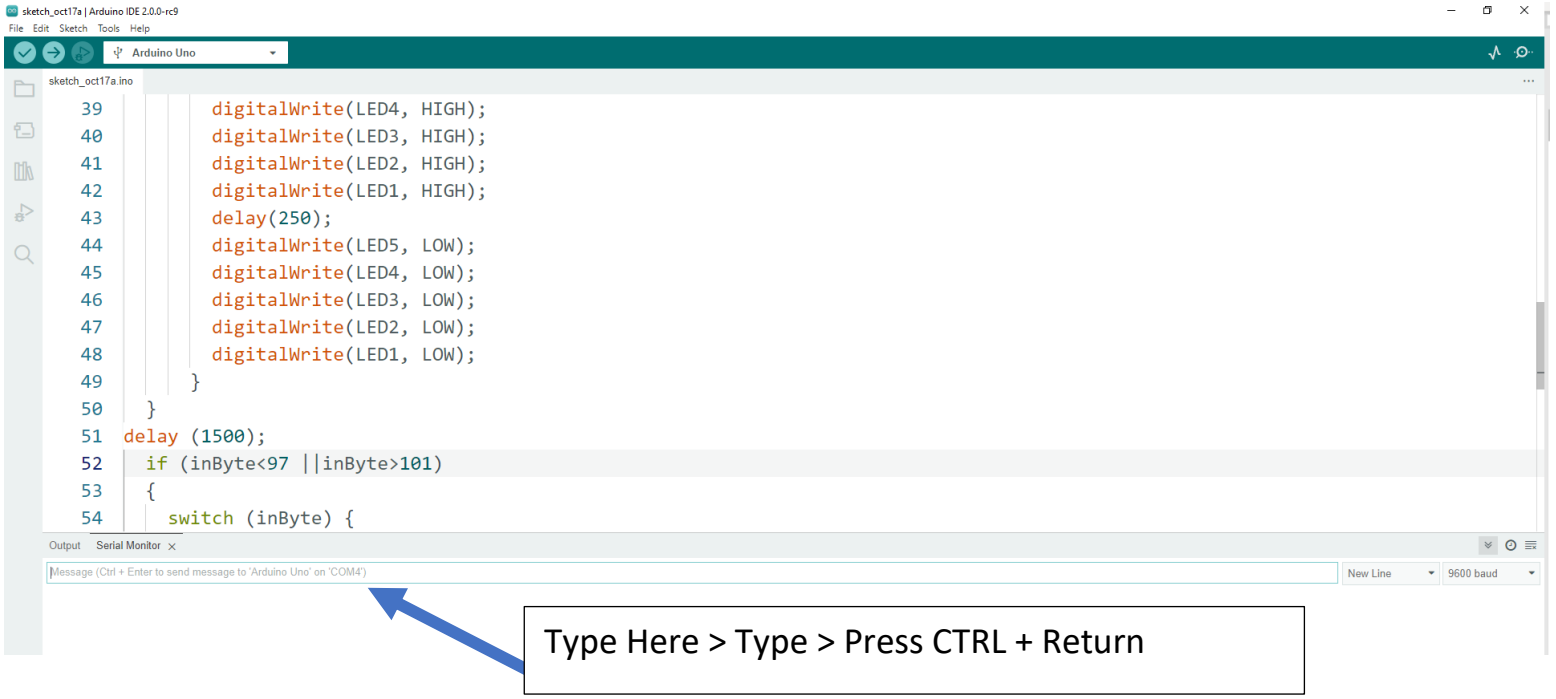
c Tools > Programmer > USBasp

This will set the communication between the PC and the Arduino

5. 1st Time Verifying or Uploading a program the Arduino IDE will ask the user to save the source code. This will make a folder called sketch and the current date with sketch file (executable file) > Change the file name to a project name (i.e Assignment 1 Volume)> Save this in proper location (Default location is in the Program Files > Arduino

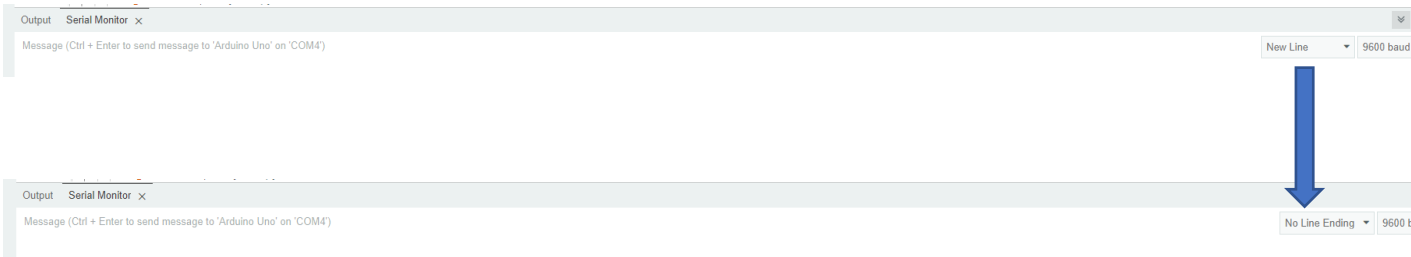
6. Serial Monitor
   a. Select Serial Monitor Icon in the Top Right Corner > Type in spot located above > Press CTRL Return for the software to accept the detail

   b. Serial Monitor will appear at the bottom of the screen



   c. **Serial Monitor Line Types:** The Serial Monitor will assign a value for each line, even if there is no input or when the user is typing in a value to be read by the program.
   New Line: Inputs a Default Value for the Serial Monitor
   No Line Ending: Will Cancel the Default Value for the Serial Monitor
   Carriage Return: Used for Non Alpha-Numeric Characters
   Both NL & CR: Used are defined in ASCII Code and can send Decimal Values

   ****Change Line Mode from New Line to No Line Ending****



Assignment 1: Base Code adding 2 Numbers

Delete the Blink Code; Type the following Code

```
//Program: Adding Two Numbers

float number1, number2, sum;

char junk = ' ';                              // Variable used to erase bad input
void setup()
{
  Serial.begin (9600);                        // Connects user to Serial Monitor for Output on Screen
                                              // 9600 represents the baud rate (characters/sec
  Serial.println ("Let's calculate sum");     // Serial.println: places Text on Serial Monitor
                                              // the ln of code moves cursor down to the next line after output

  Serial.flush();                             // Waits for the transmission of outgoing data to complete
}


void loop()
{
  // Input Process for Number 1
  Serial.println ("Enter Number 1: ");
  while (Serial.available() == 0);        // ; will wait until the input buffer has a character
  {
    number1 = Serial.parseFloat();        // Serial.parseFloat() returns the first valid number from the Serial Buffer
    Serial.print ("Number 1 = ");
    Serial.println (number1);             // Displays users input
    while (Serial.available() > 0)        // Removes non-numeric chacters from the buffer
    {
      junk = Serial.read();               // Clears the Keyboard Buffer for next input
    }
  }


  // Input Process for Number 2
  Serial.println ("Enter Number 2: ");
  while (Serial.available() == 0);
  {
    number2 = Serial.parseFloat();
    Serial.print ("Number 2 = ");
    Serial.println (number2);
    while (Serial.available() > 0)
    {
      junk = Serial.read();
    }
  }
  sum = number1 + number2;                // Calulates Sum
  Serial.print ("Sum = ");
  Serial.println (sum);                   // Outputs Sum
}
```

2. **Modify the code to do the following**
    a. Change the code to calculate the volume of a cone
    b. Flash
    c. the lights as follows
        i. Volume < 100
            - Flash ON = 2 sec
            - OFF = 2sec
            - 3 Times (USE For Loop or Do/While Loop)
        ii. 100 <= Volume <= 500
            - Flash ON = 1 sec
            - OFF = 3 sec
            - 3 Times (USE For Loop or Do/While Loop)
        iii. Volume > 500
            - Flash ON = .5 sec
            - OFF = 2sec
            - 3 Times (USE For Loop or Do/While Loop)

Part 4: Limiting the times through the loop

NOTE: There is no way to exit the Void Loop() in Arduino. However a delay statement or moving into a loop that does nothing can solve this problem. (I.E delay (100000000); or Create a Loop: while (TRUE) {} (while TRUE will always make the loop true so it will run infinitely.)
Modify the program to do the following

1. After the first time through the program ask the user of they want to find the volume of another cone. Use 0 or Yes and 1 for No
    a. If Yes repeat the program
    b. If No end the program and say “Goodbye!”

Submission: Create a Video or Share from Goolge Drive of the Arduino working >
Email To: jourdem@brightonk12.com