Arduino without a kit - ‘Arduino & Tinkercard’

키트없는 아두이노 '아두이노 & 틴커캐드'

공학박사 정경희

METAPLACE

❖(주)메타플레이스 대표이사

❖공학박사 정경희

한림대학교 창업보육센터 12203호
E-mail : metaplace@naver.com

1차시

아두이노 및 시뮬레이터 사용 방법

01 학습목표, 학습내용

학습 목표

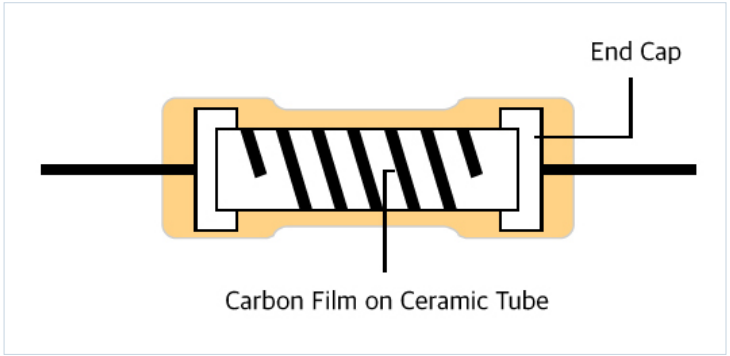
- 아날로그와 디지털 신호를 이해하고 입력과 출력을 다룰 수 있는 방법을 학습한다.
- 아두이노의 기본 개념과 구성요소를 이해한다.
- 틴커캐드를 활용하여 전기신호의 성질과 특성을 이해하고 시뮬레이터를 사용하면서 문제 해결 능력을 강화한다.

학습 내용

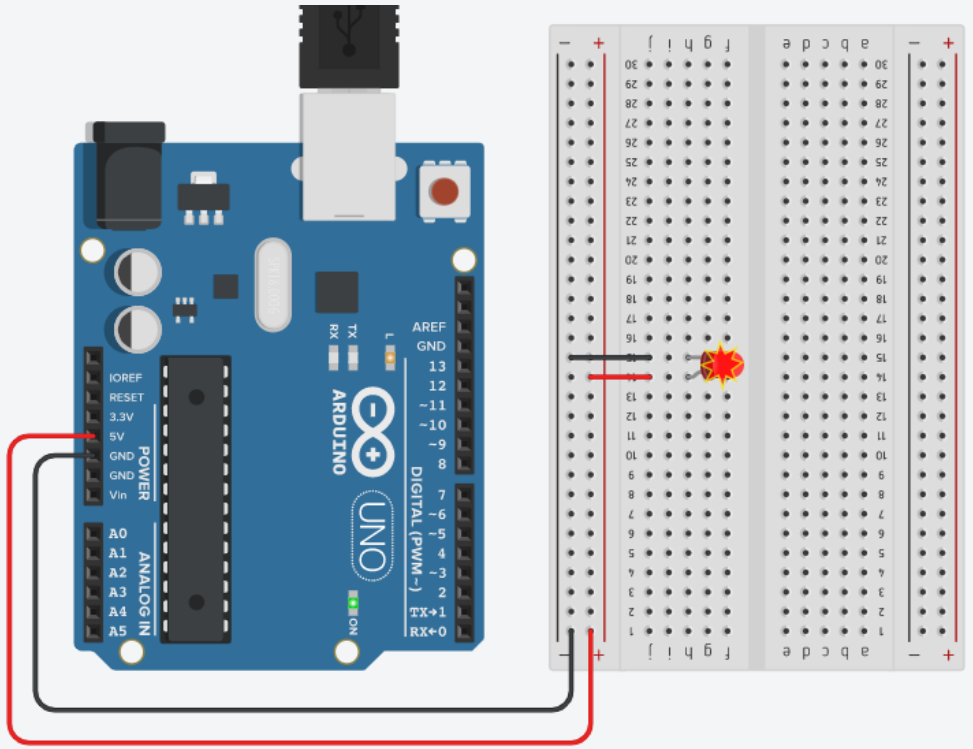
- 아두이노를 활용할 수 있는 언어를 이해하고 문법을 학습한다.
- 틴커캐드의 회원가입 및 사용방법을 익힌다.
- 시뮬레이터를 활용하여 회로 동작 원리를 확인하고 디버깅 및 오류 해결방법에 대해 알아본다.

저항

- 저항은 전류의 흐름을 방해하는 성질
- 저항의 크기는 물질의 종류나 형태에 따라 달라짐
- 전위치가 있는 두 지점이 직접 접촉했을 때, 이 접점의 저항은 0에 가까워짐에 따라 일시에 많은 양의 전류가 흐르게 되며,
- 접점에서의 소비되는 전력의 증가로 많은 열을 발생시킨다.
- 이로 인해 회로를 태우거나 손상을 주는 것을 쇼트(단락)이라고 한다.



저항없이 LED 연결시

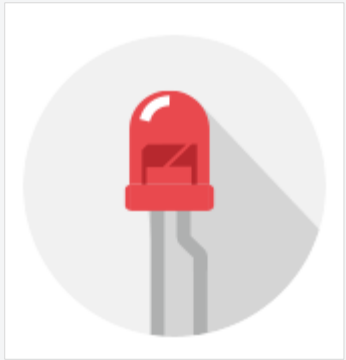


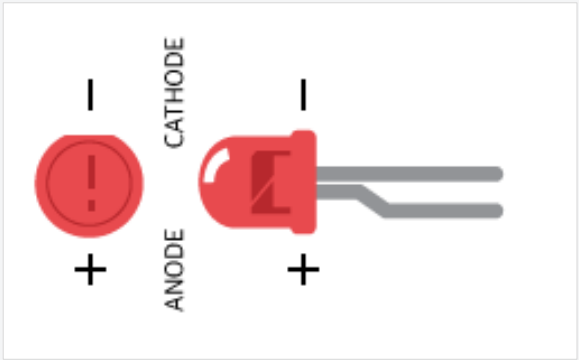
LED

- LED는 Light Emitting Diode의 약자로
  "빛을 방출하는 다이오드"
- 즉, 전기가 흐르면 빛이 나는 다이오드이다.

- LED는 재료에 따라 다양한 색을 자유롭게 구현할수 있음
- 백열전구와 달리 필라멘트를 사용하지 않으므로 외부 충격에 대해 강한 내구성과 긴 수명을 가지고 있음
- 낮은 전압으로도 밝은 빛을 얻을 수 있음
- LED는 휴대전화부터 조명기구까지 널리 사용되고 있습니다.
- LED는 단자의 극성이 정해져 있으므로 결선과 전기적 충격에 약하므로 정격전류를 넘지 않도록 주의해야 함
- 단색 LED는 일반적으로 2개의 전극단자로 구성되어 있으며, 각 단자는 극성을 가지고 있음
- 긴 단자는 애노드, 짧은 단자는 캐소드라 부른다(긴 단자에 + 전극을, 짧은 단자에는 – 전극을 연결하면 LED가 켜짐)
- LED는 일반적으로 약 2V의 전원이 필요함(2V보다 더 높은 전압을 가하면 빛의 밝기가 더 커지지만, 한계전압에 이르면 LED가 파손됨)
- LED가 동작하는 순간, 과대 전류가 흘러 LED가 파괴될수 있으므로, 전류 제한용 저항을 같이 달아 주어야 함
- 아두이노 보드의 공급전압은 5V이며, LED는 약 2V와 10mA을 소비함
- 옴의 법칙으로 저항을 계산하면 전기저항 (R) = 전압(V) / 전류의 세기(I)
- (5-2) / 0.01 = 300Ω 이 필요함
- 일반적으로 많이 사용되는 330Ω 저항을 달아주는 것이 안전함

LED의 종류



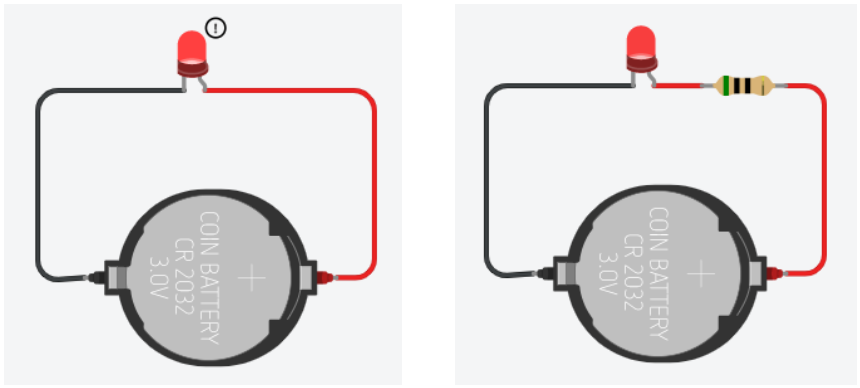


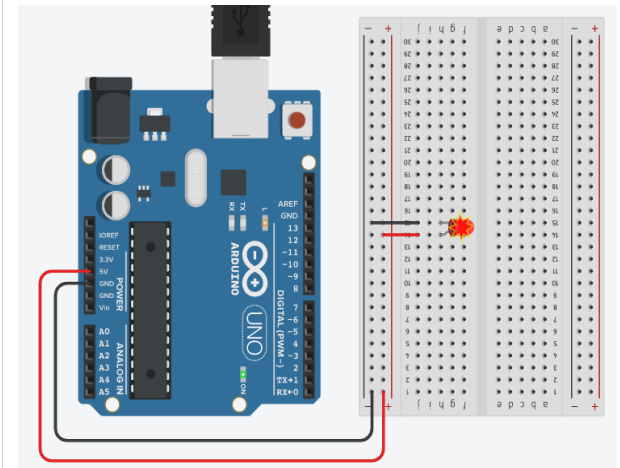
LED를 동작시켜 보자

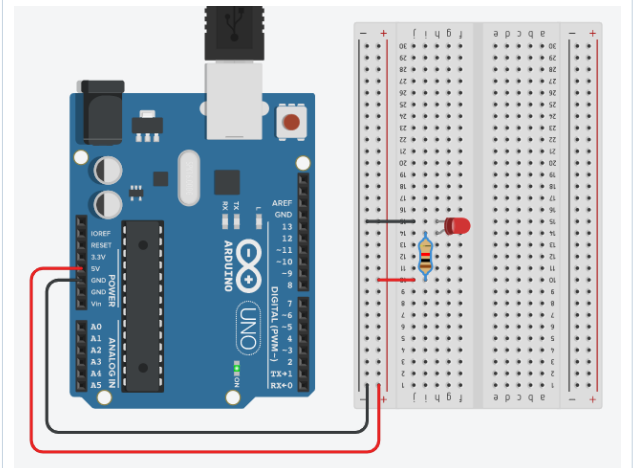
센서나 케이블을 선택하면 이름, 수량, 구성요소등 다양한 정보를 확인할 수 있다.

이름	수량	구성요소
D1	1	빨간색 LED
BAT2	1	코인 셀3V 배터리
R1	1	50Ω 레지스터

이름	수량	구성요소
D4	1	빨간색 LED
U1	1	Arduinl Uno R3
R4	1	1KΩ 레지스터





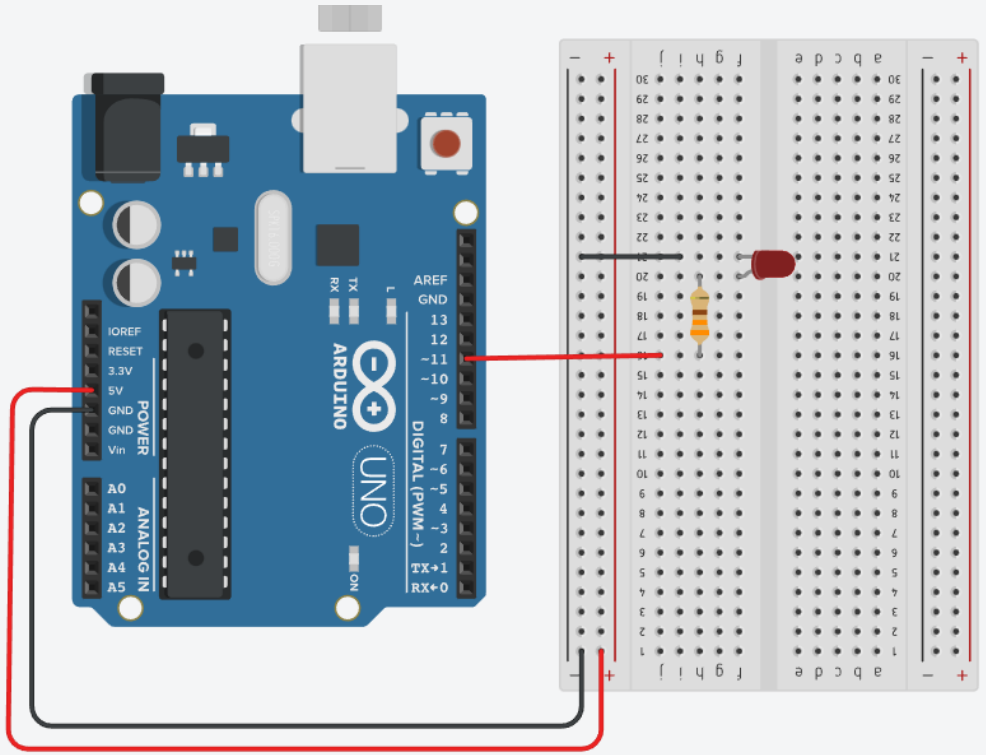


LED를 동작시켜 보자

LED의 (+)극과 (-)극을 구분하고 1초마다 깜박이는 프로그램을 작성해보자.

```
void setup() //스케치가 시작될 때 호출하는 함수
{
  pinMode(11, OUTPUT); //출력모드 11번핀 설정
}

void loop() //setup()함수 호출 이후 계속 반복되는 함수
{
  digitalWrite(11, HIGH);
  delay(1000);   // Wait for 1000 millisecond(s)
  digitalWrite(11, LOW);
  delay(1000);   // Wait for 1000 millisecond(s)
}
```

LED 회로도



01 학습정리

학습 정리

- 아두이노와 다양한 구성요소의 작동원리와 연결방법에 대한 이해
- 틴커캐드의 메뉴와 사용방법을 이해하고 LED의 ON/OFF 방법 숙지
- 코딩방법과 시뮬레이션 동작방법 실습

2차시

단색 LED와 삼색 LED의 동작

02 학습목표, 학습내용

학습 목표

- LED와 저항을 연결하는 방법에 대해 학습한다.
- PWM(펄스 폭 변조) 신호를 사용하여 실습한다.
- 단색 LED와 삼색 LED의 작동원리와 제어방법을 이해한다.

학습 내용

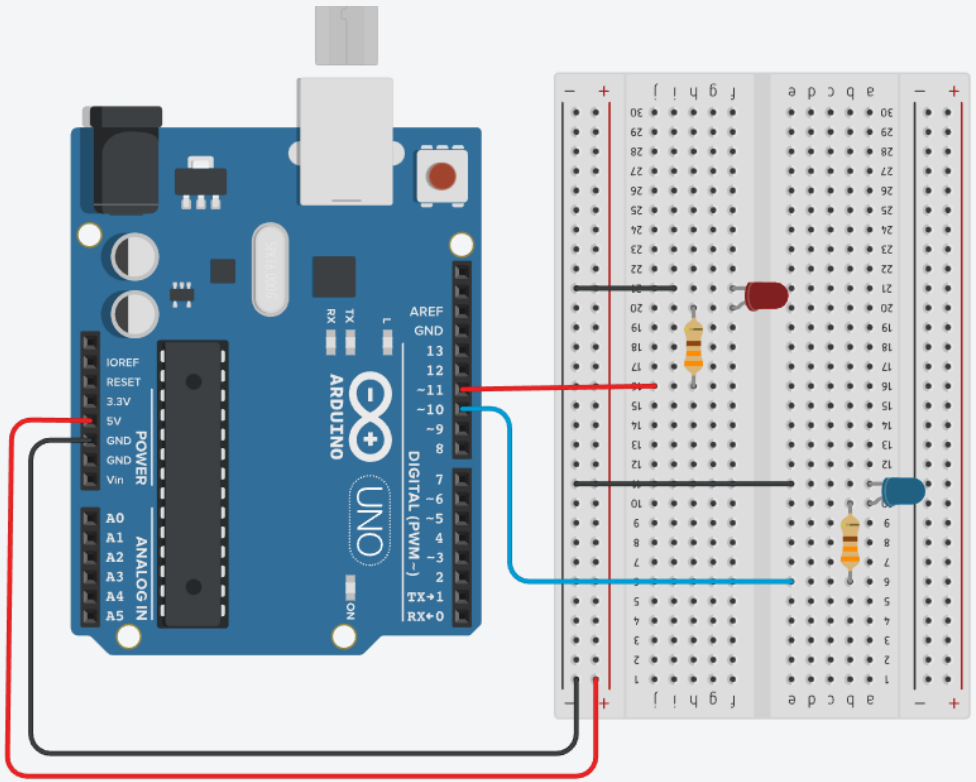
- 아날로그 신호의 PWM원리를 이해하는 실습코딩을 한다.
- RGB(빨강, 초록, 파랑) 색상 조합으로 다양한 색표현 해보기
- 컬러패턴 및 깜빡임의 변화를 프로그램 코딩으로 다양하게 조합하여 회로설계를 풍부하게 학습한다.

여러 개의 LED를 동작시켜 보자

LED를 2개 또는 3개로 늘려 여러 개의 LED가 동작하도록 하자
변수의 선언방법을 여러 개로 바꾸어보자.

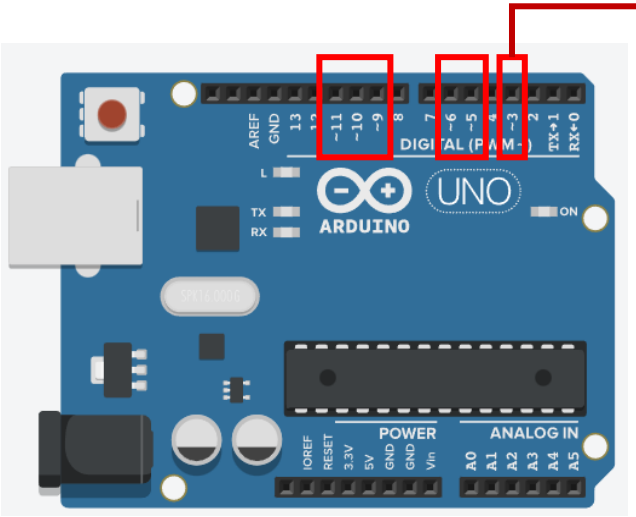
```
int LED_Pin1 = 11;
int LED_Pin2 = 10;

void setup()
{
  pinMode(LED_Pin1, OUTPUT);
  pinMode(LED_Pin2, OUTPUT);
}

void loop()
{
  digitalWrite(LED_Pin1, HIGH);
  digitalWrite(LED_Pin2, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(LED_Pin1, LOW);
  digitalWrite(LED_Pin2, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

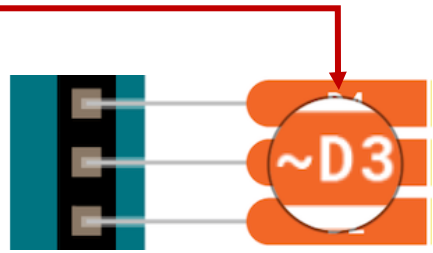
여러 개의 LED 회로도

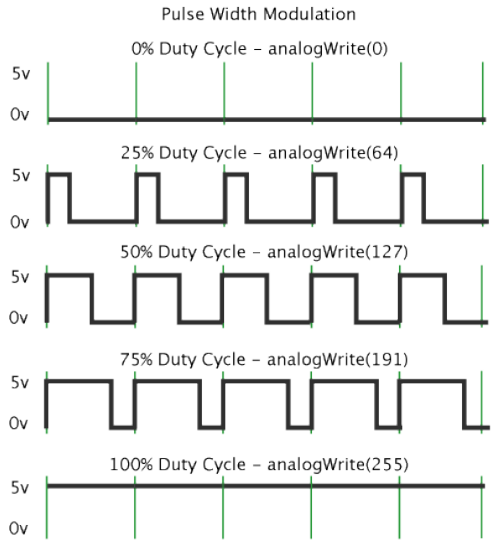


LED를 제어해보자 – PWM제어

아두이노는 아날로그 값 출력이 가능하다. 모터의 회전속도, LED의 밝기 등을 조절할 수 있어서 편리하다.







출처 : arduion.cc

- PWM(Pulse Width Modulation)은 디지털 방식으로 아날로그 결과를 얻는 기술이다.
- LED로 온-오프 패턴을 충분히 빠르게 반복하면 신호가 LED의 밝기를 제어하는 0과 Vcc 사이의 일정한 전압인 것처럼 나타난다.
- anlalogWrite(255) 의 밝기값을 analogWrite(127)로 50% 낮출 수 있다.

- PWM핀은 물결기호(~)로 표시된다.

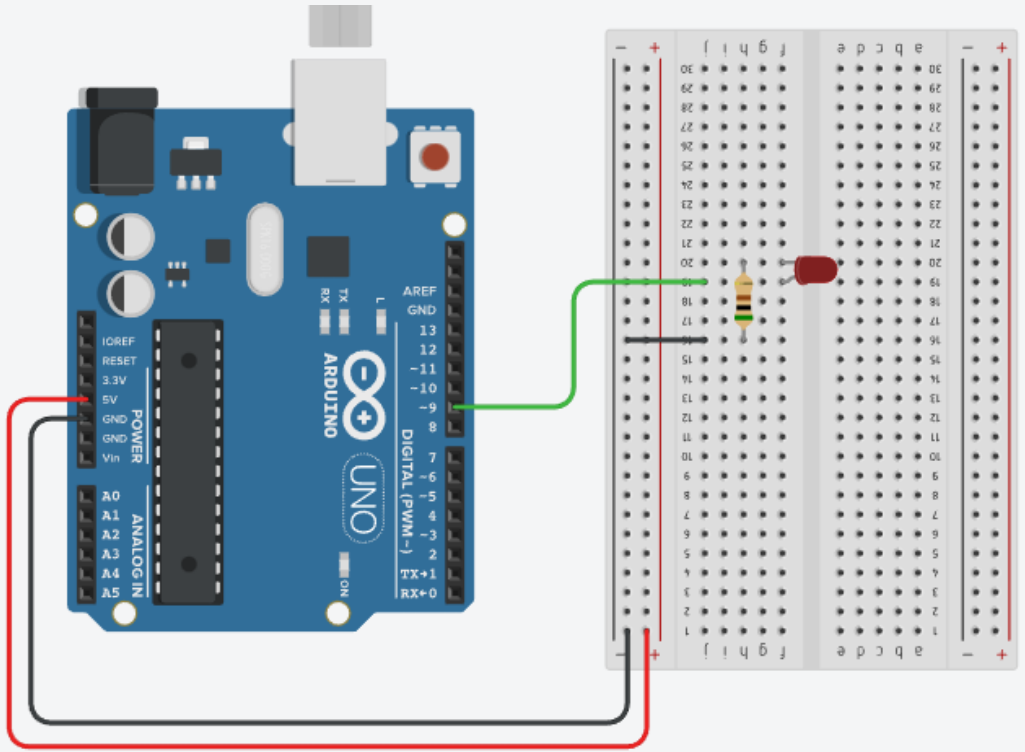
점점 밝아지는 LED를 만들어보자

analogWrite 함수를 사용하여 점점 밝아졌다가 꺼지는 프로그램을 작성해보자. 물결기호 (~)가 있는 핀번호에 연결해야한다.

```
int PWM_LED = 9;
int i = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(PWM_LED, OUTPUT);
}
```

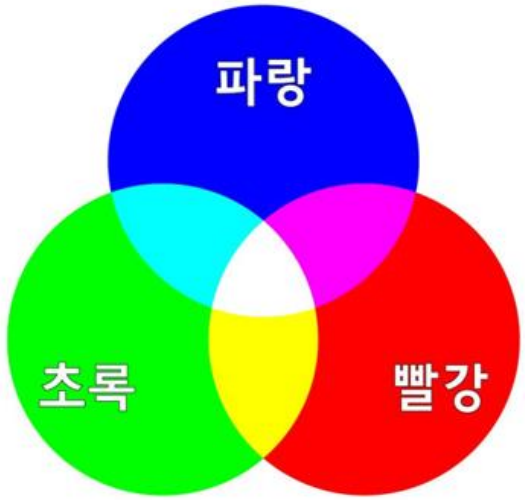
```
void loop()
{
  Serial.println("PRINT  i");
  for (i = 0; i <= 255; i += 10)
   {
    analogWrite(PWM_LED, i);
    Serial.println(i);
    delay(200);
// Wait for 200 millisecond(s)
    }
  digitalWrite(9, LOW);
  delay(500);
}
```

밝기가 밝아지는 LED 출력제어



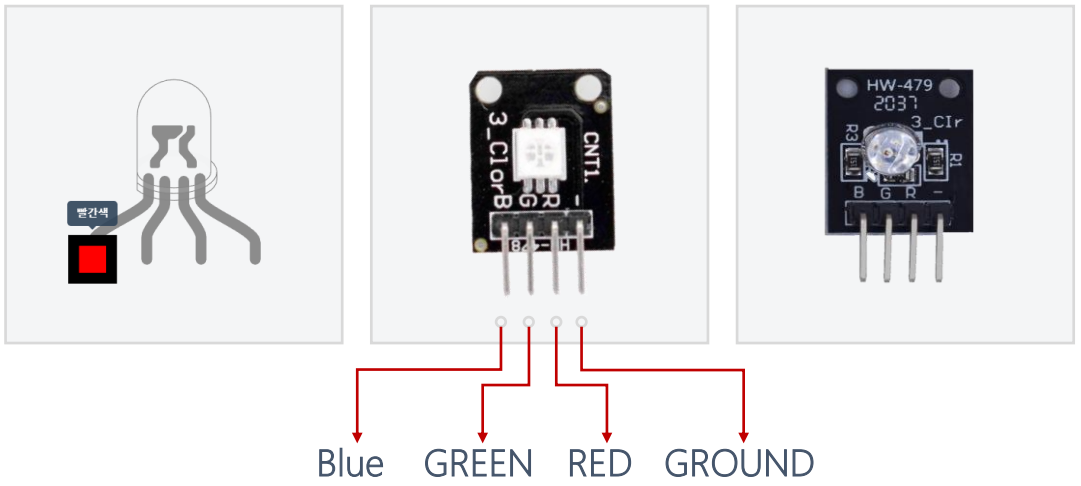
삼색 LED를 제어해 보자

빨간색, 초록색, 파란색을 빛의 삼원색이라하고, 세가지의 색을 적절히 조절하면 다양한 색을 만들어낼수 있다.
파랑+초록+빨강의 합성색은 흰색이다.



[빛의 삼원색]

RGB 삼색 LED는 제품과 종류에 따라서 색상핀 순서가 다르다.
그러므로, 항상 핀에 적혀있는 음극, R, G, B 글씨를 확인해야 한다.



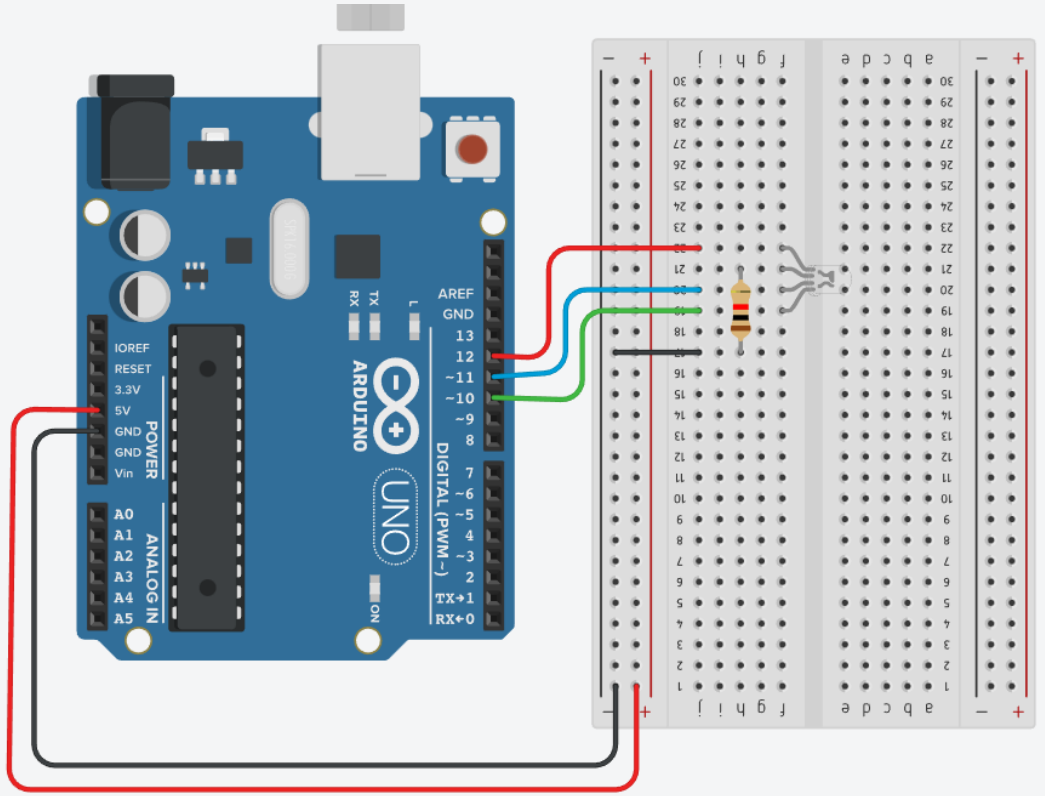
삼색 LED를 제어해 보자

삼색 LED 핀과 음극핀, 저항을 사용하여 다양한 색상의 LED를 켜보자

```
int R_LED = 12;
int G_LED = 10;
int B_LED = 11;

void setup()
{
  pinMode(R_LED, OUTPUT);
  pinMode(G_LED, OUTPUT);
  pinMode(B_LED, OUTPUT);
}
void loop()
{
  digitalWrite(R_LED,HIGH);
  digitalWrite(B_LED,LOW);
  digitalWrite(G_LED,LOW);
  delay(1000);
```

```
  digitalWrite(R_LED,LOW);
  digitalWrite(B_LED,HIGH);
  digitalWrite(G_LED,LOW);
  delay(1000);
  digitalWrite(R_LED,LOW);
  digitalWrite(B_LED,LOW);
  digitalWrite(G_LED,HIGH);
  delay(1000);
  digitalWrite(R_LED,HIGH);
  digitalWrite(B_LED,HIGH);
  digitalWrite(G_LED,LOW);
  delay(1000);
}
```

삼색 LED 회로도



학습 정리

- 여러 개의 LED를 연결하고 동시에 제어하는 방법에 대해 실습
- PWM 신호를 사용하여 LED의 밝기 실습
- 삼색LED로 RGB 컬러조합의 코딩 실습

3차시

푸쉬버튼과 슬라이드 버튼을 활용한 LED 제어

03 학습목표, 학습내용

학습 목표

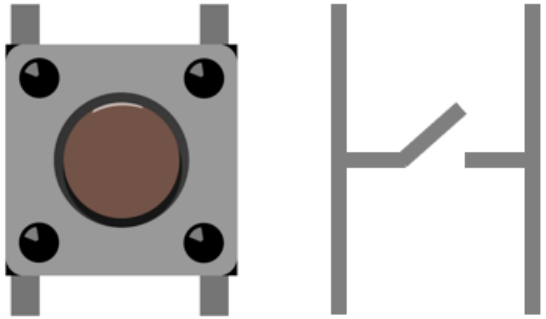
- 푸쉬버튼의 동작을 감지하고 삼색 LED를 제어하는 프로그래밍을 실습한다.
- 버튼을 여러 개 사용하여 버튼의 상태 변화에 RGB 조합 제어하기
- 슬라이드버튼과 푸쉬버튼과의 차이점을 이해하고 제어

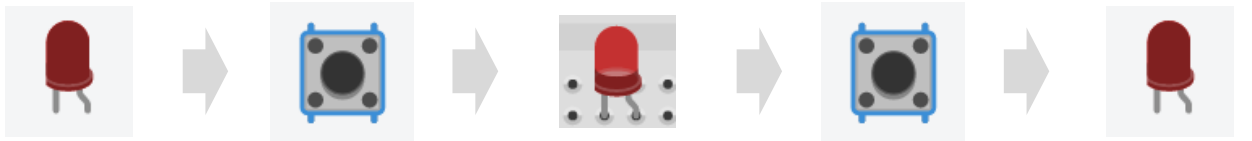
학습 내용

- 푸쉬버튼과 LED사이의 동작을 제어하고 프로그래밍 실습 한다.
- 슬라이드 버튼의 회로 제어 연결방법에 대해 알아본다.
- 시리얼모니터로 슬라이디 버튼 상태를 확인하여 코딩한다.

버튼을 활용하여 제어해보자

버튼은 스위치를 동작시키는 역할을 하는데, 4개의 단자는 회로와 스위치를 연결하는 다리역할을 한다.
회로를 연결하고 버튼을 누르면 전류가 흐르게 된다.





버튼은 누르고 있어야 회로가 연결되어 LED에 불이 켜지고, 손을 버튼에서 떼었을때 LED가 꺼지는 것을 확인할 수가 있다.

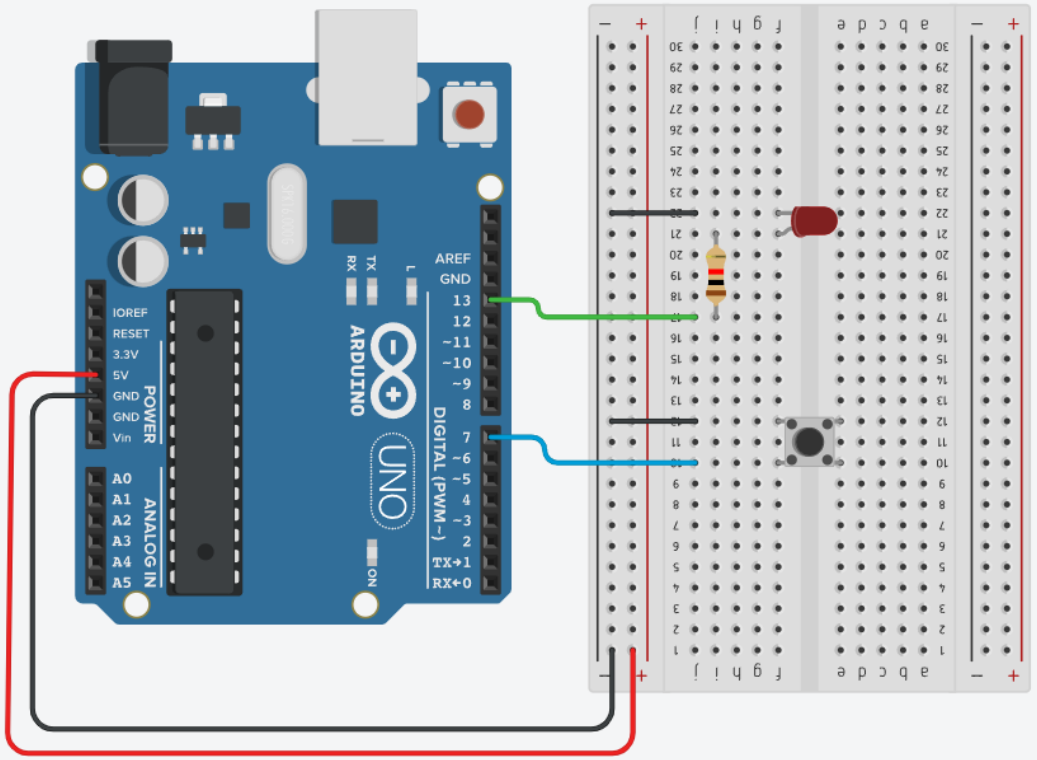
해당하는 값은 `Serial.println()`함수를 통하여 시리얼 모니터에서 확인 가능하다.

버튼을 활용하여 제어해보자

LED의 (+)극과 (-)극을 구분하고 1초마다 깜박이는 프로그램을 작성해보자.

```
int LED_pin = 13;
int Button = 7;

void setup( )
{
  pinMode(LED_pin, OUTPUT);
  pinMode(Button, INPUT_PULLUP);
}

void loop( )
{
   bool button = digitalRead(Button);

  if (button == true)
  {
    digitalWrite(LED_pin, LOW);
  }
  else
  {
    digitalWrite(LED_pin, HIGH);
  }
}
```

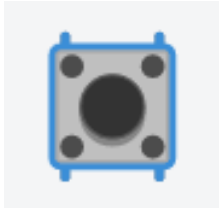
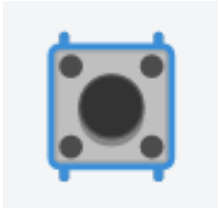
버튼으로 LED를 켜는 회로도



버튼을 활용하여 삼색 LED를 제어해보자

3개의 버튼을 활용하여 빨강, 초록, 파랑의 LED가 켜지도록 제어해보자.





RED 버튼 제어

GREEN 버튼 제어

BLUE 버튼 제어

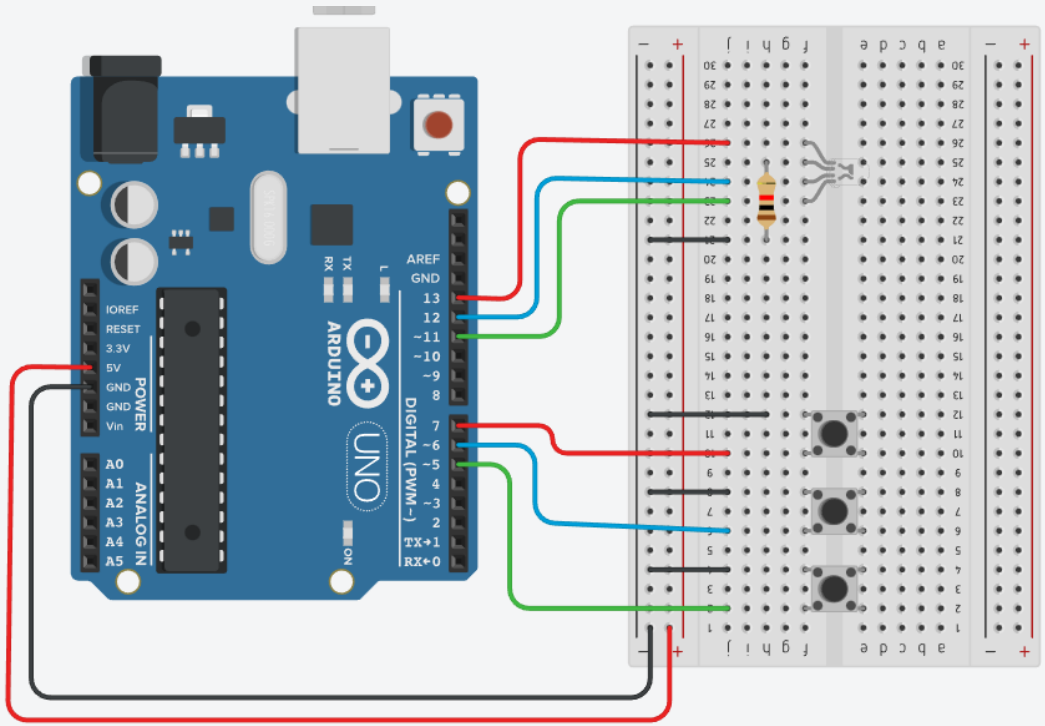
버튼을 활용하여 삼색 LED를 제어해보자

LED의 (+)극과 (-)극을 구분하고 1초마다 깜박이는 프로그램을 작성해보자.

```
int R_LED = 13;
int G_LED = 11;
int B_LED = 12;

int R_Button = 7;
int G_Button = 5;
int B_Button = 6;


void setup()
{
  pinMode(R_LED, OUTPUT);
  pinMode(G_LED, OUTPUT);
  pinMode(B_LED, OUTPUT);

  pinMode(R_Button, INPUT_PULLUP);
  pinMode(G_Button, INPUT_PULLUP);
  pinMode(B_Button, INPUT_PULLUP);
}
```

```
void loop()
{

  if(digitalRead(R_Button) == false)
  {
    digitalWrite(R_LED, HIGH);
  }
  else if(digitalRead(G_Button) == false)
  {
    digitalWrite(G_LED, HIGH);
  }
  else if(digitalRead(B_Button) == false)
  {
    digitalWrite(B_LED, HIGH);
  }
  else
  {
    digitalWrite(R_LED, LOW);
    digitalWrite(G_LED, LOW);
    digitalWrite(B_LED, LOW);
  }
}
```

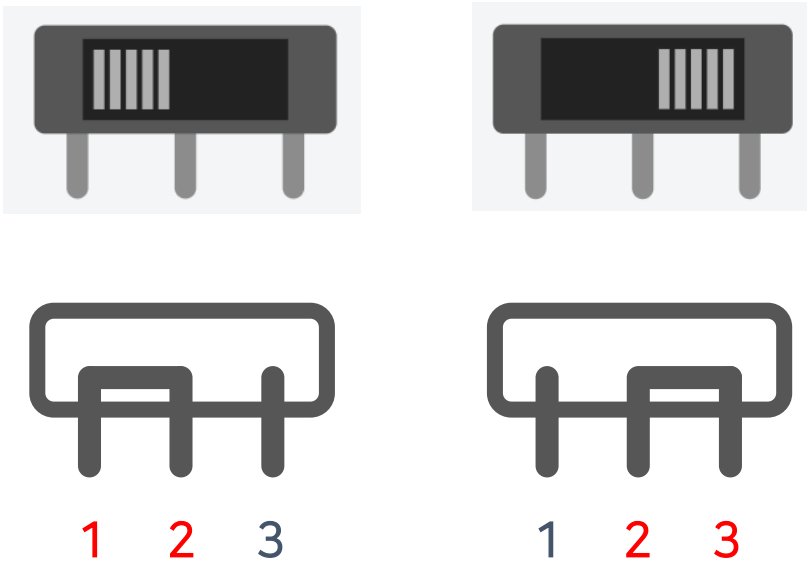
버튼 3개로 삼색 LED 켜는 회로도

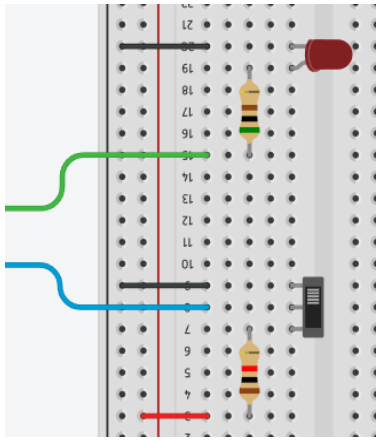


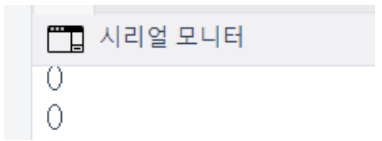
슬라이드 스위치

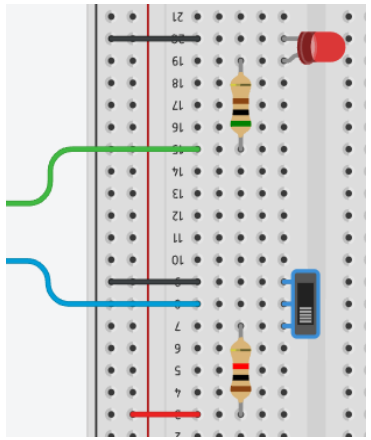
푸쉬버튼은 누르는 동안 상태가 변경되지만, 슬라이드 스위치는 상태를 변경하면 해당 상태를 계속 유지하는 특징이 있다.
슬라이드 스위치로 LED를 켜는 프로그램을 작성해보자.

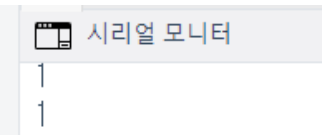
- 슬라이드 스위치는 3개의 단자가 존재한다.
- 스위치가 왼쪽에 있으면 1,2번이 연결되고, 오른쪽으로 이동시키면 2,3번이 연결된다.









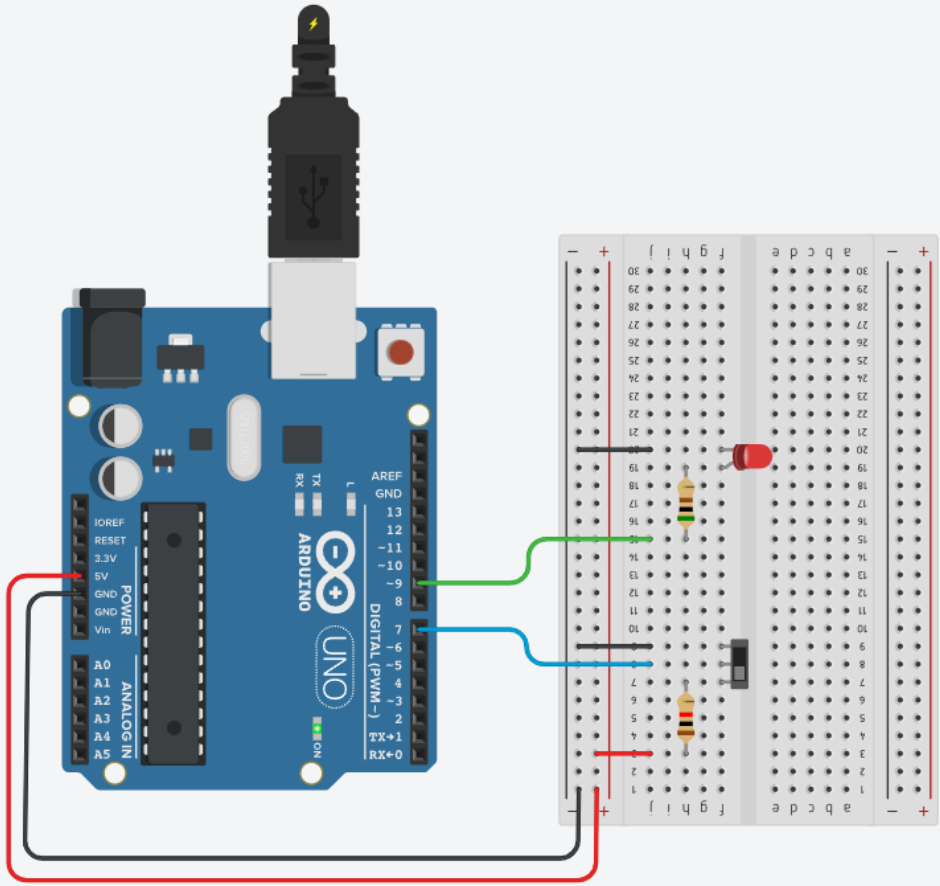


슬라이드 스위치 - LED켜기

슬라이드 스위치의 위치를 바꾸어 LED를 ON/OFF 동작하는 프로그램을 작성해보자.

```
int LED = 9;
int Switch = 7;
int val;

void setup()  {
  Serial.begin(9600);

  pinMode(LED, OUTPUT);
  pinMode(Switch, INPUT);
}

void loop()
{
  val = digitalRead(Switch);
  Serial.println(val);

  if (val == HIGH)    {
    digitalWrite(LED, HIGH);
  }
  else  {
    digitalWrite(LED, LOW);
  }
}
```

버튼 피에조 스피커



03 학습정리

학습 정리

- 푸쉬버튼의 풀업 상태를 확인하여 LED를 제어하는 실습
- 여러 개의 푸쉬버튼의 사용으로 삼색 LED를 변화값을 제어
- 푸쉬버튼의 제어값과 슬라이드 버튼의 제어값을 시리얼 모니터에서 확인

4차시

피에조 부저로 멜로디 생성과 조도센서로 빛값을 측정

04 학습목표, 학습내용

학습 목표

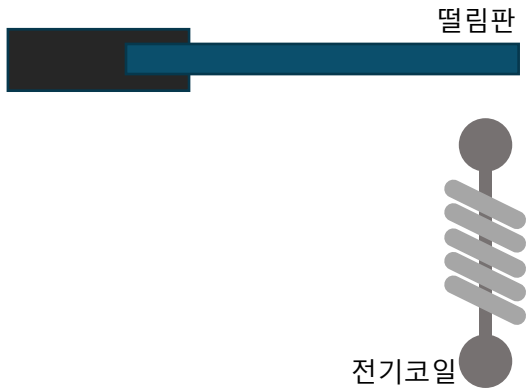
- 버튼을 활용한 멜로디 재생 및 제어방법을 익힌다.
- 피에조 부저의 멜로디 주파수를 이해한다.
- 조도센서의 조도값을 확인하여 밝기 제어 방법을 습득한다.
- 조도값에 따른 자동 LED 센서의 원리를 이해한다.

학습 내용

- 피에조 스피커의 작동원리와 제어방법을 이해한다.
- 버튼의 동작 감지를 통해 피에조 부저에서 멜로디를 재생 구현한다.
- 조도센서의 값을시리얼 모니터로 확인하고, 작동방법 및 회로도를 구성 실습한다.

소리를 만들어 보자

Buzzer는 전기적으로 연결된 전기 코일이나 전자식 스피커로 구성되어 있다. 작은 떨림판이나 진동판이 부착되어 있어 이 진동으로 인해 소리가 발생한다.



능동 버저(Active Buzzer)

수동 버저(Passive Buzzer)

1. 능동 부저(Active Buzzer): 능동 부저는 내장된 발진기로서, 전기 신호를 입력받으면 스스로 소리를 발생시킬 수 있는 부저입니다.
   전기적인 신호만 입력하면 부저 스스로가 진동하여 소리를 생성합니다. 따라서 아두이노와 같은 마이크로컨트롤러에 직접 연결하여 사용할 수 있습니다.
   주로 디지털 핀에 연결되며, 디지털 핀의 상태에 따라 소리가 발생하거나 중단됩니다.

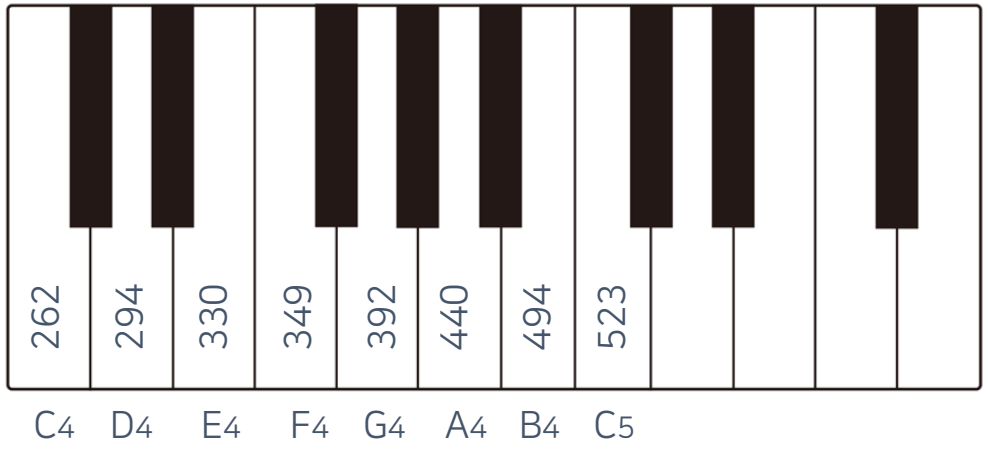
2. 수동 부저(Passive Buzzer): 수동 부저는 발진기가 내장되어 있지 않으며, 외부적인 진동 원리를 활용하여 소리를 생성하는 부저입니다.
   수동 부저는 단순히 진동하는 요소로 구성되어 있으며, 외부적인 주파수 신호를 필요로 합니다.
   따라서 수동 부저를 사용하기 위해서는 외부적인 신호 발생기(예: PWM 신호)를 사용하여 진동을 제어해야 합니다.

소리를 만들어 보자

피에조(Piezo Buzzer)는 그리스어의 "누른다"의 뜻을 가지고 있다.
특히, 아두이노 수동부저는 Tone() 또는 noTone() 함수를 사용한다.

옥타브 및 음계 표준 주파수

옥타브 / 음계	1	2	3	4	5	6	7	8
C(도)	32.7032	65.4064	130.8128	261.6256	523.2511	1046.602	2093.005	4186.009
C#	34.6478	68.2957	138.5913	277.1826	554.3653	1108.731	2217.461	4434.922
D(레)	36.7081	73.4162	146.8324	293.668	587.3295	1174.659	2349.318	4698.646
D#	38.8909	77.7817	155.5635	311.1270	622.2540	1244.508	2489.016	4978.032
E(미)	41.2034	82.4069	164.8138	329.6276	659.2551	1318.510	2637.020	5274.041
F	43.6535	87.3071	174.6141	349.2283	698.4565	1396.913	2793.826	5587.652
F#	46.2493	92.4986	184.9972	369.9942	739.9888	1479.978	2959.955	5919.911
G(솔)	48.9994	97.9989	195.9977	391.9954	783.9909	1567.982	3135.963	6271.927
G#	51.9130	103.8262	207.6523	415.3047	830.6094	1661.219	3322.438	6644.875
A(라)	55.0000	110.0000	220.0000	330.0000	880.0000	1760..000	3520.000	7040.000
A#	58.2705	116.5409	233.0819	466.1638	932.3275	1864.655	3729.310	7040.000
B(시)	61.7354	123.4708	246.9417	493.8833	987.7666	1975.533	3951.066	7902.133

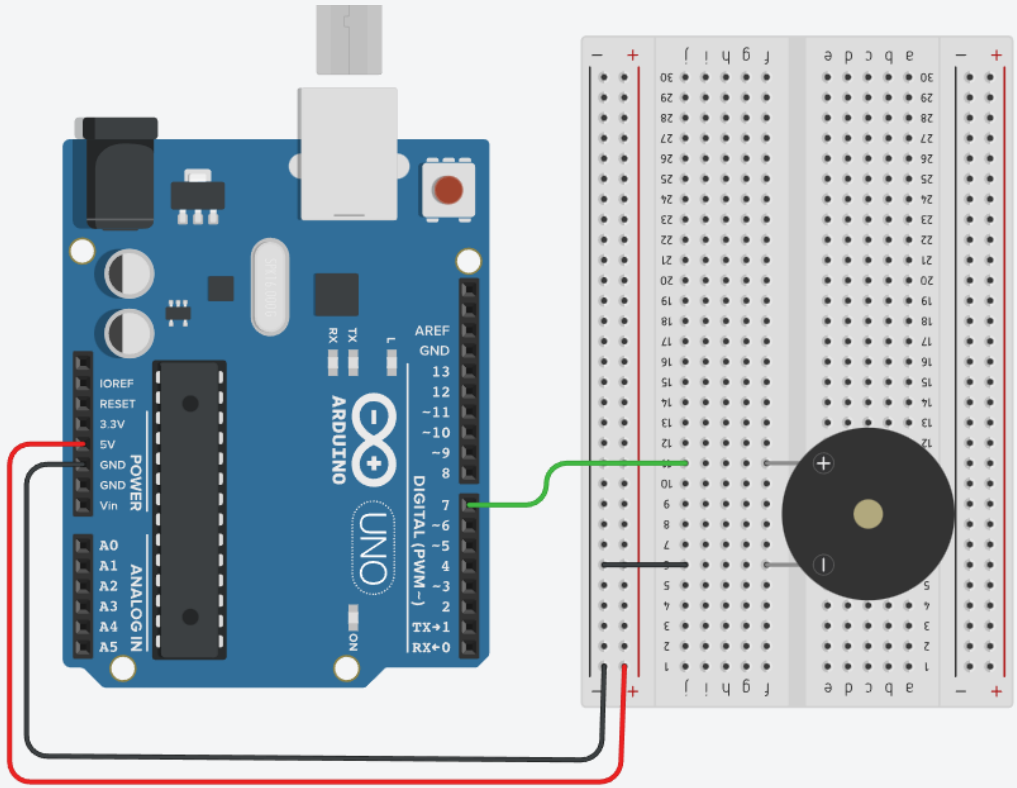


피에조 스피커 제어

피에조 부저의 (+)극을 7번핀으로 제어해보자.

```
int buzzer=7;

// 음계들의 헤르츠 값을 배열로 선언하기
int melody[] = {262,294,330,349,392,440,494,523};

void setup()
{
  for (int i=0; i<8; i++)
  {
   //피에조 스피커로 소리를 내는 함수 (핀번호, 헤르츠, 재생시간)
    tone (buzzer, melody[i],200);
    delay(500);
    noTone(buzzer);    //피에조 스피커의 소리를 끄는 함수 (핀번호)
  }

}

void loop()
{

}
```

피에조 스피커 제어



버튼으로 피에조 스피커 만들기

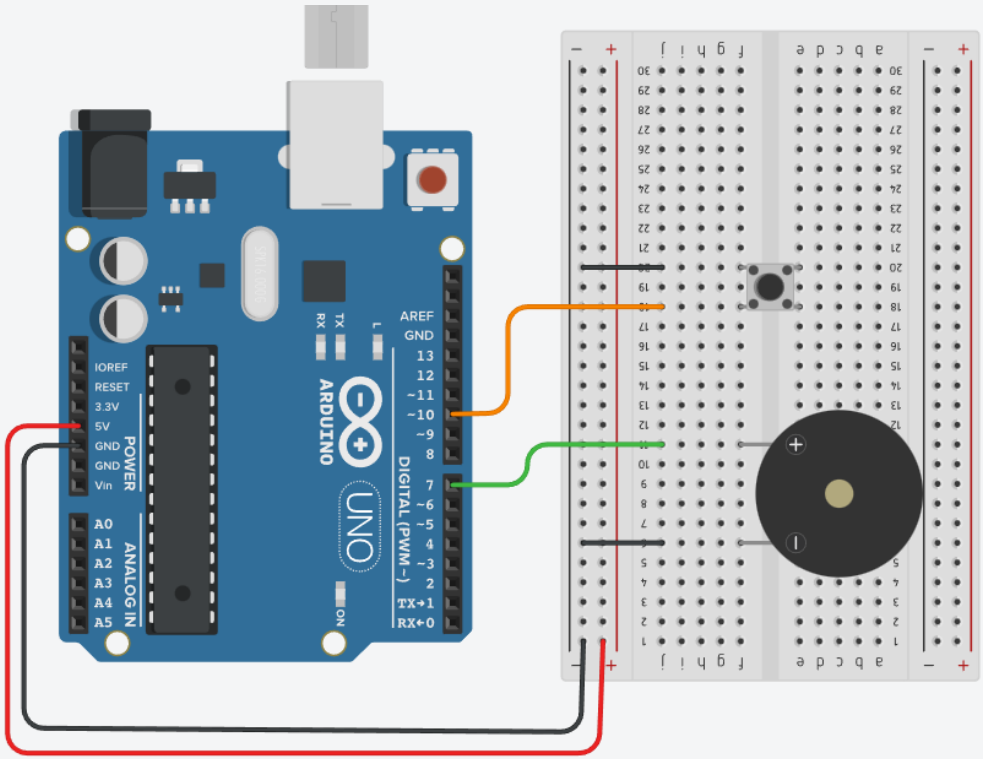
버튼을 누르면 음계를 하나씩 소리내게 하는 스피커를 만들어보자.

```
int buzzer=7;
int Button =10;
int melody[] = {262,294,330,349,392,440,494,523};
              // 음계들의 헤르츠 값을 배열로 선언하기
```

```
void setup()
{
   pinMode(Button, INPUT_PULLUP);
}
```

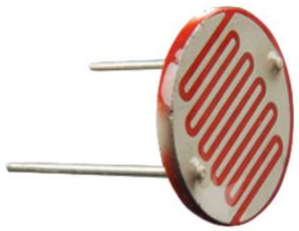
```
void loop()
{
  if(digitalRead(Button) == false)
   {
     for (int i = 0; i < 8; i++)
       {
         tone(buzzer, melody[i],250);
         delay(500);
         noTone(buzzer);
        }
     }
}
```

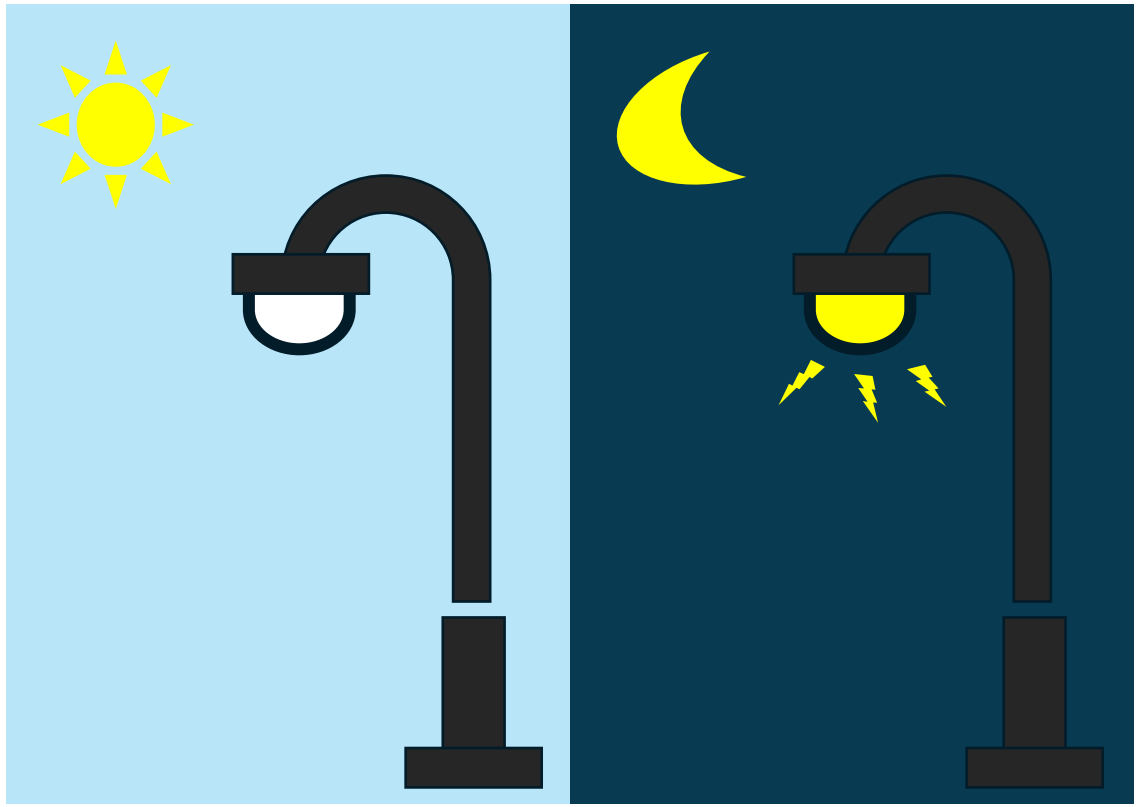
버튼 피에조 스피커



조도(광원센서를) 이용해 보자

조도센서(Photo Resistor)는 주변의 밝기 상황에 따라 변화하여 측정하는 센서이다.
빛을 받으면 내부 전자가 발생하여 전도율이 변하는 소자를 사용하고, 황화카드뮴(Cds)를 사용한 경우 Cds 센서라고도 한다.

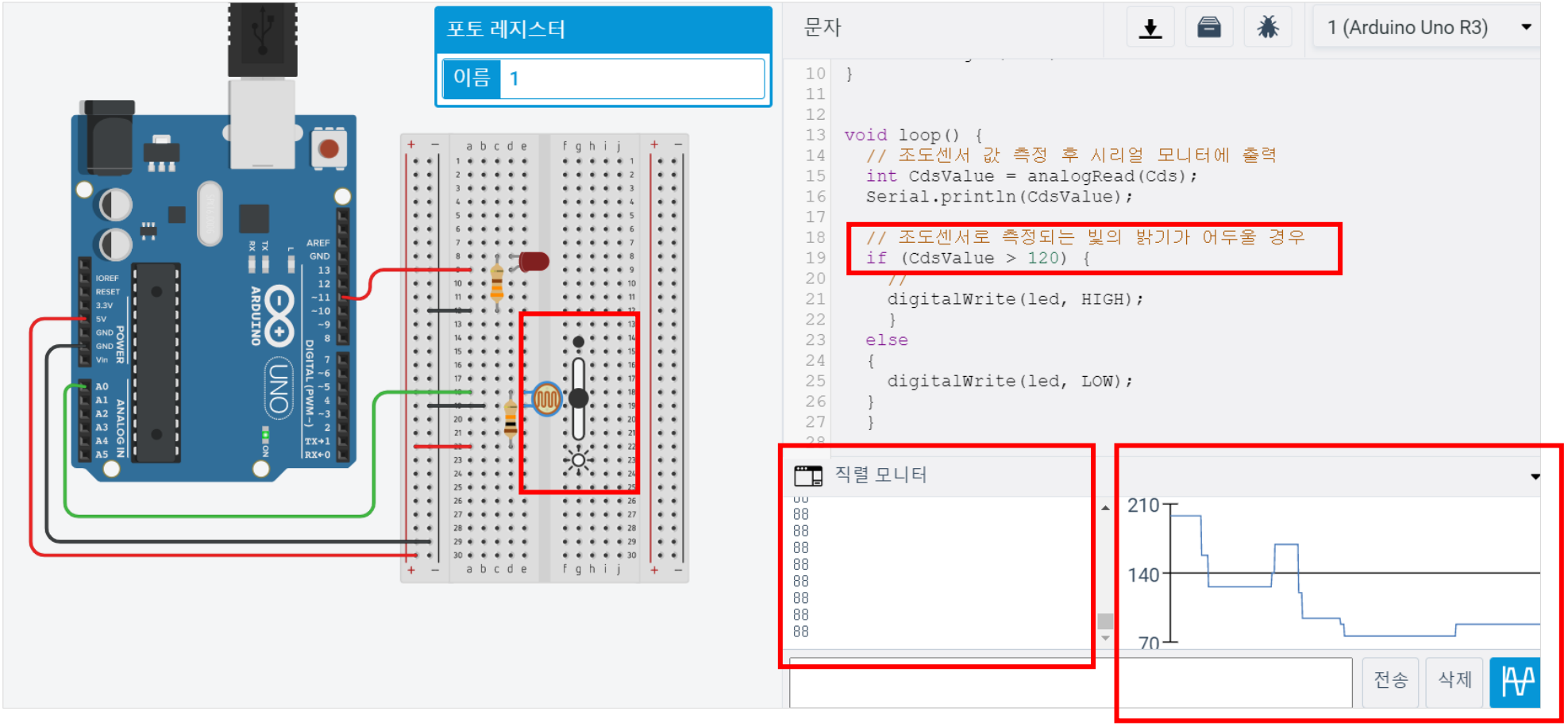




- 어두워지면 자동으로 켜지는 가로등이나, 자동차의 헤드라이트, 밝기에 따라 변하는 핸픈폰 액정 등 실생활에서도 다양하게 찾아볼수 있다.
- 조도센서는 극성이 없다.
- 저항을 같이 사용하여 회로도를 작성한다.

틴커캐드에서 조도센서 사용하는 방법

틴커캐드에서 조도센서를 사용하려면 조도센서를 클릭하여 밝고 어둡게 만들수 있다.
또한 소스코드 아래 쪽으로 시리얼 모니터와 그래프 모니터도 확인할 수 있다.

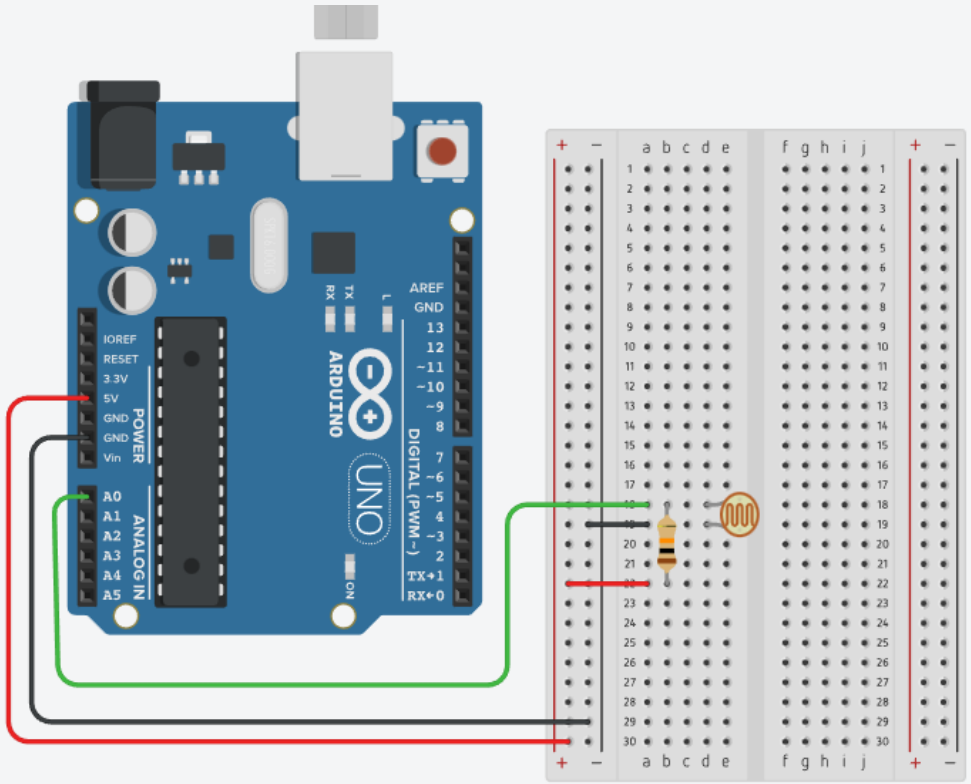


조도센서의 값을 알아보자

조도센서를 연결하고, 시리얼 프린터를 사용하는 함수를 작성해보자.

```
 int Cds = A0;        // 조도센서를 A0번에 연결

void setup()
{
  Serial.begin(9600);
}

void loop() {
  // 조도센서 값 측정 후 시리얼 모니터에 출력
  int CdsValue = analogRead(Cds);
  Serial.println(CdsValue);
  }
```

조도 센서 측정-시리얼 프린트 해보기



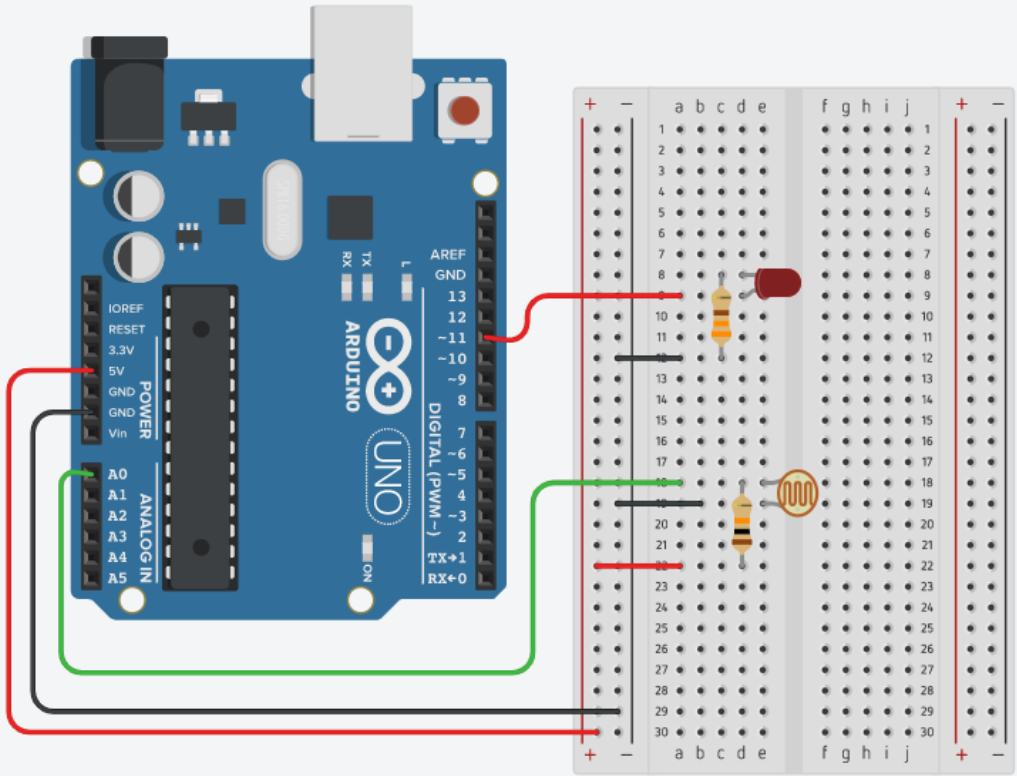
조도센서의 값에 따라 LED를 켜보자

어두워지면 LED를 켜고, 밝으면 LED를 끄는 조건문을 활용하면 자동으로 켜지고 꺼지는 가로등을 만들어 볼 수 있다.

```
int led = 11;
// LED를 11번 디지털 핀에 연결
int Cds = A0;
// 조도센서를 A0번에 연결

void setup()
{
  pinMode(led,OUTPUT);
  Serial.begin(9600);
}
```

```
void loop()
{
// 조도센서 값 측정 후 시리얼 모니터에 출력
  int CdsValue = analogRead(Cds);
  Serial.println(CdsValue);

// 조도센서로 측정되는 빛의 밝기가 어두울 경우
  if (CdsValue > 800)
   {
    digitalWrite(led, HIGH);
    }
  else
   {
    digitalWrite(led, LOW);
    }
 }
```

자동으로 켜지는 가로등 만들어보기



조도센서의 값에 따라 LED가 켜질때 소리를 내보자

LED의 (+)극과 (-)극을 구분하고 1초마다 깜박이는 프로그램을 작성해보자.

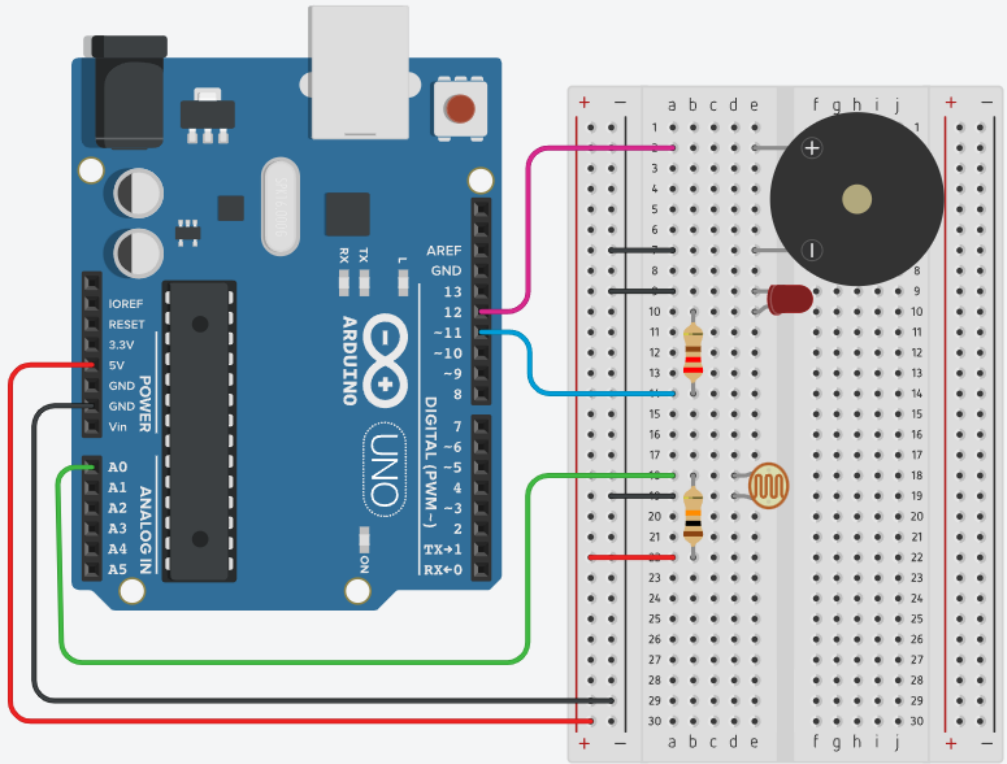
```
int led = 11;
// LED를 11번 디지털 핀에 연결
int Cds = A0;
// 조도센서를 A0번에 연결
```

```
int led = 11;
// LED를 11번 디지털 핀에 연결
int Cds = A0;
// 조도센서를 A0번에 연결
int buzzer=12;
// Buzzer를 12번에 연결
```

```
// 조도센서로 측정되는 빛의 밝기가 어두울 경우
  if (CdsValue > 800)
   {
    digitalWrite(led, HIGH);
    }
  else
   {
    digitalWrite(led, LOW);
    }
 }
```

```
// 조도센서로 측정되는 빛의 밝기가 어두울 경우
  if (CdsValue > 800)
   {
    digitalWrite(led, HIGH);
    tone(buzzer, 330, 250);
    delay(500);
    noTone(buzzer);
    }
  else
   {
    digitalWrite(led, LOW);
    }
 }
```

자동으로 켜지는 가로등



04 학습정리

학습 정리

- 피에조 스피커의 동작원리와 제어방법에 대해 이해
- 버튼동작으로 멜로디 연주를 실습
- 조도센서의 값을 측정하는 방법에 대해 이해하고 자동 조명 시스템의 원리를 이해

5차시

초음파센서를 동작해보자. 초음파센서로 LED 켜기

05 학습목표, 학습내용

학습 목표

- 초음파센서로 거리 측정하기
- 아두이노와 초음파센서 연결 및 코드 작성하기
- 거리에 따른 LED 제어 방법 실습하기
- 초음파센서로 LED 켜기

학습 내용

- 초음파센서의 작동 원리와 거리 측정 방법을 이해한다.
- 초음파센서를 아두이노와 연결하고 코드를 작성하여 거리 측정한다.
- 측정한 거리 값을 기반으로 LED를 제어하는 방법을 익힌다.

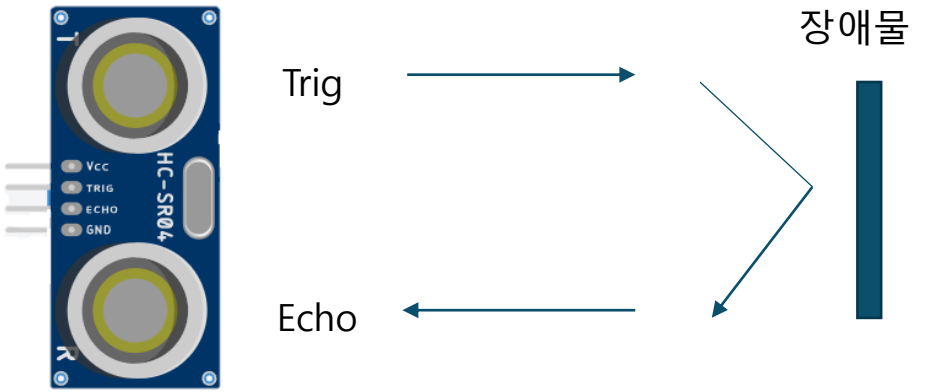
초음파 센서를 이용해보자

초음파센서의 속도는 일반 공기중에서 약 340m/s이다.
초음파 센서는 이미 로봇 청소기 같은 가전제품에서 부터 자동차의 감지센서, 초음파 탐지기 등 의료용, 산업용에 이르기까지 매우 다양한 분야에서 활용되고 있음

```cpp
// 시간을 거리로 변환 (cm 단위)
// 음속(343m/s)로 나누고, 왕복 거리를 고려하여 2로 나눕니다.
float distance = duration * 0.0343 / 2;
```

여기서 사용되는 공식은 다음과 같습니다:
거리 = (음속 × 시간) / 2
•음속: 소리의 속도를 나타내며, 대기 온도에 따라 약 343m/s로 가정합니다.
•시간: 초음파가 발사된 후 에코가 다시 수신되기까지 걸린 시간을 나타냅니다.
•왕복 거리를 고려하기 위해 2로 나눕니다. 왕복 거리는 초음파가 대상물에 부딪혀 돌아온 거리를 의미합니다.
즉, 측정된 시간을 음속으로 곱한 후 2로 나누면 거리가 됩니다. 이렇게 계산된 거리는 센티미터(cm) 단위로 표시됩니다.

예를 들어, 측정된 시간(duration)이 1000 마이크로초(1 밀리초)라고 가정하면:
거리 = (343 m/s × 0.001 s) / 2 = 0.1715 m = 17.15 cm
따라서, 이 코드에서 계산된 거리는 약 17.15 cm가 됩니다. 이 값은 측정된 대상과의 거리를 나타내게 됩니다.

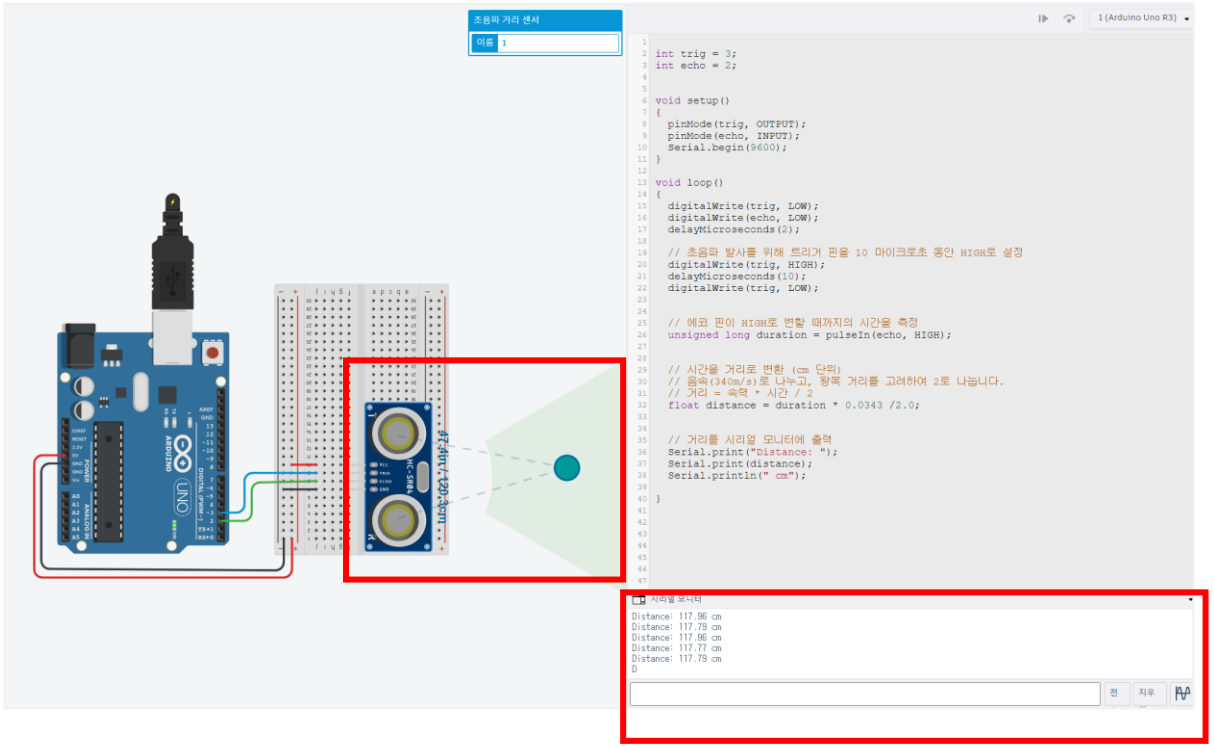


Trig : HIGH가 되면 초음파를 발사합니다.

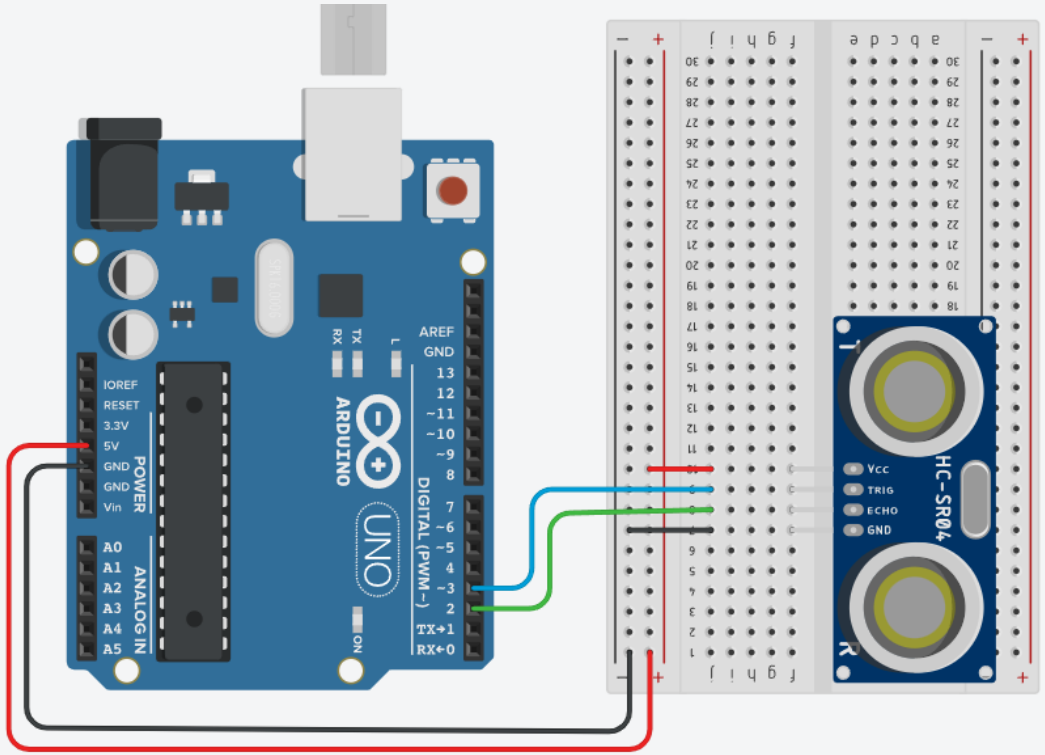
Echo : 튕겨나오는 초음파를 받아들이면 값이 HIGH가 됩니다.

초음파센서로 거리 측정해보기

초음파센서에 장애물이 감지되었을때 거리를 측정하고 계산하는 프로그램을 작성해보자. 시리얼 모니터로 거리값 출력을 해보자



초음파센서로 거리를 측정하기



초음파센서로 거리 측정해보기

초음파센서에 장애물이 감지되었을때 거리를 측정하고 계산하는 프로그램을 작성해보자. 시리얼 모니터로 거리값 출력을 해보자

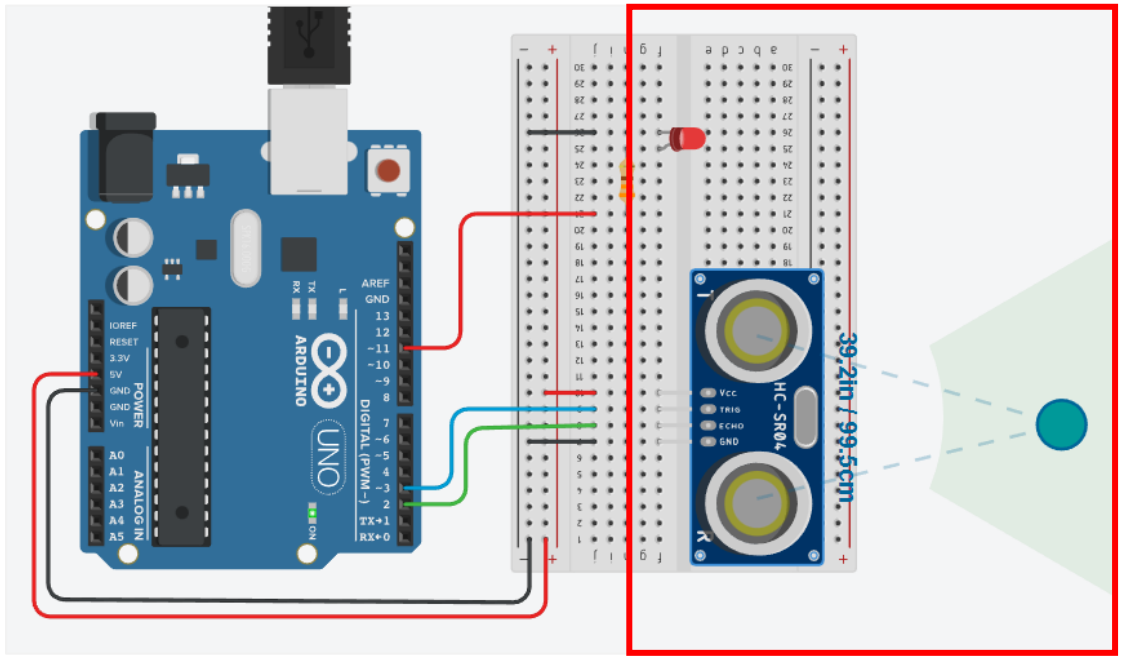
```
int trig = 3;
int echo = 2;


void setup()
{
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  Serial.begin(9600);
}
```

```
void loop()
{
  digitalWrite(trig, LOW);
  digitalWrite(echo, LOW);
  delayMicroseconds(2);

// 초음파 발사를 위해 트리거 핀을 10 마이크로초
// 동안 HIGH로 설정
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);
```

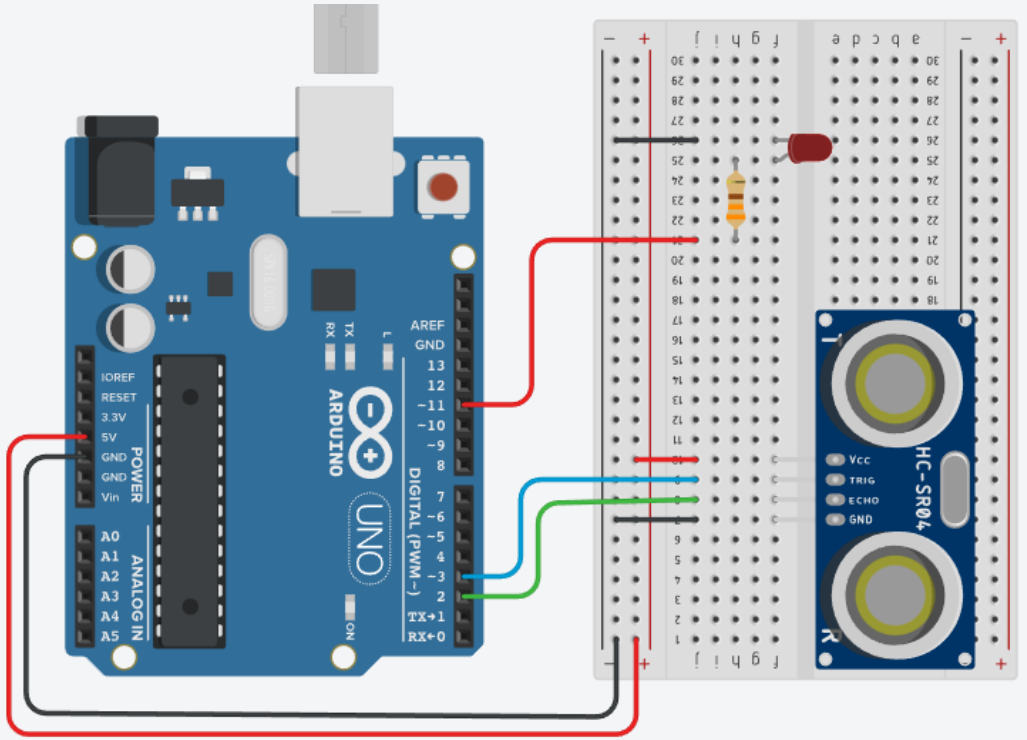
```
// 에코 핀이 HIGH로 변할 때까지의 시간을 측정
  unsigned long duration = pulseIn(echo, HIGH);

// 시간을 거리로 변환 (cm 단위)
// 음속(340m/s)로 나누고, 왕복 거리를 고려하여 2로
//나눕니다.
// 거리 = 속력 * 시간 / 2
  float distance = duration * 0.0343 /2.0;

  // 거리를 시리얼 모니터에 출력
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
}
```

일정거리에 장애물이 측정되면 LED를 켜보기

초음파센서로 거리를 측정하여 일정거리 안에 장애물이 인식되면 LED를 켜보는 프로그램을 작성해보자



장애물이 인식되면 LED를 켜기



일정거리에 장애물이 측정되면 LED를 켜보기

초음파센서로 거리를 측정하여 일정거리 안에 장애물이 인식되면
LED를 켜보는 프로그램을 작성해보자

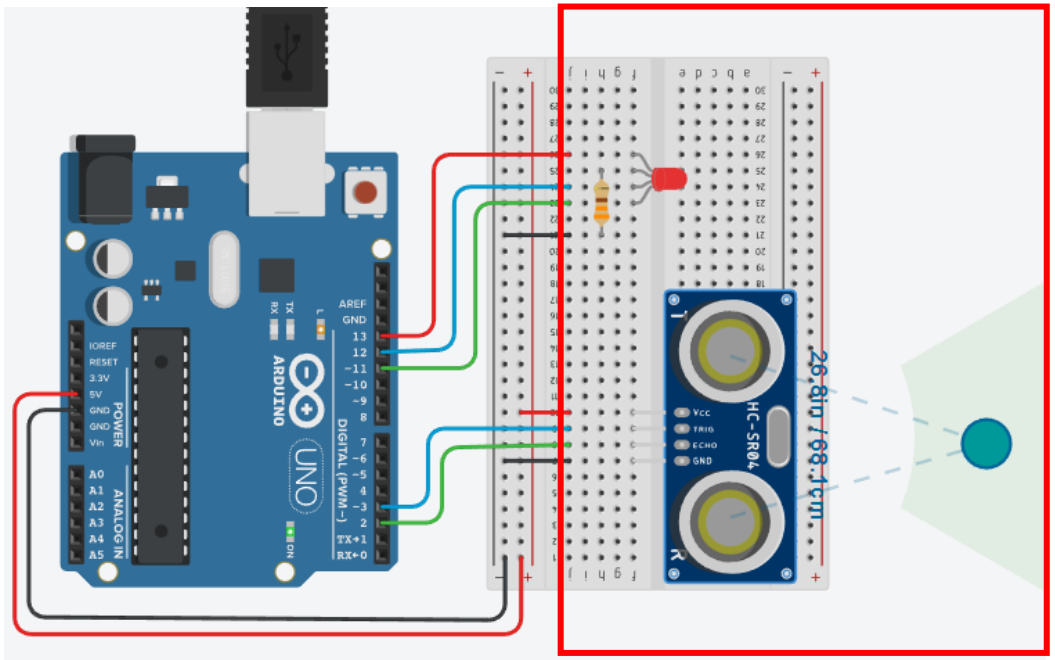
```
int trig = 3;
int echo = 2;
int Led = 11;

void setup()
{
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(Led, OUTPUT);
  Serial.begin(9600);
}
```

```
void loop()
{
  digitalWrite(trig, LOW);
  digitalWrite(echo, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

  unsigned long duration = pulseIn(echo, HIGH);

  float distance = duration * 0.0343 /2.0;
  Serial.print(distance);
  Serial.println("cm");
```

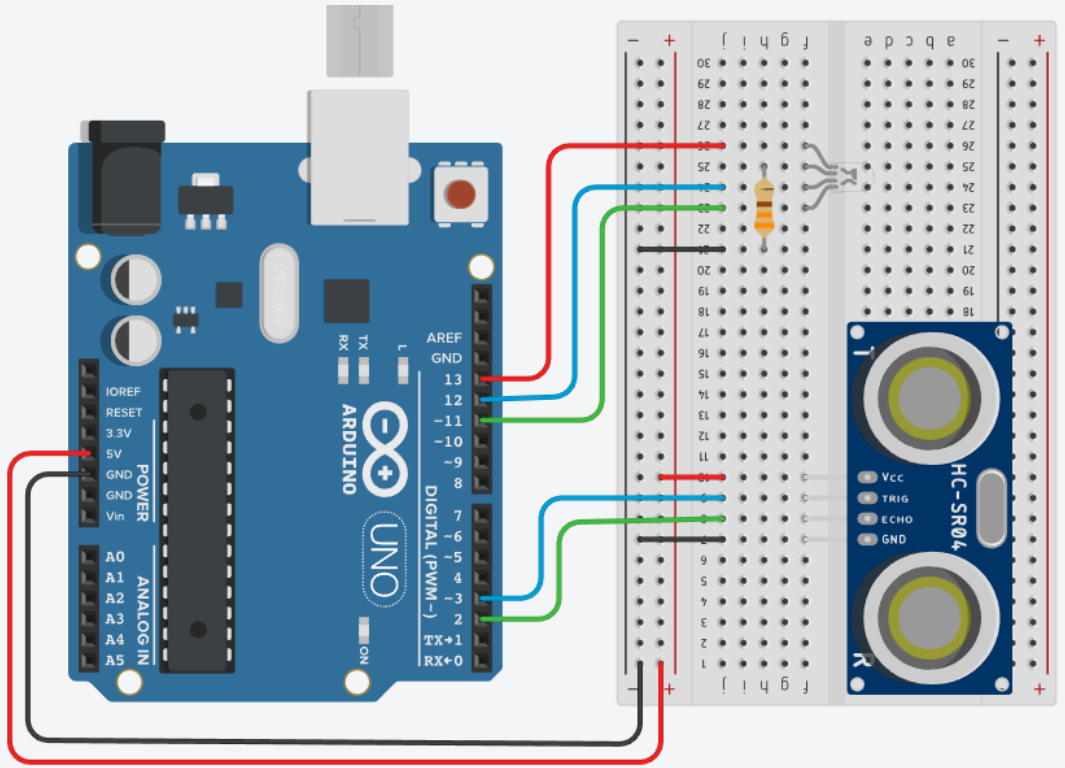
```
  if(distance < 100)
  {
    digitalWrite(Led, HIGH);
  }
  else
  {
    digitalWrite(Led, LOW);
  }
  delay(200);
}
```

거리에 따라 삼색 LED를 켜보기

초음파센서로 거리를 측정하여 10cm,20cm,30cm 일때마다 다른 색상의 LED를 켜보는 프로그램을 작성해보자



거리에 따른 다양한 LED 색을 켜보기



거리에 따른 다른 LED를 켜보기

초음파센서로 거리를 측정하여 10cm,20cm,30cm 일때마다 다른 색상의 LED를 켜보는 프로그램을 작성해보자

```
int trig = 3;
int echo = 2;
int LedR = 13;
int LedG = 11;
int LedB = 12;

void setup()
{
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  pinMode(LedR, OUTPUT);
  pinMode(LedG, OUTPUT);
  pinMode(LedB, OUTPUT);
  Serial.begin(9600);
}
```

```
void loop()
{
  digitalWrite(trig, LOW);
  digitalWrite(echo, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

  unsigned long duration = pulseIn(echo, HIGH);

  float distance = duration * 0.0343 /2.0;
  Serial.print(distance);
  Serial.println("cm");
```

```
if(distance > 200)  {
   digitalWrite(LedR, LOW);
   digitalWrite(LedG, LOW);
   digitalWrite(LedB, HIGH);
 }
 else if(distance > 100)   {
   digitalWrite(LedR, LOW);
   digitalWrite(LedG, HIGH);
   digitalWrite(LedB, LOW);
 }
 else  {
   digitalWrite(LedR, HIGH);
   digitalWrite(LedG, LOW);
   digitalWrite(LedB, LOW);
 }
delay(200);   } // loop end
```

05 학습정리

학습 정리

- 초음파센서의 거리를 계산하는 방법을 학습하고 계산된 값을 프로그래밍 하여 시리얼 모니터에 출력
- 초음파센서 거리에 따른 다양한 LED를 켜는 회로도 작성과 코딩을 실습

6차시

서보모터의 동작원리 이해와 초음파센서의 활용

06 학습목표, 학습내용

학습 목표

- 서보모터 작동 원리 이해한다.
- 서보모터 제어 방법 익히기
- 초음파센서와 서보모터 연동
- 초음파센서의 거리에 따른 서보모터 제어

학습 내용

- 서보모터의 케이블 핀 제어연결 방법에 대해 실습한다.
- 서보모터의 각도를 제어하여 움직임을 확인한다.
- 서보모터와 초음파와 같이 활용하여 초음파센서의 거리값에 따른 서보모터 연동 및 제어 방법 설명한다.

액추에이터 동작하기

액추에이터(Actuator)는 시스템에서 신호나 데이터를 받아 특정 동작을 수행하는 장치를 말한다.

엑추에이터를 제어하는 것은 아두이노에서 센서의 데이터를 분석하여 액추에이터를 동작시켜 원하는 동작을 수행하는 것이다.

[서보모터]

축을 0~180도 까지 원하는 각도로 회전할 수 있습니다.
무게 9g 서보 모터를 사용합니다.

서보 모터를 제어할 때에는 Servo 라는 라이브러리를 사용합니다.

[DC모터]

360도 회전하고 방향을 조절할 수 있습니다. 미니카, 선풍기, 자동차 바퀴에 사용합니다.

[서보모터]

[스텝모터]

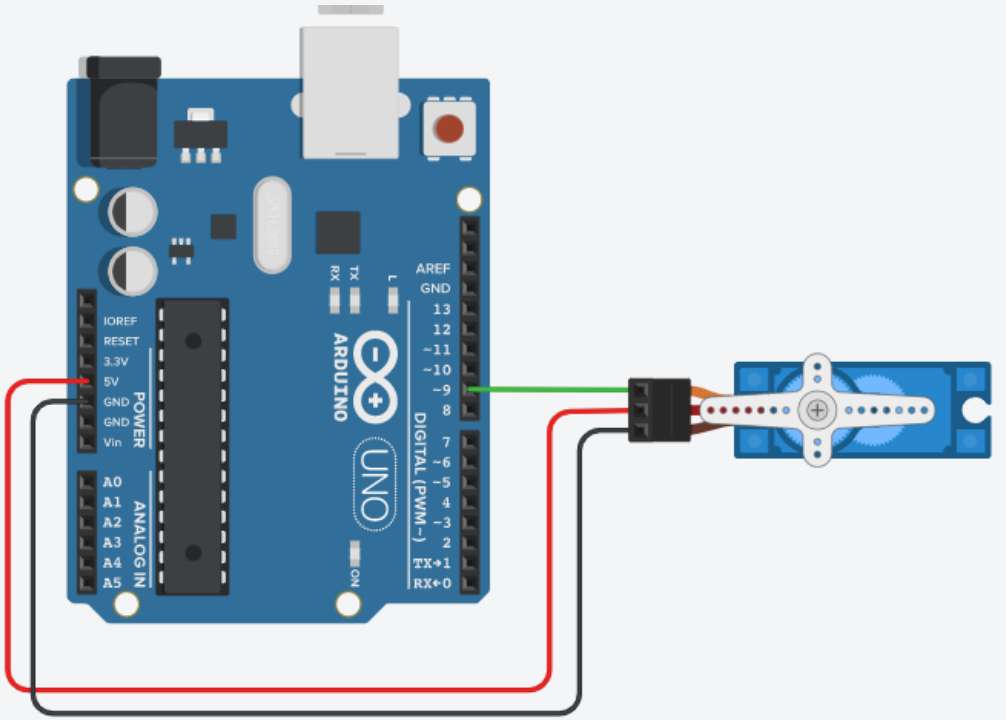
명령에 따라 정해진 스텝 수만큼 회전하므로 일정한 각도와 속도로 움직임이 가능하다.

서보모터 움직이기

서보모터는 0도 부터 180도까지만 움직인다. 차단기는 0도와 90도 사이로 조정하여 만들수 있다.

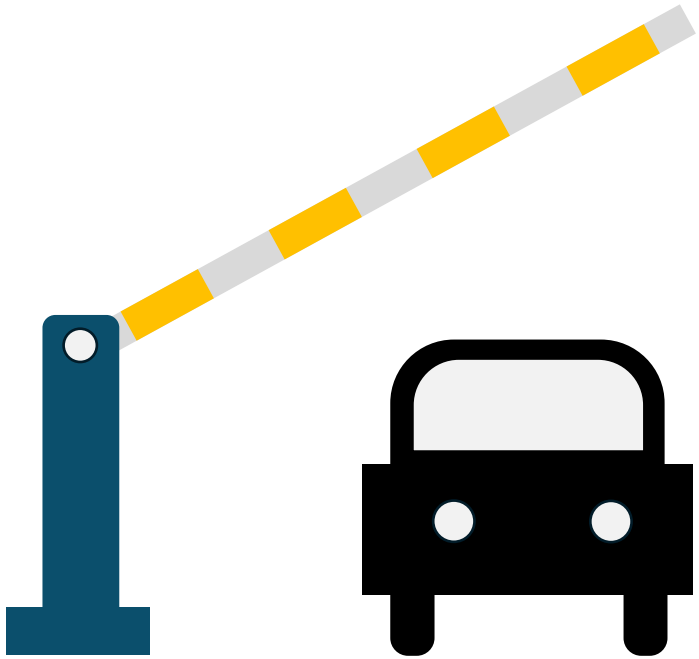
```
#include <Servo.h>
Servo servo;
void setup()
{
 servo.attach(9); // 서보모터를 9번핀에 연결
}

void loop()
{
 servo.write(90); // 서보모터를 90도 움직임
 delay(2000);
 servo.write(0); // 서보모터를 0도 움직임
 delay(2000);
}
```

서보모터를 90도와 0도 사이로 움직이기

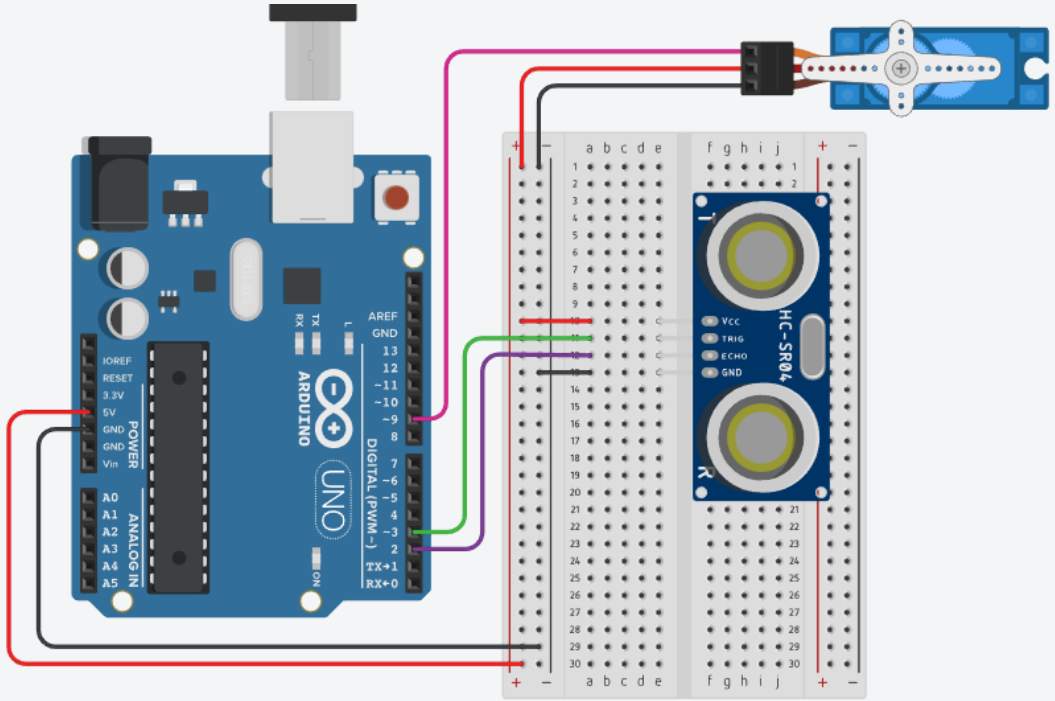


서모모터와 초음파센서로 차단기 만들기

- 초음파센서에 장애물 또는 차량등이 인식되면 서보모터를 활용하여 차단기의 움직임을 코딩할 수 있다.



차단기 만들기



서모모터와 초음파센서로 차단기 만들기

초음파센서에 장애물 또는 차량등이 인식되면 서보모터를 활용하여 차단기의 움직임을 코딩할 수 있다.

```
//센서의 송신부를 3번핀으로 선언하고
//수신부는 2번핀으로 선언합니다.
int trig = 3;
int echo = 2;

#include <Servo.h>
Servo servo;

void setup()
{
  Serial.begin(9600);
  pinMode(trig, OUTPUT);
  pinMode(echo, INPUT);
  servo.attach(9);
  // 서보모터를 9번핀에 연결
}
```

```
void loop()
{
  digitalWrite(trig, LOW);
  digitalWrite(echo, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

unsigned long duration = pulseIn(echo, HIGH);

//초음파의 속도는 초당 340미터를 이동하거나,
//29마이크로초 당 1센치를 이동합니다.
// 따라서, 초음파의 이동 거리 = duration(왕복에
//걸린시간) / 29 / 2 입니다.
  float distance = duration / 29.0 / 2.0;
```

```
 // 측정된 거리 값를 시리얼 모니터에 출력합니다.
  Serial.print(distance);
  Serial.println("cm");

// 측정된 거리가 10cm 이하라면, 아래의 블록을 실행합니다.
  if (distance < 100)
{
 servo.write(90); // 서보모터를 90도 움직임
 delay(2000);
 }
 // 측정된 거리가 10cm 이상이라면, 아래의 블록을 실행합니다.
  else
    {
     servo.write(0); // 서보모터를 0도 움직임
     delay(1000);
    }   // 0.1초 동안 대기합니다.
}
```

06 학습정리

학습 정리

- 서보모터의 연결방법을 확인하고 움직임의 작동원리에 대하여 실습
- 초음파센서의 거리값을 확인하여 서보 모터가 움직이는 각도에 따라 차단기의 원리 이해

7차시

초음파센서+부저+서보모터의 활용 및 PIR센서 제어

07 학습목표, 학습내용

학습 목표

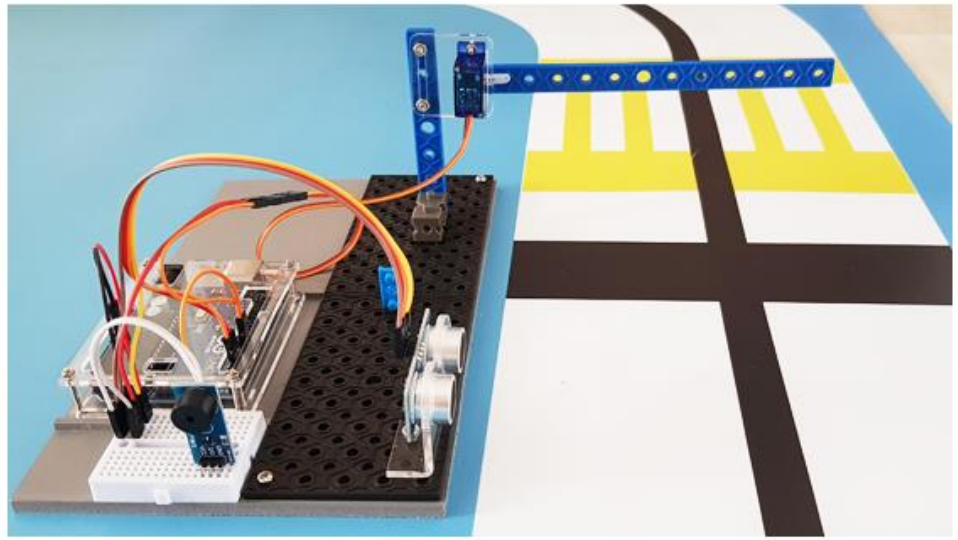
- 서보모터, 초음파센서, 부저의 기본 작동 원리 이해 및 동작방법 이해
- 부저를 활용하여 소리를 발생시키는 방법 익히기
- PIR센서와 LED를 연동하여 움직임 감지에 따른 동작 구현 방법 이해

학습 내용

- 서보모터, 초음파센서, 부저 센서들을 연동하여 활용하는 프로그래밍 작성 방법 학습한다.
- 장애물 인식시 부저 스피커가 소리날 수 있도록 제어한다.
- 인체감지센서인 PIR 센서의 동작원리를 이해한다.

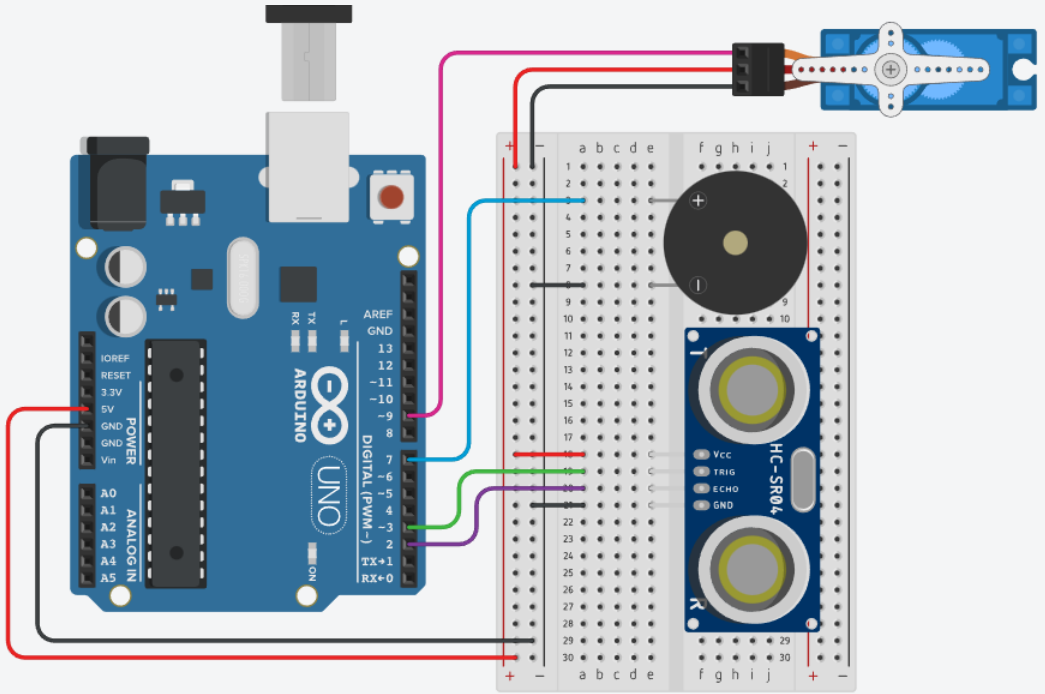
서모모터와 초음파센서로 차단기 만들기 - 소리넣기

초음파센서에 장애물 또는 차량등이 인식되면 서보모터를 활용하여 차단기의 움직임을 코딩할 수 있다.



아두이노 우노와 모터, 초음파센서, 부저와 모터뭉치 끝에 빨대나 아이스크림 막대등을 활용하여 차단기를 작동시키는 예제 프로젝트를 만들어서 작동을 확인할 수 있다.

차단기 만들기 - 소리넣기



서모모터와 초음파센서로 차단기 만들기

앞 예제에서 필요한 부분만 추가하여 수정하고 실행해보자.

```
//센서의 송신부를 3번핀으로 선언하고
//수신부는 2번핀으로 선언합니다.
int trig = 3;
int echo = 2;

#include <Servo.h>
Servo servo;
```

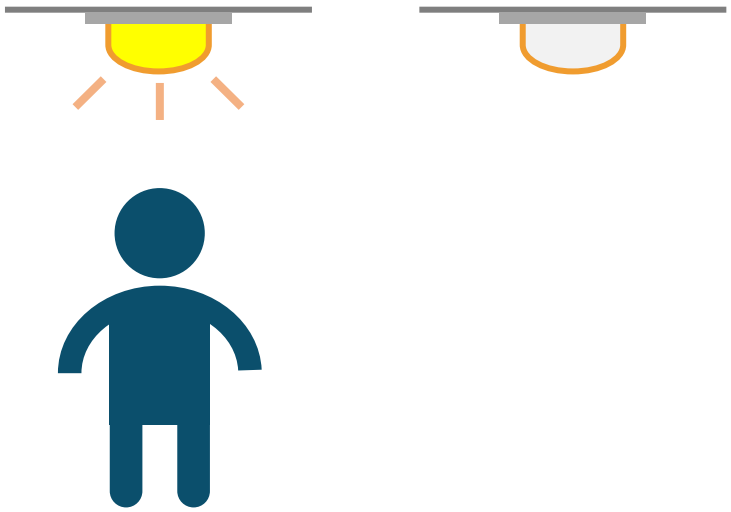
```
//센서의 송신부를 3번핀으로 선언하고 //수신부는 2번핀으로 선언합니다.
int trig = 3;
int echo = 2;
int piezoPin = 7;
int notes[] = {6900, 4906, 6900, 4906, 6900};
```

```
// 측정된 거리가 10cm 이하라면, 아래의 블록을 실행합니다.
  if (distance < 100)
{
 servo.write(90); // 서보모터를 90도 움직임
 delay(2000);
 }
```

```
// 측정된 거리가 10cm 이하라면, 아래의 블록을 실행합니다.
  if (distance < 100)
   {
    for (int i = 0; i < 5; i++)
     {
       tone(piezoPin, notes[i], 150);
       delay(150);
      }
    servo.write(90); // 서보모터를 90도 움직임
    delay(2000);
   }
```

동작을 감지해보자

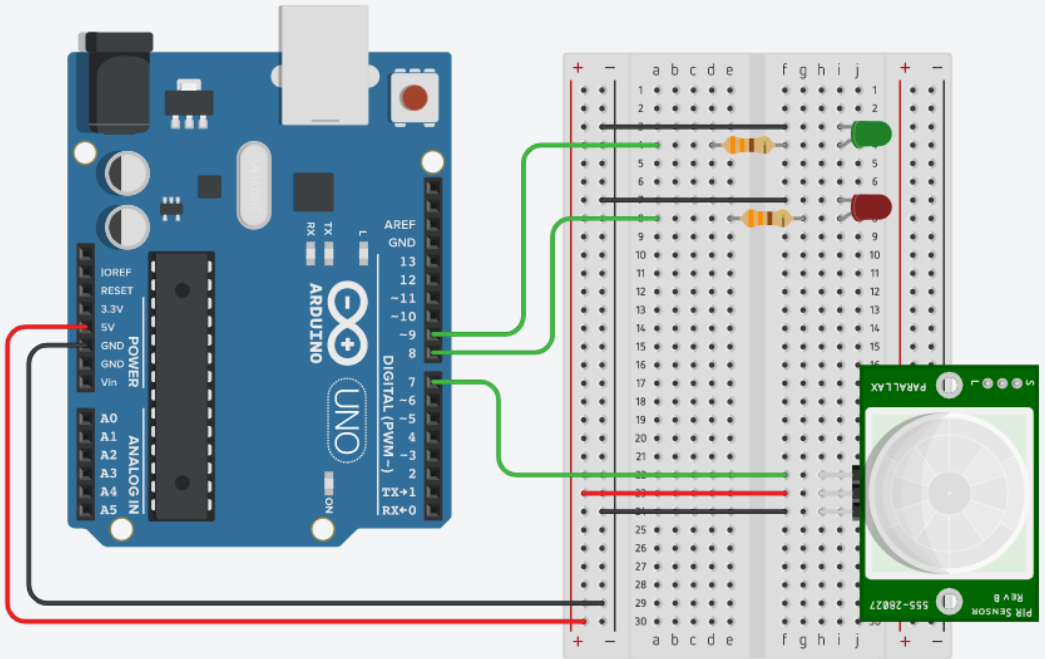
근적외선 PIR 센서 : 인체 감지 모션센서에 대해서 알아보자



사람 또는 물체가 있을 때

사람 또는 물체가 없을 때

보통 건물이나 아파트 출입구, 복도, 현관문 천장등에 부착되어 사람이나 동물의 움직임이 감지되면 조명을 켜고, 꺼주는 역할을 하는 센서이다.
감도조절과 신호지속시간을 조절할 수 있다.

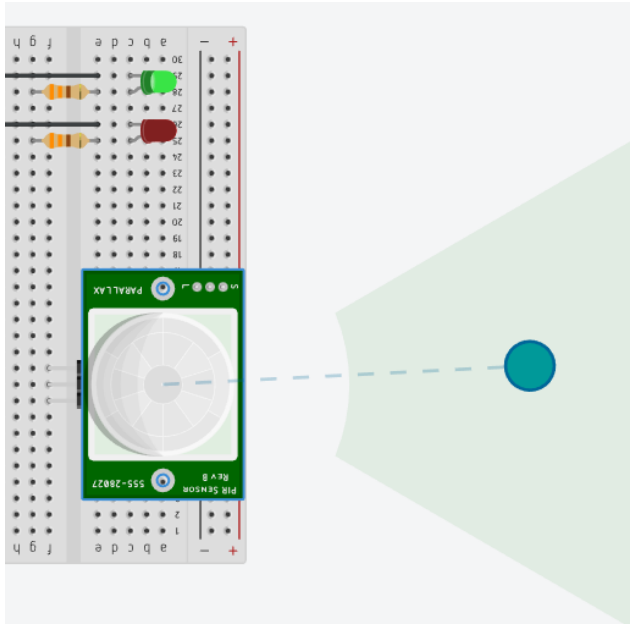


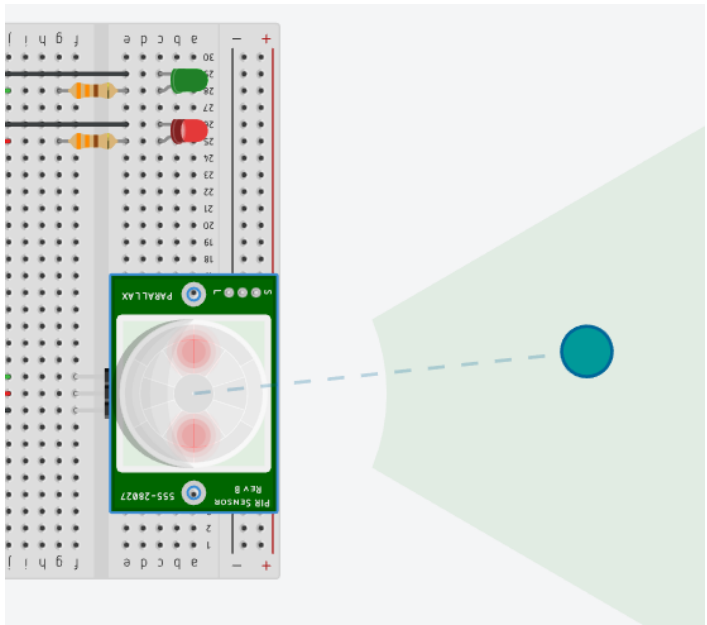
동작을 감지해보자

근적외선 PIR 센서 : 인체 감지 모션센서에 대해서 알아보자

감도조절

신호지속시간조절





센서의 감도조정 가변저항/ Delay 시간조절 가변저항으로 조절하여 사용

움직임이 없을때에는 초록색 LED/움직임이 감지되었을때 빨간색 LED가 켜짐

동작을 감지해보자

인체감지가 되면 초록색 LED가 켜지고, 움직임이 없으면 빨간색 LED를 켜도록 프로그래밍 해보자.

```
int ledPin1 = 9;                    // 1번 LED
int ledPin2 = 8;                    // 2번 LED
int inputPin = 7;                   // PIR 센서 신호핀
int pirState = LOW;     // PIR 센서 초기상태는 움직임이 없음을 가정
int val = 0;            // 센서 신호의 판별을 위한 변수

void setup()
 {
    pinMode(ledPin1, OUTPUT);    // 1번 LED를 출력으로 설정
    pinMode(ledPin2, OUTPUT);    // 2번 LED를 출력으로 설정
    pinMode(inputPin, INPUT);    // 센서 Input 설정
    Serial.begin(9600);
 }
```

```
void loop()
  {
    val = digitalRead(inputPin);     // 센서 신호값을 읽어와서 val에 저장
    if (val == HIGH)
       {                          // 센서 신호값이 HIGH면(인체 감지가 되면)
         Serial.println(val);
         digitalWrite(ledPin1, LOW);           // 1번 LED ON
         digitalWrite(ledPin2, HIGH);          // 2번 LED OFF
        }
     else
       {                          // 센서 신호값이 LOW면(인체감지가 없으면)
         Serial.println(val);
         digitalWrite(ledPin1, HIGH);          // 1번 LED OFF
         digitalWrite(ledPin2, LOW);            // 2번 LED ON
       }
   }
```

07 학습정리

학습 정리

- 초음파센서에 장애물이 인식되었을때,
  차단기 원리로 서보모터를 움직이고
  부저음으로 동작구현을 실습
- 실생활의 인체감지센서 PIR센서의
  동작원리를 이해하여 LED를 켜는 실습

8차시

I2C LCD를 활용하여 정보를 출력

08 학습목표, 학습내용

학습 목표

- LCD 모듈의 기본 동작 원리와 텍스트 출력 방법 이해
- 온도센서와 LCD 모듈을 연결한 회로의 구성 방법 이해
- 온도센서로 측정한 온도 값을 LCD에 표시하고 LED 제어하는 방법 익히기

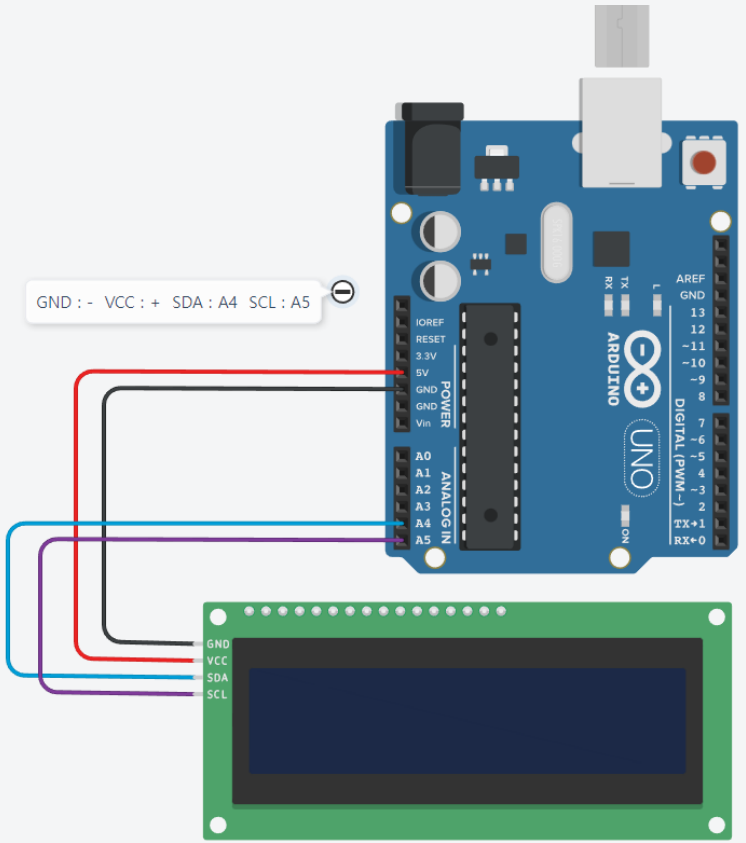
학습 내용

- LCD 모듈 기본 동작 및 텍스트 출력 방법 학습한다.
- 온도센서와 LCD센서의 특성을 파악하고 회로구성방법과 틴커캐드의 설정방법을 학습한다.
- 온도값에 따른 LED를 작동시키는 방법에 대해 알아본다.

LCD - 정보를 출력해보자

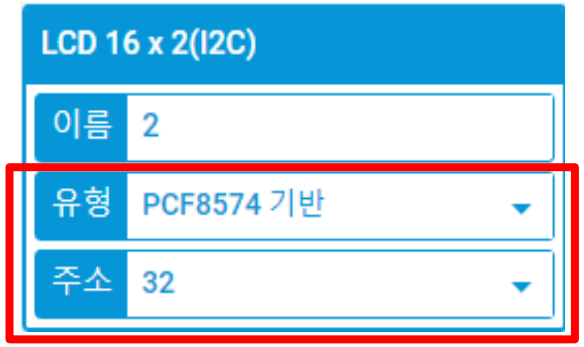
LCD(액정표시장치)를 연결하고 제어하여 간단한 텍스트를 출력해보자.

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(32, 16, 2);   // 0x27 ,0x3F 두가지 확인

// set the LCD address to 0x27(0x3F)
// for a 16 chars and 2 line display

void setup()
  {
      lcd.init();
      lcd.backlight();
      lcd.setCursor(0,0);
      lcd.print("Hello, everyone!");
      lcd.setCursor(0,1);
      lcd.print("Enjoy Day");
  }

void loop()
  {
  }
```



LCD - 옵션 및 라이브러리 추가하기

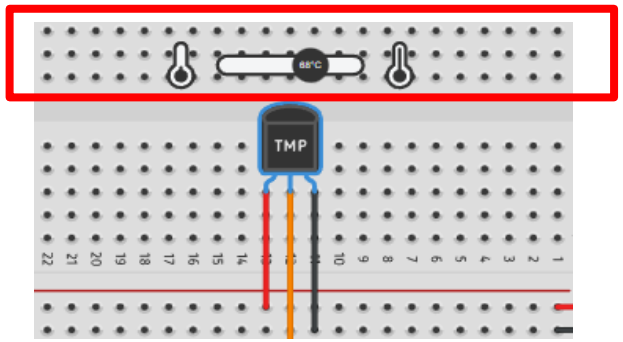
- 틴커캐드에서 LiquidCrystal I2C 라이브러리를 추가하는 방법을 확인하고, 유형 부분에서 “PCF8574기반” 옵션을 변경하자.
  전달되는 주소값은 실제 아두이노에서는 0x27, 0x3F를 사용하고 시뮬레이션에서는 주소값이 32이다.



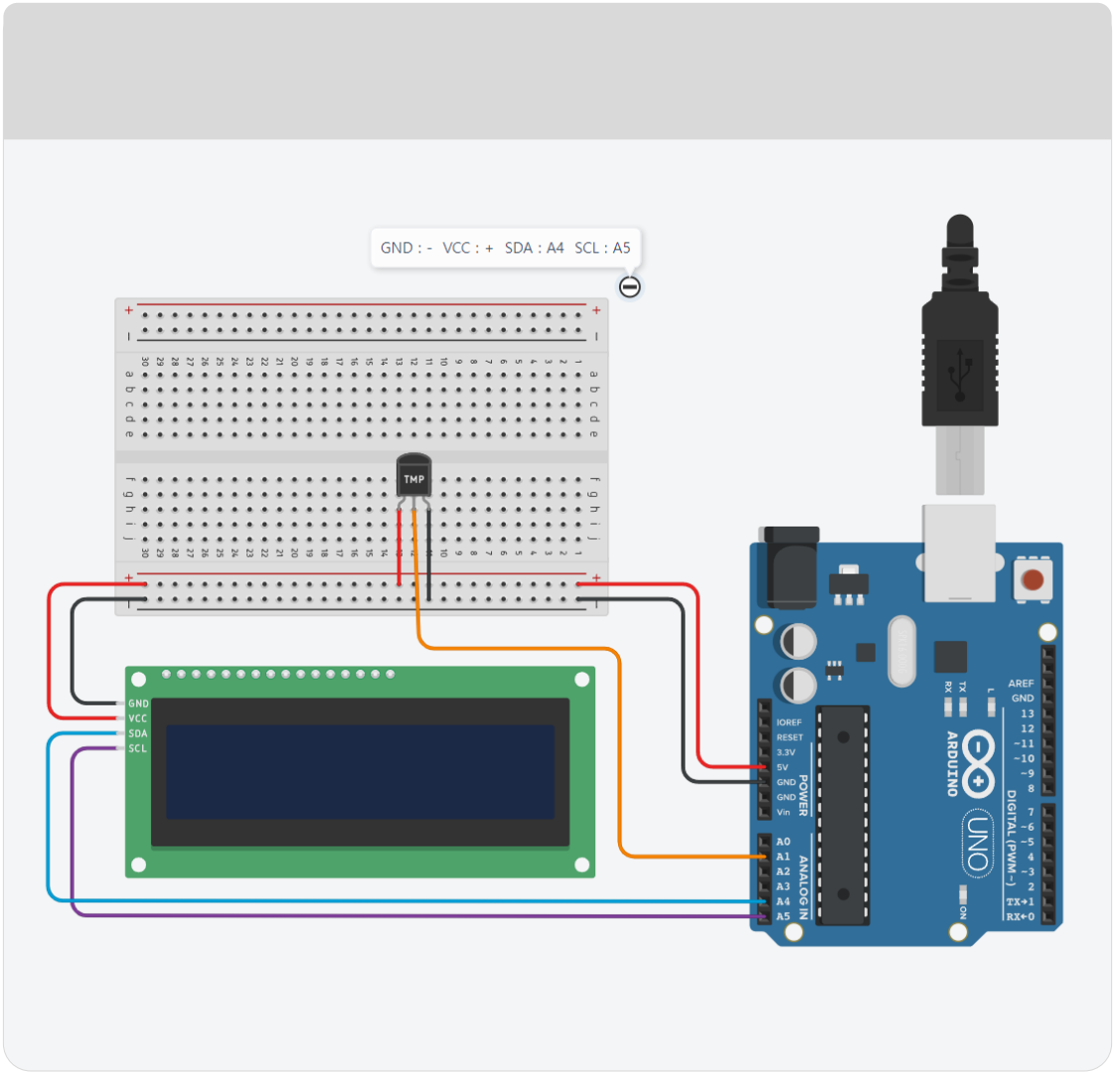
LCD - 온도를 측정하여 출력해보기

온도센서를 아두이노에 연결하여 온도를 측정하고 LCD에 표현해보자. 센서의 동작원리 와 온도측정 코드작성법을 이해하자.

온도센서를 클릭하여 온도 변화를 확인가능



센서		핀번호
LED		A0
TMP		A1
LCD	GND	-
	VCC	+
	SDA	A4
	SCL	A5



온도를 측정하여 LCD에 출력해보기

- 아날로그 입력을 사용할때 ADC(아날로그-디지털 변환기)의 참조 전압을 내부 1.1V로 설정한다. 아날로그 입력의 정확도를 높이기 위해 사용된다.

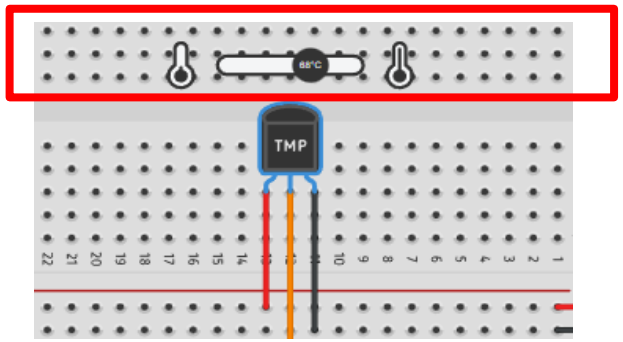
```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(32,16,2); //16문자, 2줄

// LM35 온도 센서에 연결된 아날로그 핀
int Tmp36Pin = A1;
// 아날로그 값을 저장하기 위한 변수
int reading;

void setup()
{
  analogReference(INTERNAL);
   //ADC 참조 전압을 INTERNAL 1.1V
  lcd.init();
  lcd.backlight();
  Serial.begin(9600);
}
```

```
void loop()
{
  lcd.setCursor(0,0);
  reading = analogRead(Tmp36Pin);
  lcd.print("temp : ");
  lcd.print(reading);
  delay(500);

}
```

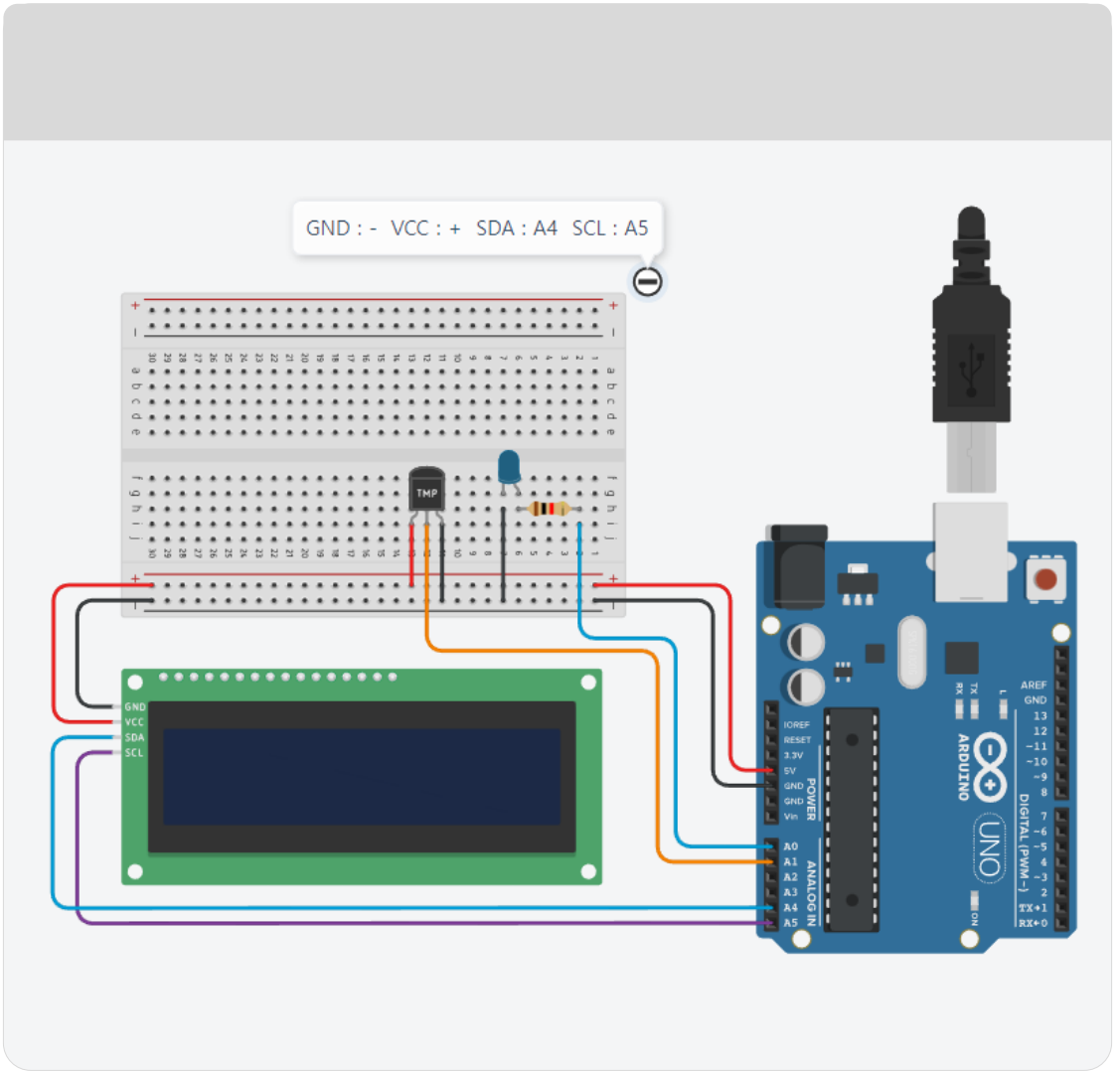
LCD - 온도를 측정하여 출력해보기

온도센서를 아두이노에 연결하여 온도를 측정하고 LCD에 표현해보자. 센서의 동작원리 와 온도측정 코드작성법을 이해하자.

온도센서를 클릭하여 온도 변화를 확인가능



센서		핀번호
LED		A0
TMP36		A1
LCD	GND	-
	VCC	+
	SDA	A4
	SCL	A5



온도를 측정하여 LCD에 출력해보기

아날로그 입력을 사용할때 ADC(아날로그-디지털 변환기)의 참조 전압을 내부 1.1V로 설정한다. 아날로그 입력의 정확도를 높이기 위해 사용된다.

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(32,16,2); //16문자, 2줄

int ledPin = A0;     // LED에 연결된 디지털 핀
int Tmp35Pin = A1;    // LM35 온도 센서에 연결된 아날로그 핀
int reading;         // 아날로그 값을 저장하기 위한 변수
float temperature;  // 온도 값을 저장하기 위한 변수
float voltage;       // 전압 값을 저장하기 위한 변수

void setup()
  {
    analogReference(INTERNAL);
                     // ADC 참조 전압을 내부 1.1V로 설정
    lcd.init();
    lcd.backlight();
    pinMode(ledPin, OUTPUT) ;
    Serial.begin(9600);
  }
```

```
void loop()
  {
    lcd.setCursor(0,0);
    // lm35Pin으로부터 아날로그 값을 읽음
    reading = analogRead(Tmp35Pin);
    //아날로그 값을 전압으로 변환 (0-5V)
    voltage = reading * 5.0/1023.0;
    temperature = (voltage- 0.5)*10;   // 전압을 온도로 변환 (섭씨)

    lcd.print("temp : ");
    lcd.print(temperature);

   if(temperature > 28)
     {
       digitalWrite (ledPin, HIGH);
     }
   else
     {
       digitalWrite(ledPin, LOW);
      }
  delay(1000);
 }//loop end
```

08 학습정리

학습 정리

- I2C LCD의 라이브러리 설정방법에 대하여 실습
- LCD 출력방법 이해
- 온도값을 LCD에 출력하고 일정온도값에 따라 LED를 켜고 LCD에 출력하는 실습

9차시

압력센서를 이용한 데이터 확인과 LED, 서보모터의 활용

09 학습목표, 학습내용

학습 목표

- 압력센서의 동작원리를 알아본다.
- 압력센서의 값 범위를 시리얼 모니터로 확인한다.
- 압력센서와 LED, 서모모터의 제어 방법에 대해 알아본다.

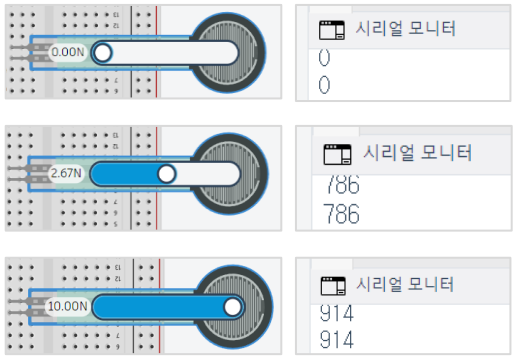
학습 내용

- 압력센서의 작동 원리와 응용을 이해한다.
- 압력센서로 측정한 값에 따라 LED를 제어하는 방법을 익힌다.
- 압력센서로 측정한 값에 따라 서보모터의 움직음을 제어하는 방법을 익힌다.

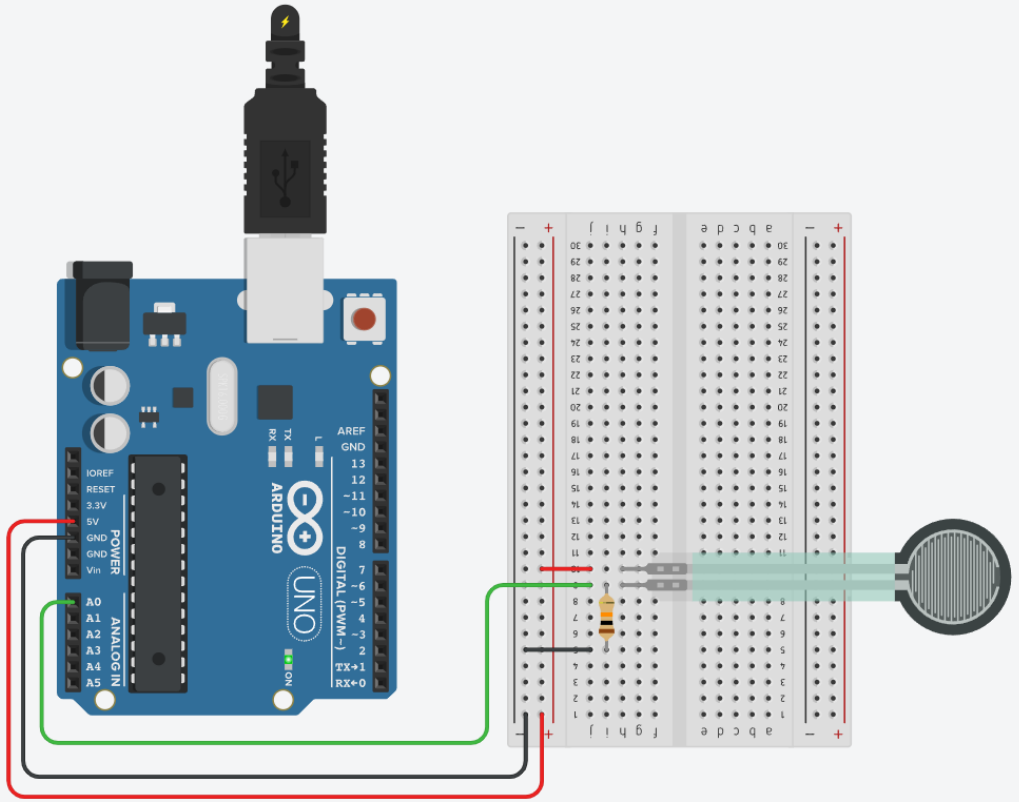
압력센서 값을 확인하기

압력센서를 아날로그핀인 A0에 연결하여, 압력센서의 값 범위를 확인하도록 코딩한다.

```
int press_sensor = A0;
int val = 0;

void setup()
{
  Serial.begin(9600);
}
void loop()
{
  val = analogRead(press_sensor);
  Serial.println(val);
}
```

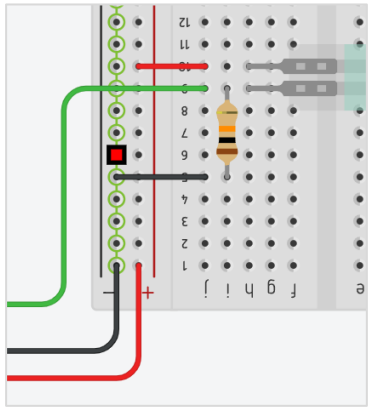


압력센서 data 값을 시리얼모니터에 출력하기



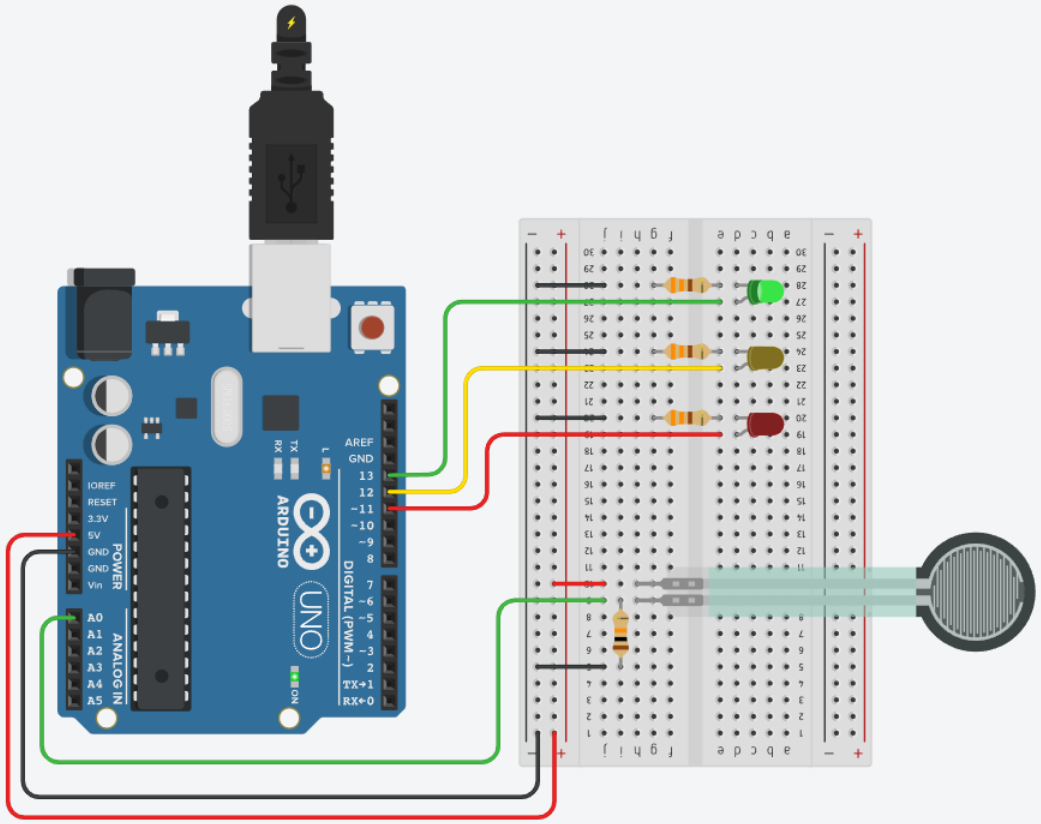
압력센서 값을 확인하기- LED

- 압력센서의 1핀쪽에는 VCC를 연결하고, 2핀쪽에 10옴짜리 저항을 연결하고, GND연결을 완성한다.



센서		핀번호
GREEN LED		13
YELLOW LED		12
RED LED		11
Pressure sensor	1	+
	2	A0

압력값에 따라 다른 LED 켜기



압력센서 값을 확인하기- LED

시리얼 모니터에서 압력센서 값을 확인하여 조건문에 따라 압력이 작으면 초록색 LED를, 압력이 중간정도이면 노란색 LED를 압력이 커지면 경고를 나타내는 빨간색 LED를 켜는 프로그램을 작성한다.

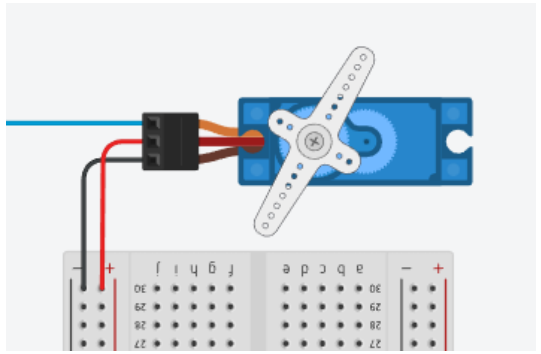
```
int press_sensor = A0;
int val = 0;
int Rled = 11;
int Yled = 12;
int Gled = 13;

void setup()   {
  Serial.begin(9600);
  pinMode(Rled, OUTPUT);
  pinMode(Yled, OUTPUT);
  pinMode(Gled, OUTPUT);
}
```

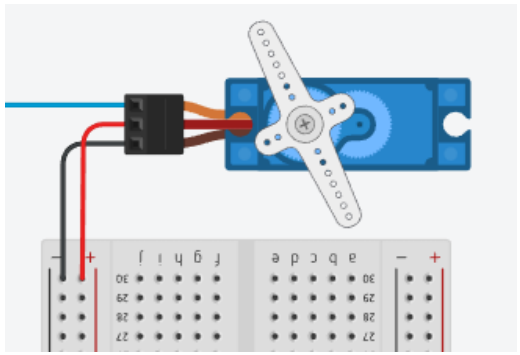
```
void loop()
{
  val = analogRead(press_sensor);
  Serial.println(val);
  if (val > 900)  {
    digitalWrite(Rled,HIGH);
    digitalWrite(Yled,LOW);
    digitalWrite(Gled,LOW);
  }
  else if (val > 500)  {
    digitalWrite(Rled,LOW);
    digitalWrite(Yled,HIGH);
    digitalWrite(Gled,LOW);
  }
```

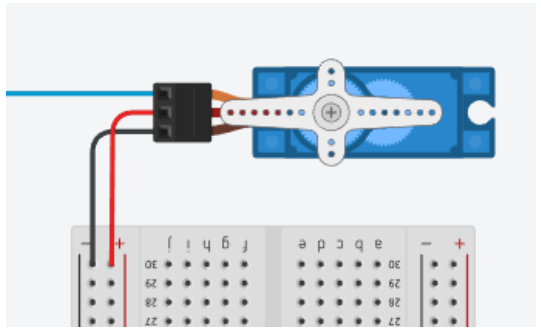
```
  else  {
      digitalWrite(Rled,LOW);
      digitalWrite(Yled,LOW);
      digitalWrite(Gled,HIGH);
    }
  }
```

압력센서와 서보모터 움직이기

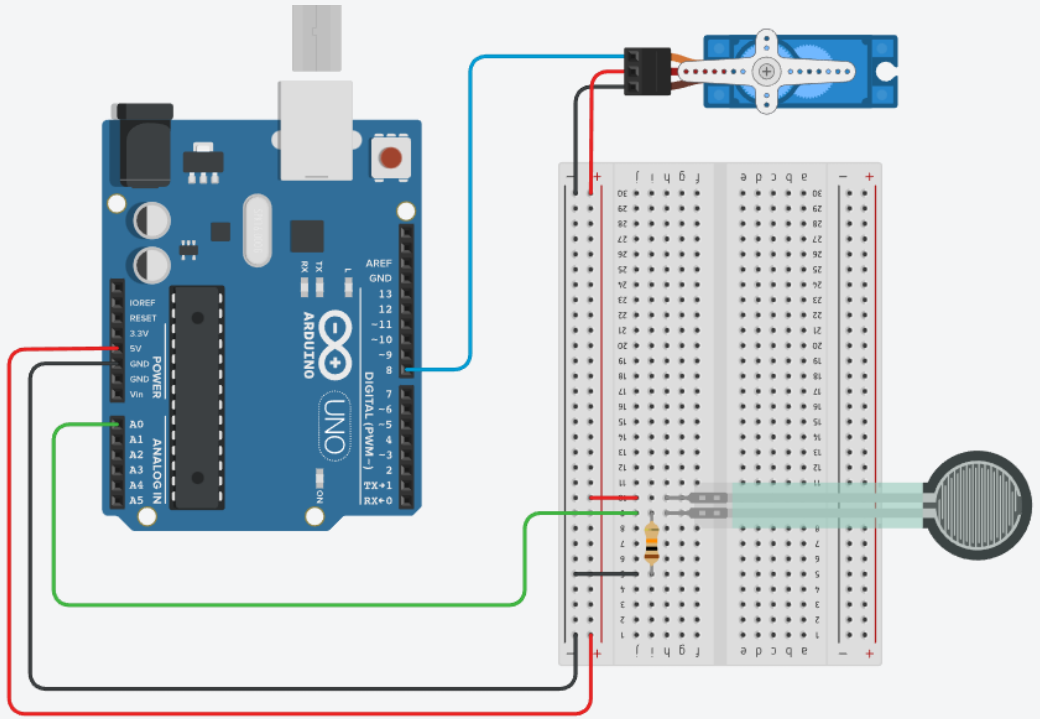
압력센서에 서보모터를 연결추가하여 압력센서 값의 범위에 따라 다른 각도로 움직이는 프로그램을 작성해보자.







압력값에 따라 서보모터 움직이기



압력센서와 서보모터 움직이기

압력센서에 서보모터를 연결추가하여 압력센서 값의 범위에 따라 다른 각도로 움직이는 프로그램을 작성해보자.

```
#include <Servo.h>

int press_sensor = A0;
int val = 0;

Servo servoMotor;

void setup()
{
  Serial.begin(9600);
  servoMotor.attach(8);
// 서보 모터를 8번 핀에 연결
}
```

```
void loop()
{
  val = analogRead(press_sensor);
  Serial.println(val);

  // 압력 값에 따라 각도 조절
  if (val < 300) {
    servoMotor.write(0);
    // 첫 번째 범위 (0도)
  }
  else if (val < 500) {
    servoMotor.write(60);
    // 두 번째 범위 (60도)
  }
```

```
else if (val < 700) {
    servoMotor.write(120);
    // 두 번째 범위 (120도)
  }
else {
    servoMotor.write(180);
    // 세 번째 범위 (180도)
  }
delay(500);
   // 각도 변화를 확인하기 위한 딜레이
}
```

09 학습정리

학습 정리

- 압력센서 값을 아날로그값을 읽어오고 해당 값의 범위를 확인하는 코딩실습
- 압력 값에 따라 단계별로 LED를 켜는 다양한 동작을 수행하는 프로그램 작성 방법 습득

10차시

가스센서로 가스 누출 확인 시스템 및 알람울리기

학습 목표

- 가스센서의 작동 원리와 LED 제어 방법을 이해한다.
- 가스센서로 감지한 값에 따라 LED와 부저를 제어한다.

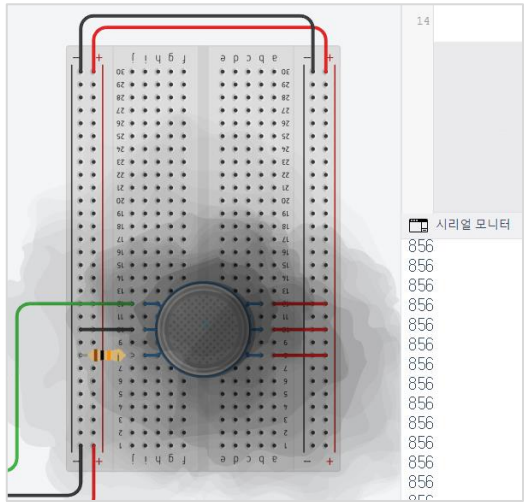
학습 내용

- 가스센서의 값을 시리얼 모니터로 확인하고 가스 연기의 움직임을 제어한다.
- 가스센서의 값을 감지하여 가스누출 확인시 LED와 부저에 의해 알림 기능을 설정한다.

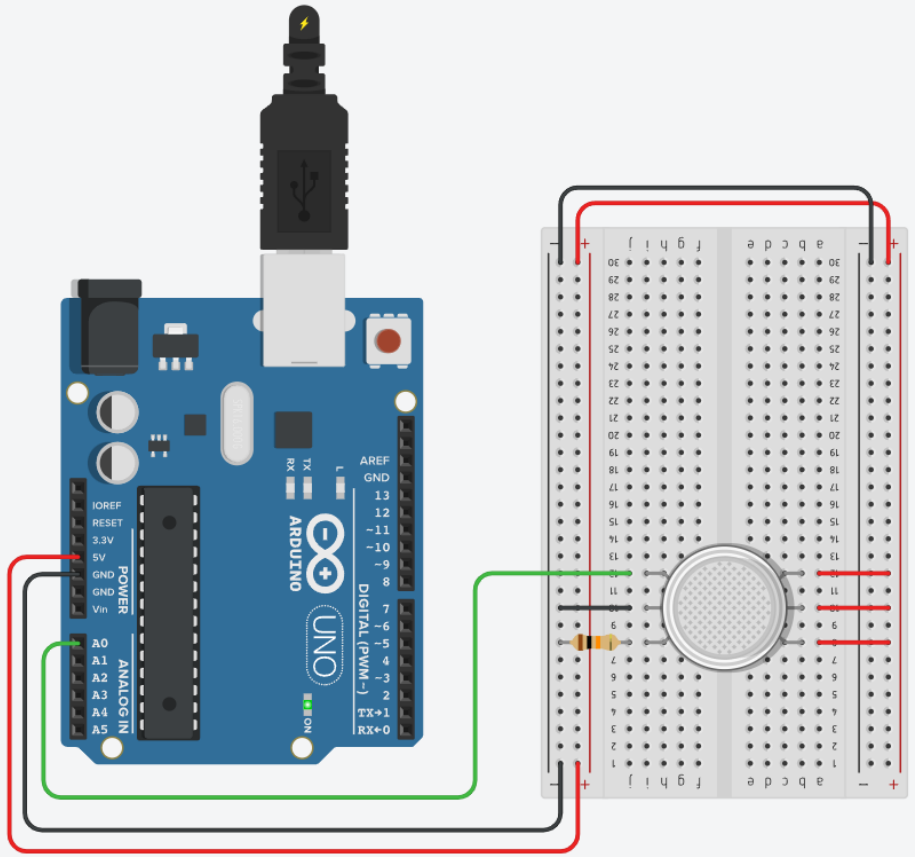
가스센서 – 값 확인하기

가스연기가 센서위에 있을때 값이 856, 멀리있을때 약 400의 값을 시리얼 모니터에서 확인해서 프로그래밍 하자.

```
int gas_sensor = A0;
int val = 0;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  val = analogRead(gas_sensor);
  Serial.println(val);
}
```



가스센서의 값을 모니터링 해보기



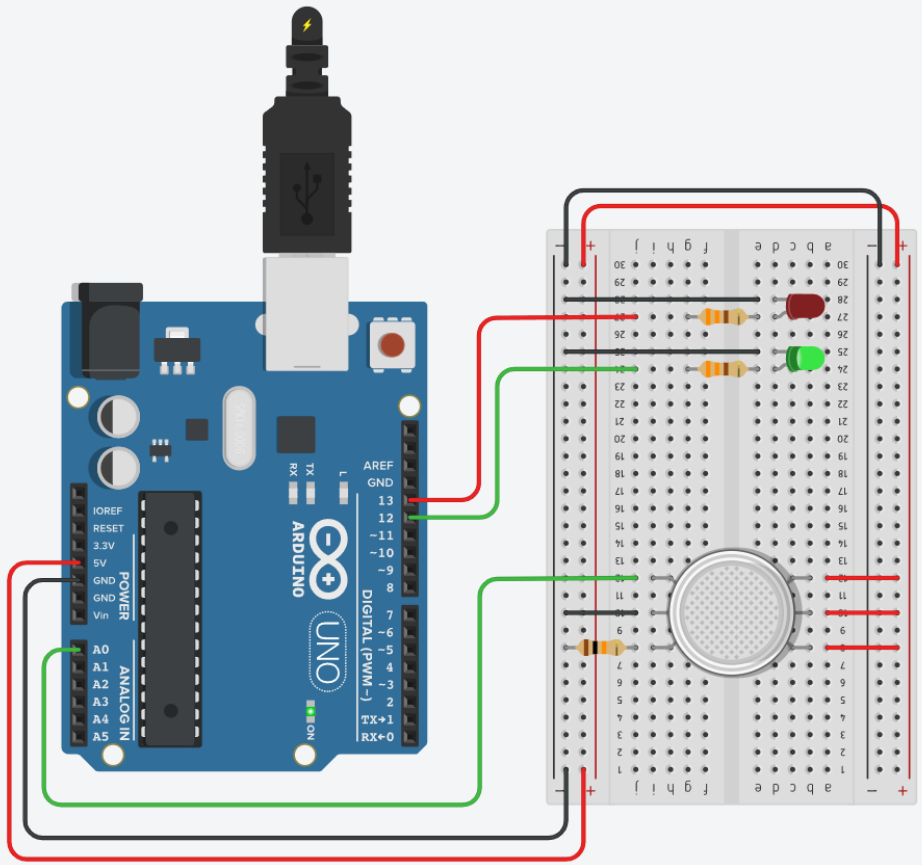
가스센서 – LED로 가스누출 확인하기

가스누출이 없을때에는 초록색 LED를 켜고, 가스연기가 가스센거 가까이 와서 값이 높아지면, 경고성인 빨간색 LED를 켠다.

```
int gas_sensor = A0;
int Rled = 13;
int Gled = 12;
int val = 0;

void setup()
{
  Serial.begin(9600);
  pinMode(Rled,OUTPUT);
  pinMode(Gled,OUTPUT);
}
```

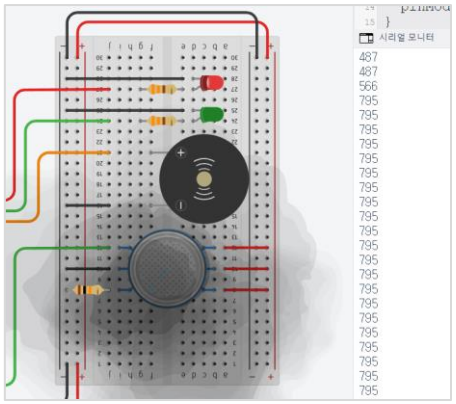
```
void loop()
{
 val = analogRead(gas_sensor);
  Serial.println(val);
  if(val > 500)  {
    digitalWrite(Gled, LOW);
    digitalWrite(Rled, HIGH);
  }
  else  {
    digitalWrite(Gled, HIGH);
    digitalWrite(Rled, LOW);
  }
 }
```

가스센서에 연기가 감지되면 LED 켜기



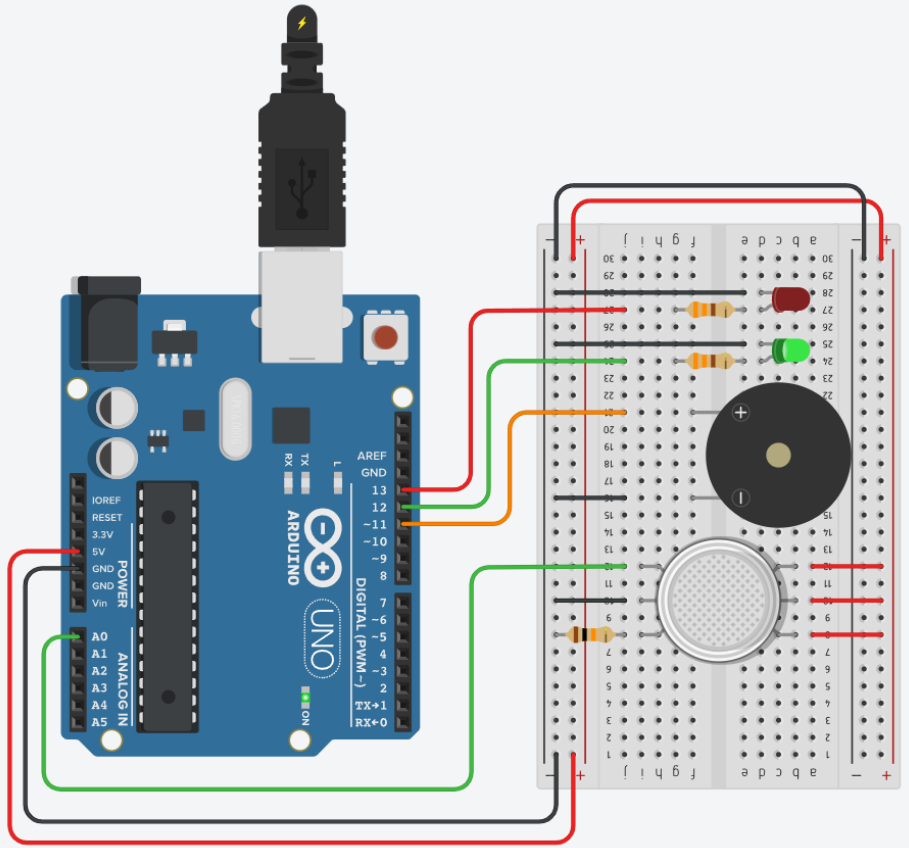
가스센서 – 가스누출시 LED와 알람 울리기

가스누출이 없을때에는 초록색 LED를 켜고, 가스연기가 가스센거 가까이 와서 값이 높아지면, 경고성인 빨간색 LED를 켠다. 그리고, 알람을 켜 주위에 알려주는 프로그램을 작성하자.



센서		핀번호
RED LED		13
GREEN LED		12
Buzzer		11
GAS sensor	A1	(-)극 (저항)
	H1	(-)극
	A2	A0
	B1,B2,H2	(+)극

가스가 누출되면 알람을 울려 알려주기



가스센서 – 가스누출시 LED와 알람 울리기

가스누출이 없을때에는 초록색 LED를 켜고, 가스연기가 가스센거 가까이 와서 값이 높아지면, 경고성인 빨간색 LED를 켠다. 그리고, 알람을 켜 주위에 알려주는 프로그램을 작성하자.

```
int gas_sensor = A0;
int Rled = 13;
int Gled = 12;
int buzzerPin = 11;
int val = 0;

int alarm[] = {523, 1046, 2093};

void setup()  {
  Serial.begin(9600);
  pinMode(Rled,OUTPUT);
  pinMode(Gled,OUTPUT);
}
```

```
void loop()  {
  val = analogRead(gas_sensor);
  Serial.println(val);
  if(val > 500)     {
    digitalWrite(Gled, LOW);
    digitalWrite(Rled, HIGH);
    play_alarm();
  }
 else  {
    digitalWrite(Gled, HIGH);
    digitalWrite(Rled, LOW);
  }
}
```

```
 void play_alarm()
{
    for (int i = 0; i < 3; i++) {
      tone(buzzerPin, alarm[i], 200);
      delay(200);
  }
}
```

10 학습정리

학습 정리

- 가스센서의 활용법과 틴커캐드에서 가스연기의 움직임을 알고 제어
- 가스센서와 LED, 부저를 활용하여 가스 누출시 감지시스템의 원리를 이해

11차시

IR센서와 리모컨으로 LED 켜기

학습 목표

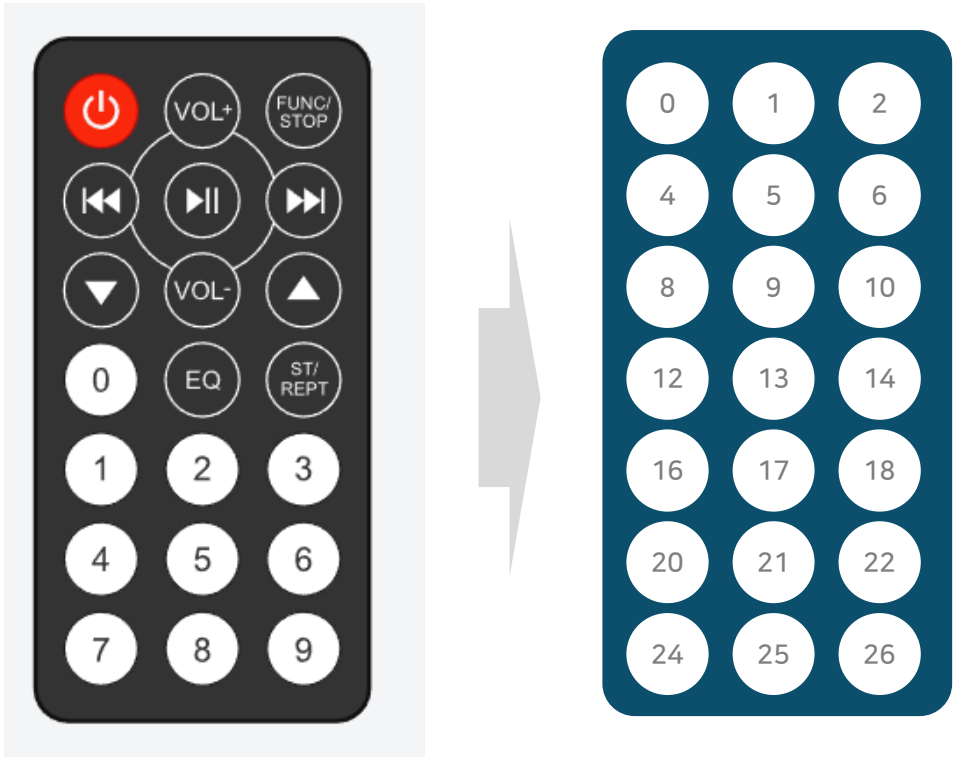
- IR 센서와 리모컨의 제어원리를 이해한다.
- 리모컨의 각 버튼의 입력 값을 확인한다.
- IR센서와 리모컨으로 LED 제어 실습한다.

학습 내용

- IR센서에서 리모컨과의 값 전달방법에 대한 프로그래밍 실습을 한다.
- IR신호를 활용하여 LED를 켜고, 버튼 1개의 ON/OFF 스위치 프로그램 활용방법을 익힌다.

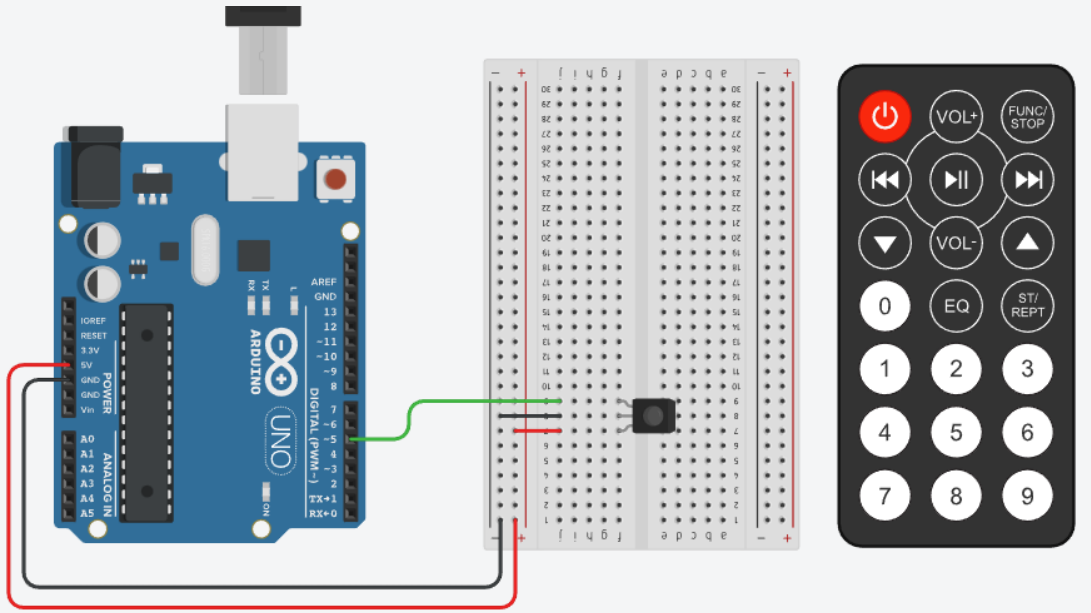
IR센서와 리모컨 제어하기

IR센서와 리모컨을 연결하고 이를 이용하여 해당 키값의 커캔드값을 시리얼 모니터로 확인해보자.



[리모컨 위치에 따른 command 값]

리모컨을 눌렀을때 해당 값을 확인하기



IR센서와 리모컨 제어하기

IR센서와 리모컨을 연결하고 이를 이용하여 해당 키값의 커캔드값을 시리얼 모니터로 확인해보자.

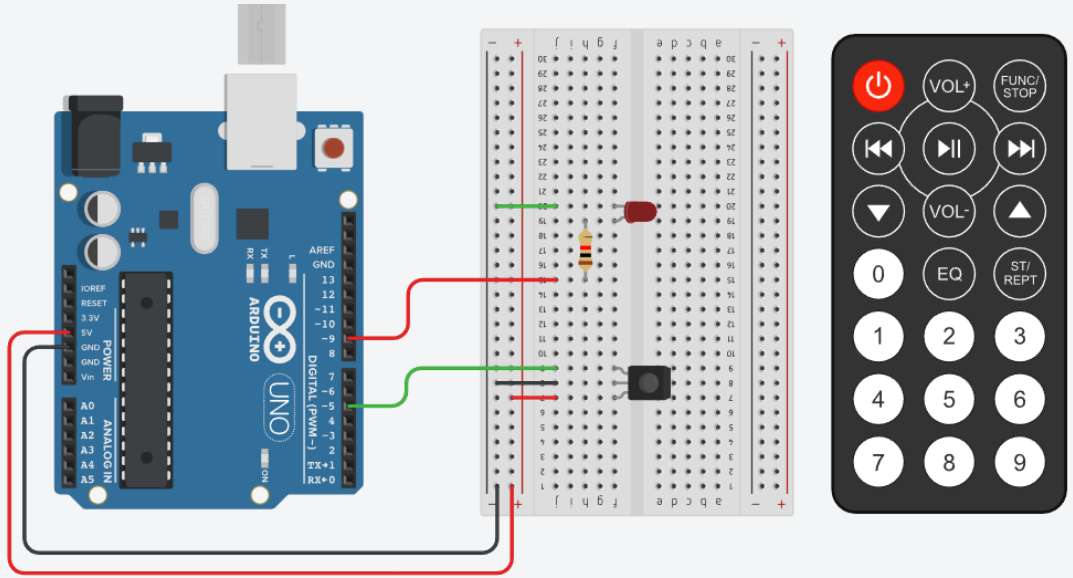
```
#include <IRremote.h>


#define IR_RECEIVE_PIN 5



void setup() {
  Serial.begin(9600);
  IrReceiver.begin(IR_RECEIVE_PIN);
}

void loop() {
  if (IrReceiver.decode()) {
    IrReceiver.resume();
    int command = IrReceiver.decodedIRData.command;
    Serial.println(command);
   }
 }
```

리모컨으로 LED켜기

IR센서와 리모컨을 연결하고 해당번호를 누르면 LED를 켜는 프로그램을 작성해보자.

```
#include <IRremote.h>
#define IR_RECEIVE_PIN 5
int Rled = 9;

void setup() {
  Serial.begin(9600);
  IrReceiver.begin(IR_RECEIVE_PIN);
  pinMode(Rled, OUTPUT);     // LED는 9번핀 사용
}

void loop() {
  if (IrReceiver.decode()) {
    IrReceiver.resume();
    int command = IrReceiver.decodedIRData.command;
    Serial.println(command);
    if(command == 16)
      digitalWrite(Rled, HIGH);
    else
      digitalWrite(Rled, LOW);
    delay(100);
    }
 }
```

리모컨으로 LED 조명을 켜고 끄기



리모컨으로 LED켜기- 토글키 만들기

IR센서와 리모컨을 연결하고 해당번호를 누르면 LED를 켜는 프로그램을 작성해보자.

```
#include <IRremote.h>
#define IR_RECEIVE_PIN 5

int Rled = 9;
int flag = 0;

void setup() {
  Serial.begin(9600);
  IrReceiver.begin(IR_RECEIVE_PIN);
  pinMode(Rled, OUTPUT);     // LED는 9번핀 사용
}

void loop()
{
  if (IrReceiver.decode()) {
    IrReceiver.resume();
    int command = IrReceiver.decodedIRData.command;
```

```
if(command == 16 && flag == 0)
    {
      Serial.print(command);
      Serial.print(",");
      Serial.println(flag);
      digitalWrite(Rled, HIGH);
      flag = 1;
    }
    else if(command == 16 && flag == 1)
    {
      Serial.print(command);
      Serial.print(",");
      Serial.println(flag);
      digitalWrite(Rled, LOW);
      flag = 0;
    }
    delay(100);
  }
 }
```

11 학습정리

학습 정리

- IR센서와 리모컨의 라이브러리를 연결하는 방법에 대해 학습
- 리모컨의 IR센서 입력값으로 LED를 제어하고 토글키가 가능한 프로그래밍 실습

12차시

힘센서를 활용한 LED+부저+서보모터의 제어

학습 목표

- 휨센서와 휨센서값에 대해 이해하기
- 휨센서를 이용한 LED와 부저의 프로그래밍 방법에 대해 실습하기
- 휨센서를 이용한 서보모터 제어하기

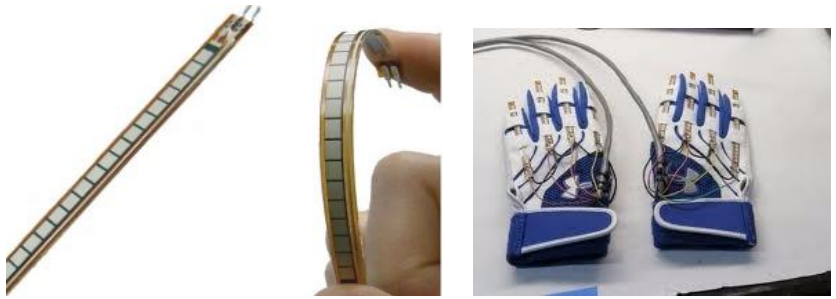
학습 내용

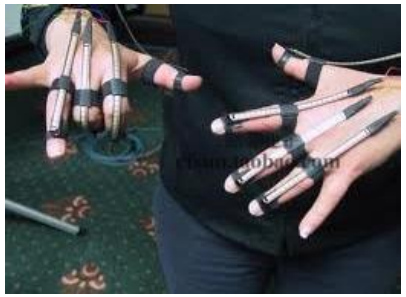
- 휨센서의 작동 원리와 휨센서값의 의미를 이해한다.
- 휨센서로 감지한 값을 이용하여 LED와 부저를 제어하는 방법을 익힌다.
- 휨센서로 측정한 값을 이용하여 서보모터 2개를 제어하는 방법에 대해 프로그래밍 실습한다.

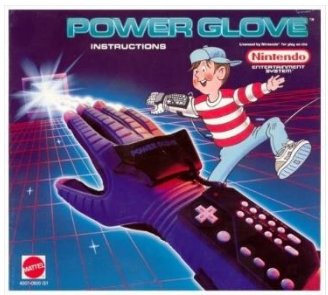
휨센서(Flex Sensor)

구부러짐에 따라 저항값이 변화하여, 변화된 저항값을 아날로그 입력으로 감지하여 구부러진 정도를 제어하거나 측정에 사용한다.

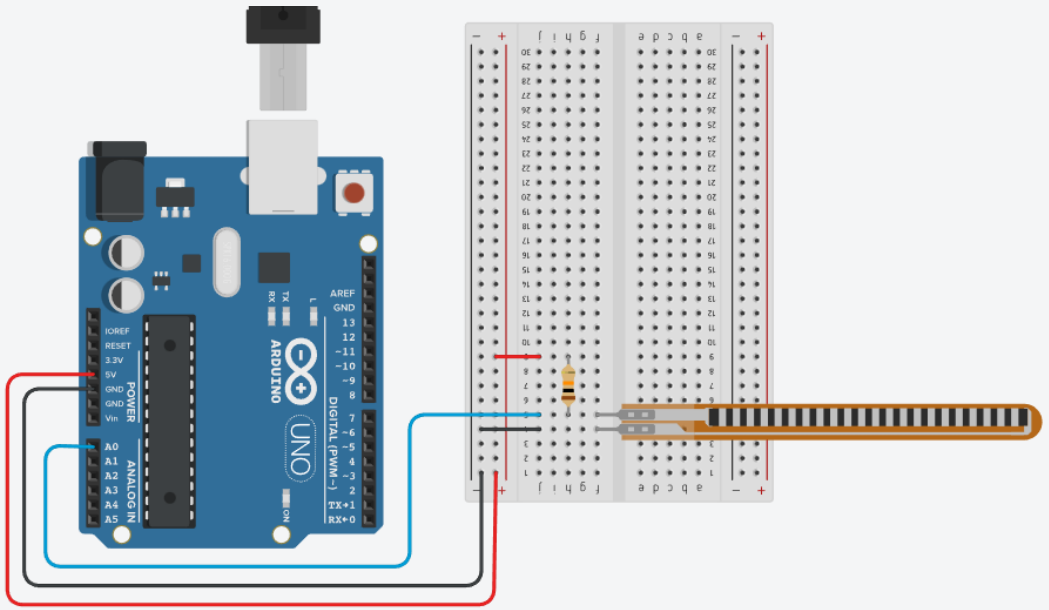
- Flex Sensor는 얇고 유연한 소재로 만들어져, 로봇장비, 웨어러블 디바이스 등 다양한 기기에 적용 가능
- 설치와 연결이 용이하며, 센서의 크기를 적절히 조절 가능하다.







휨센서 값 확인하기

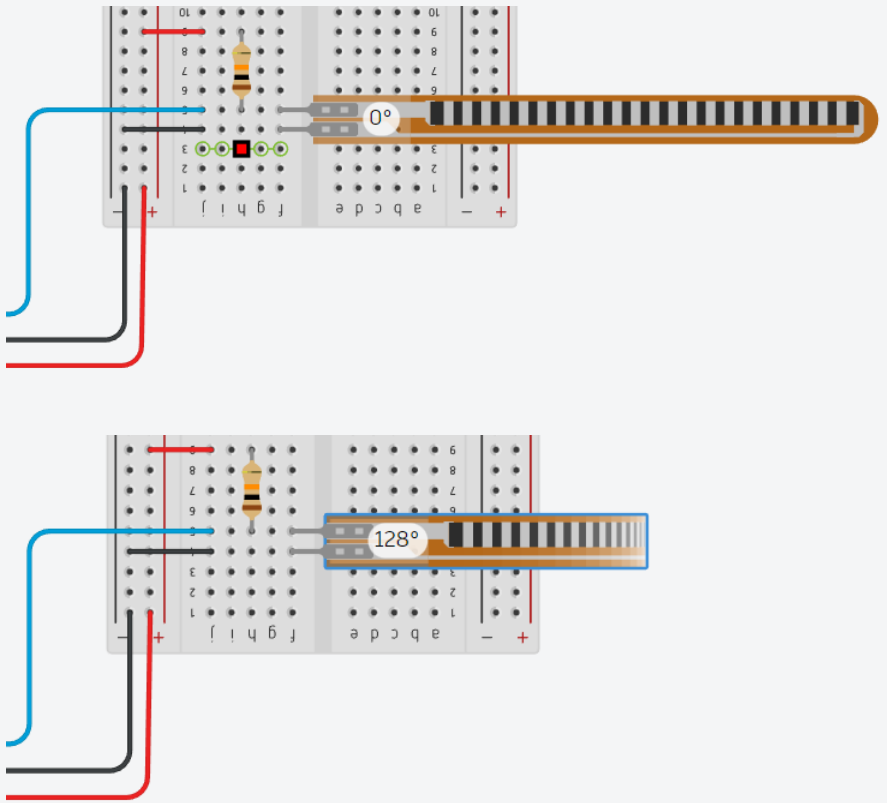


휨센서(Flex Sensor)- 값확인하기

LED의 (+)극과 (-)극을 구분하고 1초마다 깜박이는 프로그램을 작성해보자.ㄴ

```
int flex_sensor = A0;
void setup()
  {
    Serial.begin(9600);
  }

void loop()
 {
  int sensor = analogRead(flex_sensor);
  Serial.println(sensor);
  delay(100);
 }
```

휨센서를 구부려 값 확인하기



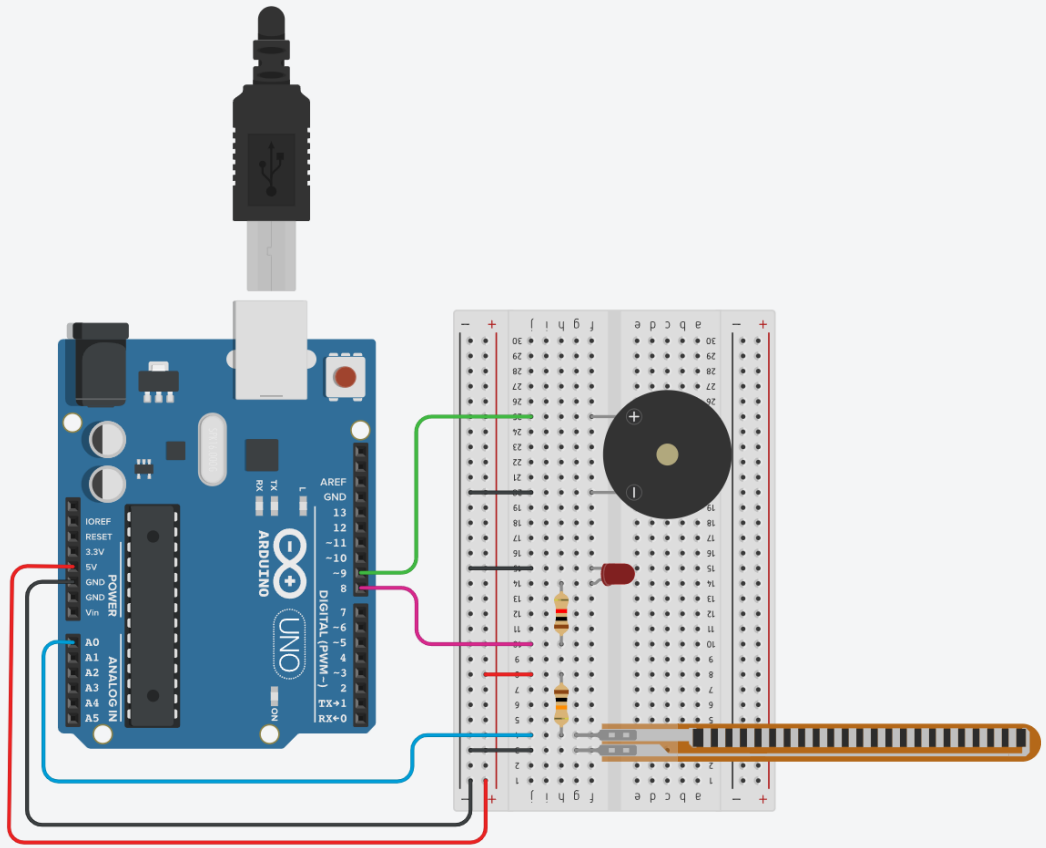
휨센서(Flex Sensor)- 구부러진 정도 확인하기

시리얼모니터에서 구부러지는 정도값을 확인하여 일정한 값 이상으로 구부러졌을때 부저와 LED를 켜보는 프로그래밍을 작성해보자.

```
int Rled = 8;
int buzzer = 9;
int flex_sensor = A0;

void setup()
 {
    Serial.begin(9600);
    pinMode(Rled, OUTPUT);
 }

void loop() {
  int sensor = analogRead(flex_sensor);
  Serial.println(sensor);
```

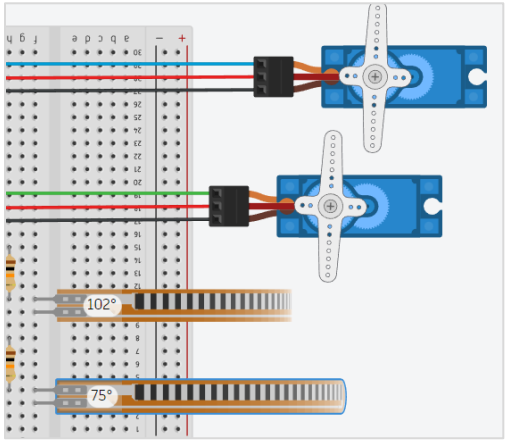
```
   if(sensor > 900)
    {
     //LED를 켜는 조건
     digitalWrite(Rled,HIGH);
     tone(buzzer,2093);
    }
   else
    {
     //LED를 끄는 조건
     digitalWrite(Rled,LOW);
     noTone(buzzer);
    }
  delay(100);
}
```

구부러지는 정도에 따라서 LED와 부저 울리기



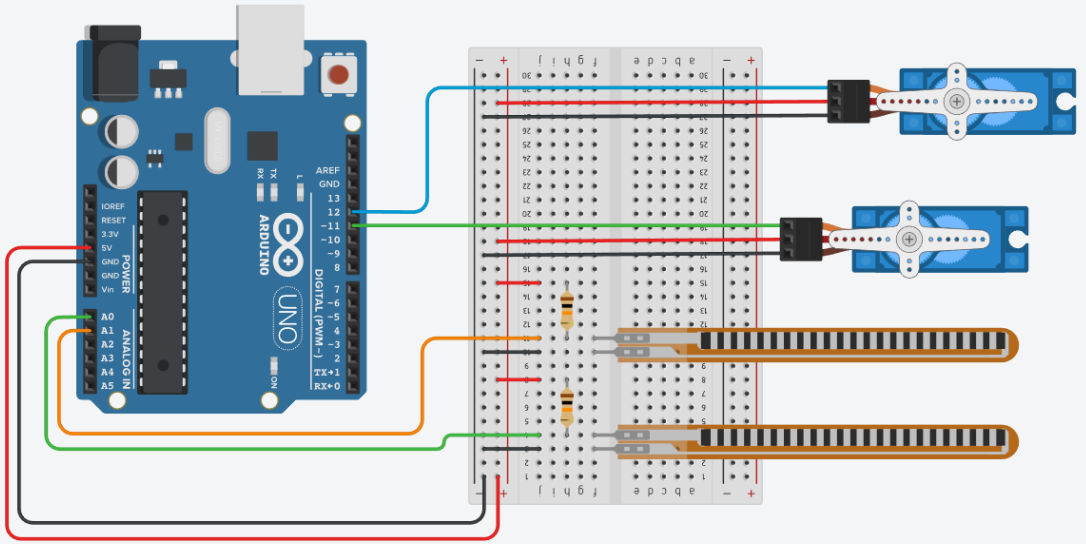
휨센서(Flex Sensor)- 서보모터움직이기

휘어진 정도값을 확인하여, 일정각도로 휘어지면 서보모터를 90로 움직이는 프로그램을 작성해보자.



센서	핀번호
서보모터 1	12
서모모터 2	11
휨센서 1	A0
휨센서 2	A1

휨정도에 따라서 서보모터 움직이기



휨센서(Flex Sensor)- 서보모터움직이기

- 여러개의 서보모터를 활용하면 센서 글러브와 같은 프로젝트가 가능하다.

```
#include <Servo.h>
Servo myservo1;
Servo myservo2;

int servo1 = 11;
int servo2 = 12;
int flex_sensor1 = A0;
int flex_sensor2 = A1;

void setup() {
  Serial.begin(9600);
  myservo1.attach(servo1);
  myservo2.attach(servo2);
  myservo1.write(0);
  myservo2.write(0);
}
```

```
void loop() {
  int sensor1 = analogRead(flex_sensor1);
  Serial.println(sensor1);

  if(sensor1 > 800)   {
    myservo1.write(90);
    delay(2000);
   }
  else  {
      myservo1.write(0);
   }
  delay(100);
```

```
  int sensor2 = analogRead(flex_sensor2);
  Serial.println(sensor2);

  if(sensor2 > 800)   {
     myservo2.write(90);
     delay(2000);
    }
   else  {
       myservo2.write(0);
    }
}
```

12 학습정리

학습 정리

- 휨센서의 값을 시리얼 모니터에서 확인하고 제어값 범위를 확인
- 휨센서 값 범위에 따라 LED로 확인하고 피에조 부저로 멜로디 출력을 실습
- 휨센서값을 활용해 서보모터의 움직임을 확인

13차시

시계세그먼트 출력과 카운트 다운 만들기

학습 목표

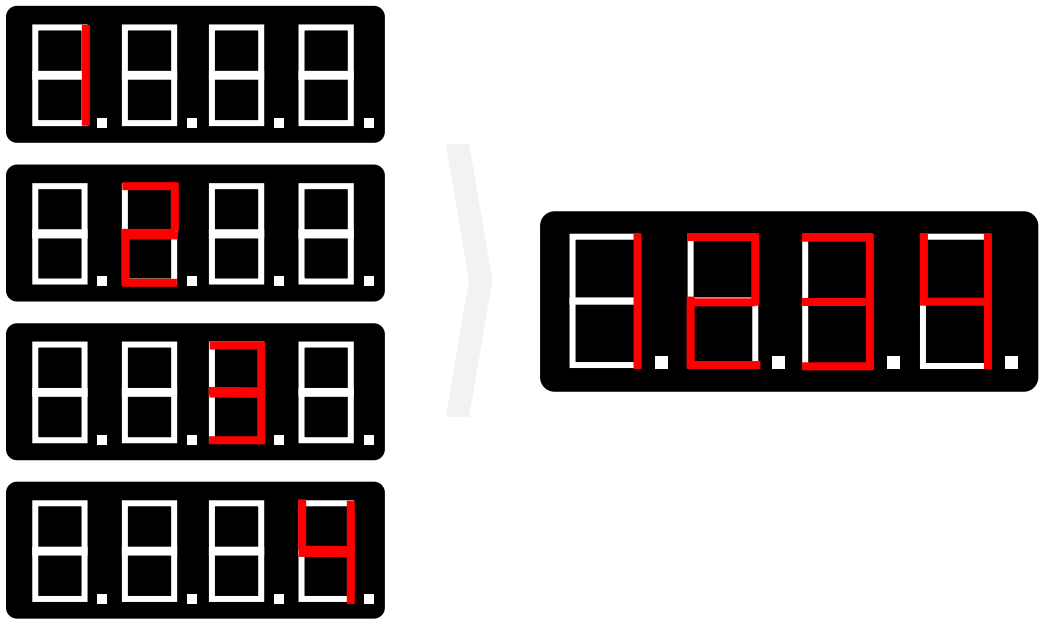
- 시계 세그먼트 디스플레이의 이해한다.
- 시계 세그먼트 회로도를 작성하고, 프로그래밍한다.
- 시계 세그먼트로 숫자 출력 실습한다.
- 시계 세그먼트로 카운트다운 구현한다.
- 시계 세그먼트를 활용하기 위한 라이브러리를 설치한다.

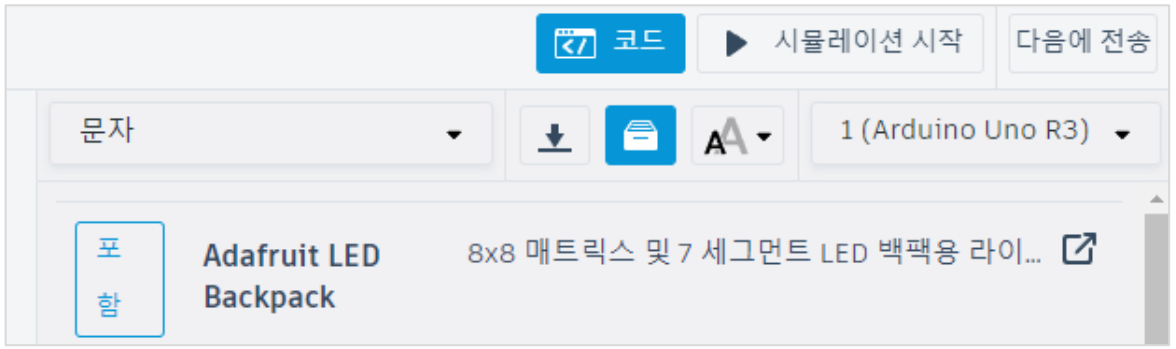
학습 내용

- 시계 세그먼트 디스플레이의 작동 원리와 구성을 이해한다.
- 시계 세그먼트 디스플레이를 사용하여 숫자를 출력하는 실습을 통해 익힌다.
- 시계 세그먼트 디스플레이를 활용하여 카운트다운을 구현하는 방법을 실습한다.

7-세그먼트 시계 디스플레이

- 각 세그먼트의 위치별 값이 빠르게 디스플레이스 되면서 1234의 값이 한번에 출력되는것 처럼 인식된다.



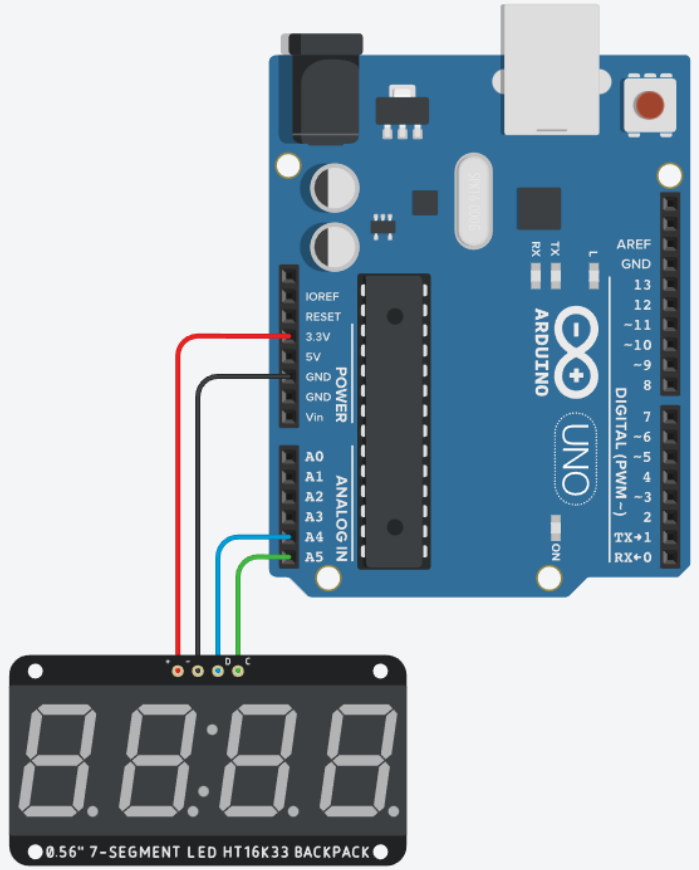


7-세그먼트 시계 디스플레이

7-세그먼트가 정상적으로 작동되는지 확인하고,
1234라는 숫자가 DEC(10진수)형태로 표시되도록 한다.

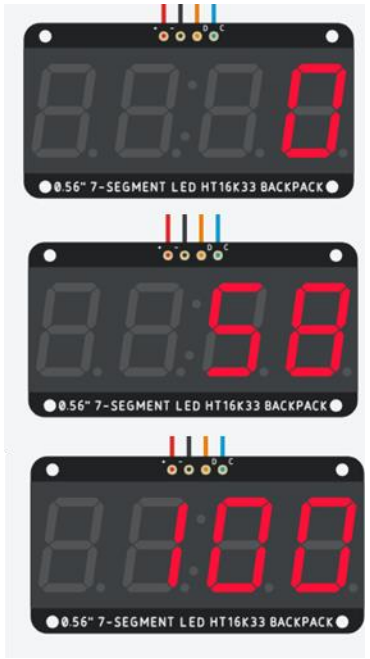
```
#include <Wire.h>
#include <Adafruit_LEDBackpack.h>
Adafruit_7segment matrix = Adafruit_7segment();

void setup() {
  Serial.begin(9600);
  Serial.println("7-Segment test");
  Wire.begin(); // I2C 통신 시작
  matrix.begin(113);
  // pass in the address 0x70
}

void loop()   {
  matrix.print(1234, DEC);
  matrix.writeDisplay();
}
```

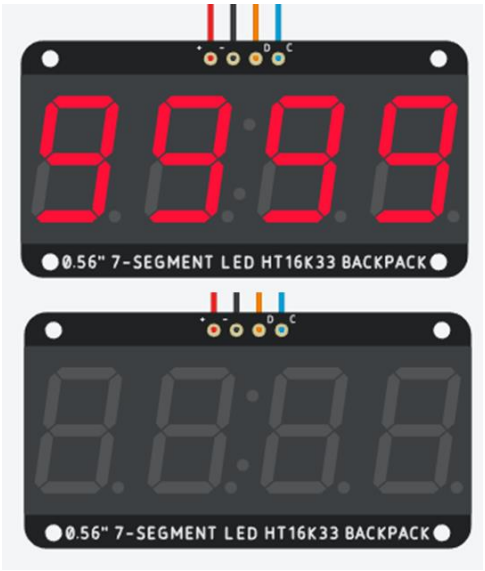
시계세그먼트에 숫자 출력하기



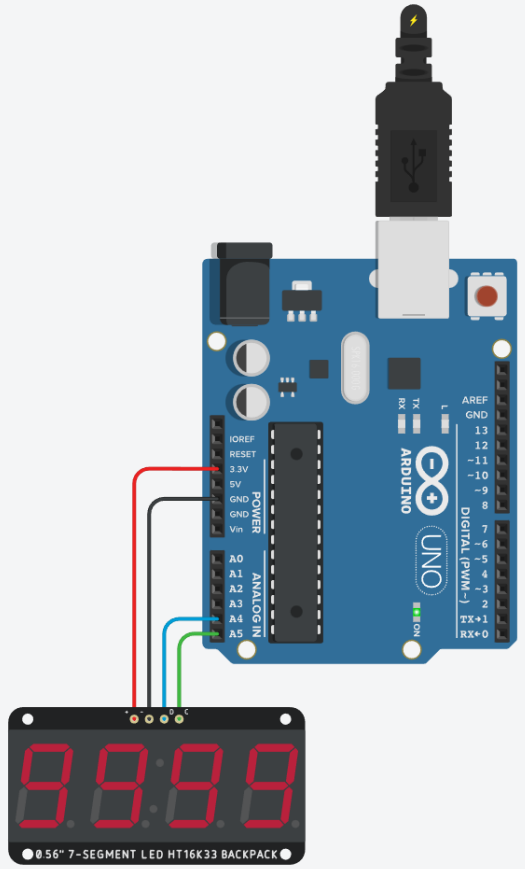
시계세그먼트 - 100까지 숫자세기

- 0부터 100까지 숫자를 카운트하고, 9999를 3회 깜빡이고 다시 숫자 카운트를 하는 예제를 작성해 보자.





시계세그먼트에 숫자세기



7-세그먼트 시계 - 100까지 숫자세기

- 0부터 100까지 숫자를 카운트하고, 9999를 3회 깜빡이고 다시 숫자 카운트를 하는 예제를 작성해 보자.

```
#include <Wire.h>
#include <Adafruit_LEDBackpack.h>
Adafruit_7segment matrix =
Adafruit_7segment();

void setup() {
  Serial.begin(9600);
  Serial.println("7-Segment test");
  Wire.begin(); // I2C 통신 시작
  matrix.begin(113);  // address 0x70
}
```

```
void loop()
{
  for (int counter = 0; counter <= 100; counter++)
   {
     matrix.print(counter);
     matrix.writeDisplay();
     delay(20);
    }
  delay(500);
  matrix_clear();
}
```

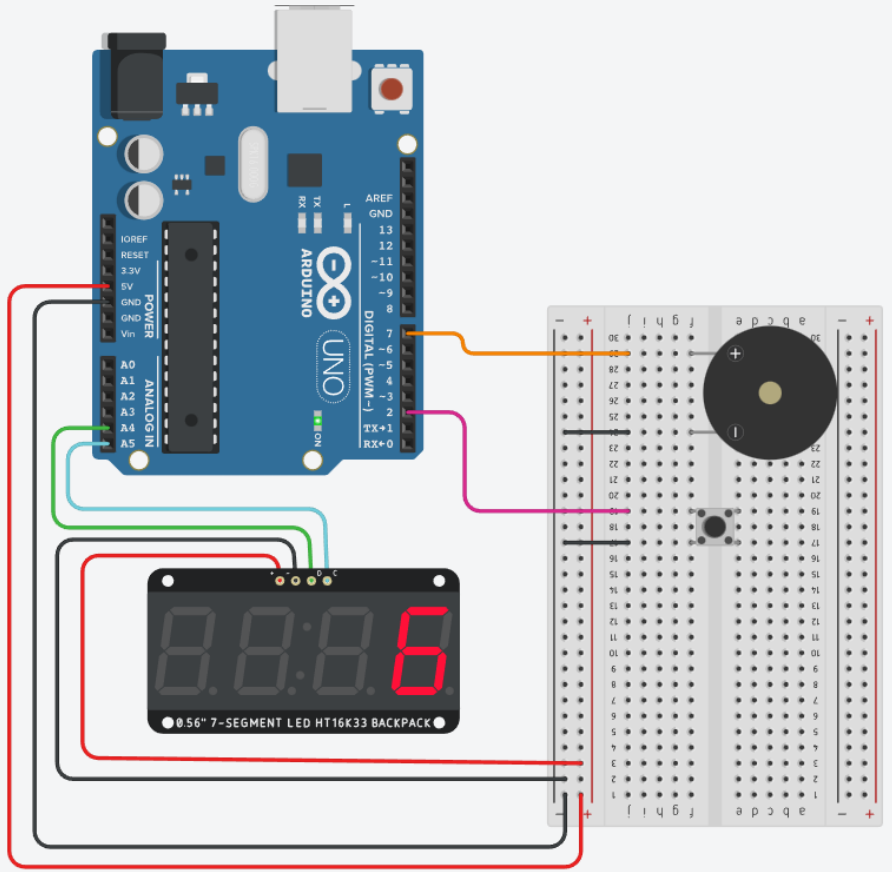
```
// 9999를 3회 깜박이게 하기 위해 반복문 사용
void matrix_clear()
{
  for (int i = 0; i < 3; i++)
    {
     matrix.print(9999);
       // 3자리 이하의 값으로 수정
     matrix.writeDisplay();
     delay(500);
     matrix.clear();
     matrix.writeDisplay();
     delay(500);
     }
    delay(500);
 }
```

7-세그먼트와 버튼&부저

버튼을 누르면 10부터 0까지 카운트 되고 카운트가 끝나면 부저가 울리는 프로그램을 작성해보자.

```
#include <Wire.h>
#include <Adafruit_LEDBackpack.h>
Adafruit_7segment matrix = Adafruit_7segment();
int buzzerPin = 7;
int buttonPin = 2;
int melody[] = {523, 1046, 2093};

void setup() {
  Serial.begin(9600);
  Serial.println("7 Segment Start!");
  Wire.begin(); // I2C 통신 시작
  matrix.begin(113);  // pass in the address 0x70
  pinMode(buttonPin, INPUT_PULLUP);
}
```

버튼을 눌러 카운트다운을 시작하기



7-세그먼트와 버튼&부저

버튼을 누르면 10부터 0까지 카운트 되고 카운트가 끝나면 부저가 울리는 프로그램을 작성해보자.

```
void loop()
{
  if (digitalRead(buttonPin) == false)
  {
     for (int counter =10; counter >= 0; counter--)
      {
          matrix.print(counter);
          matrix.writeDisplay();
          delay(1000);
       }
      play_melody();
  }
}
```

```
void play_melody()
{
    for (int i = 0; i < 3; i++) {
      tone(buzzerPin, melody[i], 200);
      delay(200);
  }
}
```

13 학습정리

학습 정리

- 7- 세그먼트 시계 디스플레이에 라이브러리를 설치
- 간단한 숫자를 디스플레이 실습
- 숫자를 디스플레이하기 위한 반복문 실습
- 버튼을 활용해 카운트 다운을 세고 부저울림

14차시

토양습도센서를 활용하여 스마트 팜 원리 이해

학습 목표

- 토양습도센서 값 확인하기
- 토양습도센서 값 범위에 따라 습도 (%)로 매핑하는 프로그래밍을 실습한다.
- 토양습도센서 값을 LCD 확인해 본다.
- 토양습도센서 값 모터 추가한다.

학습 내용

- 토양습도센서의 작동 원리와 값을 확인하는 방법을 이해한다.
- 토양습도센서로부터 읽어온 값을 LCD에 표시하는 방법을 익힌다.
- 토양습도센서 값을 기반으로 모터를 추가하여 제어하는 방법을 익힌다.

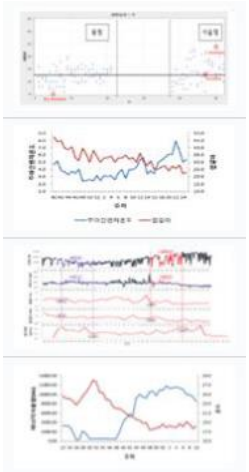
토양습도센서- 스마트팜 관리하기

스마트팜은 각종 센서로 온도.습도, 이산화탄소, 일사량을 측정하여 토양수분, 영양공급 상태를 확인하고 조절하여 최적한 생육환경을 만드는 ICT기반 농업기술이다.

토양습도센서/환기팬/워터펌프

<센서 등 ICT 기자재가 설치된 스마트팜 온실>

출처 : 전자신문



출처 : 농사로

```cpp
map(value, fromLow, fromHigh, toLow, toHigh)
```

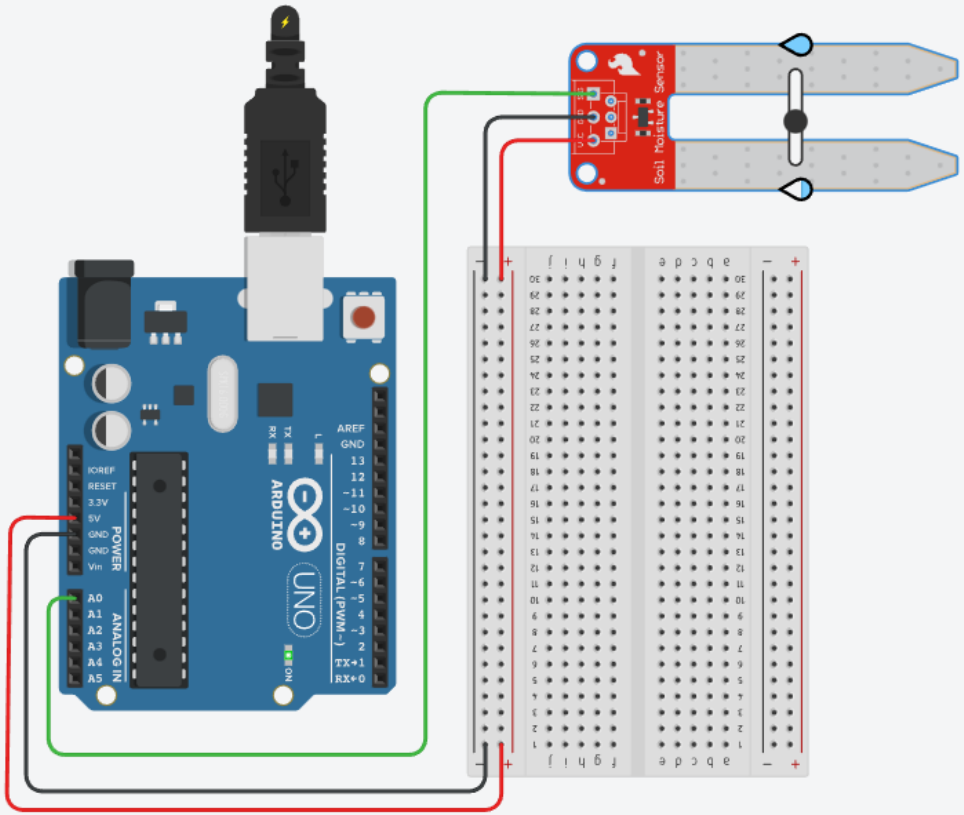
- value: 변환할 값입니다.
- fromLow: 입력 범위의 최솟값입니다.
- fromHigh: 입력 범위의 최댓값입니다.
- toLow: 출력 범위의 최솟값입니다.
- toHigh: 출력 범위의 최댓값입니다.

토양습도센서

토양습도센서의 수분량을 감지하여 시리얼 모니터에서 값을 확인하고, 수분량에 따라 전기적인 특성이 변화하는 값을 이해하자.

```
int Moisture = A0;      // 수분센서를 아닐로그 A0핀에 할당한다.

Void setup() {
  Serial.begin(9600);
}


Void loop()  {
//수분 선언
  int value = analogRead(Moisture);
// 수분센서에서 아날로그값을 읽어 value 변수에 저장
  int valueper = map(value, 0, 876, 0, 100);
// 수분센서의 아날로그값을 0부터 876까지의 범위에서 0부터 100으로 매핑하여
// valueper 변수에 저장
// 이렇게 함으로써 수분값을 백분율로 변환하여 표시한다.
  Serial.println(value);
}
```

토양습도센서 수분 값 확인하기

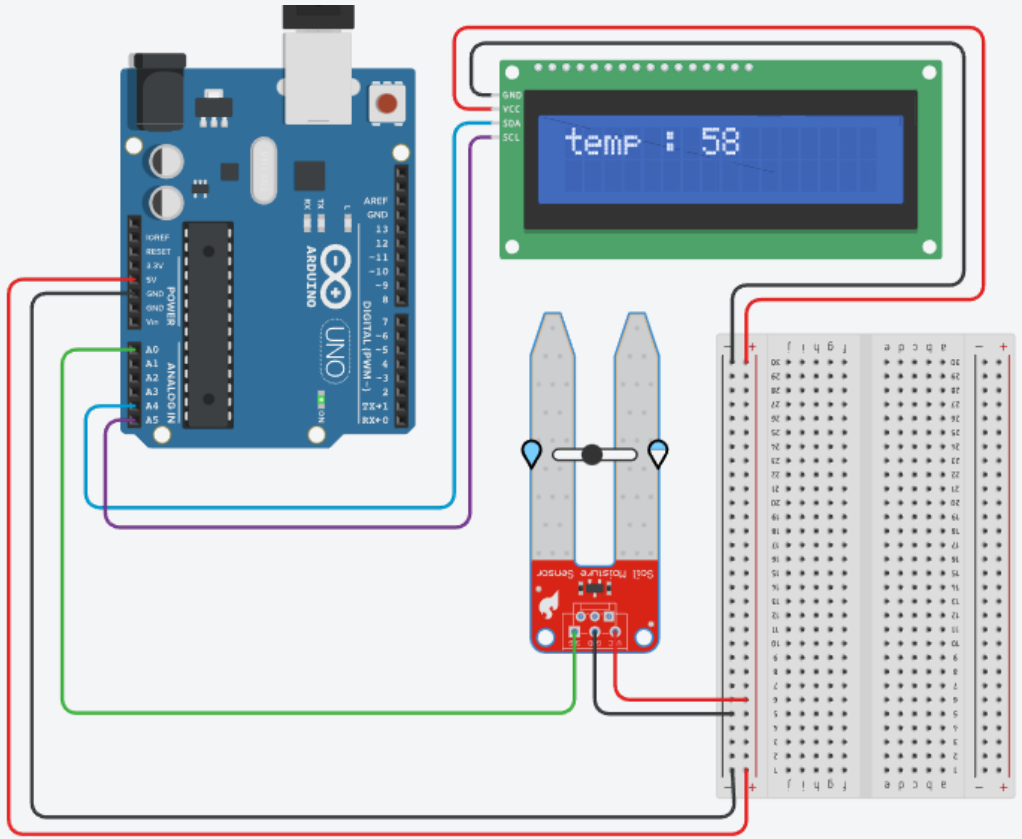


토양습도센서 – LCD 출력

변환된 수분값 백분율을 LCD에 출력하여 언제든지 변환하는 값을 확인가능하도록 코딩해보자.

- value: 변환할 값입니다.
- fromLow: 입력 범위의 최솟값입니다.
- fromHigh: 입력 범위의 최댓값입니다.
- toLow: 출력 범위의 최솟값입니다.
- toHigh: 출력 범위의 최댓값입니다.
- map() 함수는 value 값을 입력 범위(fromLow ~ fromHigh)에서 출력 범위(toLow ~ toHigh)로 선형 변환하여 반환한다.
- 예를 들어, 위의 코드에서는 value 값을 0에서 876의 범위에서 0에서 100의 범위로 변환하고 있다.
- 즉, analogRead() 함수를 통해 읽어온 아날로그 값인 value는 0부터 876 사이의 값일 것이다. 이 값을 map() 함수를 사용하여 0부터 100 사이의 값으로 변환하고, 변환된 값을 valueper 변수에 저장합니다. 이렇게 함으로써 valueper 변수에는 0에서 100 사이의 수분 값이 저장되게 된다.

토양습도센서 수분 값을 LCD에서 확인하기



토양습도센서

시리얼 모니터에서 출력하는 함수 코딩부분을 삭제하고, LCD 위치를 설정하고, 저장된 수분값을 LCD에 출력하는 코딩을 작성하자.

```
#include <LiquidCrystal_I2C.h>


LiquidCrystal_I2C lcd(32,16,2);
int Moisture= A0;


void setup()
  {
    Serial.begin(9600);
    lcd.init();
    lcd.backlight();
  }
```

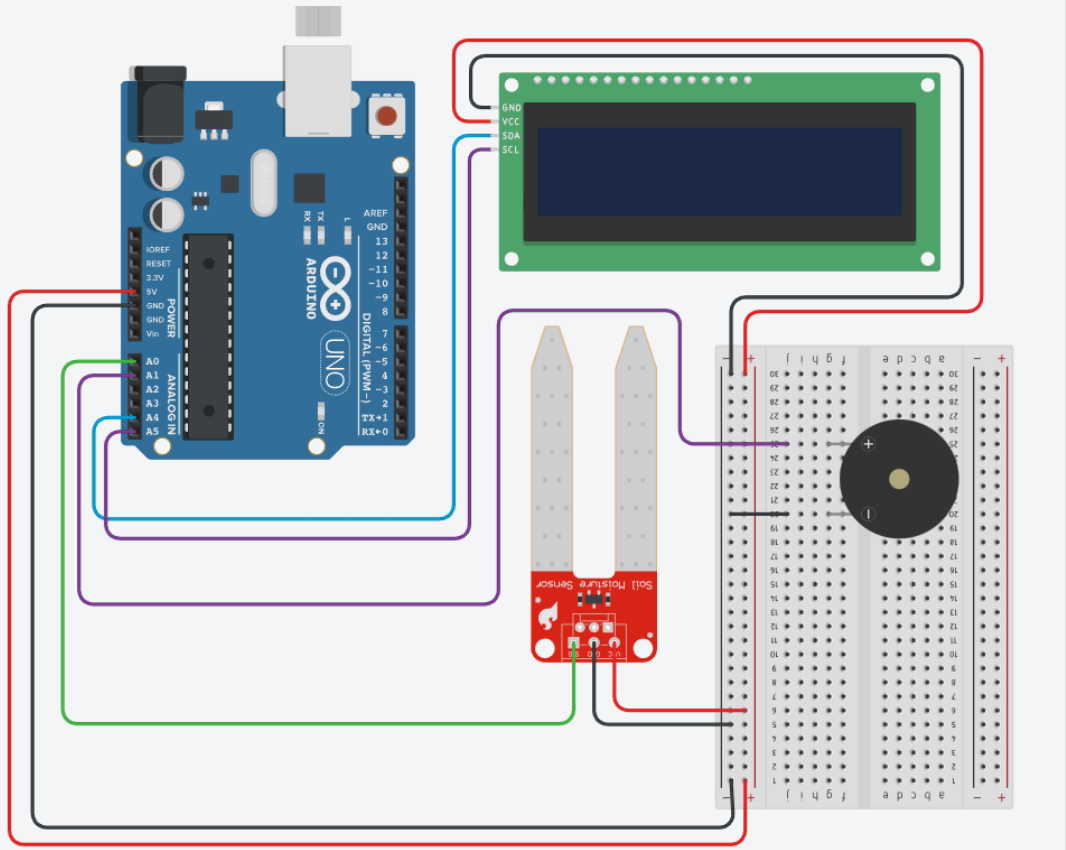
```
void loop()
 {
int value = analogRead(Moisture);
  int valueper = map(value, 0, 876, 0, 100);
  lcd.setCursor(0,0);
  lcd.print("temp : ");
  lcd.print(valueper);
  lcd.print("%");
  delay(1000);
  Serial.println(value);
  }
```

토양습도센서 – LCD 출력+알림울리기

변환된 수분값 백분율을 LCD에 출력하고 물부족인 습도가 50% 이하이면 알람을 울려서 알려주는 프로그램을 작성해보자.

센서		핀번호
LCD	SDA	A4
	SCL	A5
토양습도센서		A0
피에조부저		A1

토양습도센서의 값이 50%이하이면 물부족 알림울리기



토양습도센서 – LCD 출력+알림울리기

변환된 수분값 백분율을 LCD에 출력하고 물부족인 습도가 50% 이하이면 알람을 울려서 알려주는 프로그램을 작성해보자.

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(32,16,2);
int Moisture= A0;
int buzzer = A1;

int alarm[] = {523, 1046, 2093};



void setup()
  {
    Serial.begin(9600);
    lcd.init();
    lcd.backlight();
  }
```

```
void loop()
 {
  //수분 선언
  int value = analogRead(Moisture);
  int valueper = map(value, 0, 876, 0, 100);
  lcd.setCursor(0,0);
  lcd.print("temp : ");
  lcd.print(valueper);
  lcd.print("%");

if(valueper < 50)
  {
    play_alarm();
  }
```

```
  delay(1000);
  Serial.println(value);
  }



void play_alarm()
{
   for (int i = 0; i < 3; i++) {
      tone(buzzer, alarm[i], 200);
      delay(200);
  }
}
```

14 학습정리

학습 정리

- 토양습도센서의 동작 원리와 값 확인 방법 학습
- 토양습도센서 값 표시를 위한 LCD 제어 방법 학습
- 토양습도센서 값을 활용한 모터 제어 방법 학습

15차시

여러 개의 네오픽셀과 네오픽셀 Jewel 의 제어

학습목표, 학습내용

학습 목표

- 네오픽셀 라이브러리를 추가하여 4개짜리 동작 이해한다.
- 네오픽셀 8개 동작하기 위해 입력과 출력 회로도 이해하기
- 네오픽셀 jewel은 원형 LED로 제어할 수 있다.

학습 내용

- 네오픽셀의 동작 원리와 4개 네오픽셀을 활용한 제어 방법을 이해한다.
- 네오픽셀을 활용한 LED 8개를 제어하는 방법을 익힌다.
- 네오픽셀 jewel을 활용한 다채로운 효과 제어 방법을 익힌다.

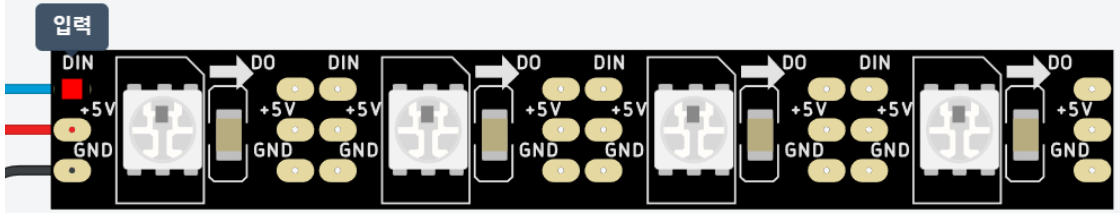
NeoPixel 스트립 4개짜리 켜기

네오픽셀 4개짜리 스트립은 DIN(data In)과 DO(data out)위치를 확인하고 PWM 핀에 연결하여 다양한 색상 및 효과를 제어할 수 있다.

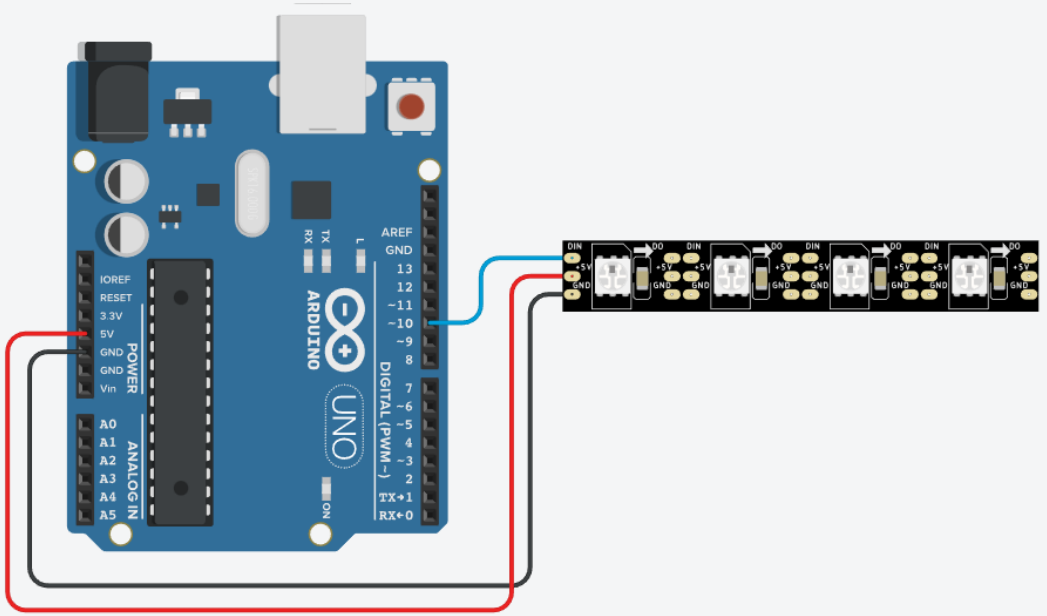
- 네오픽셀은 제조사 별로 다르지만, 틴커캐드에서는 Adafruit사의 라이브러리를 사용하여 제어할 수 있다.

포함	NeoPixel	NeoPixel LED 제어

- DIN과 DO핀 위치를 확인해서 연결하고, setPixelColor()함수의 매개변수를 확인하여 해당 값을 제어할 수 있다.



스트립 4개 네오픽셀 제어하기



NeoPixel 스트립 4개짜리 켜기

네오픽셀 스트립 4 센서를 제어해 보자.

```
#include <Adafruit_NeoPixel.h>

int numPixels = 4; //스트립 개수
int pixelsPin = 10; //PWM 핀번호 사용

Adafruit_NeoPixel pixels(numPixels, pixelsPin, NEO_GRB + NEO_KHZ800);
// (NEO_GRB + NEO_KHZ800): 색상 형식과 통신 속도를 지정

void setup()  {
  Serial.begin(9600);
  pixels.begin();
}
```

NeoPixel 스트립 4개짜리 켜기

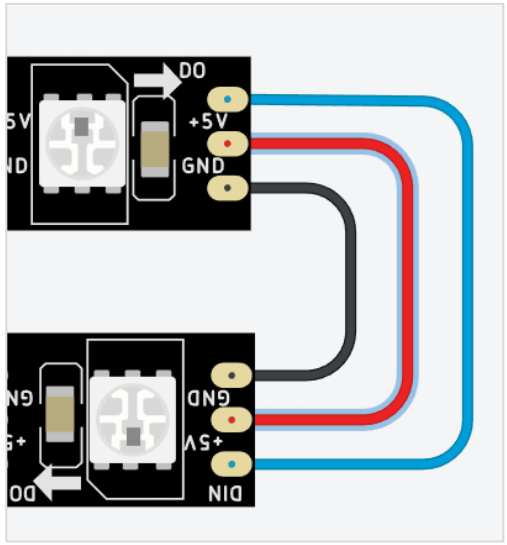
네오픽셀 스트립 4 센서를 제어해 보자.

```
void loop()
{
  pixels.clear();
  pixels.show();
  delay(100);

  pixels.setPixelColor(0,255,0,0);
  pixels.show();
  delay(200);

  pixels.setPixelColor(1,0,255,0);
  pixels.show();
  delay(200);
```

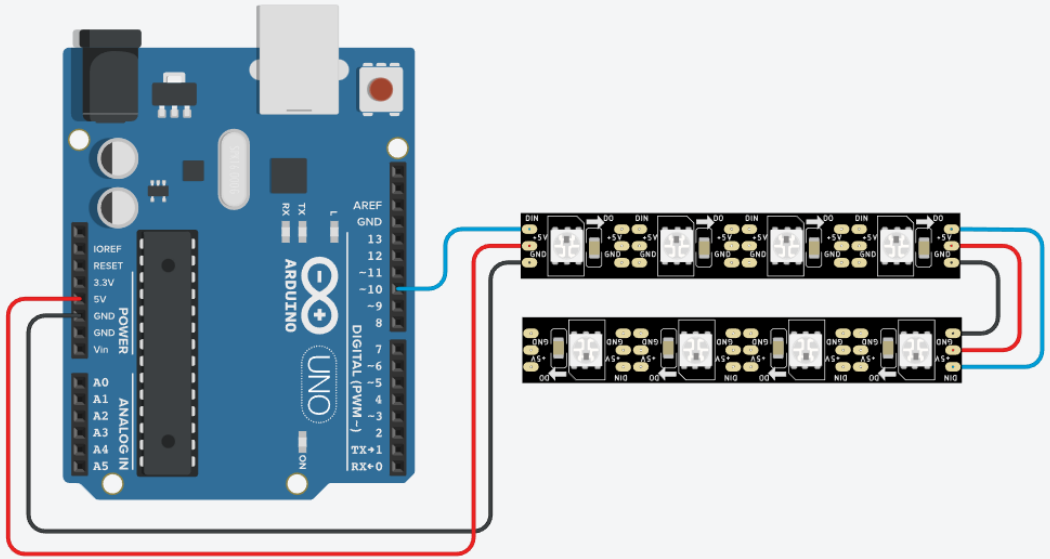
```
  pixels.setPixelColor(2,0,0,255);
  pixels.show();
  delay(200);

  pixels.setPixelColor(3,0,255,255); //청록
  pixels.show();
  delay(200);
}
```

NeoPixel 스트립 8개짜리 켜기

네오픽셀 8개짜리 스트립은 DIN(data In)과 DO(data out)위치를 확인하여 아래 그림처럼 연결한다.



8개의 스트립 LED켜기



NeoPixel 스트립 8개짜리 켜기

네오픽셀 스트립 4 센서를 제어해 보자.

```
#include <Adafruit_NeoPixel.h>

Adafruit_NeoPixel pixels(8, 10, NEO_GRB + NEO_KHZ800);
// (NEO_GRB + NEO_KHZ800): 색상 형식과 통신 속도를 지정
uint32_t yellowPixel = pixels.Color(255, 255, 0);
uint32_t magentaPixel = pixels.Color(255, 0, 255);
uint32_t cyanPixel = pixels.Color(0, 255, 255);

void setup()
{
  Serial.begin(9600);
  pixels.begin();
}
```

```
void loop()  {
  pixels.clear();
  pixels.show();
  delay(100);
  pixels.setPixelColor(0,255,0,0);
  pixels.show();
  delay(200);
  pixels.setPixelColor(1,0,255,0);
  pixels.show();
  delay(200);
  pixels.setPixelColor(2,0,0,255);
  pixels.show();
  delay(200);
```

NeoPixel 스트립 8개짜리 켜기

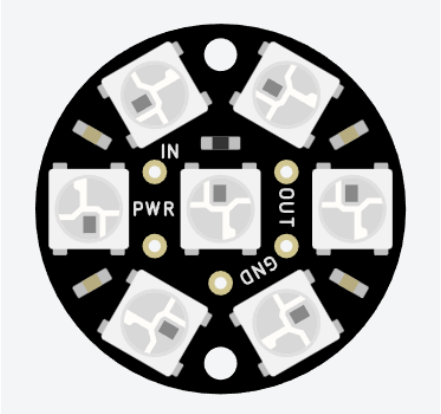
네오픽셀 스트립 4 센서를 제어해 보자.

```
pixels.setPixelColor(3,0,255,255); //청록
  pixels.show();
  delay(200);

  pixels.setPixelColor(4,yellowPixel);
  pixels.show();
  delay(200);

  pixels.setPixelColor(5,magentaPixel);
  pixels.show();
  delay(200);
```

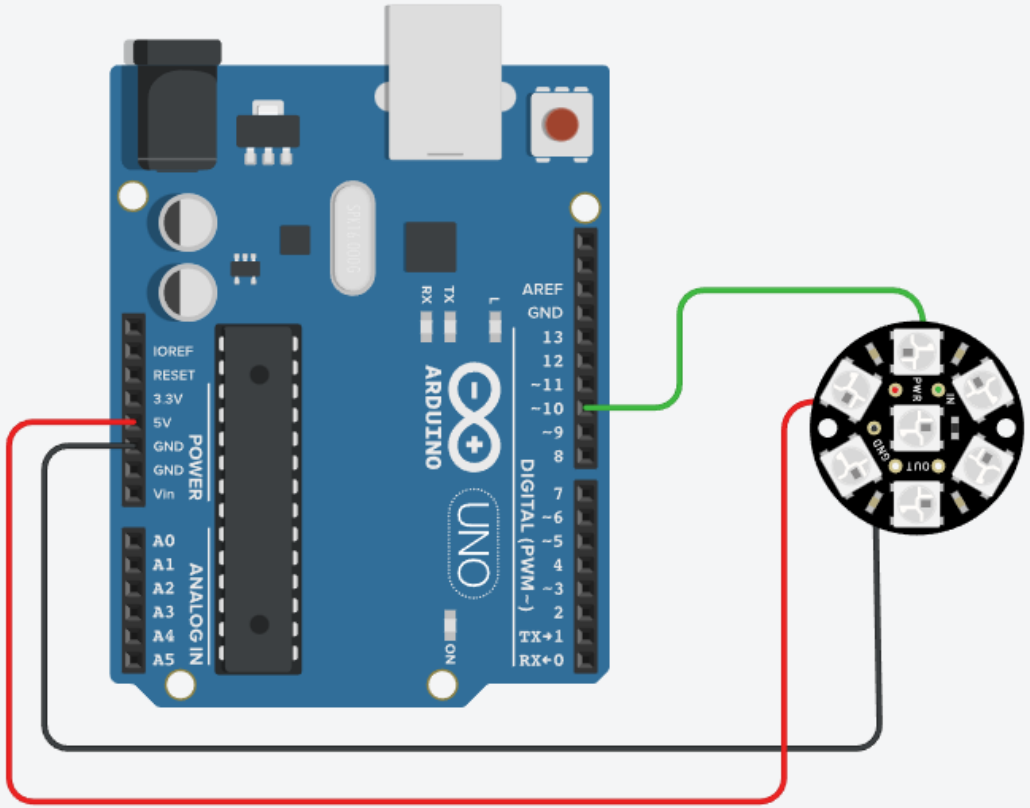
```
pixels.setPixelColor(6,cyanPixel);
  pixels.show();
  delay(200);
pixels.setPixelColor(7,255,255,255);
  pixels.show();
  delay(200);
}
```

NeoPixel Jewel

- Adafruit의 네오픽셀 LED 모듈의 일종으로, 7개 , 16개, 24개의 개별적 LED가 원형으로 배열되어 있다.



7개 네오픽셀이 원형 배열



NeoPixel Jewel

네오픽셀 Jewel 제어하기

```
#include <Adafruit_NeoPixel.h>

int pixelsPin = 10;
int numPixels = 7;

Adafruit_NeoPixel pixels(numPixels, pixelsPin,
NEO_GRB + NEO_KHZ800);



void setup()
{
  pixels.begin();
}
```

```
void loop()
{
  pixels.clear();
  pixels.show();
  delay(100);

  for (int i=0; i<7; i++)
  {
  pixels.setPixelColor(i,255,0,0);
  pixels.show();
  delay(200);
  }
}
```

15 학습정리

학습 정리

- 네오픽셀의 동작 원리와 4개 네오픽셀 제어 방법 학습
- 네오픽셀을 이용한 8개 LED 제어 방법 학습
- 네오픽셀 jewel을 이용한 다양한 효과 제어 방법 학습

16차시

키패드로 숫자,문자 입력하고 패스워드 만들기

학습 목표

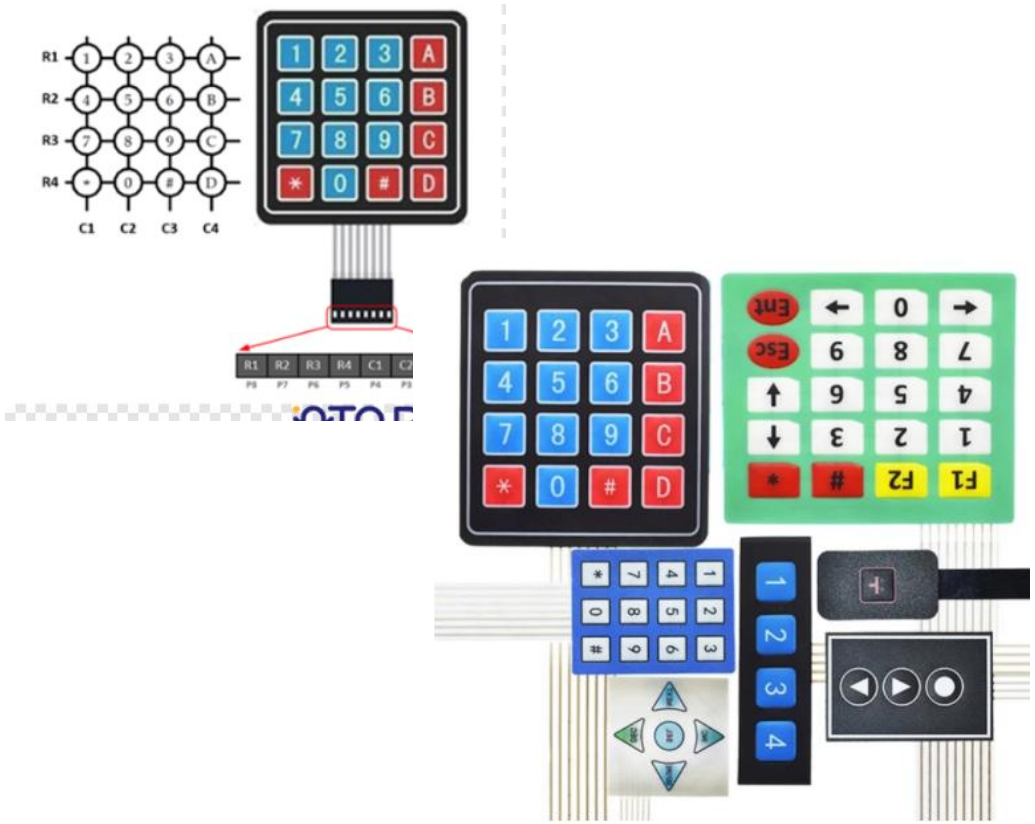
- 키패드의 행과 열을 이해하기
- 키패드를 눌렀을때 숫자와 문자 텍스트를 시리얼 모니터에서 출력해보기
- 패스워드를 생성해서 디지털 도어락의 제어 방법을 이해한다.

학습 내용

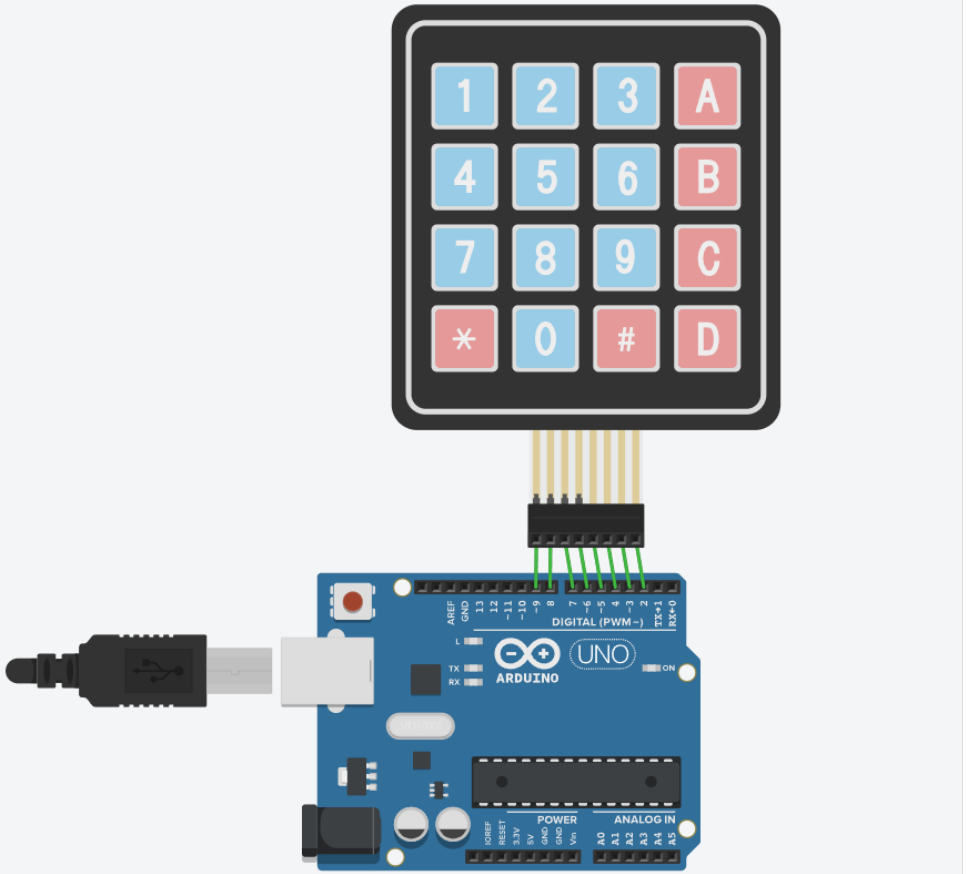
- 키패드의 배열 구조와 동작 원리를 이해한다.
- 키패드 입력을 텍스트로 출력하는 방법을 익힌다.
- 키패드를 이용하여 패스워드 생성 및 확인하는 방법을 익힌다.

키패드- 시리얼 값 확인하기

- 키패드는 보안시스템이나 비밀번호 인식 시스템등에 사용된다. 자동화시스템, 측정 및 제어장치 등 다양하게 활용가능하다.

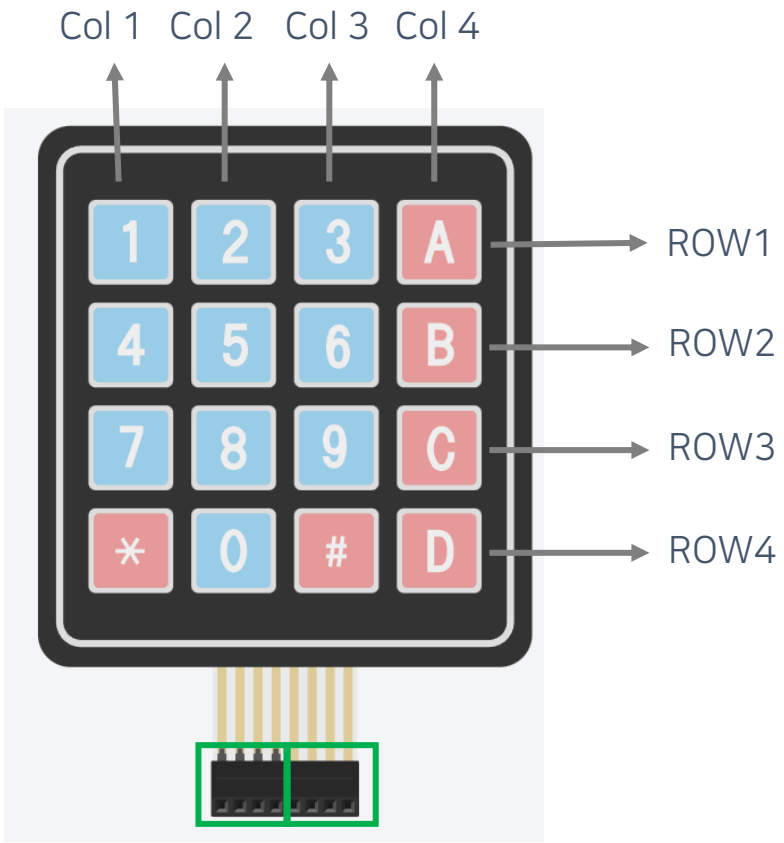


키패드 활용하기



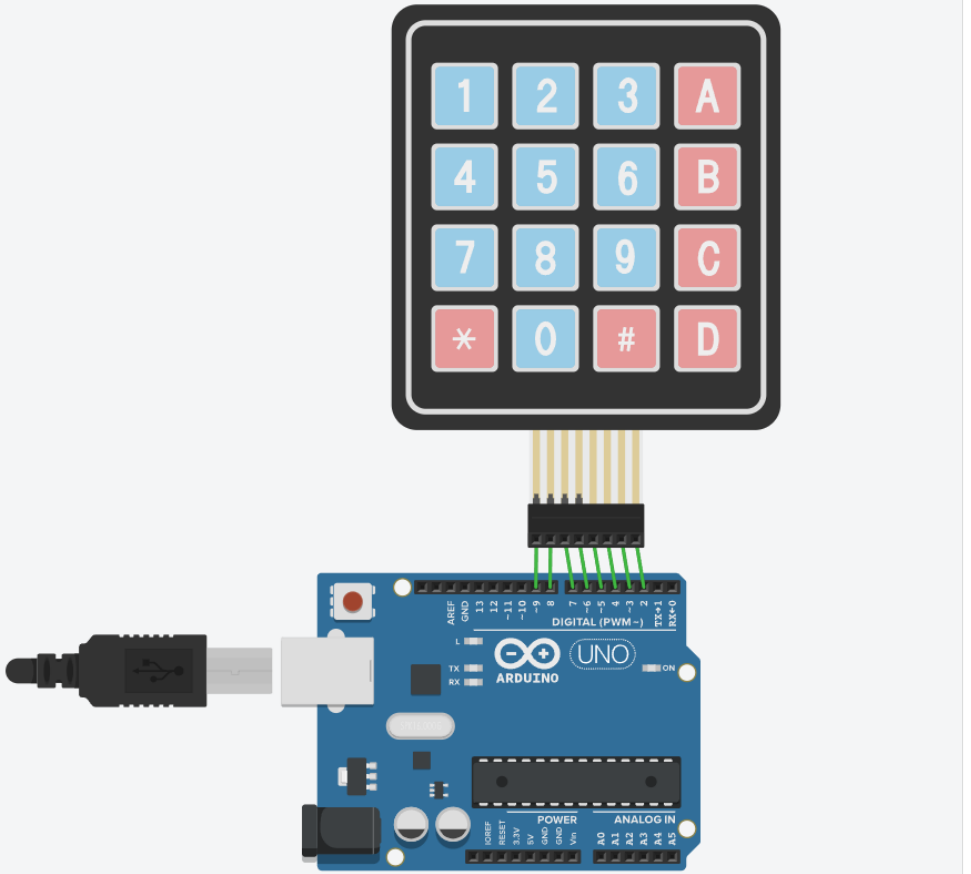
키패드- 시리얼 값 확인하기

- 키패드는 보안시스템이나 비밀번호 인식 시스템등에 사용된다. 자동화시스템, 측정 및 제어장치 등 다양하게 활용가능하다.



ROW1-ROW4 Col 1 – Col 4

키패드 활용하기



키패드 - 시리얼 값 확인하기

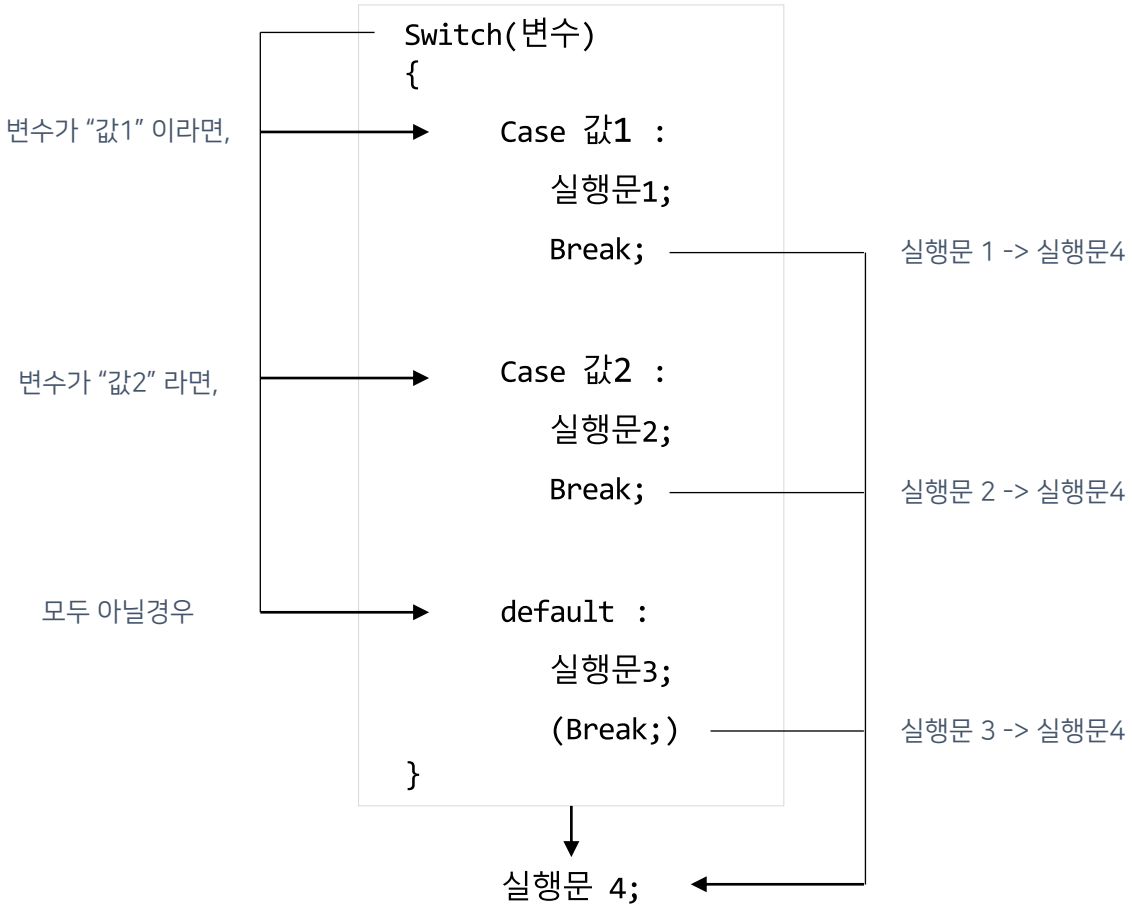
키패드의 단자와 연결된 아두이노 핀번호를 잘 확인하고, 눌려진 키값에 의해서 시리얼 모니터에 해당 키값이 나오는지 확인해 본다.

```
#include <Keypad.h>
const byte ROWS = 4;    // 행(rows) 개수
const byte COLS = 4;    // 열(columns) 개수
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'} };

// R1, R2, R3, R4 단자가 연결된 아두이노 핀 번호
byte rowPins[ROWS] = {9, 8, 7, 6};
// C1, C2, C3, C4 단자가 연결된 아두이노 핀 번호
byte colPins[COLS] = {5, 4, 3, 2};
// 키패드 객체를 생성, 행과 열 핀 번호, 그리고 키패드에 존재하는 키들의 매핑을 설정
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
```

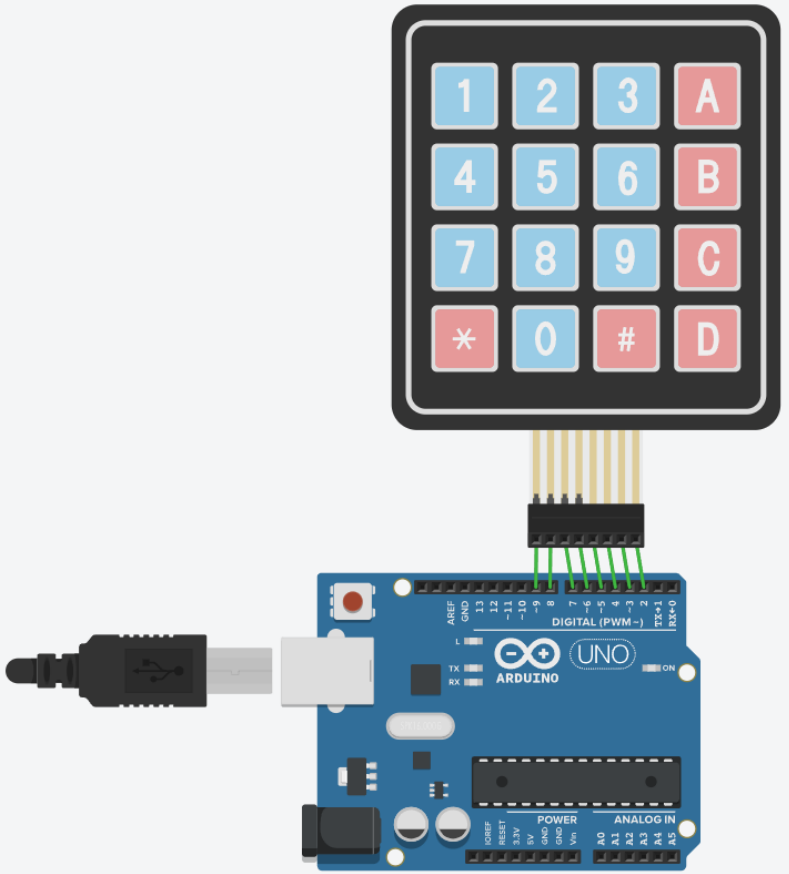
```
void setup() {
  Serial.begin(9600);
}

void loop() {
// 키패드에서 눌린 키값을 char타입변수인 key에 저장
  char key = keypad.getKey();

  if (key) // key에 값이 있을 경우, 아래코드 실행
   {
     Serial.println(key);
    }
}
```

키패드 – 텍스트 출력하기

키패드 중 A,B,C,D를 누르면 해당 텍스트가 출력되는 코딩을 작성해보자. Switch-case 문을 사용해보자.



키패드를 눌렀을때 텍스트 활용하기



키패드 – 텍스트 출력하기

시리얼 모니터에 각 문자 키패드를 눌렀을때 해당 메시지가 잘 출력되는지 확인한다.

```
#include <Keypad.h>

const byte ROWS = 4;    // 행(rows) 개수
const byte COLS = 4;    // 열(columns) 개수
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}  };
byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};
```

```
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS );

void setup() {
  Serial.begin(9600);
}
```

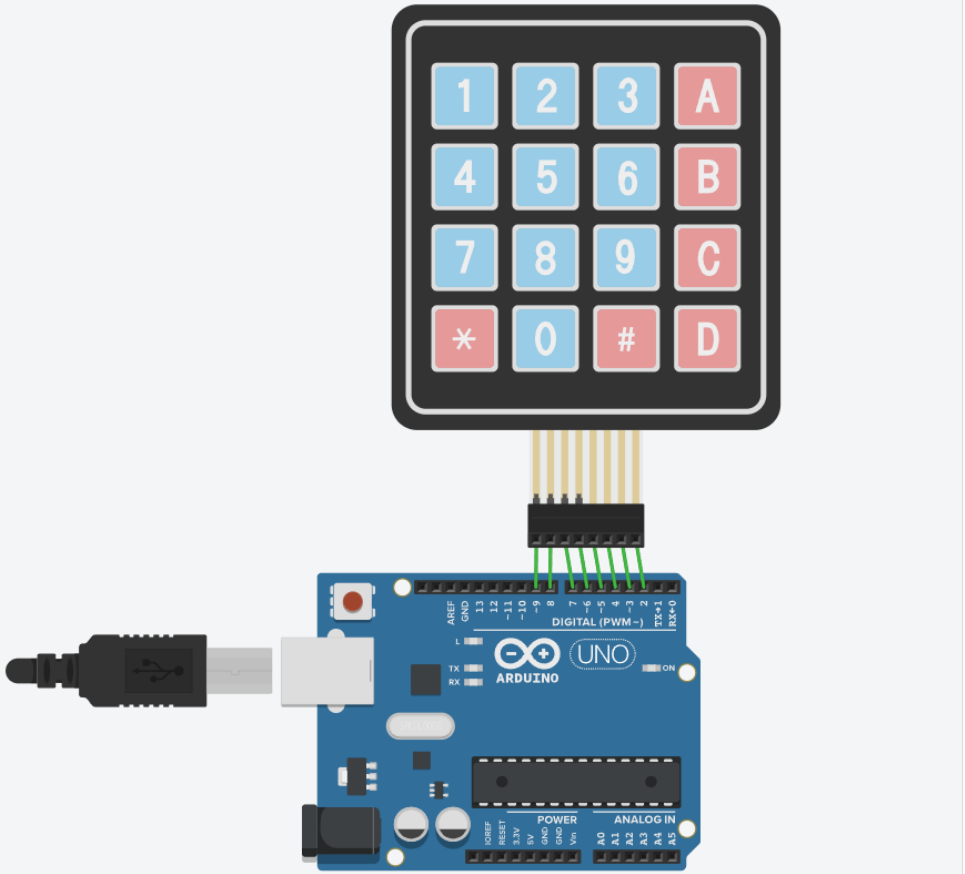
```
void loop() {
  char key = keypad.getKey();
  if (key)   {
    switch(key)    {
      case 'A' :
            Serial.println("Hello!");
            break;
      case 'B' :
            Serial.println("Arduino!");
            break;
      case 'C' :
            Serial.println("Simulator!");
            break;
       case 'D' :
            Serial.println("OKay!");
            break;
      default :
        Serial.println(key);
            break;
    }    }   }
```

키패드- 비밀번호 확인하기

- 아두이노 틴커캐드 키패드에서는 비밀번호를 입력한 후 #을 눌렀는지 확인하여 패스워드를 검증하는 코딩을 작성해보자.

****

키패드를 비밀번호 입력기로 활용하기



키패드- 비밀번호 확인하기

- 임의의 패스워드 “1234AD”를 누른후 #을 눌러 패스워드를 마무리하면 실행문을 실행한다.

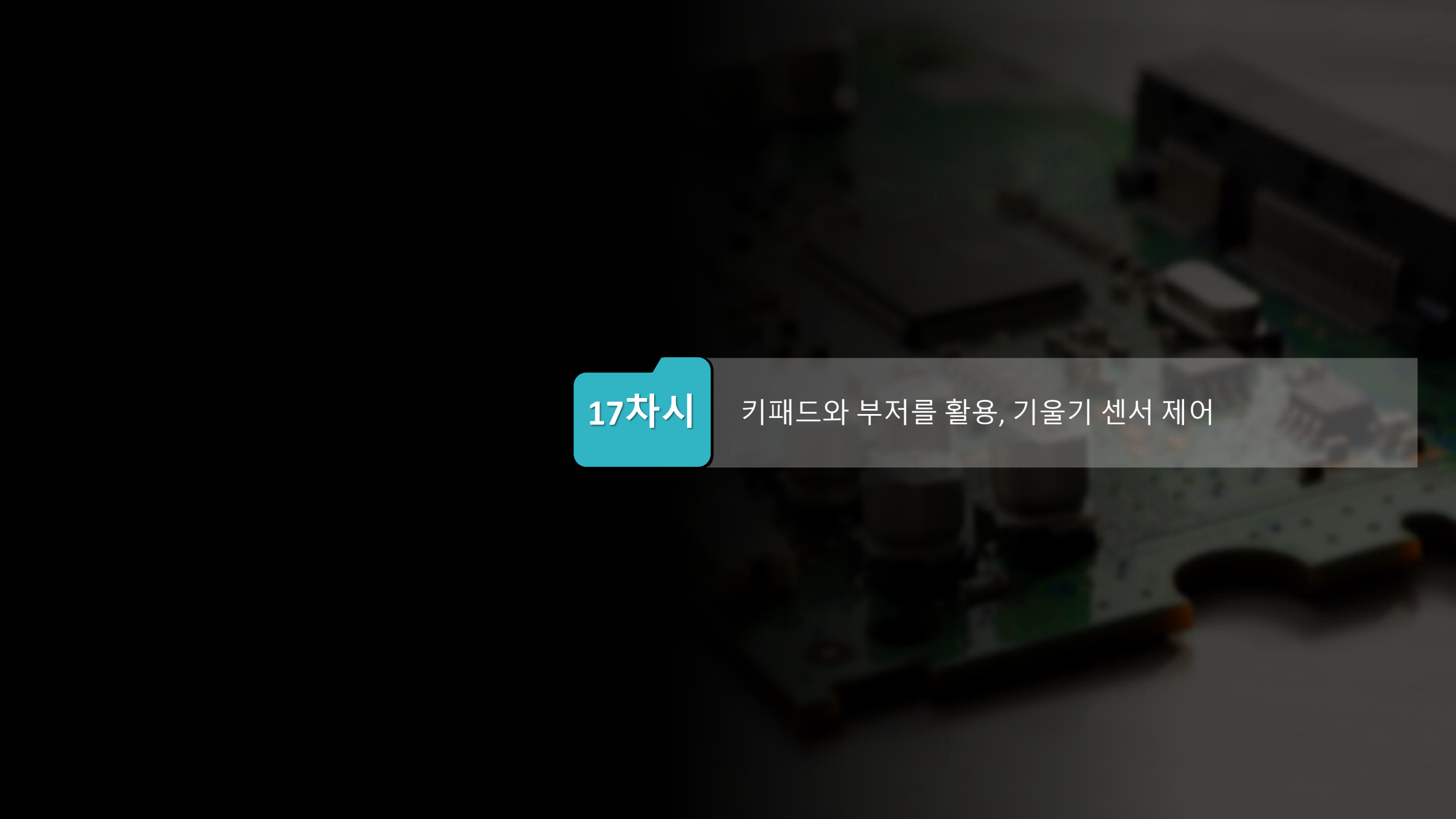
```
#include <Keypad.h>
const byte ROWS = 4;    // 행(rows) 개수
const byte COLS = 4;    // 열(columns) 개수
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}  };
byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS,
COLS);

String Password = "1234AD"; // 6자리 패스워드로 설정
String userPassword = "";
```

```
void loop() {
  char key = keypad.getKey();
  if (key) {
    if (key == '#') {
      // 입력이 완료되면 패스워드 확인
      if (userPassword == Password) {
        Serial.println("Welcome! HOME");
        // 여기에 패스워드가 올바른 경우
      } else {
        Serial.println("Incorrect Password!");
        // 여기에 패스워드가 잘못된 경우
      }
      // 패스워드 확인 후 변수 초기화
      userPassword = "";
    } else {
      // 패스워드 입력 중이면 입력값을 추가
      userPassword += key;
    }
  }
}
```

16 학습정리

학습 정리

- 키패드의 행과 열 구조, 입력 동작 원리 학습
- 키패드 입력 값을 인식하여 텍스트 출력 방법 학습
- 키패드로 패스워드 입력, 생성, 확인하는 프로그래밍 학습



17차시

키패드와 부저를 활용, 기울기 센서 제어

학습목표, 학습내용

학습 목표

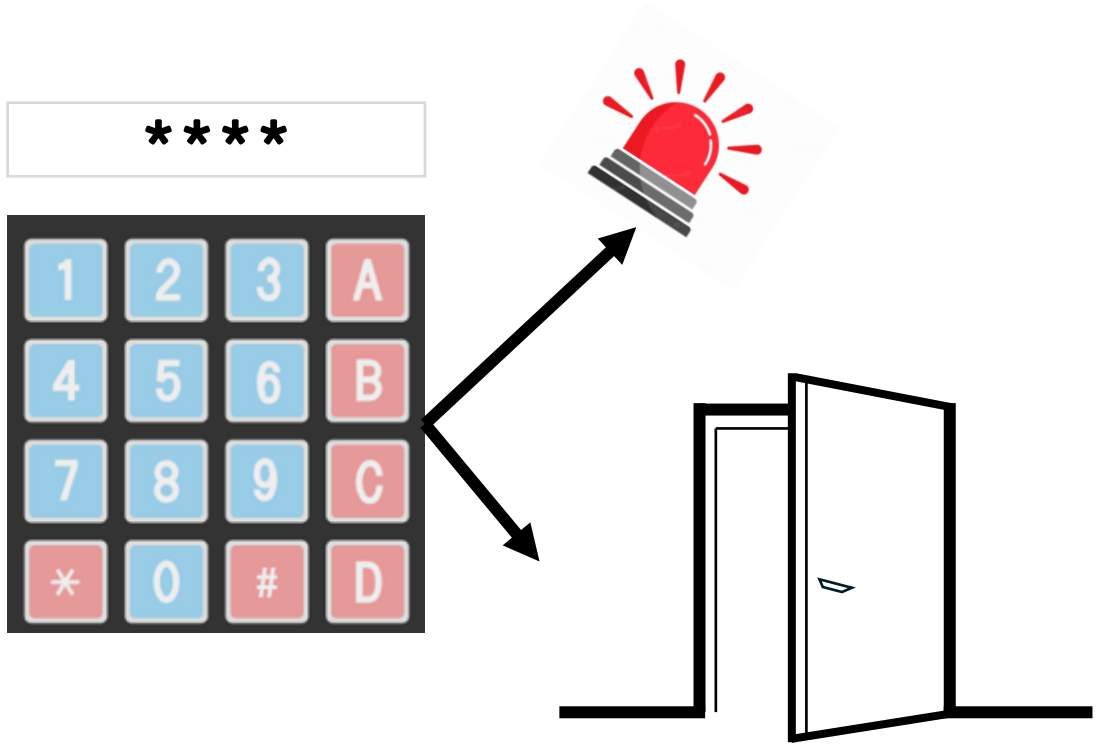
- 패스워드와 소리알람을 연동하여 디지털 도어락 시스템을 이해한다.
- 패스워드 오류입력시 부저를 통하여 다시 입력할 수 있도록 알려준다.
- 기울기센서의 활용에 대해 알아본다.

학습 내용

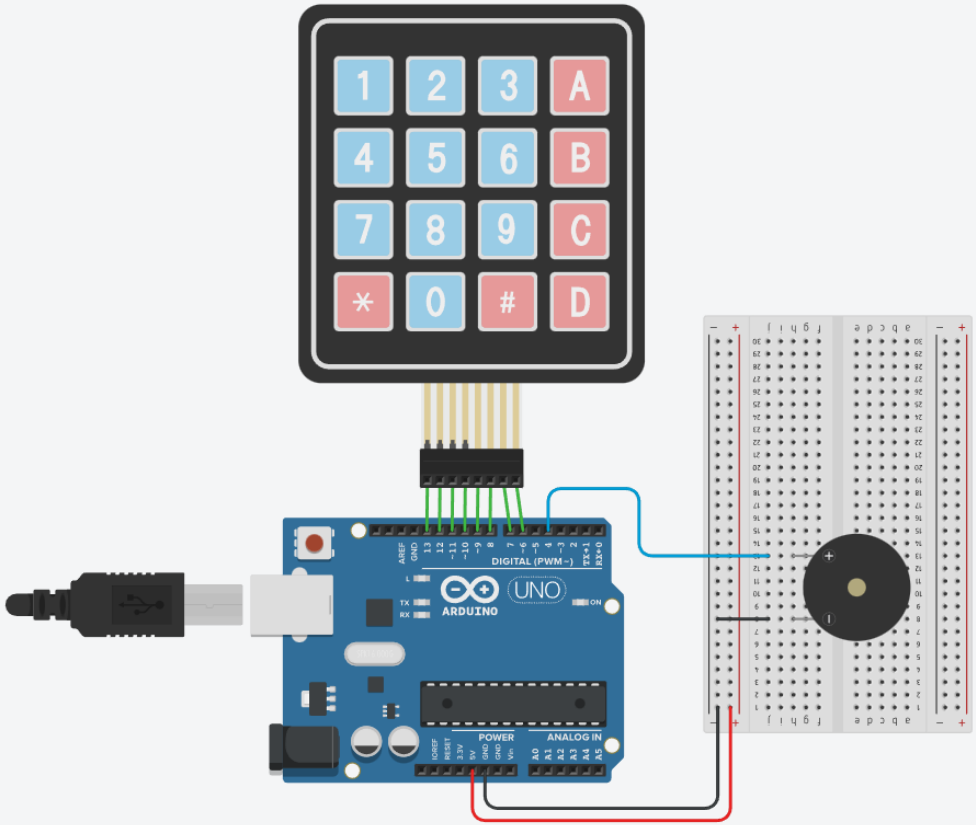
- 올바른 패스워드 입력시와 올바르지 않은 패스워드 입력시 다른 부저의 소리가 나도록 프로그래밍 작성을 실습한다.
- 기울기 센서의 값 범위를 확인하여 기울기 센서값에 따라 다른 LED가 켜지는 프로그래밍으로 제어해 본다.

키패드- 비밀번호 확인한 후 틀리면 소리나기

캐패드의 비밀번호를 확인후 틀리면 피에조 부저에서 소리가 나도록 하는 시뮬레이션을 작성해보자.



키패드에서 비밀번호 오류시 부저 울리기



키패드- 비밀번호 확인한 후 틀리면 소리나기

임의의 패스워드 “1234AD”를 누른후 #을 눌러 패스워드를 마무리하면 실행문을 실행한다.

```
#include <Keypad.h>
const byte ROWS = 4;    // 행(rows) 개수
const byte COLS = 4;    // 열(columns) 개수
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}  };
byte rowPins[ROWS] = {13, 12, 11, 10};
byte colPins[COLS] = {9, 8, 7, 6};
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

String Password = "1234AD"; // 6자리 패스워드로 설정
String userPassword = "";
int buzzerPin =4;
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(buzzerPin, OUTPUT);
}

void loop() {
  char key = keypad.getKey();
 if (key) {
    if (key == '#') {
      // 입력이 완료되면 패스워드 확인
      if (userPassword == Password) {
        Serial.println("Welcome! HOME");
        // 여기에 패스워드가 올바른 경우 수행할 동작 추가 가능
      } else {
        Serial.println("Incorrect Password!");
        // 여기에 패스워드가 잘못된 경우 수행할 동작 추가 가능
        for (int i = 0; i < 4; i++) // 5번 반복
            playErrorTone();
             }
```

키패드- 비밀번호 확인한 후 틀리면 소리나기

- 임의의 패스워드 “1234AD”를 누른후 #을 눌러 패스워드를 마무리하면 실행문을 실행한다.

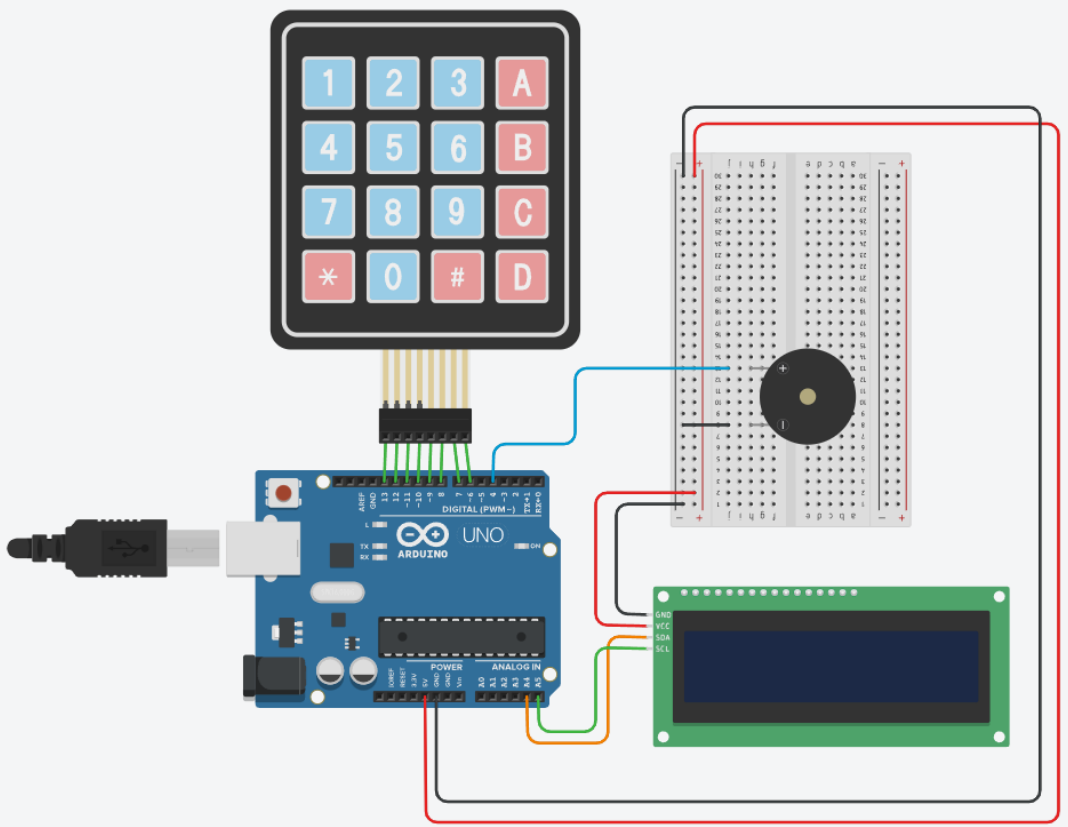
```
  userPassword = "";
    } else {
      // 패스워드 입력 중이면 입력값을 추가
      userPassword += key;
    }
  }
}

void playErrorTone() {
  tone(buzzerPin, 1000); // 1000Hz 주파수의 소리를 내기
  delay(200); // 0.2초 동안 소리 유지
  noTone(buzzerPin); // 부저 소리 중지
  delay(100); // 0.1초 동안 소리 중지 상태 유지
}
```

[프로젝트] 스마트 홈 비밀번호로 출입하기

비밀번호를 눌러 입력된 비밀번호가 맞으면 문이 열리고, 잘못된 비밀번호일 경우에는 다시 입력할 수 있도록 프로젝트를 작성해보자.

이름	수량	구성요소
U1	1	Arduino Uno R3
PIEZO1	1	피에조
U2	1	PCF8574 기반, 32 LCD 16 x 2(I2C)
KEYPAD1	1	4x4 키패드

센서		핀번호
피에조 부저		D4
LCD	GND	-
	VCC	+
	SDA	A4
	SCL	A5



[프로젝트] 스마트 홈 비밀번호로 출입하기

임의의 패스워드 “1234AD”를 누른후 #을 눌러 패스워드를 마무리하면 실행문을 실행한다.
LCD에는 비밀번호를 입력할 수 있도록 화면을 지우고, 커서 위치를 초기화 한다.

```
#include <LiquidCrystal_I2C.h>
#include <Keypad.h>
LiquidCrystal_I2C lcd(32, 16, 2);      // 0x27, 0x3F 확인
int buzzerPin = 4;
const byte ROWS = 4;    // 행(rows) 개수
const byte COLS = 4;    // 열(columns) 개수
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D’}   };
byte rowPins[ROWS] = {13, 12, 11, 10};
byte colPins[COLS] = {9, 8, 7, 6};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
```

```
String Password = "1234AD"; // 6자리 패스워드로 설정
String userPassword = "";

int melody[] = {523, 1046, 2093};

void setup() {
  Serial.begin(9600);
  pinMode(buzzerPin, OUTPUT);
  lcd.init();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0,0);

}
```

[프로젝트] 스마트 홈 비밀번호로 출입하기

임의의 패스워드 “1234AD”를 누른후 #을 눌러 패스워드를 마무리하면 실행문을 실행한다.
패스워드가 입력되면 피에조 부저에서 소리가 나고, LCD에 상황을 출력한다.

```
void loop() {
  char key = keypad.getKey();
  if (key) {
    if (key == '#') {
        // 입력이 완료되면 패스워드 확인
     if (userPassword == Password) {
        lcd.clear();
        lcd.setCursor(0,0);
        lcd.print("Welcome HOME!");
        play_melody();
        }
     else  {
        playTone();
        lcd.clear();
        lcd.setCursor(0,0);
        }
```

```
// 패스워드 확인 후 변수 초기화
      userPassword = "";
    } else {
      // 패스워드 입력 중이면 입력값을 추가
      userPassword += key;
      lcd.print(key);
    }
  }
}

void play_melody() {
  for (int i = 0; i < 3; i++) {
    tone(buzzerPin, melody[i], 200);
    delay(200);
  }
}
```

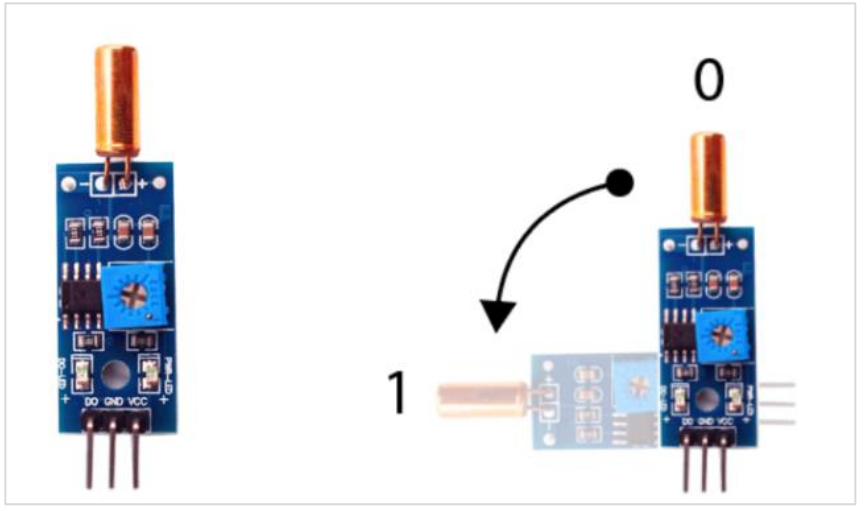
기울기(Tilt Sensor)센서

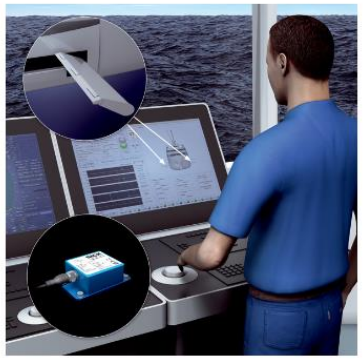
기울어진 정도에 따라서 LED를 켜는 시뮬레이션을 작성해보자.

- 기울기 센서는 중력을 사용하여 물리적으로 전기적 신호를 계산합니다.
- 센서 안에는 수은이 들어있어 수은의 위치에 따라 HIGH 또는 LOW 신호를 출력합니다.
- 센서를 기울일 경우 수은이 떨어져 스위치가 열리고 센서를 똑바로 세우면 스위치가 닫히는 구조입니다.

[왼쪽] 내부에 단자가 있어서 붙거나 떨어짐

[오른쪽] 수은이 움직이면서 단락이 연결되거나 떨어짐



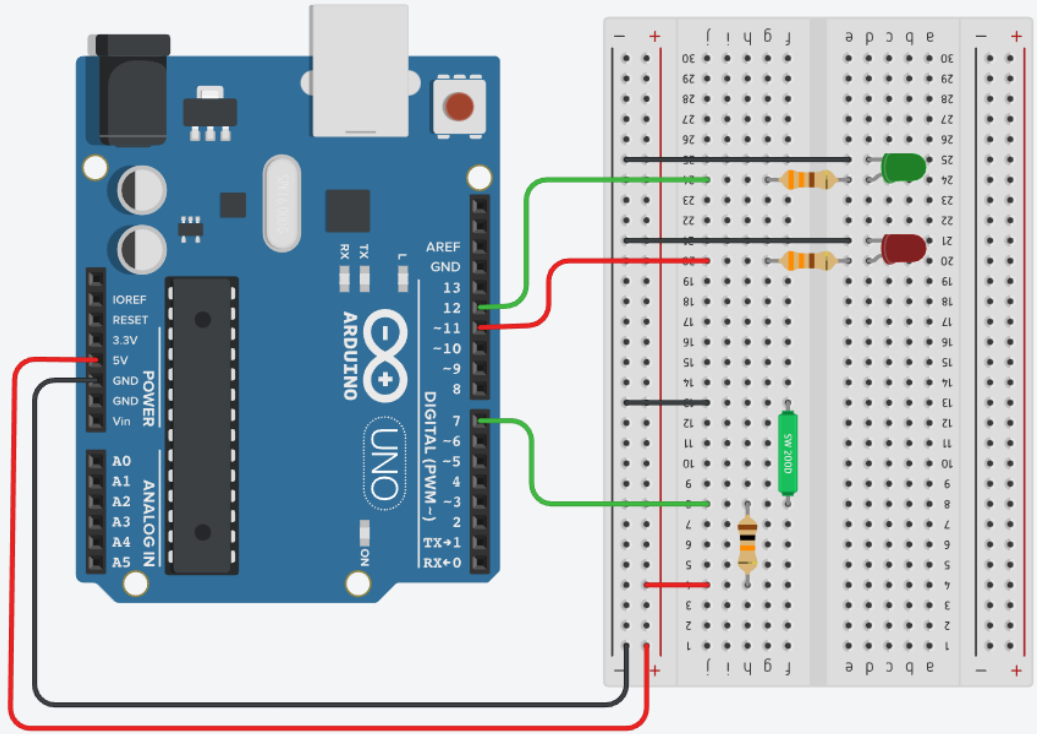


기울기(Tilt Sensor)센서

기울어진 정도에 따라서 LED를 켜는 시뮬레이션을 작성해보자.

```
int Rled = 11;
int Gled = 12;
int Tilt_pin = 7;
int val;

void setup()
{
  Serial.begin(9600);
  pinMode(Rled, OUTPUT);
  pinMode(Gled, OUTPUT);
  pinMode(Tilt_pin, INPUT);
}
```

기울기센서와 LED켜기

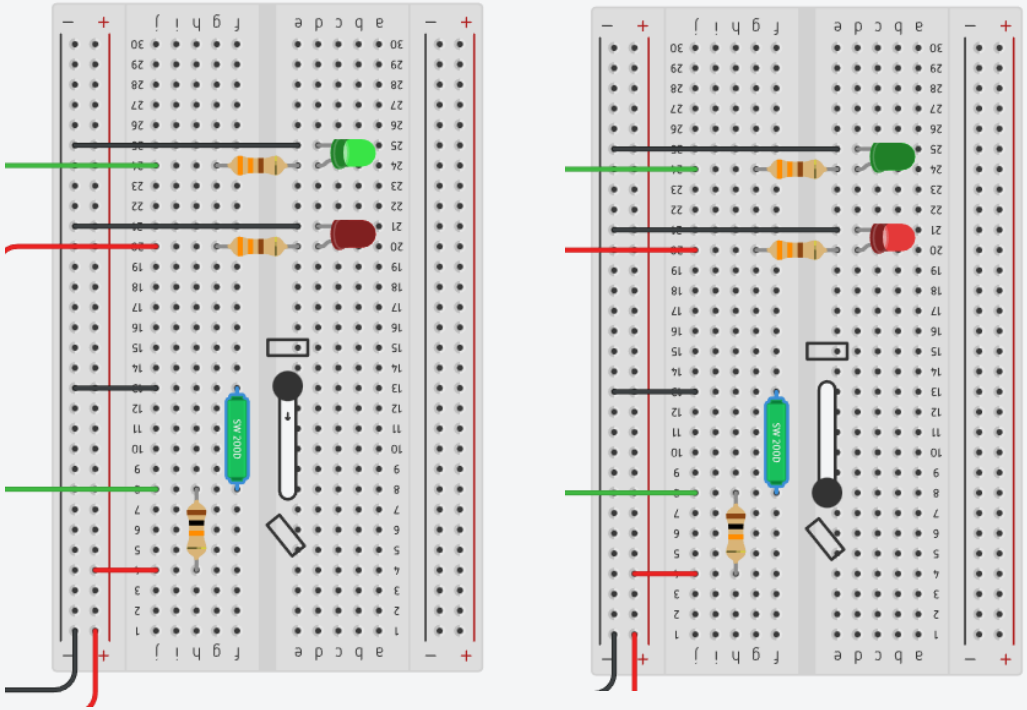


기울기(Tilt Sensor)센서

R기울기센서의 값을 시리얼 모니터에서 확인하고, LED를 켜는 프로그램을 작성해보자.

```
void loop()
{
  val = digitalRead(Tilt_pin);
  if (val == false)  {
    Serial.print("Tilt State! : " );
    Serial.println(val);
    digitalWrite(Rled, HIGH);
    digitalWrite(Gled, LOW);
  }
  else  {
     Serial.println(val);
    digitalWrite(Gled, HIGH);
    digitalWrite(Rled, LOW);
  }
}
```

균형을 잡는 기울기 센서와 LED



17 학습정리

학습 정리

- 실생활에서 키패드를 활용하여 디지털 도어락의 비밀번호 입력 방법 이해
- 기울임 센서 값 범위를 시리얼 모니터에서 확인하여 위험상황을 알수 있도록 LED를 활용하여 작동하도록 실습

18차시 스마트 홈 시스템 프로젝트 1

학습 목표

- 스마트 홈 시스템의 룸 시스템에 대해 알아보자.
  - 자동 LED 작동
  - 자동 블라인드 작동
  - 아침과 저녁에 자동으로 켜지는 LED 스탠드 작동

학습 내용

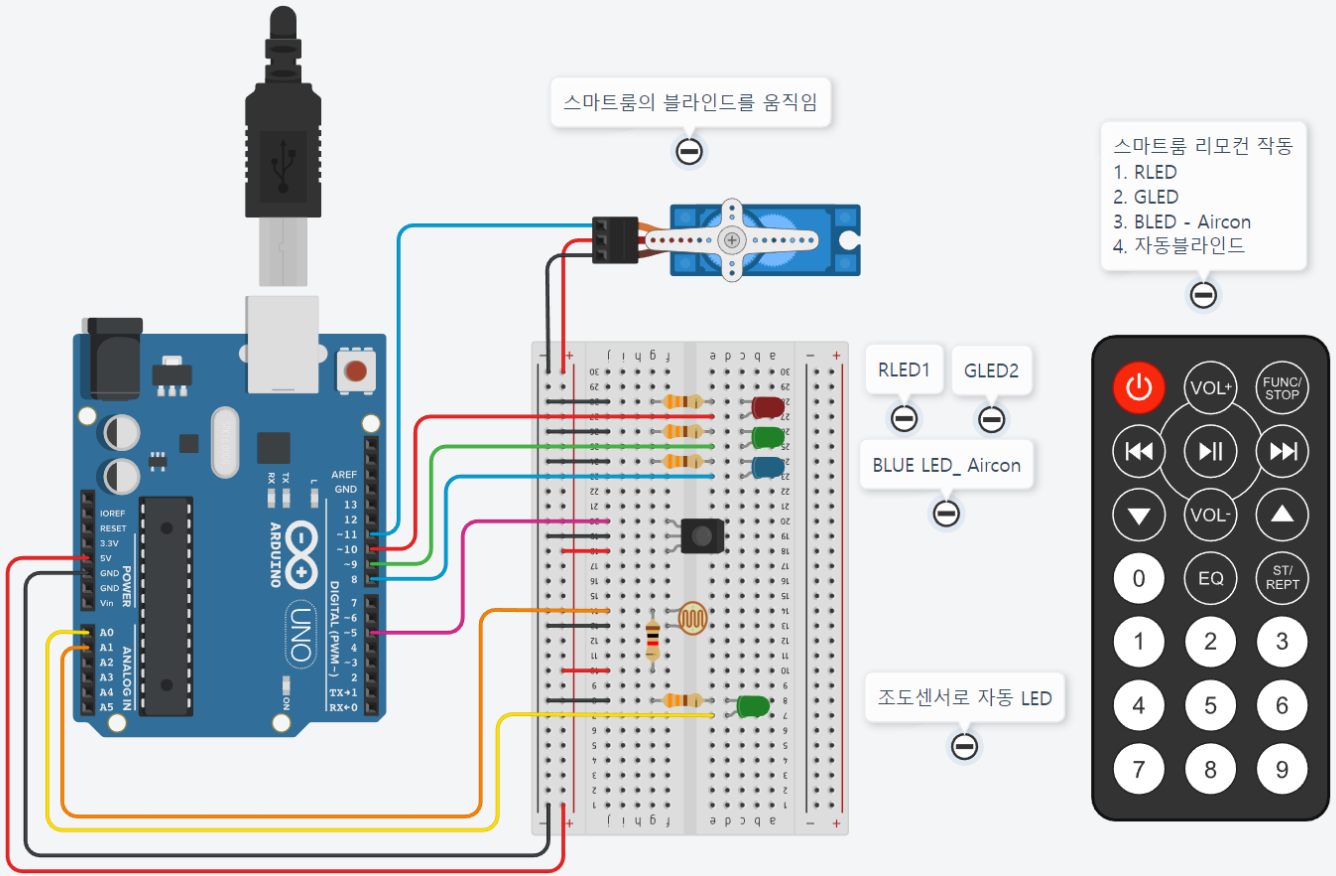
- IR센서와 리모컨을 활용하여 리모컨 조종으로 자동전등 시스템을 이해하고,
- 리모컨작동으로 방안의 자동 블라인드 시템을 알아본다.
- 아침, 저녁의 빛의 양에 따라 자동으로 켜지는 스탠드 시스템을 실습한다.

[프로젝트] 스마트 홈 제어하기

스마트홈 프로젝트 1

센서	핀번호
서보모터	11
RLED	10
GLED	9
BLED	8
IR센서	5
조도센서	A1
GLED1	A0

스마트 홈 – Room에서의 기능 회로도



스마트홈 프로젝트 1

IR센서와 리모컨으로 LED를 자동으로 켜는 시스템 설계 및 조도센서의 빛값에 의해 자동으로 켜지는 LED 시스템, 리모컨으로 블라인드를 자동으로 움직이게 하는 프로젝트 설계

```
#include <IRremote.h>
#include <Servo.h>

#define IR_RECEIVE_PIN 5
int Rled = 10;
int Gled = 9;
int Bled = 8;
int Rflag = 0;
int Gflag = 0;
int Bflag = 0;
int Sflag = 0;
int Yled = A0;
int Cds = A1;

Servo myservo;
```

```
void setup()
{
  Serial.begin(9600);
  pinMode(Rled, OUTPUT);
  pinMode(Gled, OUTPUT);
  pinMode(Bled, OUTPUT);
  pinMode(Yled, OUTPUT);
  IrReceiver.begin(IR_RECEIVE_PIN);
  myservo.attach(11);
  myservo.write(0);
}

void loop() {
   int CdsValue = analogRead(Cds);
```

```
if (CdsValue > 800)
   {
     digitalWrite(Yled, HIGH);
  }
  else
  {
    digitalWrite(Yled, LOW);
  }

if (IrReceiver.decode())
  {
    int command = IrReceiver.decodedIRData.command;
    Serial.println(command);
    delay(100);
    IrReceiver.resume();
```

스마트홈 프로젝트 1

IR센서와 리모컨으로 LED를 자동으로 켜는 시스템 설계 및 조도센서의 빛값에 의해 자동으로 켜지는 LED 시스템, 리모컨으로 블라인드를 자동으로 움직이게 하는 프로젝트 설계

```
if(command == 16 && Rflag == 0)  {
   Serial.print(command);
   Serial.print(",");
   Serial.println(Rflag);
   digitalWrite(Rled, HIGH);
   Rflag = 1;
  }
else if(command == 16 && Rflag == 1)
  {
    Serial.print(command);
    Serial.print(",");
    Serial.println(Rflag);
    digitalWrite(Rled, LOW);
    Rflag = 0;
  }
```

```
if(command == 17 && Gflag == 0)  {
      Serial.print(command);
      Serial.print(",");
      Serial.println(Gflag);
      digitalWrite(Gled, HIGH);
      Gflag = 1;
    }
else if(command == 17 && Gflag == 1)
    {
      Serial.print(command);
      Serial.print(",");
      Serial.println(Gflag);
      digitalWrite(Gled, LOW);
      Gflag = 0;
    }
```

```
if(command == 18 && Bflag == 0)   {
      Serial.print(command);
      Serial.print(",");
      Serial.println(Bflag);
      digitalWrite(Bled, HIGH);
      Bflag = 1;
    }
 else if(command == 18 && Bflag == 1)
    {
      Serial.print(command);
      Serial.print(",");
      Serial.println(Bflag);
      digitalWrite(Bled, LOW);
      Bflag = 0;
    }
```

스마트홈 프로젝트 1

IR센서와 리모컨으로 LED를 자동으로 켜는 시스템 설계 및 조도센서의 빛값에 의해 자동으로 켜지는 LED 시스템, 리모컨으로 블라인드를 자동으로 움직이게 하는 프로젝트 설계

```
if(command == 20 && Sflag == 0)   {
  myservo.write(90);
  delay(1000);
  Sflag = 1;
    }
else if(command == 20 && Sflag == 1)
    {
      myservo.write(0);
      delay(1000);
      Sflag = 0;
    }
  }// if end
}//loop end
```

18 학습정리

학습 정리

- IR센서와 리모컨의 1번, 2번, 3번, 4번의 버튼작동으로 LED와 서보모터를 움직여 스마트홈시스템 중 방안에서의 상황 프로젝트를 실습
- 조도센서로 자동 스탠드 시스템을 실습

19차시 스마트 홈 시스템 프로젝트 2

학습 목표

- 스마트 홈 시스템의 거실 시스템에 대해 알아보자.
  - 주방의 가스점검 시스템
  - 거실의 TV화면 제어 시스템
  - 방문자의 침입상황 제어 시스템

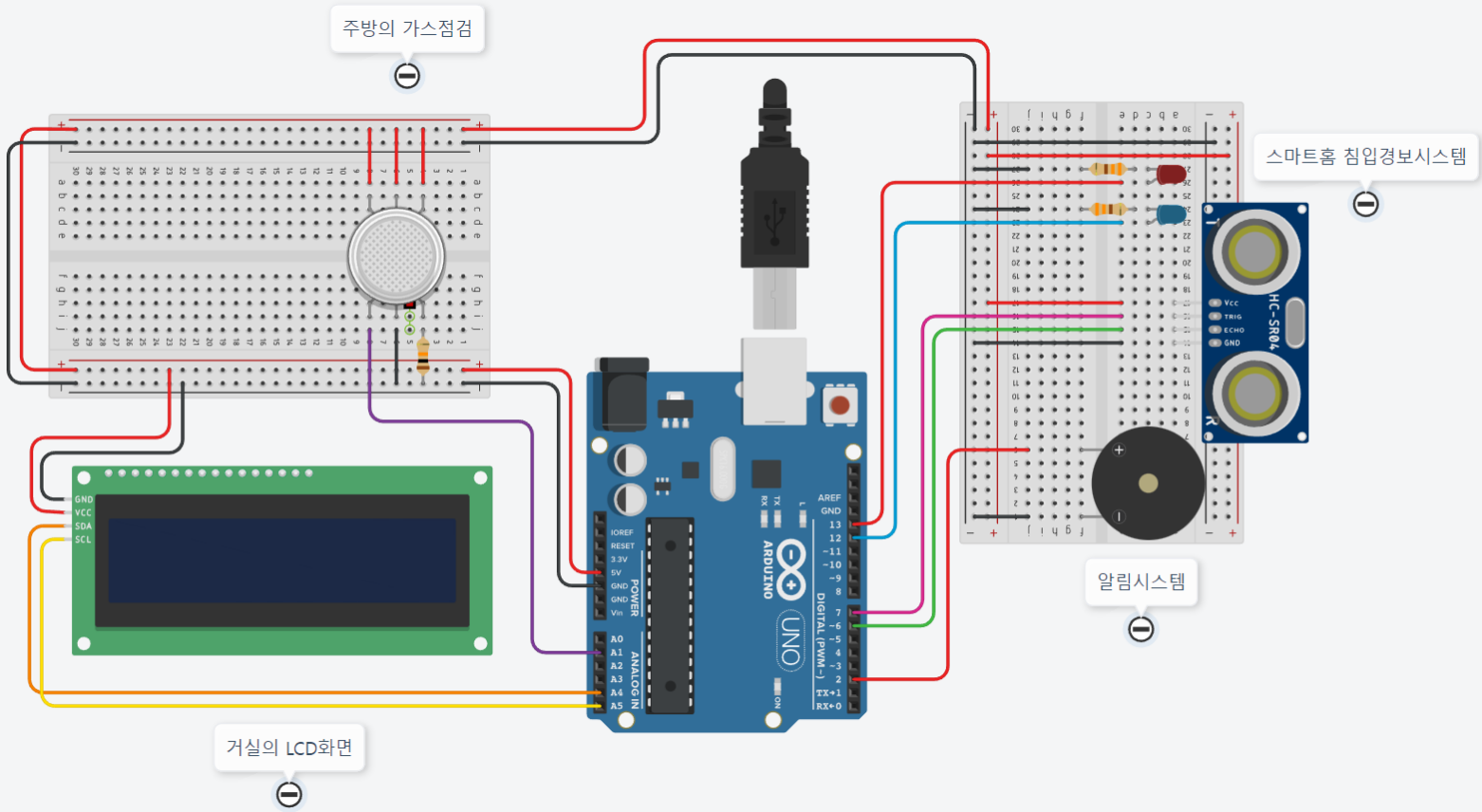
학습 내용

- 스마트홈의 거실에서 상황을 이해한다.
- 주방의 가스누출 감지 시스템 회로도를 작성하고 알림기능을 프로그래밍한다.
- 스마트 TV 출력을 제어하고, 원하지않는 방문자 침입 상태 시스템도 제어 프로그래밍한다.

스마트홈 프로젝트 2

센서		핀번호
가스센서		A1
RLED		13
BLED		12
I2C LCD	SDA	A4
	SCL	A5
초음파센서	Trig	7
	Echo	6
Piezo Buzzer		2

스마트 홈 – 거실과 주방 시스템 회로도



스마트홈 프로젝트 2

스마트홈의 거실 프로젝트로, 주방의 가스누출시 부저소리와 LCD출력시스템 설계, 초음파센서로 장애물인식 또는 침입상태가 확인되면 LED와 부저를 통해 알림을 해준다.

```
#include <LiquidCrystal_I2C.h>

int gas_sensor = A1;
int buzzerPin = 2;
int Rled = 13;
int Bled = 12;
int trig = 7;
int echo = 6;

int val = 0;
int gas_flag =0;
int lcd_flag =0;
LiquidCrystal_I2C lcd(32,16,2);

int alarm[] = {523, 1046, 2093};
```

```
void setup()  {
    Serial.begin(9600);
    pinMode(Rled,OUTPUT);
    pinMode(Bled,OUTPUT);
    pinMode(trig, OUTPUT);
    pinMode(echo, INPUT);

    lcd.init();
    lcd.backlight();
  }
```

```
void loop()  {

  digitalWrite(trig, LOW);
  digitalWrite(echo, LOW);
  delayMicroseconds(2);
  digitalWrite(trig, HIGH);
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

  unsigned long duration = pulseIn(echo, HIGH);
  float distance = duration * 0.0343 /2.0;
```

스마트홈 프로젝트 2

스마트홈의 거실 프로젝트로, 주방의 가스누출시 부저소리와 LCD출력시스템 설계, 초음파센서로 장애물인식 또는 침입상태가 확인되면 LED와 부저를 통해 알림을 해준다.

```
if(distance < 150)  {
    digitalWrite(Rled, HIGH);
    digitalWrite(Bled, LOW);
    delay(50);
    digitalWrite(Rled, LOW);
    digitalWrite(Bled, HIGH);
    delay(50);
    play_alarm();
  }
  else
  {
    digitalWrite(Rled, LOW);
    digitalWrite(Bled, LOW);
  }
  delay(200);
```

```
  val = analogRead(gas_sensor);

   if(val > 500 )  {
     Serial.println(val);
     digitalWrite(Bled, LOW);
     digitalWrite(Rled, HIGH);
     play_alarm();
     delay(300);
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Danger! ");
     lcd.setCursor(0,1);
     lcd.print("gas leakage");
     gas_flag=0;
     }
```

```
  else if(val < 500 && gas_flag == 0) {
     Serial.println(val);
     digitalWrite(Bled, HIGH);
     digitalWrite(Rled, LOW);
     lcd.clear();
     lcd.setCursor(0,0);
     lcd.print("Welcome! ");
     lcd.setCursor(0,1);
     lcd.print("SmartHome System!");
     gas_flag = 1;
   }

   delay(200);
}
```

스마트홈 프로젝트 2

스마트홈의 거실 프로젝트로, 주방의 가스누출시 부저소리와 LCD출력시스템 설계, 초음파센서로 장애물인식 또는 침입상태가 확인되면 LED와 부저를 통해 알림을 해준다.

```
void play_alarm()
{
    for (int i = 0; i < 3; i++)
   {
      tone(buzzerPin, alarm[i], 200);
      delay(200);
   }
}
```

19 학습정리

학습 정리

- LCD에서 스마트홈 월패드 제어 시스템을 이해하고, 출력하는 프로그래밍 실습
- 가스누출시 LED와 부저로 알림서비스를 제어하는 프로그램을 작성하고, 초음파센서의 감지시스템으로 경보알림을 제어 실습

20차시 스마트 홈 시스템 프로젝트 3

학습 목표

- 스마트 홈 시스템의 디지털 도어락 출입시스템
  - 디지털 도어락의 패스워드 감지 시스템
  - 자동 문열림 시스템
  - 현관 자동 센서등 제어 시스템

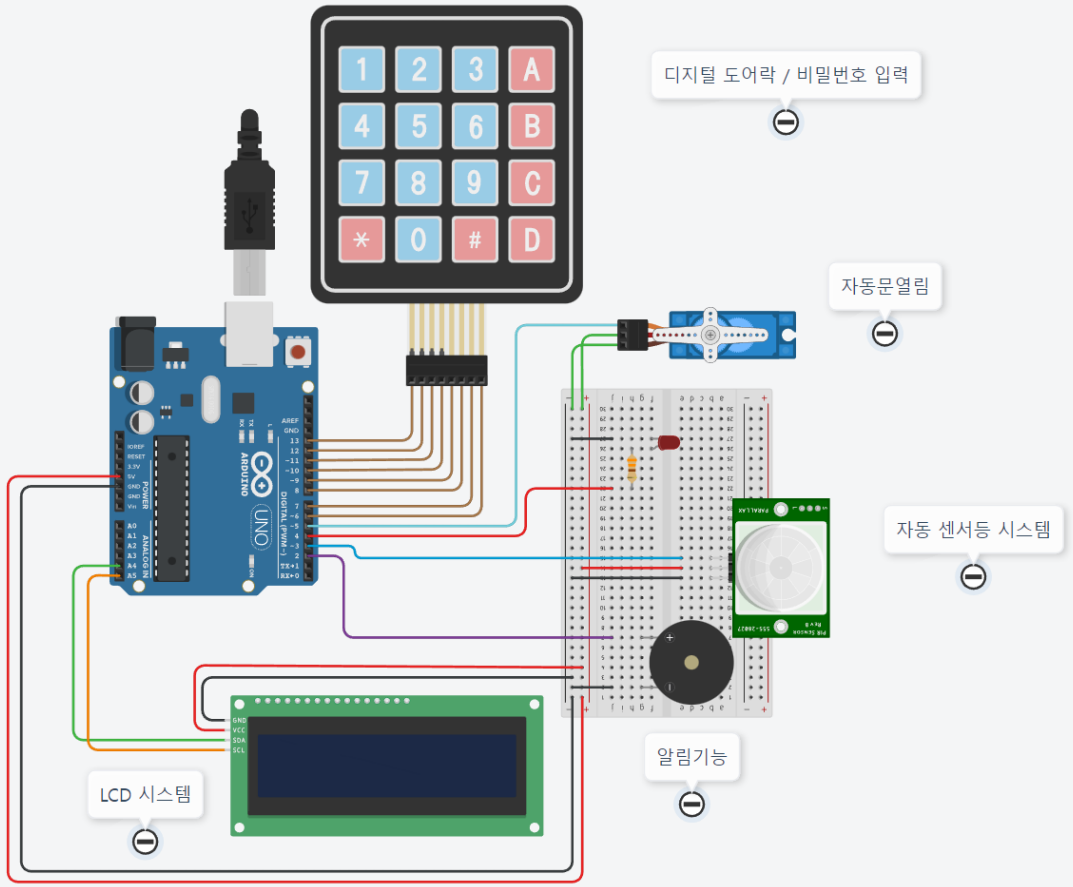
학습 내용

- 키패드를 활용하여 패스워드를 입력하고, LCD에 상황을 출력하는 프로그래밍 제어 회로도를 작성한다.
- 올바른 패스워드로 서보모터를 움직인다.
- PIR센서로 자동센서등 시스템에 대해 이해한다.

스마트홈 프로젝트 3

센서		핀번호
4x4 키패드	Row 1-4	13-10
	Col 1-4	9-6
I2C LCD	SDA	A4
	SCL	A5
Piezo Buzzer		2
PIR센서		3
RLED		4
서보모터		5

스마트 홈 – 디지털 도어락과 인체감지센서 회로도



스마트홈 프로젝트 3

스마트홈의 출입문 시스템으로 패스워드가 맞으면 서보모터로 자동문열림 시스템과 잘못된 패스워드시 부저를 울려준다. 출입문 통과시 인체감지센서에 의해 자동전등 시스템이 작동된다.

```
#include <Keypad.h>
#include <LiquidCrystal_I2C.h>
#include <Servo.h>
LiquidCrystal_I2C lcd(32, 16, 2);
Servo servo;
int buzzerPin =2;
int alarm[] = {523, 1046, 2093};
int ledPin = 4;
int inputPin = 3;
// PIR 센서 신호핀
int pirState = LOW;
// PIR 센서 초기 움직임이 없음을 가정
int val = 0;
// 센서 신호의 판별을 위한 변수
```

```
const byte ROWS = 4;    // 행(rows) 개수
const byte COLS = 4;    // 열(columns) 개수
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}  };
byte rowPins[ROWS] = {13, 12, 11, 10};
byte colPins[COLS] = {9, 8, 7, 6};
Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

String Password = "1234AD"; // 6자리 패스워드로 설정
String userPassword = "";
```

스마트홈 프로젝트 3

스마트홈의 출입문 시스템으로 패스워드가 맞으면 서보모터로 자동문열림 시스템과 잘못된 패스워드시 부저를 울려준다. 출입문 통과시 인체감지센서에 의해 자동전등 시스템이 작동된다.

```
void setup()
  {
    Serial.begin(9600);
    servo.attach(5);
    // 서보모터를 5번핀에 연결
    servo.write(0);
   // 서보모터를 0도 움직임

    lcd.init();
    lcd.backlight();
    lcd.setCursor(0,0);
    lcd.print("Welcome! ");
    lcd.setCursor(0,1);
    lcd.print("SmartHome System!");
  }
```

```
void loop() {

val = digitalRead(inputPin);
// 센서 신호값을 읽어와서 val에 저장
if (val == HIGH)
  {
     Serial.println(val);
     digitalWrite(ledPin, HIGH);
   }
 else
  {
   Serial.println(val);
   digitalWrite(ledPin, LOW);
  }
```

```
  char key = keypad.getKey();
   if (key) {
     lcd.clear();
     if (key == '#') {
       // 입력이 완료되면 패스워드 확인
       if (userPassword == Password) {
         lcd.setCursor(0,0);
         lcd.print("Welcome! Home!");
         lcd.setCursor(0,1);
         lcd.print("Open the Door!");
         play_alarm();
         servo.write(120);
         delay(2000);
         }
```

스마트홈 프로젝트 3

스마트홈의 출입문 시스템으로 패스워드가 맞으면 서보모터로 자동문열림 시스템과 잘못된 패스워드시 부저를 울려준다. 출입문 통과시 인체감지센서에 의해 자동전등 시스템이 작동된다.

```
else {
        Serial.println("Incorrect Password!");
        lcd.setCursor(0,0);
        lcd.print("Incorrect Password!! ");
        lcd.setCursor(0,1);
        lcd.print("Try Again!");
        // 여기에 패스워드가 잘못된 경우
        for (int i = 0; i < 4; i++) // 5번 반복
            playErrorTone();
      }
      // 패스워드 확인 후 변수 초기화
      userPassword = "";
    }
```

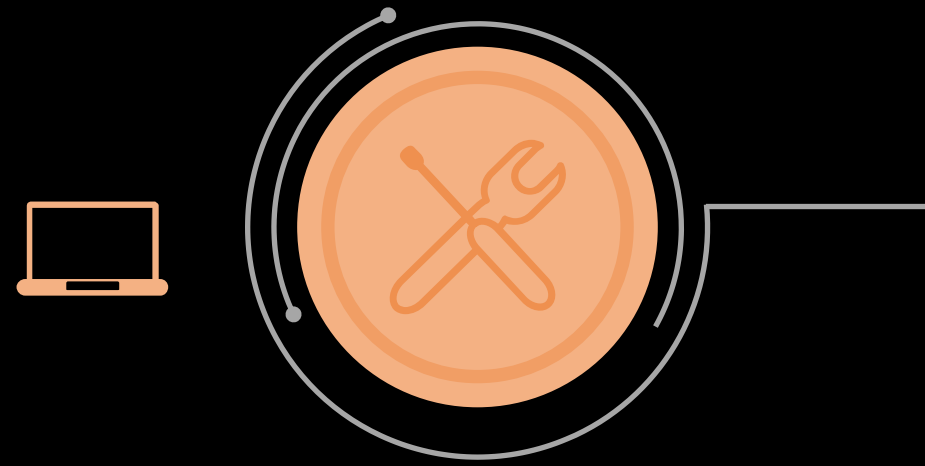
```
else {
// 패스워드 입력중이면 입력값을 추가
      userPassword += key;
    }
  }
}
```

```
void playErrorTone() {
  tone(buzzerPin, 1000);
// 1000Hz 주파수의 소리를 내기
  delay(200); // 0.2초 동안 소리 유지
  noTone(buzzerPin); // 부저 소리 중지
  delay(100);
}

void play_alarm()
{
   for (int i = 0; i < 3; i++) {
      tone(buzzerPin, alarm[i], 200);
      delay(200);
  }
}
```

20 학습정리

학습 정리

- 스마트 홈 실생활의 디지털 도어락제어 시스템에 대해 실습
- 올바른 패스워드입력시 월패드 시스템에 출력하는 상황연출 및 제어 실습
- 현관의 움직임 감지에 따른 자동 센서등 시스템의 제어 실습



본 콘텐츠는 한림대학교 sw중심대학사업의 지원에 의한 콘텐츠로

가천대학교, 호서대학교 sw중심대학사업단 공동 참여로 개발 되었음.