Engineering Principles Name_____
Electrical Engineering
Color Wheel 4.6.1 Software: Test Date_____ Team _____

Problem statement

Develop a persistence-of-vision color wheel utilizing 8 RGB diodes and an Arduino UNO R3 board.

Programming: Getting Started

1. Open the file that you used for the prior assignment. If the file has been modified by another member of your team since you last worked on it you may need to download it from Canvas.
    a. Go to your group's home page (under the Courses menu)
    b. Select Files and find the latest version of your file
    c. Save the file in the directory that it was in before – should be the same name as the file you are downloading.
2. Open the file with Arduino

Programming: Variable Scope

A variable is a place in memory that we have given a name so that we can reference it easily. It is used to store information. In C, variables can only be seen in the section of code that they are defined in. If a variable is defined in the code file outside of any functions, like we did with our color pin arrays, its scope is global; every function in the file can see and reference the variable. If we define a variable inside a function, only *that* function can see the variable. Another function could have a variable of the same name, but it wouldn't necessarily be the same value. We can also define a variable inside a loop (or any set of curly braces { }). It is good programming style to define variables within the smallest scope possible to get the job done.

Programming: Loops

`for` loops allow us to repeat the same operation a fixed number of times. It will be very useful in our code as we crunch through the rows and columns of our image files. `for` loops have three sections in parentheses (), each separated by a semicolon.

```
for (initializer; end condition; increment) {statements;}
```

- *initializer* runs once before anything else happens. It is common to declare the counting variable in *initializer*. This variable will only be defined within the scope of the `for` loop.
- *end condition* checks if we are done yet. If the value of the expression is TRUE, the for loop ends and nothing else happens.
- *increment*, executes after all the *statements*, just before *end condition* is checked. It is normally used to count how many times the loop has run. The ++ operator adds one to the variable that it follows or proceeds. You could also write i += 1 or i = i + 1 to get the same result.

```
for (int i=0; i<8; i++) {
  clearColorPin(i);
}
```

The code above will run 8 times. The variable `i` will range from 0 to 7.

3. Create a function called clearRowPins(void) which uses a for loop to turn off all eight of the LED row pins.
4. Create a second function called setRowPins(void) which uses a for loop to turn ON all eight of the LED row pins.

5. In the setup() function after all the initialization code we wrote in the previous assignment create a for loop that turns on each color (red, green and blue or 0, 1, 2) in turn and then waits 1 second (delay(1000);) Before the for loop call setRowPins(); after it call clearRowPins.

6. **In your lab notebook** write down what you expect the code to do?

7. Test this code using your group's hardware.
   In your lab notebook describe what your code actually did. If different from your expectations, explain. Share the results with your team.

For loops can be nested. This means that one of the statements inside a `for` loop is another `for` loop. When nesting `for` loops it is common to indent each loop to help keep track of things – especially the curly braces. The following code creates a two-dimensional array called timesTable and populates it with the answers we learned back in elementary school.

```
int timesTable[10][10];
for (int i=0; i<10; i++) {
  for (int j=0; j<10; j++) {
     timesTable[i][j] = i * j;
  }
}
```

8. Create a set of three nested for loops. The outermost loop should cycle through each of the eight row pins in turn, turning on that row at the beginning, and turning it off at the end. The next inner loop (between the on and off commands) should count to 128. The inner most loop should cycle between the three colors setting each and then waiting for a short time (1 or 2 mSec).

9. **In your lab notebook** write down what you expect the code to do.
10. Test this code using your group's hardware.
11. **In your lab notebook** describe what your code actually did. If different from your expectations, explain. Share the results with your team.

Programming: Finish and Save

12. Save your code and upload it to your group's file space on Canvas. Use a new name so that prior versions of the code are not over-written. You should also submit it to your teacher as instructed.