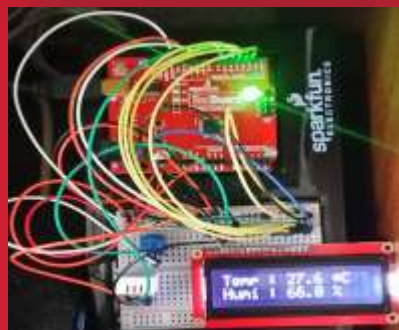


# Arduino-basic

[wk02]

## Serial Comm.



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)

<b>AR01</b>	염현제
<b>AR02</b>	강민수
<b>AR03</b>	구병준
<b>AR04</b>	김종민
<b>AR05</b>	박성철
<b>AR06</b>	이승현
<b>AR07</b>	이창호
<b>AR08</b>	변성현
<b>AR09</b>	손성빈
<b>AR10</b>	안예찬
<b>AR11</b>	유종인
<b>AR12</b>	이석민
<b>AR13</b>	이주원
<b>AR14</b>	정재영
<b>AR15</b>	차요신

<b>AR16</b>	하태성
<b>AR17</b>	강현이
<b>AR18</b>	신종원
<b>AR19</b>	최진솔
<b>AR20</b>	김경미
<b>AR21</b>	김경영
<b>AR22</b>	김규년
<b>AR23</b>	김민재
<b>AR24</b>	김영록
<b>AR25</b>	송다은
<b>AR26</b>	정지환
<b>AR27</b>	김종건



# Arduino SW

<http://fritzing.org/home/>

fritzing.org: Fritzing Fritzing

**fritzing** electronics made easy

Projects Parts Download Learning Services Contribute **FORUM** **FAB**

**fritzing APP** Download the free Fritzing App and start building immediately!

Fritzing is an open-source hardware initiative that makes electronics accessible as a creative material for anyone. We offer a software tool, a community website and services in the spirit of Processing and Arduino, fostering a creative ecosystem that allows users to document their prototypes, share them with others, teach electronics in a classroom, and layout and manufacture professional pcbs.

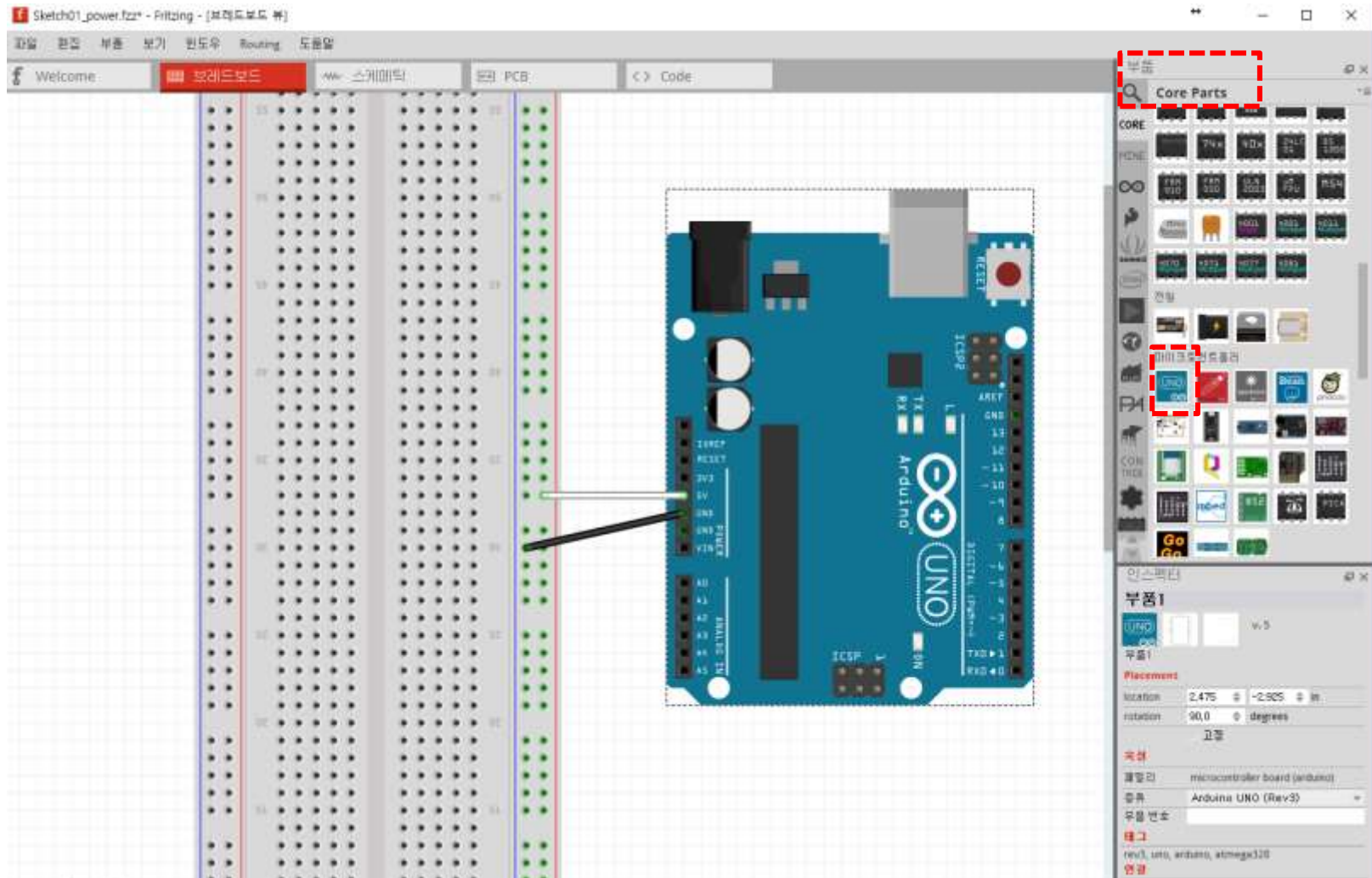
**Download and Start**  
Download our latest version 0.9.3b released on June 2, 2016 and start right away.

**Produce your own board**  
With Fritzing Fab you can easily and inexpensively turn your circuit into a real, custom-made PCB. Try it out now!

**Participate**  
Fritzing can only act as a creative platform if many



# Fritzing configuration – power

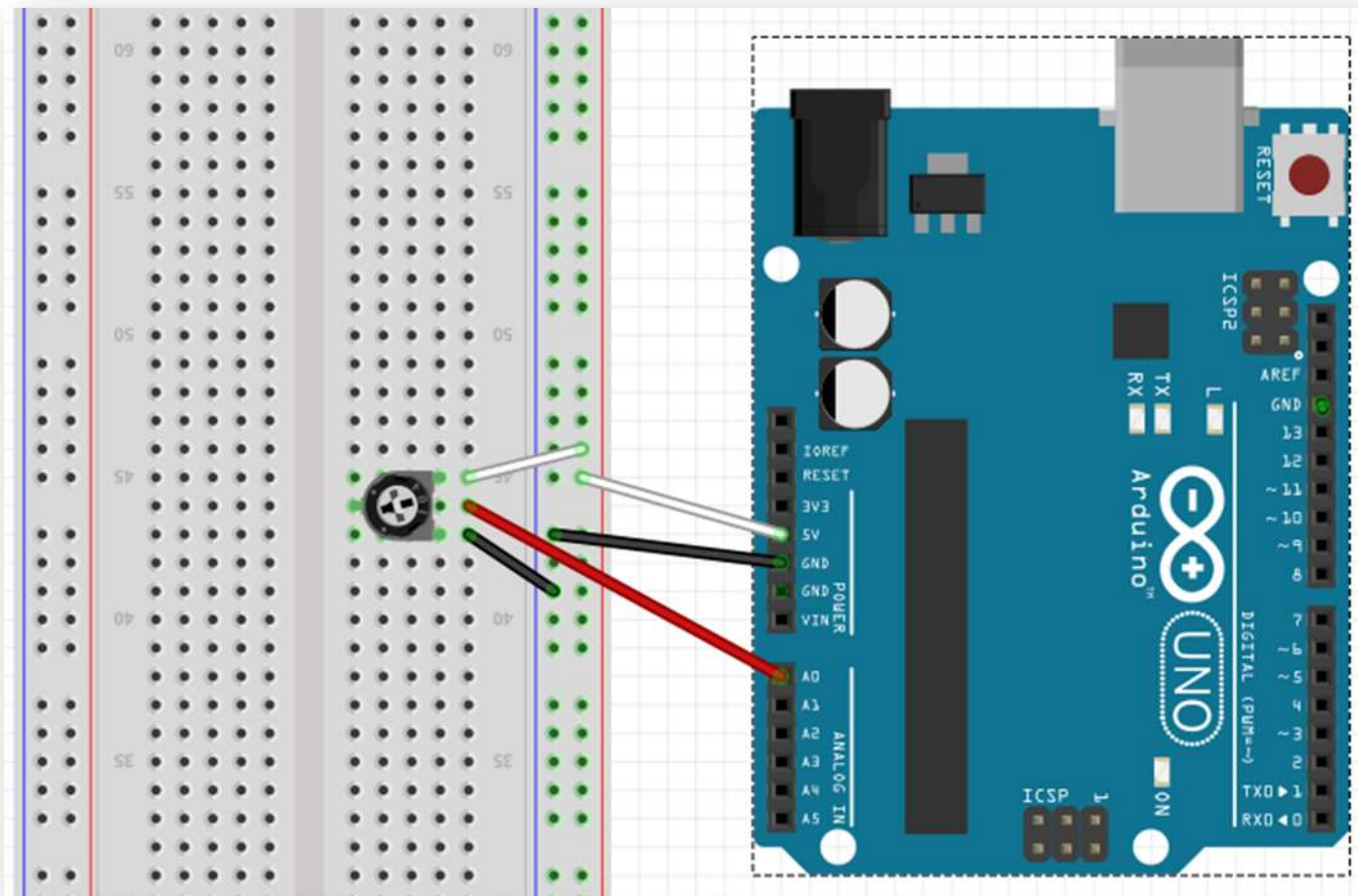




# Arduino circuits



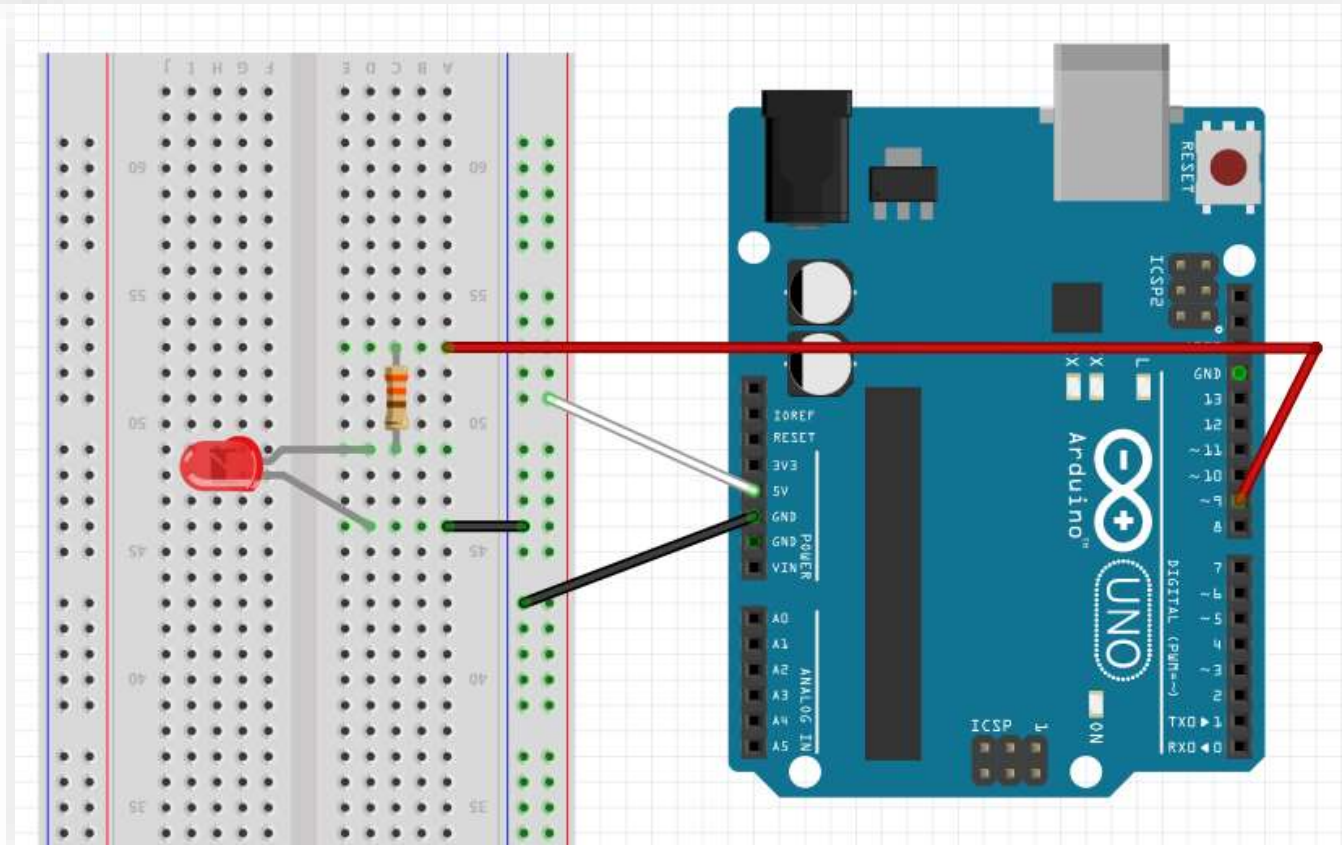
## 0.A1 Potentiometer (가변 저항기)



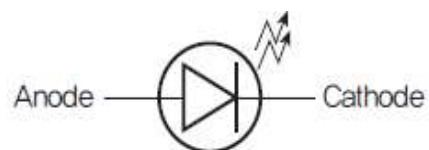
Parts : 가변저항기



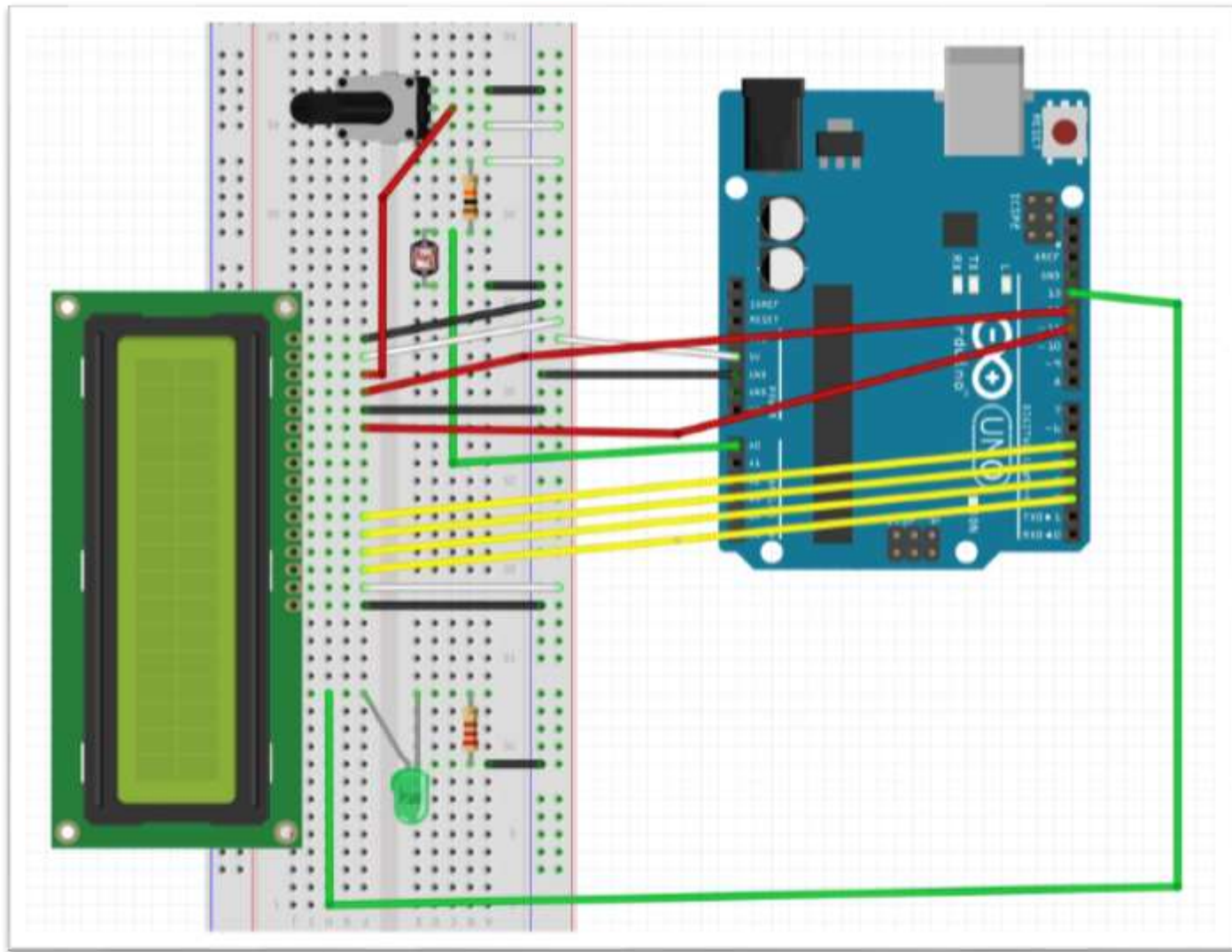
# 0.A2 single LED



**Parts : LED (1), R (330  $\Omega$  X 1)**



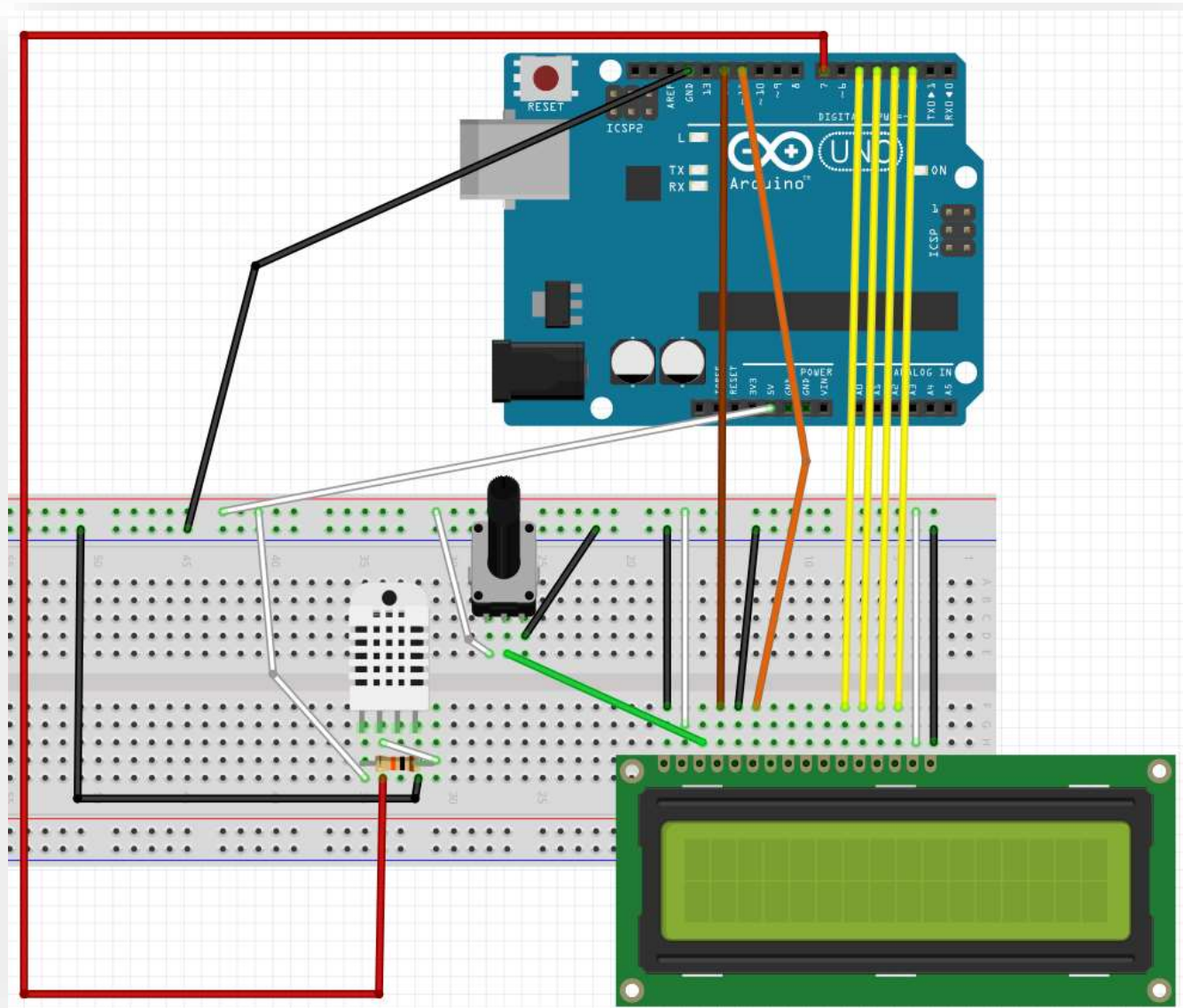
## 0.A5 Display of luminosity







# 0.A6 Display of Temperature & Humidity

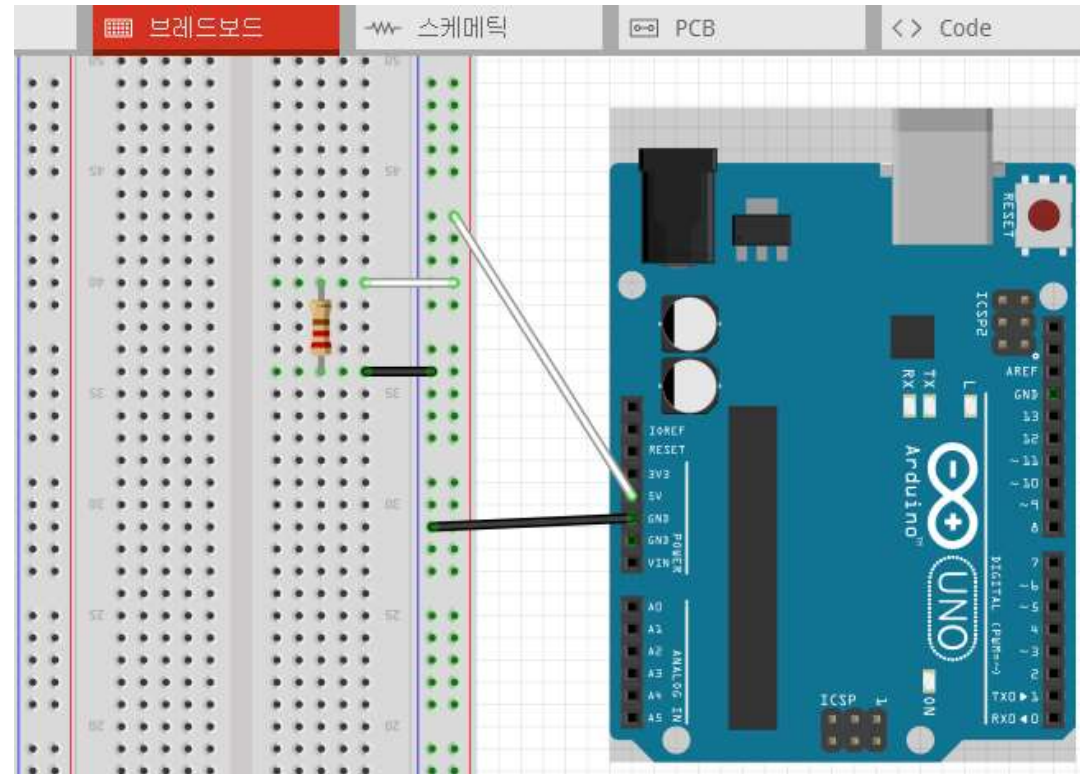
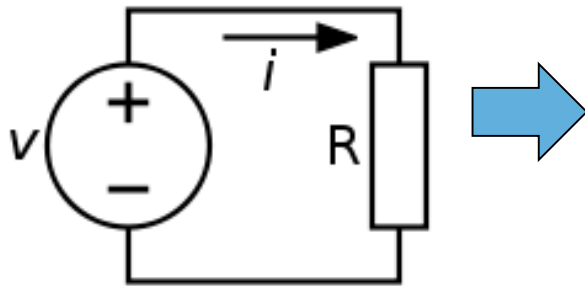




# Arduino circuits

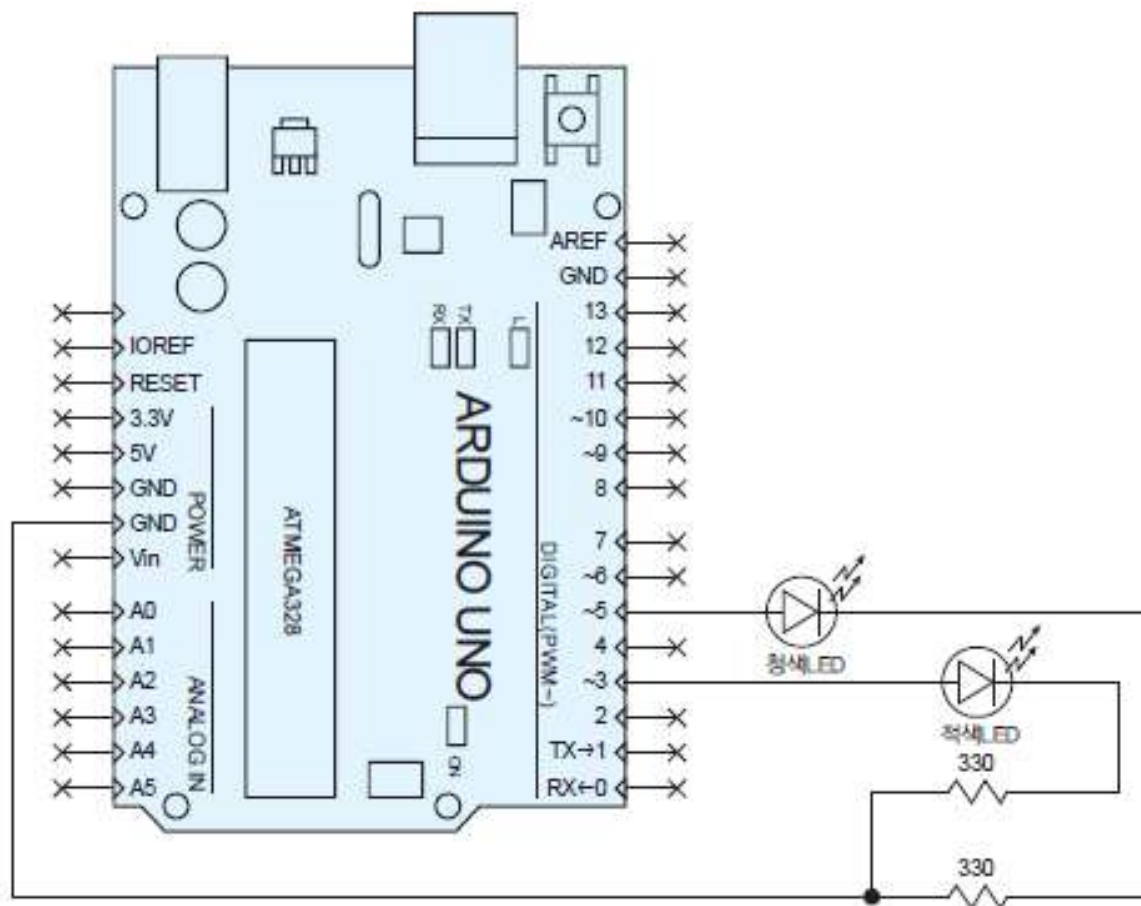
DIY

## [Fritzing] Simple R circuit



ARnn\_R.fzz

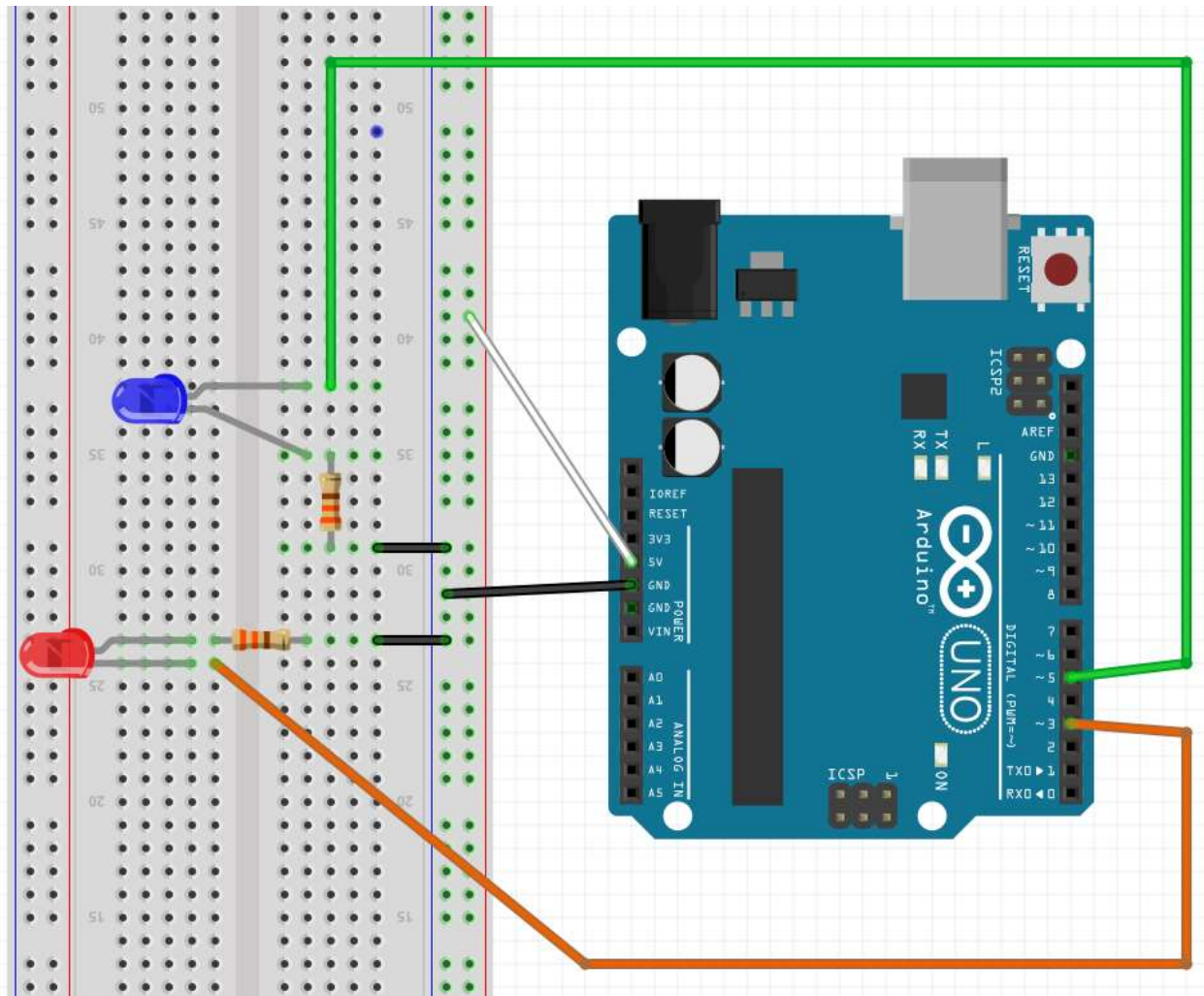
# DIY Two LEDs



**Parts : LED(청, 적) , R (330 Ω X 2)**

**ARnn\_2Led.fzz**

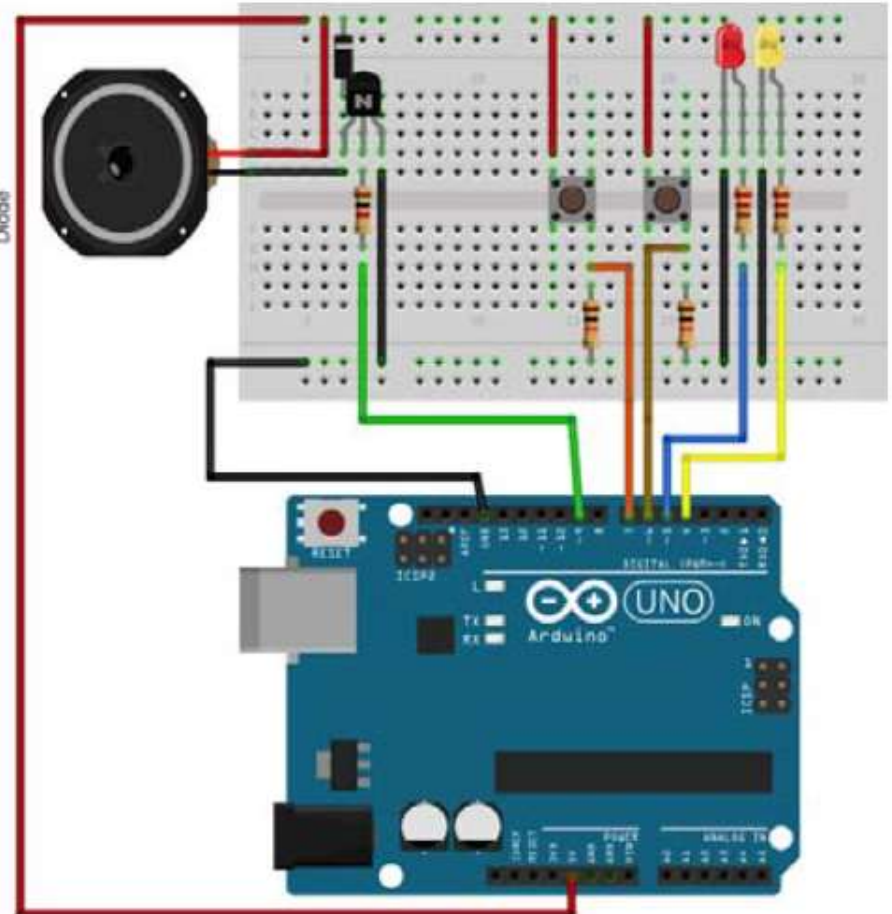
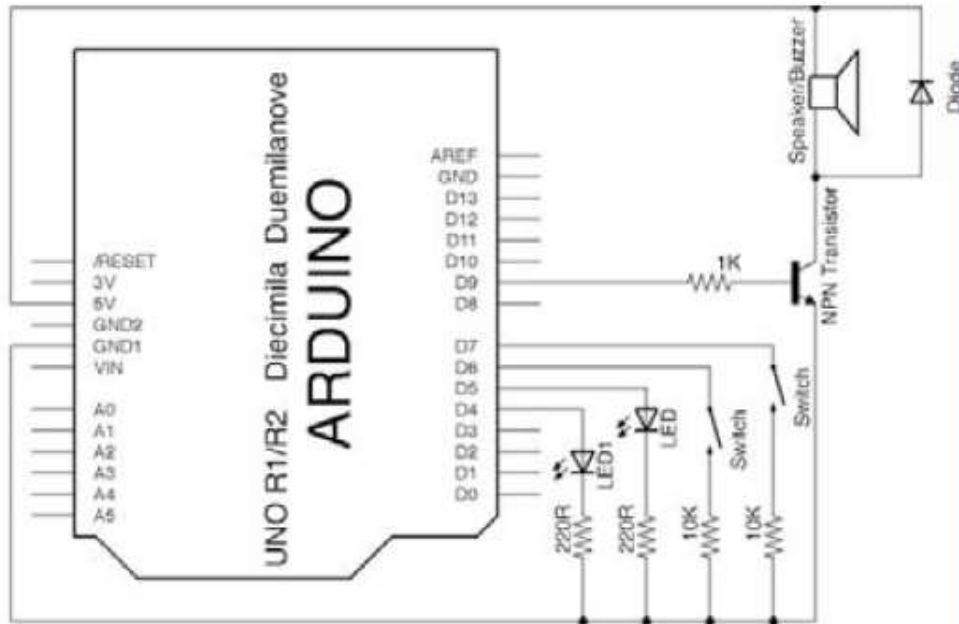
# DIY Two LEDs



ARnn\_2Led.fzz



# Arduino circuits : challenge – (Metronome)



Circuit schematic and breadboard connections diagram for a digital metronome



# Arduino SW: IDE



HOME BUY SOFTWARE PRODUCTS LEARNING FORUM SUPPORT BLOG

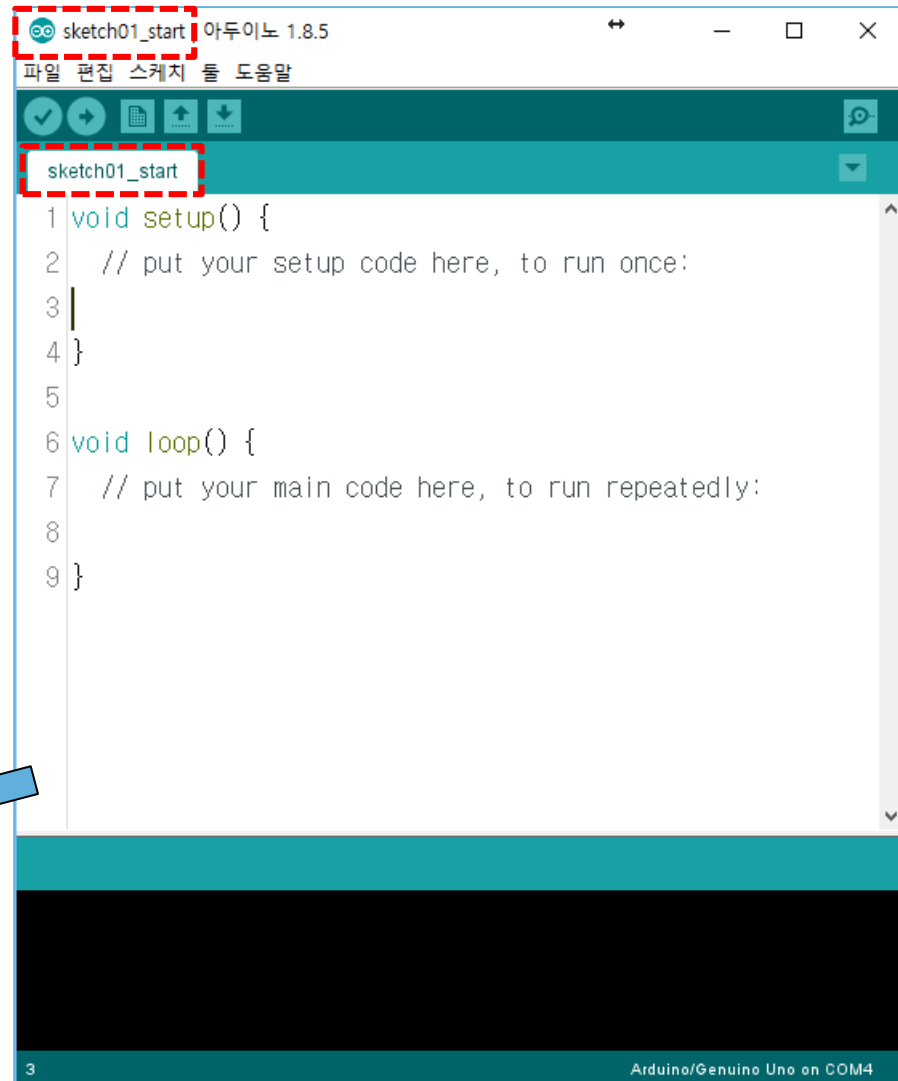
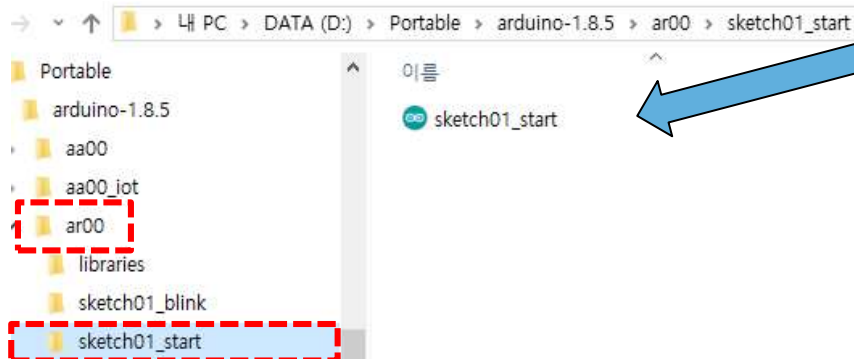
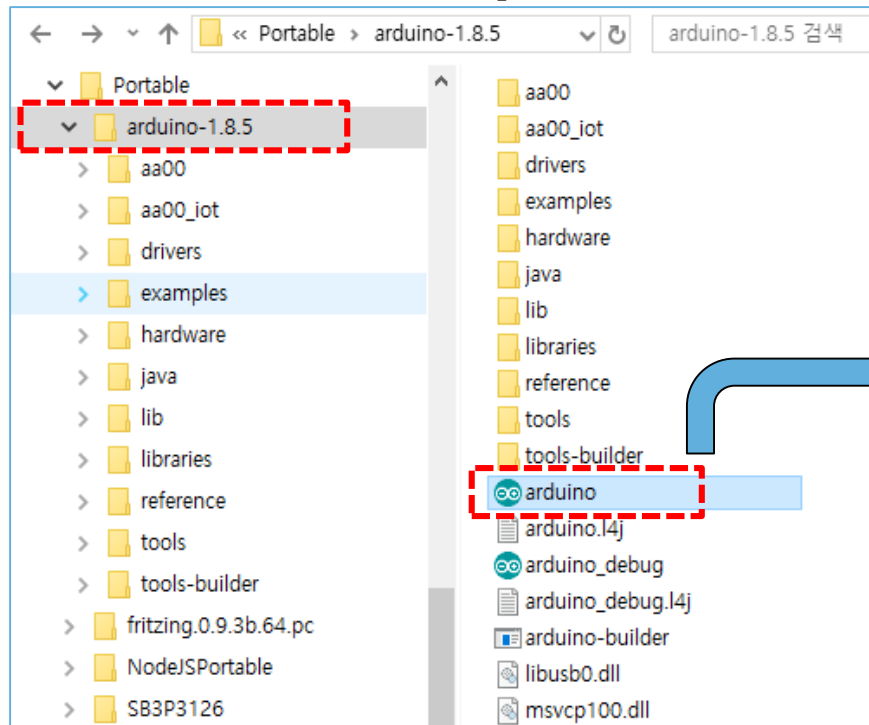
<https://www.arduino.cc/>





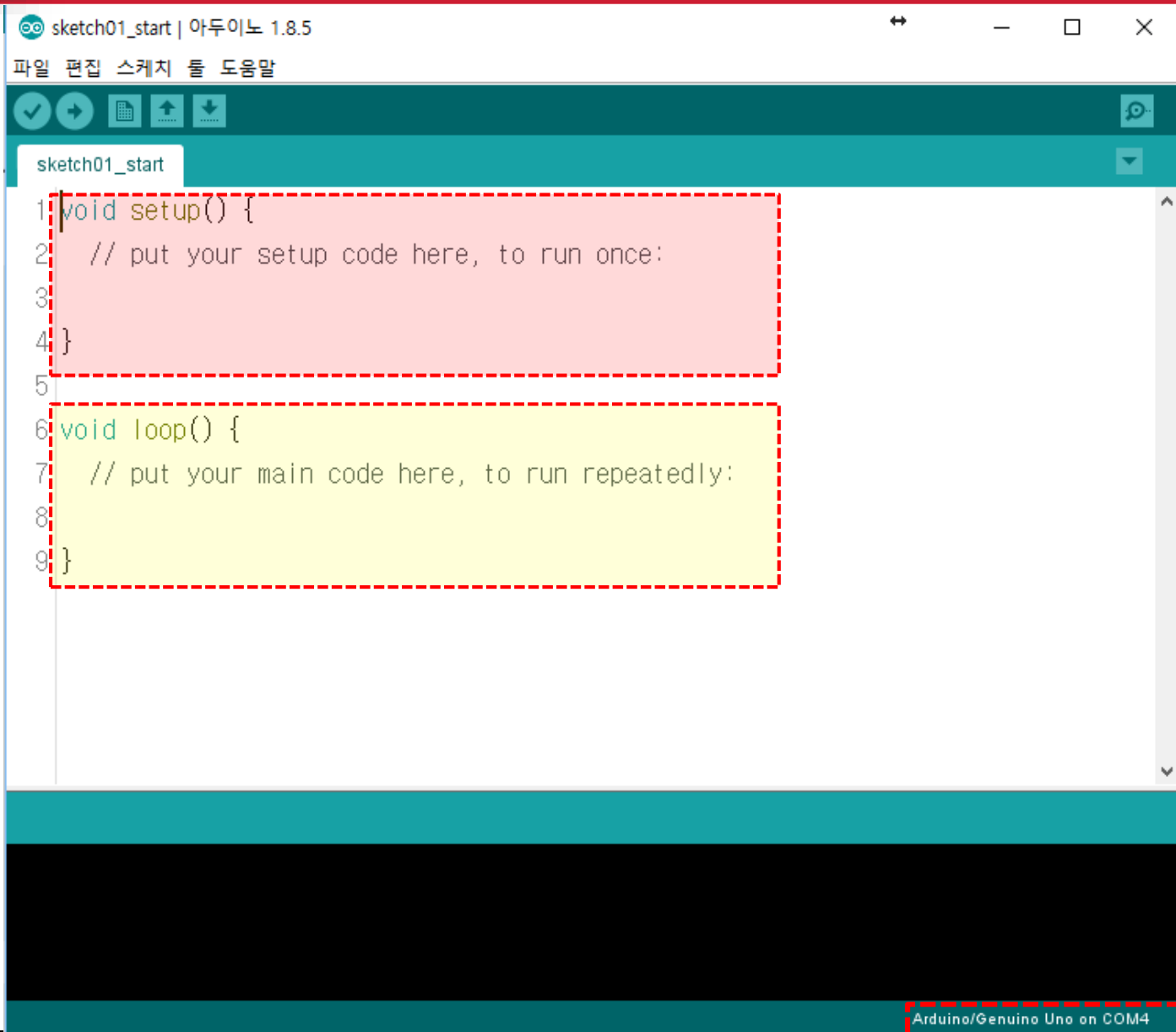
# A1.2 Arduino Portable (V1.8.5~9)

## Make folder **ar00** in portable Arduino folder





# A1.3 Arduino Portable IDE







# A1.4 Arduino Portable IDE

sketch01\_start | 아두이노 1.8.5

파일 편집 스케치 툴 도움말

새 파일	Ctrl+N
열기...	Ctrl+O
최근 파일 열기	>
스케치북	>
예제	>
닫기	Ctrl+W
저장	Ctrl+S
다른 이름으로 저장...	Ctrl+Shift+S
페이지 설정	Ctrl+Shift+P
인쇄	Ctrl+P
환경설정	Ctrl+Comma
종료	Ctrl+Q

환경설정

설정 네트워크

스케치북 위치:  
D:\WPortableWarduino-1.8.5\war00Wsketch01\_start 찾아보기

에디터 언어: 시스템 기본설정 (아두이노를 재시작해야 함)

에디터 글꼴 크기: 20

Interface scale: ☒ 자동 100% (아두이노를 재시작해야 함)

다음 동작중 자세한 출력 보이기: ☐ 컴파일 ☐ 업로드

컴파일러 경고: None

☒ 줄 번호 표시

☐ 코드 폴딩 사용하기

☒ 업로드 후 코드 확인하기

☐ 외부 에디터 사용

☒ Aggressively cache compiled core

☒ 시작시 업데이트 확인

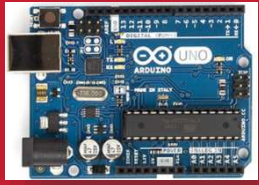
☒ 스케치 파일을 저장할때 새로운 확장자(.pde -> .ino)로 업데이트

☒ 검증 또는 업로드 할 때 저장하기

추가적인 보드 매니저 URLs [http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)

추가적인 환경 설정은 파일에서 직접 편집할 수 있습니다  
C:\WUsers\Wylsh-HCIt\WAppData\WLocal\WArduino15\Wpreferences.txt  
(아두이노가 실행되지 않는 경우에만 수정 가능)

확인 취소



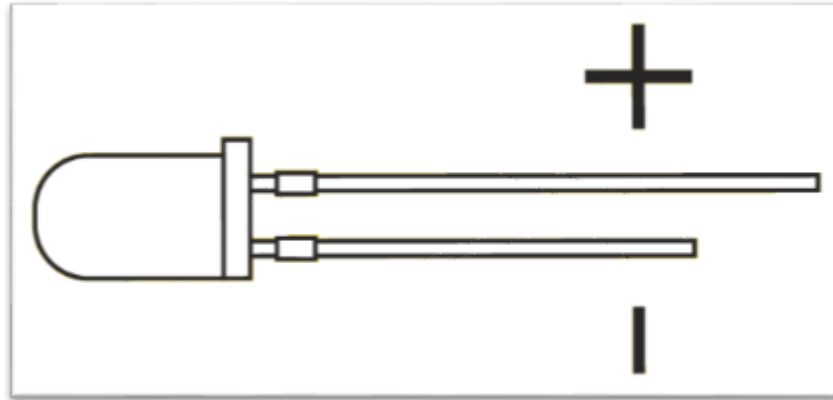
Hello



# Blink

## a LED

## Polarity of LED



**Find the longer leg, which should indicate the positive, anode pin.**

<https://learn.sparkfun.com/tutorials/polarity/diode-and-led-polarity>



# Blink a LED!

🔄 Blink | 아두이노 1.8.6

파일 편집 스케치 툴 도움말

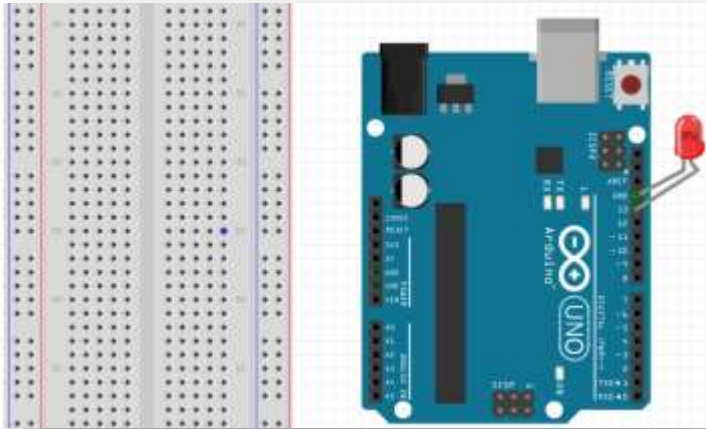
```
✓ → 📄 ⬆ ⬇
```

Blink \$

```
1 /*
2  Blink
3  Turns an LED on for one second, then off for one second, repeatedly.
4  */
5
6 // the setup function runs once when you press reset or power the board
7 void setup() {
8   // initialize digital pin LED_BUILTIN as an output.
9   pinMode(LED_BUILTIN, OUTPUT);
10 }
11
12 // the loop function runs over and over again forever
13 void loop() {
14   digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
15   delay(1000);                      // wait for a second
16   digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the voltage LOW
17   delay(1000);                      // wait for a second
18 }
```



# Blink a LED!



**+ : D13**  
**- : GND**

```
Blink$
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4  */
5
6
7 // the setup function runs once when you press reset or power the board
8 void setup() {
9   // initialize digital pin 13 as an output.
10  pinMode(13, OUTPUT);
11 }
12
13 // the loop function runs over and over again forever
14 void loop() {
15   digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
16   delay(1000);            // wait for a second
17   digitalWrite(13, LOW);  // turn the LED off by making the voltage LOW
18   delay(1000);            // wait for a second
19 }
```





# DIY-0. Blink a LED

## Change output pin to D12

```
Blink$
1 /*
2  Blink
3  Turns on an LED on for one second, then off for one second, repeatedly.
4  */
5
6
7 // the setup function runs once when you press reset or power the board
8 void setup() {
9   // initialize digital pin 13 as an output.
10  pinMode(13, OUTPUT);
11 }
12
13 // the loop function runs over and over again forever
14 void loop() {
15  digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
16  delay(1000);           // wait for a second
17  digitalWrite(13, LOW);  // turn the LED off by making the voltage LOW
18  delay(1000);           // wait for a second
19 }
```

**What do you do?**

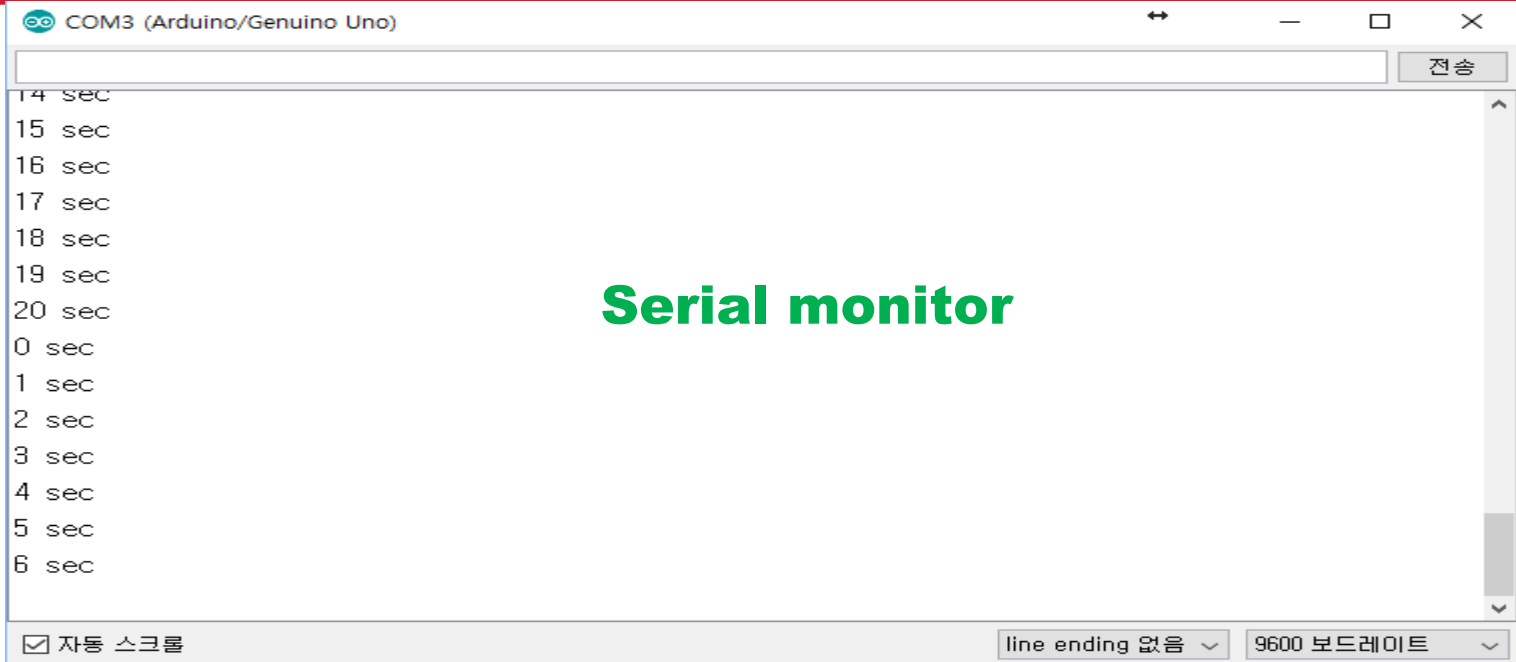
**Upload again!**

**Save ARnn\_blink.png**



# 2. Serial comm. monitor & plotter

## 2. Serial monitor & plotter



## 2. Serial comm.

### 시리얼 통신

- 2.1    Arduino에서 컴퓨터로 데이터 전송하기
- 2.2    변수 유형별로 컴퓨터에 전송하기
- 2.3    Arduino에서 시리얼 통신을 이용하여  
        데이터 수신하기 (next week)

# 2. 시리얼 통신 (Serial comm.)

## 시리얼 통신

### UART (Universal Asynchronous Receiver/Transmitter)

RS-232

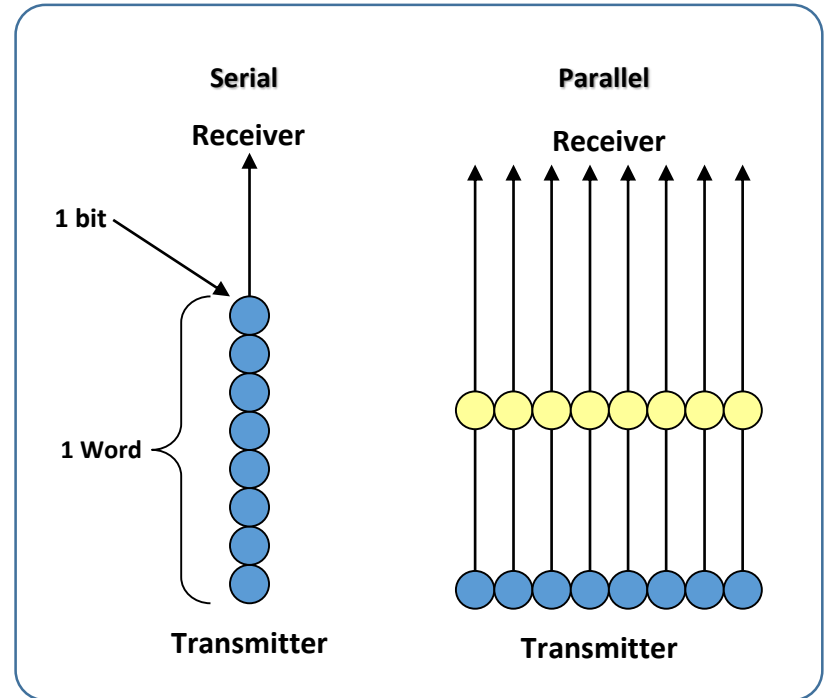
RS-422

RS-485

Arduino에서는 다음과 같은 목적으로 사용

Debugging : 프로그램의 오류를 수정하는 작업

데이터 통신 : Arduino와 컴퓨터 혹은 다른 장치와의 통신





# 2.1.1 Arduino에서 컴퓨터로 데이터 전송하기

EX 2.1

## Arduino에서 컴퓨터로 변수와 문자열 전송하기 (1/3)

- 실습목표**
1. Arduino에서 문자열과 데이터를 시리얼 통신을 이용하여 컴퓨터로 전송한다.
  2. 전송할 데이터는 0부터 1초 간격으로 1씩 증가하는 숫자와 'sec'라는 문자열이다.
  2. Arduino IDE의 시리얼 모니터에서 이를 확인해 본다.

**Hardware** Arduino와 PC를 USB 케이블로 연결한다.

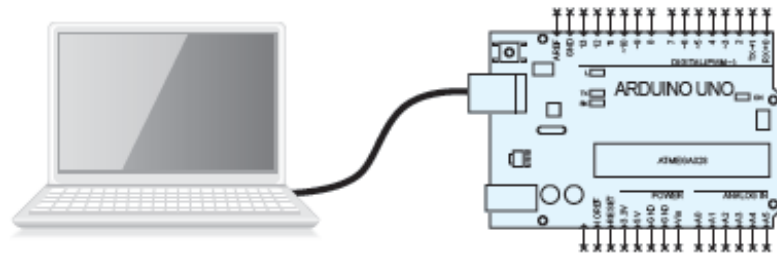


그림 2.1 Arduino와 PC와의 연결

## 2.1.2 Arduino에서 컴퓨터로 데이터 전송하기

EX 2.1

### Arduino에서 컴퓨터로 변수와 문자열 전송하기 (2/3)

Commands • `Serial.begin`(전송속도)

시리얼 통신 포트를 컴퓨터와 연결한다. 전송속도는 bps(bits per sec)로 일반적으로 9600으로 설정한다. 19200, 57600, 115200 등의 값을 설정할 수 있다.

• `Serial.print`(전송내용)

괄호 안의 내용을 시리얼 통신으로 전송한다. 따옴표로 구분된 부분은 텍스트를 직접 전송하고 따옴표 없이 변수를 써주면 변수의 값이 전송된다.

• `Serial.println`(전송내용)

‘Serial.print’와 같으나 전송 뒤 줄 바꿈을 한다.

• `delay`(지연시간 in ms)

지연시간에는 잠시 동작을 지연시키기 위한 값을 넣는다. 1/1000초 단위로 넣는다. 즉 1초를 지연시키기 위해선 1000의 값을 입력시킨다.

## 2.1.3 Arduino에서 컴퓨터로 데이터 전송하기

ex\_2\_1 | 아두이노 1.8.2

↔

—

□

×

파일 편집 스케치 툴 도움말

✓

→

📄

⬆

⬇

ex\_2\_1

```

1 /*
2  예제 2.1
3  Arduino에서 컴퓨터로 변수와 문자열 전송하기
4  */
5
6  int number = 0;           // -32768~32767 범위의 변수 number 설정, 초기값은 0
7
8  void setup() {
9    Serial.begin(9600);     // 9600bps로 시리얼 통신 설정
10
11
12  void loop() {
13    Serial.print(number);   // number 변수값 출력
14    Serial.println(" sec"); // " sec"를 출력 후 줄 바꿈
15    delay(1000);           // 1초동안 지연시킨다.
16    number++;              // number 변수값을 하나 증가시킨다.
17  }
18

```

업로드 완료.

스케치는 프로그램 저장 공간 1862 바이트(5%)를 사용, 최대 32256 바이트.

전역 변수는 동적 메모리 192바이트(9%)를 사용, 1856바이트의 지역변수가 남음. 최대는 2048 바이트.

COM3 (Arduino/Genuino Uno)

36 sec

37 sec

38 sec

39 sec

40 sec

41 sec

42 sec

43 sec

44 sec

45 sec

46 sec

47 sec

48 sec

49 sec

50 sec

51 sec

52 sec

53 sec

54 sec

55 sec

56 sec

57 sec

58 sec

59 sec

60 sec

☒ 자동 스크롤

8

Arduino/Genuino Uno on COM3

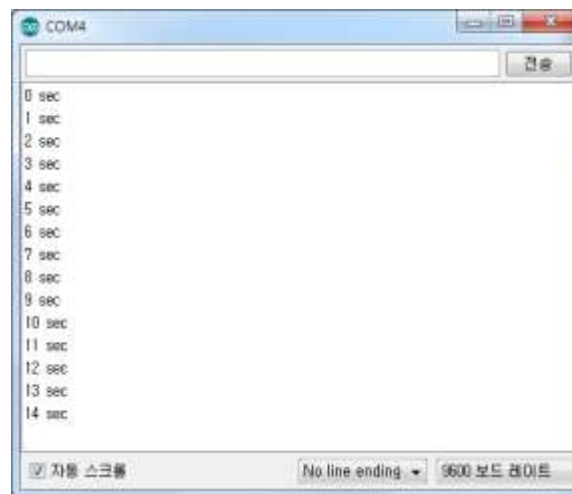
## EX 2.1

## Arduino에서 컴퓨터로 변수와 문자열 전송하기 (3/3)

- Sketch 구성**
1. 시리얼통신을 시작한다. 'Serial.begin()' 명령어로 할 수 있다.
  2. 변수를 전송하기 위해 'Serial.print()' 명령어를 사용하고,  
문자열 전송 후 줄 바꿈을 하기 위해서 'Serial.println()' 명령어를 사용한다.
  3. 루프를 1초마다 실행하기 위해서 'delay()' 명령어로 시간지연을 시켜준다.

**실행 결과** IDE의 시리얼 모니터를 실행시켜 Arduino에서 전송되는 메시지를 확인할 수 있다.

- 시리얼 플로터로 시각화



응용 문제 1. 2초, 5초 단위로 시간을 변경해 보자.

2. 자신만의 메시지를 출력해보자.

3. [DIY-1]

delay를 0.2초로 설정후

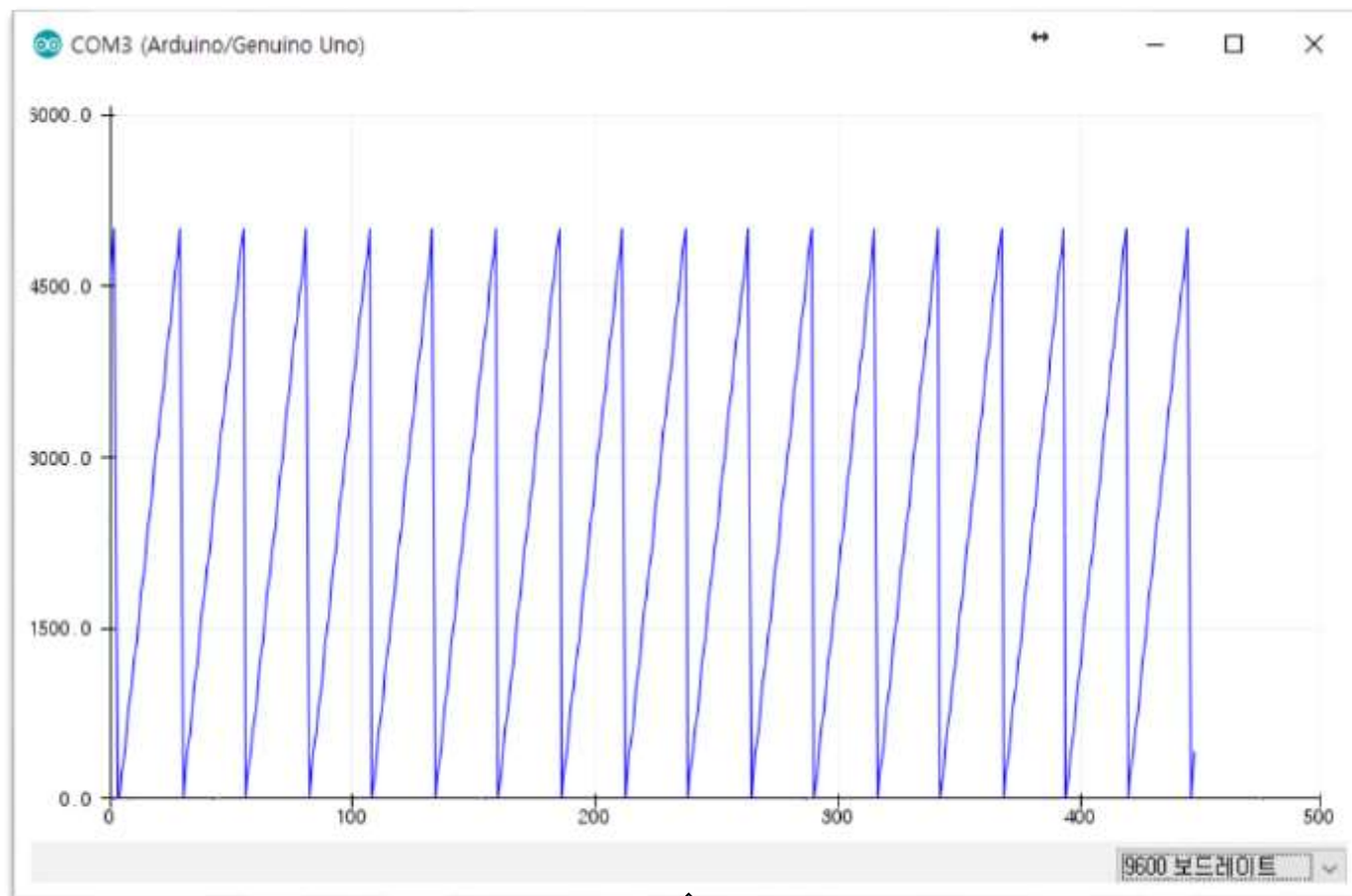
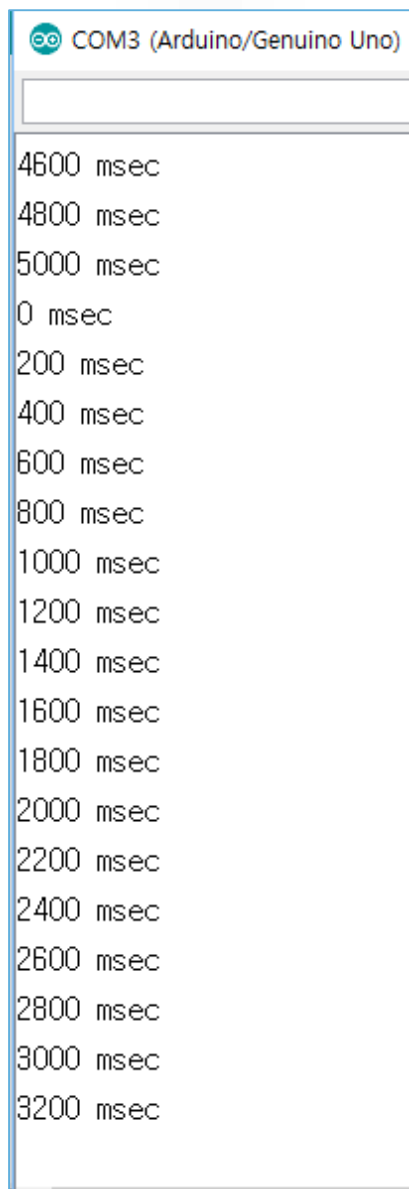
5초 마다 number를 초기화하여 (시간 초기화)

시리얼플로터로 톱니파를 발생.

시간은 ms로 계산해서 출력



# DIY-1. sawtooth signal



Save

ARnn\_sawtooth.png





# DIY-1. sawtooth signal : Code-1

```
hp00_diy2_sawtooth
1 /*
2  예제 2.1 : hp00_sawtooth
3  Arduino에서 컴퓨터로 변수와 문자열 전송하기
4  */
5
6 int number = 0;           // -32768~32767 범위의 변수 number 설정, 초기값은 0
7
8 void setup() {
9   Serial.begin(9600);     // 9600bps로 시리얼 통신 설정
10 }
11
12 void loop() {
13   Serial.print(number+200); // number 변수값 출력
14   Serial.println(" msec"); // " msec"를 출력 후 줄 바꿈
15   delay(200);              // 0.2초동안 지연시킨다.
16   number++;                // number 변수값을 하나 증가시킨다.
17
18   if (number > 25) {
19     number = 0;
20   }
21 }
```



# DIY-1. sawtooth signal : Code-2

```
hp00_diy2_sawtooth2
1 /*
2 예제 2.1 : hp00_sawtooth
3 Arduino에서 컴퓨터로 변수와 문자열 전송하기
4 */
5
6 int number = 0;           // -32768~32767 범위의 변수 number 설정, 초기값은 0
7
8 void setup() {
9   Serial.begin(9600);     // 9600bps로 시리얼 통신 설정
10 }
11
12 void loop() {
13   Serial.print(number);   // number 변수값 출력
14   Serial.println(" msec"); // " msec"를 출력 후 줄 바꿈
15   delay(200);             // 0.2초동안 지연시킨다.
16   //number++;             // number 변수값을 하나 증가시킨다.
17
18   if (number < 5000) {
19     number += 200;
20   }else {
21     number = 0;
22   }
23 }
```



# DIY-1. sawtooth signal : Code-3

```
hp00_diy2_sawtooth3
1 /*
2  예제 2.1 : hp00_sawtooth
3  Arduino에서 컴퓨터로 변수와 문자열 전송하기
4  */
5
6  int number = 0;           // -32768~32767 범위의 변수 number 설정. 초기값은 0
7
8  void setup() {
9      Serial.begin(9600);   // 9600bps로 시리얼 통신 설정
10 }
11
12 void loop() {
13     Serial.print(number);  // number 변수값 출력
14     Serial.println(" msec"); // " msec"를 출력 후 줄 바꿈
15     delay(200);           // 0.2초동안 지연시킨다.
16     number += 200;        // number 변수값을 200 증가시킨다.
17
18     if (number > 5000) {   // 5초(5000ms) 경과하면 number 초기화
19         number = 0;
20     }
21 }
```



## 2. 시리얼 통신 (Serial comm.)

### 2.2

### 변수 유형별로 컴퓨터에 전송하기

```
*** Hello Arduino ***
```

```
*** char Value ***
```

```
Binary:1000001
```

```
Decimal:65
```

```
Hexadecimal:41
```

```
ASCII:A
```

```
*** int Value ***
```

```
int Value:65
```

```
char(intValue):A
```

```
*** float Value ***
```

```
float Value:65.00
```

## 2.2.1 변수 유형별로 컴퓨터에 전송하기

- ✓ 사용 목적에 따라 다양한 변수 유형 중 선택하여 사용
- ✓ C 언어와 유사함

표 2.1 Arduino 변수 유형

변수 유형	바이트	변수 범위	용도
boolean	1	true(1) 혹은 false(0)	참(1) 아니면 거짓(0) 값을 나타냄.
char	1	-128~127 혹은 아스키 코드 값과 매칭되는 문자	부호가 있는 1바이트 숫자를 나타냄
unsigned char	1	0~255	0~255의 숫자를 나타낼 때 사용함
byte	1	0~255	unsigned char과 동일
int	2	-32,768~32,767	부호가 있는 2바이트 숫자를 나타냄
short	2	-32,768~32,767	int와 동일
unsigned int	2	0~65,535	부호가 없는 2바이트 숫자를 나타냄
word	2	0~65,535	unsigned int와 동일
long	4	-2,147,483,648 ~ 2,147,483,647	부호가 있는 4바이트 숫자를 나타냄
unsigned long	4	0~4,294,967,295	부호가 없는 4바이트 숫자를 나타냄
float	4	$-3.4028235 \times 10^{38}$ ~ $3.4028235 \times 10^{38}$	소수점 있는 숫자를 나타냄
double <sup>1)</sup>	4 or 8	$-3.4028235 \times 10^{38}$ ~ $3.4028235 \times 10^{38}$ Or $-1.79 \times 10^{308}$ ~ $1.79 \times 10^{308}$	소수점 있는 숫자를 나타냄. 4바이트의 경우 float과 동일 8바이트의 경우 변수 범위가 커짐
String	-	문자열에 따라 다름	문자열을 나타냄.

## 2.2.2 변수 유형별로 컴퓨터에 전송하기

✓ 실제 전송은 아스키코드 (ASCII Code)를 전송함

ASCII CODE TABLE

10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자	10	HEX	문자
0	0x00	NULL	22	0x16	STN	44	0x2C	,	66	0x42	B	88	0x58	X	110	0x6E	n
1	0x01	SOH	23	0x17	ETB	45	0x2D	-	67	0x43	C	89	0x59	Y	111	0x6F	o
2	0x02	STX	24	0x18	CAN	46	0x2E	.	68	0x44	D	90	0x5A	Z	112	0x70	p
3	0x03	ETX	25	0x19	EM	47	0x2F	/	69	0x45	E	91	0x5B	[	113	0x71	q
4	0x04	EOT	26	0x1A	SUB	48	0x30	0	70	0x46	F	92	0x5C	\	114	0x72	r
5	0x05	ENQ	27	0x1B	ESC	49	0x31	1	71	0x47	G	93	0x5D	]	115	0x73	s
6	0x06	ACK	28	0x1C	FS	50	0x32	2	72	0x48	H	94	0x5E	^	116	0x74	t
7	0x07	BEL	29	0x1D	GS	51	0x33	3	73	0x49	I	95	0x5F	_	117	0x75	u
8	0x08	BS	30	0x1E	RS	52	0x34	4	74	0x4A	J	96	0x60	`	118	0x76	v
9	0x09	HT	31	0x1F	US	53	0x35	5	75	0x4B	K	97	0x61	a	119	0x77	w
10	0x0A	LF	32	0x20	SP	54	0x36	6	76	0x4C	L	98	0x62	b	120	0x78	x
11	0x0B	VT	33	0x21	!	55	0x37	7	77	0x4D	M	99	0x63	c	121	0x79	y
12	0x0C	FF	34	0x22	"	56	0x38	8	78	0x4E	N	100	0x64	d	122	0x7A	z
13	0x0D	CR	35	0x23	#	57	0x39	9	79	0x4F	O	101	0x65	e	123	0x7B	{
14	0x0E	SO	36	0x24	\$	58	0x3A	:	80	0x50	P	102	0x66	f	124	0x7C	
15	0x0F	SI	37	0x25	%	59	0x3B	;	81	0x51	Q	103	0x67	g	125	0x7D	}
16	0x10	DEL	38	0x26	&	60	0x3C	<	82	0x52	R	104	0x68	h	126	0x7E	~
17	0x11	DC1	39	0x27	'	61	0x3D	=	83	0x53	S	105	0x69	i	127	0x7F	DEL
18	0x12	DC2	40	0x28	(	62	0x3E	>	84	0x54	T	106	0x6A	j			
19	0x13	DC3	41	0x29	)	63	0x3F	?	85	0x55	U	107	0x6B	k			
20	0x14	DC4	42	0x2A	*	64	0x40	@	86	0x56	V	108	0x6C	l			
21	0x15	NAK	43	0x2B	+	65	0x41	A	87	0x57	W	109	0x6D	m			

그림 2.2 ASCII 코드표





## 2.2.3 변수 유형별로 컴퓨터에 전송하기

EX 2.2

### 변수 유형별 Arduino에서 컴퓨터로 전송하기 (1/4)

- 실습목표**
1. Arduino에서 컴퓨터로 데이터를 전송할 때 변수 유형별로 출력한다.
  2. char로 선언된 변수, int로 선언된 변수, float로 선언된 변수를 'Serial.print' 명령어를 이용하여 PC로 전송하자.
  3. 'Serial.print' 명령어의 출력 옵션을 변경하여 전송해 보자.
  4. 'Serial.write' 명령어로 문자열을 출력해 보자.
  5. 각 변수 유형별 출력되는 차이를 비교해 보자.

**Hardware** Arduino와 PC를 USB 케이블로 연결한다.

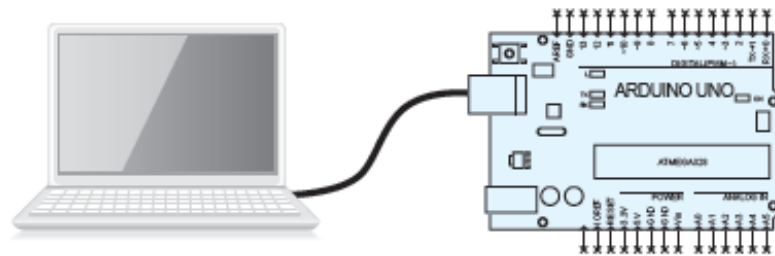


그림 2.1 Arduino와 PC와의 연결



## 2.2.4 변수 유형별로 컴퓨터에 전송하기

EX 2.2

### 변수 유형별 Arduino에서 컴퓨터로 전송하기 (2/4)

- Commands
- `Serial.write(char 변수);`  
char변수에 해당하는 **ASCII 코드값의 문자를 출력**한다.
  - `Serial.print(변수, BIN);`  
변수를 2진수(Binary)로 표시한다.
  - `Serial.print(변수, DEC);`  
변수를 10진수(Binary)로 표시한다.
  - `Serial.print(변수, HEX);`  
정해진 변수를 16진수(Hexadecimal)로 표시한다.



## 2.2.4 변수 유형별로 컴퓨터에 전송하기

### EX 2.2 Serial write & print

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

void loop(){
  const uint8_t temp[5] = {'1', '2', '3', '4', '5'};
  Serial.write(1); // write()함수로 1 전송
  delay(500);
  Serial.write(49); // write()함수로 49 전송
  delay(500);
  Serial.print(1); // print()함수로 1 전송
  delay(500);
  Serial.write('a'); // write()함수로 'a' 전송
  delay(500);
  Serial.write(temp, 5); // write()함수로 temp 배열을 5만큼 전송
  delay(500);
  Serial.write("12345"); // write()함수로 string 값 전송
  delay(500);
  Serial.write("\n"); // 줄바꿈
  delay(500);
}
```

COM10 (Arduino/Genuino Uno)

```
5
11a1234512345
11a1234512345
11a1234512345
11a1234512345
11a1234512345
11a1234512345
11a1234512345
```

**Serial.write()** 함수는  
**Serial.print()** 함수와 마찬가지로  
데이터 값을 시리얼 통신으로 송신하는  
기능을 합니다.

위의 소스코드 결과 화면을 보면  
**write()** 함수와 **print()** 함수가 모두  
1 을 전송하는데 (첫번째와 세번째)  
**write()** 함수는 이상한 값을  
출력합니다. **write(49)**를 출력해야  
제대로 값이 1이 나오는 것을 확인 할  
수 있습니다.



## 2.2.5 변수 유형별로 컴퓨터에 전송하기

EX 2.2

### 변수 유형별 Arduino에서 컴퓨터로 전송하기 (3/4)

- Sketch 구성
1. '65'란 숫자를 char형, int형, float형 변수에 각각 저장한다.
  2. 'Binary:', 'Decimal:', 'Hexadecimal:', 'ASCII:' 등 네 가지 문자열을 저장하여 호출하여 사용한다.
  3. 변수에 저장된 숫자를 2진수형, 10진수형, 16진수형, ASCII 코드형 등 Serial.print 명령어의 옵션을 변경하여 전송한다.
  4. 옵션이 변경될 때마다 문자열을 호출하여 함께 출력한다.
  5. loop가 반복될 때마다 숫자를 1씩 증가시킨다. float형은 0.1씩 증가시킨다.



## 2.2.6 변수 유형별로 컴퓨터에 전송하기

EX 2.2

### 변수 유형별 Arduino에서 컴퓨터로 전송하기 (4/4)

**실습 결과** IDE의 시리얼 모니터를 실행시켜 Arduino에서 전송되는 메시지를 확인할 수 있다.  
10초 간격으로 증가된 값에 대하여 출력하게 된다. 이때 2진수, 10 진수, 16진수로 표시되고, 증가된 숫자에 해당하는 아스키코드의 문자가 전송된다. 'float'형 변수는 소수점이 함께 표시된다.

```
*** char Value ***  
Binary:1000001  
Decimal:65  
Hexadecimal:41  
ASCII:A  
  
*** int Value ***  
*** float Value ***  
float Value:65.00
```

**Quiz** 1. `Serial.write(floatValue);`  
위의 명령이 오류가 나는 이유는?



# code

ex\_2\_2\_start.ino \$

```
1 /*
2  예제 2.2
3  변수 유형별 Arduino에서 컴퓨터로 전송하기
4  */
5
6 // 65란 숫자를 유형별 변수에 저장한다.
7 char   charValue = 65;
8 int     intValue = 65;
9 float  floatValue = 65.0;
10
11 // 문자열 네가지를 설정한다.
12 String stringValue[]={ "Binary:", "Decimal:", "Hexadecimal:", "ASCII:" }; //
13
14 void setup() {
15     // 9600bps로 시리얼 통신 설정
16     Serial.begin(9600);
17 }
18
19 void loop() {
20
21     Serial.println("*** Hello Arduino by ARnn ***");
22     Serial.println();
23
24     // 'char Value'를 출력하고 문자열과 숫자를 변수 유형별로 출력한다.
25     Serial.println("*** char Value ***");
26     Serial.print(stringValue[0]); // stringValue 중 첫 번째 문자열 출력
27     Serial.println(charValue, BIN); // 2진수 형태로 출력
28     Serial.print(stringValue[1]); // stringValue 중 두 번째 문자열 출력
29     Serial.println(charValue, DEC); // 10진수 형태로 출력
30     Serial.print(stringValue[2]); // stringValue 중 세 번째 문자열 출력
31     Serial.println(charValue, HEX); // 16진수 형태로 출력
32     Serial.print(stringValue[3]); // stringValue 중 네 번째 문자열 출력
33     Serial.write(charValue); // charValue에 해당하는 ASCII 코드값 출력
```

```
*** char Value ***
Binary:1000001
Decimal:65
Hexadecimal:41
ASCII:A
```

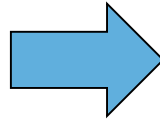
```
*** int Value ***
*** float Value ***
float Value:65.00
```

```
34 // 줄바꿈
35 Serial.println();
36 // 줄바꿈
37 Serial.println();
38
39 // 'int Value'를 출력하고 문자열과 숫자를 변수 유형별로 출력한다.
40 Serial.println("*** int Value ***");
41
42 // 'float Value'를 출력하고 문자열과 숫자를 변수 유형별로 출력한다.
43 Serial.println("*** float Value ***");
44 Serial.print("float Value:");
45 Serial.println(floatValue);
46 //Serial.write(floatValue);
47
48 Serial.println();
49
50 charValue++; // charValue 1 증가
51 intValue++; // intValue 1 증가
52 floatValue += 0.1; // floatValue 0.1 증가
53
54 delay(10000); // 10초동안 지연시킨다.
55 }
```



## 2.2.7 final result

```
*** char Value ***  
Binary:1000001  
Decimal:65  
Hexadecimal:41  
ASCII:A  
  
*** int Value ***  
*** float Value ***  
float Value:65.00
```



**\*\*\* Hello Arduino by ARnn \*\*\***

```
*** char Value ***  
Binary:1000001  
Decimal:65  
Hexadecimal:41  
ASCII:A  
  
*** int Value ***  
int Value:65  
char(intValue):A  
  
*** float Value ***  
float Value:65.00
```





## DIY-2. Escape from loop()

응용 문제 [DIY-2] 0~15까지 10진수를 2진수와 16진수로 출력하는 스케치를 작성해보자

```
COM3 (Arduino/Genuino Uno)

Number = 0, Binary:0, Hexadecimal:0
Number = 1, Binary:1, Hexadecimal:1
Number = 2, Binary:10, Hexadecimal:2
Number = 3, Binary:11, Hexadecimal:3
Number = 4, Binary:100, Hexadecimal:4
Number = 5, Binary:101, Hexadecimal:5
Number = 6, Binary:110, Hexadecimal:6
Number = 7, Binary:111, Hexadecimal:7
Number = 8, Binary:1000, Hexadecimal:8
Number = 9, Binary:1001, Hexadecimal:9
Number = 10, Binary:1010, Hexadecimal:A
Number = 11, Binary:1011, Hexadecimal:B
Number = 12, Binary:1100, Hexadecimal:C
Number = 13, Binary:1101, Hexadecimal:D
Number = 14, Binary:1110, Hexadecimal:E
Number = 15, Binary:1111, Hexadecimal:F
Mission completed!
```

[Hint]

1. `int number = 0; // starting number`
2. `loop()`에서 1초 간격으로 `number`를 1씩 증가
3. 옆의 방식으로 결과 출력
4. `number`가 15를 초과하면 `loop()` 탈출  
`exit(0); // loop 탈출 함수`

ARnn\_loop\_escape.png



## DIY-2. Escape from loop()

응용 문제 [DIY-2 - hint] 0~15까지 10진수를 2진수와 16진수로 출력하는 스케치를 작성해보자

```
1 /*  
2  DIY-2  
3 */  
4  
5 // start number  
6 int number = 0;  
7  
8 // 문자열 두가지를 설정한다.  
9 String stringValue[]={"Binary:", "Hexadecimal:"}; // array  
10  
11 void setup() {  
12   // 9600bps로 시리얼 통신 설정  
13   Serial.begin(9600);  
14 }
```

```
16 void loop() {  
17  
18   // 'char Value'를 출력하고 문자열과 숫자를 변수 유형별로 출력한다.  
19   Serial.print("Number = ");  
20   Serial.print(number);  
21   Serial.print(", ");  
22   Serial.print(stringValue[0]); // stringValue 중 첫 번째 문자열 출력  
23   Serial.print(number, BIN); // 2진수 형태로 출력  
24   Serial.print(", ");  
25   Serial.print(stringValue[1]); // stringValue 중 첫 번째 문자열 출력  
26   Serial.print(number, HEX); // 16진수 형태로 출력  
27   // 줄바꿈  
28   Serial.println();  
29  
30   number++; // number 1 증가  
31  
32  
33  
34  
35  
36  
37  
38   delay(1000); // 1초동안 지연시킨다.  
39 }
```

your code !!!



## DIY-3. sum from 1 to 100

응용 문제 [DIY-3] 1에서 100까지 정수의 합을 계산하는 스케치를 작성해보자

```
COM10 (Arduino/Genuino Uno)

Number = 86, Sum = 3741
Number = 87, Sum = 3828
Number = 88, Sum = 3916
Number = 89, Sum = 4005
Number = 90, Sum = 4095
Number = 91, Sum = 4186
Number = 92, Sum = 4278
Number = 93, Sum = 4371
Number = 94, Sum = 4465
Number = 95, Sum = 4560
Number = 96, Sum = 4656
Number = 97, Sum = 4753
Number = 98, Sum = 4851
Number = 99, Sum = 4950
Number = 100, Sum = 5050

ARnn: 1 + 2 + ... + 100 =5050
```

[Hint]

1. `int number = 0; // starting number`
  2. `int sum = 0;`
  3. `loop()`에서 **0.1초 간격**으로 `number`를 1씩 증가시키면서 `sum`에 합한다.
- ✓ 옆의 방식으로 결과 출력
  - ✓ `startNum` 가 100을 초과하면 `loop()` 탈출  
`exit(0); // loop 탈출 함수`



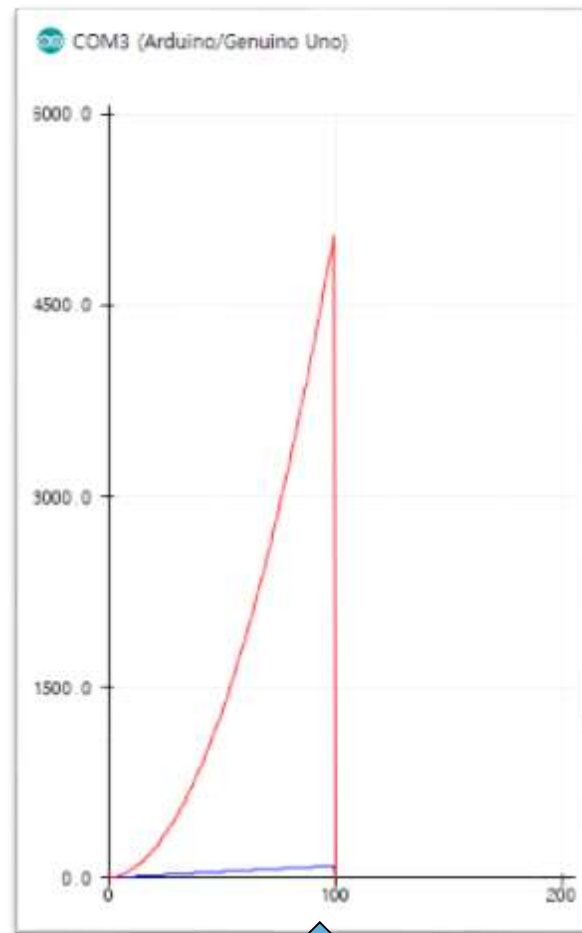
## DIY-3. sum from 1 to 100

응용 문제 [DIY-3] Results on serial monitor and plotter

COM10 (Arduino/Genuino Uno)

Number = 86,	Sum = 3741
Number = 87,	Sum = 3828
Number = 88,	Sum = 3916
Number = 89,	Sum = 4005
Number = 90,	Sum = 4095
Number = 91,	Sum = 4186
Number = 92,	Sum = 4278
Number = 93,	Sum = 4371
Number = 94,	Sum = 4465
Number = 95,	Sum = 4560
Number = 96,	Sum = 4656
Number = 97,	Sum = 4753
Number = 98,	Sum = 4851
Number = 99,	Sum = 4950
Number = 100,	Sum = 5050

ARnn:  $1 + 2 + \dots + 100 = 5050$



ARnn\_sum100.png



# [Practice]

## ◆ [wk02]

- **Serial comm.**
- **Complete your project**
- **upload folder: Arnn\_Rpt01**  
**in github.com repo “arnn”**

## ◆ [Target of this week]

- Complete your works
- Save your outcomes and upload 4 figures in [github.com](https://github.com)

제출폴더명 : **ARnn\_Rpt01**

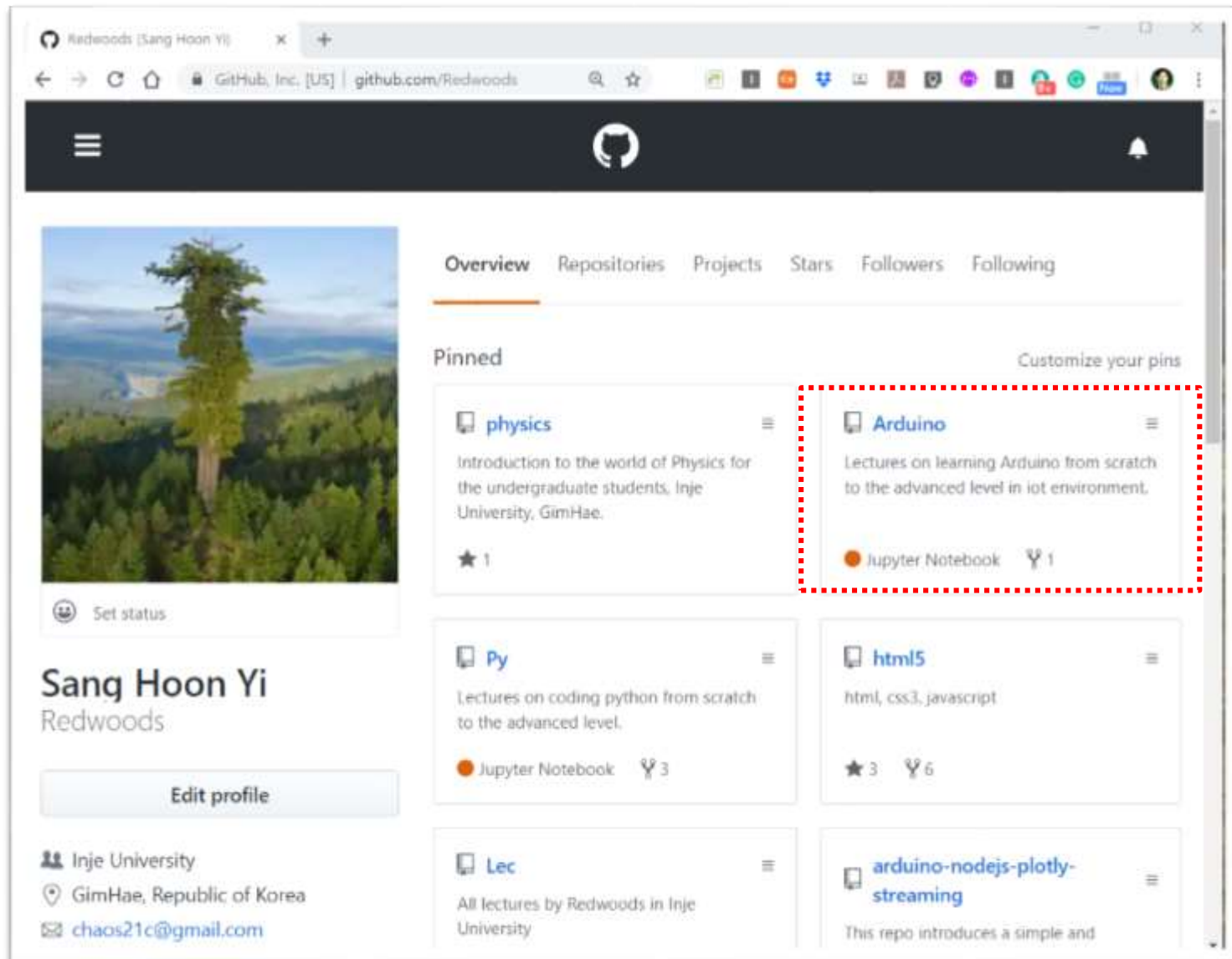
### - 제출할 파일들

- ① **ARnn\_blink.png**
- ② **ARnn\_sawtooth.png**
- ③ **ARnn\_loop\_escape.png**
- ④ **ARnn\_sum100.png**

## ● References & good sites

- ✓ <http://www.nodejs.org/ko> Node.js
- ✓ <http://www.arduino.cc> Arduino Homepage
- ✓ <http://www.w3schools.com> By w3schools
- ✓ <http://www.github.com> GitHub
- ✓ <http://www.google.com> Googling





Redwoods (Sang Hoon Yi)

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

Overview Repositories Projects Stars Followers Following

Pinned Customize your pins

**physics**  
Introduction to the world of Physics for the undergraduate students, Inje University, GimHae.  
★ 1

**Arduino**  
Lectures on learning Arduino from scratch to the advanced level in iot environment.  
Jupyter Notebook 🍴 1

**Py**  
Lectures on coding python from scratch to the advanced level.  
Jupyter Notebook 🍴 3

**html5**  
html, css3, javascript  
★ 3 🍴 6

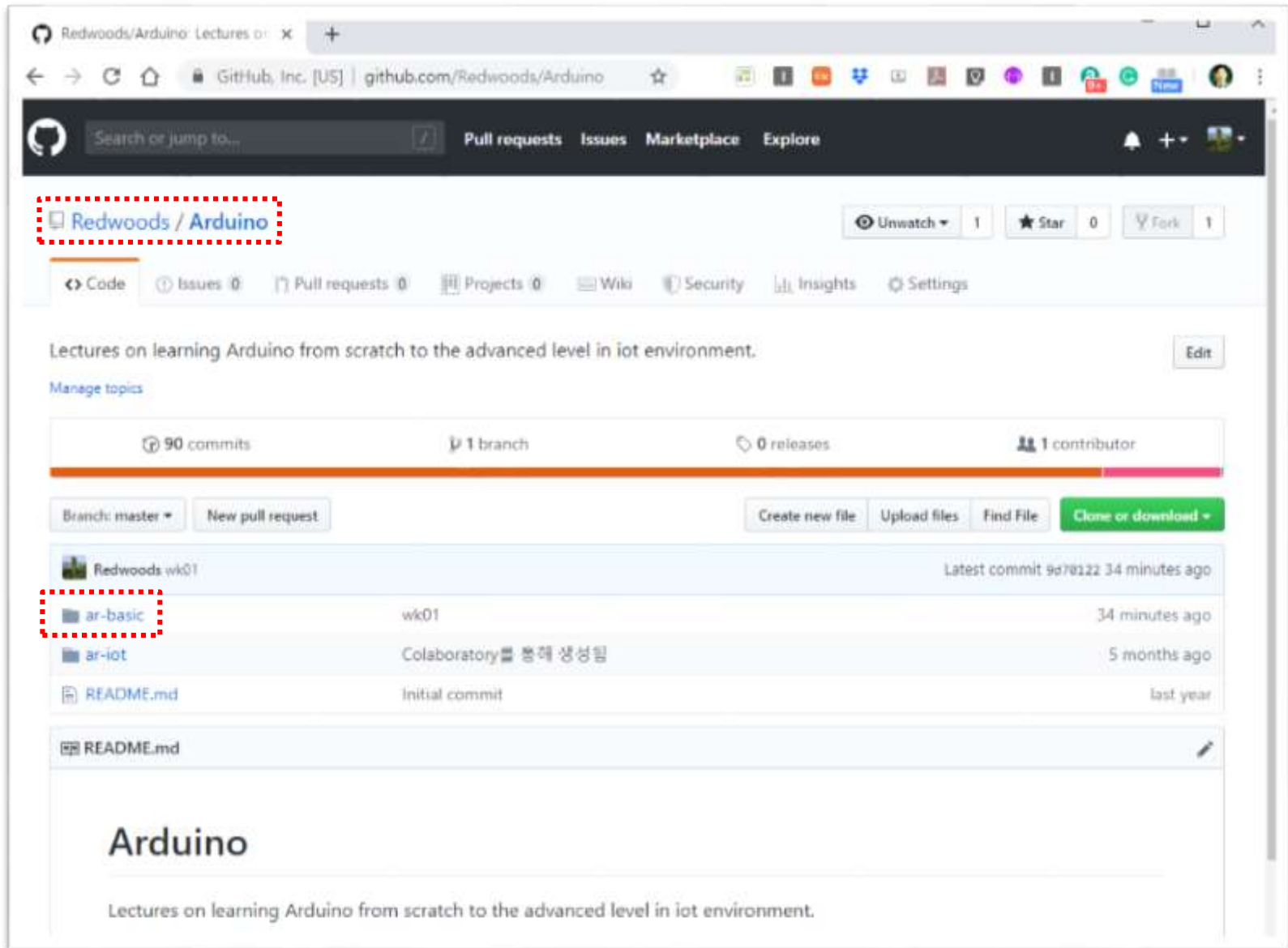
**Lec**  
All lectures by Redwoods in Inje University

**arduino-nodejs-plotly-streaming**  
This repo introduces a simple and

**Sang Hoon Yi**  
Redwoods

Edit profile

Inje University  
GimHae, Republic of Korea  
chaos21c@gmail.com



The screenshot shows the GitHub repository page for **Redwoods/Arduino**. The repository description is "Lectures on learning Arduino from scratch to the advanced level in iot environment." The repository has 90 commits, 1 branch, 0 releases, and 1 contributor. The file list includes **ar-basic** (34 minutes ago), **ar-iot** (5 months ago), and **README.md** (last year). The **ar-basic** file is highlighted with a red dashed box. The repository is currently on the **master** branch.

Redwoods / Arduino

Unwatch 1 Star 0 Fork 1

Code Issues 0 Pull requests 0 Projects 0 Wiki Security Insights Settings

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

Manage topics

90 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find File Clone or download

Redwoods (wk01)	Latest commit 9d70122 34 minutes ago
ar-basic	wk01 34 minutes ago
ar-iot	Colaboratory를 통해 생성됨 5 months ago
README.md	Initial commit last year

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 캡 (피랑,노랑,초록,빨강,하양)</b> 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 (빨강,노랑,초록,하양,파랑)</b> 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>가열기 센서 스위치</div> <div>센서의 가열기에 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 사용합니다.</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 움직임 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 인식 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 되어있습니다.</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 묶은 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>					