**Outline**

Implement the remaining Arduino beginner lessons. Modify these lesson programs to expand the use of “for” loops, “if” statements, and serial console input and output.

**Objectives**

* Use strings,
* Demonstrate the ability to manipulate string data in a computer program
* Use assignment statements correctly with arithmetic expressions in computer programs
* Use comparison operators (i.e., equal to, not equal to, greater than, less than, greater than or equal to, less than or equal to),
* Write programs that incorporate user input,
* Write programs that incorporate screen output;
* Demonstrate the ability to manipulate and convert data in a computer program (e.g., parse strings; convert data types such as numeric to string, and string to numeric; convert ‘yes’ or ‘no’ to Boolean);
* Use sequence control structures to create programming solutions;
* Use repetition control structures to create programming solutions;

**Prerequisites**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Prerequisite Module(s)** | **Level** | **Student Initial** | **Teacher Initial** | **Date** |
| Module B.1 Arduino Basic Blink | Level 3 |  |  |  |

**Materials**

* Arduino Development Environment (IDE)
* Arduino proto board
* Web Documentation: “Smraza UNO Documentation” folder in the “ICS3C0” Repository
  + Getting started guide.pdf
  + Public\_materials🡪Ebook🡪Arduino book.pdf
  + Lessons Folder

**Level 0: LED Trailing Effects**

1. Implement the lesson titled “LED Trailing Effects”.
2. Locate on-line documentation that describes the C language “for” loop.

the loop() function does precisely what its name suggests, and loops

consecutively, allowing your program to change and respond. Use it to actively

control the Arduino board.

Example:

int buttonPin = 3;

* 1. What is the index and how is it used?

A "For" Loop is used to repeat a specific block of code a known number of times

For example, if you want to check the grade of every student in a class, you loop from 1 to that number.

* 1. When does the for loop end?

If you test at the end of the loop, the loop always runs at least one time.

* 1. How is a “for” loop different from a “while” and a “do” loop?

A do/while loop will always execute the code in the do{} block first and then evaluate the condition. While for loop allows you to initiate a counter variable, a check condition, and a way to increment your counter all in one line

1. Research the “<” Comparitor.
   1. List all the other comparitors defined for the C language.
      * Standard C library provides qsort() that can be used for sorting an array the function uses QuickSort algorithm to sort the given array.
   2. Modify the “for” loop to use the “<=” comparator

<= (less than or equal to) is used in conjunction with a comparison operator, tests whether a certain condition has been reached, such as an input being above a certain number.

1. Research the “++” incrementor operator.
   1. Explain how this is different from the “=+ 1” assignment

++ is a unary operator that can be applied to variables and increments the value they hold. For

example often for loops have, as their increment-expr something like counter++

* 1. Modify the “for” loop to use the “=+” assignment

+= (compound addition) Performs a mathematical operation on a variable with another constant or variable. The += (et al) operators are just a convenient shorthand for the expanded syntax, listed below.

**First Code:**

int BASE = 3;

int NUM = 8;

void setup()

{

for (int i = BASE; i < BASE + NUM; i ++)

{

pinMode(i, OUTPUT);      //set port ‘i’ as an output port

}

}

void loop()

{

for (int i = BASE; i < BASE + NUM; i ++)

{

digitalWrite(i, LOW);      // Turn OFF the I/O board LED

delay(200);

}

for (int i = BASE; i < BASE + NUM; i ++)

{

digitalWrite(i, HIGH);    // Turn ON the I/O board LED

delay(200);

}

}

Attachments area

**Modified Code:**

int BASE = 2;

int NUM = 6;

void setup()

{

for (int i = BASE; i <= BASE + NUM; i ++)

{

pinMode(i, OUTPUT);      //set port ‘i’ as an output port

}

}

void loop()

{

for (int i = BASE; i <= BASE + NUM; i ++)

{

digitalWrite(i, LOW);      // Turn OFF the I/O board LED

delay(200);

}

for (int i = BASE; i <= BASE + NUM; i ++)

{

digitalWrite(i, HIGH);    // Turn ON the I/O board LED

delay(200);

}

}

**Level 1: Traffic Light**

1. Implement the lesson titled “Traffic Light”.
2. Research the C language “if” statement. Think about how to use it to select the different actions for different traffic lights.
3. Modify the program to replace the three blocks of code (green, yellow, red light) with a single “for” loop.
   1. Use an “if” statement to create a special case for the yellow light.
   2. Implement the countdown timer for the yellow light as a “nested” loop.

**Level 2: Serial Monitor**

1. Locate on-line documentation for the implementation and use of the Arduino serial monitor.
2. Modify the traffic light to print status information to the serial monitor.
   1. Colour of the light
   2. Countdown index for the yellow light

**Level 3: Fading**

1. Implement the lesson titled “Fading”.
2. Modify the main “for” loop to decrement from 255 down to 0.
3. Modify the program to read a number from serial monitor and use that number in the fade down to loop.
4. Modify the program to add code to check that the value read from the serial monitor is a valid number and not some random string.

**Level 4: Traffic Control**

1. Modify the traffic control program to read a string from the serial console. If the string is “red”, “green’, or “yellow”, the light should immediately change to that colour.

**Achievement Record**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attainment Level** | **Student Initial** | **Teacher Initial** | **Date** |
| Level 0: Trailing Effects |  |  |  |
| Level 1: Traffic Light |  |  |  |
| Level 2: Serial Monitor |  |  |  |
| Level 3: Fading |  |  |  |
| Level 4: Traffic Control |  |  |  |