day70 Arduino blink & superblink blink, superblink
3/1/22 and 3/2/22

Before you plug in your Arduino, is there a folder on the Desktop with your name? If not, please create a new folder with your name as the folder name. This is where you will store your Arduino sketches.

Launch the Arduino IDE (integrated development environment). If it asks you if you want to upgrade say "no".

Plug your Arduino in to one of the four USB ports on the front of your PC.

Part 1:

When you first open the IDE you get an empty, blank sketch. Save this sketch to your folder with this name "blink_yourname". By default it should contain the following:

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

You're going to start by writing the program that Mr. Hays just did on the board.

Replace the line in the **void setup** section that starts with "//" with the following:

```
  pinMode(13,OUTPUT);
```

so you end up with this:

```
void setup() {
  pinMode(13,OUTPUT);
}
```

The command we wrote tells the Arduino that we're going to be using pin 13 as an output pin. We'll be using some pins as inputs and some as outputs in our various projects.

Anything you put after two slashes (//) is a comment. Put a comment in at the top of your document with your name and the date on it. Do this right now. Comments are ignored by the compiler, but they are helpful to people, for example, to me when I'm grading your work.

In the loop function, replace the line that starts with "//" below loop() with the four lines shown below:

```
void loop() {
  digitalWrite(13,HIGH);
  delay(1000);
  digitalWrite(13,LOW);
  delay(1000);
}
```

Go to the Tools>Board menu and make sure that the "Arduino Uno" board is selected. Go to the Tools>Port menu and choose the active COM port that has "Arduino Uno" after it. If there is no COM port listed with the text after it, call me over and we'll figure out what is going on.

Click the Verify button (the left-most button that looks like a check mark) to make sure that you entered the code correctly. The Verify button tries to compile your program. If you have typed it in correctly, the bottom of the window will report something like this:

```
Sketch uses 928 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local
variables. Maximum is 2048 bytes.
```

If you get any errors we need to fix your code before we can continue. The most common error is something about a missing semicolon. For the most part every line that doesn't start or end in a { or } needs a semicolon. Read the error text to see if you can figure out what is wrong.

Once your program compiles, save it again, then click the right arrow icon, which tells the IDE to upload your sketch to the Arduino. If no errors occur, you will see an LED light on the board blink on and off, one second on, one second off. That's what our code above does.

Here's a brief breakdown of what is going on in our loop function:

- "`pinMode(13,OUTPUT)`" says that pin 13 will be used for turning something on and off.
- "`digitalWrite(13,HIGH)`" tells the program to turn on pin 13 ("HIGH" means "on").
- "`delay(1000)`" tells the program to wait 1000 milliseconds (1 second)
- "`digitalWrite(13,LOW)`" with a "LOW" tells the program to turn off pin 13.

So our program will do this forever, as long as it receives power: turn the light on and off every second. Not the most profound thing to be doing, but an excellent way for us to get started.

Change one of the times: Make the delay after the HIGH call be longer than the one after the LOW call. Save and re-upload your code to the Arduino. Can you see the difference? Remember that we're dealing with milliseconds, so you're not likely to notice the difference if you don't make a big change. But don't make too big of a change, or you'll sit there having to wait that length of time.

Copy and paste the on/off/delay lines so you have two sets. Change the times on one set compared with the other. Upload. (You don't have to do a Verify before trying to Upload, by the way, Arduino will tell you if there is an error before it Uploads your code.)

Show a neighbor. Play with the delay times. Explore. Experiment. Have some fun.

Call me over to see your program.

Part 2:

Do a Save as… and save a new sketch also in your folder on the Desktop named super_blink_yourname. Arduino doesn't let you put spaces in sketch names.

We're going to use a variable to set how long the delay on/off times are for the LED light.

In Python we could make a variable by saying "x = 0". In Processing you have to say what type of data a variable is going to hold before you can use it. This is called "declaring" a variable. We do this **above** the setup() and loop() functions, below your comment at the top that has your name in it.

Type the following above "void setup()" and below the comment with your name:

```
int x = 1;
```

The above line creates a variable called "x" that will hold integer values (whole numbers). Then we're storing a 1 in x. After declaring it, the variable works just like variables in Python or Scratch.

Then in the loop() function use x instead of a number in both the delay commands:

```
delay(x);
```

Also, change your code to only have one set of ons and offs (that is a HIGH call, a delay, then a LOW call and another delay). Change all delay calls to delay(x). Then after the last one, add this line:

```
x = x + 1;
```

This will make x get larger and larger very quickly, making the blinking happen slower and slower. Upload your new sketch to the board and see how it works.

The Arduino board has a reset button (near where the USB cable plugs into the board). Press this to make your program start over. (The program automatically starts over whenever you power up (i.e. plug in) the Arduino, upload a program, or press the reset button.)

This program starts blinking the light on and off super fast, getting slower as it goes, in fact, one millisecond slower each cycle through. Before long it gets very very slow.

Part 3:

Open the Reference item from the Help menu in the Arduino application. Nearly all possible Processing commands are listed here.

Look at the **if** command entry, and add an if statement in your loop() function to set the value of x to 0 if it gets larger than (>) 1000. If statements are structured using curly brackets { }. One way to think about it is that in Python you kept everything that was "inside" of an if statement indented under the if statement. With Arduinos we keep all the commands inside of an if statement between curly brackets. Look at the examples, and set up an if statement inside your loop() function that resets x to zero if it x is greater than 1000. Change your x = x + 1 line to x = x+100 so that x grows much faster. Upload the program and see it run. Can you see when the if statement is triggered? Play with the values and see if you can do something different or interesting.

Get me to check you off when you are done. If you're done and you want to keep playing with it, can you make the light blink slower and slower, then switch to going faster and fast, then repeat? Do something totally cool and show it off. (Do a save as again before you mess with your code, though.)

Turn in your superblink sketch on the Google Classroom. It is in a folder with that name where you saved it. If you want to explore more info about Arduinos click on this link: https://www.arduino.cc/