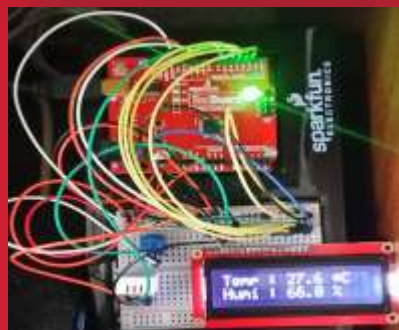


# Arduino-basic

[wk11]

## Analog Input I.



Learn how to code Arduino from scratch

Comsi, INJE University

2<sup>nd</sup> semester, 2019

Email : [chaos21c@gmail.com](mailto:chaos21c@gmail.com)



# My ID (ARnn)

**AR01**    엄현제

**AR02**    강민수

**AR03**    구병준

**AR04**    김종민

**AR05**    박성철

**AR06**    이승현

**AR07**    이창호

**AR08**    변성현

**AR09**    손성빈

**AR10**    안예찬

**AR11**    유종인

**AR12**    이석민

**AR13**    이주원

**AR14**    정재영

**AR15**    차요신

**AR16**    하태성

**AR17**    강현이

**AR18**    신종원

**AR19**    최진솔

**AR20**    김경미

**AR21**    김경영

**AR22**    김규년

**AR23**    김민재

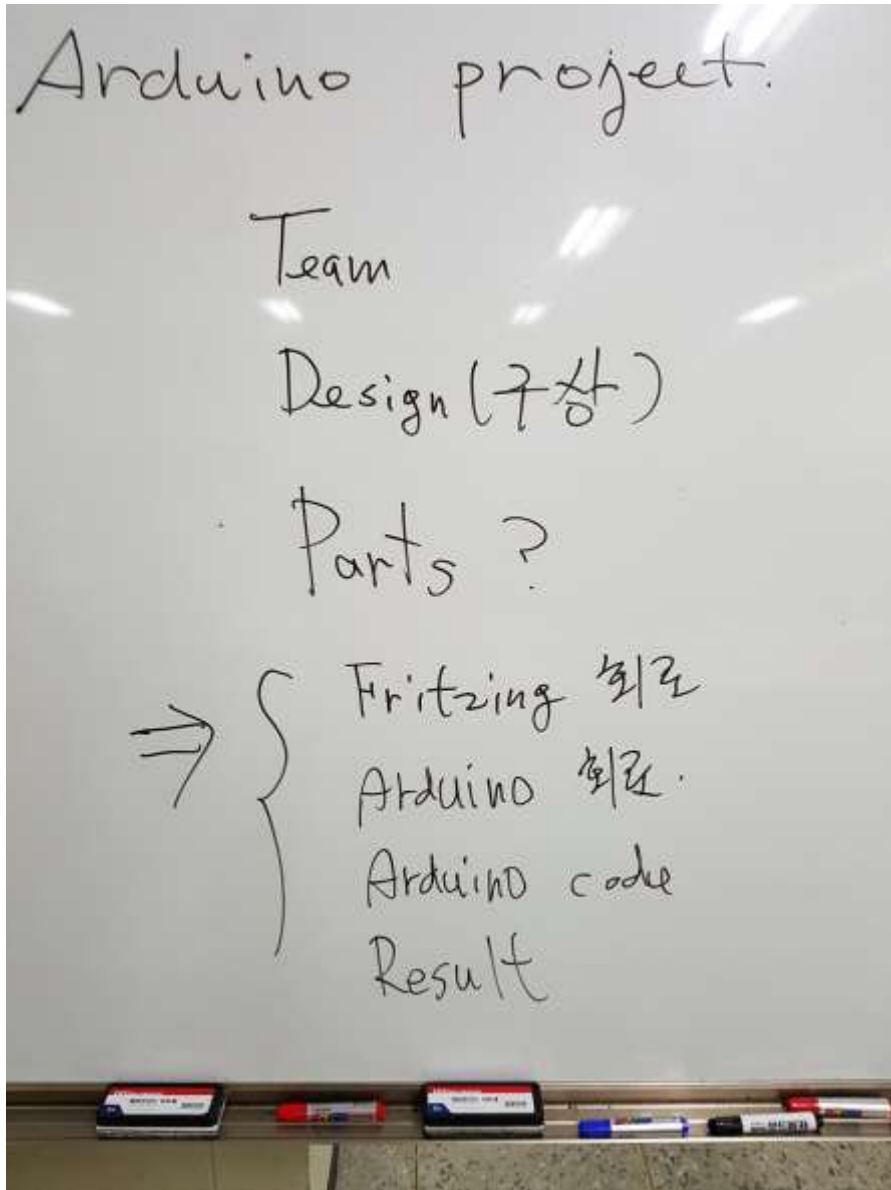
**AR24**    김영록

**AR25**    송다은

**AR26**    정지환

**AR27**    김종건

# Arduino team project



- 2명/팀
- 구상 소개 (12.05, 12.10), ppt 준비
- 부품은 수업 세트 기준
- 팀당 발표 자료 준비
- 발표 : 12월12일 (목)
  - ✓ PPT 발표 및 시연 (동영상도 가능)
- 참고
  - 추가 부품은 조별로 개별 조달.



# [Review]

## ◆ [wk10]

- **Arduino : Digital input**
- **Complete your project**
- **Submit file : ARnn\_Rpt06**



# 5. Digital input

14

6EA



택트스위치 (12x12x7)

스위치를 누르고 있을 경우만  
ON됩니다.

15

각3EA



택트스위치 캡  
(파랑,노랑,초록,빨강,하양)

택트스위치를 사용할 때  
스위치간의 구분을 할 수 있습니다.

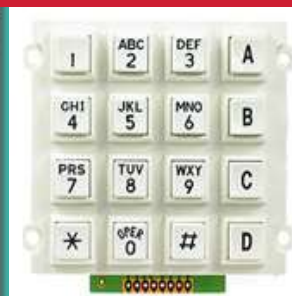
24

1EA



4x4 16 키패드 모듈

16개의 버튼을  
사용할 수 있습니다.



# wk10 : Practice-06 : ARnn\_Rpt06

## ◆ [Target of this week]

- Complete your works
- Save your outcomes
- Upload all in github.

제출폴더명 : **ARnn\_Rpt06**

### 제출할 파일들

- ① **ARnn\_Switch.ino**
- ② **ARnn\_Switch\_good.ino**
- ③ **ARnn\_Switch\_time.png**
- ④ **ARnn\_Switch\_time.ino**
- ⑤ **ARnn\_all\_keys.png**
- ⑥ **\*.ino**

# wk10 : Practice-06 : ARnn\_Switch\_time

ARnn\_Switch\_time

```

4  */
5  #include <Wire.h>
6  // I2C LCD 라리브러리 설정
7  #include <LiquidCrystal_I2C.h>
8  // LCD I2C address 설정
9  // PCF8574:0x27, PCF8574A:0x3F
10 LiquidCrystal_I2C lcd(0x27,16,2);
11 // LCD address:0x27, 16X2 LCD, 0x3F
12
13 // 2번핀을 스위치 입력으로 설정
14 const int inputPin = 2;
15
16 // 현재의 시간을 저장하기 위한 변수
17 long startTime = 0;
18 // 실제 스위치가 눌린 후 지연되는 시간
19 long swCountTimer = 0;
20
21 void setup() {
22
23   // 스위치 입력을 위하여 2번핀을 입력으로 설정하고 풀업시킨다
24   pinMode(inputPin, INPUT_PULLUP);
25   // 시리얼 통신을 설정한다
26   Serial.begin(9600);
27   lcd.init();
28   lcd.clear();
29   lcd.backlight();
30   lcd.setCursor(0,0);
31   lcd.print("Press button");
32 }

```

```

34 void loop(){
35   // 스위치 입력이 발생하였을 경우 실행
36   if(digitalRead(inputPin) == LOW){
37     // 현재의 시간을 startTime 변수에 넣는다.
38     startTime=millis();
39     // 스위치가 입력되는 동안 지연시킨다.
40     while(digitalRead(inputPin)==LOW);
41     // swCountTimer 변수에 스위치가 눌려진 시간을 넣는다.
42     // 여기까지 측정된 시간에서 앞서 저장한 시간이
43     // 스위치가 눌려진 시간이 된다
44     swCountTimer = millis() - startTime;
45     // LCD로 경과 시간을 출력한다..
46     delay(100);
47     // 모두 삭제
48     lcd.clear();
49     // 커서를 좌측 상단으로
50     lcd.setCursor(0,0);
51     lcd.print("ARnn time span");
52     // 커서를 두 번째 줄로
53     lcd.setCursor(0,1);
54     lcd.print(swCountTimer);
55     lcd.print(" ms");
56   }
57 }

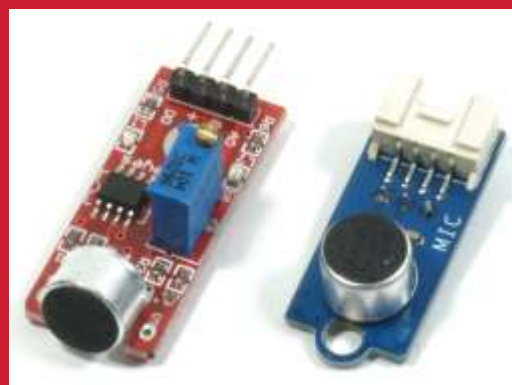
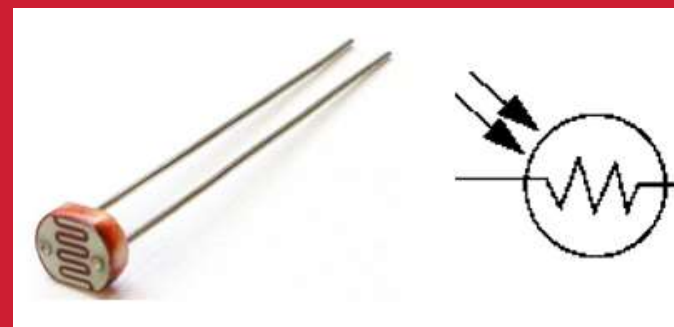
```



161mss ← Bug!



# 6. Analog input





## 6. 아날로그 신호 입력

- 6.1    포텐쇼미터 입력 (가변저항기)
- 6.2    빛 입력 (CdS, LDR)
- 6.3    온도 측정 (LM35, TMP36)
- 6.4    수위 측정
- 6.5    아날로그 조이스틱
- 6.6    소리 입력



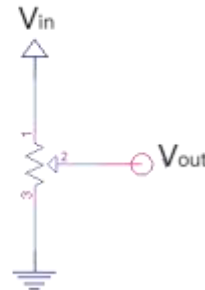
# 6.1 potentiometer

## 가변저항기



# 6.1 포텐쇼미터 (가변저항 조절)

## 포텐쇼미터 (Potentiometer)



- ✓ 회전, 직선 변위를 감지하는 센서
- ✓ 위치에 따라 저항 값이 변화함. (가변저항기)
- ✓ ADC를 이용하여 변화된 저항에 전압을 인가하여 전압의 변화를 감지

# 6.1.1 포텐쇼미터 (가변저항 조절)

## EX 6.1

## 포텐쇼미터 입력 (1/3)

### 실습목표

1. 포텐쇼미터를 회전에 따라 LED의 점멸 주기를 조절해 보자.
2. 포텐쇼미터의 값을 아날로그 핀을 통하여 0~1023 범위로 읽는다.
3. 이를 0~100의 범위의 숫자로 변경한다. (또는 0~10 kΩ, 0~5V)
4. 변경된 숫자를 참고하여 LED의 듀티비를 조절한다.
5. 현재 포텐쇼미터의 값을 시리얼 통신으로 출력한다.

### Hardware

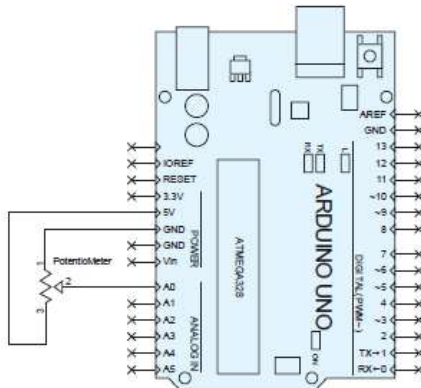
1. 실험에 사용할 포텐쇼미터는 10kΩ 사양의 3개의 핀이 있다. 1, 3번핀에서 측정되는 저항값을 포텐쇼미터의 회전에 따라 2번핀으로 나누게 된다. 즉 1, 3번핀 사이의 저항값을 포텐쇼미터가 회전을 하면 2번핀이 1, 2번핀 사이의 저항과 2, 3번핀 사이의 저항으로 나누게 된다.
2. 이 때 1, 3번핀 양단에 전압차가 발생하면 옴의법칙에 의해서 전압은 저항값의 비율로 나뉘지게 된다. 즉 1, 2번핀의 저항값과 2, 3번핀의 저항값의 비가 5:5라면 전압도 5:5로 나뉜다.
3. 포텐쇼미터 1번핀에 GND, 3번핀에 5V를 연결한다.
4. 포텐쇼미터의 2번핀을 아날로그입력 0번핀(A0)에 연결한다.
5. 포텐쇼미터를 회전시키면 1, 2번핀 사이의 저항이 변화한다. 저항의 변화에 따라 전압도 변화한다. 이 때 전압의 범위는 0~5V이다. 이를 ADC로 읽어 포텐쇼미터의 회전 각도를 알아낼 수 있다.

# 6.1.2 포텐쇼미터 (가변저항 조절)

EX 6.1

## 포텐쇼미터 입력 (2/3)

Hardware



Commands

- `analogRead`(아날로그 핀번호)

아날로그핀에서 아날로그 값을 읽는다. 0~5V사이의 전압을 0~1023사이의 값으로 표현한다.

- `map` (변수명, 범위1 최소값, 범위1 최대값, 범위2 최소값, 범위2 최대값)

변수명의 변수의 범위1의 범위와 범위2의 범위에 매칭시킨다. 즉 변수가 0~1023의 범위를 갖고 이를 0~100의 범위로 매칭하려면 '`map(변수명, 0, 1023, 0, 100)`'의 명령어로 매칭시킬 수 있다.

## 6.1.3 포텐쇼미터 (가변저항 조절)

### EX 6.1

### 포텐쇼미터 입력 (3/3)

- Sketch 구성**
1. A0핀에 아날로그 입력을 받고 내장 LED인 13번 핀을 출력으로 사용한다.
  2. 'analogRead()' 명령어로 포텐쇼미터 값을 읽는다.
  3. 'map()' 명령어로 **포텐쇼미터값(0~1023)**과 LED의 **듀티비(0~100)**를 매칭시킨다.
  4. ADC 값과 듀티비를 시리얼통신으로 PC에 전송한다.
- 실행 결과**
1. 포텐쇼미터를 회전시킬 때 마다 LED의 점등 주기가 변경된다.
  2. 시리얼 모니터에 'ADC Value: XXX, Duty cycle: XXX%'이 표시된다.

## 6.1.4 포텐쇼미터 : code-1

```

6 // 0번 아날로그핀을 포텐쇼미터 입력으로 설정한다.
7 const int potentiometerPin = 0;
8
9 //13번 핀에 연결되어 있는 내장 LED를 출력으로 사용한다.
10 const int ledPin = 13;
11
12 void setup() {
13 // 13번 핀을 출력으로 설정한다.
14 pinMode(ledPin, OUTPUT);
15 // 시리얼 통신을 설정한다.
16 Serial.begin(9600);
17 }

```

COM6

```

ADC Value: 157. Duty cycle: 15%
ADC Value: 158. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%
ADC Value: 155. Duty cycle: 15%

```

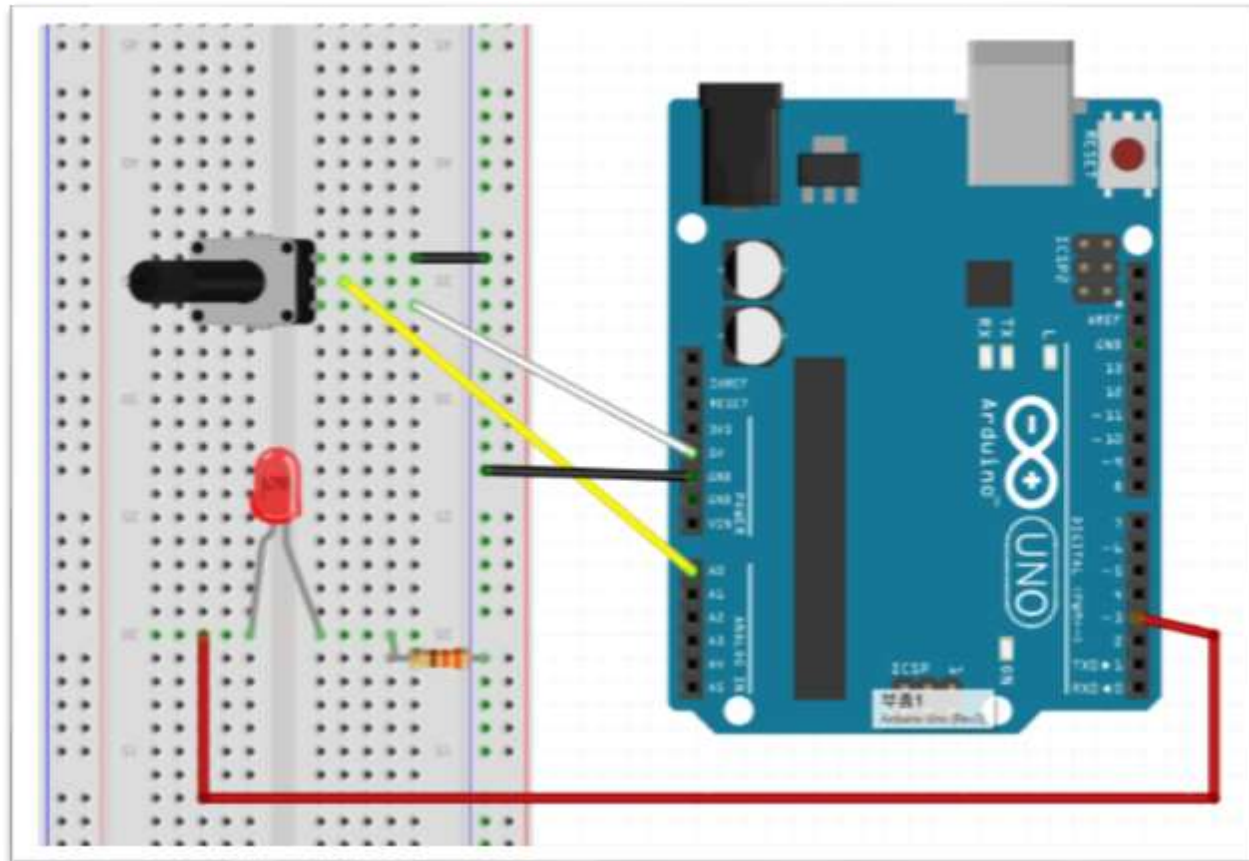
```

19 void loop(){
20
21     int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
22     int duty;      // LED 점멸 주기 (0~100%)
23
24     // 포텐쇼미터 값을 읽는다.
25     adcValue = analogRead(potentiometerPin);
26     // 포텐쇼미터 값을 0~100의 범위로 변경한다.
27     duty = map(adcValue, 0, 1023, 0, 100);
28
29     // LED를 duty ms 만큼 점등한다.
30     digitalWrite(ledPin, HIGH);
31     delay(duty);
32
33     // 나머지 시간에는 소등시킨다.
34     digitalWrite(ledPin, LOW);
35     delay(100-duty);
36
37     // 시리얼 통신으로 ADC 값과 Duty를 출력한다.
38     Serial.print("ADC Value: ");
39     Serial.print(adcValue);
40     Serial.print(". Duty cycle: ");
41     Serial.print(duty);
42     Serial.println("");
43 }

```

## 6.1.5 포텐쇼미터 (가변저항 조절) - DIY

- DIY 응용 문제
1. delay함수를 사용하지 말고 4.2절의 예제를 참고하여 PWM 단자를 이용하여 LED의 밝기를 조절해 보자.
- PWM을 지원하는 디지털 3 번 핀에 단색 LED를 연결. (330 ohm 저항 연결)





## 6.1.6 포텐쇼미터 (DIY: code-2, pwm)

```

6 // 0번 아날로그핀을 포텐쇼미터 입력으로 설정한다.
7 const int potentiometerPin = 0;
8
9 //13번 핀에 연결되어 있는 내장 LED를 출력으로 사용한다.
10 const int ledPin = 13;
11
12 // #3 pin is defined to PWM output pin
13 const int pwmOutputPin = 3;
14
15 void setup() {
16 // 13번 핀을 출력으로 설정한다.
17 pinMode(ledPin, OUTPUT);
18 // 시리얼 통신을 설정한다.
19 Serial.begin(9600);
20 }

```

COM5

```

ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1023, Duty cycle: 100%, pwm: 255
ADC Value: 1022, Duty cycle: 99%, pwm: 254
ADC Value: 1023, Duty cycle: 100%, pwm: 255

```

```

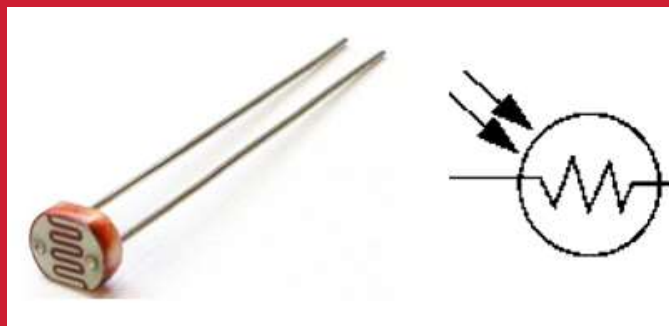
22 void loop(){
23   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
24   int duty;      // LED 점멸 주기 (0~100%)
25   int pwm;       // pwm 출력용
26
27   // 포텐쇼미터 값을 읽는다.
28   adcValue = analogRead(potentiometerPin);
29   // 포텐쇼미터 값을 0~100의 범위로 변경한다.
30   duty = map(adcValue, 0, 1023, 0, 100);
31
32   // LED를 duty ms 만큼 점등한다.
33   digitalWrite(ledPin, HIGH);
34   delay(duty);
35   // 나머지 시간에는 소등시킨다.
36   digitalWrite(ledPin, LOW);
37   delay(100-duty);
38
39   // pwmOutputPin Led ON
40   pwm = map(adcValue, 0, 1023, 0, 255);
41   analogWrite(pwmOutputPin, pwm);
42   // 시리얼 통신으로 ADC 값과 Duty를 출력한다.
43   Serial.print("ADC Value: ");
44   Serial.print(adcValue);
45   Serial.print(". Duty cycle: ");
46   Serial.print(duty);
47   Serial.println("%");
48 }

```



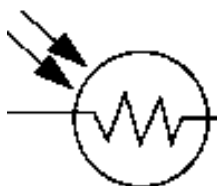
## 6.2 CdS, LDR

# 조도센서



# 6.2 조도 센서 (빛의 밝기 측정)

## CdS 센서



- ✓ CdS 분말을 세라믹 기판 위에 압축하여 제작
- ✓ 빛이 강할 수록 저항 값이 감소 → 광 가변저항
- ✓ ADC를 이용하여 변화된 저항에 전압을 인가하여 전압의 변화를 감지
- ✓ 자동 조명장치, 조도 측정 등에 사용

### 럭스

다른 뜻에 대해서는 [Lux](#) 문서를 참조하십시오.

럭스(lux, 기호 **lx**)는 빛의 **조명도**를 나타내는 **SI 단위**이다. 럭스는 **루멘**에서 유도

$$1 \text{ lx} = 1 \text{ lm/m}^2 = 1 \text{ cd}\cdot\text{sr}\cdot\text{m}^{-2}$$

럭스의 예 [\[편집\]](#)

I 밝기차	예
$10^{-5} \text{ lux}$	가장 밝은 별(시리우스)의 빛 <sup>[1]</sup>
$10^{-4} \text{ lux}$	하늘을 덮은 완전한 별빛 <sup>[1]</sup>
$0.002 \text{ lux}$	대기광이 있는 달 없는 맑은 밤 하늘 <sup>[1]</sup>
$0.01 \text{ lux}$	초승달
$0.27 \text{ lux}$	맑은 밤의 보름달 <sup>[1][2]</sup>
$1 \text{ lux}$	열대 위도를 덮은 보름달 <sup>[3]</sup>
$3.4 \text{ lux}$	맑은 하늘 아래의 어두운 황혼 <sup>[4]</sup>
$50 \text{ lux}$	거실 <sup>[5]</sup>
$80 \text{ lux}$	복도/화장실 <sup>[6]</sup>
$100 \text{ lux}$	매우 어두운 낮 <sup>[1]</sup>
$320 \text{ lux}$	권장 오피스 조명 (오스트레일리아) <sup>[7]</sup>
$400 \text{ lux}$	맑은 날의 해뜰이 뜨는 해넘이
$1000 \text{ lux}$	인공 조명 <sup>[1]</sup> ; 일반적인 TV 스튜디오 조명
$10,000\text{--}25,000 \text{ lux}$	낮 (직사광선이 없을 때) <sup>[1]</sup>
$32,000\text{--}130,000 \text{ lux}$	직사광선

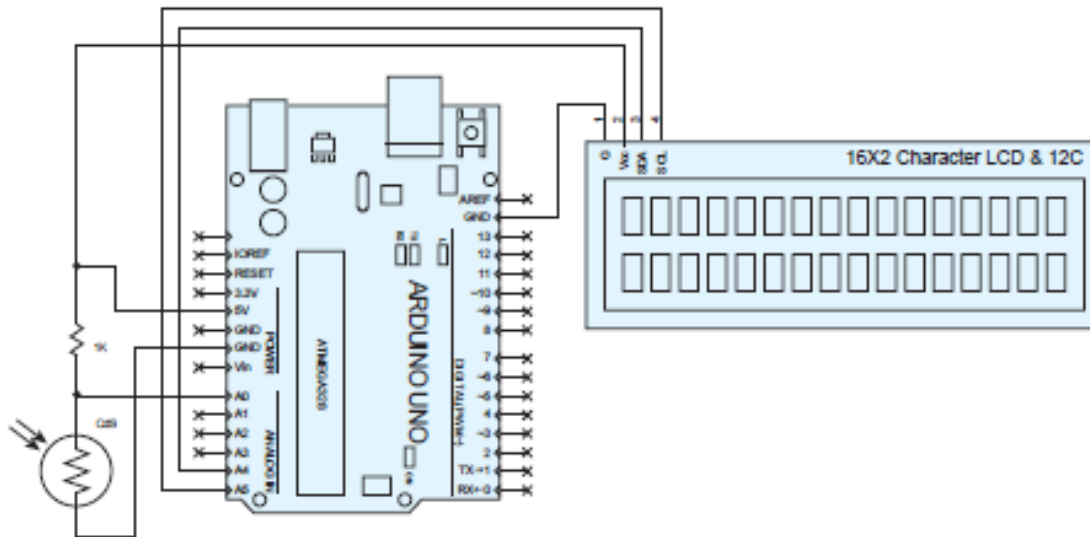
## 6.2.1 조도 센서 (빛의 밝기 측정)

### EX 6.2 빛 입력 (1/3)

**실습목표** CdS 셀을 이용하여 조도를 측정해 보자.

1. CdS 셀로 측정된 조도를 아날로그 핀을 통하여 0~1023 범위로 읽는다.
2. ADC 값을 LCD 모듈로 0~100%의 범위로 출력한다.

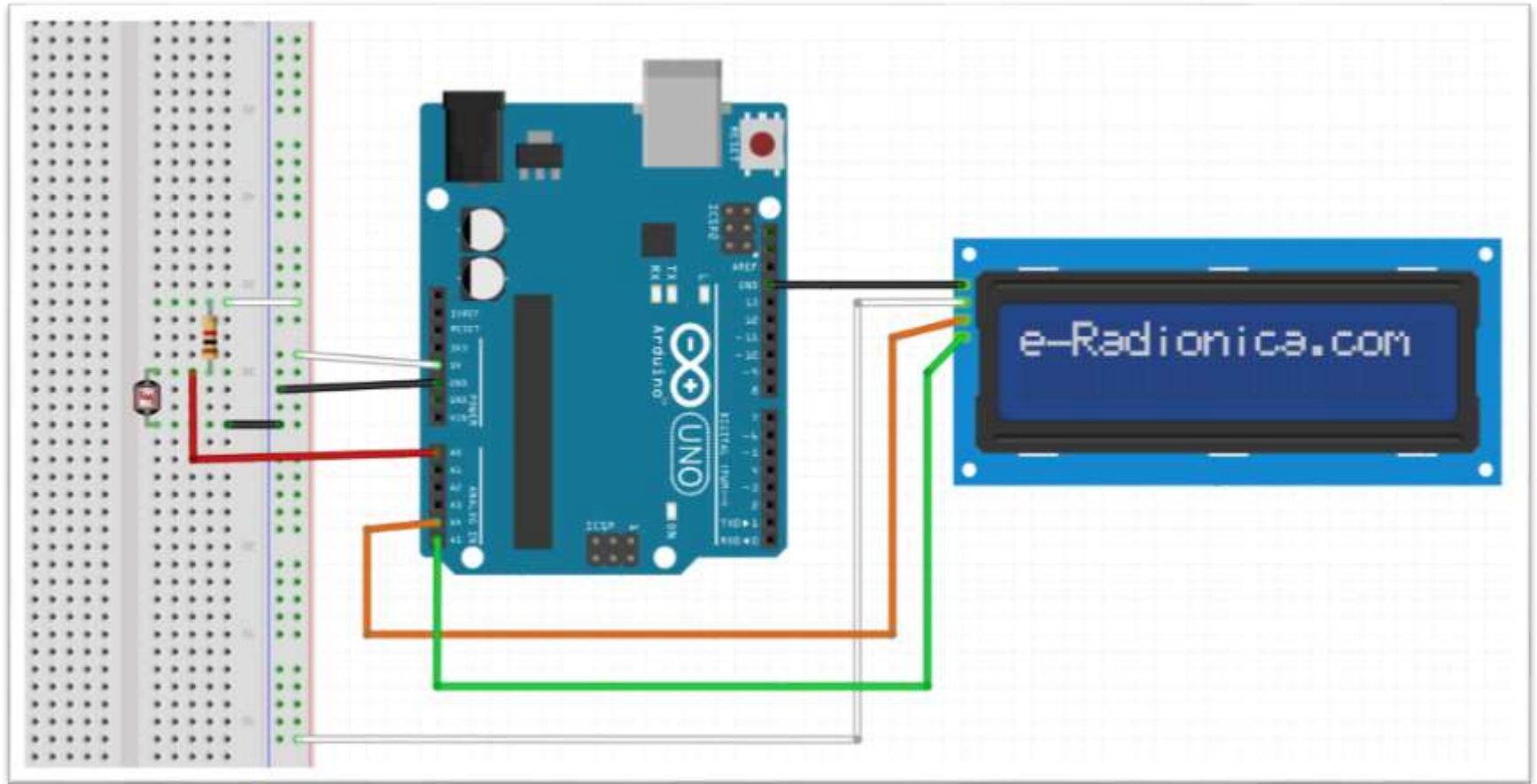
- Hardware**
1. CdS셀과 1k $\Omega$ 저항을 연결한 뒤 저항의 한쪽 끝은 5V에 CdS셀의 한쪽 끝은 GND에 연결한다.
  2. 저항과 CdS셀 사이를 아날로그입력핀 A0에 연결한다.
  3. I2C LCD 모듈의 Vcc, GND를 Arduino의 5V, GND에 연결한다.
  4. I2C LCD 모듈의 SDA는 A4에 SCL은 A5에 연결한다.



## 6.2.1 조도 센서 (빛의 밝기 측정)

EX 6.2

빛 입력 (1/3)



**CdS는 10 kΩ 저항과 직렬로 연결**

## 6.2.2 조도 센서 (빛의 밝기 측정)

### EX 6.2

### 빛 입력 (2/3)

#### Commands

- `analogRead`(아날로그 핀번호)

아날로그 핀에서 아날로그 값을 읽는다. 0~5V사이의 전압을 0~1023 사이의 값으로 표현한다.

- `map`(변수명, 범위1 최소값, 범위1 최대값, 범위2 최소값, 범위2 최대값)

변수명의 변수의 범위1의 범위와 범위2의 범위에 매칭시킨다. 즉 변수가 0~100의 범위를 갖고

이를 50~200의 범위로 매칭하려면 '`map`(변수명, 0, 100, 50, 200)'의 명령어로 매칭시킬 수 있다.

- `LiquidCrystal_I2C`(I2C 주소, 가로 글자수, 세로 글자수)

→ LCD 모듈이 연결된 I2C 주소와 LCD의 가로, 세로 글자수를 설정한다.

- `lcd.init()`; LCD 모듈을 설정한다.

- `lcd.clear()`; lcd란 이름의 LCD 모듈의 화면의 모든 표시를 지우고 커서를 왼쪽 위로 옮긴다.

- `lcd.home()`; lcd란 이름의 LCD 모듈의 커서를 왼쪽 위로 옮긴다.

- `lcd.setCursor`(행, 열); lcd란 이름의 LCD 모듈의 커서를 원하는 위치로 이동시킨다.

- `lcd.print`(데이터); lcd란 이름의 LCD 모듈에 데이터를 출력한다.

- `lcd.noBacklight()`; lcd란 이름의 LCD 모듈의 백라이트를 소등한다.

- `lcd.backlight()`; lcd란 이름의 LCD 모듈의 백라이트를 점등한다.

## 6.2.3 조도 센서 (빛의 밝기 측정)

### EX 6.2 빛 입력 (3/3)

- Sketch 구성**
1. CdS 센서로부터 읽은 빛의 밝기를 I2C 16X2 LCD 모듈로 출력하기 위해 CdS 센서 입력 핀 설정과 LCD 모듈 설정을 한다.
  2. CdS 센서로부터 읽은 ADC값을 LCD에 출력하고 밝기를 %로 나타낸다.

**실습 결과** ADC 값과 조도값이 표시된다..

```
ADC: 500
Illuminance: 50 %
```



## 6.2.4 조도 센서 (빛의 밝기 측정) : code-1

```

6 // I2C 통신 라이브러리 설정
7 #include <Wire.h>
8 // I2C LCD 라이브러리 설정
9 #include <LiquidCrystal_I2C.h>
10
11 // LCD I2C address 설정
12 LiquidCrystal_I2C lcd(0x3f, 16, 2);
13
14 // 0번 아날로그핀을 CdS 셀 입력으로 설정한다.
15 const int CdSPin = 0;
16

```

```

17 void setup() {
18
19 // 16X2 LCD 모듈 설정하고 백라이트를 켜다.
20 lcd.init(); // LCD 설정
21 lcd.backlight();
22
23 // 메시지를 표시한다.
24 lcd.print("ex 6.2");
25 lcd.setCursor(0, 1);
26 lcd.print("CdS Cell Test");
27 // 3초동안 메시지를 표시한다.
28 delay(3000);
29
30 // 모든 메시지를 삭제한 뒤
31 // 숫자를 제외한 부분들을 미리 출력시킨다.
32 lcd.clear();
33 lcd.setCursor(0, 0);
34 lcd.print("ADC : ");
35 lcd.setCursor(0, 1);
36 lcd.print("Illuminance:");
37 lcd.setCursor(15, 1);
38 lcd.print("%");
39 }

```



## 6.2.4 조도 센서 (빛의 밝기 측정) : code-2

```

41 void loop(){
42
43   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
44   int illuminance; // 현재의 밝기. 0~100%
45
46   // CdS cell을 통하여 입력되는 전압을 읽는다.
47   adcValue = analogRead(CdSPin);
48   // 아날로그 입력 값을 0~100의 범위로 변경한다.
49   illuminance = map(adcValue, 0, 1023, 100, 0);
50
51   // 전에 표시했던 내용을 지우고
52   // LCD에 ADC 값과 밝기를 출력한다.
53   // 지우지 않으면 이전에 표시했던 값이 남게 된다.
54

```



ARnn\_cds.png 로 저장...

```

55   // 전에 표시했던 내용을 지운다.
56   lcd.setCursor(9,0);
57   lcd.print(" ");
58   // ADC 값을 표시한다
59   lcd.setCursor(9,0);
60   lcd.print(adcValue);
61
62   // 전에 표시했던 내용을 지운다.
63   lcd.setCursor(13,1);
64   lcd.print(" ");
65   // 밝기를 표시한다
66   lcd.setCursor(12,1);
67   lcd.print(illuminance);
68
69   delay(1000);
70 }

```

## 6.2.5 조도 센서 (빛의 밝기 측정) - DIY

### DIY

### 응용 문제

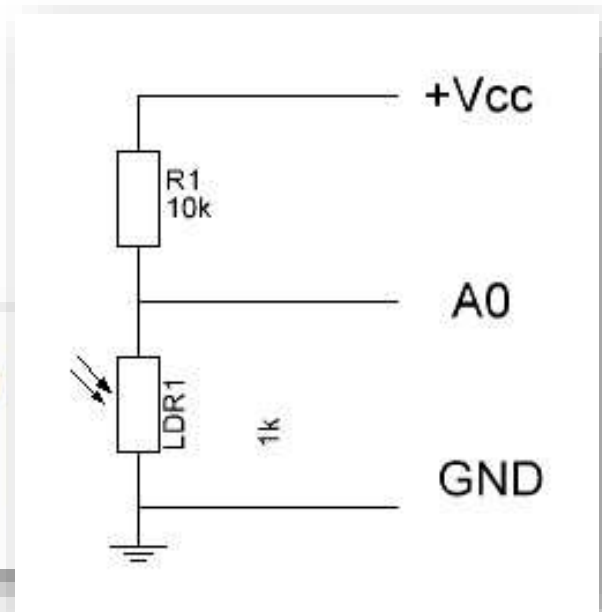
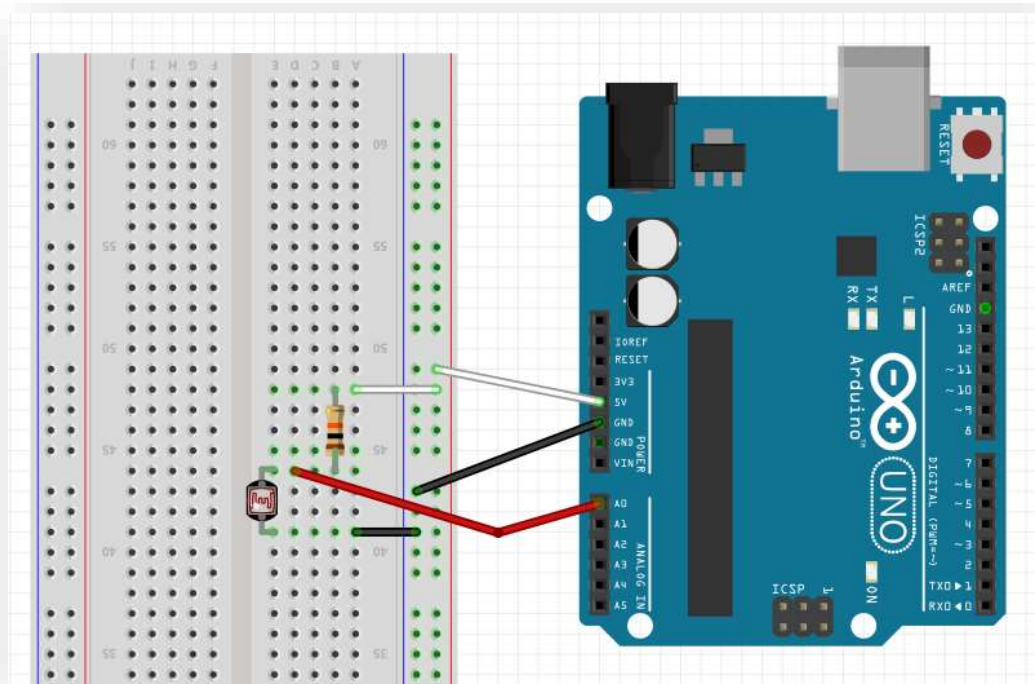
1. 손으로 가렸을 때 LCD 모듈의 백라이트가 켜지고, 가리지 않았을 때 백라이트가 꺼지도록 수정하여 보자.
2. 손으로 가렸을 때 13번핀에 연결된 LED가 켜지고, 가리지 않았을 때 꺼지도록 수정하여 보자.

위의 1, 2가 동시에 실행되는 아두이노 코드를 완성하시오.

→ ARnn\_cds\_project.ino 로 저장 제출

# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로



**Parts : photocell CdS, R (10 k $\Omega$  X 1)**

저항과 광센서 사이에서 전압 값을 **A0**로 측정

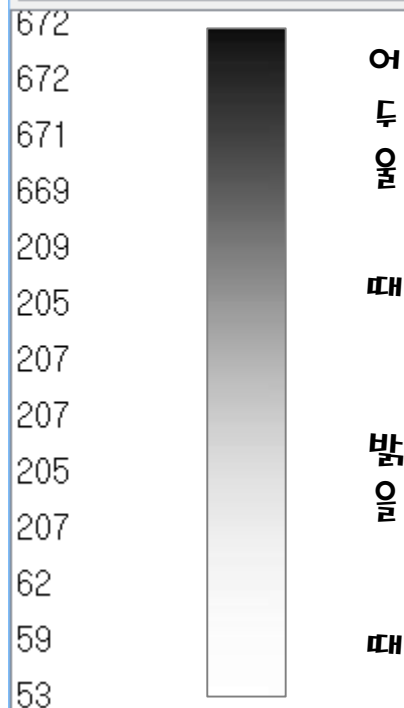
# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 - 측정 1.

CdS\_start

```
1 #define CDS_INPUT 0
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8
9   int value = analogRead(CDS_INPUT);
10  Serial.println(value);
11
12  delay(1000);
13 }
14
```

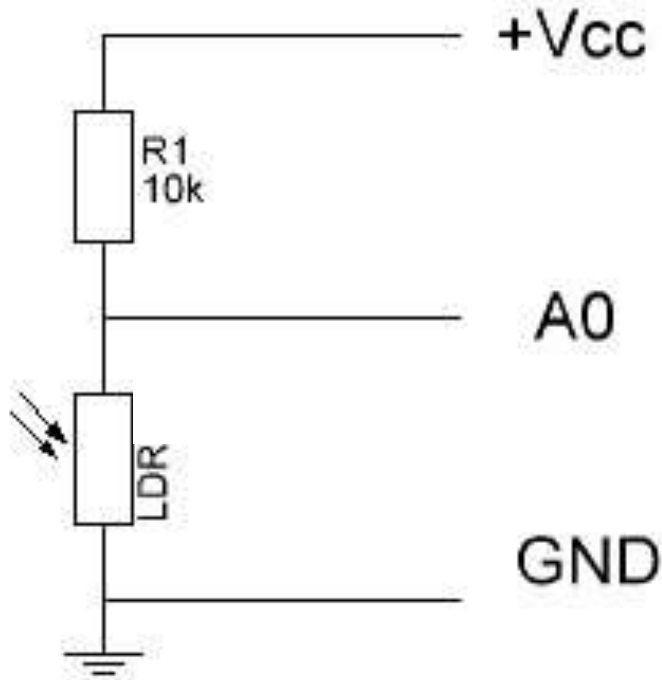
COM11 (Arduino/Genuino Uno)



어두우면 측정 값이 커지고 밝을수록 값이 작아진다 ???

# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 분석 (1/2)



**LDR's (Light dependent resistors) have a low resistance in bright light and a high resistance in the darkness.**

**If you would use the LDR as the lower part of a voltage divider, then in darkness there would be a high voltage over the LDR, while in bright light, there would be a low voltage over that resistor.**

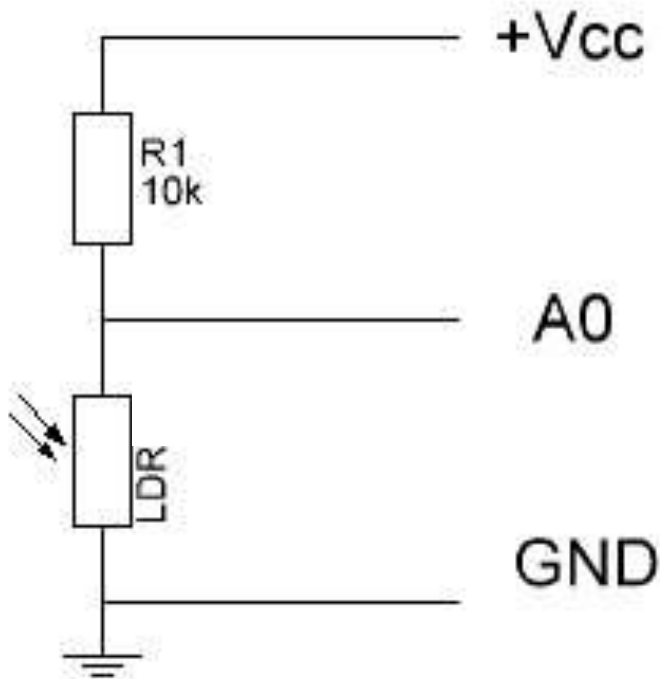
어두우면 측정 값이 작아지고 밝을수록 값이 커져야 된다.  
그리고 측정 값은 **lux**로 표현된다.

$$V_{out} = \frac{R_{ldr}}{R_1 + R_{ldr}} * V_{cc}$$

**A0에서 측정되는 LDR**  
**양단의 전압 = V<sub>out</sub>**

# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 분석 (2/2)



- (a)  $V_{out} = \frac{R_{ldr}}{(R_1 + R_{ldr})} * V_{CC}$ ,
- (b)  $R_{ldr} = \frac{10 * V_{out}}{(5 - V_{out})} (k\Omega)$ ,
- (c)  $V_{out} = value * V_{CC}/1023$ ,
- (d)  $Lux = \frac{500}{R_{ldr}}$ ,
- (e)  $Lux = (\frac{2500}{V_{out}} - 500)/10 (lux)$ .

$$V_{out} = \frac{R_{ldr}}{R_1 + R_{ldr}} * V_{cc}$$

**A0**에서 측정되는 **LDR**  
양단의 전압 = **V<sub>out</sub>**

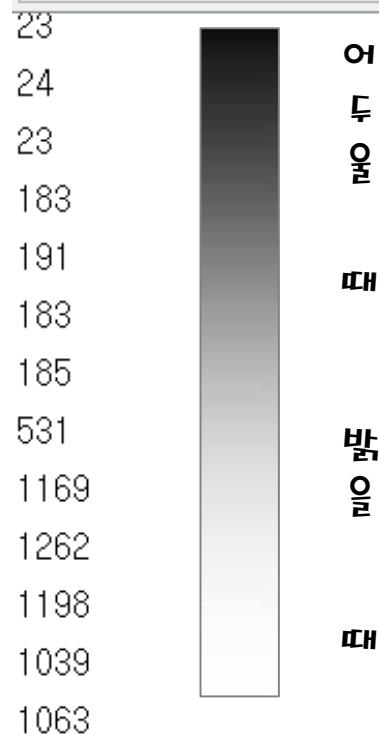
# [참고] 조도 센서 (빛의 밝기 정밀 측정)

## CdS 센서 회로 - 측정 2.

```

sketch08_CdS2
1 // lux
2 #define CDS_INPUT 0
3
4 void setup() {
5   Serial.begin(9600);
6 }
7 void loop() {
8   int value = analogRead(CDS_INPUT);
9   Serial.println(int(luminosity(value)));
10  delay(1000);
11 }
12
13 //Voltage to Lux
14 double luminosity (int RawADC0){
15   double Yout=RawADC0*5.0/1023; // 5/1023 (Vin = 5 V)
16   double lux=(2500/Yout-500)/10;
17   // lux = 500 / Rldr, Yout = Ildr*Rldr = (5/(10 + Rldr))*Rldr
18   return lux;
19 }
  
```

COM11 (Arduino/Genuino Uno)



밝을수록 측정 값이 커지고  
어두울수록 값이 작아진다 !!!



# 온도센서





## 6.3 온도 센서 (주변 온도측정)

LM35



- ✓ 온도 측정을 위한 센서 ( $-55^{\circ}\text{C}$  and  $150^{\circ}\text{C}$ )
- ✓ 전원과 접지를 연결하면 Vout에 0~500도까지 0.01V 단위로 전압 출력이 발생
- ✓ ADC를 이용하여 이 값을 읽어 온도를 측정
- ✓ [단점] 온도 측정 오차가 크다. (TMP36이 정확)

# 6.3 온도 센서 (주변 온도측정)

Google

lm35 voltage temperature relation



전체

이미지

동영상

뉴스

더보기

설정

도구



검색결과 약 50,700개 (0.40초)

**LM35.** The **LM35 temperature** sensor provides an output of 10mV per degree Celsius, with an accuracy of 0.5°C at 25°C. It can be powered by any DC **voltage** in the range 4 – 30v. The operating range is -55°C to +150°C. The output of the LM 35 is 10 mV (0.01 volts) per degree Celsius.



LM35 Temperature detector

[LM35 Precision temperature measurement - Back](#)  
[www.magics-notebook.com/lm35.html](http://www.magics-notebook.com/lm35.html)

 이 결과에 관한 정보  사용자 의견

[LM35 temperature equation? - Arduino Forum](#)

<https://forum.arduino.cc/index.php?topic=99421.0>  이 페이지 번역하기

2012. 4. 1. - 답글 4개 - 작성자 3명

**Equation** to get **temperature** using **LM35**: temp = (5.0 ... supply you use, the analog **voltage** reading will range from about 0V (ground) to about ...

<b>LM35: Temperature</b> Readings are not right	게시물 15개	2015년 4월 5일
Guide: accurately read an <b>LM35</b>	게시물 15개	2010년 2월 7일
<b>LM35</b> thermometer	게시물 15개	2008년 10월 18일

[forum.arduino.cc](#) 검색결과 더보기

what is conversion system of LM35 (temperature sensor) in Celsius ...

## 6.3 온도 센서 (주변 온도측정)

### LM35 온도-전압 특성

#### LM35



- Three-Pin
  - TO-92 Package
  - Easy to Use
  - 4V-20V Operating Range
  - 60 $\mu$ A Max Current Draw
- Analog Output
  - 0.5 $^{\circ}C$  Accuracy at 25 $^{\circ}C$
  - Easily read by Arduino
  - Highly Linear Transfer Function
  - 10 ( $mV/^{\circ}C$ ) Slope

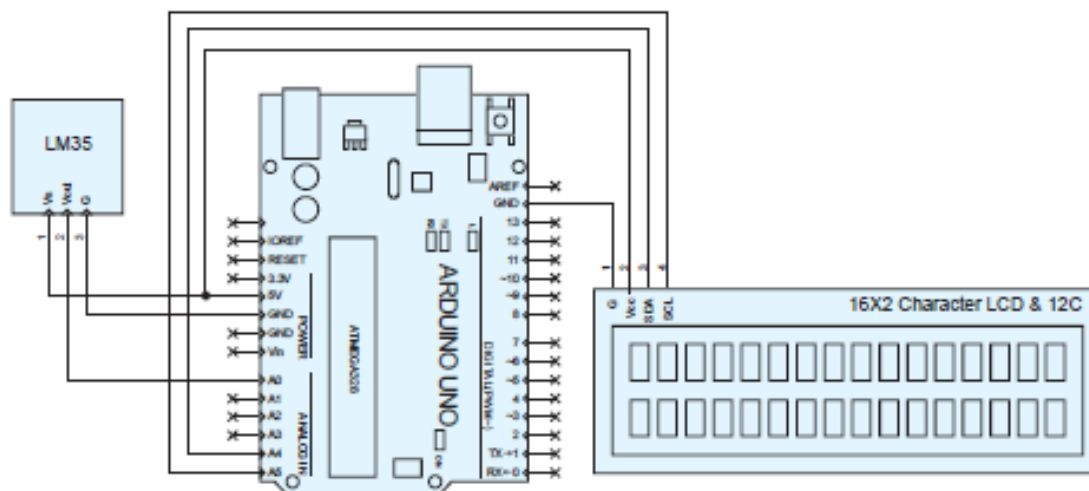
✓ 전원과 접지를 연결하면  $V_{out}$ 에 0~500  $^{\circ}C$  까지 0.01V 단위로 전압 출력(0~5000mV)이 발생

# 6.3.1 온도 센서 (주변 온도측정)

## EX 6.3 온도 측정 (1/3)

- 실습목표**
1. LM35 센서로부터 현재 온도를 아날로그 입력핀으로 측정한다.
  2. 측정된 값을 LCD에 표시해 보자.

- Hardware**
1. LM35의 Vs와 G 핀을 Arduino의 5V와 GND에 연결한다.
  2. LM35의 Vout을 아날로그입력핀 A0에 연결한다.
  3. I2C LCD 모듈의 Vcc, GND를 Arduino의 5V, GND에 연결한다.
  4. I2C LCD 모듈의 SDA는 A4에 SCL은 A5에 연결한다.



## 6.3.2 온도 센서 (주변 온도측정)

### EX 6.3 온도 측정 (2/3)

- Commands**
- `analogRead`(아날로그 핀번호)  
아날로그 핀에서 아날로그 값을 읽는다. 0~5V사이의 전압을 0~1023 사이의 값으로 표현한다.
  - `map`(변수명, 범위1 최소값, 범위1 최대값, 범위2 최소값, 범위2 최대값)  
변수명의 변수의 범위1의 범위와 범위2의 범위에 매칭시킨다. 즉 변수가 0~100의 범위를 갖고 이를 50~200의 범위로 매칭하려면 '`map(변수명, 0, 100, 50, 200)`'의 명령어로 매칭시킬 수 있다.
  - `LiquidCrystal_I2C`(I2C 주소, 가로 글자수, 세로 글자수)  
LCD 모듈이 연결된 I2C 주소와 LCD의 가로, 세로 글자수를 설정한다.
  - `lcd.init( );` LCD 모듈을 설정한다.
  - `lcd.clear( );` lcd란 이름의 LCD 모듈의 화면의 모든 표시를 지우고 커서를 왼쪽 위로 옮긴다.
  - `lcd.home( );` lcd란 이름의 LCD 모듈의 커서를 왼쪽 위로 옮긴다.
  - `lcd.setCursor(행, 열);` lcd란 이름의 LCD 모듈의 커서를 원하는 위치로 이동시킨다.
  - `lcd.print(데이터);` lcd란 이름의 LCD 모듈에 데이터를 출력한다.
  - `lcd.noBacklight();` lcd란 이름의 LCD 모듈의 백라이트를 소등한다.
  - `lcd.backlight();` lcd란 이름의 LCD 모듈의 백라이트를 점등한다.

## 6.3.3 온도 센서 (주변 온도측정)

### EX 6.3 온도 측정 (3/3)

- Sketch 구성**
1. LM35 입력을 받기 위한 아날로그 핀을 설정한다.
  2. ADC로 읽은 값과 실제 온도와의 관계는 연산을 통하여 계산한다.
  3. I2C LCD 모듈에 ADC값과 현재 온도를 출력한다.
  4. 온도를 출력 할 때 “°” 기호는 표 3.1 LCD 문자 코드표에서 찾아 코드를 이용하여 출력한다.

**실습 결과** ACD 값과 온도가 표시된다.

```
ADC: 250
Temp. is 25 °C
```



## 6.3.4 온도 센서 (주변 온도측정): code-1

```
ex_6_3
1 /*
2  예제 6.3
3  LM35를 이용한 온도 측정
4  */
5
6 // I2C 통신 라이브러리 설정
7 #include <Wire.h>
8 // I2C LCD 라이브러리 설정
9 #include <LiquidCrystal_I2C.h>
10
11 // LCD I2C address 설정
12 // PCF8574:0x27, PCF8574A:0x3F
13 LiquidCrystal_I2C lcd(0x27, 16, 2);
14
15 // 0번 아날로그핀을 LM35 입력으로 설정한다.
16 const int LM35Pin = 0;
17
```

```
18 void setup() {
19
20   // 16X2 LCD 모듈 설정하고 백라이트를 켜다.
21   lcd.init(); // LCD 설정
22   lcd.backlight();
23
24   // 메시지를 표시한다.
25   lcd.print("ex 6.3");
26   lcd.setCursor(0, 1);
27   lcd.print("Checking Temp.");
28
29   // 3초동안 메시지를 표시한다.
30   delay(3000);
31
32   // 모든 메시지를 삭제한 뒤
33   // 숫자를 제외한 부분들을 미리 출력시킨다.
34   lcd.clear();
35   lcd.setCursor(0, 0);
36   lcd.print("ADC : ");
37   lcd.setCursor(0, 1);
38   lcd.print("Temp. is ");
39
40   //LCD 문자표에서 '°' 기호를 직접 써준다
41   lcd.setCursor(13, 1);
42   lcd.write(B11011111); // '°' 기호 문자코드
43   lcd.setCursor(14, 1);
44   lcd.print("C"); // 'C'를 표시한다.
45 }
```

## 6.3.4 온도 센서 (주변 온도측정): code-2

```

47 void loop(){
48   int adcValue; // 실제 센서로부터 읽은 값 (0~1023)
49   long temp; // 현재의 온도
50
51   // LM35의 Vout을 읽는다.
52   adcValue = analogRead(LM35Pin);
53   // 온도값으로 환산한다. 오버플로우 방지를 위하여 500L로 표시한다.
54   // 500L의 경우 500을 32비트 long 형태의 숫자로 나타내 준다.
55   temp = (adcValue * 500L) / 1023;
56
57   // 전에 표시했던 내용을 지우고 LCD에 ADC 값과 온도를 출력한다.
58   // 지우지 않으면 이전에 표시했던 값이 남게 된다.
59   lcd.setCursor(9,0);
60   lcd.print(" ");
61   // ADC 값을 표시한다
62   lcd.setCursor(9,0);
63   lcd.print(adcValue);
64
65   // 전에 표시했던 내용을 지운다.
66   lcd.setCursor(10,1);
67   lcd.print(" ");
68   // 온도를 표시한다
69   lcd.setCursor(10,1);
70   lcd.print(temp);
71
72   delay(2000);
73 }

```





## 6.3.5 온도 센서 (주변 온도측정)

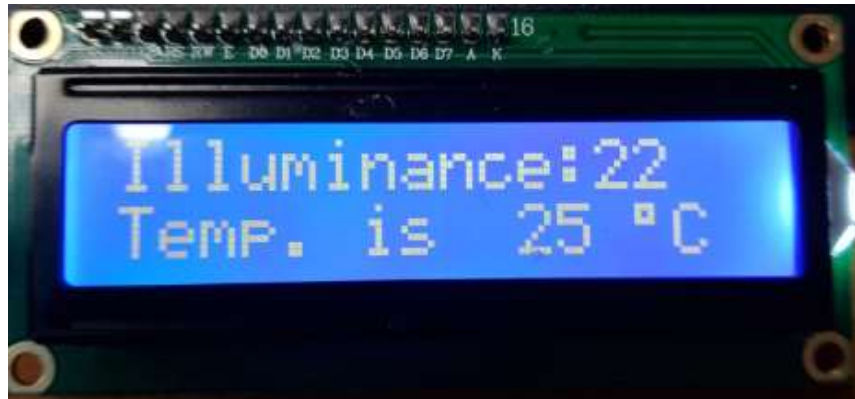
### DIY

예제 6.2를 참고하여 LCD에 현재 온도, 조도를 함께 표시해 보자.

### 응용 문제

아두이노 코드를 완성하시오.

→ ARnn\_cds\_lm35.ino 로 저장하고 제출



→ ARnn\_cds\_lm35.png 로 저장하고 제출



# [Practice]

## ◆ [wk11]

- **Arduino : Analog input I.**
- **Complete your project**
- **Submit folder : Arnn\_Rpt07**

# wk11 : Practice-07 : ARnn\_Rpt07

## ◆ [Target of this week]

- Complete your works
- Save your outcomes
- Upload all in github.

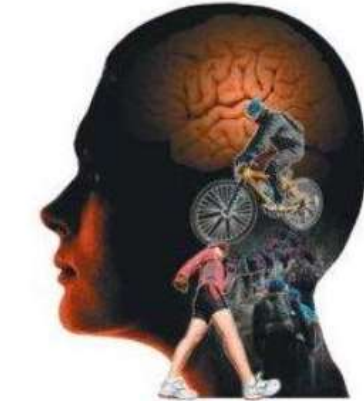
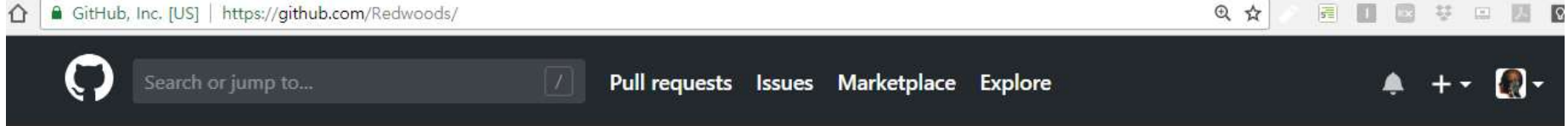
제출폴더명 : **ARnn\_Rpt07**

### 제출할 파일들

- ① **ARnn\_pwm.png**
- ② **ARnn\_cds.png**
- ③ **ARnn\_cds\_project.ino**
- ④ **ARnn\_cds\_lm35.ino**
- ⑤ **ARnn\_cds\_lm35.png**
- ⑥ **\*.ino**

## ● References & good sites

- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling
- ✓ <https://www.youtube.com> Youtube



**Redwoods Yi**

Redwoods

Add a bio

GimHae, Republic of Korea

chaos21c@gmail.com

Overview

Repositories 7

Stars 2

Followers 1

Following 0

## Pinned repositories

Customize your pinned repositories

Py

Lectures on coding python from scratch to the advanced level.

Jupyter Notebook

Arduino

Lectures on learning Arduino from scratch to the advanced level in iot environment.

Lec

All lectures by Redwoods in Inje University

Jupyter Notebook

hw-coding

Resource for lecture of Hardware Programming (2017, Inje university)


Arduino

171 contributions in the last year




Contribution settings

Redwoods/Arduino: Lect

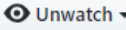
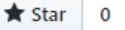

GitHub, Inc. [US] | https://github.com/Redwoods/Arduino

 Search or jump to...

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

Redwoods / Arduino





 1  0  0


[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#) [Settings](#)


Lectures on learning Arduino from scratch to the advanced level in iot environment.




Edit



Add topics

 2 commits  1 branch  0 releases  1 contributor

Branch: master  [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download](#)

 Redwoods 2018 start Latest commit 38ca9e0 28 minutes ago

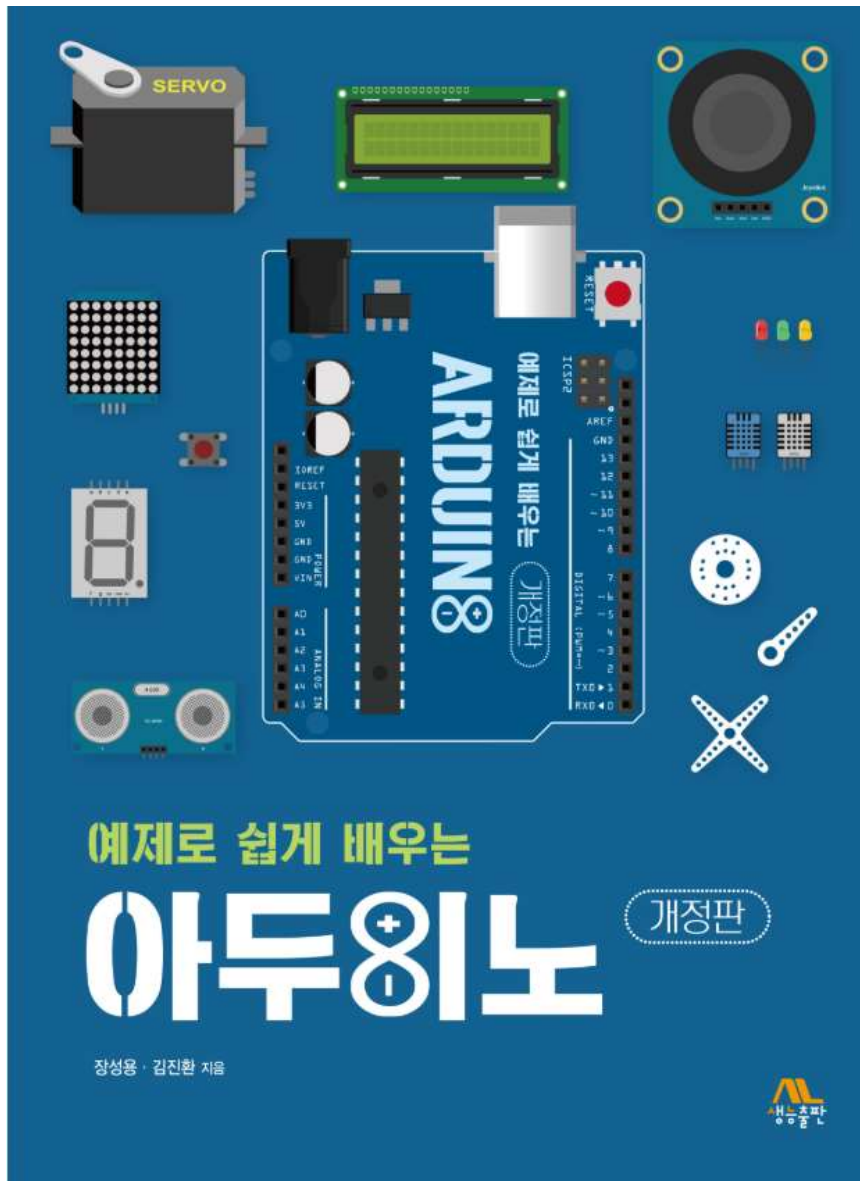
 ar-basic	2018 start	28 minutes ago
 ar-iot	2018 start	28 minutes ago
 README.md	Initial commit	43 minutes ago

 README.md 

## Arduino

---

Lectures on learning Arduino from scratch to the advanced level in iot environment.





[http://arduinostory.com/goods/goods\\_view.php?goodsNo=1000000306](http://arduinostory.com/goods/goods_view.php?goodsNo=1000000306)



## 상급키트 구성품

<b>1</b> 1EA  <b>아두이노 우노 R3 DIP</b> 아두이노 우노 R3 (DIP) 호환보드 기본 메인보드입니다.	<b>2</b> 1EA  <b>9V 배터리 홀더</b> 9V 배터리를 연결하여 아두이노에 외부전원을 공급할 수 있습니다.	<b>3</b> 1EA  <b>7세그먼트 4채널</b> 7세그먼트가 4개 연결된 형태의 부품입니다. 총 12개의 핀을 사용합니다.	<b>4</b> 1EA  <b>7세그먼트 1채널</b> 공통 음극 7세그먼트 시계나 점수 등의 숫자를 표현 할 때 많이 사용됩니다.
<b>5</b> 1EA  <b>74HC595N</b> 기본 메인보드입니다. 74HC595N LED, 드로메트릭스, NFD 제어 IC 입니다.	<b>6</b> 1EA  <b>65핀 점퍼 와이어</b> 브레드보드에 연결할 때 사용하는 65핀 점퍼와이어 입니다.	<b>7</b> 1EA  <b>무지개 점퍼선 F-M 20cm</b> M타입과 F타입이 양쪽으로 달린 무지개 점퍼선입니다.	<b>8</b> 1EA  <b>투명 부품 케이스 대,소</b> 키트 구성품을 담을 수 있는 투명 부품 케이스입니다.
<b>9</b> 1EA  <b>가변저항10K</b> 물리변 저항값이 바뀝니다. (0~10KΩ)	<b>10</b> 1EA  <b>1602 I2C LCD</b> 아두이노 16X2 I2C LCD 모듈입니다. LCD입니다.	<b>11</b> 1EA  <b>저항</b> 100, 220, 330, 1K, 2K, 4.7K, 10K, 47K, 100K	<b>12</b> 1EA  <b>브레드 보드 830홀</b> 브레드 보드 830홀(봉무형) 센서 테스트나, 회로 프로토타입을 작성할 때 사용됩니다.

<b>13</b> 1EA  <b>수동부저</b> 아두이노의 tone함수를 통해 소리를 내는 부저입니다.	<b>14</b> 6EA  <b>택트스위치 (12x12x7)</b> 스위치를 누르고 있을 경우만 ON됩니다.	<b>15</b> 3EA  <b>택트스위치 컵</b> (파랑, 노랑, 초록, 빨강, 하양) 택트스위치를 사용할 때 스위치간의 구분을 할 수 있습니다.	<b>16</b> 3EA  <b>조도센서</b> 빛을 감지하거나 빛의 밝기를 아날로그로 출력해주는 CDS 센서입니다.
<b>17</b> 5EA  <b>LED 5mm</b> (빨강, 노랑, 초록, 하양, 파랑) 기본으로 사용되는 LED입니다. 동작전압 : 2.2~2.4V 사용전류 : 20mA 미만	<b>18</b> 1EA  <b>헤더핀 1x40/2.54mm</b> 핀 간격은 2.54mm이며 헤더핀의 길이는 약 1.15cm입니다.	<b>19</b> 1EA  <b>USB케이블 50cm</b> PC와 아두이노 우노 보드를 연결하여 프로그램을 다운로드 할 때 사용합니다.	<b>20</b> 1EA  <b>저항값 카드</b> 저항값을 쉽게 확인 할 수 있는 카드입니다. 사이즈 : 60mm x 50mm
<b>21</b> 1EA  <b>능동부저</b> Signal 단자가 HIGH 일 때 약 2.5kHz의 음이 발생합니다.	<b>22</b> 1EA  <b>5V 1채널 릴레이 모듈</b> 아두이노의 디지털 핀과 모듈 하단의 IN 핀들을 연결해 릴레이를 제어할 수 있는 모듈입니다.	<b>23</b> 1EA  <b>8x8 도트 매트릭스 모듈</b> LED로 다양한 연출을 할 수 있습니다.	<b>24</b> 1EA  <b>4x4 16 키패드 모듈</b> 16개의 버튼을 사용할 수 있습니다.

# 아두이노 키트(Kit) : Part-2

<div>25</div> <div>1EA</div> <div></div> <div>무선 리모콘 키트</div> <div>핵파선을 사용해서 리모콘 기능을 구현할 수 있습니다.</div>	<div>26</div> <div>2EA</div> <div></div> <div>가을기 센서 스위치</div> <div>센서의 가을기에 따라 스위치 역할을 합니다.</div>	<div>27</div> <div>1EA</div> <div></div> <div>or</div> <div>사운드 센서 모듈</div> <div>아두이노와 호환되는 사운드센서 모듈입니다.</div>	<div>28</div> <div>1EA</div> <div></div> <div>불꽃 센서</div> <div>근거리 화재, 불꽃을 감지하는 센서입니다.</div>	<div>37</div> <div>1EA</div> <div></div> <div>DC 5V 스텝 모터</div> <div>28BYJ 28BYJ48 스텝 모터 중 저렴한 편에 속하는 모델입니다. 5개의 핀을 사용합니다.</div>	<div>38</div> <div>1EA</div> <div></div> <div>DS1302 RTC 모듈</div> <div>아두이노 등 마이크로컨트롤러에서 사용이 가능합니다.</div>	<div>39</div> <div>1EA</div> <div></div> <div>아두이노 우노 프로토 쉼드</div> <div>UNO 보드에서 회로를 간단히 짜기 위해 보드 위에 얹어 사용하는 쉼드입니다.</div>	<div>40</div> <div>1EA</div> <div></div> <div>3축 가속도 센서 모듈</div> <div>가속도를 측정할 수 있는 센서입니다.</div>
<div>29</div> <div>1EA</div> <div></div> <div>모터 드라이버 모듈</div> <div>ULN2003 스텝 모터 드라이버 모듈 5V ~ 12V를 사용할 수 있습니다.</div>	<div>30</div> <div>1EA</div> <div></div> <div>LM35 온도 센서</div> <div>온도를 아날로그 값으로 출력합니다.</div>	<div>31</div> <div>1EA</div> <div></div> <div>수위 센서 모듈</div> <div>센서가 액체에 잠긴 정도를 아날로그 값으로 출력합니다.</div>	<div>32</div> <div>1EA</div> <div></div> <div>SG90 서보모터</div> <div>Vcc, GND, 신호선, 총 3개의 핀이 있습니다. 로봇팔이나 자동차, 비행기 조종에 사용됩니다.</div>	<div>41</div> <div>1EA</div> <div></div> <div>5V DC모터</div> <div>5V DC모터</div>	<div>42</div> <div>1EA</div> <div></div> <div>인체 감지 센서 모듈</div> <div>핵파선을 이용해 움직임을 감지하는 센서입니다. 오선이 감지되면 HIGH 신호를 출력합니다.</div>	<div>43</div> <div>5EA</div> <div></div> <div>다이오드 1N4001</div> <div>다이오드 1N4001</div>	<div>44</div> <div>5EA</div> <div></div> <div>세라믹 캐패시터 (22pF)</div> <div>세라믹 캐패시터 (22pF)</div>
<div>33</div> <div>1EA</div> <div></div> <div>초음파 거리 센서 모듈</div> <div>5V를 사용하여 만직 거리는 2cm에서 500cm입니다.</div>	<div>34</div> <div>1EA</div> <div></div> <div>조이스틱 모듈</div> <div>기본적으로 조이스틱 모듈은 두개의 가변저항이 서로 수직으로 회전하는 형태로 되어 있습니다.</div>	<div>35</div> <div>1EA</div> <div></div> <div>온습도 센서 모듈</div> <div>아두이노 온습도 센서중 가장 대중적으로 사용되는 DHT11 디지털 센서입니다.</div>	<div>36</div> <div>1EA</div> <div></div> <div>RGB LED 모듈</div> <div>RGB LED 모듈로 RGB LED 세개를 하나로 모아둔 상품입니다.</div>	<div>45</div> <div>5EA</div> <div></div> <div>세라믹 캐패시터 (1uF)</div> <div>세라믹 캐패시터 (1uF)</div>	<div>46</div> <div>5EA</div> <div></div> <div>트랜지스터 2N2222</div> <div>트랜지스터 2N2222</div>	<div>47</div> <div>5EA</div> <div></div> <div>트랜지스터 BC547</div> <div>트랜지스터 BC547</div>	<div>48</div> <div>5EA</div> <div></div> <div>트랜지스터 BC557</div> <div>트랜지스터 BC557</div>
<div>49</div> <div>2EA</div> <div></div> <div>전해 캐패시터 (50V 10uF)</div> <div>전해 캐패시터 (50V 10uF)</div>	<div>50</div> <div>2EA</div> <div></div> <div>전해 캐패시터 (50V 100uF)</div> <div>전해 캐패시터 (50V 100uF)</div>	<div></div>					