

PC 제어 프로그래밍

학습2: 프로그래밍하기

교과목 소개

NCS 학습 모듈

✦ PC 제어 프로그램 개발

학습 1	프로그램 개발 준비하기	5 주간
학습 2	프로그래밍하기	
학습 3	시뮬레이션 및 프로그램 수정보완 하기	

2-1. 모듈별 프로그래밍

학습목표

- 제어 프로그램 설계서를 바탕으로 하여 초급기술자가 데이터 구조를 파악할 수 있도록 프로그램 데이터구조를 코딩할 수 있다.
- 사용자 인터페이스 설계서를 바탕으로 화면 사용자 인터페이스 설계를 할 수 있다.
- 장비 인터페이스 사양서를 바탕으로 장비 인터페이스 통신 모듈을 코딩할 수 있다.

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 하드웨어 & 소프트웨어 & 프로그램

하드웨어(Hardware)는 사전적인 의미로 철물, 기계를 의미한다.

생활에서 사용하고 있는 전화, 냉장고, TV, 리모컨 등등 모든 전자기기들이 바로 하드웨어이다. 하지만, 이것은 형태만 있을 뿐 움직일 수 없다.

우리가 원하는 기능 즉, 전화를 걸고, 받고, 음식물을 보관하고, TV를 켜고 끄고 등등의 기능을 하도록 하기 위해서 필요한 것이 있다. 바로 소프트웨어(Software)다.

소프트웨어(Software)는 철물 덩어리가 어떤 기능을 할 수 있도록 순서와 절차에 맞게 지시를 내리는 명령들을 모아 놓은 것이다. 그리고 이것을 **프로그램(Program)**이라고 한다.

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자



명령문의 끝 ; (세미콜론)

우리가 한글을 배울 때 한 문장이 끝나면 .(마침표)를 찍어준다.

프로그램에서도 명령이 끝났다는 것을 알려주기 위해서 ; (세미콜론)을 찍어준다.

세미콜론이 나오지 않으면 아직 명령어 문장이 끝나지 않은 것으로 인식한다.

```
int a = 10;
```

<- 명령문 종료

```
int num = 23
```

<- 명령문 종료 안됨

```
;
```

<- 여기서 종료 됨

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 명령문 집합 블록 { }

여러 개의 명령문이 모여서 하나의 기능을 수행할 때 명령문을 묶어주는 것을 블록 { } 이라 한다.

블록 { 명령문; } 안은 또 하나의 영역으로 변수를 선언할 수도 있고, 이런 변수를 지역 변수라고 한다.

```
void add(int a, int b, int cnt)
```

```
{  
    int r = a + b;  
  
    Serial.print(" Result = " );  
    Serial.println( r);  
}
```

3개의 명령문 블록

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 자료형(Data Type)

우리가 택배를 보낼 때 물건의 크기에 맞는 상자에 물건을 담아서 보낸다.
큰 상자에 작은 물건을 담아서 보내면 적당한 상자보다 상자 구매 비용이 더 발생할 것이고, 반대로 작은 상자에 물건을 담을 경우, 들어가지 않아서 물건을 잘라내고 담아야 할 것이다.

프로그램에서도 데이터를 저장하기 위해서 메모리(Memory)라는 공간이 있다.

이 공간이 무한대로 많은 것이 아니기 때문에 저장할 데이터에 맞는 공간만큼만 사용을 해야 한다.

그래서 데이터 별로 사용 가능한 최적의 크기를 지정해 두었는데 이것이 바로 **자료형(Data Type)**이라 한다.

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

타입 (TYPE)		특징	사용 예
논리형	boolean	- 크기 : 1바이트 - 데이터 : true, false - 범위 : 0, 1	boolean isOn=false;
문자형	char	- 크기 : 1바이트 - 데이터 : 문자 한 개 - 범위 : -128 ~ 127	char c='a';
	unsigned char	- 크기 : 1바이트 - 데이터 : 문자 한 개 - 범위 : 0 ~ 255	char c='a';

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

정수 수치형	byte	<ul style="list-style-type: none"> - 크 기 : 1바이트 - 데이터 : 정수 - 범 위 : 0 ~ 255 	byte b=B1001;
	int	<ul style="list-style-type: none"> - 크 기 : 2바이트 - 데이터 : 정수 - 범 위 : -32,768 ~ 32,767 	int pin=13;
	unsigned int word	<ul style="list-style-type: none"> - 크 기 : 2바이트 - 데이터 : 정수 - 범 위 : 0 ~ 4,294,967,295 	unsigned int pin=13;
	long	<ul style="list-style-type: none"> - 크 기 : 4바이트 - 데이터 : 정수 - 범 위 : $-2^{32-1} \sim 2^{32-1}-1$ 	long value=186000L;
	unsinged long	<ul style="list-style-type: none"> - 크 기 : 4바이트 - 데이터 : 정수 - 범 위 : $0 \sim 2^{32}-1$ 	unsigned long time;
실수 수치형	float	<ul style="list-style-type: none"> - 크 기 : 4바이트 - 데이터 : 실수 - 범 위 : $-2^{32-1} \sim 2^{32-1}-1$ - 정밀도 : 소수점 이하 7 	float f = 0.123;
	double	<ul style="list-style-type: none"> - 크 기 : 4바이트 - 데이터 : 실수 - 범 위 : $-2^{32-1} \sim 2^{32-1}-1$ - 정밀도 : 소수점 이하 14 	double d = 0.121234;

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 상수(Constance)

우리가 사용하는 지역번호 053은 대구, 02는 서울이라고 정해 둔 것처럼 프로그램에 필요한 숫자 값들을 우리가 이해하기 쉬운 단어로 선언해 둔 것을 상수(Constance)라 한다. 그리고 상수는 프로그램 중간에 값을 변경할 수 없다.

상수는 프로그램 개발 시에 만들 수도 있고, 아두이노에서는 스케치 프로그램시에 사용될 상수를 미리 만들어서 선언해 두었다.

```
//- 핀으로 보낼 데이터 값
#define HIGH          1    //- On
#define LOW           0    //- Off

//- 핀의 입출력용도
#define INPUT         0    //- 입력 모드
#define OUTPUT        1    //- 출력 모드
```

```
void setup( ) {
    pinMode(13, OUTPUT);    //- 상수 사용
}
```

개발자가 프로그램 시에 새롭게 만들 수도 있다.

```
[ 방법 1 ] const int PIN_NO = 13;          //- 상수 선언
[ 방법 2 ] #define PIN_NUM  13             //- 매크로 상수 선언

void setup( ) {
    pinMode(PIN_NUM, OUTPUT);               //- 매크로 상수 사용
    pinMode(PIN_NO, OUTPUT);               //- 상수 사용
}
```

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 변수(Variable)

변수는 프로그램 중에 사용되는 데이터를 저장하는 공간이다.

우리가 메모를 하거나 연습장에 계산식을 적는 것처럼 프로그램에서 일을 하기 위해서 사용되는 데이터를 담아두는 연습장과 같은 것을 변수(Variable)라 한다.

이것은 상수와 반대로 언제든지 변경이 가능하다.

아두이노에서는 데이터를 메모리라는 공간에 저장한다. 그리고 메모리 공간을 효율적으로 사용하기 위해서 저장할 데이터 종류별로 사용가능한 메모리 크기를 정해서 자료형(Data Type)으로 만들어 두었다.

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

사용할 공간의 크기를 정하고 그 공간에 데이터를 저장하거나 꺼내 오기 위해서 이름을 붙여주어야 한다. 이것을 변수 선언이라 한다.

[변수 선언 및 초기화 형식]

데이터 크기	저장 공간 이름 ;	//- 변수 선언
데이터 크기	저장 공간 이름 = 데이터 ;	//- 변수 선언 + 초기화

int ledPin = 13;	//- 변수 선언 + 초기화
int count;	//- 변수 선언
void setup() {	
pinMode(ledPin , OUTPUT);	//- 변수 사용
count = 100;	//- 변수 초기화
}	

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 배열(Array)

만약 한 사람에게 동일한 물건을 6개 보낸다고 생각해보자.

6개의 상자에 물건을 하나씩 담고, 6번 주소를 적어서 보내야 한다.

차라리 1개의 큰 상자에 6개 물건을 담고 주소를 한번 적어서 보낸다면 좀 더 수월할 것이다.

프로그램에서도 동일한 자료형의 데이터를 여러 개 저장해야 할 경우 배열(Array)이라는 것을 사용한다. 배열은 연속된 메모리 공간을 제공하고, 이 공간을 하나의 이름으로 사용할 수 있도록 했다.

대신에 여러 개의 데이터가 있으므로 식별하기 위해서 번호를 부여하고 번호로 데이터가 담긴 공간에 접근할 수 있다. 이 번호를 **첨자** 또는 **인덱스(Index)**라 한다.

예를 들어서 아파트 101동에는 동일한 구조의 집들이 10채가 있다. 이 10채의 집을 구별하기 위해서 101호, 201호, 301호,, 1001호 방식으로 번호가 부여되어 있다.

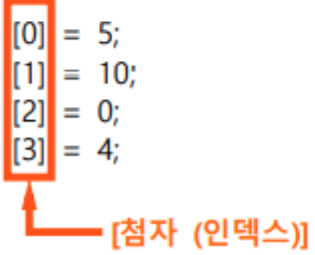
여기서 101동이 배열이름이 되고, 101 ~ 1001이 바로 첨자가 된다.

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

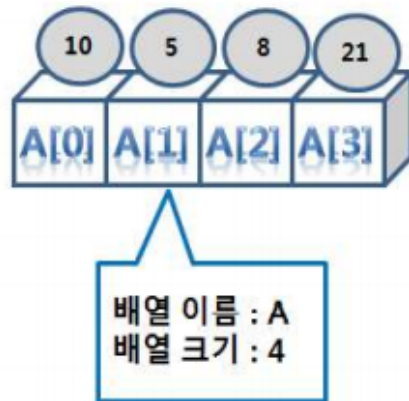
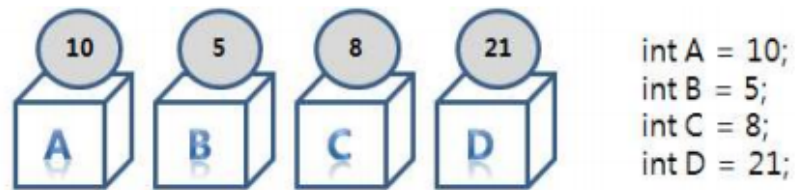
첨자는 배열의 크기를 의미하며 범위는 0 ~ (배열 크기-1)이다.

[배열 선언 및 사용]	
자료형 배열이름 [첨자]	
int values [4];	//- int형 4개의 데이터가 저장되는 공간
values [0] = 5;	//- 0번째 공간에 5 저장
values [1] = 10;	//- 1번째 공간에 10 저장
values [2] = 0;	//- 2번째 공간에 0 저장
values [3] = 4;	//- 3번째 공간에 4 저장

 [첨자 (인덱스)]

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자



* 선언 방법(1)

```
int A[4] = {10, 5, 8, 21}; // - 선언+초기화
```

* 선언 방법(2)

```
int A[4];  
A[0] = 10;  
A[1] = 5;  
A[2] = 8;  
A[3] = 21;
```

// - 선언
// - 초기화

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 문자열(string)

하나의 문자가 아닌 문장이나 단어 등 여러 개의 문자가 모여 있는 것을 문자열(string)이라 한다.

문자열은 쌍 따옴표("")로 감싸서 표기를 한다. 그리고 한 개 이상의 문자가 모여 있기 때문에 char 타입의 배열에 저장해야 한다.

주의할 것이 하나있다.

문자열은 기본적으로 끝에 Null 종료 문자라는 것이 붙는다.

왜냐하면, 데이터 전달 시 문자열의 끝을 알기 위해서 사용한다.

그래서 문자열을 배열에 저장할 경우에는 '문자개수 + 널 종료 문자'로 배열 크기를 잡아 주어야 한다.

Null 종료 문자는 ASCII code 값은 0이고, 특수 문자 '\0'로 약속되어 있다.

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 문자열(string)

```
char messages[10] = "Good";  
char messages[ ] = "Happy";  
char messages[8] = "arduino";           <= 7문자 + null 종료 문자  
char str1[8] = { 'a', 'r', 'd', 'u', 'i', 'n', 'o' };   <= 문자 7개 저장  
char str1[8] = { 'a', 'r', 'd', 'u', 'i', 'n', 'o', 'W' }; <= 문자열 저장
```


아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 함수(Function)

프린터는 프린트를 한다. 계산기는 계산을 한다.

프로그램에서 특정 기능을 수행하는 코드를 모아 놓은 것을 함수 Function이라한다.

프린트를 할 때 마다 프린터기를 만들지 않는 것처럼 함수를 잘 만들어 두면 특정 기능 수행할 경우 만들어진 함수를 사용만 하면 된다.

함수를 사용함으로써 중복되는 코드가 줄어들고, 이미 만들어진 코드를 재사용할 수 있어서 좋다.

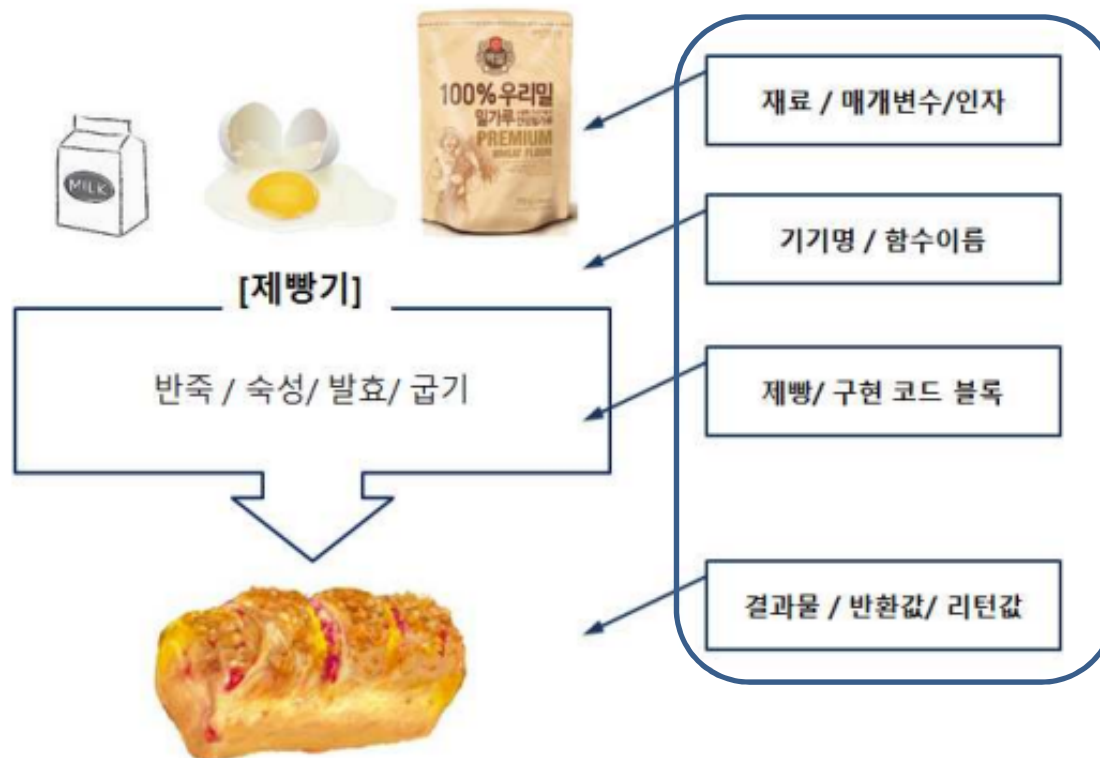
아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 함수(Function)

예를 들어서 빵을 만드는 제빵기가 있다.

제빵기가 빵을 만들기 위해서는 밀가루, 달걀, 우유 등등의 재료가 필요하고 이 재료를 제빵기 안에 넣어주면 제빵기가 빵을 만들어서 준다.



아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 함수(Function)

[함수 선언]

```
반환타입 함수이름(매개변수타입 매개변수이름)
{
    // 기능 구현 코드

    return 결과; // 결과 리턴(반환)
}
```

함수기능	정수 2개를 더하기	<pre>int add(int a, int b) { int result = a + b; return result; }</pre>
함수이름	add	
매개변수	int a, int b	
반환타입	int	

함수가 기능을 수행 후에 결과물이 있을 수도 있고 없을 수도 있다.

만약 결과물이 없을 경우 사용하는 키워드가 있다. 바로 void이다.

함수를 선언할 경우 반환 결과가 없다면 void라고 반환타입에 작성하면 된다.

함수기능	메시지 출력	<pre>void displayMessage(String msg) { Serial.println(msg); }</pre>
함수이름	displayMessage	
매개변수	String msg	
반환타입	void	

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 함수(Function)

만들어진 함수를 사용하는 것을 **함수 호출**이라 한다.

[함수 호출]

함수이름(매개변수이름);

```
void displayMessage(String msg)
{
    Serial.println(msg);
}
```

```
int add(int a, int b)
{
    int result = a + b;
    return result;
}
```

```
int total =0;
String msg="Total : ";

for(int num=0; num<10; l++)
{
    total += add( num, num*10);
}

msg += total;

displayMessage(msg);
```

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 전역 변수와 지역 변수

변수는 선언되는 위치에 따라서 변수의 사용 범위가 달라지는데 이것을 스코프(Scope)라하며 전역 변수, 지역 변수라 부른다.

	전역 변수(Global Variable)	지역 변수(Local Variable)
위치	함수 코드 블록 밖에 선언	함수 코드 블록 안에 사용
범위	같은 파일 안에 있는 모든 함수에서 사용 가능	다른 함수 사용 불가. 해당 함수에서만 사용 가능

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 전역 변수와 지역 변수

```
int globalVar=10;

void setup( ) {
    globalVar = 11;
    localVar = 21;    // loop()함수 안에 선언된 변수, 다른 함수 사용불가
    ERROR 발생
}

void loop( ) {
    int localVar=20;

    localVar=21;
    globalVar=12;    //- 함수 밖에 선언되어 있는 전역변수로서 사용 가능
}
```

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 연산자(Operator)

연산자(Operator)란? 프로그램에서 기능을 수행하기 위해서 계산에 사용되는 다양한 기호들이며 연산의 대상이 되는 것을 피연산자(Operation)이라 한다.

(예) $10 + 20$	연산자 : $+$	피연산자 : $10, 20$
---------------	-----------	-----------------

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 연산자(Operator)

연산자	기호	역할
대입	=	오른쪽에 있는 것을 왼쪽에 넣기
		(예) int a = 10;
산술	+, -, *	덧셈, 뺄셈, 곱셈, 나눗셈, 나머지 연산 수행
	/, %	(예) 1+3, 10-8, 2*9, 10/3, 7%2
관계	==, !=	피연산자 값의 관계 검사 후 true, false 반환
	<, > <=, >=	(예) int a = 10; int b = 30; a == b -> false a != b -> true
논리	&&,	관계 연산자와 같이 사용되어 두 개 이상의 조건식을 검사 후 true, false 반환
		(조건A) && (조건B) 조건A, B 모두 참일 때 true
		(조건A) (조건B) 조건A, B 중 하나만 참이면 true
		(예) int a=10; int b=30; (a > b) && (a != b) -> false (a > b) (a != b) -> true

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 연산자(Operator)

연산자	기호	역할
증감	++, --	변수의 값을 1 증가 또는 1 감소시키는 연산자
		(예) int a = 10; a++; // a는 11 a--; // a는 10
복합	+=, -=	대입 연산자와 다른 연산자를 결합해 식을 간단히 하는 연산자
	*=, /= %=	(예) a = a+10; -> a += 10; b = b/2; -> b /= 2;

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 제어문 - 조건문

프로그램 실행 시에 조건에 따라서 프로그램의 실행 순서를 변경하기 위한 것이다.

[조건문 if~문]

기능	프로그램 실행 중 조건에 따라서 프로그램 실행 순서를 변경하는 것	
형태	if(조건){ A } 문	(조건식)이 참인 경우만 A 실행
	if(조건){ A } else{ B } 문	(조건식)이 참인 경우 A, 거짓이면 B실행
	if(조건)~ else if(조건)~ 문	조건이 여러 개인 경우 사용 첫 번째, 두 번째 조건 반복 검사 후 실행

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 제어문 - 조건문

[조건문 if~문]

예시	<pre>if(val == 0) { digitalWrite(13, LOW); }</pre>	<p>//- 조건이 참인 경우만 동작하는 경우</p>
	<pre>if(val == 0) { digitalWrite(13, LOW); } else { digitalWrite(13, HIGH); }</pre>	<p>//- 참인 경우의 동작</p> <p>//- 거짓인 경우의 동작</p> <p>* 참/거짓에 따라 다른 동작을 하는 경우</p>
	<pre>if(val == 0){ digitalWrite(14, LOW); } else if(val == 1) { digitalWrite(13, LOW); } else { digitalWrite(13, HIGH); digitalWrite(14, HIGH); }</pre>	<p>//- 조건이 0일 경우 동작</p> <p>//- 조건이 1일 경우 동작</p> <p>//- 그 외의 경우의 동작</p> <p>* 조건 하나에 다양한 동작이 있는 경우</p>

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 제어문 - 선택문

다양한 경우의 수가 존재하는 경우 사용되는 제어문이다.

[선택문 switch~case~문]

기능	하나의 조건에 여러 경우가 발생할 경우 사용 조건문의 if~ else if~ else~구문과 동일 기능이지만 코드는 훨씬 간결 단! 조건은 반드시 변수만 사용 가능	
형태	<pre>switch(조건 변수){ case 경우1: break; case 경우2: break; default: break; }</pre>	(조건 변수)의 각 경우에 따른 case문 실행 해당하는 경우가 없을 경우 default 실행

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 제어문 - 선택문

[선택문 switch~case~문]

예시

```
switch(val) // 조건 변수 val에 여러 경우의 처리
{
    case 0: // val의 값이 0일 경우
        digitalWrite(13, LOW);
        digitalWrite(14, LOW);
        break;

    case 1: // val의 값이 1일 경우
        digitalWrite(13, LOW);
        digitalWrite(14, HIGH);
        break;

    case 2: // val의 값이 2일 경우
        digitalWrite(13, HIGH);
        digitalWrite(14, LOW);
        break;

    default: // 이 외의 경우
        digitalWrite(13, HIGH);
        digitalWrite(14, HIGH);
        break;
}
```

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 제어문 - 반복문

프로그램 실행 시에 동일한 코드를 계속적으로 실행해야 할 경우 사용된다.

중복 코드를 제거할 수 있다.

[반복문 for~문]

기능	지정된 횟수만큼 동일한 코드 반복 실행	
형태	<pre>for(시작; 완료; 증감){ //- 반복 코드 }문</pre>	반복의 횟수를 아는 경우에 사용 시작 - 반복 횟수 초기화 완료 - 반복 횟수 지정 증감 - 반복 횟수 계산
예시	<pre>for(int i=0; i<10; i++) { //- 10번 반복 digitalWrite(13, HIGH); delay(1000); digitalWrite(13, LOW); delay(1000); }</pre>	

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 제어문 - 반복문

[반복문 while~문]

기능	조건이 true인 경우까지만 반복 수행	
형태	<pre>while(조건식) { // 반복할 내용 }</pre>	반복의 횟수를 알 수 없는 경우 (조건식)이 참일 경우만 수행됨
예시	<pre>int i=0; while(i<10) { digitalWrite(13, HIGH); delay(1000); digitalWrite(13, LOW); delay(1000); i++; }</pre>	

아두이노 스케치의 기본 구조와 문법

1-1. 프로그래밍 기본을 알자

▶ 주석문

프로그램을 작성 시 코드에 대한 설명을 적어둔 것을 주석(Comment)이라 한다.
시간이 지난 뒤에 기능을 파악 및 수정 보완 작업이 수월해지기 때문이다.

컴파일 및 프로그램 실행에는 영향을 미치지 않는다.

한 줄 주석	//	[예] int a = 20; // boolean bOn = false; char name[10]="Tom";
여러 줄 주석	/* ~ */	[예] /* int a = 20; boolean bOn = false; char name[10] = "Tom"; */

아두이노 스케치의 기본 구조와 문법

1-2. 아두이노 스케치(Sketch) 프로그램

스케치(Sketch) 프로그램은 C, C++언어를 사용한 프로그램 개발 도구로써 개발, 컴파일, 실행까지 모두 가능하다.

일반적인 C, C++ 프로그램과는 조금 다른 구조로 `main()` 함수가 보이지 않는다.

대신 필수 함수 2개가 존재하며 반드시 구현해 주어야 한다.

아두이노 스케치의 기본 구조와 문법

1-2. 아두이노 스케치(Sketch) 프로그램

▶ 스케치 프로그램의 기본 구조

```
#define PIN_NUM 10

//-----
//- 전역변수 : 객체타입 변수, 프로그램 제어 변수
//-----
boolean bOn = false;
RgbLed led;

//-----
//- 초기화 함수 : 프로그램 초기화 및 설정, 전원 On된 후 1번만 실행
//-----
void setup( ) {
}

//-----
//- 기능 구현 함수 : 전원이 Off되기 전까지 무한 반복 수행
//-----
void loop( ) {
}
```

아두이노 스케치의 기본 구조와 문법

1-2. 아두이노 스케치(Sketch) 프로그램

▶ 스케치 프로그램의 기본 구조

아두이노에 전원을 인가하면 내부적으로 코드에서 `setup()`이라는 함수를 찾아 함수의 내용을 실행한다. 그 이후, 다시 `loop()`라는 함수를 찾고 `loop()` 함수를 무한히 반복하게 된다.

