아두이노프로그래밍

5차과제

8장5절일부 연습문제 풀이

2020.05.21.목

컴퓨터공학과

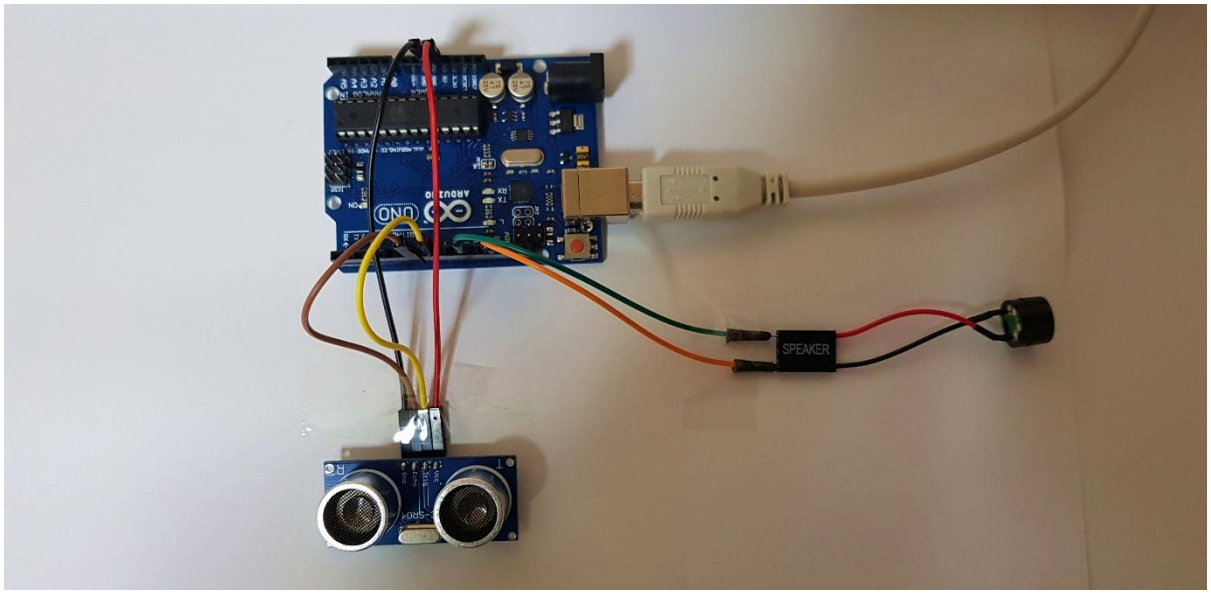
2019305059

이현수

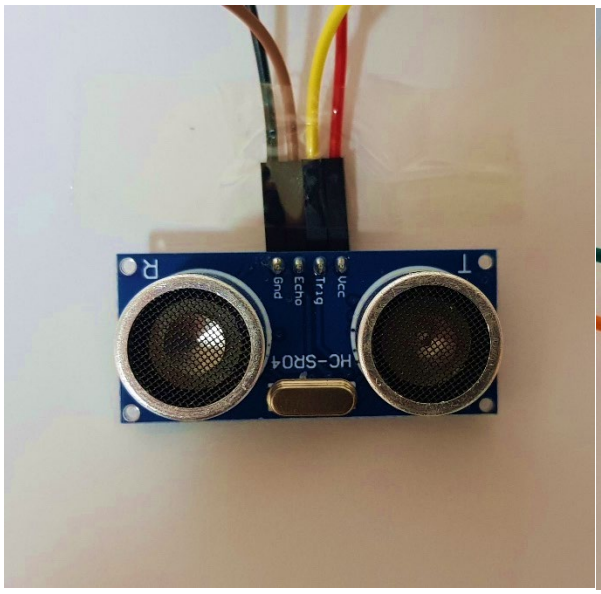
▪4번

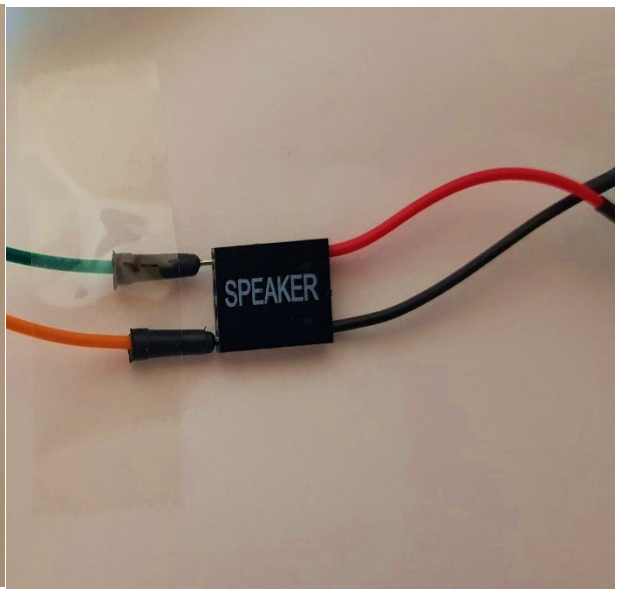
- 스피커를 연결하여 센서 앞의 장애물을 움직여 연주하는 프로그램을 만들기
- 이를 시연하는 동영상을 제작.
- 특정 범위(예를 들어 10cm~40cm)를 넘어가면 스피커에 대해 noTone()을 수행하여 소리를 나지 않게 함.

▪회로도

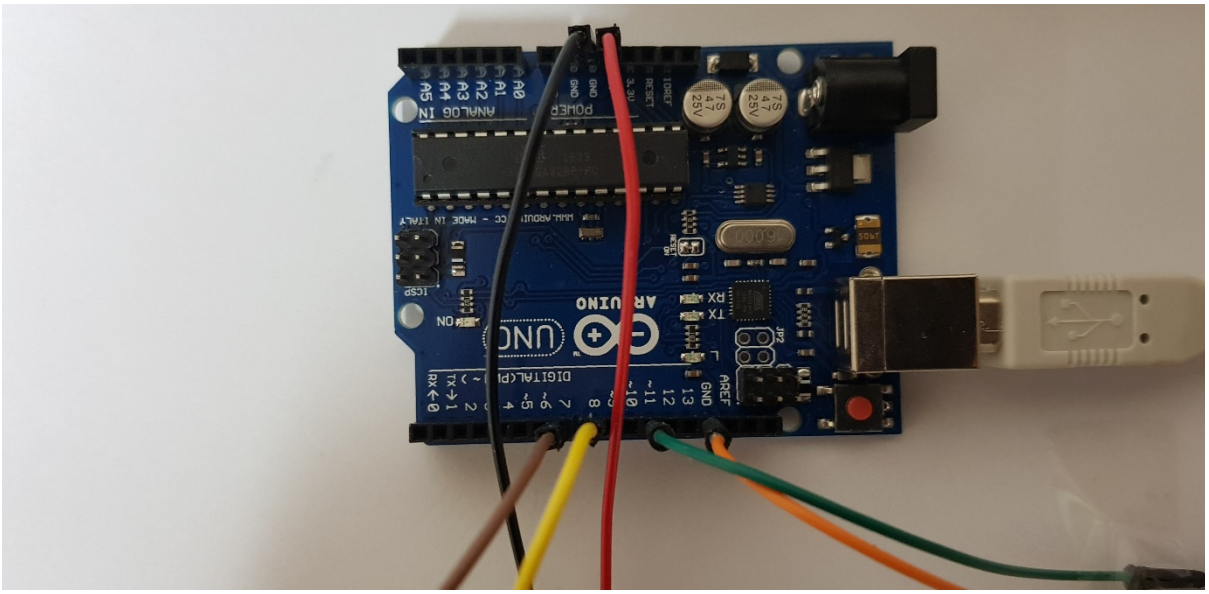


아두이노 우노보드에 초음파센서, 스피커를 연결한다.





초음파센서의 GND는 검정 / Echo는 갈색 / Trig는 노랑 / Vcc는 빨간색 점퍼케이블로 연결한다.
스피커의 (+)는 초록 / (-)는 주황색 점퍼케이블로 연결한다.



초음파센서의 GND(검정)은 GND, Echo(갈색)은 7번, Trig(노랑)은 8번, Vcc(빨강)은 5V에 연결한다.
스피커의 (+)(초록)은 11번, (-)(주황)은 GND에 연결한다.

▪소스코드

```
#define echoPin 7
#define trigPin 8
#define Temperature 27
long duration, distance;
double sSpeed_cm_us;
int SPK=11;
```

echoPin을 7로 정의하고, trigPin을 8로 정의한다. Temperature 온도를 27로 정의한다.

long형 duration, distance를 전역변수로 선언한다.

double형 sSpeed_cm_us를 선언한다.

그리고 스피커 GPIO단자가 11번이므로 SPK를 11번으로 초기화한다.

```
void setup(){
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  sSpeed_cm_us=(331.5+0.6*Temperature)/pow(10,4);
}
```

Setup() 함수에는 통신속도를 9600으로 설정하고,
핀모드를 trigPin을 출력으로 echoPin을 입력으로 설정한다.

velocity_of_sound[m/s] = (331.5 + 0.6*Temperature)
velocity_of_sound[m/us] = (331.5 + 0.6*Temperature) / 10^6
velocity_of_sound[cm/us] = (331.5 + 0.6*Temperature) * 10^2 / 10^6
velocity_of_sound[cm/us] = (331.5 + 0.6*Temperature) / 10^4

sSpeed_cm_us 는 소리 속도를 cm/us로 계산한 것이다. 위 공식으로 계산식을 만든다.

```
void loop(){
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance=(duration*sSpeed_cm_us)/2;
```

loop() 함수에서 digitalWrite(trigPin, LOW); delayMicroseconds(2);를 해줘 오류를 방지한다.

그리고 Trig 단자에 10uS의 펄스(pulse) 신호를 인가하여 센서에 초음파 발생을 지시한다.
이때 코드는 digitalWirte(trigPin, HIGH); delayMicroseconds(10); digitalWrite(trigPin, LOW); 코드이다.
tringPin에 H로 만든 후 10uS 동안 유 지한 후 L로 만들어주면 된다.

그 후 duration에 pulseIn()함수를 이용해 Echo 단자로 유입된 펄스의 H 부분이 유지된 시간을 측정한다.

(1) distance[m] = (high_width_time[s] × velocity_of_sound[m/s]) / 2
(2) distance[m] = (high_width_time[us] × velocity_of_sound[m/s]) / (2 * 10^6)
(3) distance[cm] = 10^2 * (high_width_time[us] × velocity_of_sound[m/s]) / (2 * 10^6)
(4) distance[cm] = (high_width_time[us] × velocity_of_sound[cm/us]) / 2

그 후 거리공식을 통해 실제거리를 구하고 그값을 distance변수에 저장한다.

```
  if(distance>40||distance<10){
    noTone(SPK); Serial.println("0");
    delay(50);
  }
  if(10<=distance&&distance<=14){
    tone(SPK, 262); Serial.println("1 도");
    delay(50);
  }
  else if(14<distance&&distance<=18)
  {
    tone(SPK, 294); Serial.println("2 레");
    delay(50);
  }
  else if(18<distance&&distance<=22)
  {
    tone(SPK, 330); Serial.println("3 미");
    delay(50);
  }
  else if(22<distance&&distance<=26)
  {
    tone(SPK, 347); Serial.println("4 파");
    delay(50);
  }
  else if(26<distance&&distance<=30)
  {
    tone(SPK, 392); Serial.println("5 솔");
    delay(50);
  }
  else if(30<distance&&distance<=34)
  {
    tone(SPK, 440); Serial.println("6 라");
    delay(50);
  }
  else if(34<distance&&distance<=37)
  {
    tone(SPK, 494); Serial.println("7 시");
    delay(50);
  }
  else if(37<distance&&distance<=40)
  {
    tone(SPK, 524); Serial.println("8 도");
    delay(50);
  }
}
```

이후 조건문을 사용해서 최대값 40cm, 최소값 10cm 범위를 벗어나면 noTone()함수를 통해 스피커를 꺼준다. 그리고 “0”을 출력한다.

그 이외에 10~40cm 범위를 적절히 분배해 낮은도~높은도까지 주파수를 tone()함수를 통해서 출력한다. 그리고 ‘1~8’, ‘도~도’를 출력한다.

각각의 조건문에 delay(50);를 통해 잡음과 초음파센서의 오류를 방지한다.

▪실행

실행을 시키면 테라텀 화면에 계속해서 해당되는 숫자와 음이 출력된다. 장애물이 나타나면 해당 범위의 주파수는 스피커를 통해서 출력된다.

▪실험과정에서의 경험과 습득 사실

- 연주를 할 때 작은 수첩이나 손으로 하는것보다 A4용지 정도의 크기 물체를 장애물로 사용해 인식시키는게 편하다.

- 소리속도, 거리공식을 알게됨.

▪5번

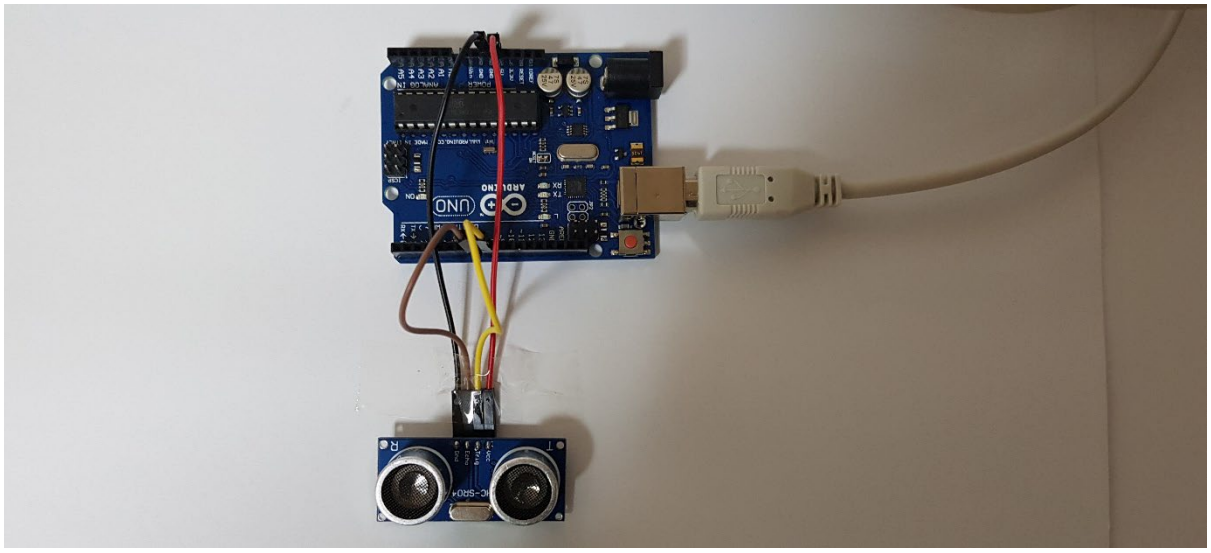
- 장애물까지의 거리를 mm 단위로 반환하는 함수를 다음과 같이 제작.

1) digitalWrite(), digitalRead(), pulseIn() 함수 등의 표준 함수를 사용하여 위 조건을 만족하는 함수, Read_mm_Dist(echoPin, trigPin, Temperature)를 제작.

2) 다른 표준함수는 사용하지만, pulseIn() 표준함수를 사용하지 않고 동일한 동작을 하는 자체 MyPulseIn() 함수를 설계하고 이를 기반으로 제시된 함 수 Read_mm_Dist2()를 설계.

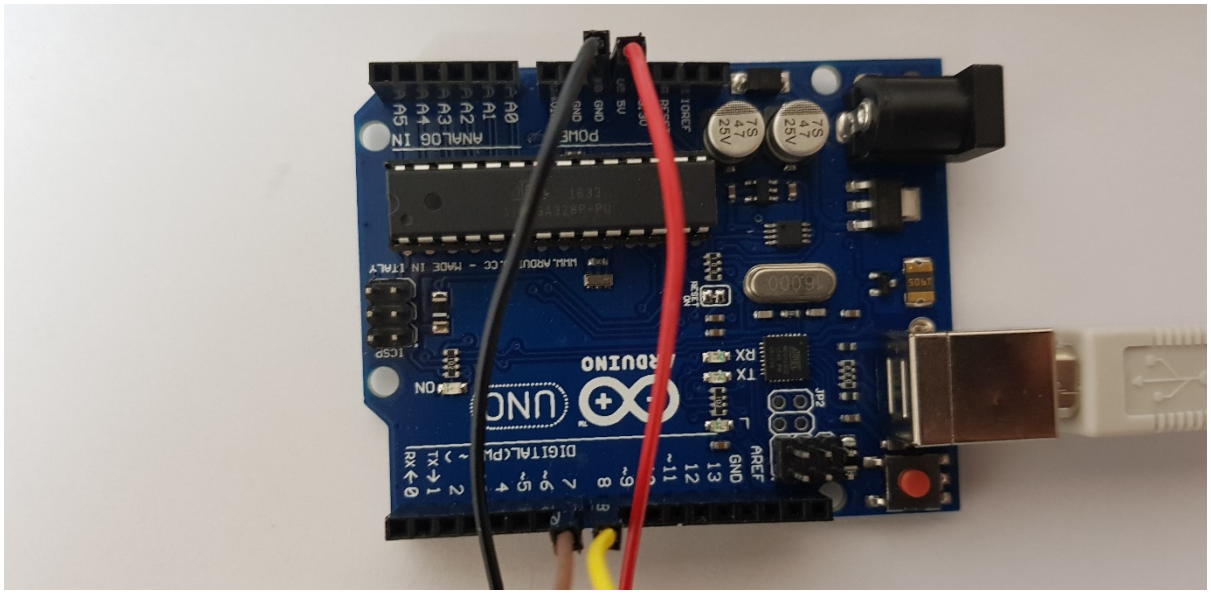
3) Read_mm_Dist()와 Read_mm_Dist2()의 거리 측정 오차가 어느 정도(%)인 지 실험을 통해 비교. 다음 관점에서 분석. ①10회 이상의 평균을 이용하여 산출 ②micros() 함수의 평균오차와 비교

▪회로도



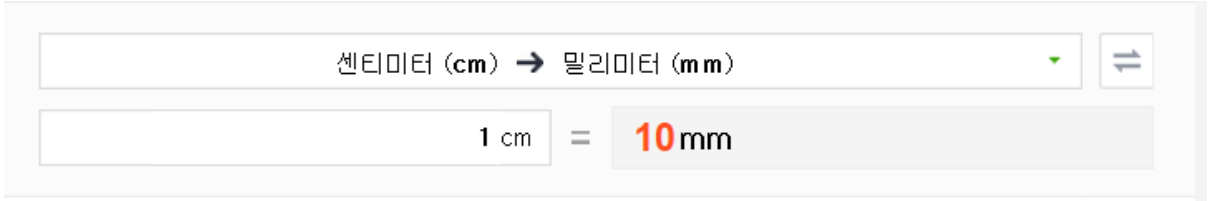
아두이노 우노보드에 초음파센서를 연결한다.

초음파센서의 GND는 검정, Echo는 갈색, Trig는 노랑, Vcc는 빨간색 점퍼케이블로 연결한다.



초음파센서의 GND(검정)은 GND, Echo(갈색)은 7번, Trig(노랑)은 8번, Vcc(빨강)은 5V에 연결한다.

▪소스코드



문제에서 mm로 나타내라고 했다. 1cm는 10mm이고, 10cm는 100mm이고, 100cm는 1000mm이다.

```
#define echoPin 7
#define trigPin 8
#define Temperature 27

long Read_mm_Dist(int echopin, int trigpin, int temperature);
long Read_mm_Dist2(int echopin, int trigpin, int temperature);
unsigned long Mypulseln();
```

echoPin을 7로 정의하고, trigPin을 8로 정의한다. Temperature 온도를 27로 정의한다.

1) 문제 함수 선언

long Read_mm_Dist(int echopin, int trigpin, int temperature);

2) 문제 함수 2개 선언

long Read_mm_Dist2(int echopin, int trigpin, int temperature);
unsigned long Mypulseln();

```
void setup(){
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}
```

setup()함수에는 통신속도를 9600으로 설정하고
핀모드를 trigPin을 출력으로 echoPin을 입력으로 설정한다.

```
void loop(){
  long distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  //distance=Read_mm_Dist(echoPin, trigPin, Temperature);
  distance=Read_mm_Dist2(echoPin, trigPin, Temperature);
  Serial.print(distance);
  Serial.println("mm");
  delay(50);
}
```

loop()함수에는 digitalWrite(trigPin, LOW); delayMicroseconds(2);를 해줘 초기 오류를 방지한다.

그리고 long형 변수 distance에 long Read_mm_Dist(); long Read_mm_Dist2(); 함수를 이용해 값을 받아온다.

그리고 distance를 출력하고, "mm"단위를 출력하고 다음줄로 넘어간다.
그리고 초음파센서의 오류 등을 방지하기 위해 delay(50); 코드를 넣는다.

---

1)문제 함수

```
long Read_mm_Dist(int echopin, int trigpin, int temperature){
  long duration;
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);

  duration = pulseIn(echopin, HIGH);
  return (331.5 + 0.6*temperature) * ((float)duration/1000000/2) * 1000;

}
```

Trig 단자에 10uS의 펄스(pulse) 신호를 인가하여 센서에 초음파 발생을 지시한다.
이때 코드는 digitalWirte(trigPin, HIGH); delayMicroseconds(10); digitalWrite(trigPin, LOW); 코드이다.
tringPin에 H로 만든 후 10uS 동안 유 지한 후 L로 만들어주면 된다.

그 후 duration에 pulseIn()함수를 이용해 Echo 단자로 유입된 펄스의 H 부분이 유지된 시간을 측정한다.

그리고 실제거리를 구하기 위해 (331.5 + 0.6*temperature) * ((float)duration/1000000/2) * 1000 식을 사용 후 값을 반환한다.

2)문제 함수

```
long Read_mm_Dist2(int echopin, int trigpin, int temperature){
  long duration;
  digitalWrite(trigpin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigpin, LOW);

  duration=MypulseIn();
  return (331.5 + 0.6*temperature) * ((float)duration/1000000/2) * 1000;
}
```

Trig 단자에 10uS의 펄스(pulse) 신호를 인가하여 센서에 초음파 발생을 지시한다.
이때 코드는 digitalWirte(trigPin, HIGH); delayMicroseconds(10); digitalWrite(trigPin, LOW); 코드이다.
tringPin에 H로 만든 후 10uS 동안 유 지한 후 L로 만들어주면 된다.

그 후 duration변수는 MypulseIn() 함수를 통해 Echo 단자로 유입된 펄스의 H 부분이 유지된 시간을 받아온다.

그 후 실제거리를 구하기 위해 (331.5 + 0.6*temperature) * ((float)duration/1000000/2) * 1000 식을 사용 후 값을 반환한다.

```
unsigned long MypulseIn(){
  unsigned long Start, End,t;
  while(1){
   if(digitalRead(echoPin)==HIGH){
     Start=micros();break;
   }
  }
  while(1){
   if(digitalRead(echoPin)==LOW){
    End=micros(); break;
   }
  }
  t = End-Start;
  return t;
}
```

MypulseIn() 함수를 살펴보면 unsigned long Start, End,t;를 선언하고
while문에 if(digitalRead(echoPin)==HIGH)사용해 echoPin에 HIGH가 들어온 순간 Start변수에 micros()함수를 통해서 자체시간을 us단위로 받아들이고 반복문을 빠져나오고,
다음 while문에 if(digitalRead(echoPin)==LOW)를 통해 echoPin이 LOW가 될 때 End변수에 micros()함수를 사용해 자체시간을 us단위로 받아들인다.
그리고 t=End-Start를 하면 Echo 단자로 유입된 펄스의 H 부분이 유지된 시간이 변수 t에 저장되고, 변수 t를 반환한다.

3) 거리측정 오차

```
///*(3)번문제
long distance;
int n=1;
long sum1=0, sum2=0;
long average1 = 0, average2 = 0;
double percent;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
while(n<=10){
  distance=Read_mm_Dist(echoPin, trigPin, Temperature);
  sum1+=distance;
  n++;
  delay(50);
}
n=1;
while(n<=10){
  distance=Read_mm_Dist2(echoPin, trigPin, Temperature);
  sum2+=distance;
  n++;
  delay(50);
}
```

거리 측정을 10회 실행 후 평균을 구한다.

while반복문을 이용해 (1)번함수를 통해 distance에 거리값을 저장후 sum에 누적 저장한다.

(2)번함수 역시 while반복문을 이용해 distance를 sum에 누적 저장한다.

```
average1=sum1/10;
Serial.print("(1)번함수 10회평균 = ");
Serial.print(average1);Serial.println("mm");
average2=sum2/10;
Serial.print("(2)번함수 10회평균 = ");
Serial.print(average2);Serial.println("mm");
percent=(((double)average2-(double)average1)/(double)average2)*100.0;
Serial.print(percent); Serial.println("%");
Serial.println("------------");
//*/
```

그리고 (1)번함수와 (2)번함수의 각각 평균을 구하고 출력한다.

그리고 거리측정 오차를 구하기 위해 double형 percent변수에 거리측정 오차를 %로 구해주고 출력한다.

▪실행

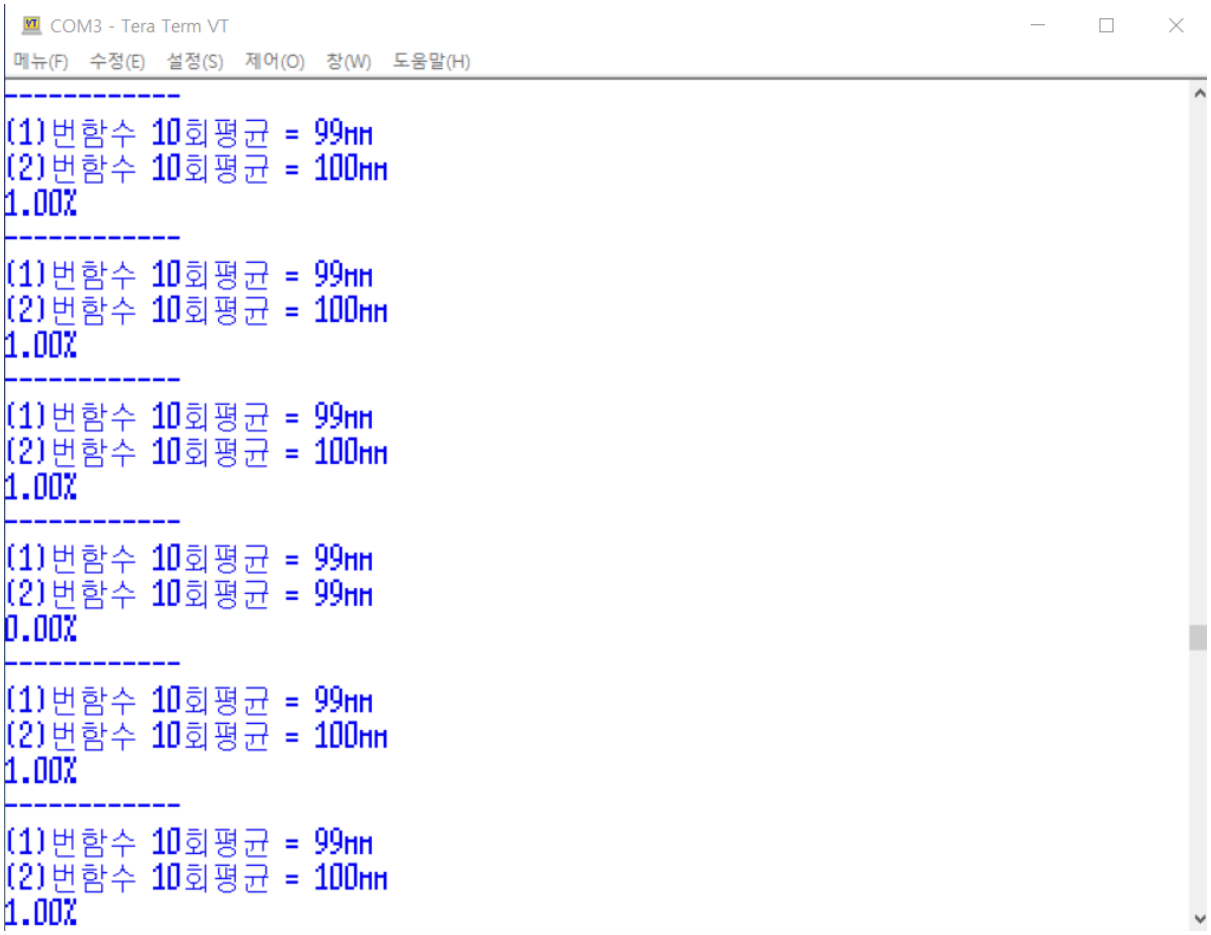
1) 문제 함수

거리가 정상적으로 mm단위로 출력된다. 107mm 는 10.7cm를 의미한다.

2) 문제함수

거리가 정상적으로 mm단위로 출력된다. 168mm 는 16.8cm를 의미한다.

3)거리측정 오차 – 같은 거리의 고정된 같은 장애물 측정



(1)번 함수를 10번 돌려서 고정된 장애물의 거리를 얻은 값의 평균과 (2)번 함수를 10번 돌려서 (1)번함수와 같은거리에 고정된 장애물의 거리를 얻은 값의 평균을 출력했다.
그리고 loop()함수로 계속 반복되 출력된다.

오차가 1%정도로 (2)번함수가 조금 더 시간이 걸리는 것을 확인할 수 있다.

micros()함수의 평균오차를 고려했을 때 저정도의 오차가 발생하는건 정상적이다.

▪실험과정에서의 경험과 습득 사실

- micros()함수를 사용해 pulseIn()함수를 구현할 수 있다는 사실을 알게됨.
- 거리 측정 후 delay()함수를 사용하지 않았을 때 오차가 매우 크게 나타났는데, delay(50);을 사용하고 나서 오차가 굉장히 많이 줄어들었다.