Computer Engineering

Game Development using Microcontrollers

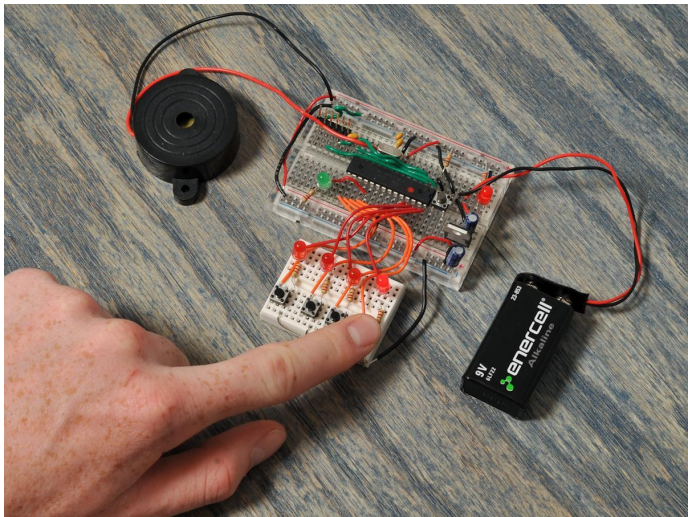
For more info see HERE

Memory Game - Simon

- We are going to create a memory game where the player follows the lights for as long as they can
- Try it https://freesimon.org

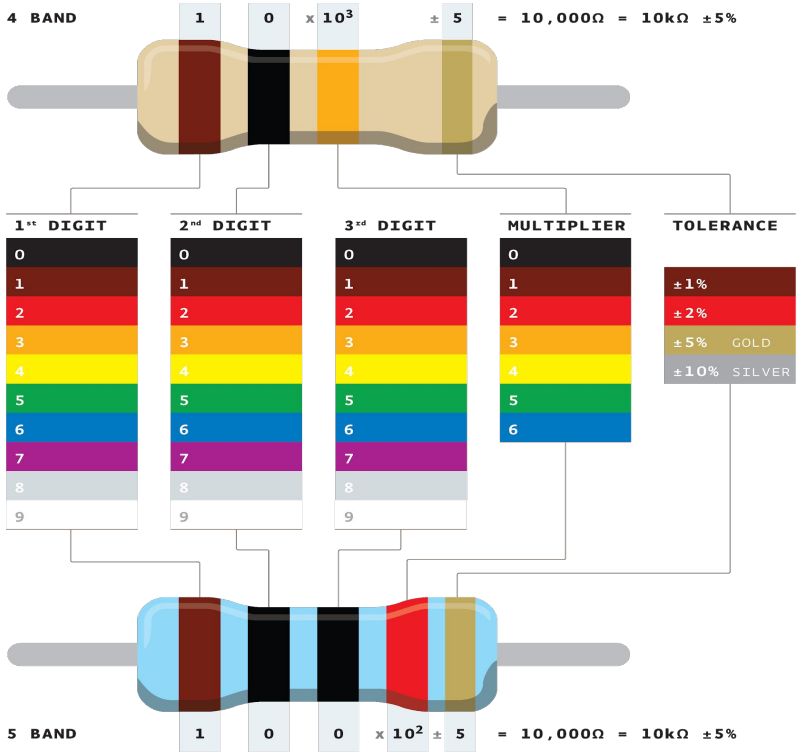
What do we need for this game?

- Interfacing is using a combination of hardware and software (programming) to allow for communication and action
- We will be working towards using an Arduino microcontroller to interface the LEDs, Buzzer, Buttons, and maybe even an LCD display



Resistors Review

- Slows the flow of current in a circuit, which also lowers voltage based on Ohm's law
- The strength of a resistor is measured in Ω (pronounced ohms)
- The colour code of the resistors tell the value of the resistor, e.g. what is the value of a resistor with: blue, green, orange, gold?
- blue = 6, green = 5, orange = 3, so 000 zero's is what I need to put on the end of 65… 65 000 Ω
- Gold is +/- 5%
- 65000 +- 5% is the answer

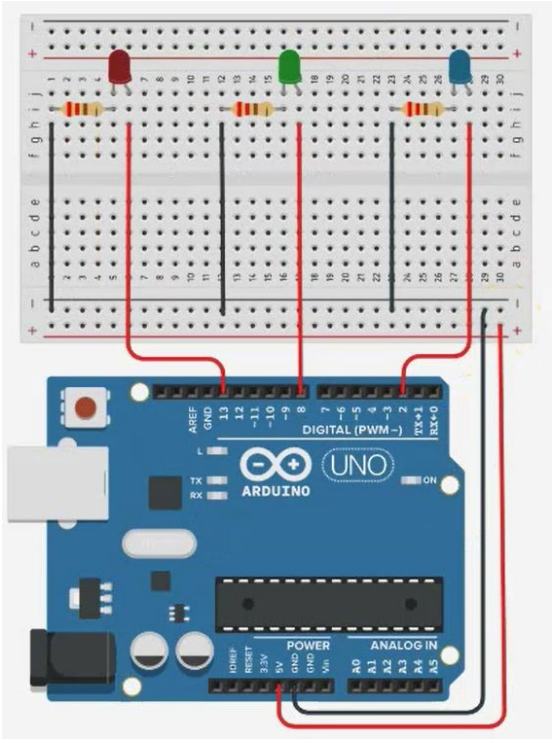


Output Circuit

- This example circuit has no inputs, just 3 output LEDs and resistors to slow down the current, and it is set to blink by this code:

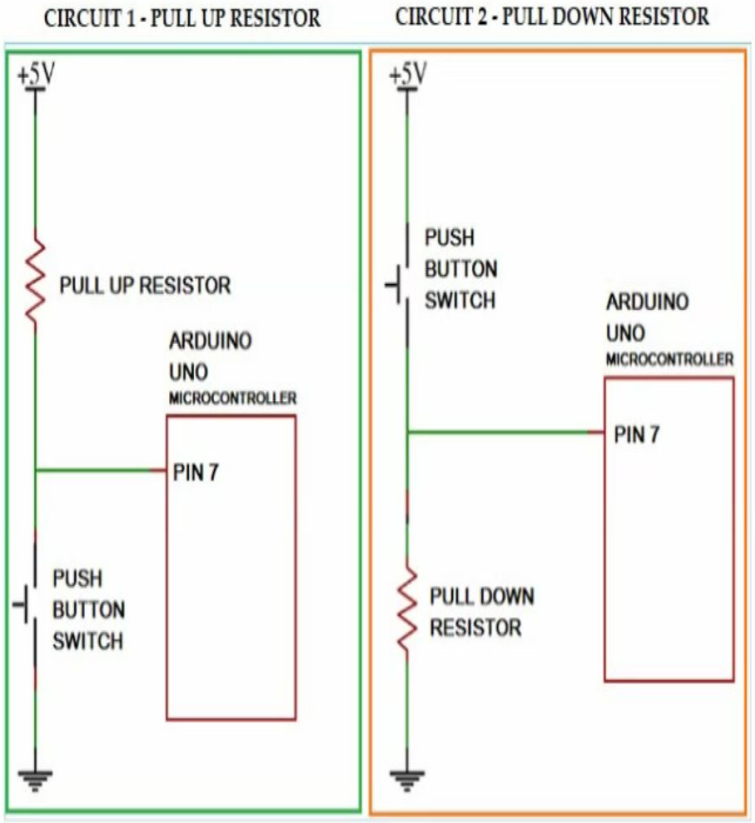
```
// This function only runs once when the Arduino is powered up
void setup() { // must start with an opening curly brace
  pinMode (2, OUTPUT); // Designate pin 2 to be a digital output for the blue LED
  pinMode (8,OUTPUT); // Designate pin 8 to be a digital output for the green LED
  pinMode (13, OUTPUT); // Designate pin 13 to be a digital output for the red LED
} // must stop with a closing curly brace

// The void loop function loops over and over when the Arduino is powered up
void loop() { // must start with an opening curly brace
  digitalWrite(2, HIGH);
  delay(1000);
  digitalWrite(2, LOW); // turn the blue LED on, delay for one second, turn it off
  digitalWrite(8, HIGH);
  delay(1000);
  digitalWrite(8, LOW); // turn the green LED on, delay for one second, turn it off
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW); // turn the red LED on, delay for one second, turn it off
} // must stop with an closing curly brace
```



Input Circuit

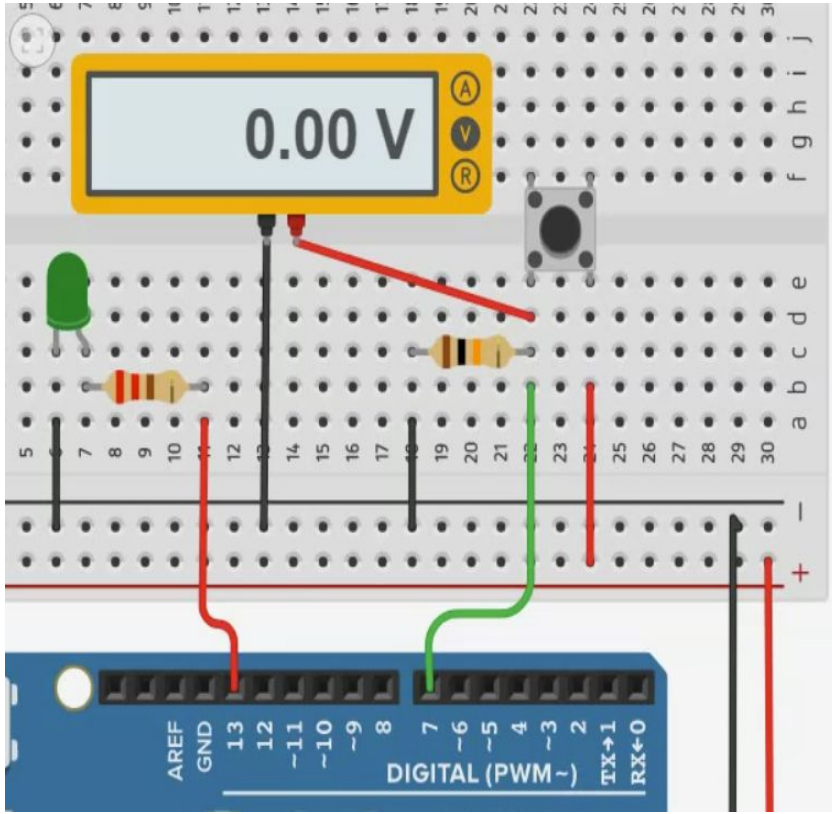
- Typical designers make: Input Circuit → Microcontroller → Output Circuit
- Input circuits may include push buttons, switches, sensors
- Digital signals for Microcontroller are either HIGH (5V) or LOW (ground or 0V)
- Example circuit 1 normally is at HIGH, when button is pushed it goes to LOW
  - PULL UP Resistor keeps it HIGH
- Example circuit 2 normally is at LOW, when button is pushed it goes to HIGH
  - PULL DOWN Resistor keeps it LOW



Pushbutton Input Circuit + LED Output Circuit

- Microcontroller Code needs to know how to react to changes from the input

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 7 as an input and 13 as an output.
    pinMode (7, INPUT);
    pinMode (13, OUTPUT);
}
// the loop function runs over and over again forever
void loop()
{
  if (digitalRead(7) == 1) { // if button is pressed
    digitalWrite(13, HIGH); //light up the LED
  } else { //if button is not pressed
    digitalWrite(13, LOW); //turn off the LED
  }
  delay(10); // Delay a little bit to improve simulation performance
}
```



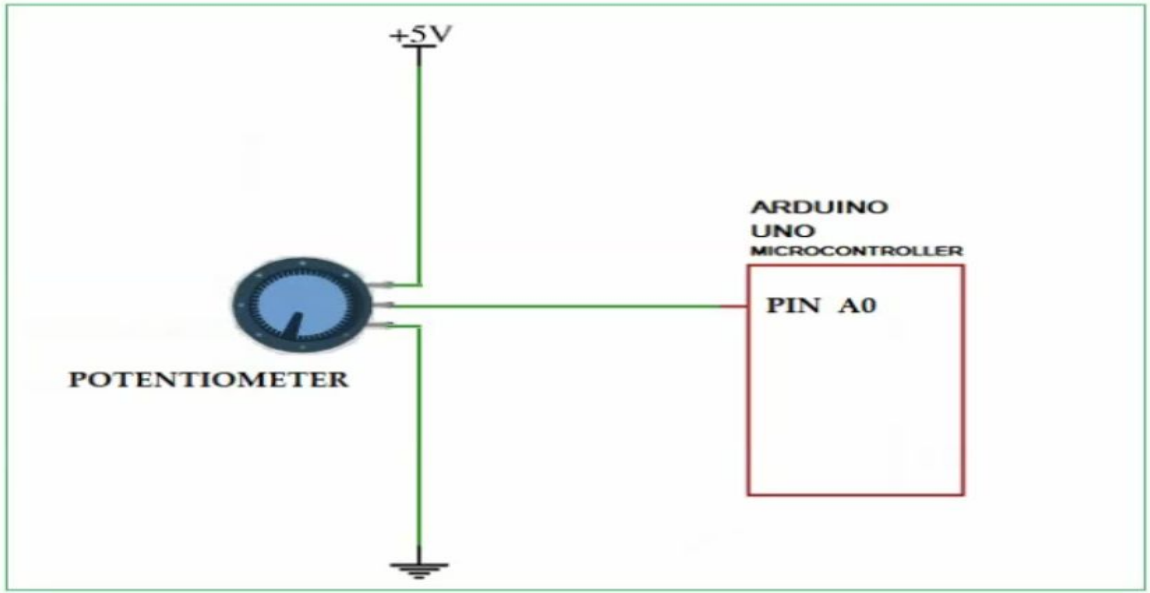
Inputs & Outputs - Digital vs Analog

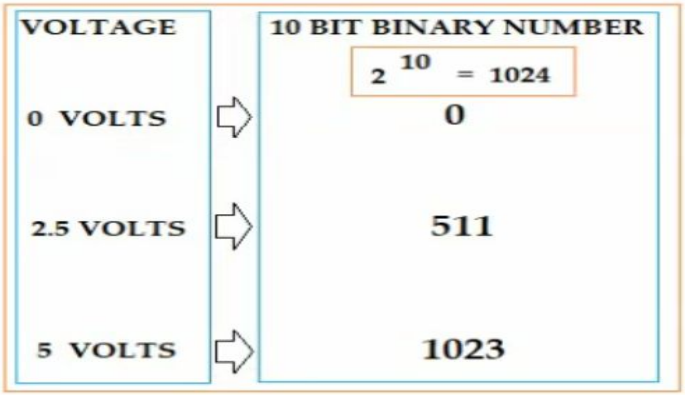
- Digital refers to signals that are either ON or OFF
- Analog refers to having a continuously variable physical quantity
- Trick Question: is the LED a Digital or Analog output?
- Answer: Depends on the controller… LEDs can be dimmed with the appropriate controller (in the Arduino microcontroller there are both digital and analog pins)
- What are some Digital/Analog Inputs & Outputs for your smartphone?

	Inputs	Outputs
Digital	ON/OFF Switch Bluetooth Keyboard (either a key is pressed or it is not)	Vibration Motor (either it shakes or it does not) (but some models have motors with variable speed controllers, so they are Analog) Flash (either it is on or it is off)
Analog	Gyro Sensor Compass Sensor	Screen (varying degrees of brightness) Speaker (varying degrees of loudness)

Using an Analog Input in the Input Circuit

- Previous examples used Digital Inputs... now we will learn about analog inputs
- Potentiometer the proper name for a "Dimmer Switch"
- The Potentiometer gives out a voltage between 0V and 5V and it gets translated into a number from 0 to 1023



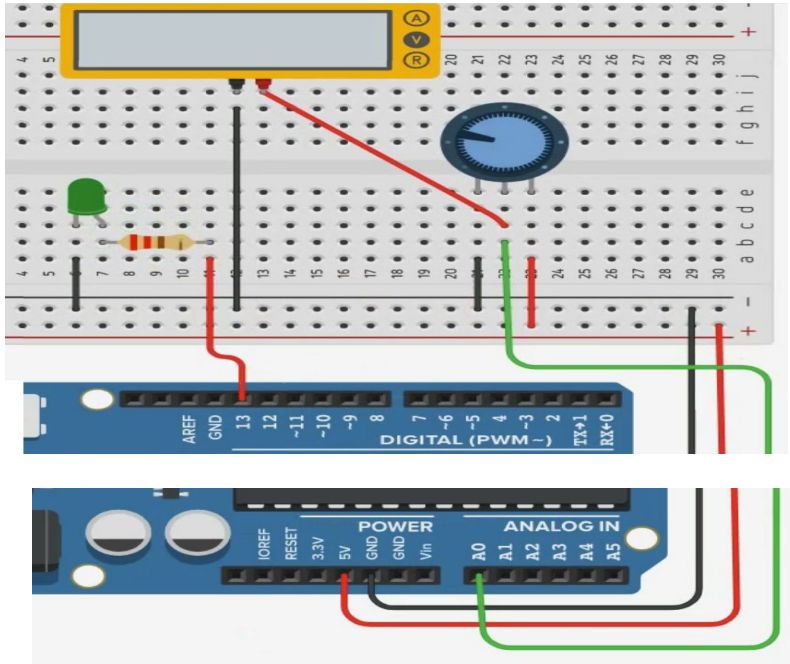


Using a Potentiometer Input in the Input Circuit

- Code for using Analog Input is a little different compared to using Digital Input

```
int sensorValue = 0;  // variable to store the value coming from the sensor

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
  sensorValue = analogRead(A0);   // read the value from the sensor:
  digitalWrite(13, HIGH);  // turn the ledPin on
  delay(sensorValue);  // stop the program for <sensorValue> milliseconds:
  digitalWrite(13, LOW);  // turn the ledPin off:
  delay(sensorValue);  // stop the program for <sensorValue> milliseconds:
}
```



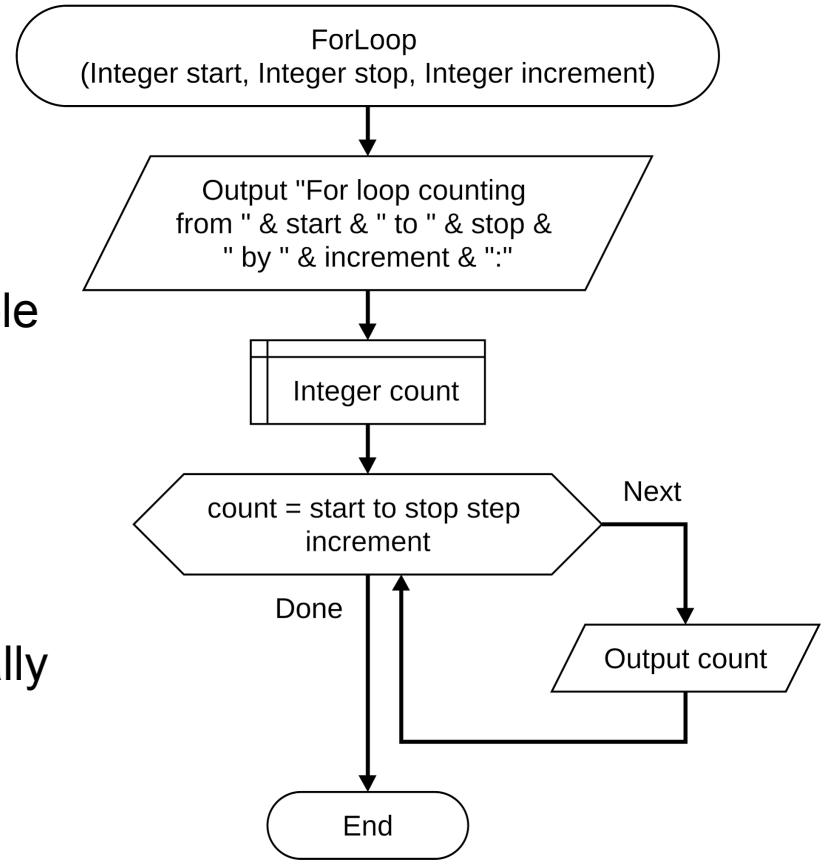
- The blinking speed of the LED depends on the potentiometer

Introduction to Loops - For Loop

- Loops are used in many different programming languages to tell the controller to go again
- A FOR loop allows you to control the number of times a loop occurs, for example

```
for (int i = 0; i < 3; i++) {
  delay(1000); // Wait a second
}
```

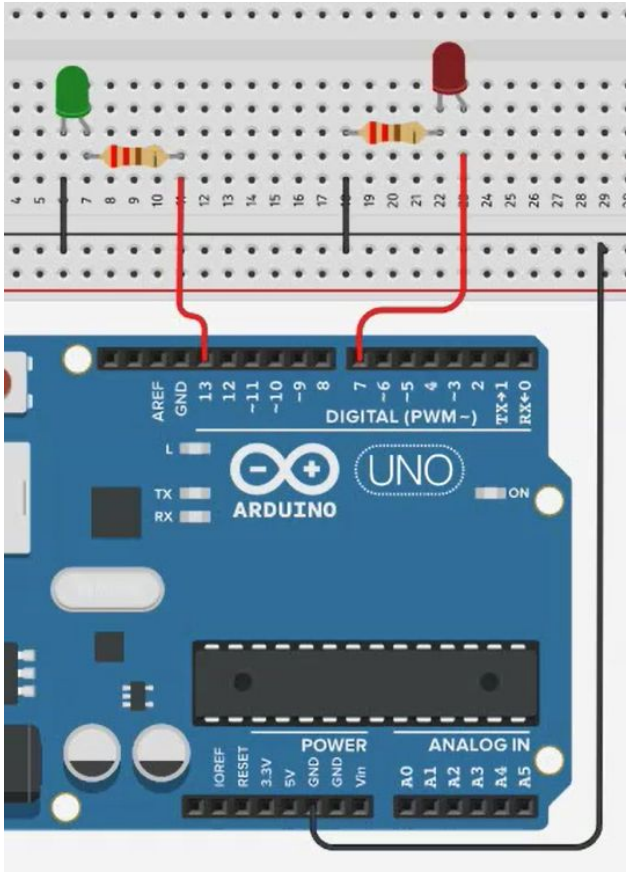
- i=0 tells the loop where to start counting from, i<3 tells the loop to keep going as long as i<3 (which means 0,1,2 only), finally i++ (which is shorthand for i = i + 1) adds one to the loop each time it loops



Using For Loops

- Example for loop code flashes red LED 3 times and then green LED 4 times

```
void setup() {
  pinMode (7, OUTPUT); // set up pin 7 as a digital output pin
  pinMode (13, OUTPUT); // set up pin 13 as a digital output pin
}
void loop() {
  for (int i = 0; i < 3; i++) { // 0,1,2 (loops a total of 3 times)
    digitalWrite(7, HIGH); // set pin 7 to high
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(7, LOW); // set pin 7 to low
    delay(1000); // Wait for 1000 millisecond(s)
  }
  for (int k = 5; k < 9; k++) { // 5,6,7,8 (loops a total of 4 times)
    digitalWrite(13, HIGH); // set pin 13 to high
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(13, LOW); // set pin 13 to low
    delay(1000); // Wait for 1000 millisecond(s)
  }
}
```

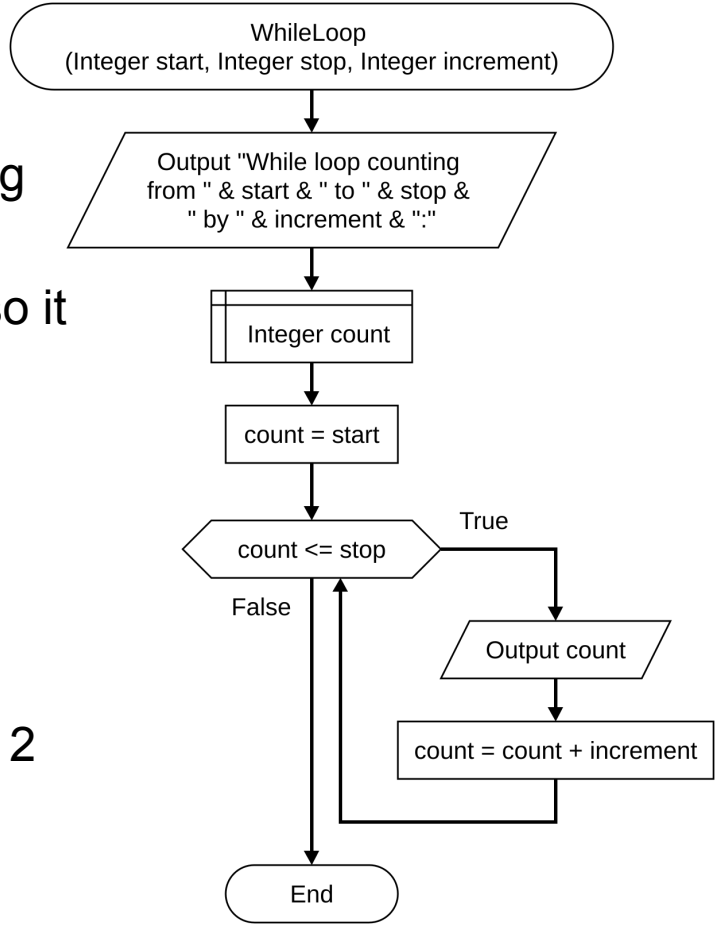


Introduction to Loops - While Loop

- In addition to the FOR loop there's the WHILE loop, A WHILE loop will continue to loop as long as a condition is TRUE
- In this example the condition is (countup < 3) so it keeps going until countup is 3 or more

```
int countup = 1;
while (countup < 3) {
  delay(1000); // Wait a second
  countup++;
}
```

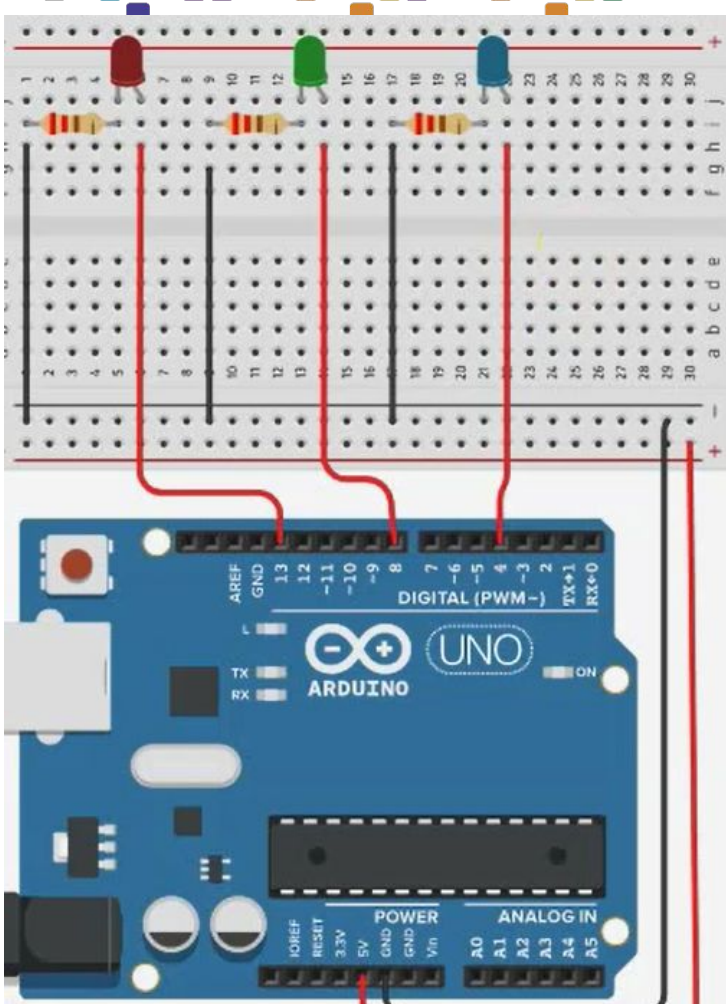
- countup is started as 1, so the while loop goes 2 times (when countup is 1 and 2)



Using While Loops

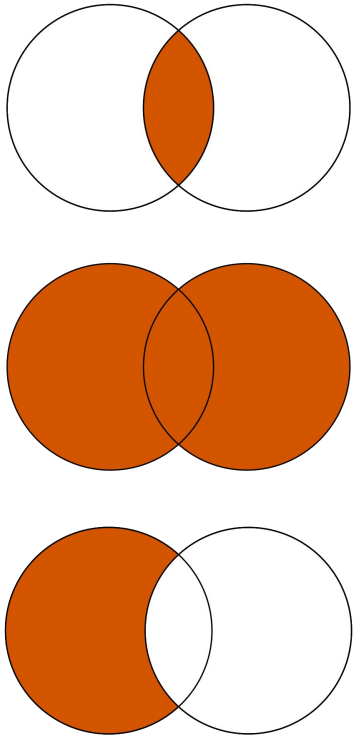
- Example while loop code flashes blue/green/red, for a total of 2 times

```
int countup = 1;
void setup() {
  pinMode (4, OUTPUT); // pin 4 for blue LED
  pinMode (8, OUTPUT); // pin 8 for green LED
  pinMode (13, OUTPUT); // pin 13 for red LED
}
void loop() {
  while (countup < 3) {
    digitalWrite(4, HIGH); // set pin 4 to high
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(4, LOW); // set pin 4 to low
    digitalWrite(8, HIGH); // set pin 8 to high
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(8, LOW); // set pin 8 to low
    digitalWrite(13, HIGH); // set pin 13 to high
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(13, LOW); // set pin 13 to low
    countup++;
}
```



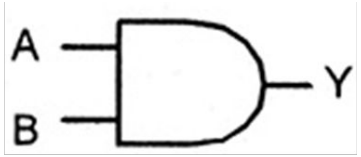
Boolean (Binary) Logic

- 1 in the Binary System represents ON or YES
- 0 in the Binary System represents OFF or NO
- Boolean logic is a part of almost every aspect of COMPUTER ELECTRONICS
- In the 1850s George Boole developed a new form of algebra, now called Boolean algebra in his honour
- Boolean equations use the binary number system to provide a very precise way of illustrating the logic of computer chips
- Boolean equations were used long before computers or even electricity was invented!

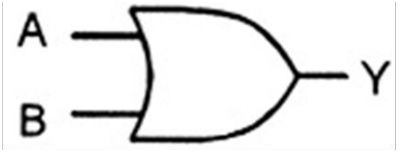


The AND and OR Gates

- The AND gate: Both inputs must be 1(YES) to get a 1(YES) as output
- The OR gate: If either input is 1 (YES) the output is 1 (YES)



A	B	Y (Output)
0	0	0
0	1	0
1	0	0
1	1	1



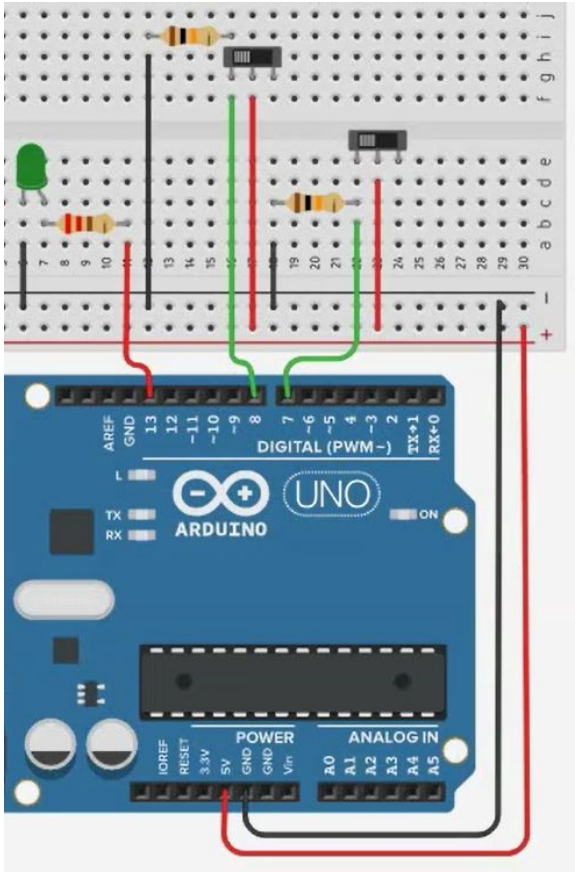
A	B	Y (Output)
0	0	0
0	1	1
1	0	1
1	1	1

Programming with AND and OR Operators

- To use the AND operator type 'and' or '&&'
- To use the OR operator type 'or' '||'
  - This "|" symbol is called 'pipe' and you will find the key on the keyboard above the backslash (above the enter key)

```
int bottomSwitch = 0; // create a variable to store the state of bottomSwitch
int topSwitch = 0; // create a variable to store the state of topSwitch
void setup() { // This function only runs once when the Arduino is powered up
  pinMode (7, INPUT); // set up pin 7 as a digital input pin
  pinMode (8, INPUT); // set up pin 8 as a digital input pin
  pinMode (13, OUTPUT); // set up pin 13 as a digital output pin
}
void loop() { // The void loop function loops over and over when the Arduino is powered up
  bottomSwitch = digitalRead (7); // read the value from pin 7 and store in bottomSwitch
  topSwitch = digitalRead(8); // read the value from pin 8 and store in topSwitch
  if (bottomSwitch == 1 && topSwitch == 1) { // using Boolean operators
    digitalWrite(13, HIGH); // set up pin 13 high if the condition is true
  } else {
    digitalWrite(13, LOW); // set up pin 13 low if the condition is false
  }
}
```

- Replace the && with || and see what happens

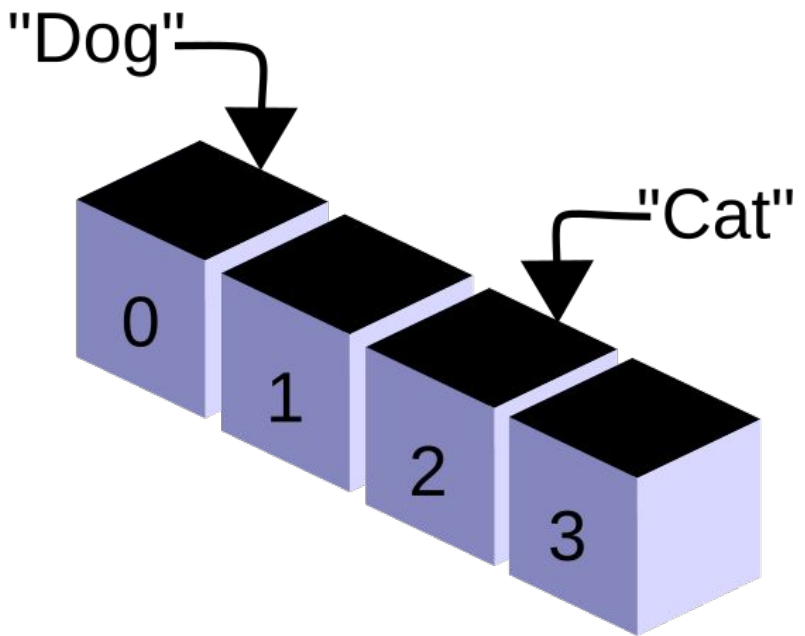


Introduction to Arrays

- Arrays is one way to refer to things using numbers
- A lot of programming refer to the first item as "zeroth"
- In this example the computer is keeping track of what animal is in the petHouse

```
string petHouse = { "Dog", "" , "Cat", "" };
```

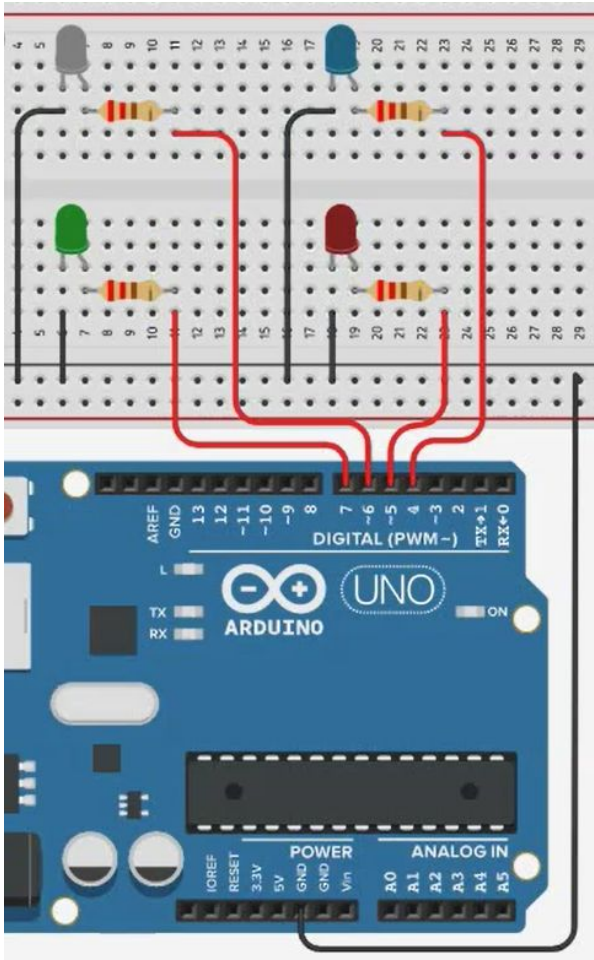
- petHouse[0] is "Dog"
- petHouse[1] is "" (nothing)
- petHouse[2] is "Cat"
- petHouse[3] is "" (nothing)



Using Arrays

- Example code simulates a random 4 colour sequence

```
int number; // This will store the current pin number which represents the LED colour
int sequence [4]; // This creates a one dimensional array to hold four numbers which are colours
void setup() {
  pinMode (4, OUTPUT); // This sets pin 4 as an output
  pinMode (5, OUTPUT); // This sets pin 5 as an output
  pinMode (6, OUTPUT); // This sets pin 6 as an output
  pinMode (7, OUTPUT); // This sets pin 7 as an output
  randomSeed (analogRead (A0)); //Do not connect anything to A0 this creates a random number
}
void loop() {
  for (int i = 0; i < 4; i++) { // This FOR loop fills the array with four spaces with...
    sequence [i] = random (4,8); // random integer from 4 to 7 (not 8)
  }
  for (int i = 0; i < 4; i++) { // outputs the array to LEDS serial monitor
    number = sequence [i]; // pull the i'th item from the array and store in number
    digitalWrite (number, HIGH); // whatever LED number is pulled from the array turn it on
    delay(1000); // hold for one second
    digitalWrite (number, LOW); // turn off the LED
    delay(4000); // wait four seconds
  }
}
```



Introduction to Functions

- Why use FUNCTIONS? Functions work well for doing repetitive tasks
- This allows for less code in the main program (void loop) making the program easier to follow and understand
- By "chunking" your code into functions it is easier to run and debug each function as compared to running all of the main code
- Example:

```
void loop() {
  flashButton(4);
  flashButton(5);
  flashButton(6);
}
void flashButton(int num) {
  digitalWrite(num, HIGH);
  delay(1000);
  digitalWrite(num, LOW);
  delay(1000);
}
```

vs

```
void loop() {
  digitalWrite(4, HIGH);
  delay(1000);
  digitalWrite(4, LOW);
  delay(1000);
  digitalWrite(5, HIGH);
  delay(1000);
  digitalWrite(5, LOW);
  delay(1000);
  digitalWrite(6, HIGH);
  delay(1000);
  digitalWrite(6, LOW);
  delay(1000);
}
```

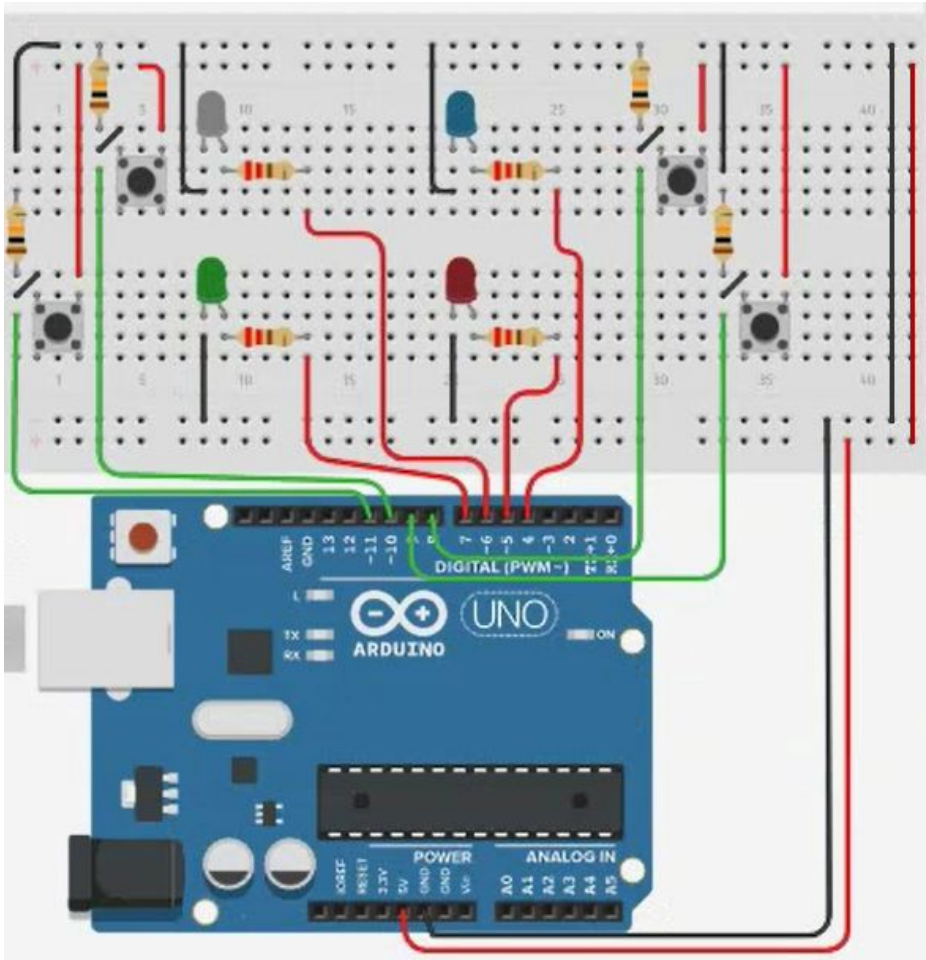
better with function flashButton

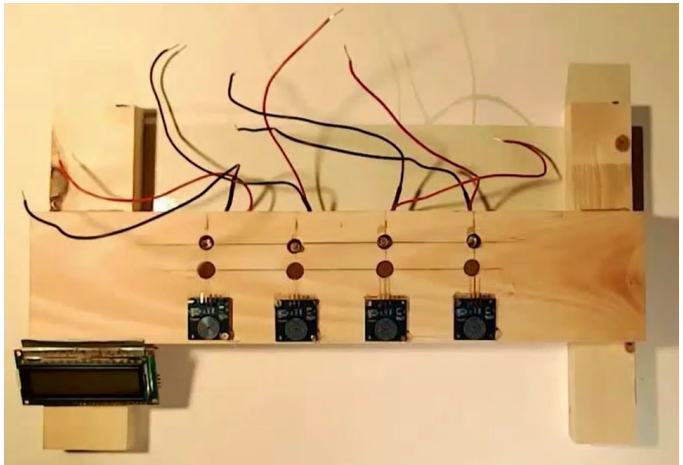
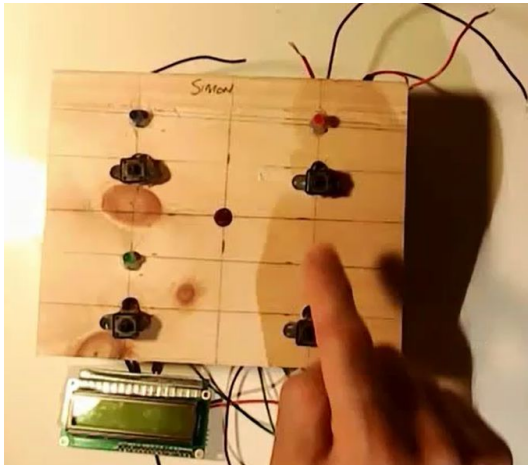
Using Functions

- Example program using a function to check which button was pressed

```
int button;
void setup() {
  pinMode (4, OUTPUT); // This sets pin 4 as an output
  pinMode (5, OUTPUT); // This sets pin 5 as an output
  pinMode (6, OUTPUT); // This sets pin 6 as an output
  pinMode (7, OUTPUT); // This sets pin 7 as an output
  pinMode (8, INPUT); // This sets pin 8 as an input
  pinMode (9, INPUT); // This sets pin 9 as an input
  pinMode (10, INPUT); // This sets pin 10 as an input
  pinMode (11, INPUT); // This sets pin 11 as an input
}
void loop() {
  while (true) { // This WHILE loop will keep running as long as the microcontroller is powered
    button = checkButtons(); // This will "call the function" checkButtons and store in button
    digitalWrite(button, HIGH); // Whatever value is returned and store in "button" will turn on
    delay(100); // Hold this on for a short time (do not make too long)
    digitalWrite(button, LOW); // Turn off the selected LED
  }
}

int checkButtons () { // We use INT for this function as it returns an integer to void loop
  if (digitalRead (8) == HIGH) {
    return 4; // Pass the parameter value of 4 to void loop
  } else if (digitalRead(9) == HIGH) {
    return 5; // Pass the parameter value of 5 to void loop
  } else if (digitalRead(10) == HIGH) {
    return 6; // Pass the parameter value of 6 to void loop
  } else if (digitalRead(11) == HIGH) {
    return 7; // Pass the parameter value of 7 to void loop
  } else {
    return 0;
  }
```



Game Development using Microcontrollers

- Students are now tasked to create a memory game where the player follows the lights for as long as they can
- Students are expected to combine all the content from above to use
  - randomly generate longer and longer number sequences
  - check if the correct button is pressed
  - optional: use a buzzer to indicate wrong button pressed, use LCD to indicate length of colour sequence