

STEAM 교육과 Arduino

박종화

(경기북과학고등학교, 교사)

I . Arduino란

아두이노란 오픈소스 하드웨어 플랫폼으로서 물리적인 장치를 구현할 수 있는 보드와, 이를 구동시키기 위한 프로그램을 수행할 수 있는 통합 개발 환경과, 오픈 소스의 철학이 들어있는 통합적인 의미를 가지고 있다. 아두이노의 배경으로는 피지컬 컴퓨팅이 있으며 이는 컴퓨터와 인간의 상호 작용의 폭을 더 넓히기 위해 보다 쉬운 개발 환경을 제공하여 전문적인 교육을 받지 않은 사람들도 쉽게 하드웨어와 프로그래밍을 통해 컴퓨터와 의사소통을 하고 자신이 구현 하고 싶은 것을 만들 수 있도록 도와주는 것이다.

아두이노를 구성하는 하드웨어는 사용자가 작성한 프로그램이 실제로 실행되는 곳으로서 AVR 계열의 마이크로 컨트롤러를 기반으로 작동을 한다. 아두이노 하드웨어는 다양한 변형이 존재하며, 오픈소스 하드웨어이니 만큼 원한다면 직접 하드웨어를 구성할 수 있다. 아두이노 보드 자체만으로는 외부 환경과의 상호작용을 할 수 없으며 다양한 센서나 부품들을 연결하여 원하는 작업을 수행하게 된다.

아두이노의 개발 환경은 프로세싱 언어의 IDE에 기반을 두고 있으며, C언어와 거의 유사한 문법을 통해서 하드웨어를 제어하게 된다. 아두이노 IDE는 윈도우 뿐만 아니라 다양한 운영체제에서 실행할 수 있으며, 프로그램을 작성하고 컴파일과 보드로의 업로드를 통해 원하는 실행 결과를 얻을 수 있게 된다. 아두이노는 전 세계적으로 수많은 교육 프로그램에서 사용되고 있으며, 특히 기술에 대한 깊은 이해 없이도 자신이 만들고자 하는 것을 쉽게 만들 수 있으며 그로 인한 많은 사용자를 가지고 있다. 정보교과 뿐만 아니라 다양한 교과에서 아두이노를 융합한 활용의 가능성에 주목하고 있으며 이는 앞으로 더욱 다양한 아두이노의 사용을 가져오게 될 것이다.

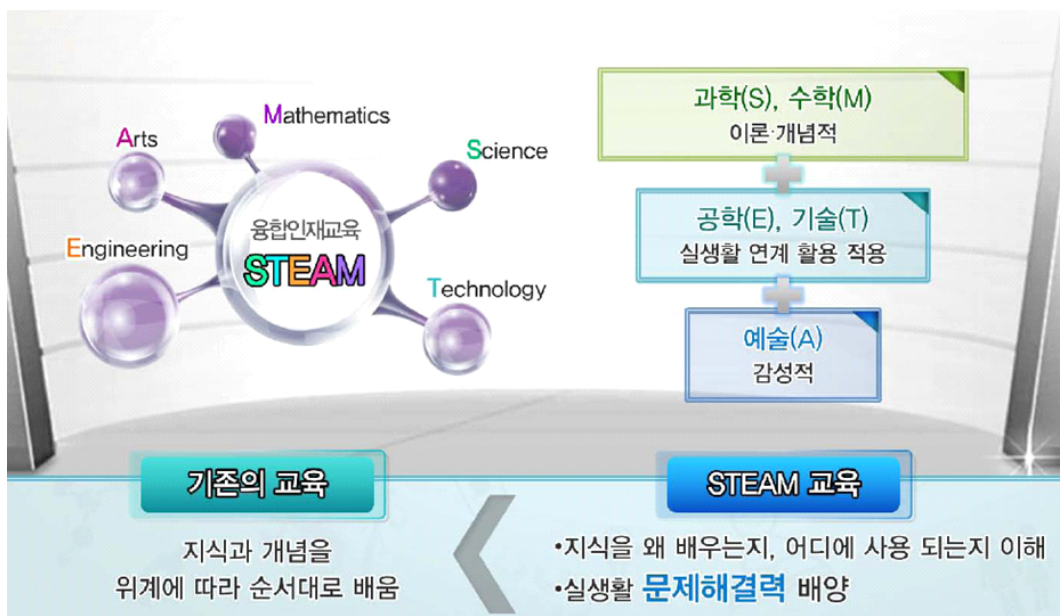
Ⅱ. 정보교과의 STEAM 사례

1. STEAM 교육 학습 준거

STEAM(융합인재교육)은 과학기술에 대한 학생들의 흥미와 이해도를 높이고 과학기술 기반의 융합적 사고와 문제해결력을 배양하는 교육이다. STEAM은 창의적 설계와 감성적 체험을 통해 과학 기술과 관련된 다양한 분야의 융합적 지식, 과정, 본성에 대한 흥미와 이해를 높여 창의적이고 종합적으로 문제를 해결할 수 있는 융합적 소양을 갖춘 인재를 양성하는 것을 목표로 하고 있다.

창의적 설계(Creative Design)는 학습자들이 주어진 상황에서 지식, 제품, 작품 등과 같은 산출물을 구성하기 위하여 창의성, 효율성, 경제성, 심미성을 발현하여 최적의 방안을 찾아 문제를 해결하는 종합적인 과정. 창의적 설계의 과정은 수학, 과학적 문제해결의 과정이나 공학의 설계 과정과 매우 유사하다.

감성적 체험(Emotional Touch)은 학습자가 학습에 대한 흥미, 자신감, 지적 만족감, 성취감 등을 느껴 학습에 대한 동기유발, 욕구, 열정, 몰입의 의지가 생기고 개인, 타인 및 공동체와의 관계, 자연과 문화 등의 의미를 발견하여 선순환적인 자기 주도적 학습이 가능하게 하는 모든 활동과 경험” 즉 학습에 대한 긍정적 감정을 느끼고 성공의 경험을 하는 것이다.

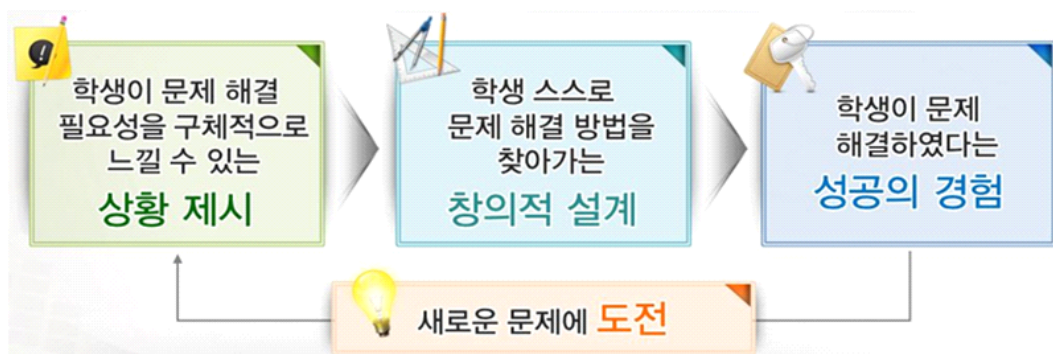


STEAM의 교육 기준은 S, T, E, A, M 중 반드시 2개 이상의 교과 혹은 요소를 포함하기를 권장하고 있다. 또한 STEAM 자체가 가지는 원래 의도에 부합하게 과학기술에 대한 흥미와 이해를 높일 수 있도록 과학 내용을 반드시 포함하기를 권고하고 있다.

STEAM에서의 상황제시란 학생이 문제 해결 필요성을 구체적으로 느낄 수 있는 것으로서, 일상생활에서 자신에게 일어날 수 있는 상황, 나의 미래에도 닥칠 수 있는 상황, 과거에도 있었던 역사적 질문 등으로 간주할 수 있으며, 가장 중요한 것은 학생의 흥미를 자극할 수 있는 상황이어야 한다. 단순한 동기유발과는 차이점을 지니게 된다.

창의적 설계과정이란 학생 스스로 문제 해결 방법을 찾아 가는 과정으로서 문제 중심, 주제중심, 프로젝트 기반 등의 학습자 중심의 열린 문제에서 증거를 활용하고, 타당한 질문을 통해 문제를 해결하는 능력의 과정이다. 이는 기존의 교사 주도하에 이루어지는 학습이 아니며, 교사가 의도한 목표에 도달하기 위해, 학생들이 스스로 해결방법을 찾아 문제상황을 해결해 나가는 과정인 것이다.

성공의 경험은 학생이 문제를 해결하였다는 것이며, 이는 적절한 어려움의 과정을 통해서 실패를 극복하며 칭찬과 동료들 통해 다양한 경험을 하게 되는 과정을 뜻한다. 이러한 감성적 체험은 학습에 대한 즐거움과 자신감을 수업에서 학생들이 가지게 되는 것이며, 문제 해결 과정에서 실패를 경험하고, 실패를 통한 성공의 경험을 통해 새로운 문제에 적극적으로 도전할 수 있는 능력을 키우는 것이다. 또한, 자기평가 및 동료평가의 과정을 통해 타인의 결과물에도 관심을 가져보며 자신만의 사고를 확립할 수 있는 다양한 경험들을 제공하는 것이다.



2. 정보과 STEAM 수업사례1

가. 주제개요

C 언어는 유닉스 운영체제를 개발하기 위해 사용이 되었으며, 기억장치를 비롯한 하드웨어를 직접 제어하기 때문에 고급 언어와 저급 언어의 특징을 모두 가지고 있다. C언어를 이용한 프로그램의 작성과 실행 과정은 PC의 콘솔 상에서 실행이 되고 동작이 되며 이로 인해 C언어가 가지는 타 분야의 적용에 대한 이해가 미흡한 실정이다.

프로그래밍 수업에서는 C언어의 기본 자료형, 표준 입출력 방법, 연산자의 활용, 제어문과 반복문을 활용하는 기법을 이미 학습하였으며, 함수를 이용하여 효과적인 프로그래밍을 작성하는 연습을 하였다. 수업에서는 지금까지 배운 프로그래밍 지식과 오픈소스 하드웨어인 아두이노(Arduino)를 활용하여 중력 가속도를 측정하는 과정을 수행한다.

중력가속도는 단진자의 주기를 측정하는 과정을 통해 구하며, 포토게이트라는 센서를 이용하여 아두이노와 연결하는 간단한 전자 회로를 구성하며 센서를 지나가는 진자의 시간과 중력가속도 값을 구하는 프로그램을 C언어를 이용하여 작성한다. 주기를 측정하기 위한 간단한 실험 장치를 직접 제작하고, 센서의 값을 입력으로 받고 처리하는 과정은 C언어를 이용하여 처리한다. 이를 통해 실제 하드웨어의 동작을 제어하고 값을 처리하는 과정을 프로그래밍 언어를 통해서 수행할 수 있으며, 실세계의 데이터를 처리 하는 과정을 통해 보다 다양한 응용 가능성에 대한 이해를 높일 수 있는 계기를 갖게 하고자 하였다.

나. 내용 목표 및 과정 목표

프로그래밍을 통해 센서의 값을 읽을 수 있다.

효율적으로 프로그래밍 하기 위한 다양한 구조들을 사용할 수 있다.

중력 가속도를 측정하기 효율적인 프로그램을 작성할 수 있다.

측정된 값에 대한 오차의 원인과 분석을 수행할 수 있다.

C언어의 다양한 프로그래밍 기법을 통합하는 과정을 알 수 있다.

C언어가 실제로 활용되는 다양한 분야에 대해서 생각할 수 있다.

중력가속도를 더욱 정밀하게 측정하기 위한 과정을 생각해 낼 수 있다.

팀원과 협력하여 문제를 해결하는 과정을 다양하게 적용하며 발표할 수 있다.

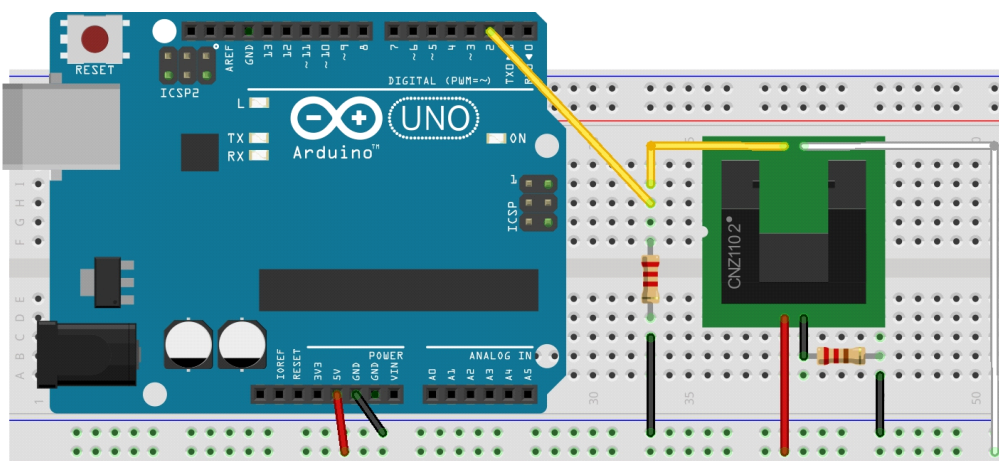
보다 다양한 실험 장치를 설계할 수 있는 가능성을 생각할 수 있다.

다. 실험 장치 제작 및 프로그래밍

단진자 운동하는 물체를 이용한 중력 가속도를 측정하기 위하여 주어진 스탠드와 실을 이용하여 포토게이트를 정확하게 지날 수 있는 실험 장치를 제작한다. 실의 길이와 추가 안정적으로 동작하기 위하여 다양한 접근 방법을 선택할 수 있다.

철재 스탠드	추
	

라. 아두이노 연결

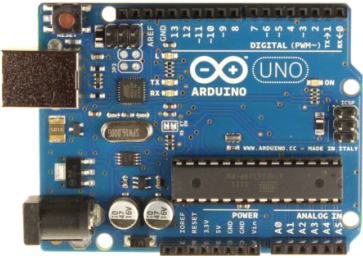
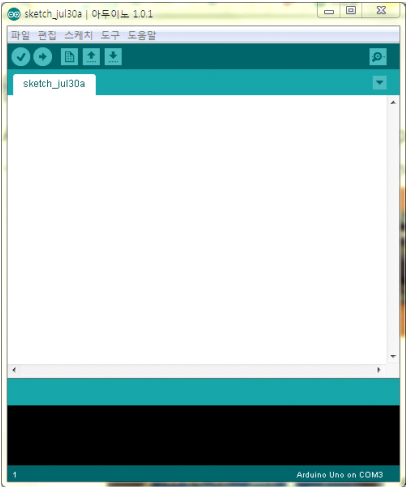


Ⅲ. 아두이노 실습

1. Arduino 개요

가. What is Arduino ?

아두이노란 오픈소스 하드웨어 플랫폼으로서 물리적인 장치를 구현할 수 있는 보드와, 이를 구동시키기 위한 프로그램을 수행할 수 있는 통합 개발 환경과, 오픈 소스의 철학이 들어있는 통합적인 의미를 가지고 있다. 아두이노의 배경으로는 피지컬 컴퓨팅이 있으며 이는 컴퓨터와 인간의 상호 작용의 폭을 더 넓히기 위해 보다 쉬운 개발 환경을 제공하여 전문적인 교육을 받지 않은 사람들도 쉽게 하드웨어와 프로그래밍을 통해 컴퓨터와 의사소통을 하고 자신이 구현 하고 싶은 것을 만들 수 있도록 도와주는 것이다. 아두이노를 구성하는 하드웨어는 사용자가 작성한 프로그램이 실제로 실행되는 곳으로서 AVR 계열의 마이크로 컨트롤러를 기반으로 작동을 한다. 아두이노 하드웨어는 다양한 변형이 존재하며, 오픈소스 하드웨어이니 만큼 원한다면 직접 하드웨어를 구성할 수 있다. 아두이노 보드 자체만으로는 외부 환경과의 상호작용을 할 수 없으며 다양한 센서나 부품들을 연결하여 원하는 작업을 수행하게 된다.

Hardware	IDE-Integrated Development Environment
	

2. 브레드보드 사용법

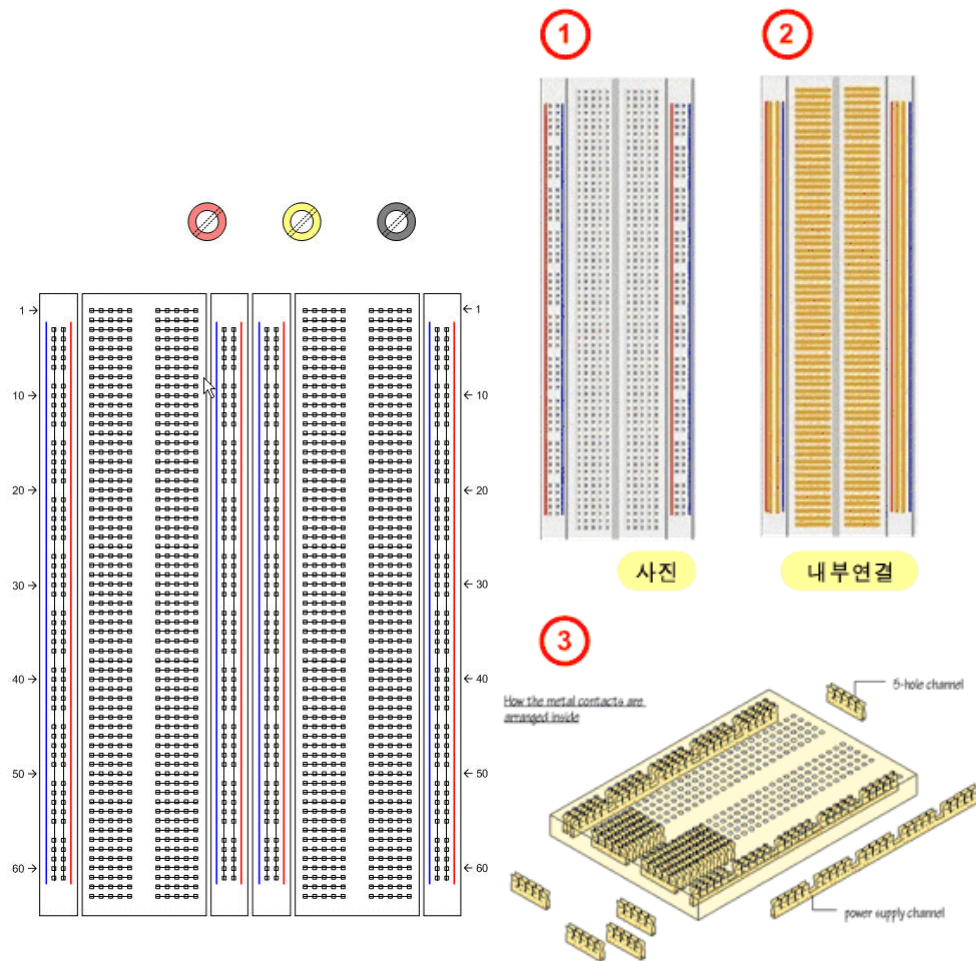
이번 예제는 아두이노 보드 상의 LED를 점멸시키는 것이 아니라 직접 브레드 보드에 하드웨어를 구성하고 프로그램을 업로드하여 LED를 점멸하는 작업을 해본다.

브레드보드(Breadboard)

브레드보드는 전자회로를 구성하는데 있어 낚땀을 하지 않고 간단하게 회로를 구성할 수 있도록 도와주는 것이다. 브레드보드는 내부적으로 배선이 되어 있으며 외부에서 전원을 공급하고 회로를 구성하여 동작하게 한다. 일반적으로 기판에 회로를 만들기 전에 시험 및 디버깅을 하는 용도로 사용이 된다.

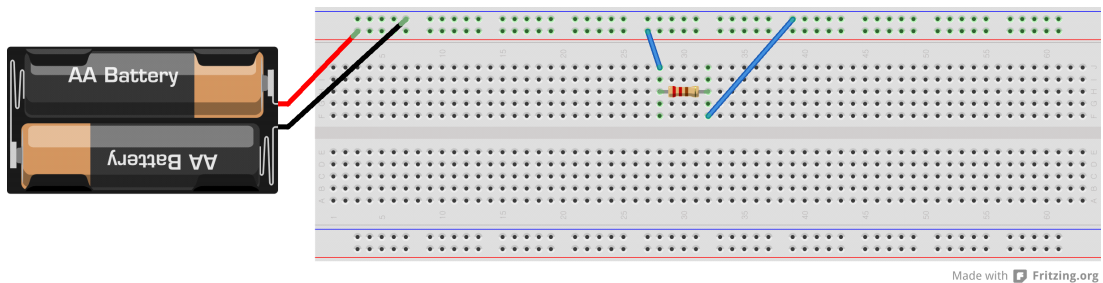
브레드보드의 사용법

일반적으로 빨간색으로 표시된 라인은 +전원, 파란색으로 표시된 라인은 GND에 연결하여 사용하며, 라인과 라인의 연결은 전선(케이블)을 이용하여 연결한다. 브레드보드의 윗쪽과 아래쪽에 있는 선들은 가로로 연결이 되어 있으며, 이 곳에 전원을 연결하여 모든 부품들이 전원을 공급 받거나, 공통 접지로 사용이 가능하게 된다. 중앙에 위치한 영역은 세로로 연결이 되어 있으며, 분리된 영역은 상하를 구분시키게 된다. 세로로 연결된 곳에 부품을 배치하는 경우에는 부품은 가로로 배치하여 회로를 구성하게 된다.



브레드보드 회로 구성 예

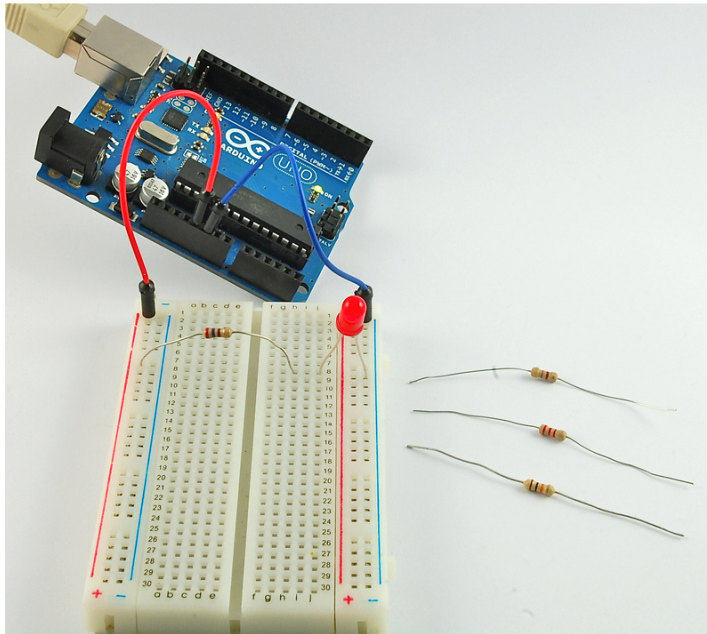
전원을 브레드 보드의 빨간색에 +, 파란색에 -를 연결한 후 부품들을 연결하여 준다. 폐회로를 구성할 수 있도록 주의하여 연결한다.



3. LED Blink With Breadboard

위에서 LED를 점멸하는 기본 예제 코드를 아두이노 보드에 부착되어 있는 소형 LED에 적용시켜 보았다. 이번 작업은 외부에 회로를 구성하여 동일하게 작동하도록 하는 작업이다.

구성도



필요 부품

아두이노 보드, 브레드보드, LED, 저항, 전선, usb cable

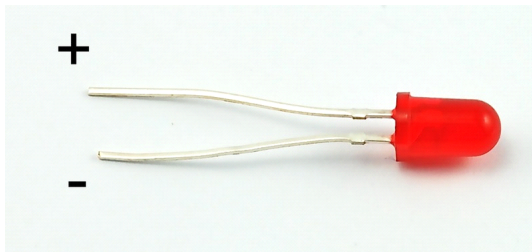
LED의 특성

LED(Light Emitting Diode)는 한쪽 방향으로만 전류가 흐르는 특성을 가지고 있으며, 다양한 색상의 빛을 낼 수 있는 저렴한 소자이다. LED는 극성을 가지고 있는 소자로서 다리가 긴 쪽이 +이며, 브레드보드에 장착할 시 극성에 주의해서 연결을 해야 한다. 또한 LED는 최대 전류가 제한되어 있으므로 LED의 파손을 방지하기 위해서 전류의 양을 제한시켜주어야 한다. LED에 흐르는 전류를 제한시켜 주기 위해서는 옴의 법칙을 이용하여 저항값을 계산한 후 회로에 적절한 저항을 추가시켜 주어야 한다. 사용하는 LED의 데이터시트를 참고하여 저항값을 계산해야 하며, 이번 작업에서는

2V의 순전압과 20mA의 전류량을 가지는 LED를 기준으로 하여 저항값을 계산한다. 이 값은 사용하는 LED 별로 다를 수 있으므로 데이터 시트를 참고하면 정확하게 저항을 계산할 수 있다. 저항의 계산은 간단한 옴의 법칙을 이용하여 다음과 같이 알 수 있다.

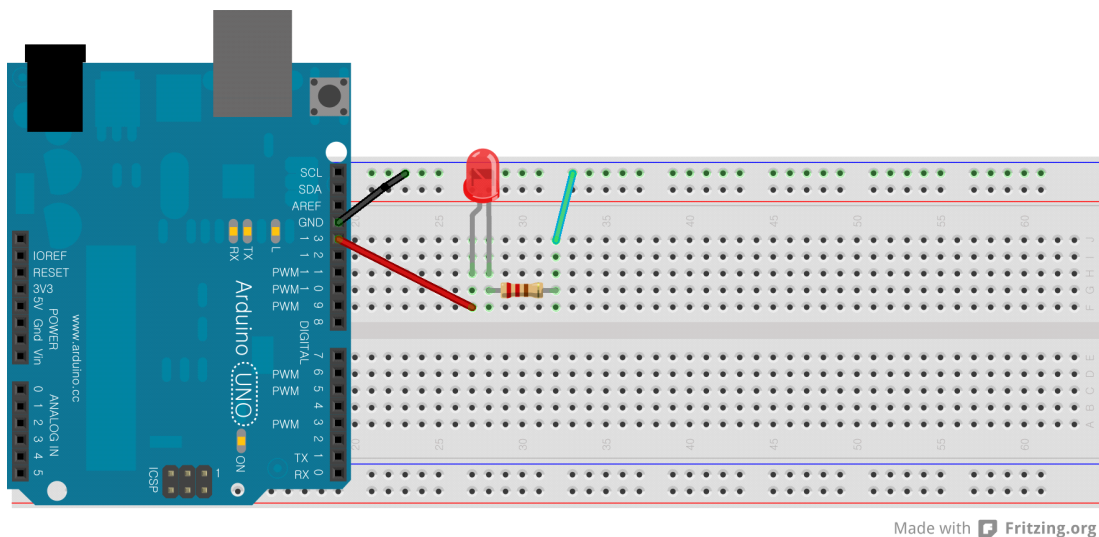
$$R = \frac{V_s - V_f}{I}$$

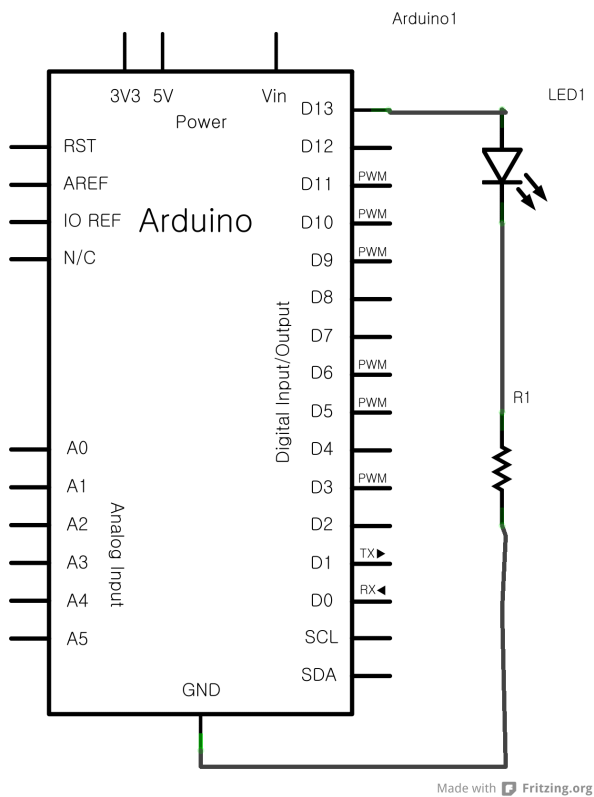
5V 아두이노를 사용하였을 경우 위 수식은 $\frac{5-2}{0.02} = 150\Omega$ 이 되며, 계산한 저항보다는 큰 저항을 사용하는 것이 안전하다. 다만 너무 높은 저항을 사용하면, LED의 밝기는 매우 약하게 될 것이다.



브레드 보드 연결

- 아두이노 13번 핀의 출력을 LED의 +에 연결하고 저항을 연결한 후 GND에 서로 연결을 시켜준다. 회로가 완전하게 구성되어 있는지 체크해보자.





프로그래밍

```
int led = 13;

void setup() {

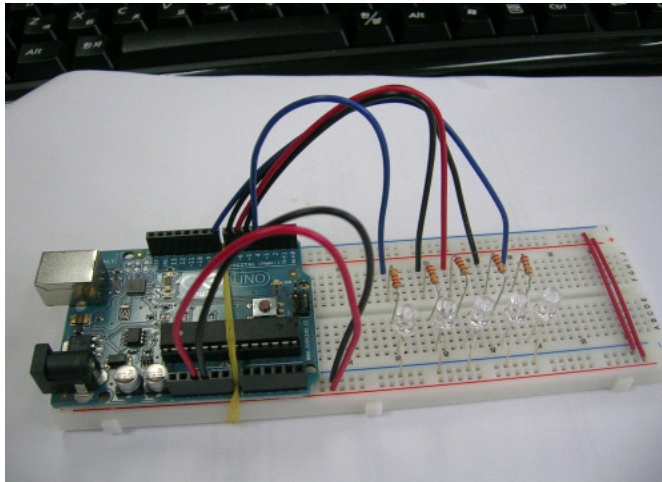
    pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(1000);
    digitalWrite(led, LOW);
    delay(1000);
}
```

4. 다중 LED 출력

이번 예제에서는 하나의 LED가 아닌 5개의 LED를 연결하여 출력하고, 연속적으로 LED를 점멸시켜 본다. for문의 사용법을 익힌다.

구성도

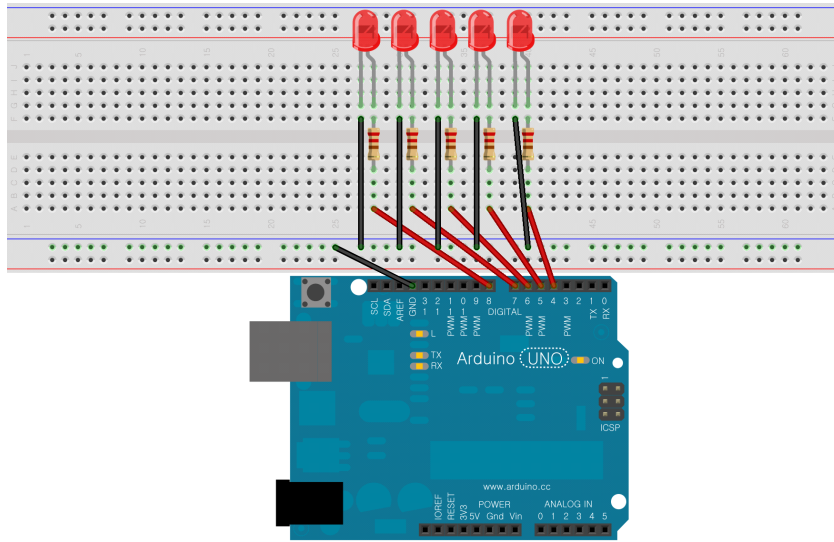


필요 부품

아두이노 보드, 브레드보드, LED, 저항, 전선, usb cable

브레드 보드 연결

- 5개의 LED를 브레드보드에 연결하고, 알맞은 저항을 연결해준다. 아두이노의 출력포트 5개를 선택하여 LED에 연결시켜주고 공통의 접지를 시켜준다.

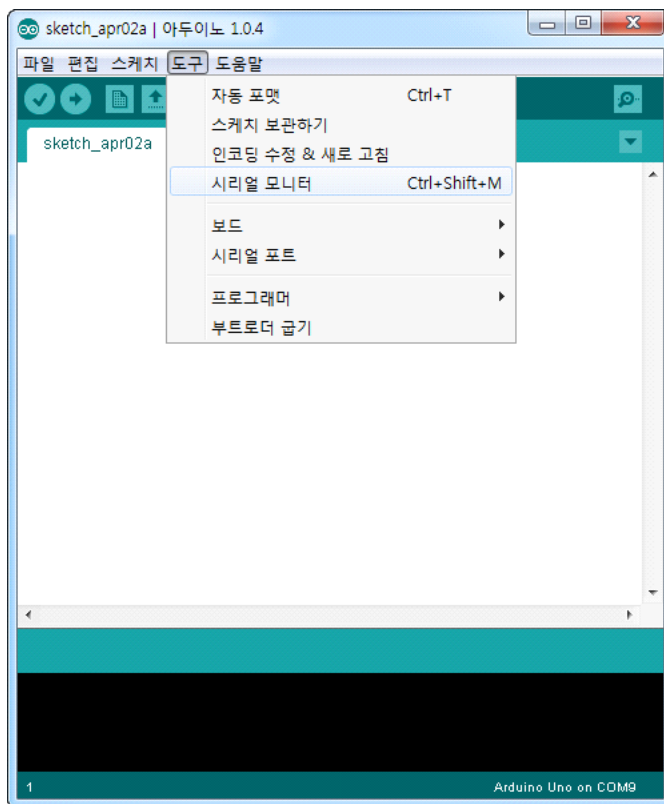


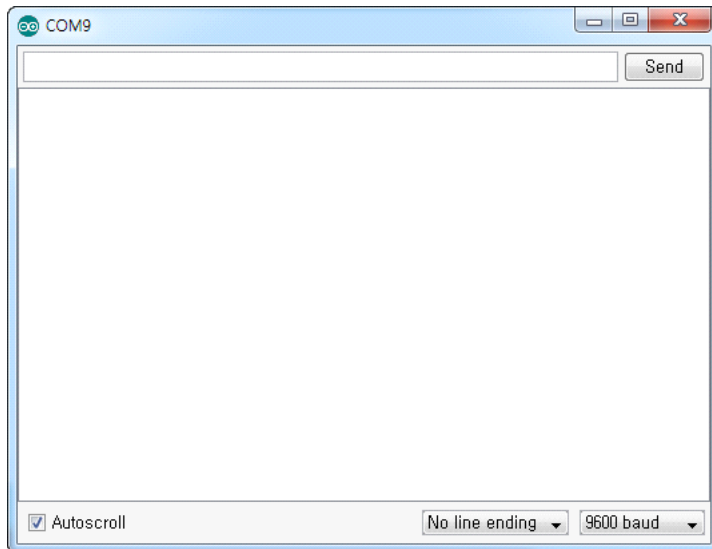
프로그램 코드

```
int del = 100;
void setup()
{
  for (int i=4; i<=8; i++) {
    pinMode(i, OUTPUT);
  }
}
void loop()
{
  for (int i=4; i<=8; i++) {
    digitalWrite(i, HIGH); // turn on LED on pin i
    delay(del);           // wait (length determined by vaue of 'del')
    digitalWrite(i, LOW);  // turn it off
  }
  for (int i=7; i>=5; i--) {
    digitalWrite(i, HIGH); // turn on LED on pin i
    delay(del);           // wait (length determined by vaue of 'del')
    digitalWrite(i, LOW);  // turn it off
  }
}
```

5. 시리얼 통신(Serial Communication)

시리얼 통신이란 하나의 선을 통하여 데이터를 순차적으로 보내고 이를 통해 두 기기의 의사소통이 이루어지는 것이다. 주위에서 흔히 볼 수 있는 USB 형태의 입력 장치들은 시리얼 통신 방식을 사용한다. 아두이노 보드는 시리얼 통신을 통해 컴퓨터와 다른 장치간에 연결을 할 수 있다. 이전의 작업에서 작성된 스케치 코드는 아두이노보드에 있는 USB-to-serial 칩을 통해 컴퓨터에 연결된 USB 포트를 시리얼 포트 사용하여 업로드가 수행된 것이다. 시리얼 통신은 프로그램의 디버깅에 효율적으로 사용될 수 있다. 업로드된 프로그램이 실행되고 있을 때 각종 변수의 값 등을 확인하기 위해서는 아두이노에서 시리얼 통신을 통해 컴퓨터로 값을 보내주고 이 값을 시리얼 모니터를 통해서 확인하게 되는 것이다.





아두이노가 보낸 데이터는 시리얼 모니터의 화면에 나타나게 되며, 사용자 컴퓨터에서 아두이노로 데이터를 보낼 때는 값을 입력 후 Send 버튼을 누르면 된다. 이때 시리얼 모니터와 아두이노 보드의 통신 속도를 동일하게 맞추어 주어야 한다.

No line Ending은 전송되는 메시지의 끝에 캐리지 리턴을 추가할 것인지를 결정 짓는 것이다. 시리얼 통신을 구현하기 위해서는 하드웨어와 소프트웨어가 필요하며, 하드웨어는 아두이노와 대상 장치 간에 전기적인 신호를 통해 정보를 주고 받는 역할을 하며, 소프트웨어는 하드웨어 위에서 인식할 수 있는 비트나 바이트를 전송하게 되는 것이다. 하드웨어와 관련한 복잡한 사항들은 아두이노의 시리얼 라이브러리가 대부분 처리하므로 크게 신경쓰지 않아도 된다. 아두이노와의 통신을 통해 주고 받는 정보를 시각적 혹은 의미있게 표현하기 위해서는 시리얼 모니터로는 한계가 있으며, 추가로 시리얼 통신 부분을 프로그래밍할 수 있는 프로그래밍 언어를 통해서 아두이노와 통신을 하게 된다. 아두이노가 보내는 온도 센서의 값을 그래프로 표현하기 위해서는 적합한 언어를 선택하고 해당 프로그램에서 센서의 값을 시각적으로 표현하는 작업을 추가로 해주어야 한다.

기본 명령

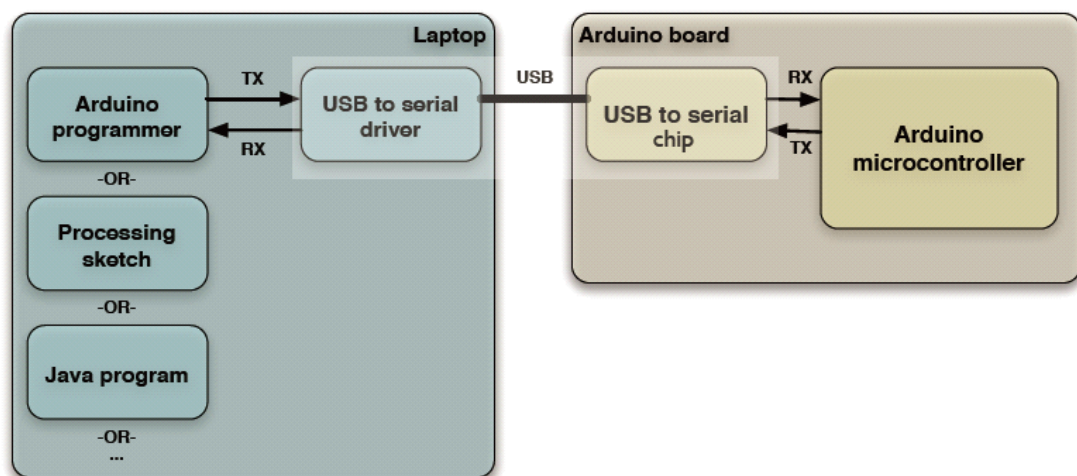
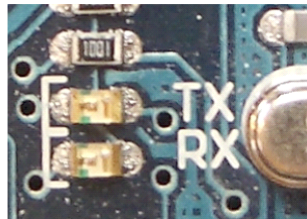
시리얼 통신을 사용하기 위해서 아두이노에 사용되는 기본 명령은 다음과 같다.

Serial.begin() - 시리얼 통신을 사용하기 위한 준비단계

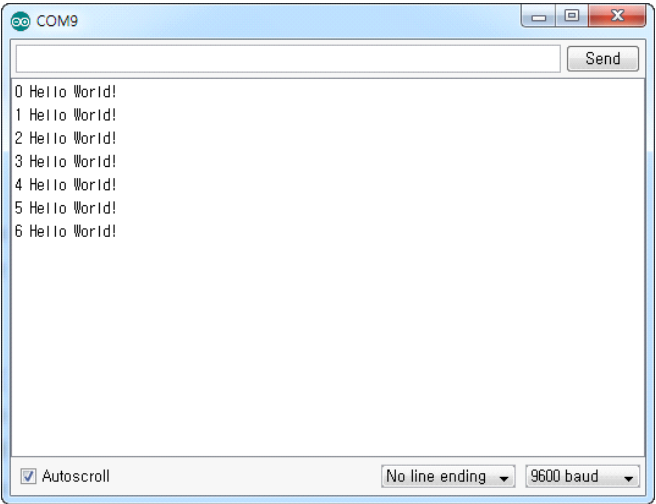
Serial.print() - 시리얼 통신을 통해 컴퓨터로 데이터를 보낸다.

Serial.read() - 시리얼 통신을 통해 컴퓨터로부터 데이터를 읽는다.

- TX – sending to PC
- RX – receiving from PC
- Used when programming or communicating



프로그램 - Serial 출력

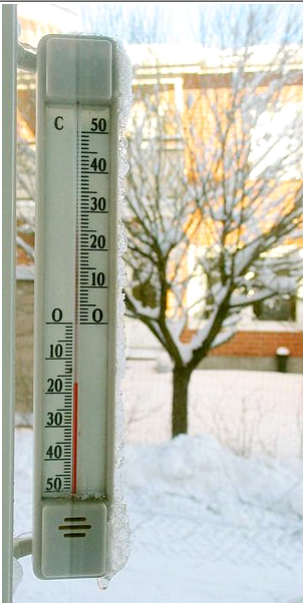

<pre> int led = 13; int i = 0; void setup() { pinMode(led, OUTPUT); Serial.begin(9600); } void loop() { Serial.println(i++); Serial.println(" Hello World!"); digitalWrite(led, HIGH); delay(500); digitalWrite(led, LOW); delay(500); } </pre>	
--	--

프로그램 분석

위 예제코드는 시리얼 포트에 1씩 증가하는 *i* 값과 문자열을 보내며, 13번 포트에 연결된 LED를 점멸하고 있다. 아두이노 보드를 보면 일정한 시간 별로 TX버튼이 점멸하는 것을 볼 수 있으며, 이는 시리얼 통신을 통해 아두이노 보드가 컴퓨터로 자료를 전송하고 있음을 의미한다.

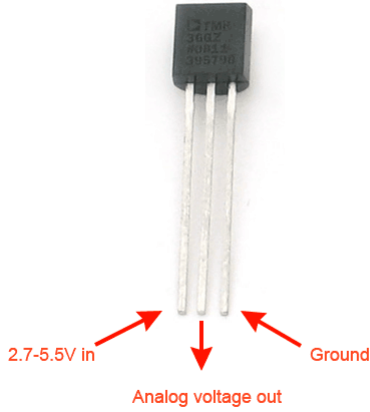
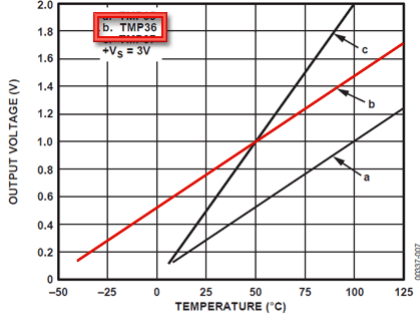
6. 온도 센서

일기예보에서는 내일의 날씨를 나타낼 때 대부분 온도 정보를 함께 보여준다. 온도는 물질의 뜨겁고 찬 정도를 나타내는 것으로서, 일상생활에서 많이 활용이 된다. 온도는 쉘시우스 온도계나 디지털 온도계를 통해 손쉽게 측정이 가능하다. 아두이노에서 편리하게 사용할 수 있는 아날로그 온도 센서를 이용하여 외부의 온도를 측정하고 그 결과를 시리얼 모니터를 통해서 확인해 보자. 이를 통해 연속된 데이터인 아날로그 값을 아두이노가 처리 하는 방식과 온도를 나타내기 위하여 변환 과정을 거치는 것을 이해할 수 있다.

쉘시우스 온도계	디지털 온도계
	

온도 센서

온도 센서(TMP36)는 다음과 같은 구조를 가지며 전원 인가를 위한 핀과, 접지를 위한 핀 그리고 온도를 알 수 있도록 아날로그 전압을 출력해주는 3개의 핀으로 구성되어 있다.

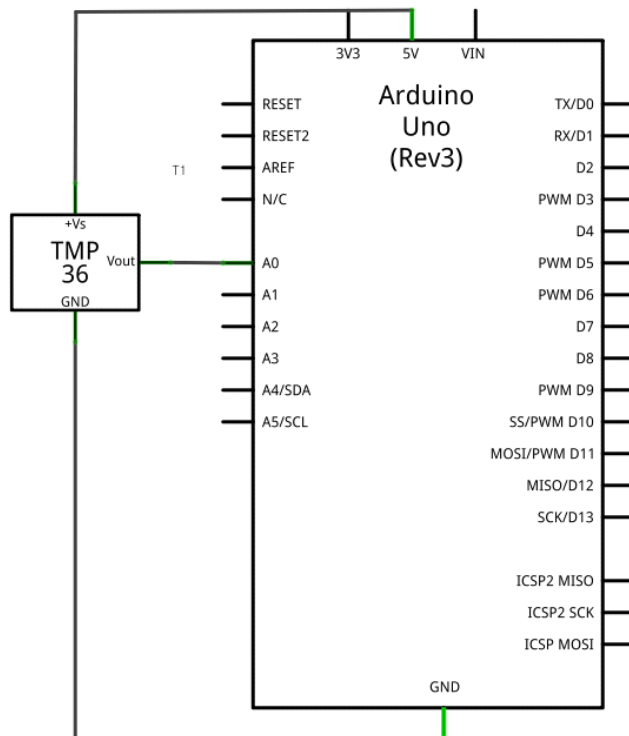
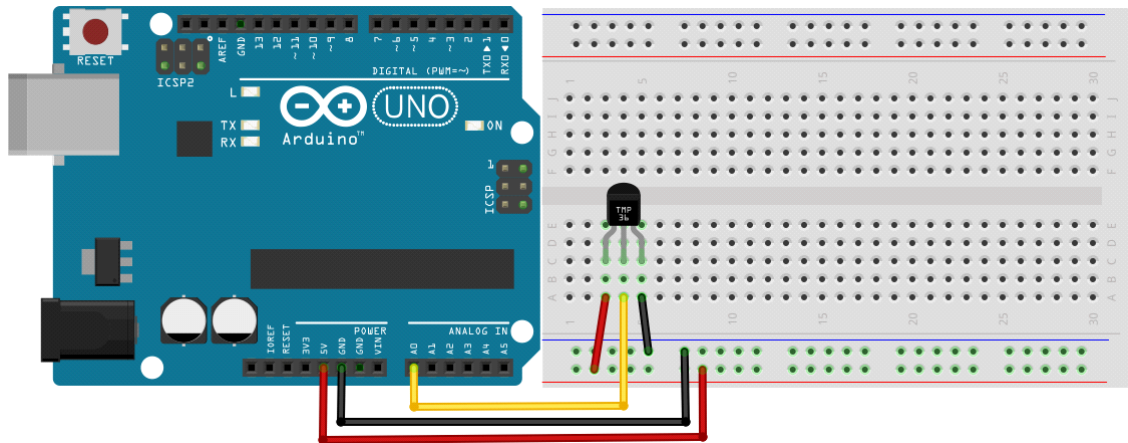
TMP36	온도와 전압 특성
	 <p>Figure 6. Output Voltage vs. Temperature</p>

TMP36은 비교적 저렴한 아날로그 온도 센서로서 추가적인 회로를 구성하지 않고도 간편하게 온도를 측정할 수 있다. TMP36은 섭씨 $-40^{\circ}\text{C}\sim 150^{\circ}\text{C}$ 까지의 온도를 측정할 수 있으며 분해능은 $10\text{mV}/^{\circ}\text{C}$ 이며 편리하게 온도 측정이 가능하다. 오차는($\pm 2^{\circ}\text{C}$) 정도가 발생한다. TMP36은 아래와 같은 특성을 지니며, 온도가 올라갈수록 전압이 증가하며, 섭씨 50도에서 1.0V 전압이 측정된다.

연속된 아날로그 신호를 아두이노 내부로 입력하는 과정에서 아날로그 데이터는 디지털로 변환이 일어나며, 아두이노의 아날로그 입력은 입력 값을 0~1023의 디지털 값으로 변환해서 보여주는 10bit 분해능을 갖추고 있다. 따라서, 입력 값을 기준 전압인 5V의 상대적인 위치값으로 변환을 한 다음 데이터 시트상에서 온도를 측정하기 위해 Offset 값의 조정을 통해 원하는 온도 값을 알아낼 수 있게 된다.

브레드 보드

VCC(왼쪽)는 아두이노의 5V에, GND(오른쪽)는 아두이노의 GND에 그리고 신호선(가운데))은 아두이노의 아날로그 0번 핀에 연결한다.



프로그램 1

스케치를 아두이노 IDE에서 작성하고 Upload 한다.

01	int sensorPin = 0;
02	void setup()
03	{
04	Serial.begin(9600);
05	}
06	void loop()
07	{
08	int reading = analogRead(sensorPin);
09	float voltage = reading * 5.0;
10	voltage /= 1024.0;
11	float temperatureC = (voltage - 0.5) * 100 ;
12	Serial.println(temperatureC);
13	delay(1000);
14	}

[스케치 설명]

01 센서가 연결되어 있는 아날로그 포트인 0번 포트를 위한 변수 생성

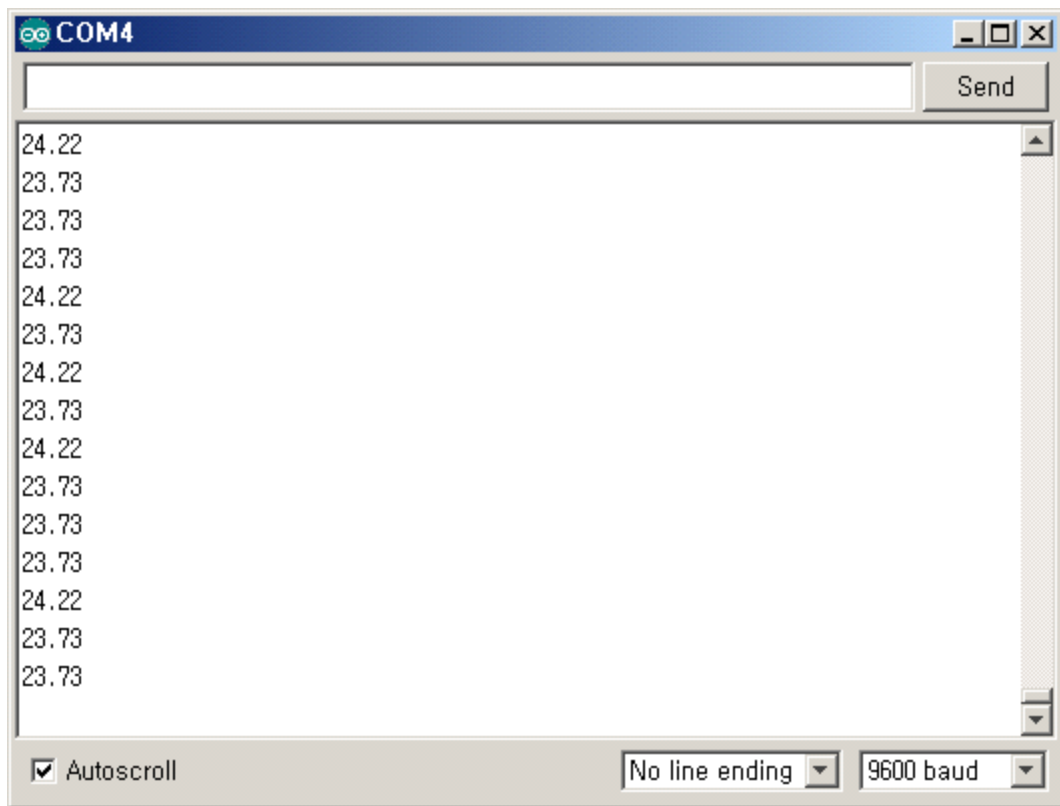
04 시리얼 통신을 위한 기본 속도를 설정해 준다.

08 A0에 들어오는 아날로그 값을 읽기 위한 함수

09-11 측정된 전압을 온도로 변환시키는 작업

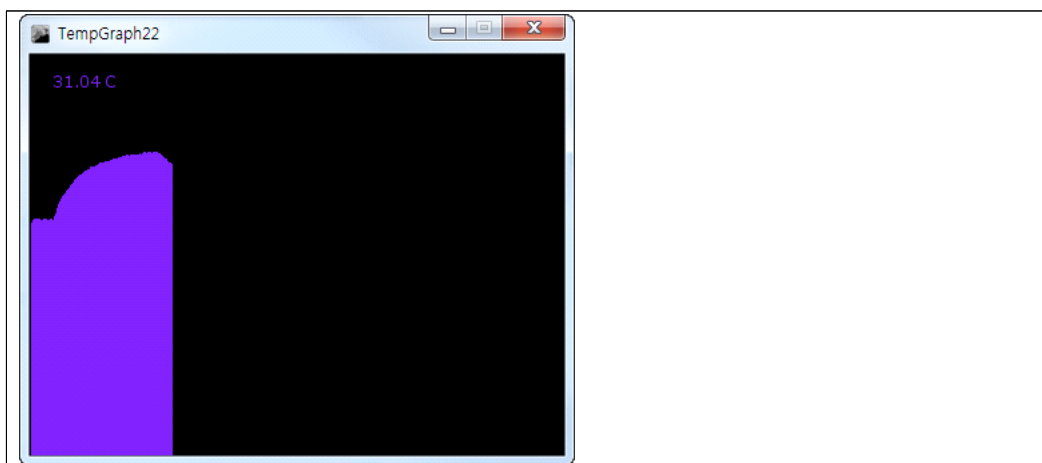
12 측정된 온도를 시리얼을 통해 컴퓨터로 전송

13 1초간 지연 시킨 후에 다시 loop 함수가 시작되도록 해준다.



프로그램 2

시리얼을 통한 온도 입력자료를 시각적으로 표현하기 위해 프로세싱을 이용하여 그래프로 표현한다.



```

import processing.serial.*;

Serial myPort;
int xPos = 1;
void setup () {
    size(400, 300);
    println(Serial.list());
    myPort = new Serial(this, Serial.list()[1], 9600);
    myPort.bufferUntil('\n');
    background(0);
}
void draw () {
}

void serialEvent (Serial myPort) {
    String inString = myPort.readStringUntil('\n');

    if (inString != null) {
        inString = trim(inString);
        float inByte = float(inString);
        fill(0);
        noStroke();
        rect(0, 0, 70, 50);
        fill(127, 34, 255);
        textFont(font12);
        textAlign(RIGHT);
        text(inString + " C" , 65, 25);
        stroke(127,34,255);
        line(xPos, height, xPos, height - inByte*7);
        if (xPos >= width) {
            xPos = 0;
            background(0);
        }
        else {
            xPos++;
        }
    }
}

```