

반기제어 응용

0726_31_박 정욱

배울 내용

- 함수, 배열에 관련된 복습과 이해
- “아날로그 데이터” 의 이해
- LED의 개념과 심화 이해
- 함수, 회로구성 응용 심화

함수

- 함수는 특정한 작업을 처리하기 위한 프로그램(코드)의 블록이다.
- 프로그램에서 동일하게 반복되는 부분을 독립된 역할로 한번 만들어 두어, 필요할 때 마다 호출해서 사용 한다.
- 반복되는 작업들을 수정하면 모든 함수에 적용되기 때문에 유지보수에 유리하다.

함수의 종류

- 라이브러리 함수: 컴파일러를 만든 곳(시스템 내부에) 정의되어있는 함수로, 자세한 리스트는 “<https://www.arduino.cc/reference/en/>” 에서 확인할 수 있다.
 - Ex) `digitalRead()`, `pinMode()`, `delay()` ...
- 사용자 정의 함수: 사용자(개발자)가 직접 만들고 호출하는 함수이며, 보통의 “함수” 를 뜻한다.

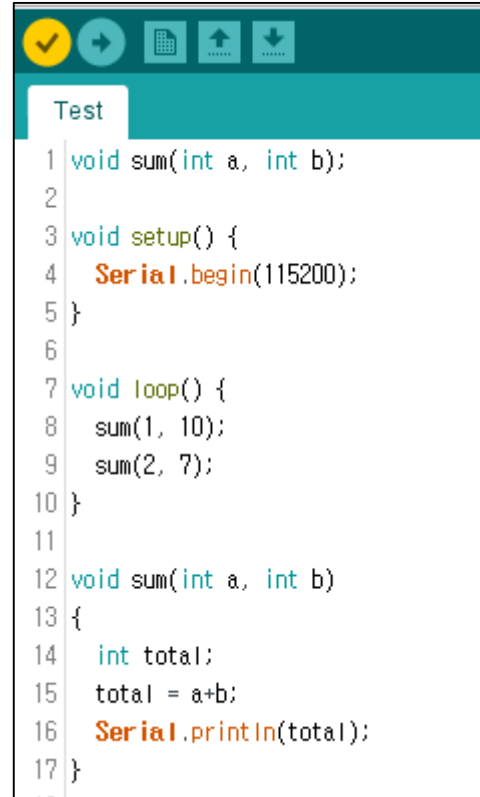
함수의 구조

자료형 함수명(매개변수) { 함수의 몸체 (데이터를 처리할 부분) }	<pre>int sum(int a, int b) { int total; total = a + b; return total; }</pre>
--	--

- 함수는 크게
 - 자료형 (void, int ...)
 - 함수 이름
 - 매개변수
 - 함수 내용
- 으로 나뉜다.

함수 실습 기초(1)

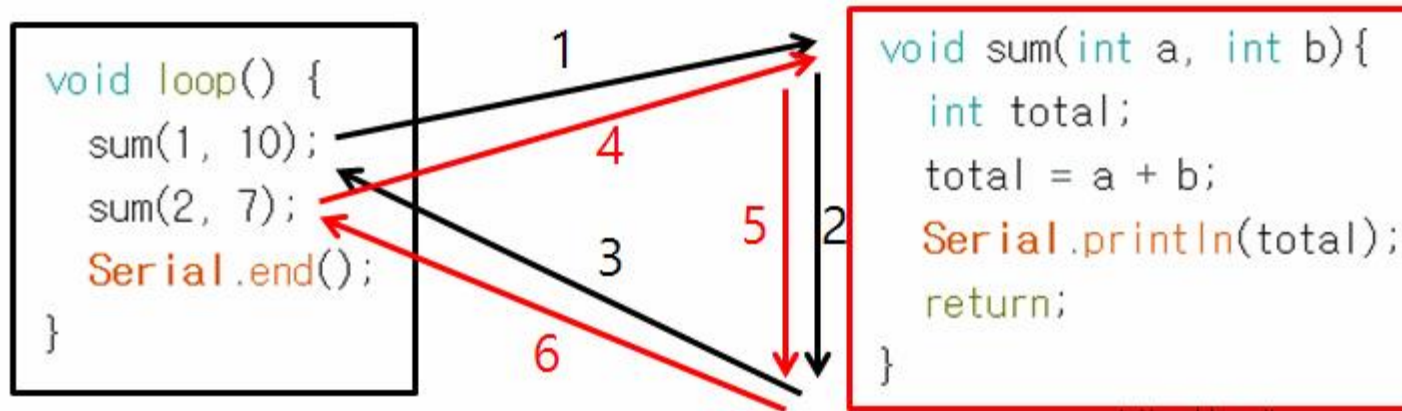
- Source



```
Test
1 void sum(int a, int b);
2
3 void setup() {
4   Serial.begin(115200);
5 }
6
7 void loop() {
8   sum(1, 10);
9   sum(2, 7);
10 }
11
12 void sum(int a, int b)
13 {
14   int total;
15   total = a+b;
16   Serial.println(total);
17 }
```

함수 실습 기초(1)

■ 해석



함수의 유형(생김새)

- 함수는 자료형(리턴값)과 매개변수에 따라 네가지로 나뉜다.
 - 매개변수 x, 리턴 값 x
 - 매개변수 o, 리턴 값 x
 - 매개변수 x, 리턴 값 o
 - 매개변수 o, 리턴 값 o

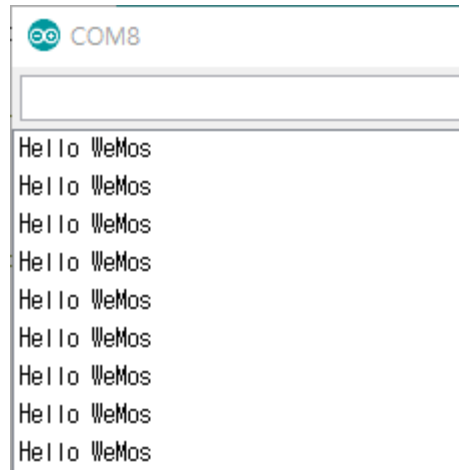
함수 실습 기초(2)

- Source
- 첫번째로, 매개변수와 반환 값(리턴 값)이 모두 없는 경우 이다.

```
Basis_function_2
1 void Text(int a, int b);
2
3 void setup() {
4   Serial.begin(115200);
5 }
6
7 void loop() {
8   Text();
9   Text();
10  Text();
11 }
12
13 void Text()
14 {
15   Serial.println("Hello WeMos");
16 }
```

함수 실습 기초(2)

- 결과



함수 실습 기초(3)

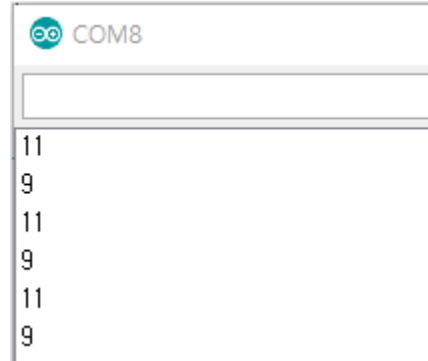
- Source
- 두번째로 매개변수는 있고, 반환 값(리턴 값)이 없는 경우 이다.

Basis_function_3

```
1 void sum(int a, int b);  
2  
3 void setup() {  
4     Serial.begin(115200);  
5 }  
6  
7 void loop() {  
8     sum(1, 10);  
9     sum(2, 7);  
10  
11 }  
12  
13 void sum(int a, int b)  
14 {  
15     int total;  
16     total = a + b;  
17     Serial.println(total);  
18 }
```

함수 실습 기초(3)

- 결과



```
COM8
11
9
11
9
11
9
```

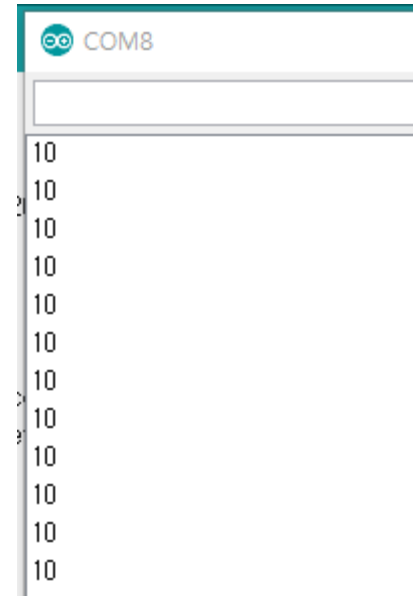
함수 실습 기초(4)

- Source
- 매개변수가 없고 반환 값은 있는 경우.

```
Basis_function_4
1 int func4();
2
3 void setup() {
4   Serial.begin(115200);
5 }
6
7 void loop() {
8   int returnValue;
9   returnValue = func4();
10  Serial.println(returnValue);
11
12  Serial.end();
13 }
14
15 int func4()
16 {
17   int i = 10;
18   return i;
19 }
```

함수 실습 기초(4)

- 결과



```
COM8
echo 10
10
echo 10
10
echo 10
10
echo 10
10
echo 10
10
echo 10
10
echo 10
10
echo 10
10
```

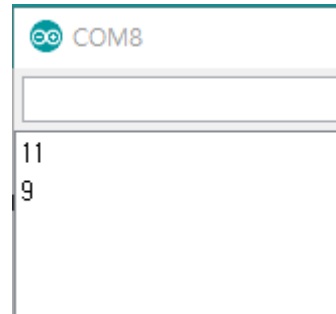
함수 실습 기초(5)

- Source
- 매개변수와 반환 값이 모두 있는 유형.

```
Basis_function_5
1 int func5(int a, int b);
2
3 void setup() {
4   Serial.begin(115200);
5 }
6
7 void loop() {
8   int total;
9   total = func5(1, 10);
10  Serial.println(total);
11  total = func5(2, 7);
12  Serial.println(total);
13
14  delay(2000);
15 }
16
17 int func5(int a, int b)
18 {
19   int t;
20   t = a + b;
21   return t;
22 }
23
```

함수 실습 기초(5)

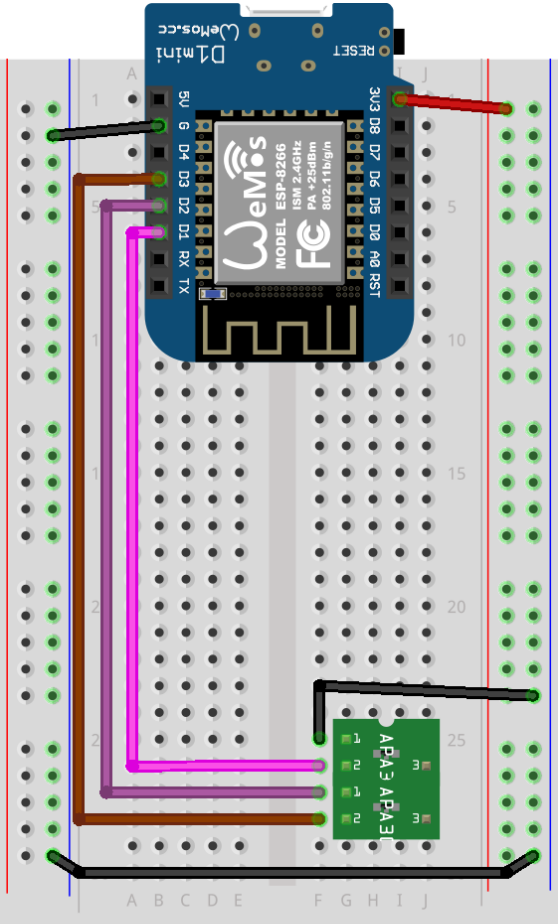
- 결과



A screenshot of a terminal window titled "COM8". The window has a white background and a thin green border. Inside, there is a white rectangular input field. Below the input field, the numbers "11" and "9" are displayed on two separate lines. A vertical cursor line is positioned to the left of the numbers.

```
COM8
[Input Field]
11
9
```


LED Array를 사용한 제어 응용



fritzing

- 회로도
- 키트에 있는 LED Array(3색 LED)를 사용하여 제작
- 첫번째 핀에 GND, 각각 나머지에 D1, D2, D3를 연결

LED Array를 사용한 제어

■ Source

```
1 const int PIN_LED_GREEN = D1;
2 const int PIN_LED_YELLOW = D2;
3 const int PIN_LED_RED = D3;
4 int i;
5
6 // 프로그램 시작시 초기화 작업
7 void setup()
8 {
9     Serial.begin(115200);    // 시리얼 통신 초기화
10    pinMode(PIN_LED_GREEN, OUTPUT);
11    pinMode(PIN_LED_YELLOW, OUTPUT);
12    pinMode(PIN_LED_RED, OUTPUT);
13
14    i = 0;
15 }
16
17 void loop() {
18     i++;
19     if ( i == 1 )
20     {
21         digitalWrite(PIN_LED_GREEN, HIGH);
22         digitalWrite(PIN_LED_YELLOW, LOW);
23         digitalWrite(PIN_LED_RED, LOW);
24         delay(2000);
25     } else if ( i == 2 ) {
26         digitalWrite(PIN_LED_GREEN, LOW);
27         digitalWrite(PIN_LED_YELLOW, HIGH);
28         digitalWrite(PIN_LED_RED, LOW);
29         delay(2000);
30     } else if ( i == 3 ) {
31         digitalWrite(PIN_LED_GREEN, LOW);
32         digitalWrite(PIN_LED_YELLOW, LOW);
33         digitalWrite(PIN_LED_RED, HIGH);
34         delay(2000);
35         i = 0;
36     }
37 }
```

LED Array를 사용한 제어 응용

- 바로 이전에 작성했던 프로그램 소스를 함수로 나누어서 LED가 반짝거리도록 코드를 수정 해 보세요.

Control_application

```

1 // 함수 정의
2 // int nextLED(int);
3 void callGreen();
4 void callYellow();
5 void callRed();
6
7 const int SWITCH = D4;
8 const int LED_GREEN = D1;
9 const int LED_YELLOW = D2;
10 const int LED_RED = D3;
11
12 int count;
13
14 void setup()
15 {
16     Serial.begin(115200);
17
18     pinMode(SWITCH, INPUT);
19     pinMode(LED_GREEN, OUTPUT);
20     pinMode(LED_YELLOW, OUTPUT);
21     pinMode(LED_RED, OUTPUT);
22
23     count = 0;
24 }
25
26 void loop()
27 {
28     count++;
29     switch (count)
30     {
31         case 1:
32             callGreen();
33             break;
34         case 2:
35             callYellow();
36             break;
37         case 3:
38             callRed();
39             count = 0;
40             break;
41     }
42     delay(500);
43 }

```

```

45 void callGreen()
46 {
47     for (int i = 0; i < 10; i++)
48     {
49         digitalWrite(LED_GREEN, HIGH);
50         delay(100);
51         digitalWrite(LED_GREEN, LOW);
52         delay(100);
53     }
54 }
55
56 void callYellow()
57 {
58     for (int i = 0; i < 10; i++)
59     {
60         digitalWrite(LED_YELLOW, HIGH);
61         delay(100);
62         digitalWrite(LED_YELLOW, LOW);
63         delay(100);
64     }
65 }
66
67 void callRed()
68 {
69     for (int i = 0; i < 10; i++)
70     {
71         digitalWrite(LED_RED, HIGH);
72         delay(100);
73         digitalWrite(LED_RED, LOW);
74         delay(100);
75     }
76 }
77

```

- QnA