Digital Technology

Programming and Hardware Fundamental using

ARDUINO UNO

Level 1

Category :12 to 18

In Collaboration with

myFlowlab™

SG Academy ™ acknowledge that there may be errors or omissions in this publication for which responsibility cannot be assumed. No liability will be accepted for loss or damage resulting from the use of information contained in this documentation or from uses as described.

Copyright © 2019 SG Academy ™. All rights reserved

Module #1

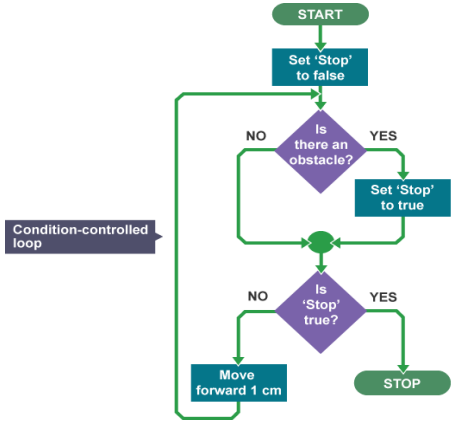
Algorithm

al·go·rithm

a rule (or set of rules) for a software program to solve problems by taking in information and giving out solutions

Algorithm VS Computer Program

Algorithm



Computer Program

```
243
244 void ProgramManager::RecallProgram() {
245     string name;
246     bool searching = true;
247     while (searching) {
248         cout << "Enter name of program: ";
249         cin >> name;
250         if (ProgramNameIsValid(name)) {
251             if (ProgramFileExists(name)) {
252                 Program thisProgram;
253                 thisProgram.SetName(name);
254                 Read(thisProgram);
255                 WriteToScreen(thisProgram);
256                 searching = false;
257             }
258             else if (ProgramExists(name)) {
259                 WriteToScreen(programs[ProgramIndex(name)]);
260                 searching = false;
261             }
262             else if (EndingRecall(name))
263                 searching = false;
264         }
265     }
266 }
267
```

Algorithm design

An algorithm design is logical flow or sequence of instructions
That must accomplish a result

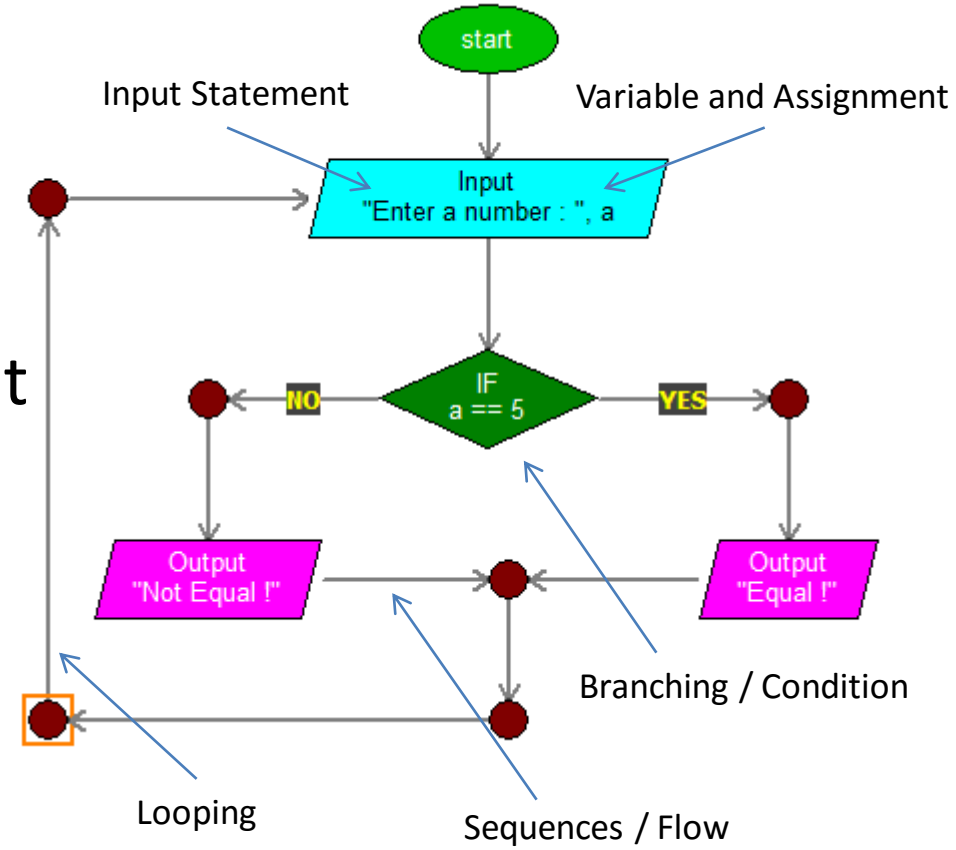
Important points:

1. *Instructions must be well ordered*
2. *Instructions must not be the same*
3. *The process must eventually end or loop*
4. *The actions must be doable*
5. *The algorithm must produce the required result.*

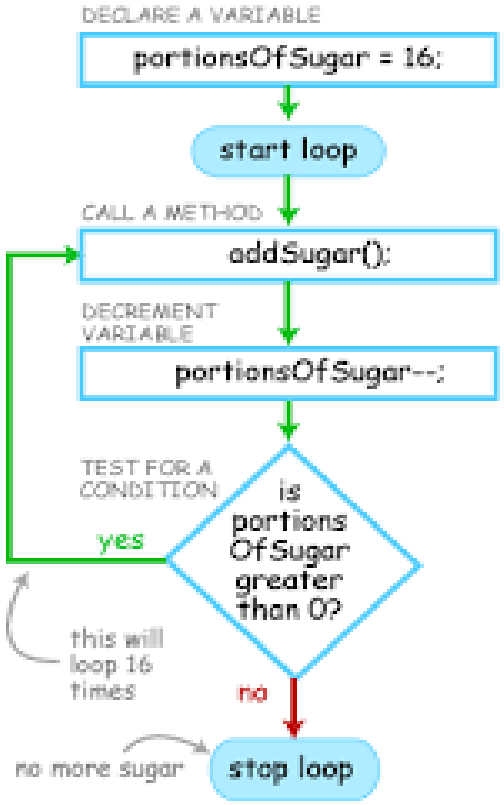
Core elements of an algorithm

Algorithms have some subset of the following critical elements:

1. Input Statement
2. Output Statement
3. Variable and Assignment
4. Branching / Condition
5. Sequences / Flow
6. Looping



Variable



What is Variable ?

Variables are used to store information

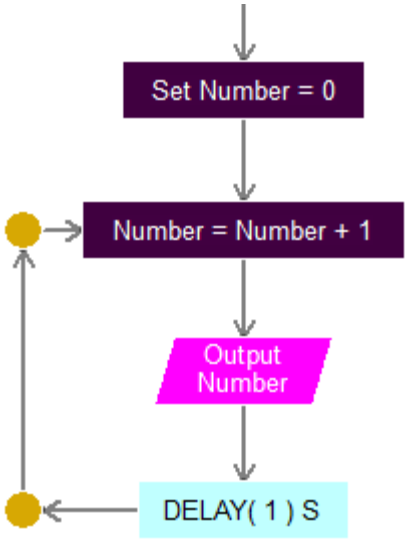
Variables are like containers that hold information. It must be given a unique name and assigned a data. This data can then be used throughout your program by referencing to variable name.

Assigning data to variable

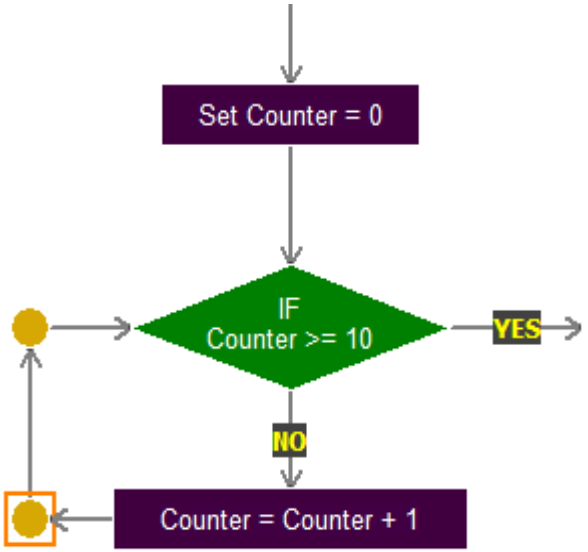
portionOfSugar = 16 **location =** London

Loops and Conditional

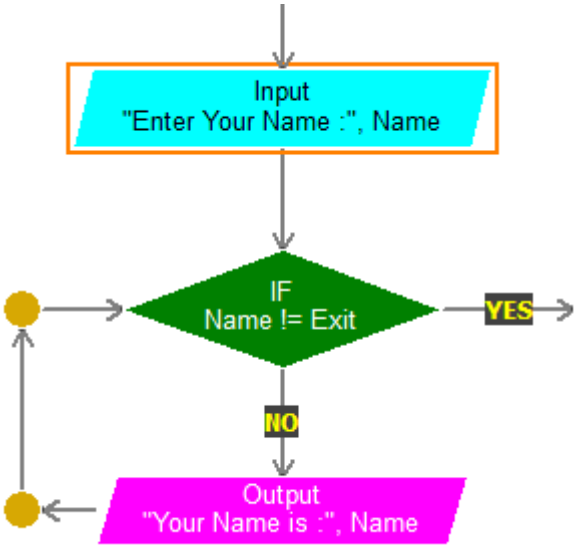
Repeat loop



For loop



While loop



Module #2

Introduction to

FlowLogic 6

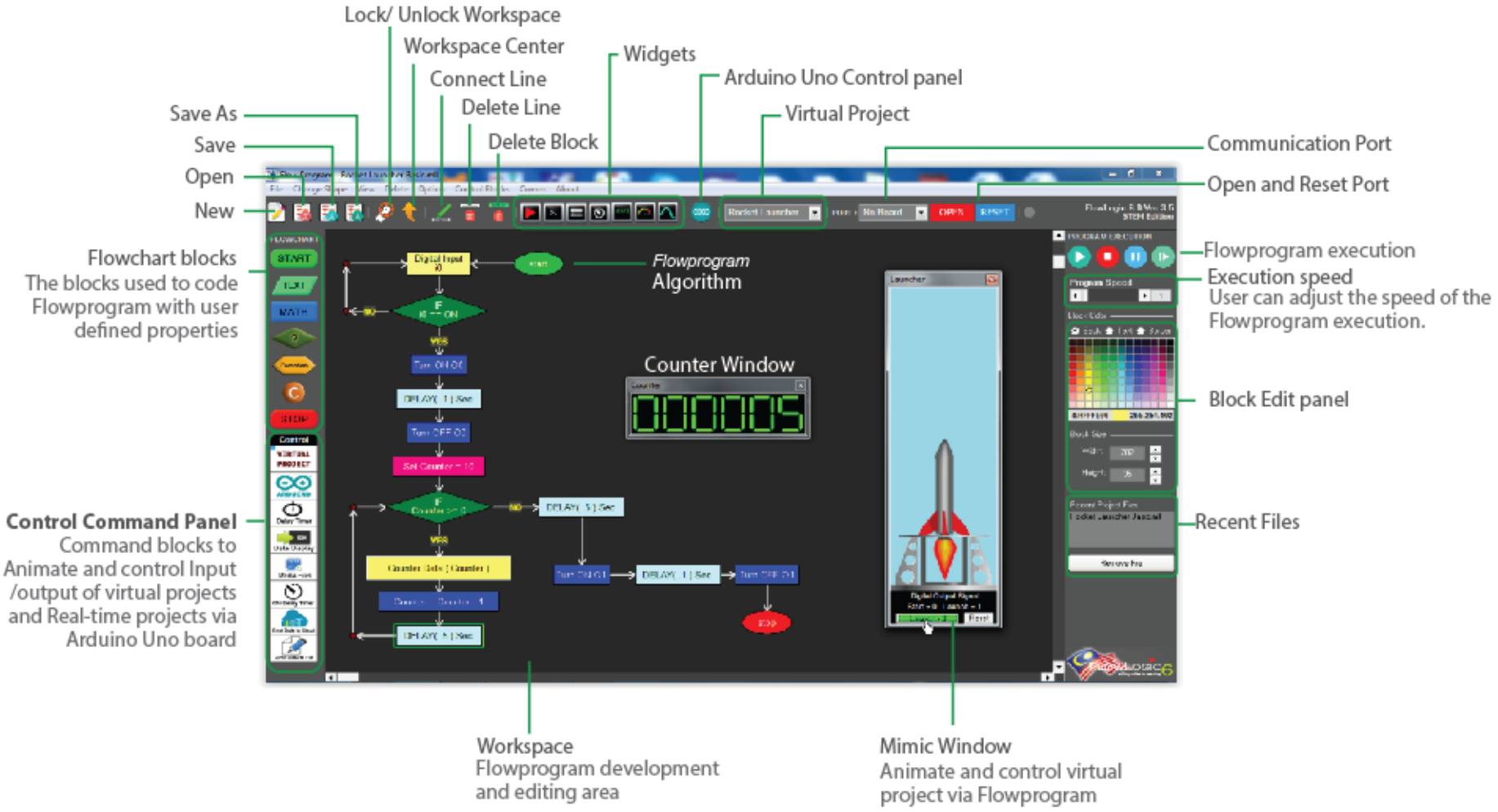
DIY - #1

FlowLogic 6

Download

1. Go to *www.myflowlab.com*
2. *Click Download*
3. *Click Download – FlowLogic 6 Ver 3.6*
4. *Click the Downloaded file to install FlowLogic 6 Version 3.6 into your computer*

FlowLogic 6 Version 3.6

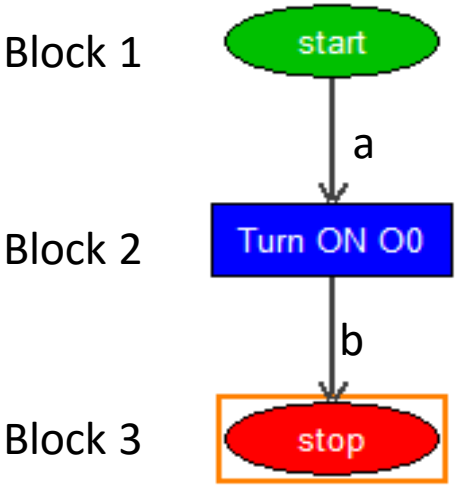


Introduce to student the FlowLogic 6 and Guide them on how to construct a FlowProgram

Editing Command Blocks

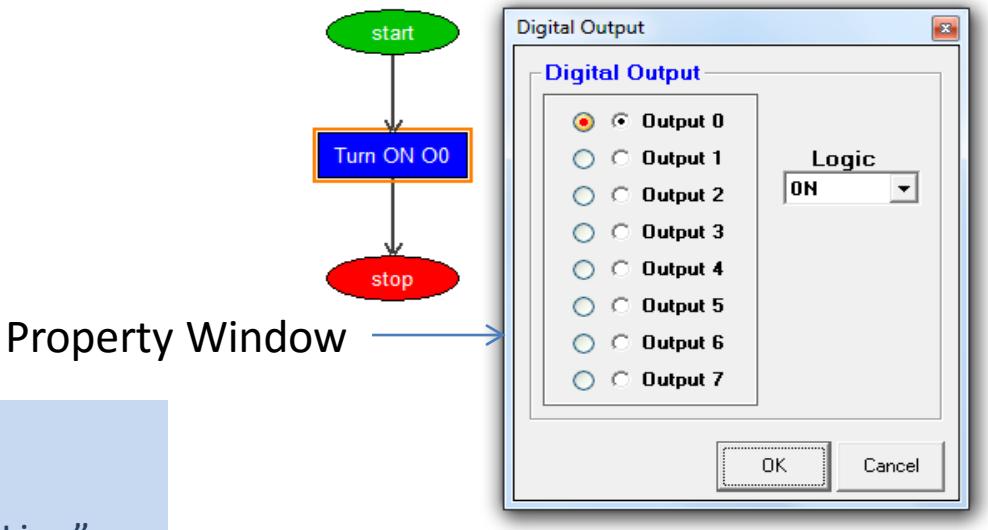
Activity - #1 – Practice Select the Blocks , define the property, Connect Line, Delete Line and Delete block to

To delete Line and Blocks



To delete line “a”, click on Block 1 and then Block 2, while mouse pointer on Block 2, right click and select “Delete Line” option from the pop-up menu.
To delete Blocks, delete all connecting line, right Click on the block and select “Delete block” option from the Pop-menu.

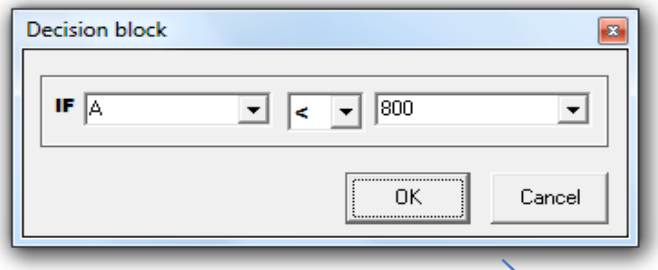
Double click the block to Edit



To edit blocks, double click on the block and make the necessary changes on the pop-up property Windows and click “Ok” when done.

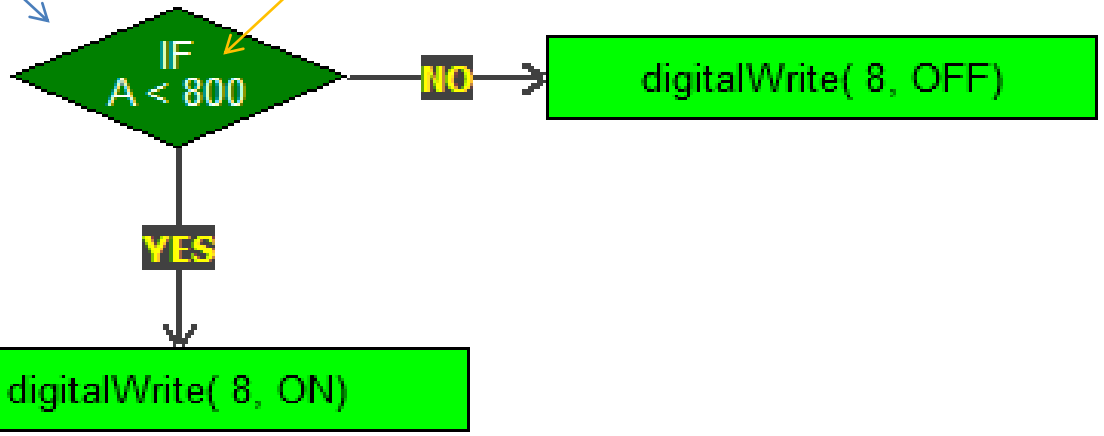
Working with Decision Blocks

Activity - #2 - Practice Select the Blocks , define the property, Connect Line to try out



Decision block
Property Window

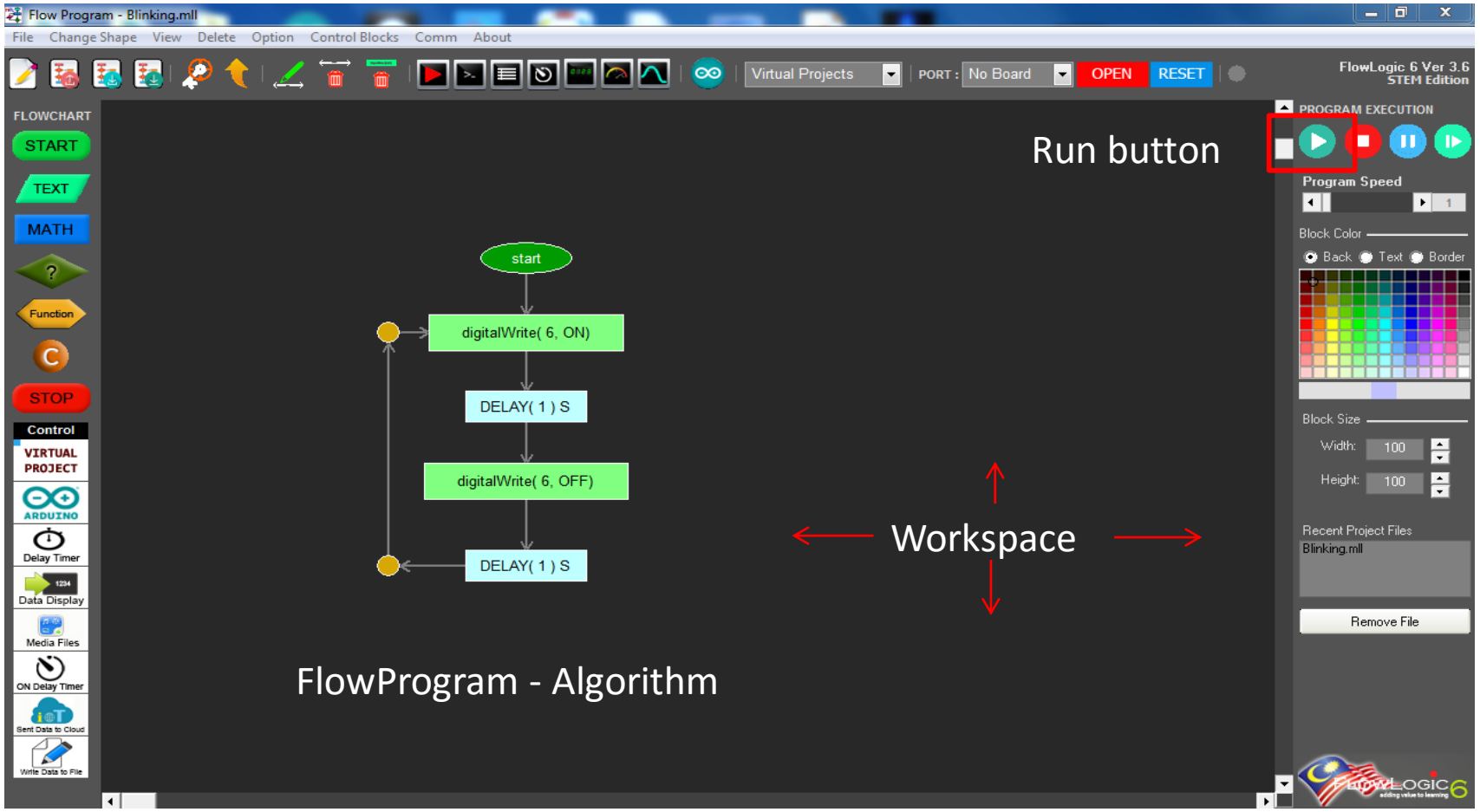
Condition statement as specified
in the property window



Decision block requires two (2) connecting point, the first connection to a block will be “YES” and the next connection will be “NO”.

Connect the lines at your discretion based on the condition statement on the decision block.

Running a FlowPogram



Explain to Students the steps required to run a FlowProgram that is loaded into the Workspace.

Module #3

Developing

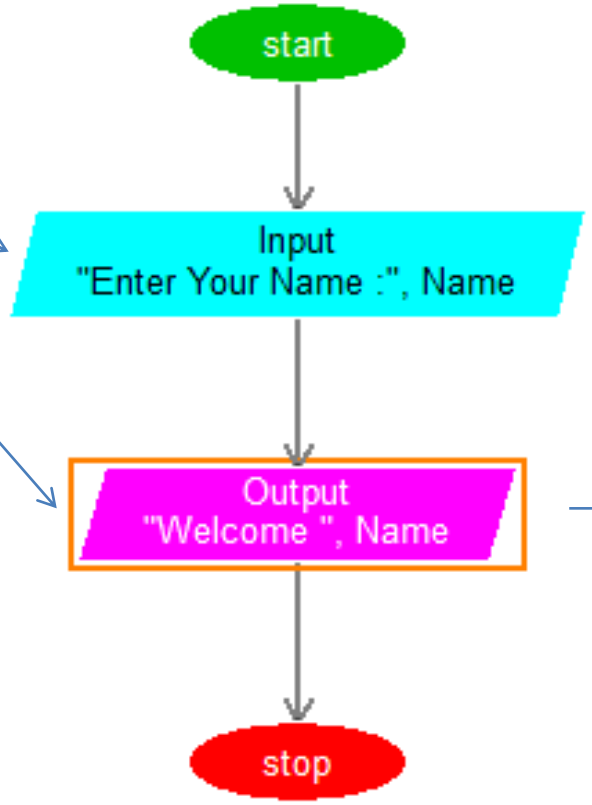
Console Applications

FlowProgram / Algorithm – Activity #3

Console applications

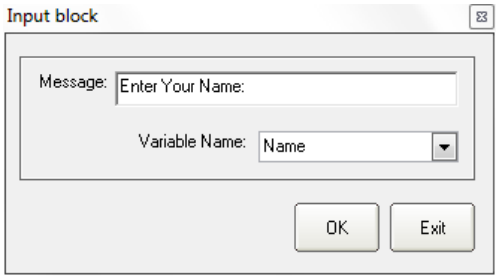
Command Blocks

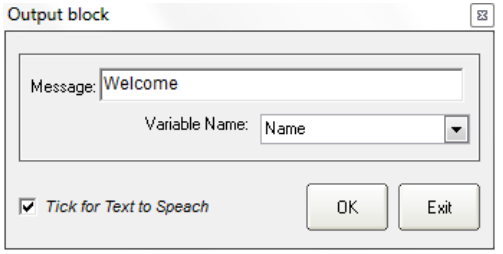
FlowProgram/Algorithm



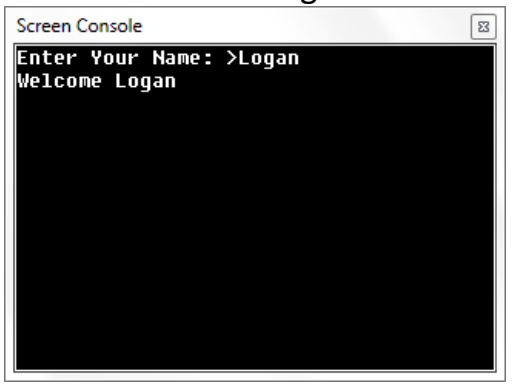
Run the FlowProgram to view the output..

Block's Property Windows



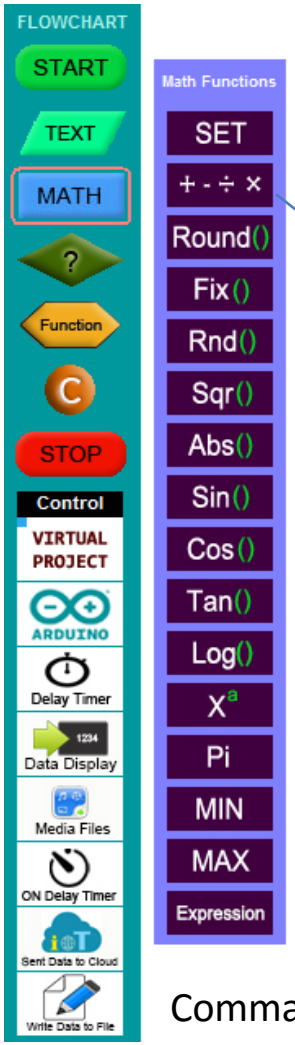


Screen Console widget



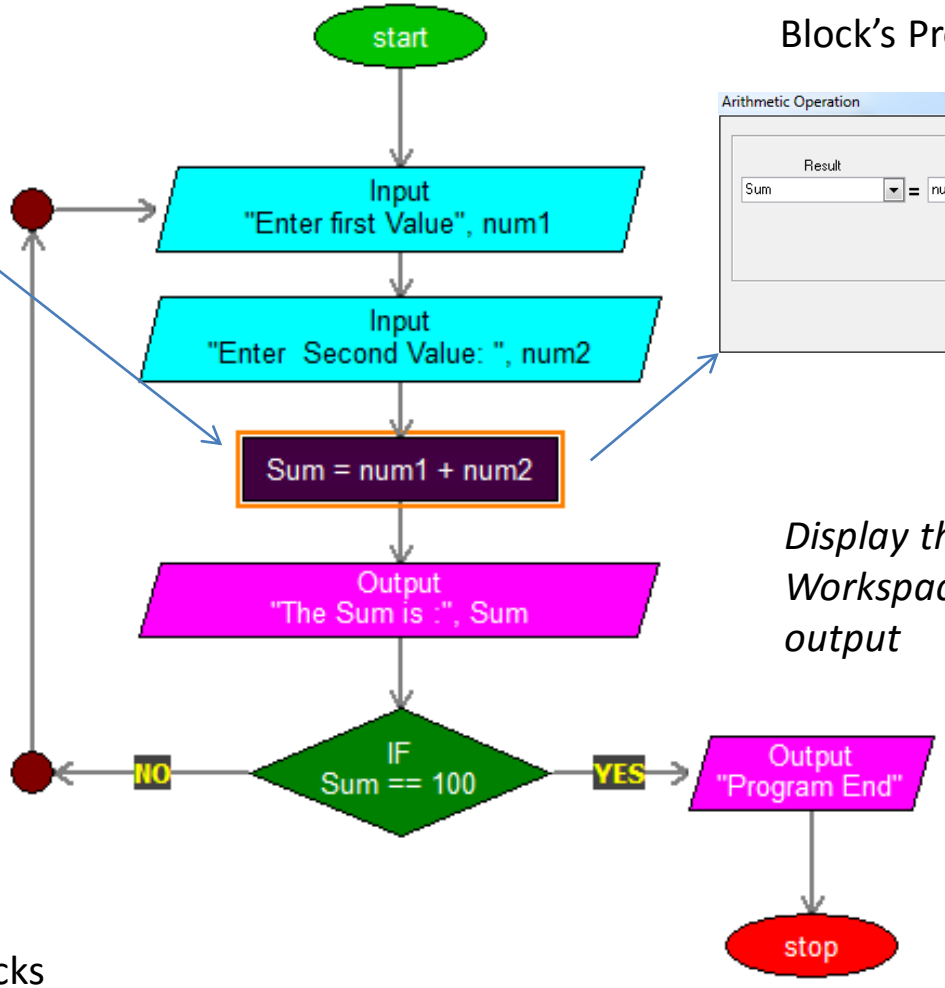
FlowProgram / Algorithm – Activity #4

Console applications

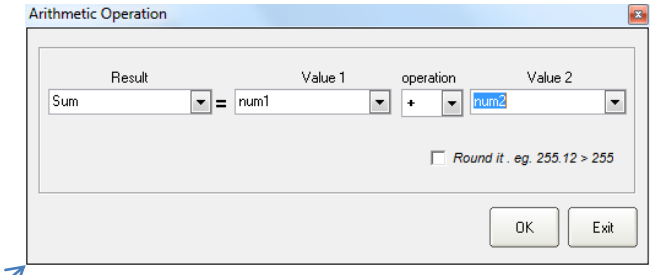


Command Blocks

FlowProgram/Algorithm



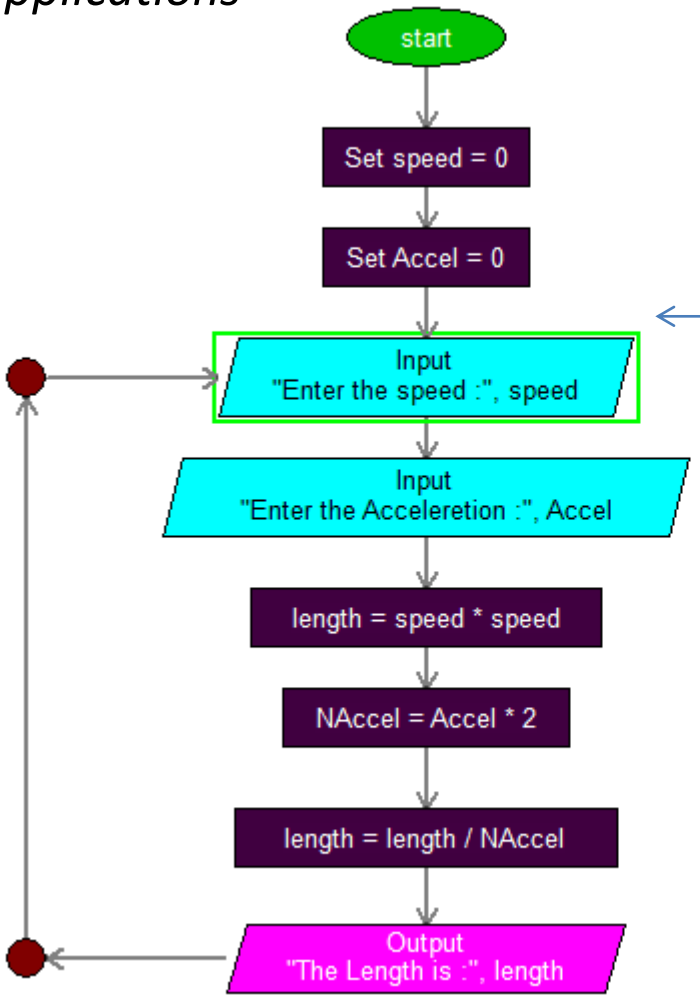
Block's Property Windows



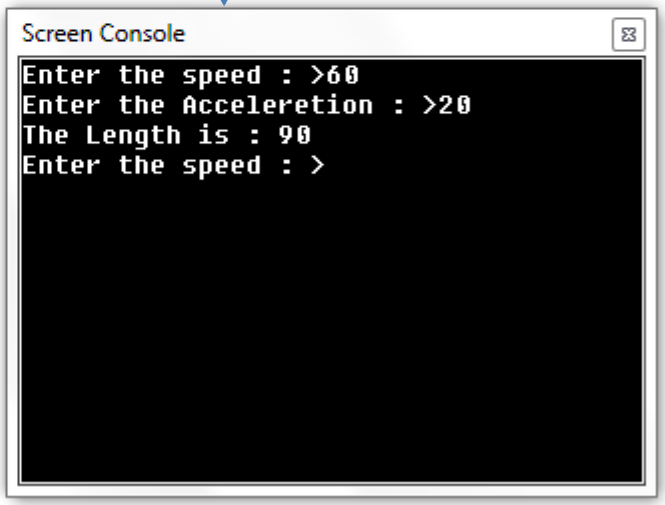
Display the Screen Console on the Workspace To view the algorithm's output

FlowProgram / Algorithm – DIY #2

Console applications



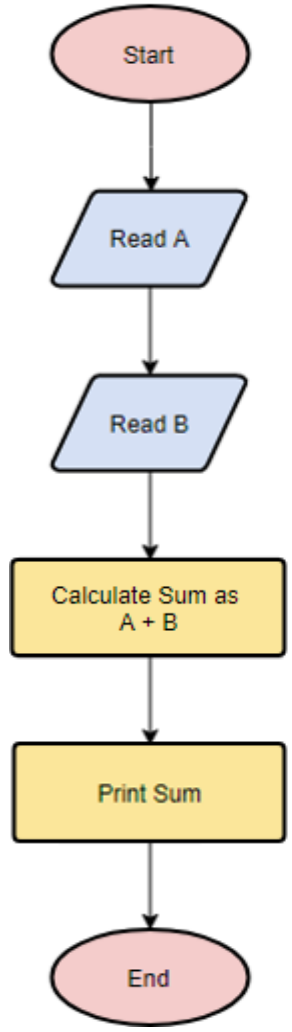
Construct the FlowProgram to produce the shown below



Computational Thinking –DIY #3

Console applications

Construct the FlowProgram to solve the given problem based on the flowchart reference



Find the sum of 529 and 256

Start

A = 529

B = 256

Sum = 529 + 256

Sum = 785

End

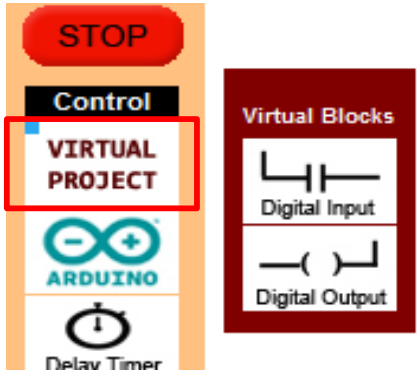
Module #4

Building Algorithm using Virtual Projects

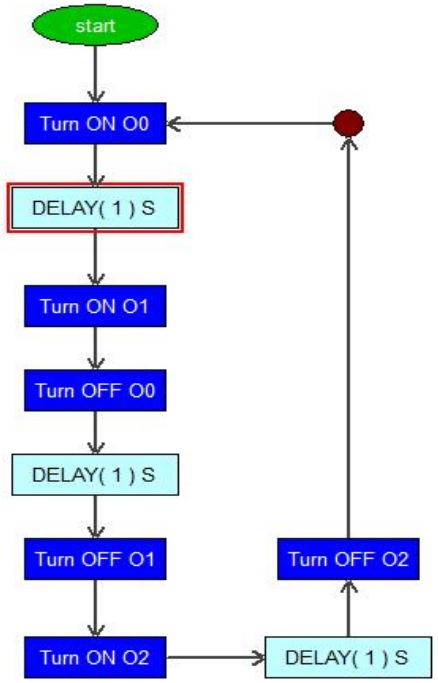
Virtual Projects

Is a on-screen mimics with Pre-assigned control pins and animations that can be programmed by using the virtual command blocks

Command Block

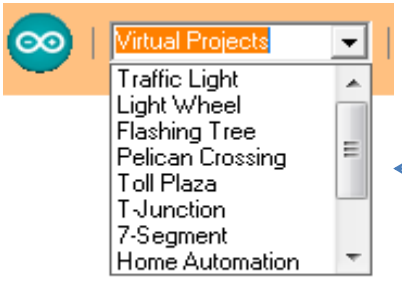


Example Code



Pre-assigned control pins

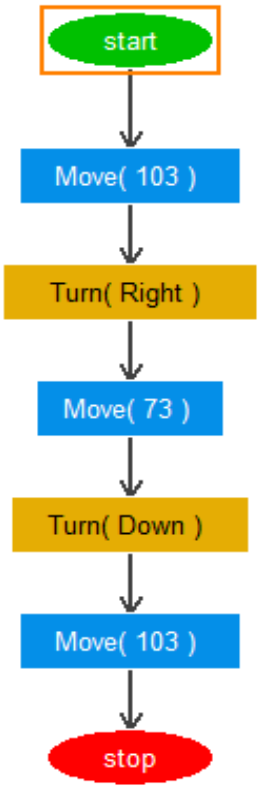
Drop down Project list from Toolbar



When running Virtual project programs, select the appropriate project from the list and place it on the Workspace.

Activity #5 : Robot Maze (Virtual Project)

In this project student will construct FlowProgram using FlowLogic 6 to navigate a Virtual Robot thru a maze.



Step 1: Select Robot Maze from the Virtual Project List

Step 2: Click Load Maze to Load Rmaze1 image from the folder

Step 3: Construct the FlowProgram as shown

Step 4: Click Run icon to execute the FlowProgram

Step 5: Click Reset to place Robot to its origin location

Execute again to test again

Activity : Get the students to load other Maze images and construct the FlowProgram to practice their skill in Flowchart programming and Computational Thinking.

DIY #5 : Robot Maze (Virtual Project)

Step 1: Select Robot Maze from the Virtual Project List

Step 2: Click Load Maze to Load Rmaze2 image from the folder

Step 3: Construct the FlowProgram to navigate the Robot thru the Maze

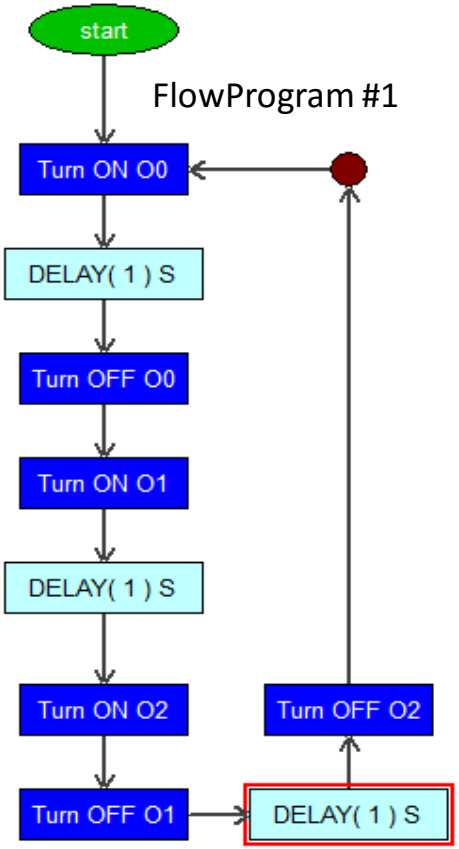
Step 4: Click Run icon to execute the FlowProgram

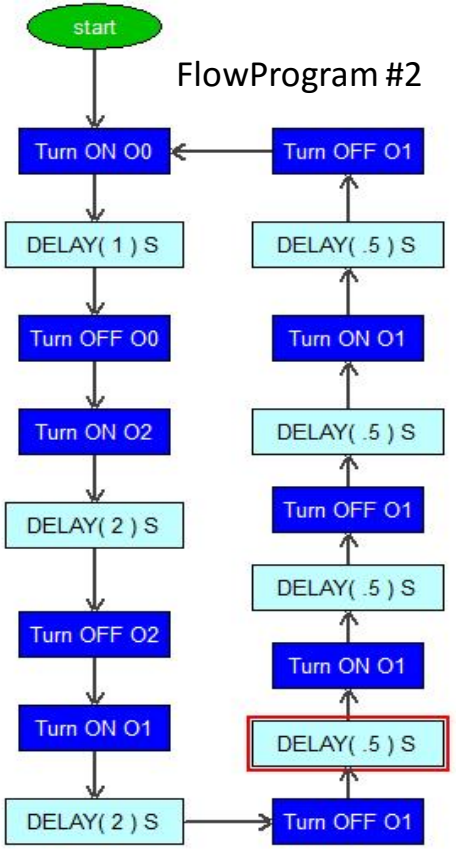
Step 5: Click Reset to place Robot to its origin location

Execute again to test again

Activity #6 : Traffic Light (Virtual Project)

Develop FlowProgram / Algorithm to control a Virtual Traffic Light system.





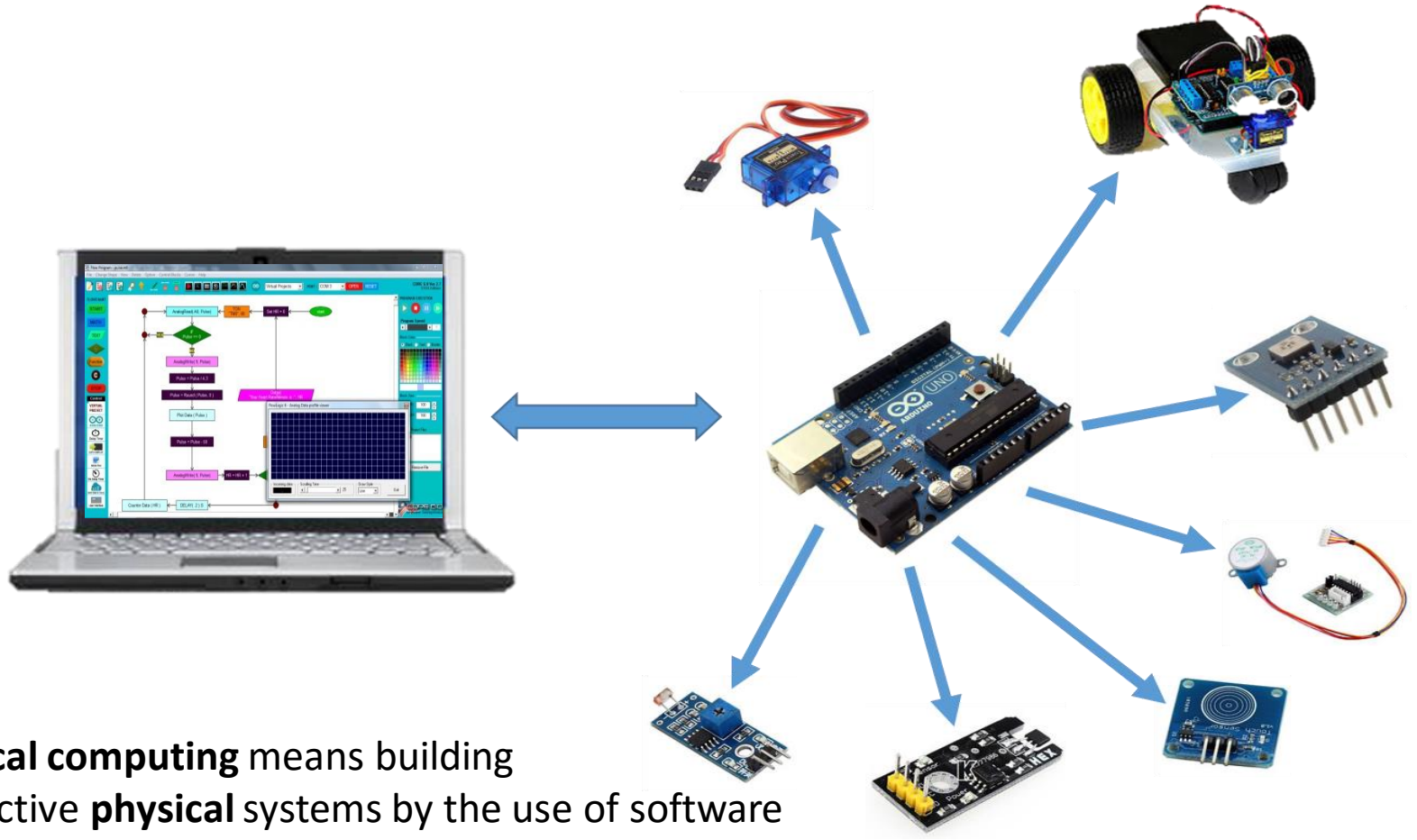
Load Traffic Light Virtual Model from the Virtual project list

Get Student to construct the FlowProgram #1 Using virtual IO pin, Test and Present.

Variant : Get the students to construct a new FlowProgram (FlowProgram #2) to Blink the Yellow Light to enhance their skill in Flowchart programming and Computational Thinking.

Module #5
Building Real-World Applications using Arduino UNO board

What Is Physical Computing



Physical computing means building interactive **physical** systems by the use of software and hardware that can sense and respond to the Real-World

Concepts: INPUT vs. OUTPUT

Inputs is a signal / information going into the board.

Output is any signal coming out from board.

Almost all systems that use physical computing will have some form of output
What are some examples of Outputs?

Concepts: INPUT vs. OUTPUT

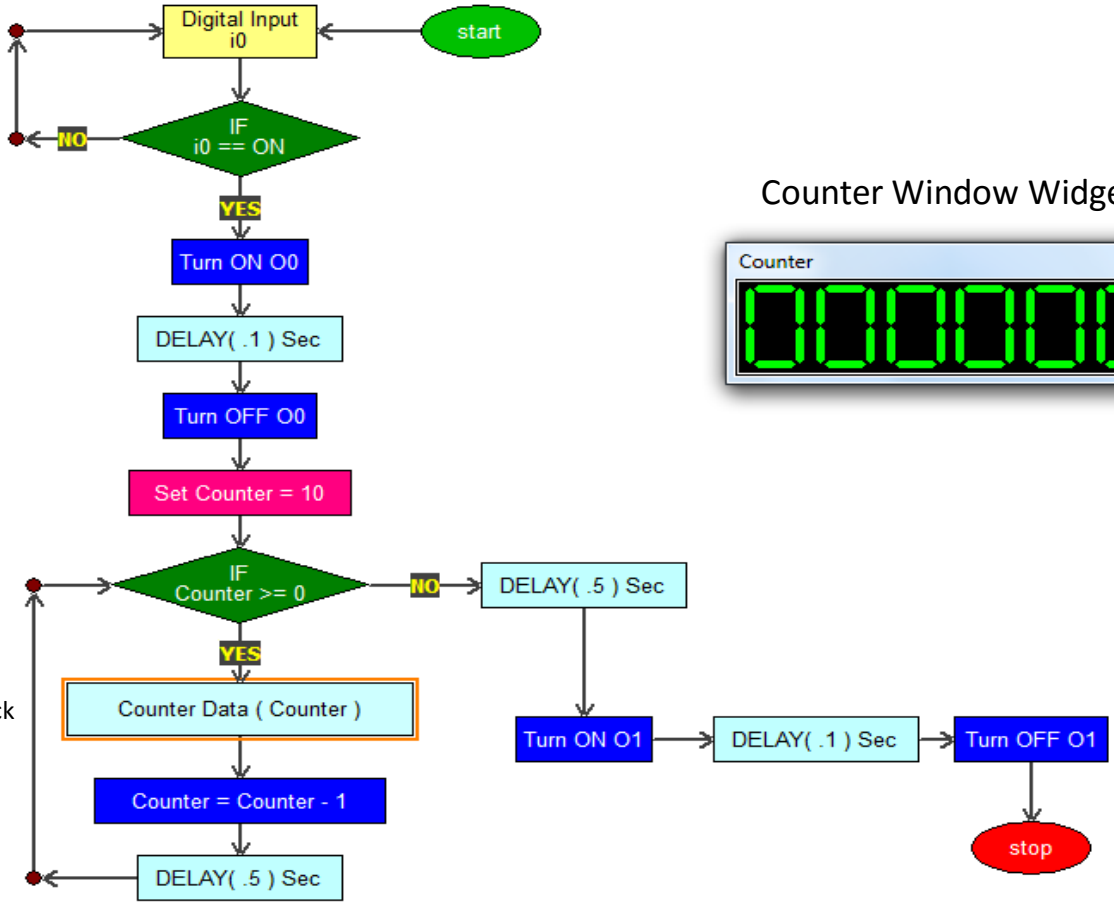
Inputs is a signal / information going into the board.

Output is any signal exiting the board.

Examples: Buttons Switches, Light Sensors, Flex Sensors, Humidity Sensors, Temperature Sensors…	Examples: LEDs, DC motor, servo motor, a piezo buzzer, relay, an RGB LED

Activity #7 : Rocket Launcher (Virtual Project)

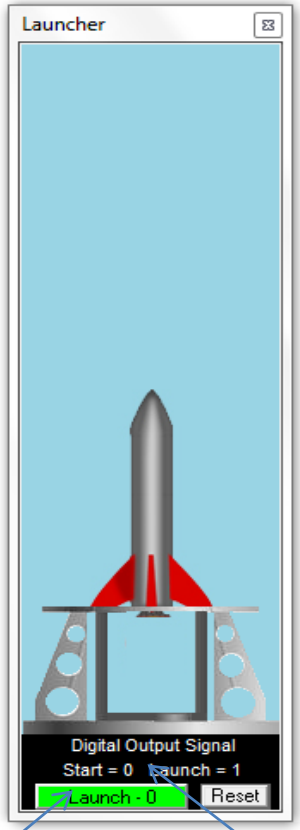
Develop FlowProgram / Algorithm to Launch a Virtual Rocket. (BASIC operation)



Data Display Block

Counter Window Widgets

Select from Virtual project list – Rocket Launcher

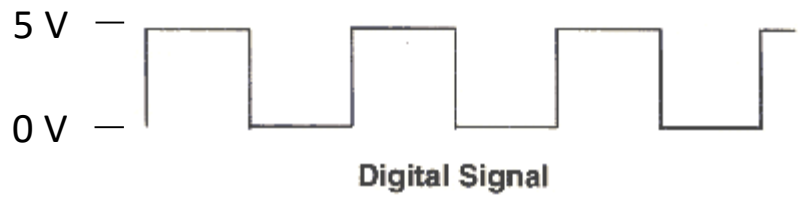


Input Virtual I/Os

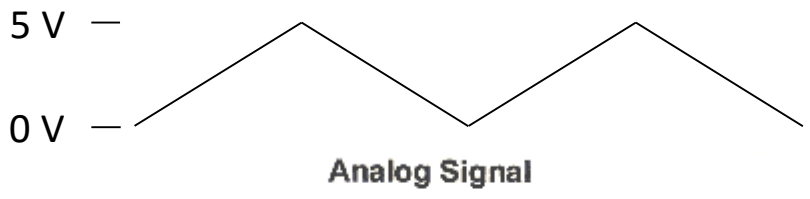
Output Virtual I/Os

Concepts: Analog vs. Digital

- Microcontrollers are **digital** devices – ON or OFF. Also called – discrete.
- **analog** signals are anything that can be a full range of values. What are some examples? More on this later…

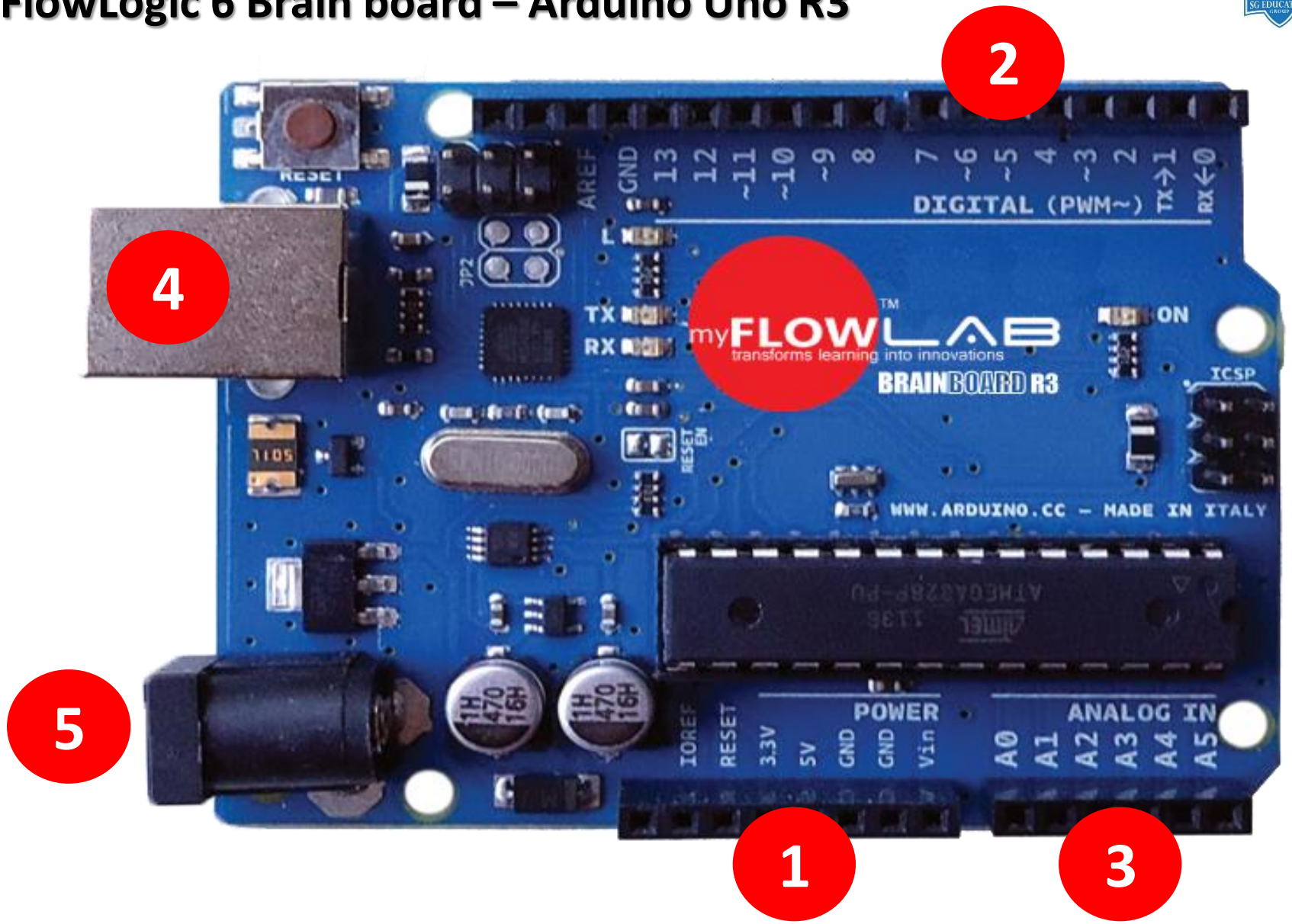


Digital Signal



Analog Signal

FlowLogic 6 Brain board – Arduino Uno R3



Power Source

Positive Terminal

+9V, VCC

+5V , VCC

9 volt battery

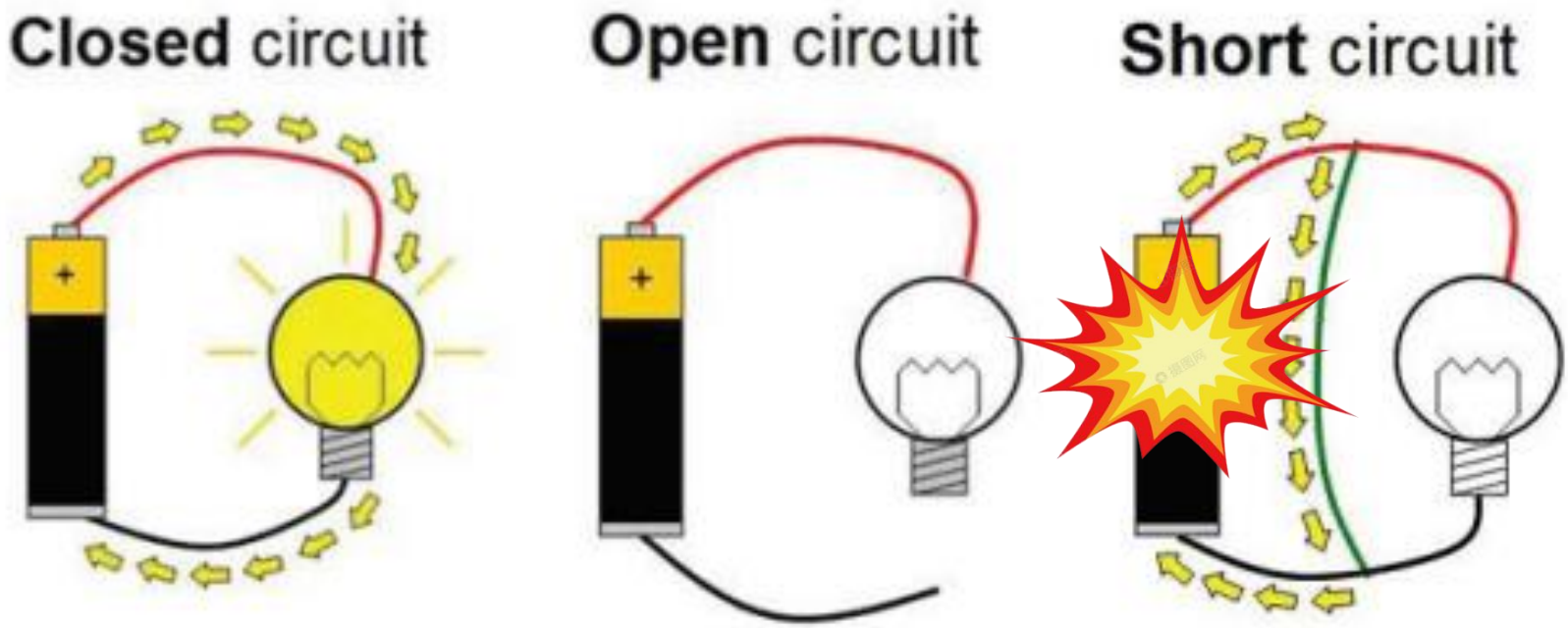
5 volt battery

Output

Negative Terminal

GND

Power Source



Power Source

USB Cable

TO Arduino Board

To Computer

Communication and Power In

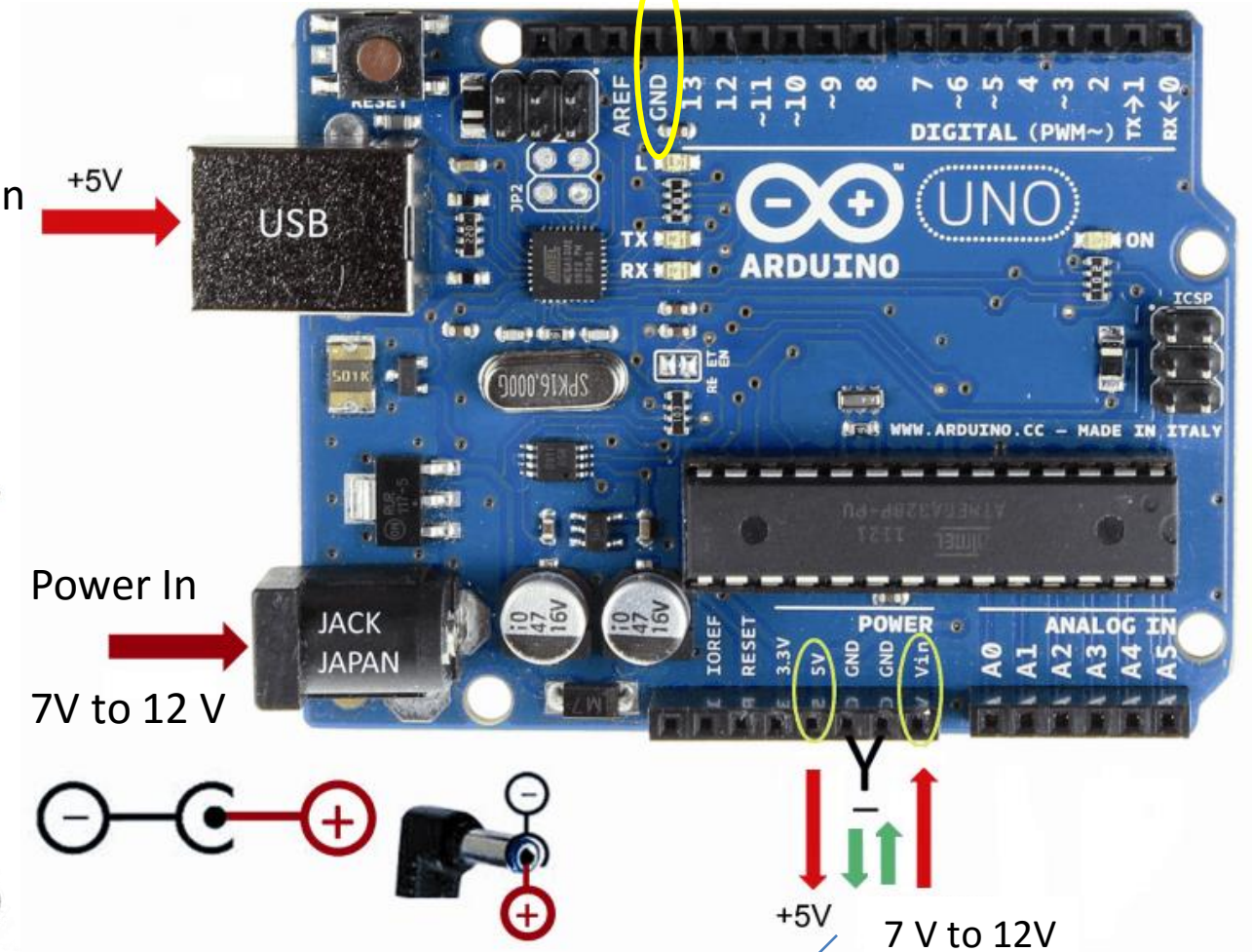
+5V

Battery

Power In

7V to 12 V

Power Adaptor

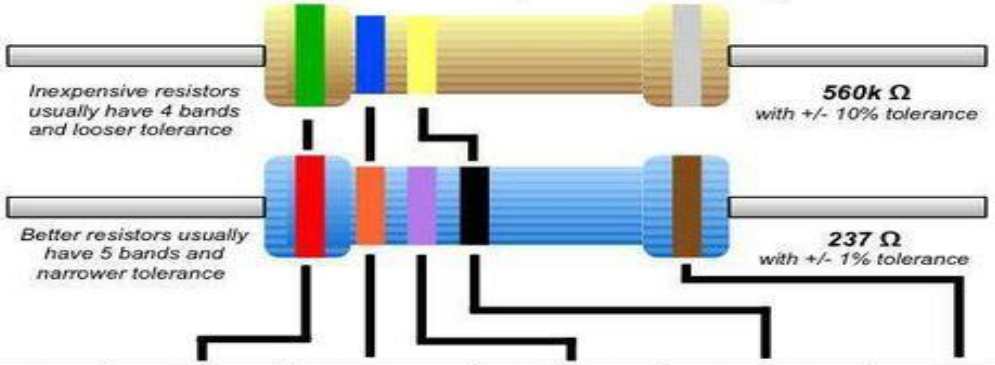


Power out to Breadboard or
Components LED, sensors, Motor….

Resistor

Resistor Identification

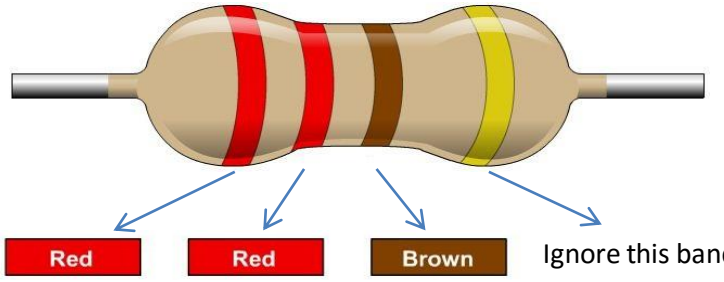
The end with more bands should point left when reading colors.



Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	x 1 Ω	
Brown	1	1	1	x 10 Ω	+/- 1%
Red	2	2	2	x 100 Ω	+/- 2%
Orange	3	3	3	x 1K Ω	
Yellow	4	4	4	x 10K Ω	
Green	5	5	5	x 100K Ω	+/- .5%
Blue	6	6	6	x 1M Ω	+/- .25%
Violet	7	7	7	x 10M Ω	+/- .1%
Grey	8	8	8		+/- .05%
White	9	9	9		
Gold				x .1 Ω	+/- 5%
Silver				x .01 Ω	+/- 10%

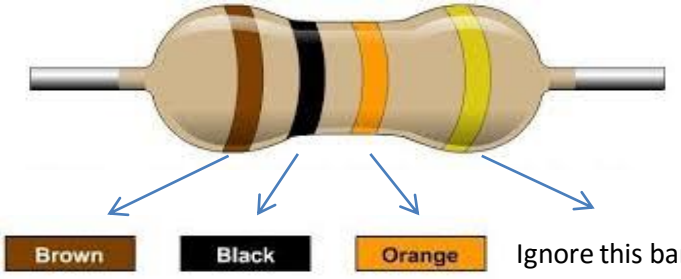
Resistor value calculation

220 Ohm 4 band resistor



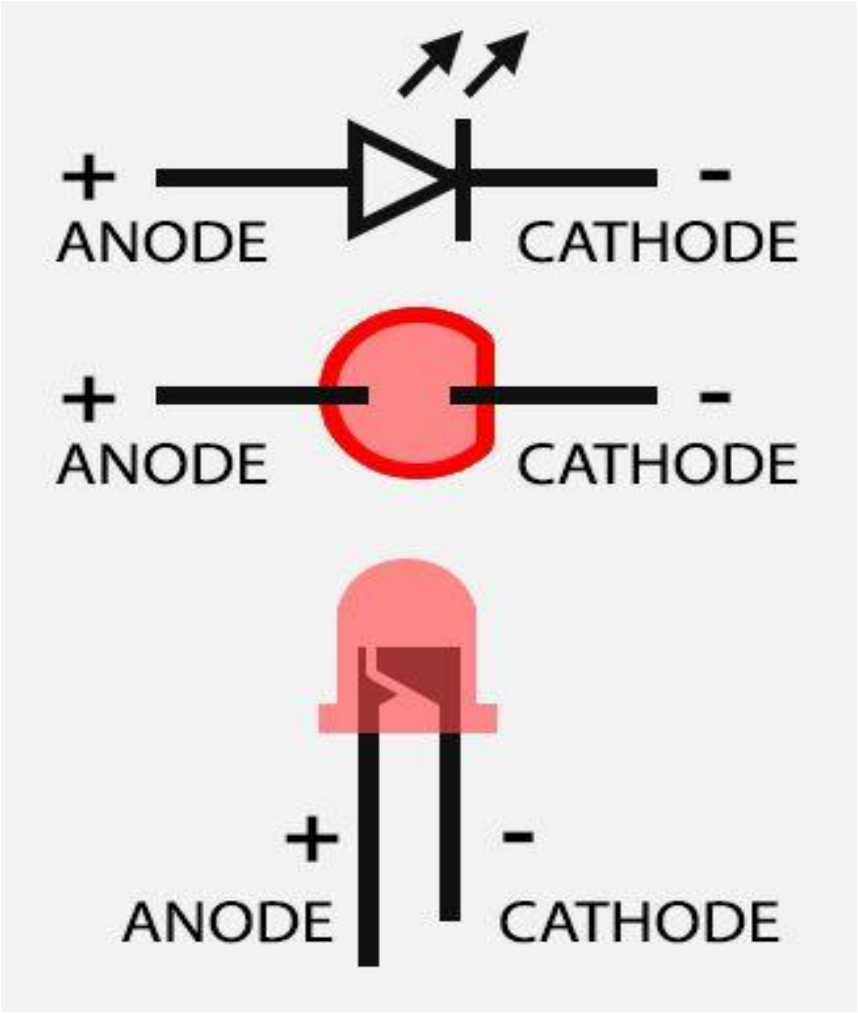
2 2 x 10 = 220 ohm

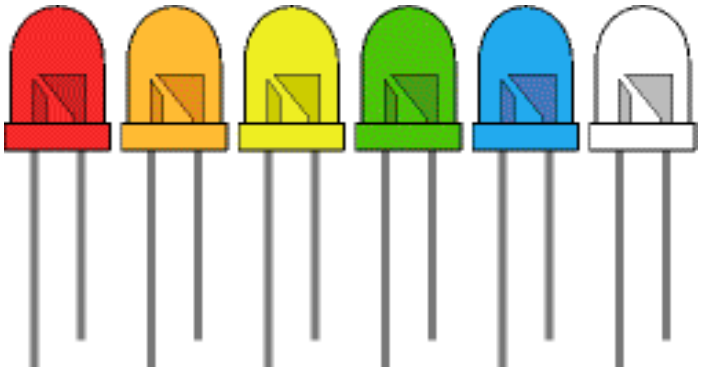
10K Ohm 4 band resistor



1 0 x 1k = 10K ohm

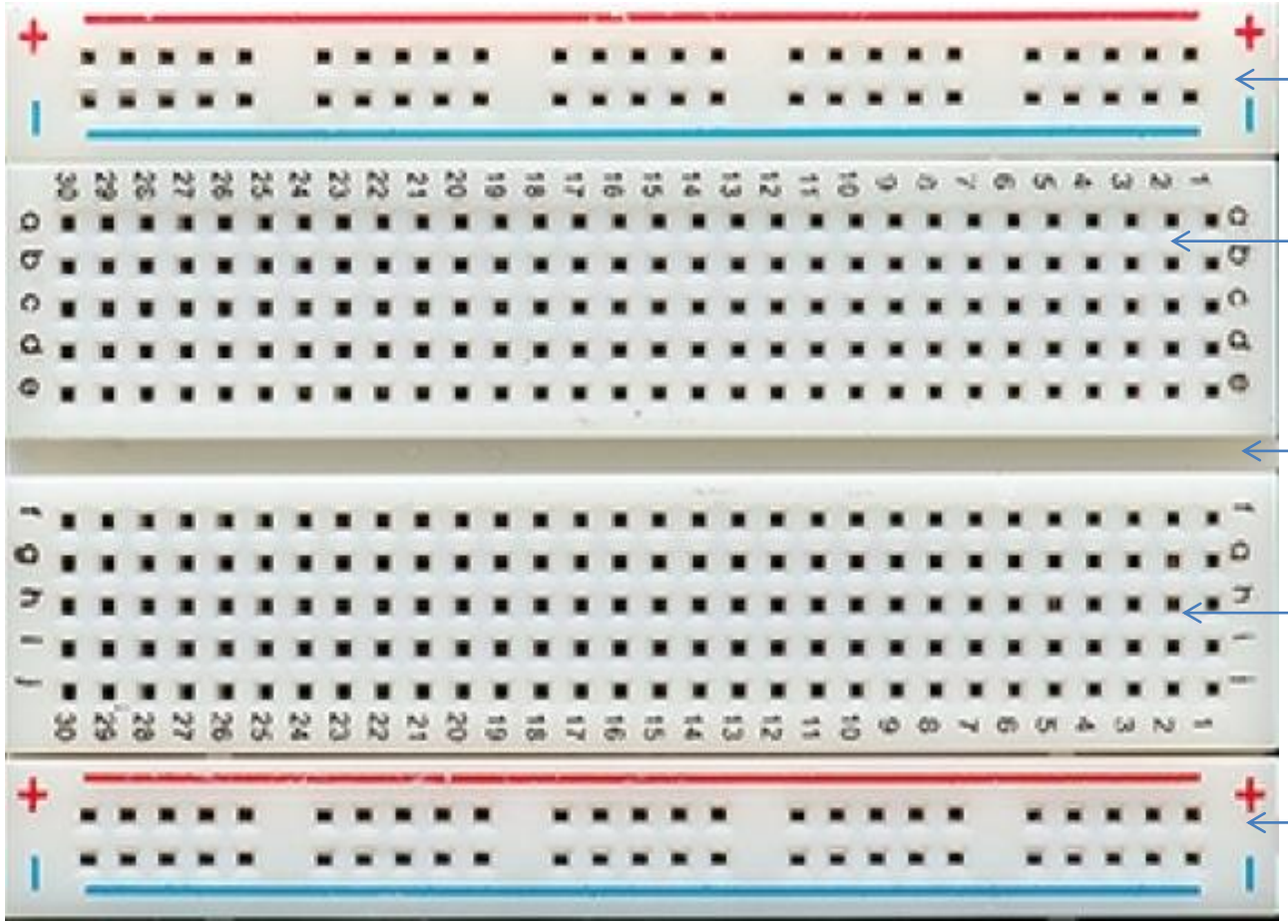
LEDs – Output Devices





Copyright © 2018 myFlowLab Inc.

Solder Less Breadboard – Half + Size



Power Rail / Bus bar

Component Insertion Area

Isolation slot

Component Insertion Area

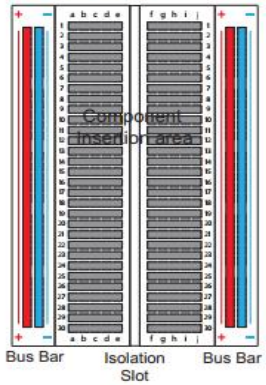
Power Rail / Bus bar

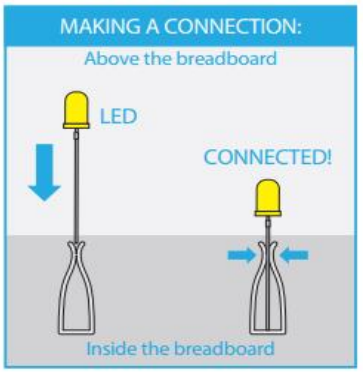
Building Circuit using Solder Less Breadboard

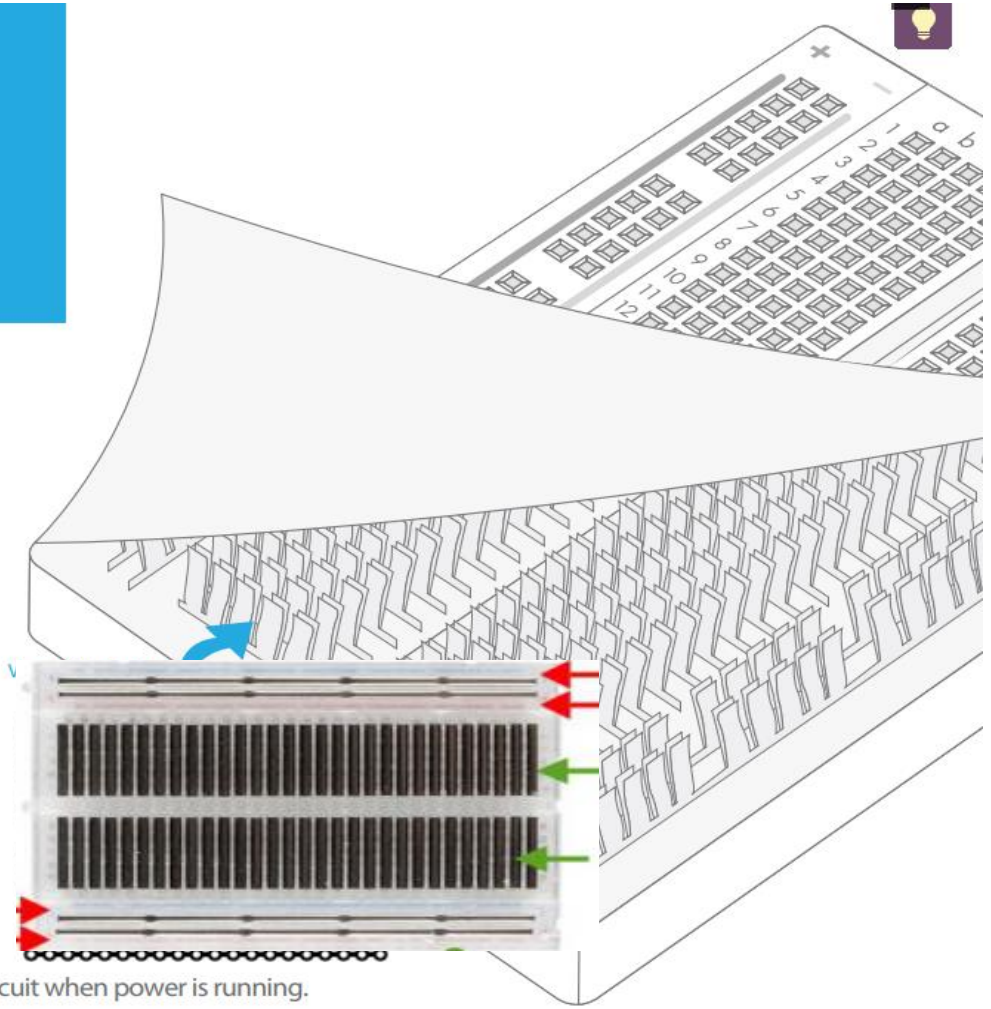
CHAPTER 7

How to use Solder Less Breadboard

HOW'S IT ALL CONNECTED?



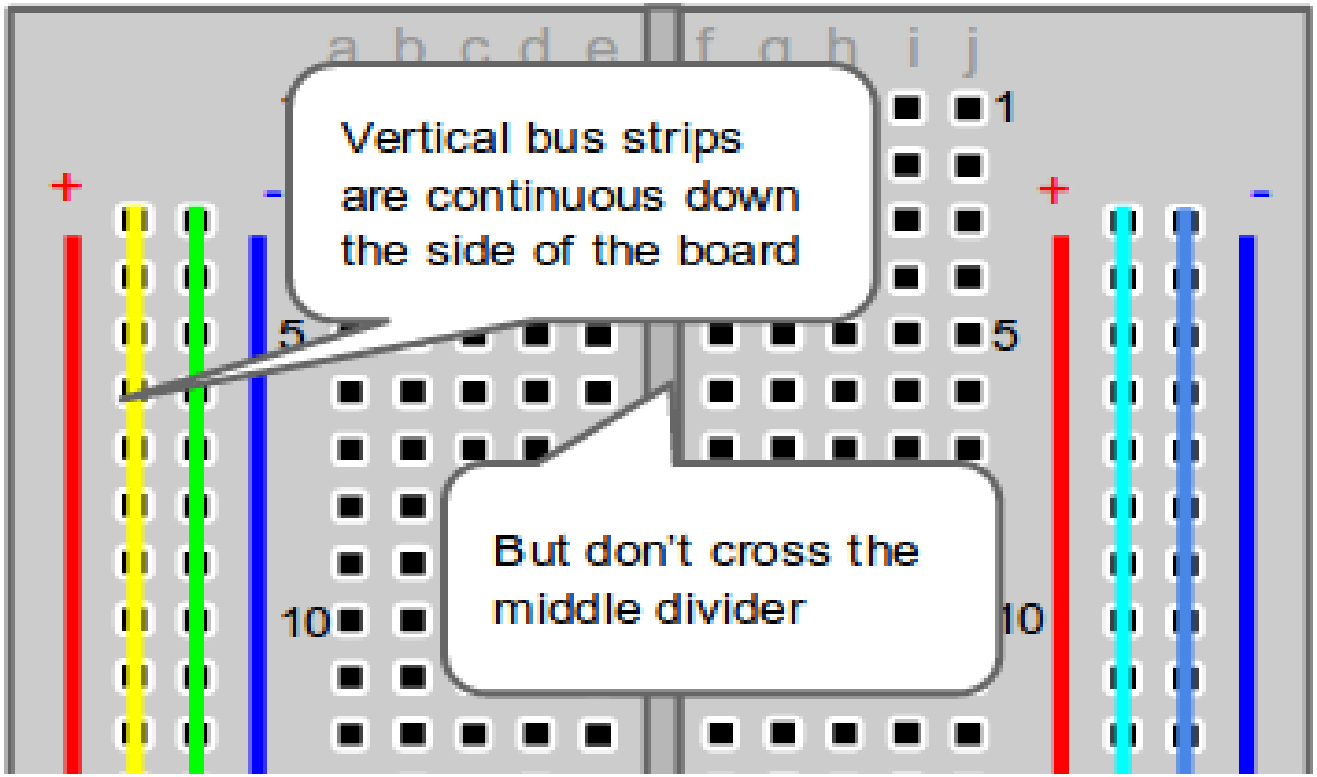




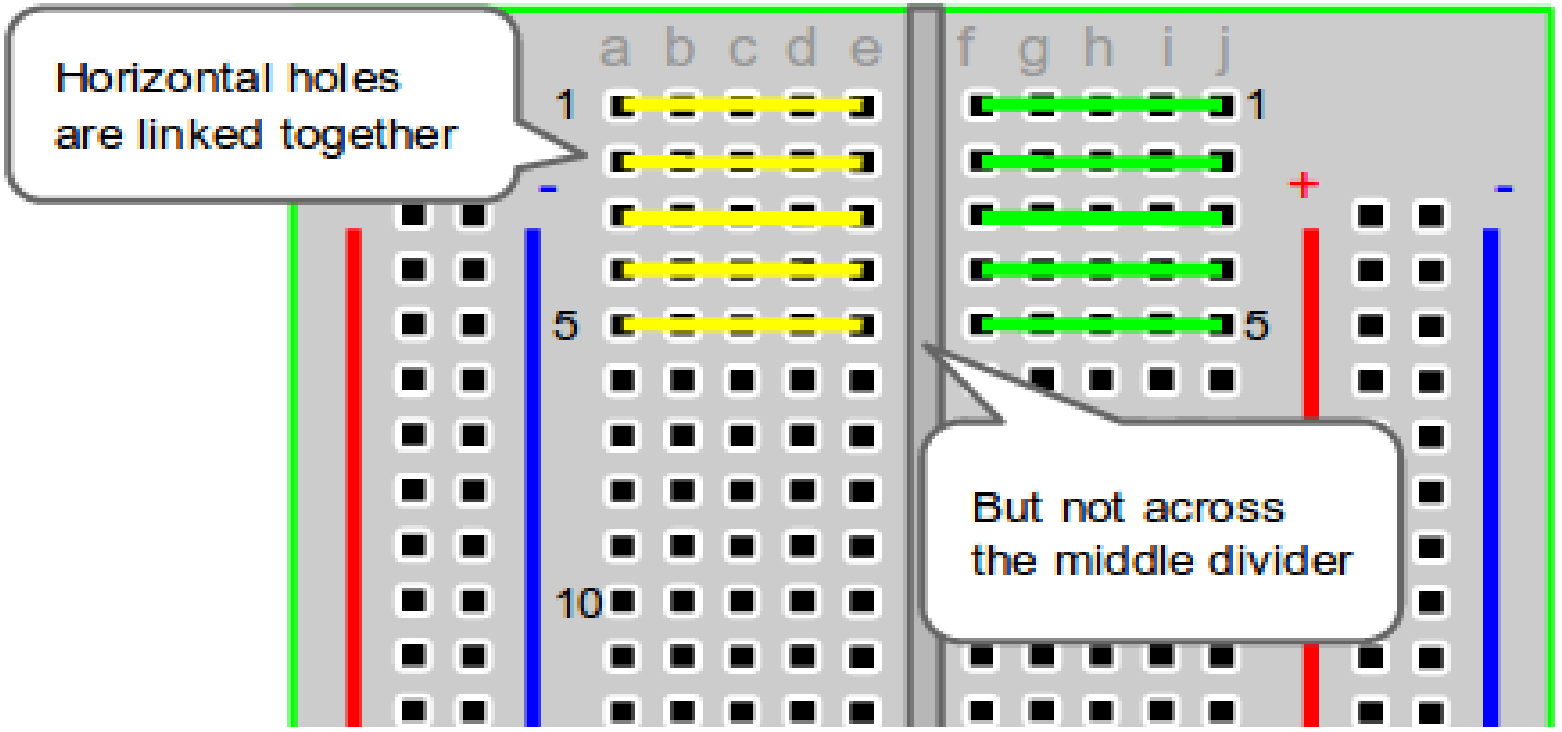
+ **Power**:
  Each + sign runs power anywhere in the vertical column.
− **Ground**:
  Each - sign runs to ground anywhere in the vertical column.

Horizontal Rows:
Each of these rows numbered 1-30 are comprised of five horizontal s
ockets. Components placed in the same row will be connected in a circuit when power is running.

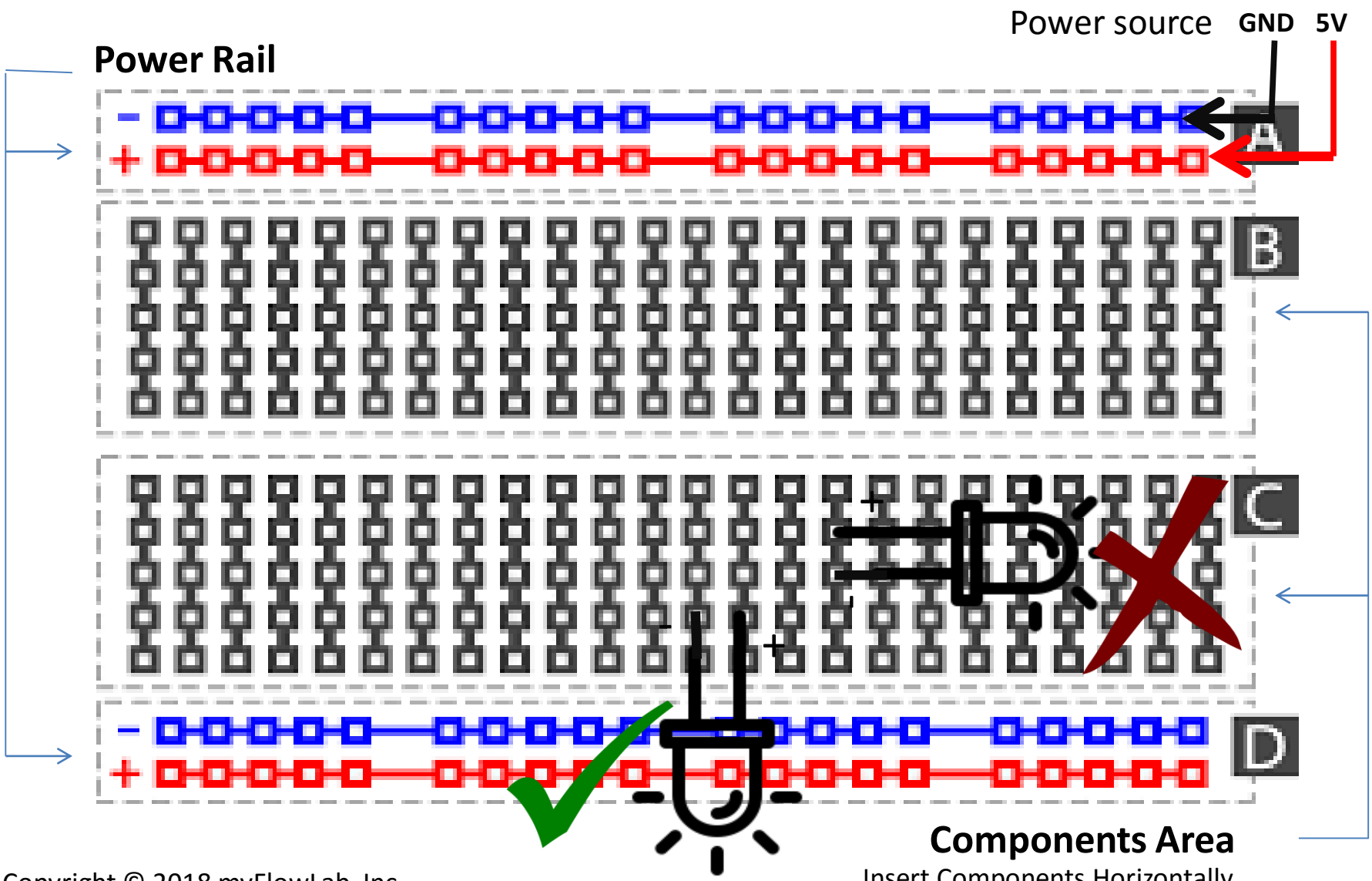
Power Rails / Bus Bar



Components Terminal



Breadboard Connection



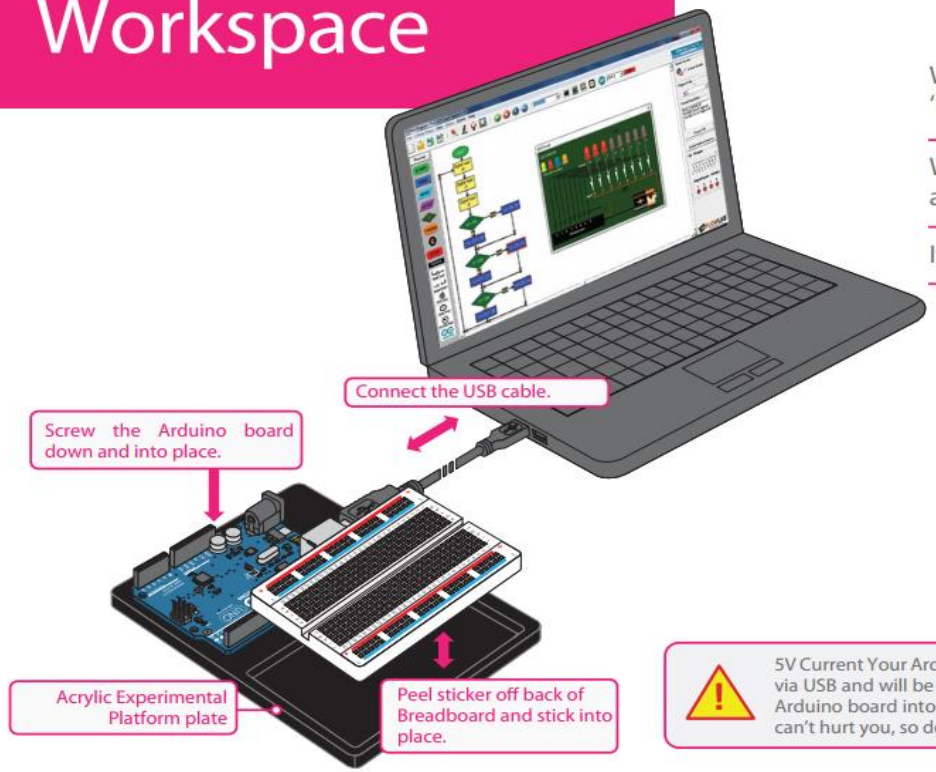
Copyright © 2018 myFlowLab Inc.

Preparing your Workspace for prototyping

CHAPTER 6

Preparing your Workspace

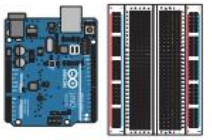
You obviously will require a Table and a PC (Windows XP, 7 or 8 Operating System installed Desktop or Laptop) and myFLOWLAB™ Prototyping board as shown below. You also must ensure you have installed our Flowlogic 5 application software in your PC. The diagram below illustrates how you could setup your workbench for experiments.



When you plug in the Arduino board to your PC, the 'ON' and 'L' led on the Arduino board should light up.

When there is a communication between **FLOWLOGIC 5** and Arduino the 'TX' and 'RX' led must light up.

If not, check the Arduino board, USB cable and USB ports.

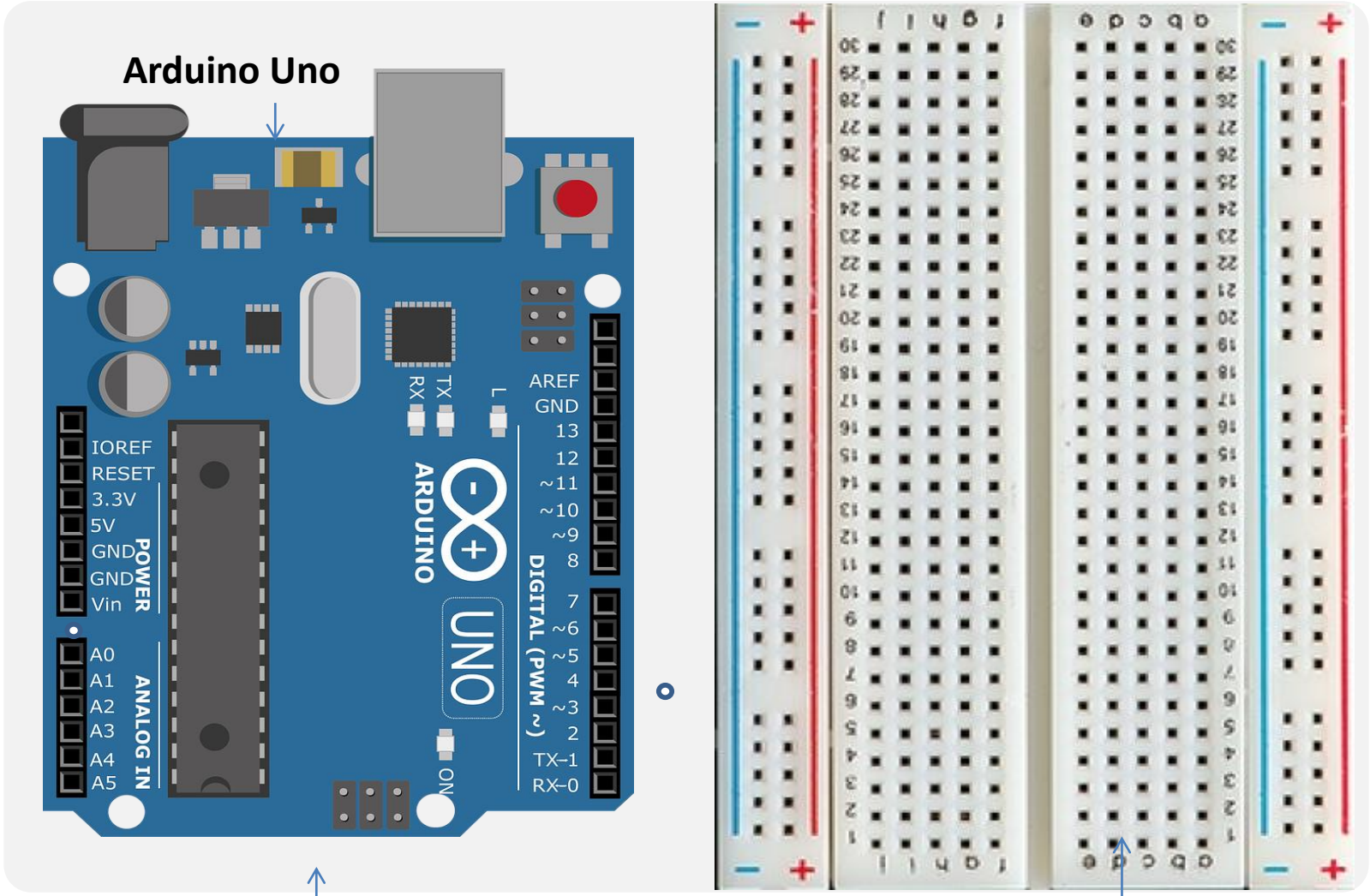


TIPS

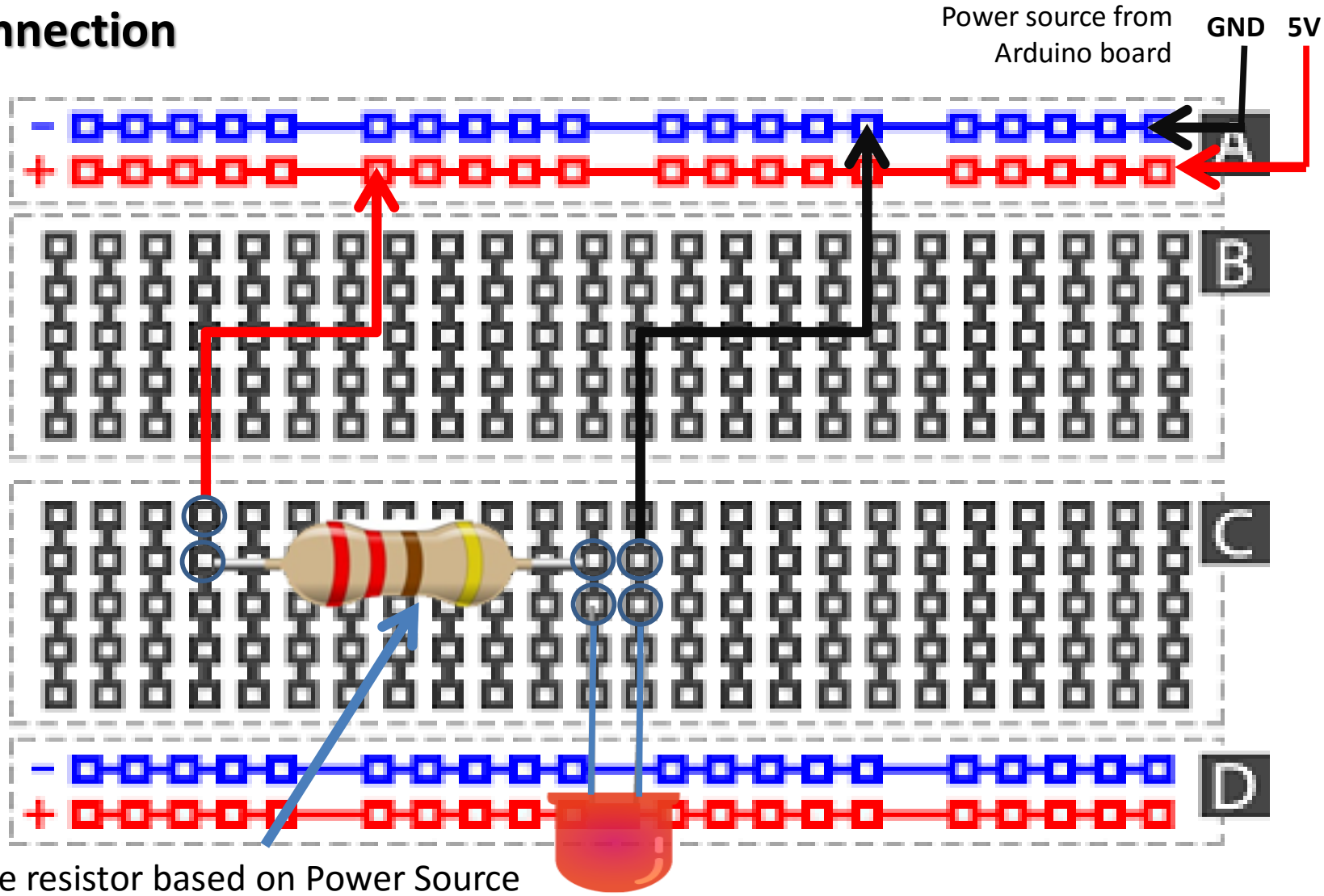
Make sure the text on the Arduino board and Breadboard is facing up so you can read them.

5V Current Your Arduino runs on five (5) volts. This is the power that will be supplied from your computer via USB and will be the driving force behind any components you use in your circuits. By plugging your Arduino board into your computer, you are supplying it with just the right voltage it needs to work. 5V can't hurt you, so don't be afraid to touch anything in your circuit.

Activity # 8 – Assemble the Prototyping Workspace as per Layout shown.



Activity # 9 - Circuit on Breadboard connection



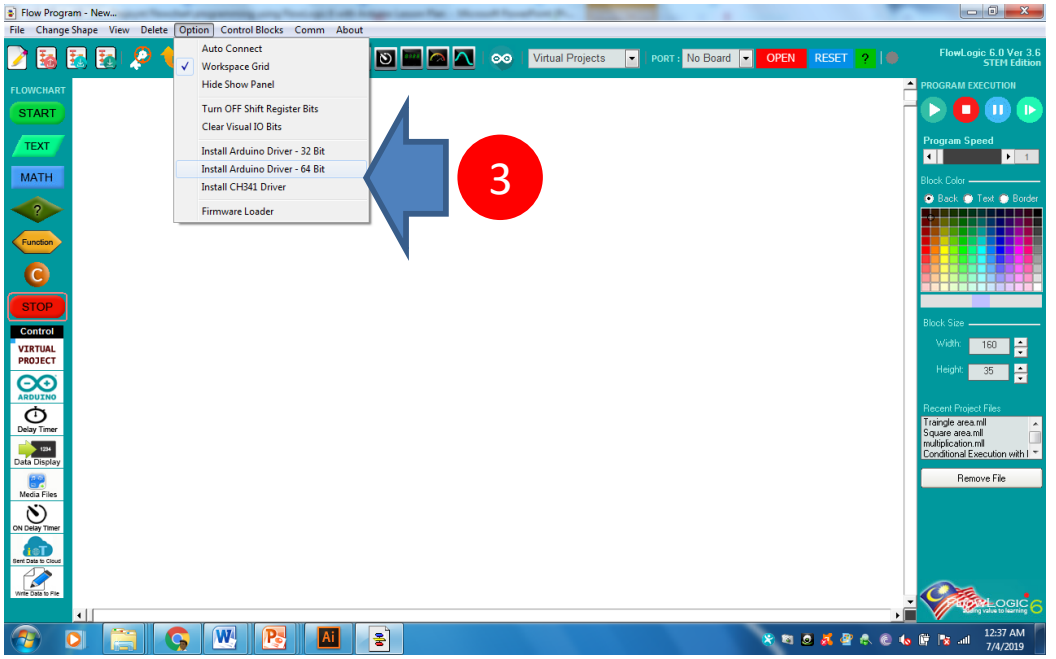
Change resistor based on Power Source

220 Ohm for 3V, 470 Ohm for 5V, 1k for 9V

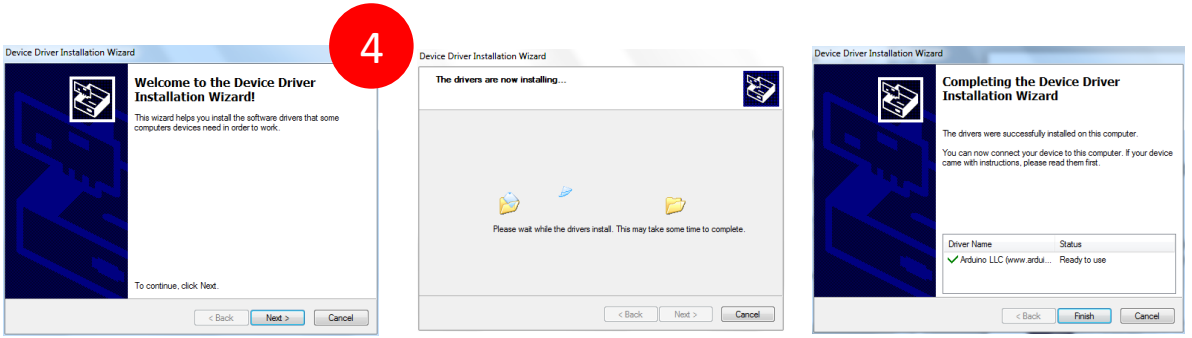
DIY # 2

Arduino USB Driver installation

1. Launch FlowLogic 6 Version 3.6
2. *From the menu, click option*
3. *Select Install Arduino USB Driver*
   Select either 32 Bit or 64 Bit
4. *The USB Driver Installation window*
   Should appear as shown below, if
   NOT, Exit FlowLogic 6 and Run it as
   Administrator.



Right click on FlowLogic 6 desktop Icon and Select "Run as Administrator from the pop-menu



Module #6

C/C++

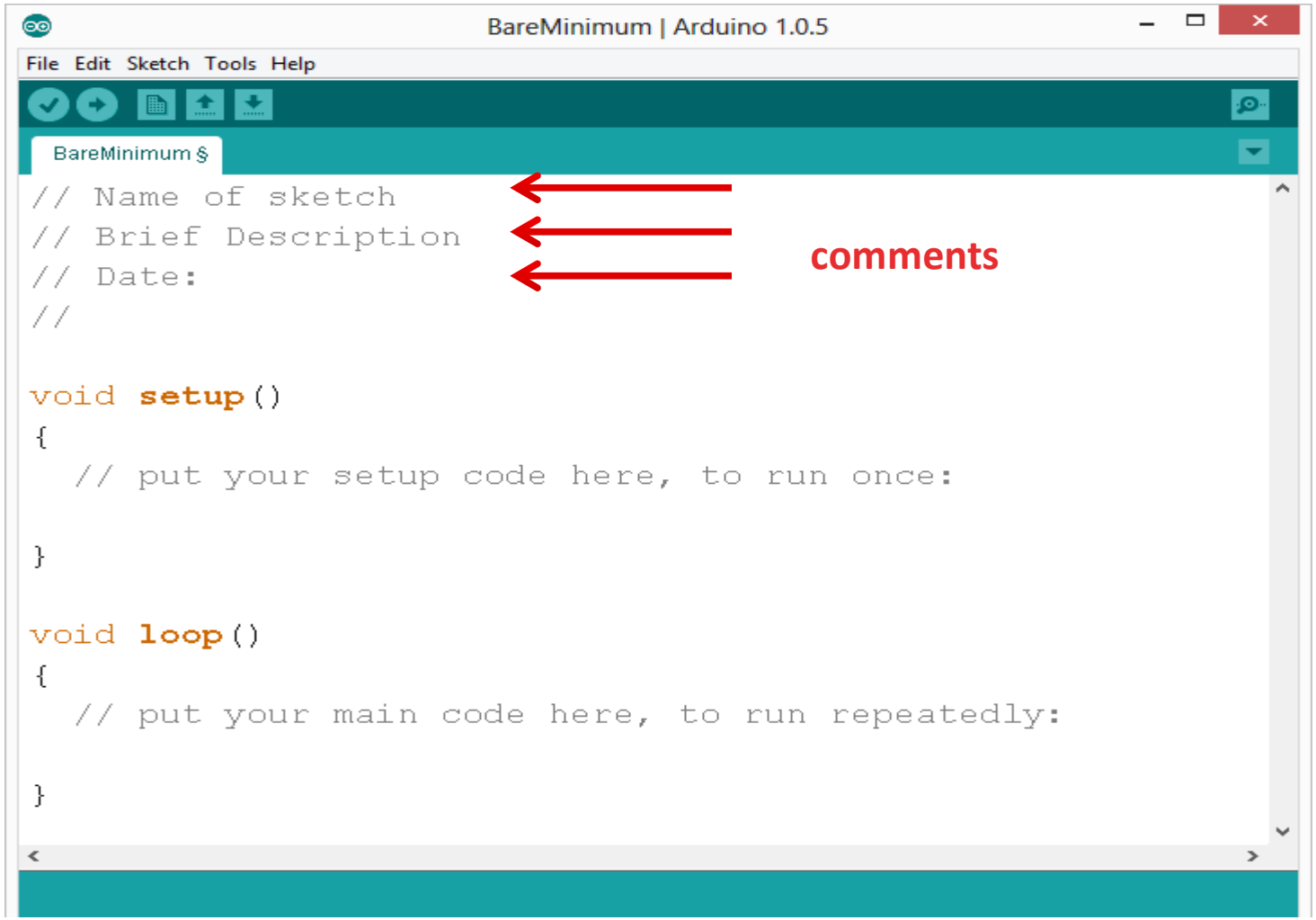
Programming IDE

DIY # 7

Download the Arduino IDE

https://www.arduino.cc/en/main/software

Arduino Integrated Development Environment (IDE)



Module #7

Getting started with Arduino Programming

Bare minimum code

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

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

Bare minimum code

- setup : It is called only when the Arduino is powered on or reset. It is used to initialize variables and pin modes

- loop : The loop functions runs continuously till the device is powered off. The main logic of the code goes here. Similar to while (1) for micro-controller programming.

Important functions

- **Serial.println(value);**
  - Prints the value to the Serial Monitor on your computer
- **pinMode(pin, mode);**
  - Configures a digital pin to read (input) or write (output) a digital value
- **digitalRead(pin);**
  - Reads a digital value (HIGH or LOW) on a pin set for input
- **digitalWrite(pin, value);**
  - Writes the digital value (HIGH or LOW) to a pin set for output

PinMode

- A pin on arduino can be set as input or output by using pinMode function.

- pinMode(13, OUTPUT); // sets pin 13 as output pin

- pinMode(13, INPUT); // sets pin 13 as input pin

Reading/writing digital values

- digitalWrite(6, LOW); // Makes the output voltage on pin 6 , 0V
- digitalWrite(6, HIGH); // Makes the output voltage on pin 6 , 5V
- int buttonState = digitalRead(2); // reads the value of pin 2 in buttonState

DELAY FUNCTION

example:

```
delay(int milliseconds)
//creates a delay in ms
delayMicroseconds(int microseconds)
//creates a delay in µs

delay(1000); //one second delay
delayMicroseconds(10); //10 µs delay
```

Infinite Loop

```
void loop()
{
    delay(20000);  //20-sec red
    light = ORANGE;
    delay(2000);   //2-sed orange
    light = GREEN;
    delay(20000);  //20-sec green
}
```

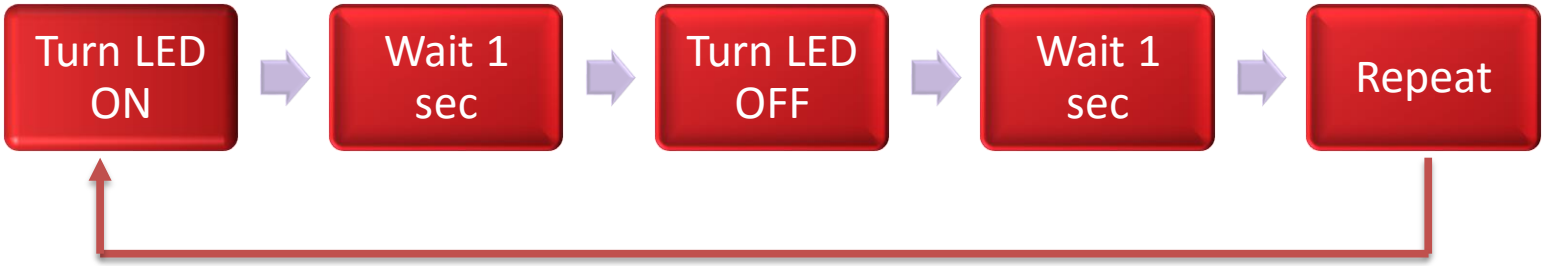
HELLO WORLD

```
/*
My First program : HELLO WORLD
*/

void setup()
{
  Serial.begin(9600);                    // initialize serial communications at 9600 bps
}

void loop() {

 // print to the serial monitor:
 Serial.print("Hello World !!! " );
 Serial.println("Hi Everyone…" );

 }
```

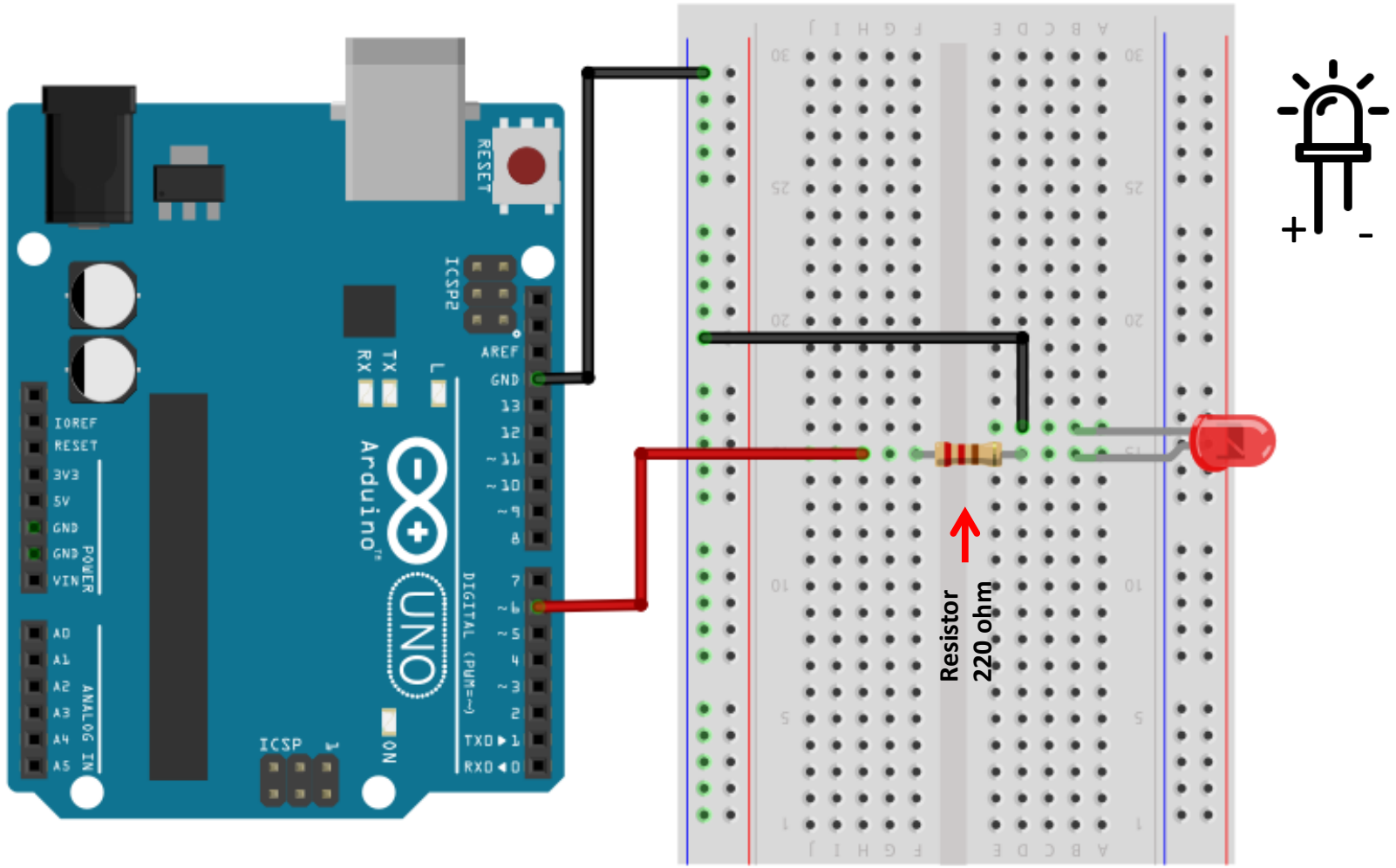
Let's start coding… with Arduino

- Project #1 – Blink
  - "Hello World" of Physical Computing

- *Psuedo-code – how should this work?*



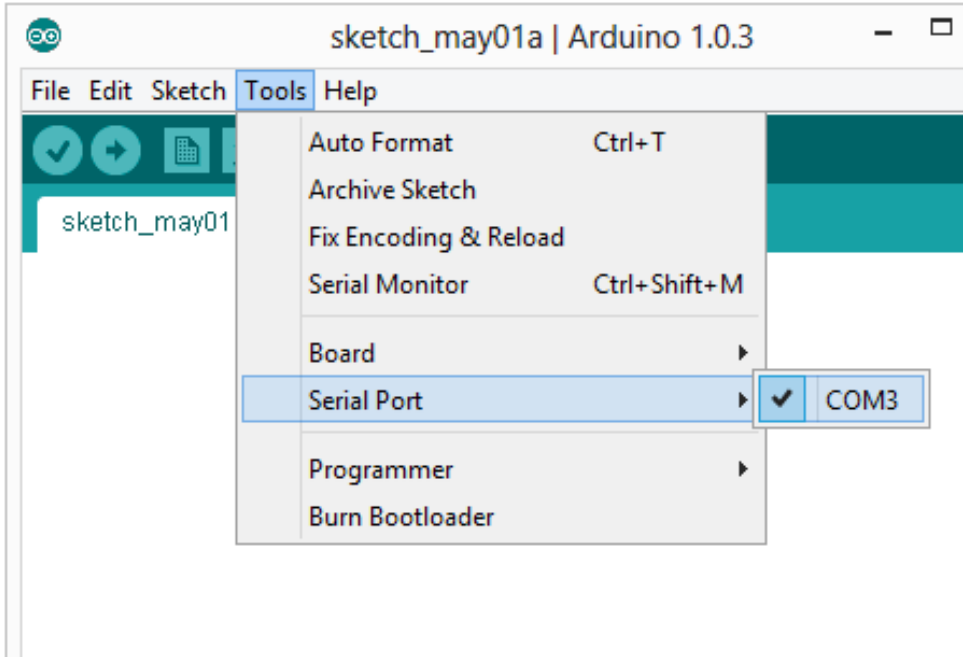
Activity # 10 - Single LED Connection



Fritzing

IDE ARDUINO Board Communication

Settings: Tools → Serial Port

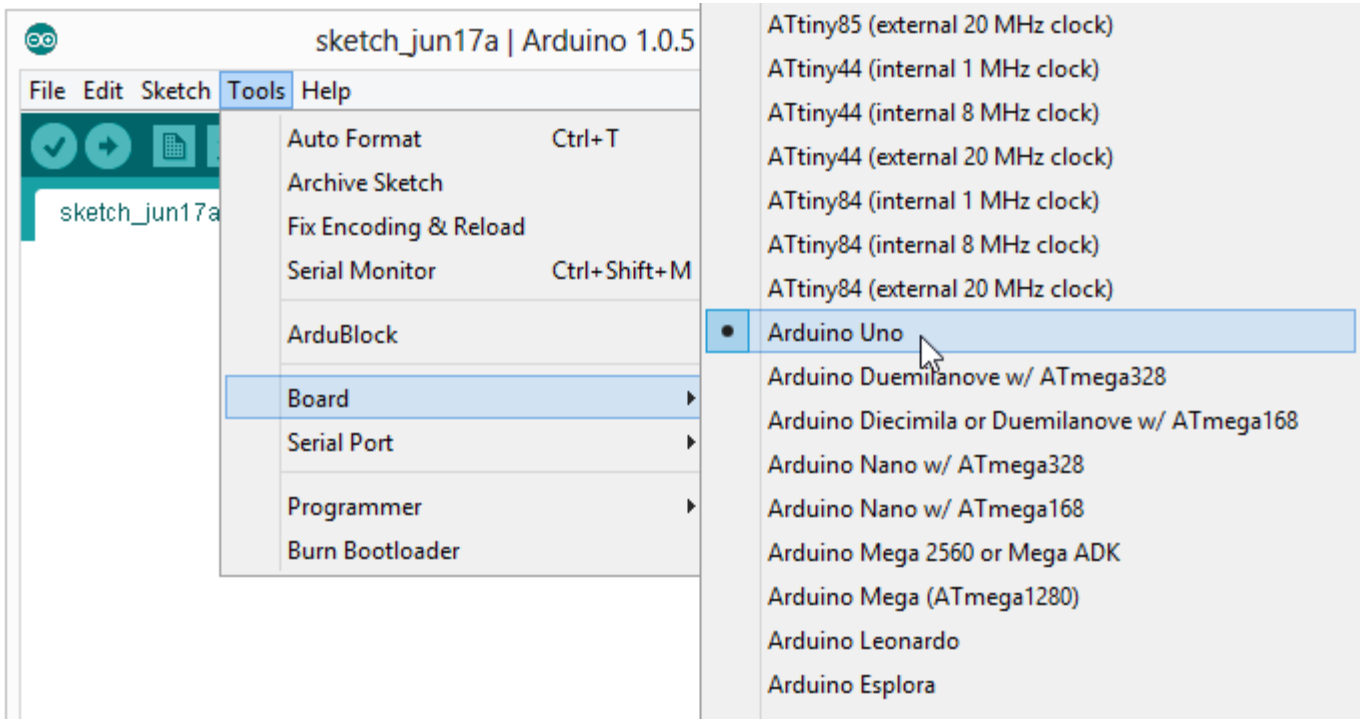


- Your computer communicates to the Arduino microcontroller via a serial port → through a USB-Serial adapter.

- Check to make sure that the drivers are properly installed.

ARDUINO Board Selection

Settings: Tools → Board



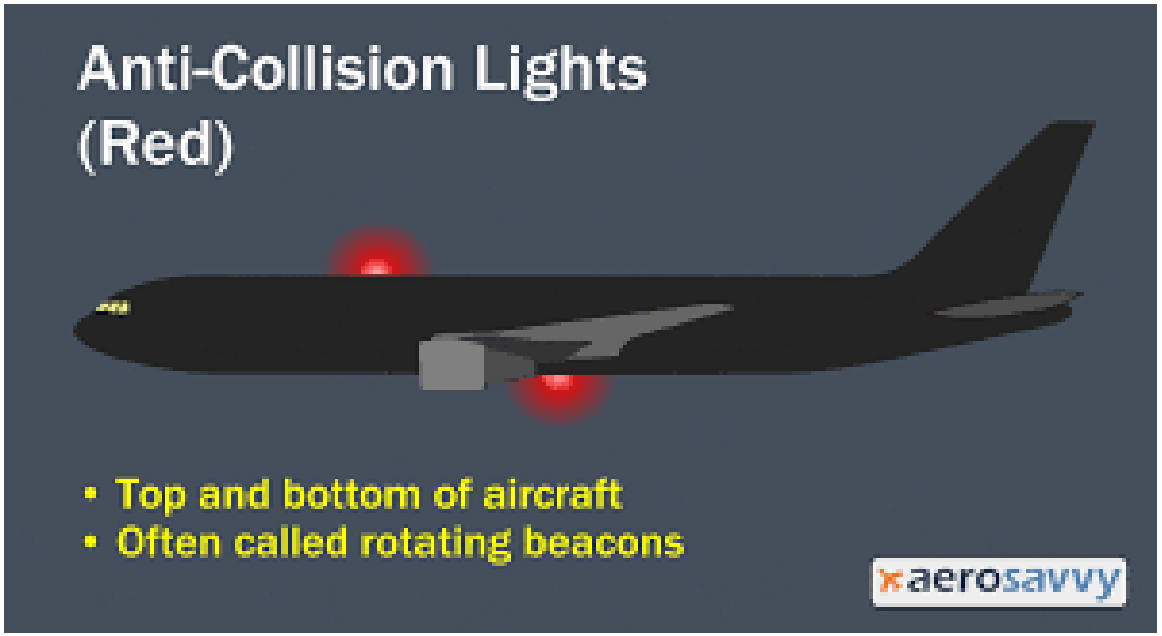
•Next, double-check that the proper board is selected under the Tools→Board menu.

Project #1: BLINKING LED

```
/*
Project #1 : BLINKING LED
*/

void setup()
{
pinMode(6, OUTPUT);          //configure pin 6 as output
}

void loop()                  //blink a LED repeatedly
{
digitalWrite(6,HIGH);        // turn the LED on
delay(1000);                 // wait 1000 milliseconds
digitalWrite(6,LOW);         // turn the LED off
delay(1000);                 // wait 1000 milliseconds
}
```

Save it as *Blinking Led*

DIY # 8 - Modify the Algorithm/FlowProgram for Aircraft Anti-Collision Strobe Light



Modify Project #1 Code to make the LED blink like an Anti Aircraft Collision Strobe Light

Save it as *Strobe Light*

LED challenges

- **Challenge 1a** – blink with a 200 ms second interval.

- **Challenge 1b** – blink to mimic a heartbeat

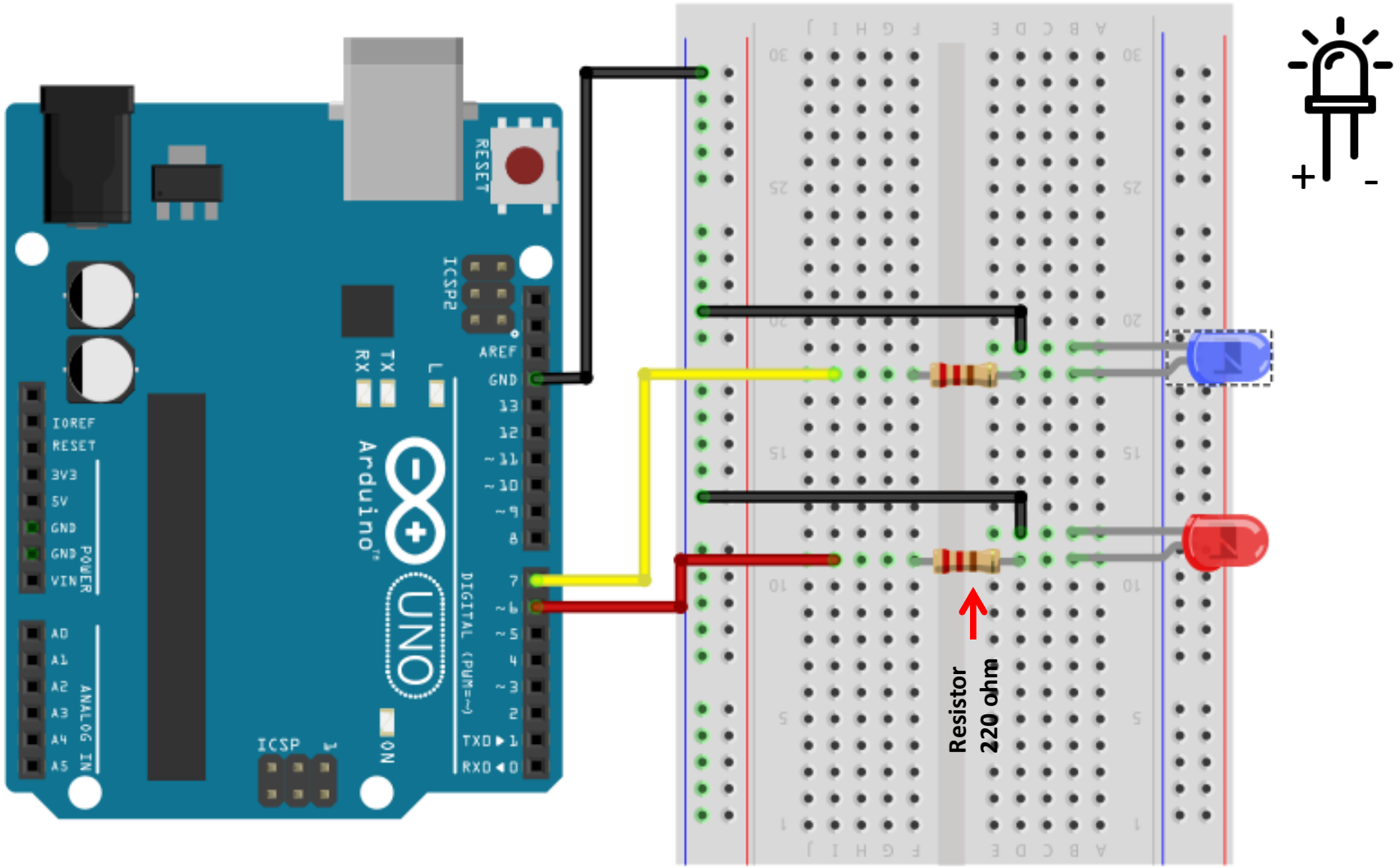
- **Challenge 1c** – find the fastest blink that the human eye can still detect…

  1 ms delay? 2 ms delay? 3 ms delay???

Building Real-World Digital Output project

2- LEDs
Police car Siren

Activity # 15 - 2 LED Connection



Fritzing

Police Car Siren Project

Activity # 16 – Code Police Car Siren Light

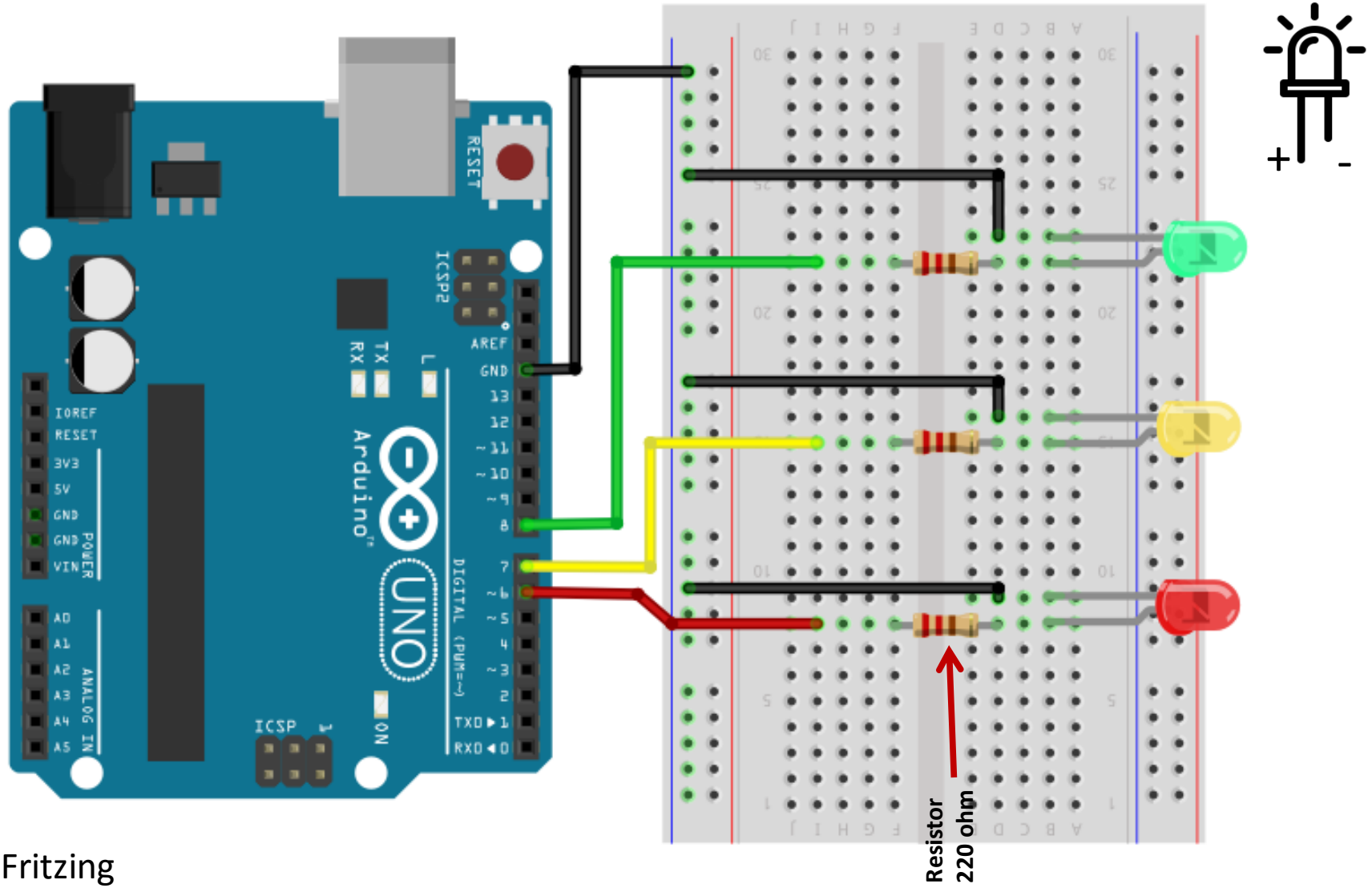
```
/*
Project #2 : POLICE SIREN LIGHT
*/

void setup()
{
pinMode(6, OUTPUT);
pinMode(7, OUTPUT);
}

void loop()
{
digitalWrite(6,HIGH);
delay(500);
digitalWrite(6,LOW);
delay(500);
digitalWrite(6,HIGH);
delay(500);
digitalWrite(6,LOW);
delay(500);
digitalWrite(7,HIGH);
delay(500);
digitalWrite(7,LOW);
delay(500);
digitalWrite(7,HIGH);
delay(500);
digitalWrite(7,LOW);
delay(500);
}
```

Save it as *Police car light*

Building Real-World Digital Output project

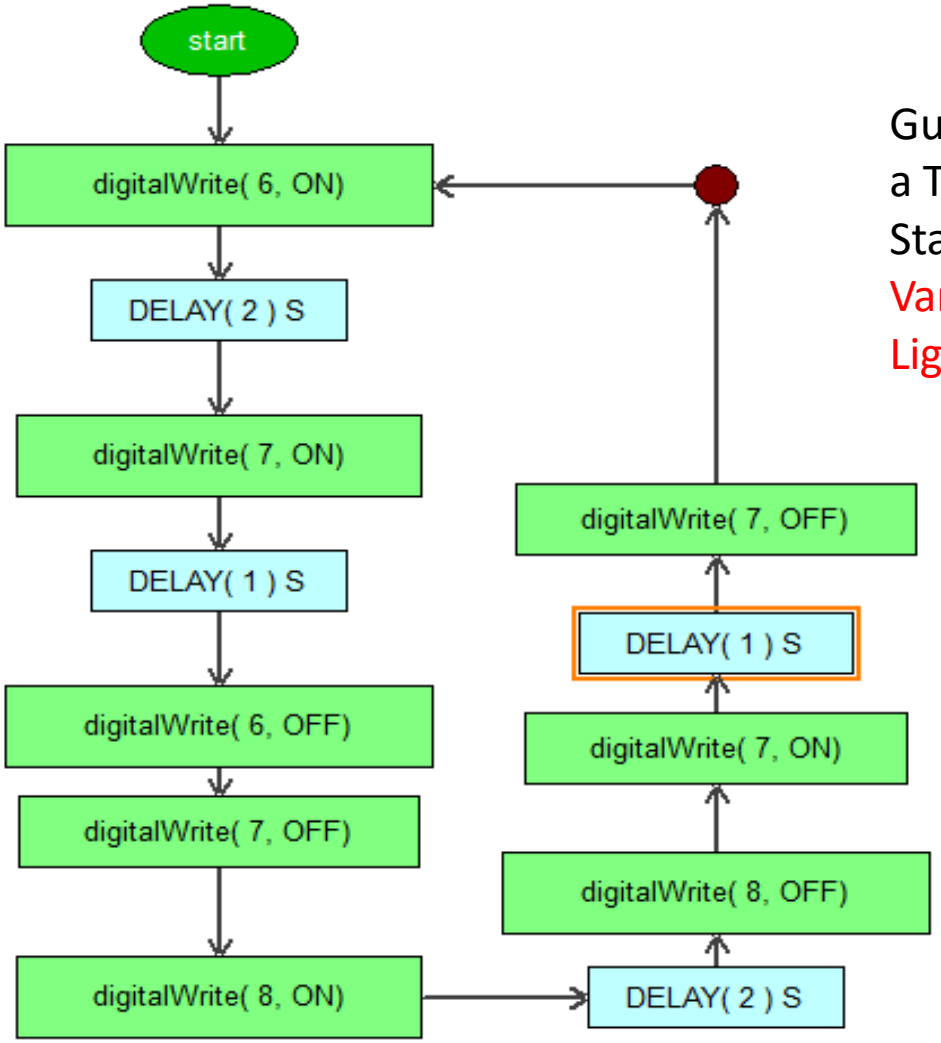
3- LEDs Traffic Light system

Activity # 17 - 3 LED Connection



Fritzing

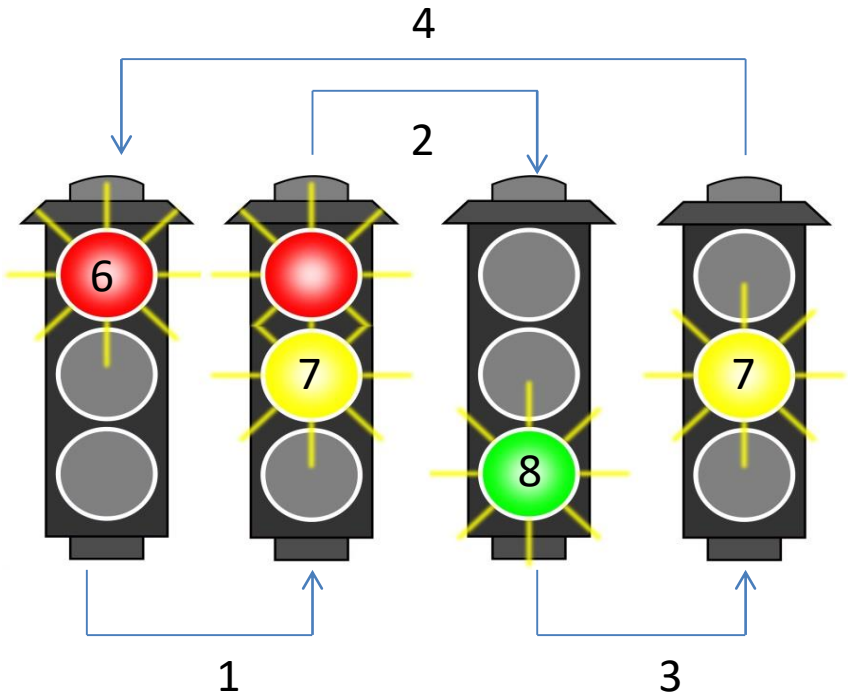
Activity # 19 - Algorithm/FlowProgram for Traffic Light system



Traffic Light Control Project

Guide student to write a program to control a Traffic as per sequence below – The UK Standards

Variation: Get the students to program the Traffic Light based on American Standards



Activity # 16 - Code for Traffic Light System

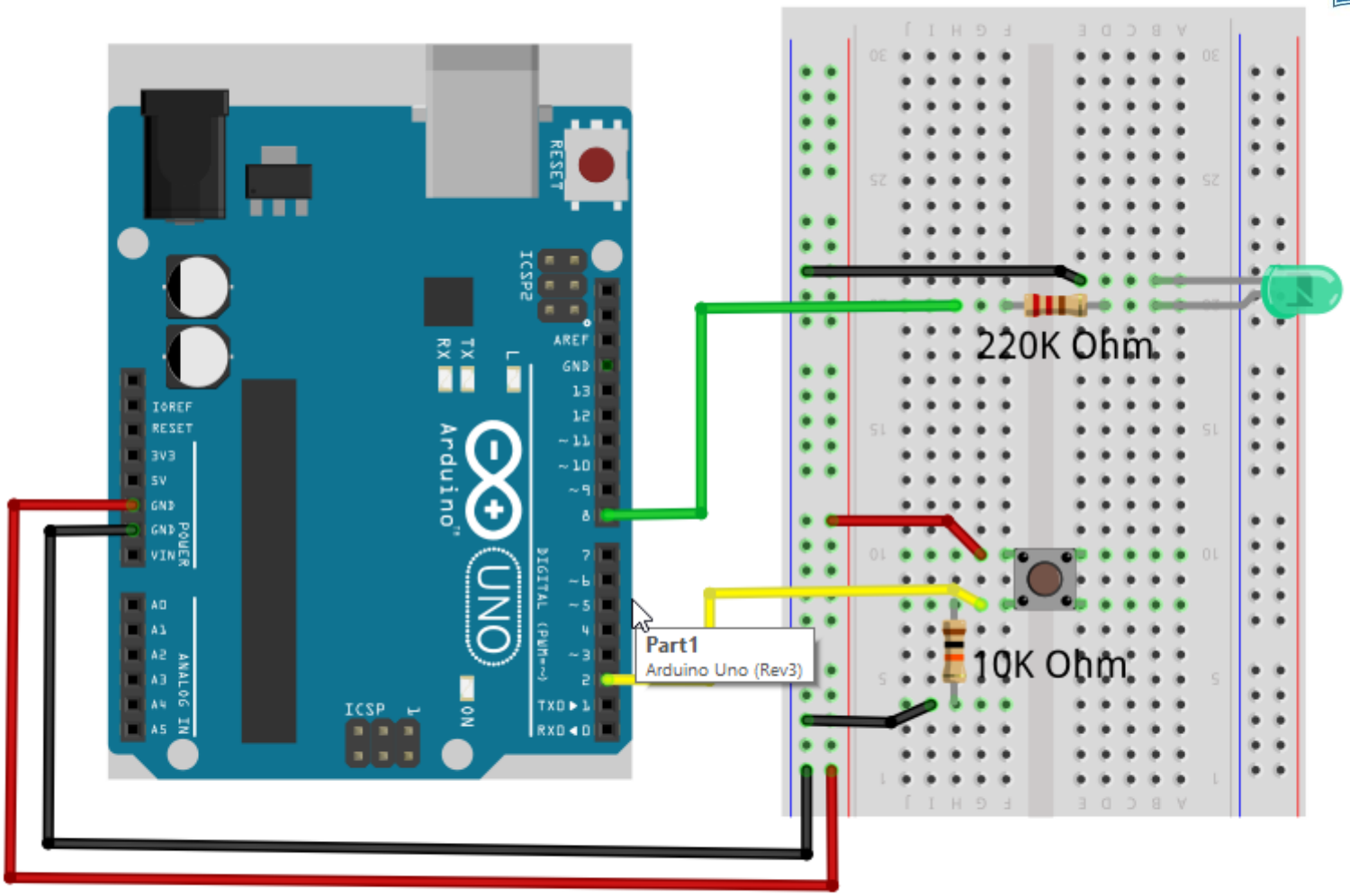
```
/*
Project #2 : TRAFFIC LIGHT SYSTEM
*/

void setup()
{
pinMode(6, OUTPUT);
}

void loop()
{
digitalWrite(6,HIGH);
delay(2000);
digitalWrite(7,HIGH);
delay(1000);
digitalWrite(6,LOW);
digitalWrite(7,LOW);
digitalWrite(8,HIGH);
delay(2000);
digitalWrite(8,LOW);
digitalWrite(7,HIGH);
delay(1000);
digitalWrite(7,LOW);
}
```

Save it as *UK Traffic light*

Building Real-World Digital Input project

LED ON/OFF

Activity # 17 – Push Button and LED Connection



Digital Push Bush button Circuit

Boolean Operators

<Boolean>	Description
( ) == ( )	is equal?
( ) != ( )	is not equal?
( ) > ( )	greater than
( ) >= ( )	greater than or equal
( ) < ( )	less than
( ) <= ( )	less than or equal

Activity # 16 – Code for Push button interface to turn ON/OFF LED

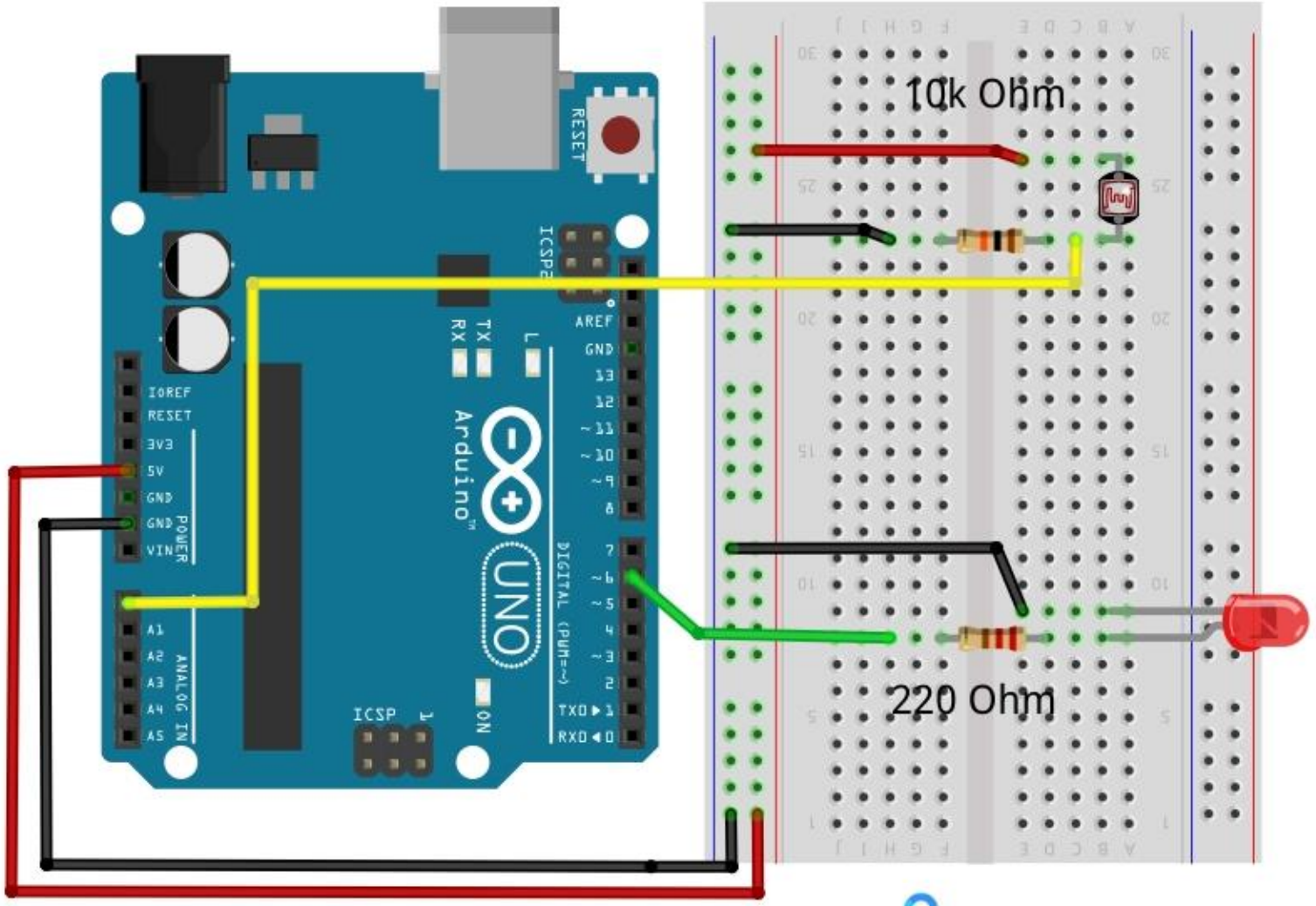
```
/*
Project #3 : Push Button interface
*/

void setup()
{
pinMode(8, OUTPUT);
pinMode(2,INPUT);
}

void loop()
{
    Int buttonState = digitalRead(2)  //Assignment
    If (buttonState == HIGH)
    {
    digitalWrite(8,HIGH);
    }
    Else
    {
    digitalWrite(8,LOW);
    }
}
```

Save it as *Push Button*

Building Real-World Analog Input project

LDR Sensor To turn ON/OFF LED

Activity # 17 – LDR Sensor and LED Connection



LDR Sensor Circuit

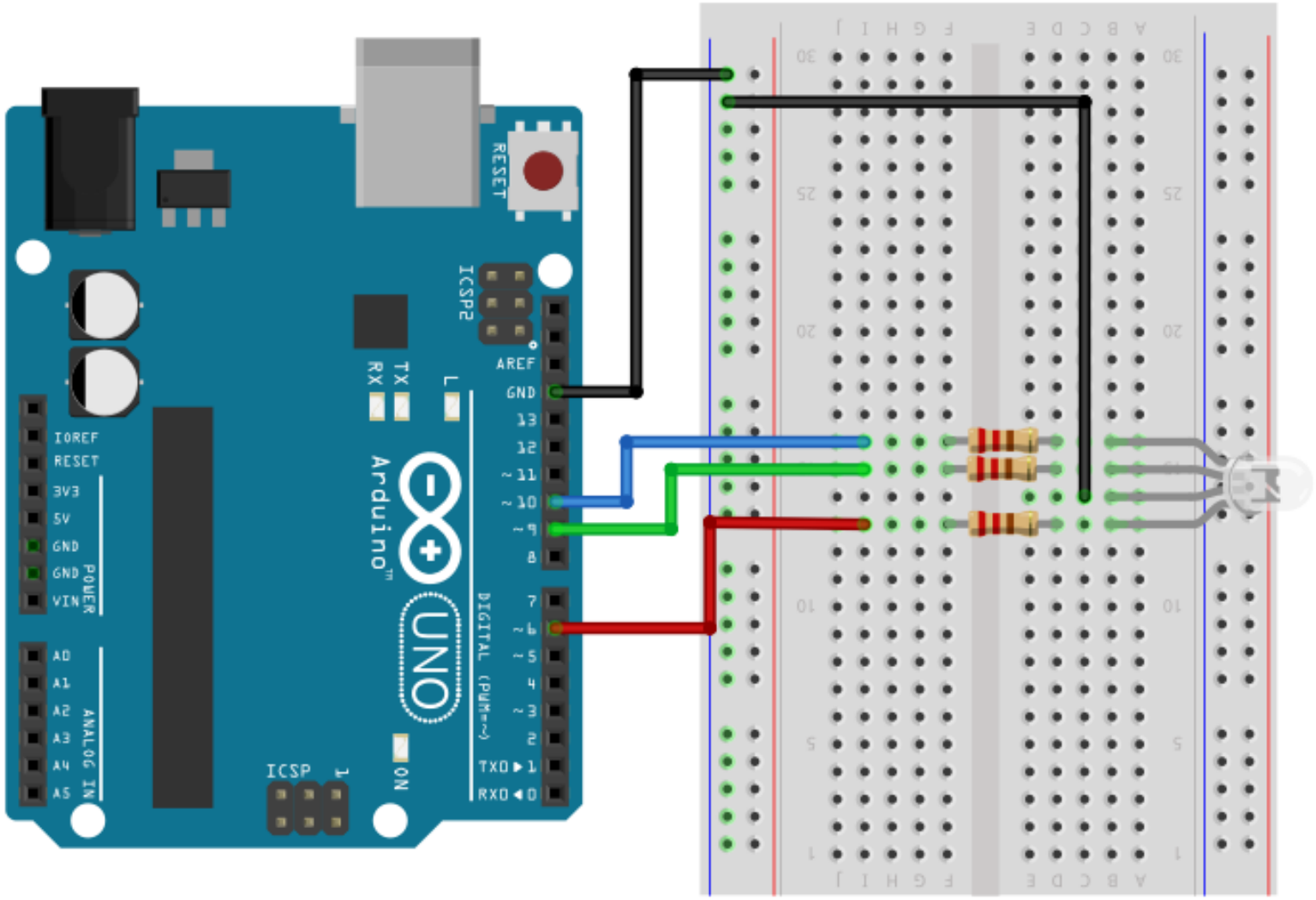
Activity # 16 – Code for Analog Input LDR interface to turn ON/OFF LED

```
/*
Project #4 : Analog Input interface
*/
void setup()
{
  Serial.begin(9600);          // initialize serial communications at 9600 bps
}

void loop() {
   sensorValue = analogRead(A0);      // read the analog in value

 // print the results to the serial monitor:
 Serial.print("sensor = " );
 Serial.print(sensorValue);

 If (sensorValue > 500)
 {
  digitalWrite(6,HIGH);
 }
 Else
 {
 digitalWrite(6,LOW);
 }

  delay(2);
}
```

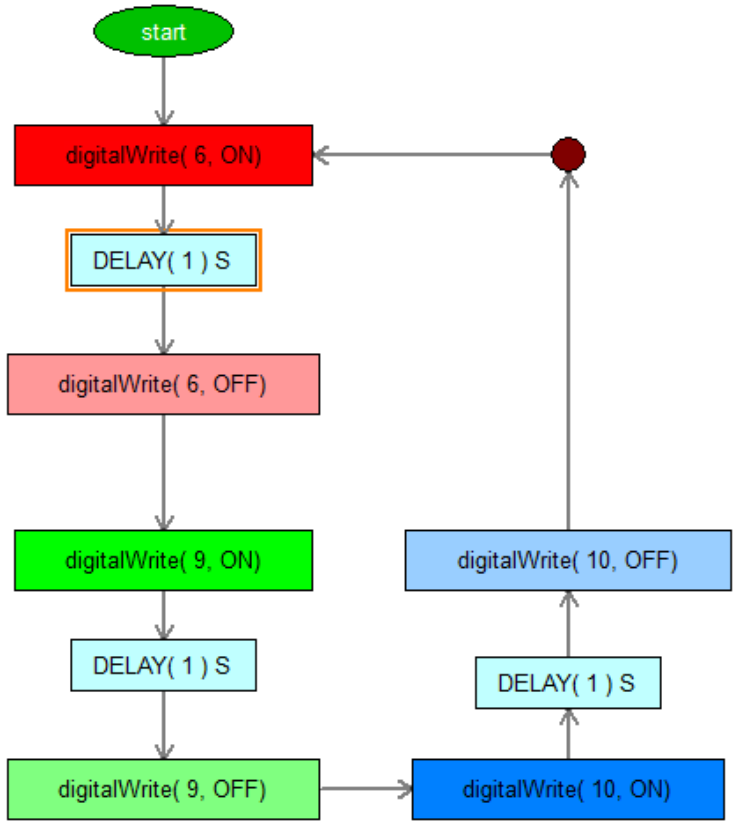
Building Real-World Analog Output project

RGB LED

Activity # 18 - RGB LED Connection



Activity # 19 –Algorithm & C/C++ programming for Color Mixing using RGB Led

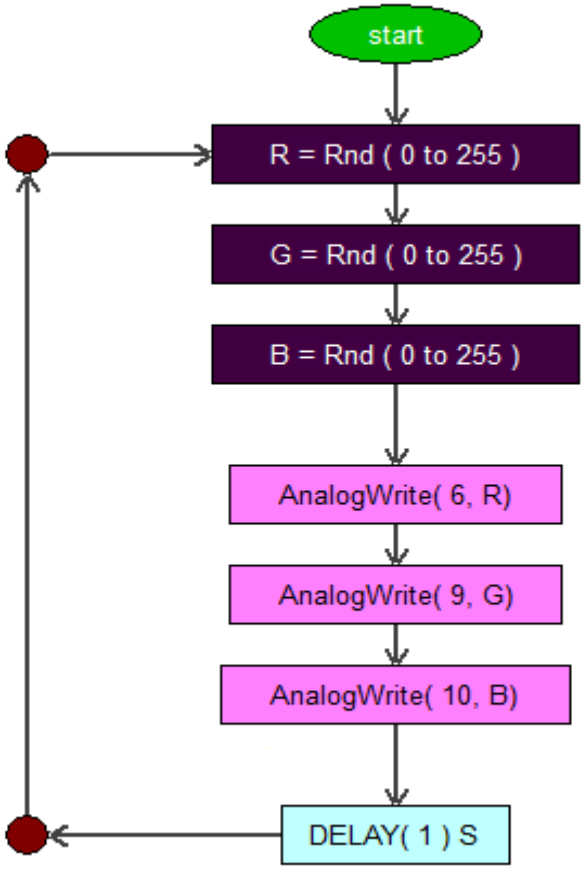


FlowProgram using DigitalWrite () Command Block

```
/*
Project #4 : RGB COLOR MIXING
*/

void setup()
{
pinMode(6, OUTPUT);
pinMode(9, OUTPUT);
pinMode(10, OUTPUT);
}

void loop()
{
digitalWrite(6,HIGH);
delay(1000);
digitalWrite(7,LOWH);
digitalWrite(9,HIGH);
delay(1000);
digitalWrite(9,LOW);
digitalWrite(10,HIGH);
delay(1000);
digitalWrite(10,LOW);
}
```

C/C++ Programming

Activity # 20 –Algorithm & C/C++ programming for Color Mixing using RGB Led

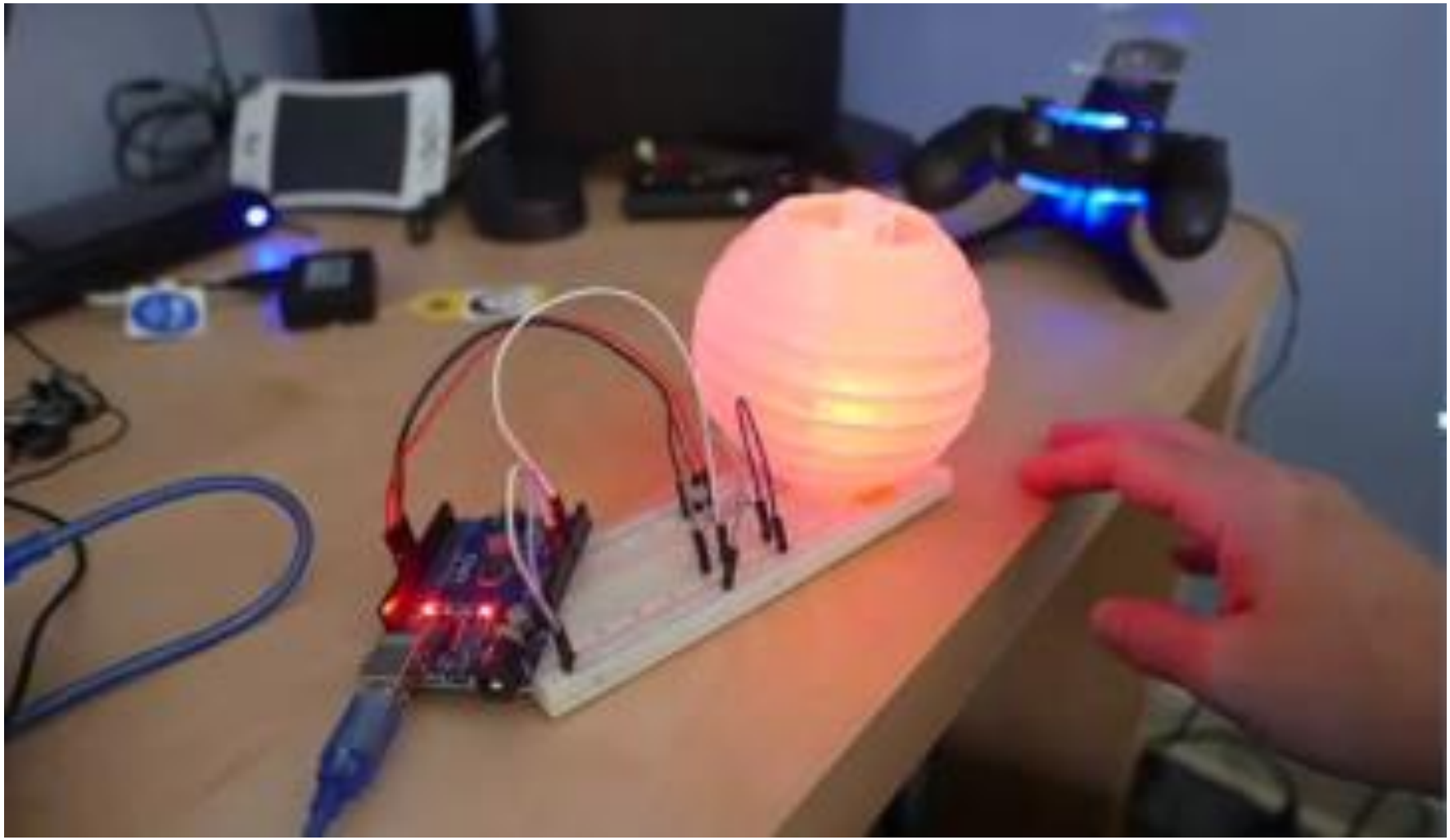


FlowProgram using AnalogWrite () Command Block

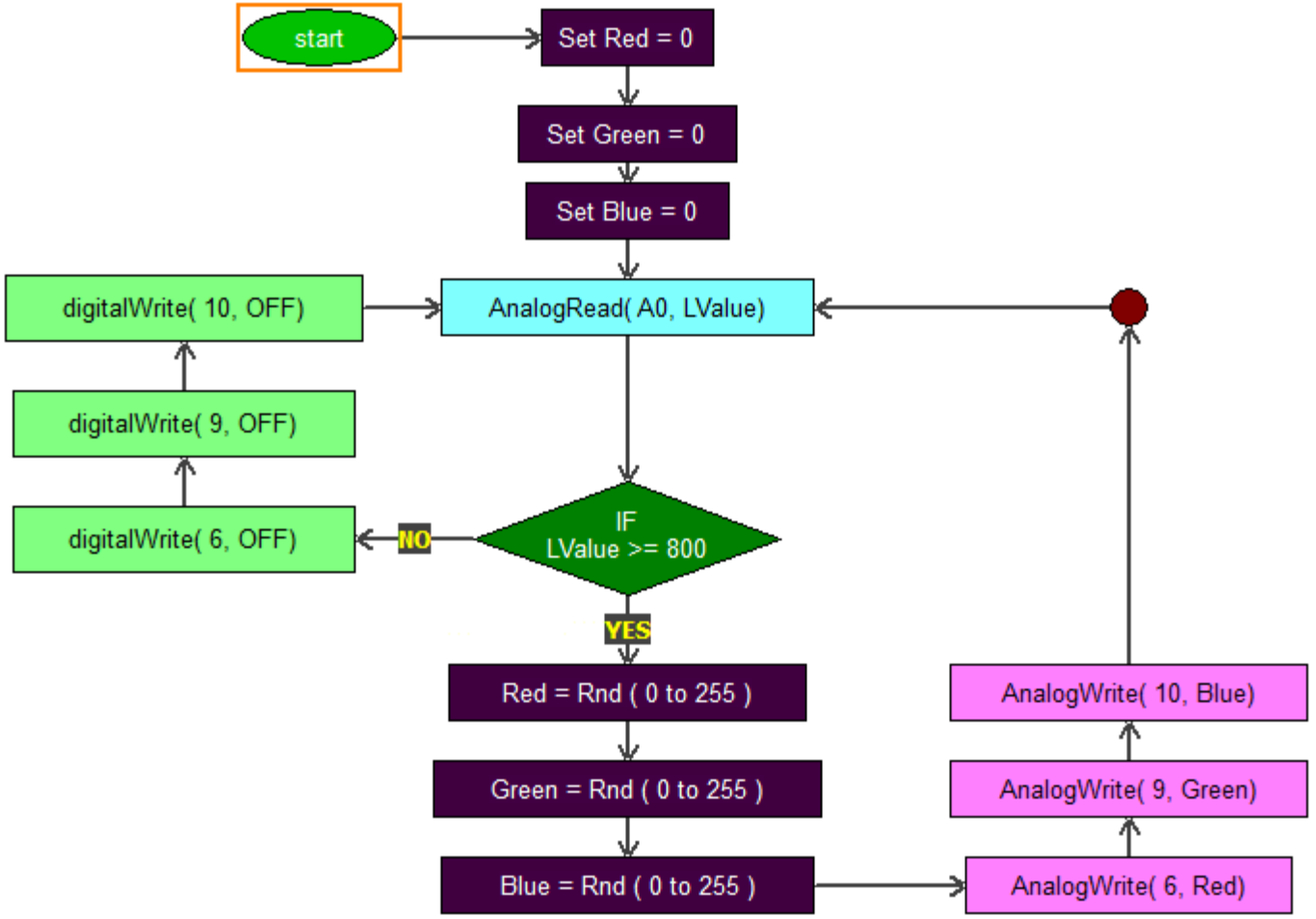
```
/*
Project #5 : RGB COLOR MIXING
*/

int R = 0;
int G = 0;
int B = 0;

void setup()
{
pinMode(6, OUTPUT);
pinMode(9, OUTPUT);
pinMode(10, OUTPUT);
}

void loop()
{
R = random(0,255);
G = random(0,255);
B = random(0,255);

AnalogWrite(6,R);
AnalogWrite(9,G);
AnalogWrite(10,B);
}
```

C/C++ Programming

Mood Lamp – Prototype Model Sample



DIY #10 – FINAL FOUNDATION PROJECT - MOOD LAMP with LDR



DIY #10
Final Project

Design a Creative Mood Lamp and
Develop FlowProgram / Algorithm to animate the colors




END OF

Digital Technology

Programming and Hardware Fundamental

Level 1

Visit www.myflowlab.com to download Beginners Guide
and online video tutorials

myFlowlab™