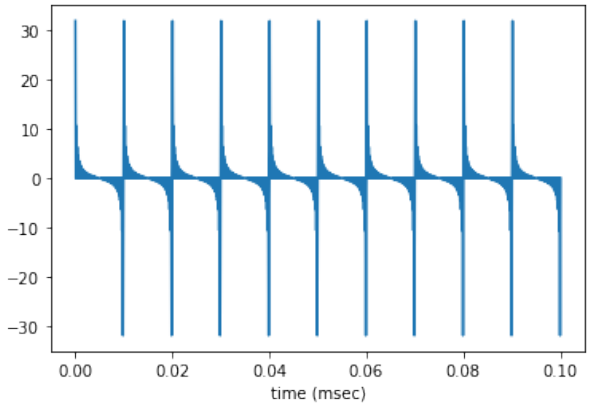
Week8 Summary

```
    s = np.sin(theta)*amp # 진폭 구현
    c = np.exp(theta*1j)
!pip install sounddevice
import sounddevice as sd
    sd.play(c.real, sr) # ipd.Audio(s, rate=sr) 와 같은 역할
```

 # sampling rate가 100Hz이면, frequency는 1, 2, ... 50까지 가능하지 100 이상은 불가능하다.
 # 주파수 표현은 주기를 돌아야 하니까, nyquist frequency = sampling rate/2 까지 가능하다.
 # CD 음질이 sampling rate = 44100Hz인 이유는 nyquist frequency = 22050Hz이고, 사람의 가청 주파수가 20000Hz이기 때문이다.

5. Generate pulse train

```
# generate samples, note conversion to float32 array
    F0 = 100; Fend = int(sr/2) #nyquist frequency
    s = np.zeros(len(t))
    for freq in range(F0, Fend+1, F0): # F0(100)부터 Fend(sr/2)까지 늘려가는데, F0(100)단계로
        theta = t*2*np.pi*freq
        tmp = amp * np.sin(theta)
        s = s + tmp
        print('tmp:', tmp)
        print('s: ', s)
  # 처음의 s값을 정의해줘야 한다. np.zeros(len(t)); sine wave를 계속 더하는 시작은 [0., 0.,…0.]
    fig = plt.figure()
    ax = fig.add_subplot(111)
    ax.plot(t[0:1000], s[0:1000]);
    ax.set_xlabel('time (msec)')
    ipd.Audio(s, rate=sr)
```




```
# sine wave 부드러웠던 곡선이 없어지고 선 하나가 남는다.
```

 # 고주파일수록 amplitude가 낮은 spectrum을 만들고, formant creation을 해준다.
 # 입술이 공명 역할을 해주는데, F1이 500이고, F2가 1500이면 입의 중간의 음을 낸다.

```
def hz2w(F, sr):
    NyFreq = sr/2;
    w = F/NyFreq *np.pi;
    return w
```

```
def resonance (srate, F, BW):
    a2 = np.exp(-hz2w(BW,srate))
    omega = F*2*np.pi/srate
    a1 = -2*np.sqrt(a2)*np.cos(omega)
```

```
    a = np.array([1, a1, a2])
    b = np.array([sum(a)])
    return a, b
```

```
RG = 0  # RG is the frequency of the Glottal Resonator
BWG = 100  # BWG is the bandwidth of the Glottal Resonator
a, b=resonance(sr,RG,BWG)
s = lfilter(b, a, s, axis=0)
ipd.Audio(s, rate=sr)
```

```
RG = 500  # RG is the frequency of the Glottal Resonator
BWG = 60  # BWG is the bandwidth of the Glottal Resonator
a, b=resonance(sr,RG,BWG)
s = lfilter(b, a, s, axis=0)
ipd.Audio(s, rate=sr)
```

```
RG = 1500  # RG is the frequency of the Glottal Resonator
BWG = 200  # BWG is the bandwidth of the Glottal Resonator
a, b=resonance(sr,RG,BWG)
s = lfilter(b, a, s, axis=0)
ipd.Audio(s, rate=sr)
```

```
s = lfilter(np.array([1, -1]), np.array([1]), s)
ipd.Audio(s, rate=sr)
```