

To lead your students through this activity, you will need your computer, attached to a projector, for projecting your code for students to see.

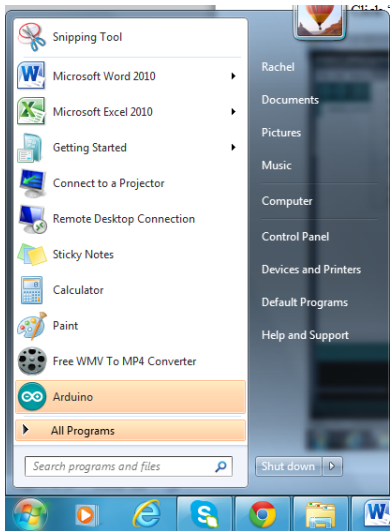
INSTALL THE SOFTWARE

Download and install the Arduino integrated development environment (IDE) (standard build), available here:

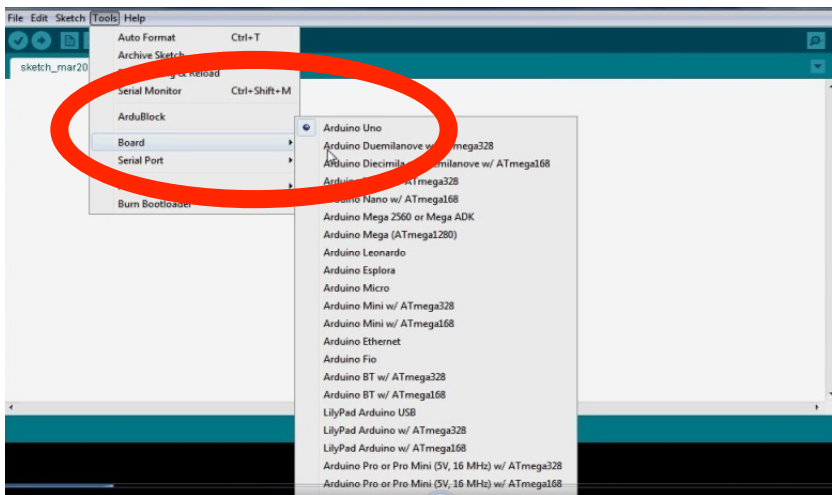
- For Windows, MacOSX and Linux:
 - Standard build: <http://arduino.cc/en/main/software>

STEP 1: PREPARE TO PROGRAM (see U07_L03_06-V1-Arduino1_Set_up_IDE.mp4)

1. **Launch the IDE** by clicking on the Arduino icon in the start menu.



2. **Select the hardware.** Click “Tools,” then “Board,” and be sure “Arduino Uno” is selected.

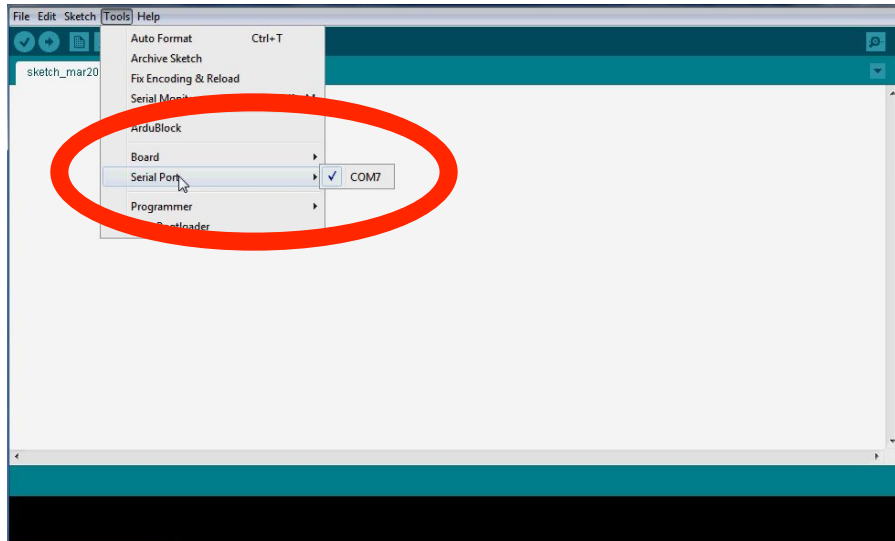


Unit 7: Programming (Electronic Music)

Lesson 3: Programming in Arduino

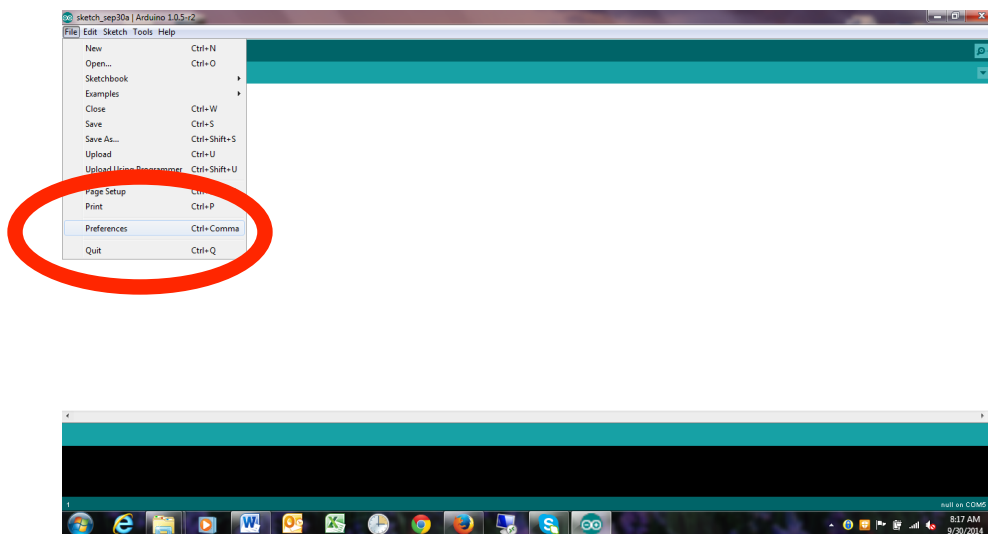
U07_L03_03-Teaching_Demo_Arduino

- 3. Select the port.** Next choose the correct communications (COM) port for the computer to use. If you only have one thing plugged into your computer, then usually there is only one COM port shown, and you can select it with the drop-down menu. Click “Tools,” and “Serial Port.” Select the correct port (it will likely default to the correct port).



If you have multiple ports, you can unplug the microcontroller and see which ports go away. Then plug it back in and see if the port comes back. This is the correct port.

- 4. Change the font size (optional).** If it will help your students to see your generated code more clearly, you may choose to increase the font size. To do this, click on “File,” then “Preferences:”

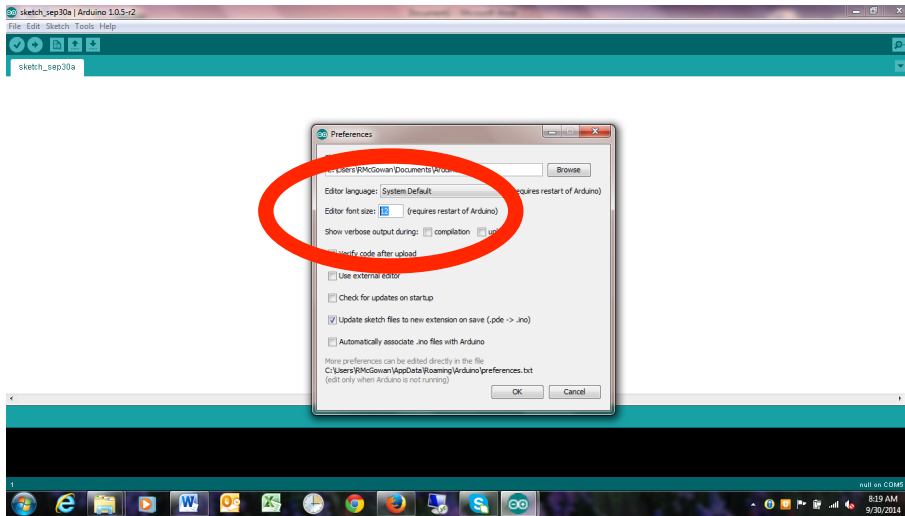


Unit 7: Programming (Electronic Music)

Lesson 3: Programming in Arduino

U07_L03_03-Teaching_Demo_Arduino

You will see a popup box; enter a font size of 16 or 18 in the appropriate box (the default is 12).

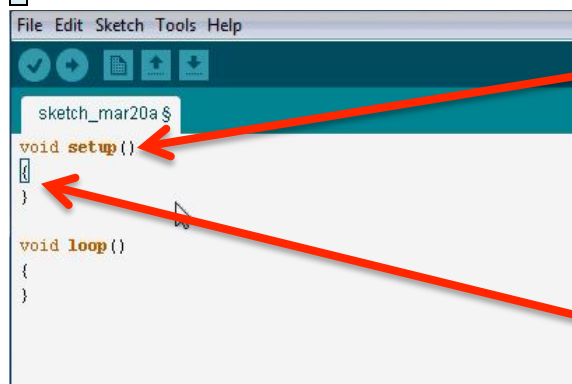


Click “OK.”

Note that it is possible you may need to close and re-open the Arduino IDE in order for this change to take effect.

- Note where and how you will program.** You will use written characters (letters, numbers, and symbols) to create your code. Your code will be displayed in the white box.
- Set up the IDE.** You must type in the following commands in order for Arduino to execute any program:

```
void setup()  
{  
}  
void loop()  
{  
}
```



No text between the parentheses, or the code will not compile.

The program, or code, gets written between these curly brackets.

7. **Test the system.** Type the following information (code) between the curly brackets, noting that the syntax is **<function(arguments);>** and that each line of code must end in a semicolon):

```
tone(12,740);
```

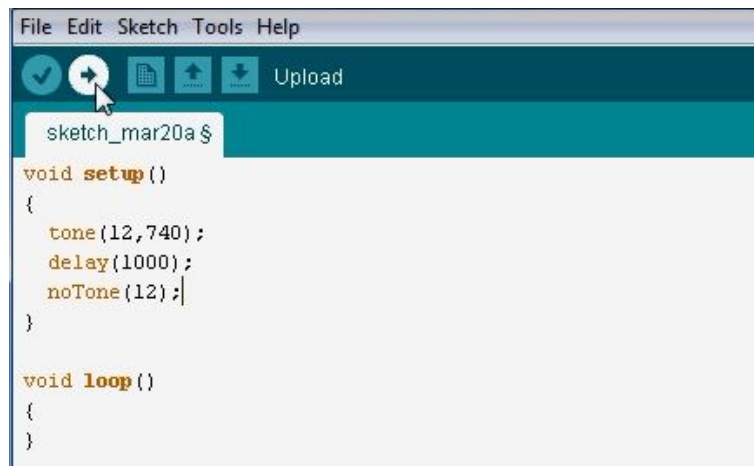
*Plays F# in the fifth octave (frequency obtained from **U07_L03_04-H1_Note_Frequencies**) through the speaker connected at output pin 12 (see system diagram for pin location)*

```
delay(1000);
```

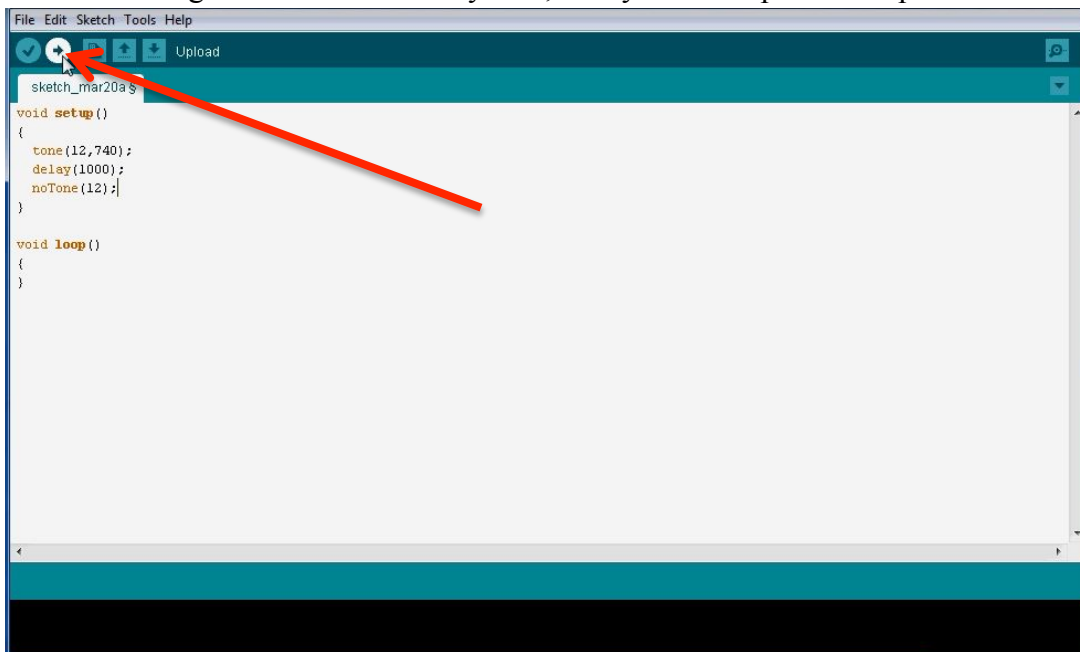
Plays that tone for 1000 ms (0.5 sec)

```
noTone(12);
```

Stops the tone after the specified time



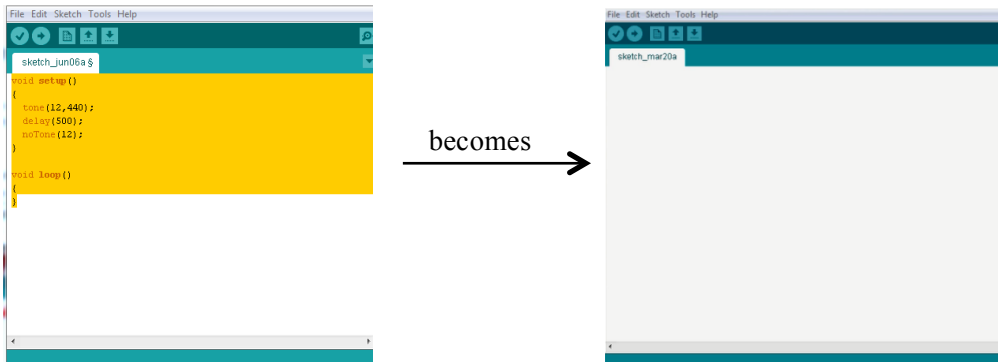
8. Click on the right-arrow to test the system; verify that the speaker beeps.



STEP 2: PLAN YOUR PROGRAM (see U07_L03_07-V2-Arduino2_Comments.mp4)

You will create a program that plays music on your system - specifically, “Row, Row, Row Your Boat” - to see how this all works. You will start with comments; this is good coding practice.

1. **Create a blank slate.** If you have not already done so, use your mouse to highlight the code you wrote for your first note, and press “delete” on your keyboard. This will delete all the code you just wrote.



2. **Insert comments.** To insert a comment, simply type two forward slashes and then the text you would like to enter.



You may want to enter your name, the title of the file, pseudocode (the plain-text version of what you want to happen) and any comments or explanations you may have for other users. (Anything you type in here will not be read as code by the computer and will not be executed; the text entered here cannot mess up your program.)

STEP 3: PROGRAM THE FIRST NOTE. (see U07_L03_08-V3-Arduino3_Functions.mp4)

In Arduino, a command is called a “function.” Just as in mathematics, functions in programming have inputs and outputs. Inputs tell the function what to output or what to do.

1. To write the function for the first note, determine what note starts the song (it is middle C, or C4, with a frequency of 262Hz) and its duration (in our example, this is 300 milliseconds). Program that note as follows:

```
tone(12,262); // tone(pin,frequency)
delay(300); // delay (duration)
noTone(12); // noTone(pin)
```

Note that the numbers in parentheses are the inputs. The comments after the “//” are informative notes and are not required.

2. **Test the programming** by clicking on the right-arrow. You should hear a short C.
3. **Save your program.** Go to “File” and select “Save” or “Save As.” Save the program with a relevant name in a logical location on your computer.
4. **Program the second note.** The second note is a C4 with the same duration as the first. What is the most efficient way to enter this new information? (The video demonstrates how to copy-and-paste code by highlighting the desired code, pressing CTRL-C, placing the cursor where the text is to be pasted, and pressing CTRL-V.)
5. **Program the third note.** This note is also a C4, but with a duration of 200 ms.
6. **Continue programming until all desired notes are present.** You do not need to program all notes, since you will soon (in step 4) use variables to create the whole song.



```
File Edit Sketch Tools Help
Song $
// play row row row your boat
tone(12,262); // tone(pin,frequency)
delay(300);
noTone(12);

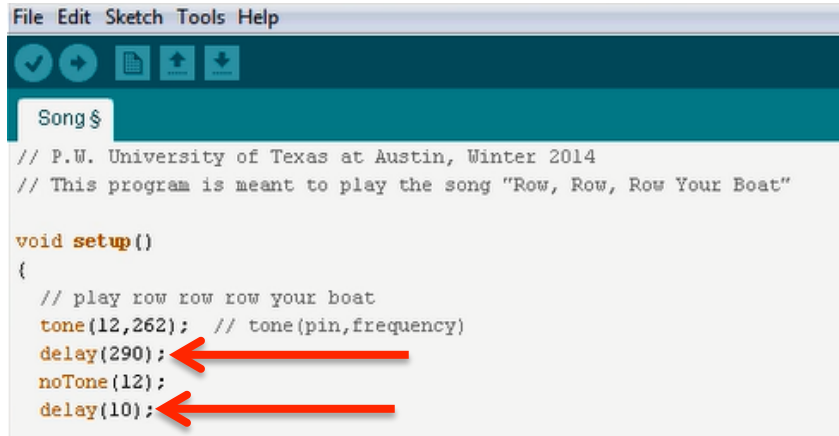
tone(12,262); // tone(pin,frequency)
delay(300);
noTone(12);

tone(12,262); // tone(pin,frequency)
delay(200);
noTone(12);

tone(12,294); // tone(pin,frequency)
delay(100);
noTone(12);

tone(12,330); // tone(pin,frequency)
delay(300);
noTone(12);
```

7. After you have coded the song, run the program and listen to be sure you have no errors. You will hear that the first notes run together, so you might want to insert a slight pause (delay) between repeated notes. Subtract the duration of the pause from that of the note:



```
File Edit Sketch Tools Help
Song $
// P.W. University of Texas at Austin, Winter 2014
// This program is meant to play the song "Row, Row, Row Your Boat"

void setup()
{
  // play row row row your boat
  tone(12,262); // tone(pin,frequency)
  delay(290);
  noTone(12);
  delay(10);
```

STEP 4: DEBUG (see U07_L03_09-V4-Arduino4_Debugging.mp4)

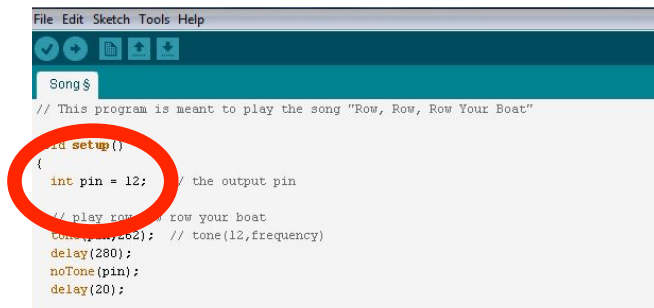
Test the code frequently to find errors. You can change tones and durations directly in the function calls.

STEP 5A: ADD VARIABLES (see U07_L03_10-V5-Arduino5_Variables.mp4)

Variables will speed up the programming (and revising) process. Add variables to the program that you created in Step 2.

1. **Add a variable to identify the pin.** Go to the beginning of your program and initialize a new integer variable to identify the pin. Give the variable a name that makes sense to you; in the video, the first variable is called “pin.”

```
int pin = 12; // the output pin, which is pin 12
```



```
File Edit Sketch Tools Help
Song $
// This program is meant to play the song "Row, Row, Row Your Boat"

void setup()
{
  int pin = 12; // the output pin
  // play row row row your boat
  tone(pin,262); // tone(12,frequency)
  delay(280);
  noTone(pin);
  delay(20);
```

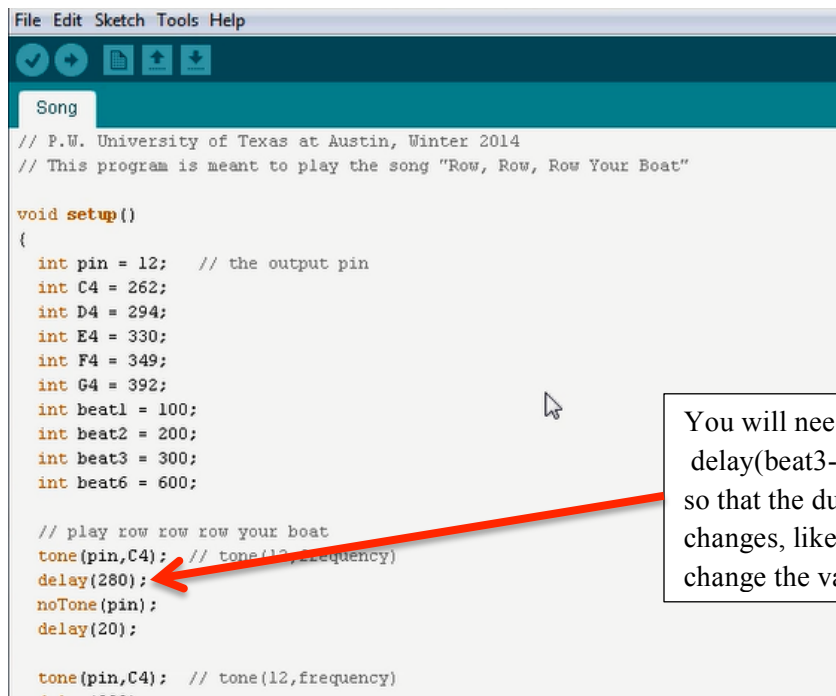
2. **Use that variable.** Everywhere you had the number “12” entered to indicate the pin, you can replace that “12” with “pin”. Now if you move the wire to a different pin, you only need to change one value (that of the variable) and the pin number will be correct everywhere. Furthermore, when entering future lines of code you won’t have to worry about a typographical error such as entering “13” instead of “12”.
3. **Add a second variable** (with a different name) immediately below the first variable; in the video, the second variable is called “C4” and the assigned value is 262 (Hz).

```
int C4 = 262; // frequency of C4
```

4. Add more variables.

```
int D4 = 294; // frequency of D4
int E4 = 330; // frequency of E4
int F4 = 349; // frequency of F4
int G4 = 392; // frequency of G4
int beat1 = 100; // duration of 100ms note
int beat2 = 200; // duration of 200ms note
int beat3 = 300; // duration of 300ms note
int beat6 = 600; // duration of 600ms note
```

5. Use the variables in your program. Wherever you had specified a frequency of 262, use C4; wherever you had a frequency of 294, use D4; and so on. Also replace individual durations with the appropriate variables (e.g., “beat1”, “beat 2”). (Note that this is not done in the video, so the following screenshot is from the beginning of the next video.)



```
File Edit Sketch Tools Help
Song
// P.W. University of Texas at Austin, Winter 2014
// This program is meant to play the song "Row, Row, Row Your Boat"

void setup()
{
  int pin = 12; // the output pin
  int C4 = 262;
  int D4 = 294;
  int E4 = 330;
  int F4 = 349;
  int G4 = 392;
  int beat1 = 100;
  int beat2 = 200;
  int beat3 = 300;
  int beat6 = 600;

  // play row row row your boat
  tone(pin,C4); // tone(12,frequency)
  delay(280);
  noTone(pin);
  delay(20);

  tone(pin,C4); // tone(12,frequency)
  ...
}
```

You will need to replace this with `delay(beat3-20);` so that the duration of this note changes, like the others, when you change the variables.

Play the program and make sure that everything is still correct. Save your program! (You should be doing this periodically.)

STEP 5B: PROGRAM LINE 2 (see U07_L03_10-V5-Arduino5_Variables.mp4)

Program the second line of the song. Start by entering a new comment—in the video, the comment is `//gently down the stream`—so that someone looking at your code will know where the second line of the song begins. The code should look like:

Unit 7: Programming (Electronic Music)
Lesson 3: Programming in Arduino

U07_L03_03-Teaching_Demo_Arduino

```
tone(pin, E4);  
delay(beat2);  
noTone(pin);
```

```
tone(pin, D4);  
delay(beat1);  
noTone(pin);
```

```
tone(pin, E4);  
delay(beat2);  
noTone(pin);
```

```
tone(pin, F4);  
delay(beat1);  
noTone(pin);
```

```
tone(pin, G4);  
delay(beat6);  
noTone(pin);
```

STEP 6: ADD OPERATORS (see **U07_L03_11-V6-Arduino6_Mathematical_Operators.mp4**)

1. **To prepare for the need for operators**, change all of your variables to half of their current values. In other words, change the values so that

```
int beat1 = 50;  
int beat2 = 100;  
int beat3 = 150;  
int beat6 = 300;
```

You will hear that the song plays very quickly. Now change all of the variables again so that

```
int beat1 = 200;  
int beat2 = 400;  
int beat3 = 600;  
int beat6 = 1200;
```

This makes the song too slow. You could keep changing all of these values until you find the right tempo, or you could use operators to make the process quicker.

2. **Use operators** to make the value of best2, beat3, and beat6 depend on the value of beat1. In particular, you will code:

```
int beat1 = 200;  
int beat2 = beat1 * 2;  
int beat3 = beat1 * 3;  
int beat6 = beat1 * 6;
```

(In the video, “beat6” is dependent on “beat3” rather than “beat1”; either way is correct.)

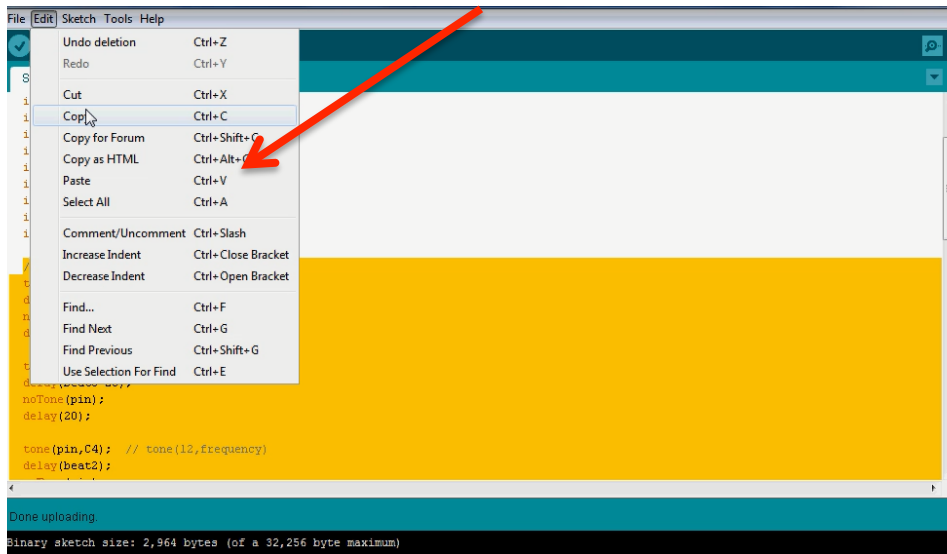
3. **Save your program and play your song.** If the variables and operators are correctly related, your song should sound correct (although the tempo may still be too slow).

4. To see the power of operators, change the value of “beat1” to any duration you choose (the video uses a value of 97). This will change the tempo of the entire song without you needing to adjust any of the other time variables. Play the song at the new tempo.
5. Use variables and operators to code the rest of the song.

STEP 7: USE LOOPS FOR REPEATED PHRASES: (see U07_L03_12-V7-Arduino7_Loops)

This step of coding is more complex, and some students and teachers may choose not to use this process (although we recommend it). Due to the complexity of the use of “for loops,” a separate document is available for teacher reference (see U07_L03_05-R2_For_Loops). To conduct the activity as in the video:

1. **Identify the instructions (functions) to be repeated.** Copy the first two lines of the song (`//row row row your boat`) and (`//gently down the stream`) and paste the code immediately below the end of those first two lines (before `//merrily merrily merrily merrily`).



Play the song and verify that the first two lines repeat once.

2. **Identify why this method may be ineffective.** What if the engineering team wanted the code repeated ten times? The code would become long and messy.
3. **Use "for loop" to repeat code more efficiently.**
 - a. Begin by deleting what you just added to your first two lines, so that your code only reflects the first two lines of the song.
 - b. Type this immediately before the beginning of the code that you want to repeat:

```
for (int i=0; i<2; i++) {
```

- c. Highlight all the code you want to repeat (the entire first two lines of the song); press "Tab" on your keyboard to indent it; and below the end of the code that will repeat, insert the closing curly bracket: }

This tells the computer (the Arduino) that everything inside those curly brackets should be repeated one time.

Simply stated, the code means this (plain text "translation"):

"For" a specified number of times beginning at zero `for (int i=0;`

If "i" is less than 2 `i<2;` (which it is, since $0 < 2$)

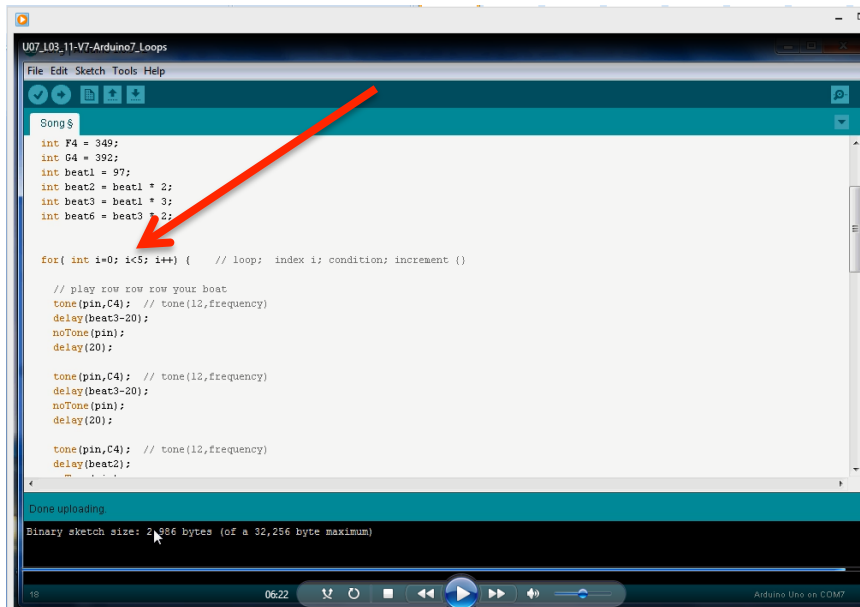
Add "1" to "i" `i++`) ("i" now becomes "1")

Everything inside the curly brackets will be executed once and i will have a value of 1.

The "for loop" then takes us back to the top. Since $i=1$ is less than 2, 1 will be added to i (so i is now 2) and the code inside the loop will repeat.

The "for loop" then takes us back to the top. Since $i=2$ is NOT less than 2, the code is not repeated. Instead, the program continues to the code that follows the "for loop".

4. **Comment your loop function:** // loop; index i; increment { }
5. **Play your code** (and save it, if it is correct).
6. **Change the condition** so the code repeats five times. This means you will need to change `i<2;` to `i<5;`



```
U07_L03_11-V7-Arduino7_Loops
File Edit Sketch Tools Help
Song $
int F4 = 349;
int G4 = 392;
int beat1 = 97;
int beat2 = beat1 * 2;
int beat3 = beat1 * 3;
int beat6 = beat3 * 2;

for( int i=0; i<5; i++) { // loop; index i; condition; increment ()

// play row row row your boat
tone(pin,C4); // tone(12,frequency)
delay(beat3-20);
noTone(pin);
delay(20);

tone(pin,C4); // tone(12,frequency)
delay(beat3-20);
noTone(pin);
delay(20);

tone(pin,C4); // tone(12,frequency)
delay(beat2);
}

Done uploading.
Binary sketch size: 2,986 bytes (of a 32,256 byte maximum)
```

7. **Play the new code to verify it works correctly.**
8. **If your song allows for repetition,** continue coding your song using loops.