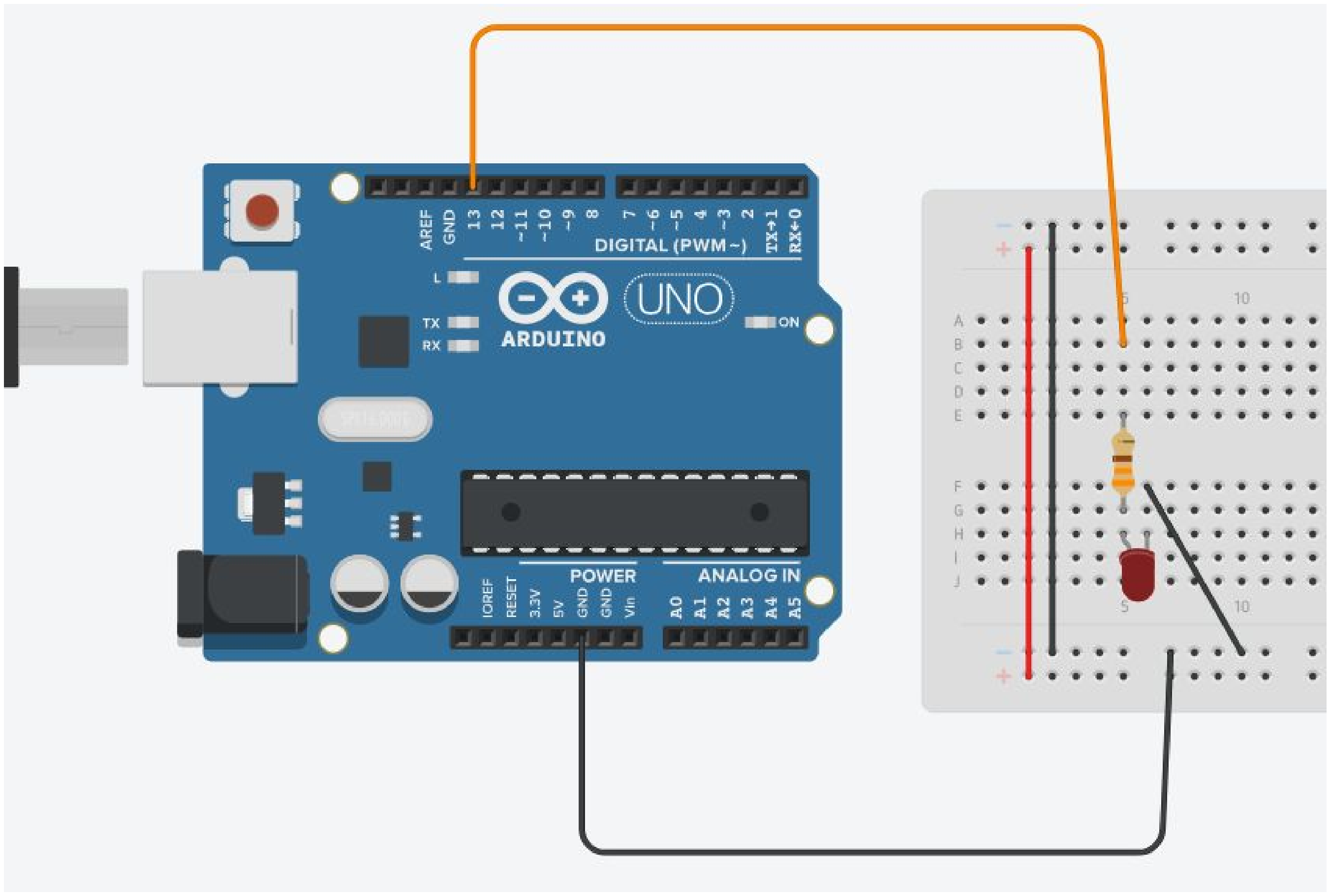
Exercise 1 - Morse Code

This document lets you experiment with basic digital output involving one LED. This exercise will give you practice using the sequential structure to write programming code.

Instructions:

Go to the TinkerCad site (tinkercad.com) and login using your username and password. Under components, type in Arduino and drag it onto the stage. Do the same for the breadboard, resistor, and LED. Next, wire the circuit exactly as it appears below. Notice that the LED and resistor (330 Ohms) are wired in series with digital I/O pin 13. You will reference the appropriate pin # in your code. Once you have wired the circuit, click on Code at the top of the screen. Initially it will be in block mode (similar to Scratch) so you will need to change it from blocks to text. Here is where all of your code is going to be typed in.



Program 1: Morse Code (Erase all of the code that is currently in the text area and then copy and paste the text in the box on the next page). **Going forward, every Arduino program you write will have a header at the top.** Remember that the header part must be filled out fully with your name, date, and a brief description of what the program does).

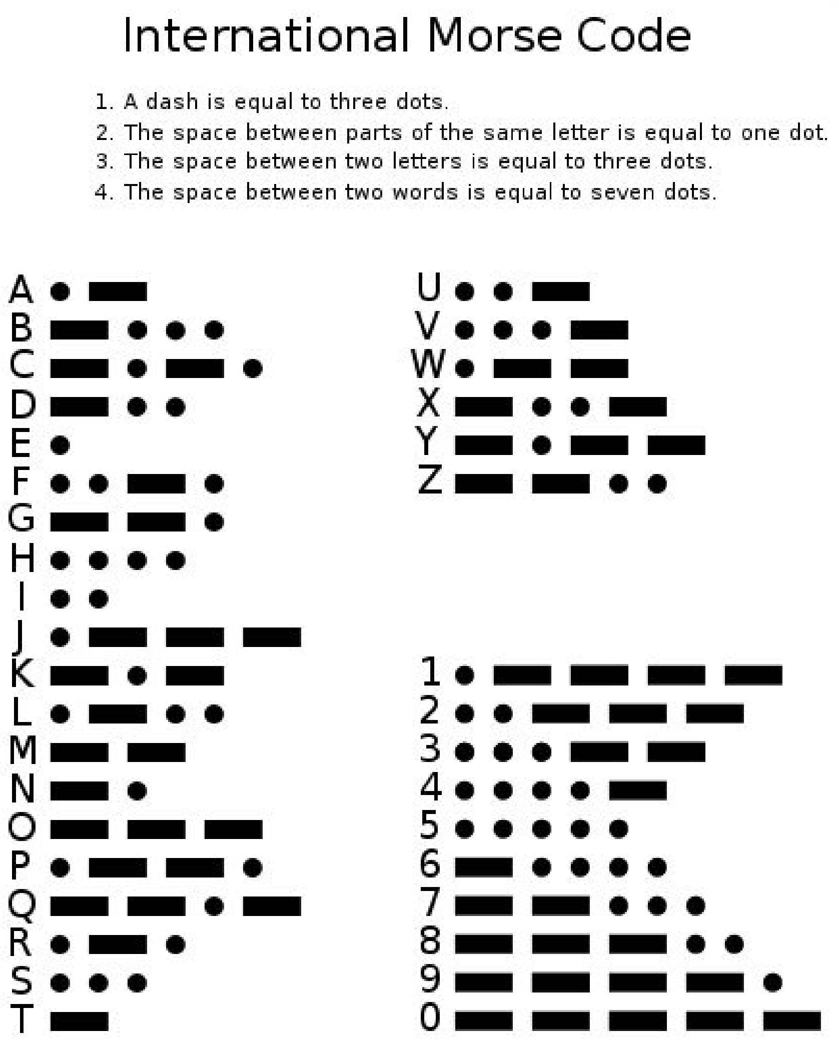
```
/*
Author:
Date:
Description: This code is written to spell out a student number using a morse code blinking pattern.
*/
void setup()
{
        /*  This is a comment spanning multiple lines.  It does not run and is only intended for the
            programmer to see.
            We first set the LED pin as output using the pinMode command. The general pinMode
            command is pinMode(pin number being used, OUTPUT or INPUT);
        */
        Serial.begin(9600); //opens serial port and sets data rate to 9600 bits per second
        Serial.println("Starting the program ....");
        pinMode(13, OUTPUT); // this is an example of a single line comment
 }

void loop()
{
        /*
        In the loop we set the LED on first for a second and then we turn it off. This is done by the
        digitalWrite and delay commands. The general form of digitalWrite and delay are:
        digitalWrite(pin_number, HIGH/LOW);
        HIGH turns on the led and LOW turns the LED off. Delay is used as delay(time in
        milliseconds); this delays the time between the execution of 2 commands.
        */
        Serial.println("This part of the code turns an LED on/off every second.");
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(13,LOW);
        delay(1000);
}
```

Execute your code now by clicking on **START SIMULATION.** If the LED blinks then you have wired your circuit correctly. Move on to the next page to modify the code.

Morse Code Overview

Letters and numbers are represented in Morse code by long and short sounds, usually called dots and dashes. A dash should equal three dots in length.

Did you know…. Morse code is a way of communication using dashes and dots of light, sound or radio waves. For example, when the Titanic signalled for help it sent ● ● ● ▬ ▬ ▬ ● ● ● over the radio, which stands for S.O.S (short for Save Our Souls).



International Morse Code

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to seven dots.

A	● ▬	U	● ● ▬
B	▬ ● ● ●	V	● ● ● ▬
C	▬ ● ▬ ●	W	● ▬ ▬
D	▬ ● ●	X	▬ ● ● ▬
E	●	Y	▬ ● ▬ ▬
F	● ● ▬ ●	Z	▬ ▬ ● ●
G	▬ ▬ ●		
H	● ● ● ●		
I	● ●		
J	● ▬ ▬ ▬		
K	▬ ● ▬	1	● ▬ ▬ ▬ ▬
L	● ▬ ● ●	2	● ● ▬ ▬ ▬
M	▬ ▬	3	● ● ● ▬ ▬
N	▬ ●	4	● ● ● ● ▬
O	▬ ▬ ▬	5	● ● ● ● ●
P	● ▬ ▬ ●	6	▬ ● ● ● ●
Q	▬ ▬ ● ▬	7	▬ ▬ ● ● ●
R	● ▬ ●	8	▬ ▬ ▬ ● ●
S	● ● ●	9	▬ ▬ ▬ ▬ ●
T	▬	0	▬ ▬ ▬ ▬ ▬

For the purposes of our program, we will represent dashes by the LED staying on for one second (or 1000 ms) and by dots making the LED stay on for 333 milliseconds. In between the dots and dashes representing the same letter or number, the LED will stay off for 333 ms. If you are on the last dot/dash of a letter or number, then the pause in between is equal to one second.

Example:
In the loop() function, you can replace existing code to represent the letter R:

```
void loop()
{
      /*prints to the monitor to let you know what letter / number is being
        Represented */
      Serial.println("This part of the morse code that represents the letter R.");
      digitalWrite(13, HIGH); // this represents the dot
      delay(333); //stays ON for a third of a second (333 milliseconds)
      digitalWrite(13, LOW);
      delay(333); //the LED will stay OFF for 333 seconds
      digitalWrite(13,HIGH); //this represents the dash
      delay(1000);
      digitalWrite(13, LOW);
      delay(333);
      digitalWrite(13, HIGH); // this represents the dot
      delay(333);
      digitalWrite(13, LOW); // this represents the OFF
      delay(1000); // this represents the one second OFF period before the next
                   // character is shown.
}
```

Your next step is to add a line above where you see **void setup()** and declare a new variable. Type this line out exactly as it appears:
int ledPin; //this is how to declare an **int**eger variable called ledPin

Inside the setup() function, copy this line of code in:
ledPin = 13; //assign the variable ledPin a value of 13.

Throughout your program, wherever you see the number **13**, replace it with the variable name **ledPin**
Now, if you ever need to change the pin number the LED is connected to, you only need to update the variable and not every line of code. This is a very useful technique in programming and it will save you time on larger projects. Use **ledPin** in your code going forward.

Next, declare two more global integer variables called dot_time and dash_time above the setup function. Set dot_time to 333 in the setup function and set dash_time to 1000 in the setup function as well.

Remember to comment your code throughout the program and use Serial.println() statements to write to the console to say what the code is doing.

Your Turn: You will spell out the last three digits of your student number using the morse code pattern. Remember to comment your code and also print out to the screen what is happening using Serial.println. You only need to print out which number you are on so you can visually check that your circuit is functioning properly. Printing out text to the screen is helpful for troubleshooting.

You would have noticed that you have a lot of lines of code in the loop() function and it is very repetitive. Every time you represent a dot or a dash it always consists of two parts: turning ON and

then OFF the LED. We can simplify our code by introducing your own function! Read the Functions PowerPoint located here:
https://drive.google.com/open?id=1FiuGc91JcVAe6i7fz1ftT7IJoCGIZM29QJDAmfq89iM

Before the setup function but after your global variable, declare a new function called flash. Void means that the function does not return a value. Pin is the pin number being used for the LED, on_time is how many milliseconds the LED will be on for and off_time is how long the LED will be off for (in milliseconds).

```
void flash(int pin, int on_time, int off_time)
{
      digitalWrite(pin, HIGH);
      delay(on_time);
      digitalWrite(pin, LOW);
      delay(off_time);
}
```

Now within your loop() function, you can call your new function flash.

Example:
```
void loop()
{
      flash(13,333, 333);
}
```

Now what used to take you four lines of code, now takes one line of code! This may not seem like a significant improvement but as you work on larger scale projects, it will save you a lot of time and space in memory.

TO SUBMIT:

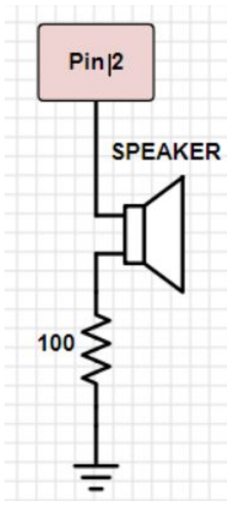
Once you are finished and have tested out your code, click on the ⬇ symbol. The Arduino file will appear in your Downloads folder. Locate the .INO file and rename it to
yourFirstName_MorseCode.ino
Where it says **FirstName**, you will type in your name (e.g. Robert).

You will screen record your circuit working. I recommend screencastify which is a popular Google Chrome plugin but you can use any program you wish. You will use a microphone to narrate the inner workings of your circuit (e.g. components in your circuit) and you will walk me through your code and explain what is happening at each and every step of your code.

Upload your Arduino programming file and video to this Assignment post. There is still one more program to write so do not click on "Turn In" yet until you have completed both programs.

You will complete ONE of the two extensions listed below. You can do both if you would like to. Extension #2 is a bit harder.

EXTENSION 1:
Disconnect the LED and resistor from your circuit. You will be adding a new component: a buzzer. . Connect the anode of the speaker to pin 12 through a jumper wire. The cathode will connect to a 100 Ohm resistor that goes to ground (GND -). Use one specific note to represent a dash and a DIFFERENT note to represent a dot.



TO DO: Do some research on the ***tone*** function and make a series of beeps to show the morse code pattern instead of showing it only on the LED.

Once your code is working, change the name of your program to
yourFirstName_MorseCode2.ino

Now think of your own extension to this assignment that you think would be suitable for a TEJ3M class. Explain it and make sure you meet the requirements from the rubric and the Checklist for the Arduino Portfolio.

EXTENSION 2:

Use an RGB LED instead of the regular LED. RGB LED's use PWM signals to change the values of the red, green, and blue connectors to display a specific colour. Use a different colour for dashes and dots. Create your own functions to simplify your code, especially for the colours that you choose. This requires some research on your part but since Arduino is considered open source hardware it is very easy to find sample code online and tutorials.

To submit:
Download your code and rename it to **yourFirstName_MorseCode_extension.ino**
Create a video of your circuit working and explain what is happening in the code. Upload your video.

Answer the portfolio questions.