임베디드컴퓨팅
Embedded Computing
(0009488)
Digital Output, part 2

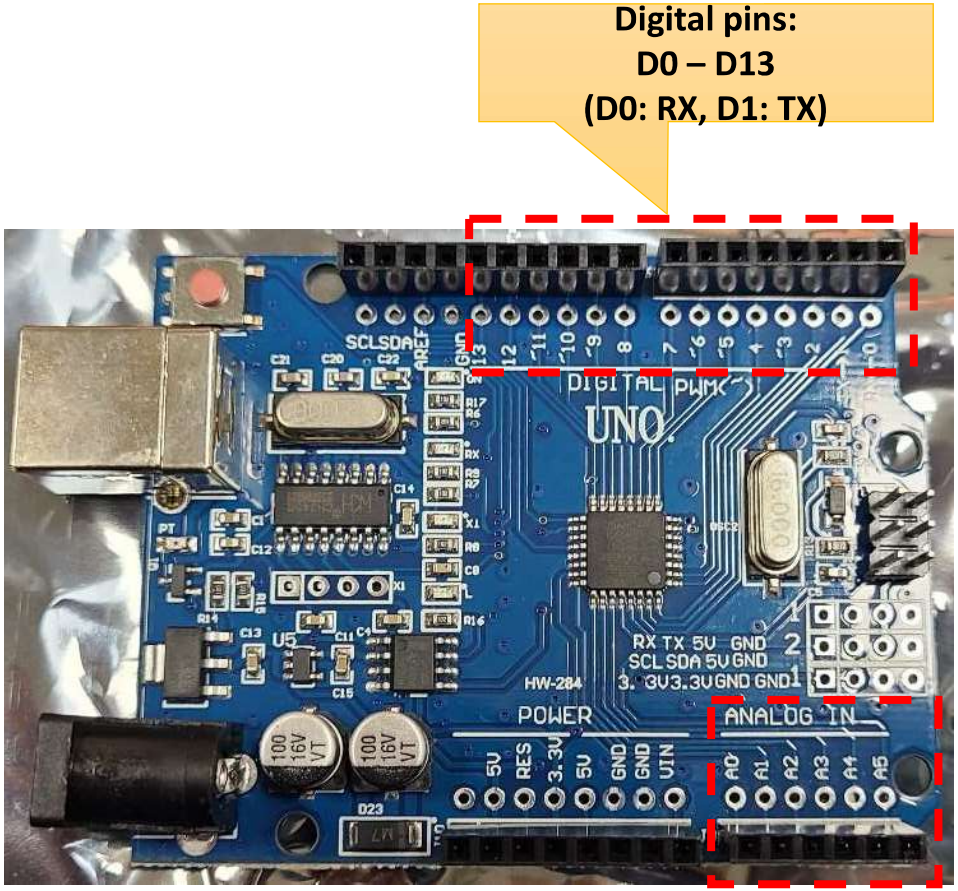
2022년 2학기

정보기술대학 정보통신공학과
김 영 필
ypkim@inu.ac.kr

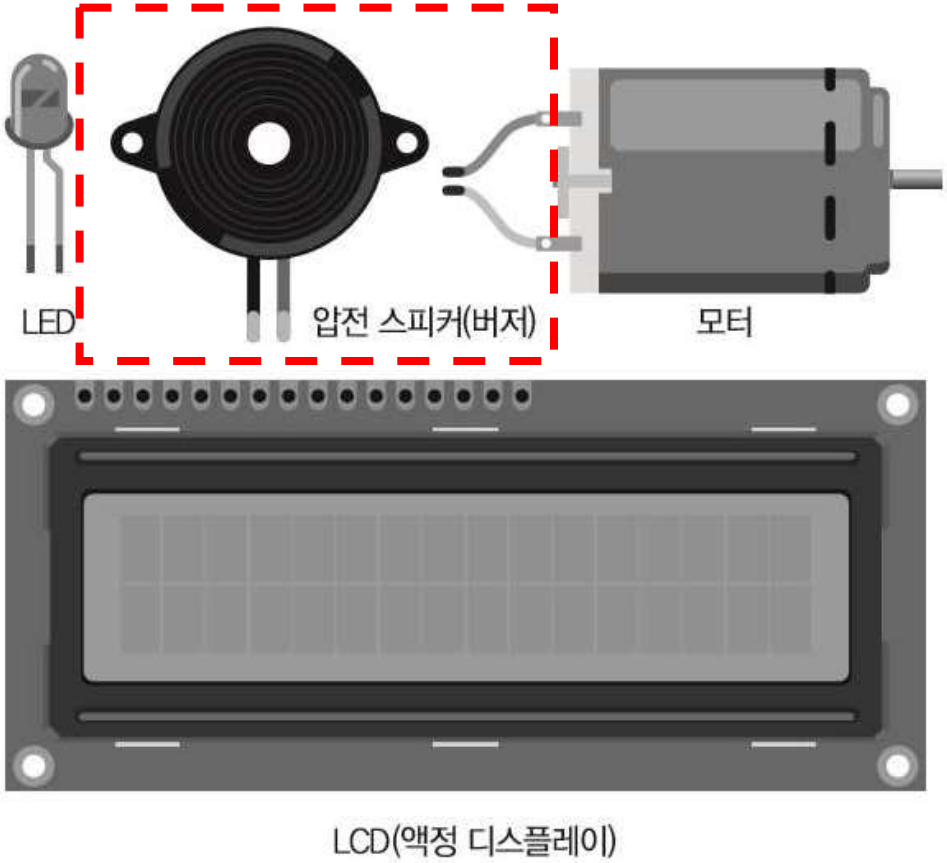
인천대학교
INCHEON NATIONAL UNIVERSITY

Reminder: How to measure signal related value in Arduino?

- Need to control, I/O pins (analog/digital) in Arduino
- How to control?
  - Use built-in functions
    - analogWrite(), analogRead()
    - pinMode(), digitalWrite(), digitalRead()
- What to control?
  - LED
- Need to determine control method
  - Analog or Digital



Reminder:
Output devices for Arduino



출력 방법	아날로그 출력	디지털 출력
사용하는 함수	analogWrite	pinMode와 digitalWrite
사용하는 전자 부품	• LED, 스피커 • 팬 • 일부 모터* 등	• LED, 스피커 • 적외선 리모컨용 LED 등

* 일반 모터는 아날로그로 제어할 수도 있고 디지털로 제어할 수도 있다.

그림 Source: 길벗, "모두의 아두이노"

Piezoelectric Speaker (buzzer)

Piezoelectric speaker

- A loudspeaker that uses the piezoelectric effect for generating sound.
  - Known as a piezo bender,

    piezo, buzzer, crystal

    loudspeaker, beep speaker
- Piezo speaker supports both analog and digital output

The prefix piezo- is Greek for 'press' or 'squeeze'

Ref. - https://en.wikipedia.org/wiki/Piezoelectric_speaker

How to make sound?

- In piezoelectric speakers, the **vibration of the membrane produces sound**.
- **When the interval between HIGH and LOW** is **approximately equal, vibration occurs** and a sound is produced.
- For analog output, Arduino uses a Pulse Width Modulation(PWM)
  - This is a simulated analog output in a digital way.

PWM (pulse width modulation)

- The analog output of Arduino adjusts the voltage by changing the **period of digital HIGH and LOW**.
- PWM using ***analogWrite*** function changes the voltage between HIGH and LOW at a frequency of **490Hz**
  - HIGH and LOW are switched at intervals of **1/490 second**.

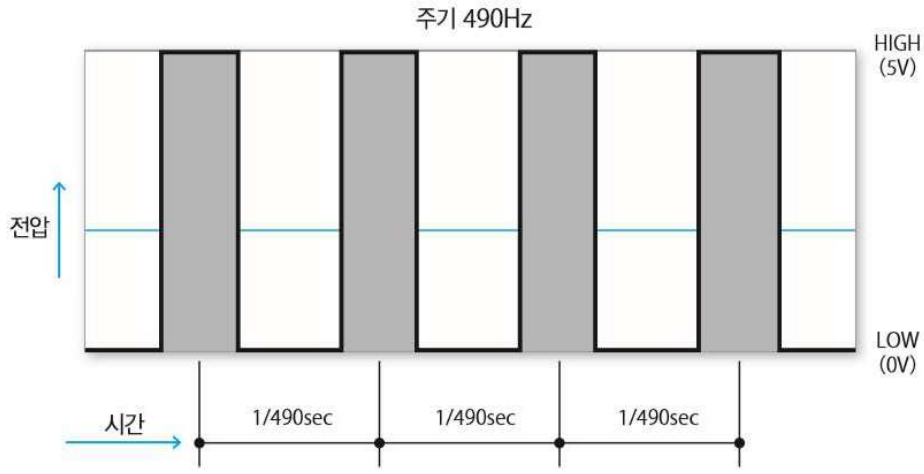


그림 Source: 길벗, "모두의 아두이노"

Let’s try Analog output

- Connect pins of Piezo speaker into GND and D11 (one of ~PWM pins)
- Try the below sketch code

```
void setup()
{
}

void loop() {
    analogWirte(11, 255 / 2);
}
```

What happens?

PWM waves by analogWrite

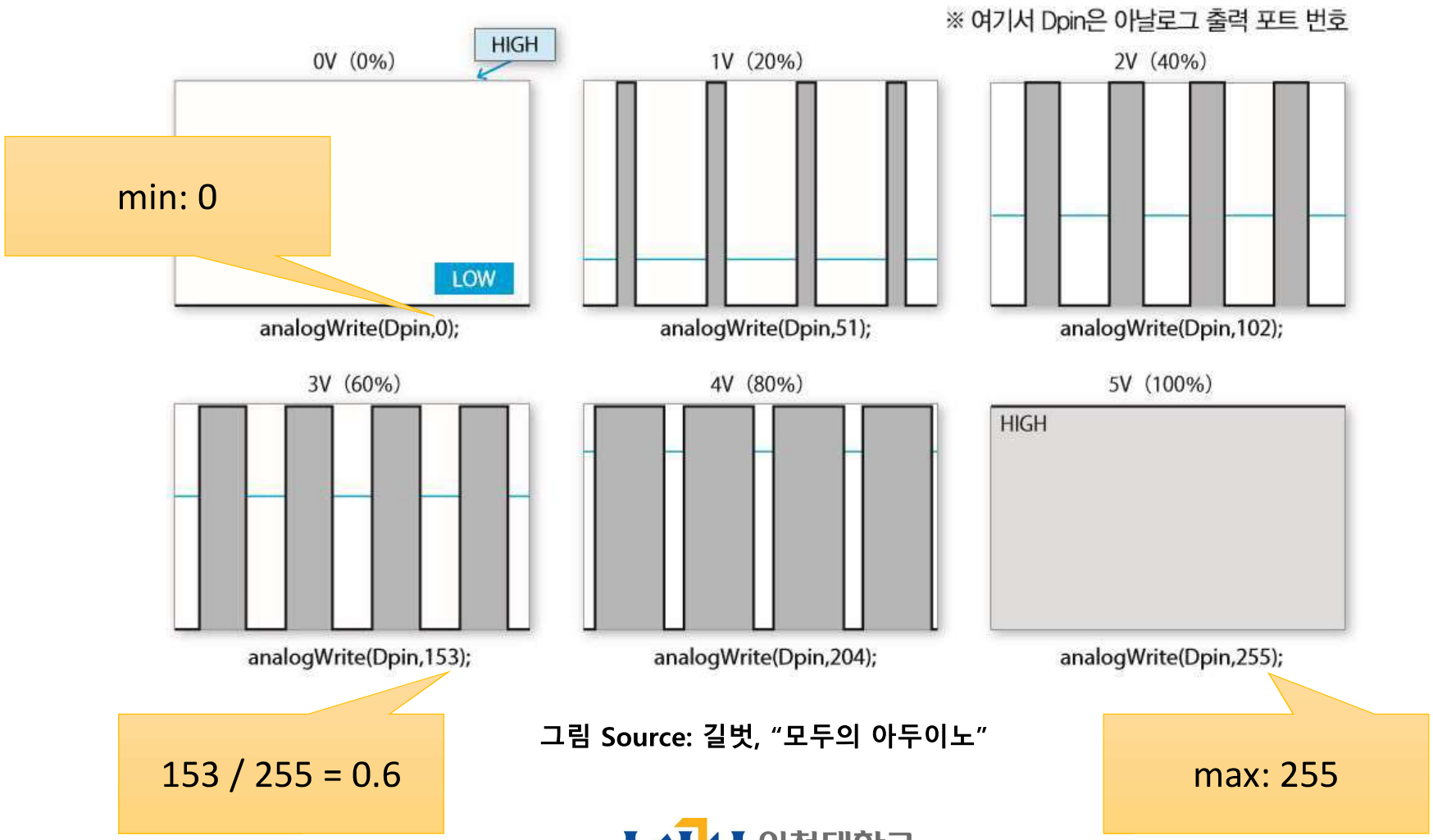


그림 Source: 길벗, "모두의 아두이노"

Arduino Setup for Digital Output for Piezo

- Try to make sound from speaker with digital control using **pinMode()** and **digitalWrite()** function
- Connect the two cables of the piezoelectric speaker to **D12 and GND**
- Piezoelectric speakers have no polarity
  - don't have to care about +/-

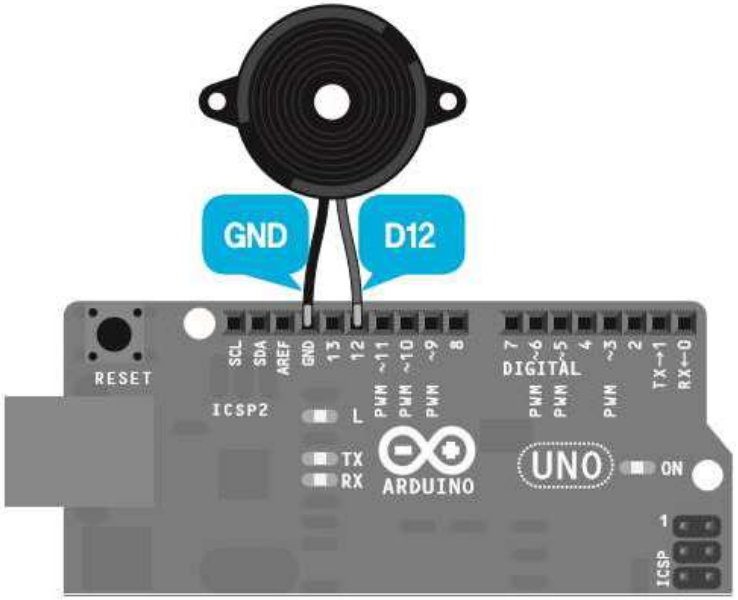


그림 Source: 길벗, "모두의 아두이노"

Sketch for Piezo: Alarm

```
#define DX 12
int delay_time = 2;
void setup()
{
    pinMode(DX, OUTPUT);
}
void loop()
{
    run_alarm();
    delay(500);
}
void run_alarm() {
  for(int i = 0 ; i < 10 ; i++){
    digitalWrite(DX, HIGH);
    delay(delay_time);
    digitalWrite(DX, LOW);
    delay(delay_time);

  }
}
```

one cycle becomes 4ms(2ms HIGH + 2ms LOW), and when calculated as 1 second, it becomes 250Hz (=1000/4)

Let's change delay time (ms) :

2, 10, 100, 1000

the time (period) of one cycle of HIGH and LOW is equal to make a sound

Let's try to make both delay_time different. What happens?

How to change scale?

- Control the **loudness of the speaker sound** with **periodic** Vibration
- We need to write a function that makes the scale sound.
- To make a sound with a period of m for 1 second, the speaker membrane **repeats On and Off m times per second**
  - On = up and down signal
  - Off = down and up signal
- The original speaker sounds as **a** Sine wave, but in this sketch it sounds as **a** Square wave

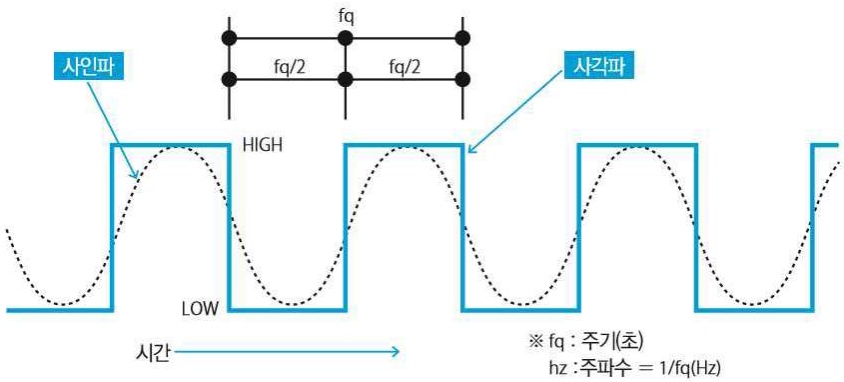


그림 Source: 길벗, "모두의 아두이노"

Frequency (Hz) for scale

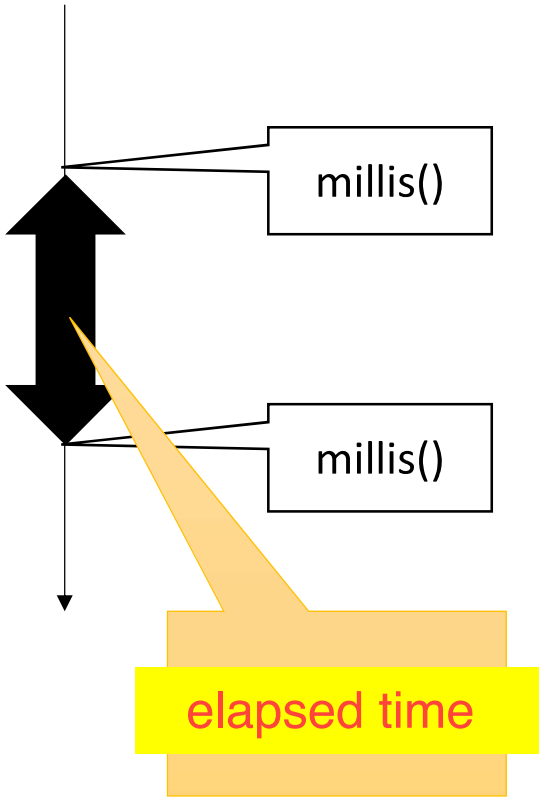
음계	3	4	5	6	7	8	9
B	247	494	988	1976	3951	7902	
A#	233	466	932	1865	3729	7459	
A	220	440	880	1760	3520	7040	14080
G#		415	831	1661	3322	6645	13290
G		392	784	1568	3136	6272	12544
F#		370	740	1480	2960	5920	11840
F		349	698	1397	2794	5588	11176
E		330	659	1319	2637	5274	10548
D#		311	622	1245	2489	4978	9956
D		294	587	1175	2349	4699	9397
C#		277	554	1109	2217	4435	8870
C		262	523	1047	2093	4186	8372

* 이 음계는 도: C, 레: D, 미: E, 파: F, 솔: G, 라: A, 시: B이다.

그림 Source: 길벗, "모두의 아두이노"

millis()

- Returns the number of milliseconds passed since the Arduino board began running the current program.
  - Overflow after approximately 50 days.
- Syntax
  - time = millis()
- Parameters
  - None
- Returns
  - Number of milliseconds passed since the program started. **(unsigned long)**.



Ref. - https://www.arduino.cc/reference/en/language/functions/time/millis/

delayMicroseconds()

- Pauses the program for the amount of time (in microseconds) specified by the parameter.
  - There are a thousand microseconds in a millisecond and a million microseconds in a second.
  - Currently, the largest value that will produce an accurate delay is 16383; larger values can produce an extremely short delay.
- Syntax
  - delayMicroseconds(us)
- Parameters
  - us: the number of microseconds to pause. Allowed data types: unsigned int.
- Returns
  - Nothing

```
int outPin = 8;             // digital pin 8

void setup() {
  pinMode(outPin, OUTPUT);    // sets the digital pin as output
}

void loop() {
  digitalWrite(outPin, HIGH); // sets the pin on
  delayMicroseconds(50);      // pauses for 50 microseconds
  digitalWrite(outPin, LOW);  // sets the pin off
  delayMicroseconds(50);      // pauses for 50 microseconds
}
```

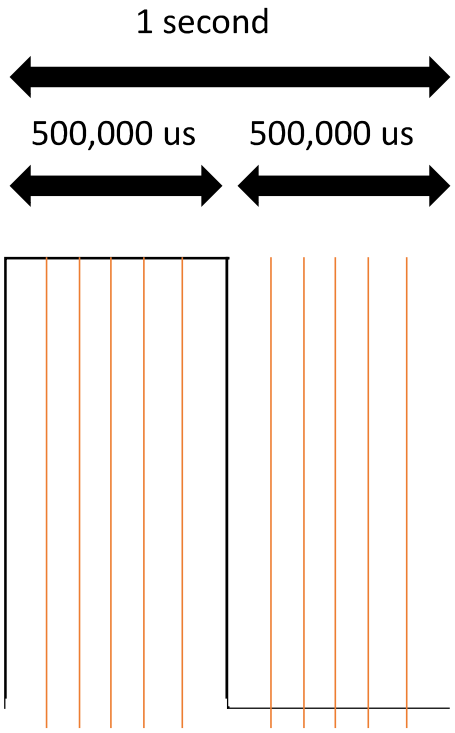
Ref. - https://www.arduino.cc/reference/en/language/functions/time/delaymicroseconds/

playTone

```
#define DX 12

int abc[] = {262, 294, 330, 349, 392, 440, 494, 523};

void mtone(int dx, int hz, unsigned long tm) {
  unsigned long t = millis();
  unsigned long ns = (long)500000 / hz;
  while ( millis() - t < tm ) {
    digitalWrite(dx, HIGH);
    delayMicroseconds(ns);
    digitalWrite(dx, LOW);
    delayMicroseconds(ns);
  }
}
```



playTone, cont.

```
void setup() {
  pinMode(DX, OUTPUT);
}
void loop() {
  for (int i=0; i<8; i++) {
    mtone(DX, abc[i], 500);
    delay(50);
  }
}
```

Using a built-in func, tone()/noTone()

- Generates a square wave of the specified frequency (and 50% duty cycle) on a pin.
- A duration can be specified, otherwise the wave continues until a call to noTone().
- The pin can be connected to a piezo buzzer or other speaker to play tones. Only one tone can be generated at a time.
- Syntax
  - tone(pin, frequency) / noTone(pin)
  - tone(pin, frequency, duration)
- Parameters
  - **pin**: the Arduino pin to generate the tone.
  - **frequency**: the frequency of the tone in Hz. (**unsigned int**)
  - **duration**: the duration of the tone in **milliseconds** (optional). (**unsigned long**)
- Returns
  - Nothing

Let's update our playTone using built-in functions.

Ref. - https://www.arduino.cc/reference/en/language/functions/advanced-io/tone/

How to apply melody?

- Can store a melody into a single data structure?
- Or, define your own data structure for melody!

- How can you get it from the data structure?
- How to translate it into sound?

- One of simple ways is to use an array!
  - Make INDEX value meaningful!
  - If needed, use mapping (number <-> note).
- Or, define your own structucture (struct)

playMelody

```
#define DXPIN 12

void setup() {
}
void loop() {
  int i;
  for (i = 0; i < 12 ; i++) {
    tone(DXPIN,                    );
    delay(500);
    noTone(DXPIN);
    delay(500);
  }
}
```

Assignment: playMelody 2.0

- Requirements
  - Change or rewrite playMelody sketch program to support sound duration.
    - E.g. SOL 500ms, SOL 500ms, MI 500ms, MI 500ms, RE 1000ms
  - Change the melody to the simple song you enjoy.
    - Melody length is max. 12 notes.
  - Write block-type comments in the top of your source code, which includes "your student no., your name, writing date, what you feel about this assignment, etc."

- Results
  - (a source code file) sketch source code (***"sketchfilename.ino"***)
  - (a video capture file) capture a scene of playing music in your Arduino using your smartphone (max. 1GB file).