## Demo 5 #1

1. Single program for mono and stereo. Write a single Python program to play both mono and stereo wave files. The program should determine the number of channels by reading the wave file information.

   Verify that your program can play both mono and stereo wave files encoded with 16-bits per sample.

**Python Code: 05_1.py**

```
import pyaudio
import wave
import struct
import math

def clip16( x ):
    if x > 2**15-1:
        x = 2**15-1
    elif x < -2**15:
        x = -2**15
    else :
        x = x
    return (x)

gain = 1

wavfile = 'author.wav'
# wavfile = 'sin01_stereo.wav'
print('Play the wave file %s.' % wavfile)

wf = wave.open( wavfile, 'rb' )

num_channels    = wf.getnchannels()     # Number of channels
RATE            = wf.getframerate()     # Sampling rate (frames/second)
signal_length   = wf.getnframes()       # Signal length
width           = wf.getsampwidth()     # Number of bytes per sample
print('The file has %d channel(s).'           % num_channels)
print('The frame rate is %d frames/second.'    % RATE)
print('The file has %d frames.'               % signal_length)
print('There are %d bytes per sample.'        % width)

p = pyaudio.PyAudio()
# Open audio stream
```

```
stream = p.open(
   format      = pyaudio.paInt16,
   channels    = num_channels,
   rate        = RATE, #also change the frequency
   input       = False,
   output      = True )

if num_channels == 1:
          # Get first frame
          input_string = wf.readframes(1) #a sample from mono, 2 sample from stereo

          while len(input_string) > 0:
   # Convert string to number
                    input_tuple = struct.unpack('h', input_string)  # One-element tuple
                    input_value = input_tuple[0]                    # Number
   # ?Compute output value
                    output_value = int(clip16(gain * input_value))  # Integer in allowed range
   #16 bits
   # Convert output value to binary string
                    output_string = struct.pack('h', output_value)
   # Write binary string to audio stream
                    stream.write(output_string)
   # Get next frame
                    input_string = wf.readframes(1)
          print('* Finished')

elif num_channels == 2:
          # Read first frame
          input_string = wf.readframes(1)

          while len(input_string) > 0:
   # Convert string to numbers
                    input_tuple = struct.unpack('hh', input_string)  # produces a two-element tuple
   # Compute output values
                    output_value0 = int(clip16(gain * input_tuple[0]))
                    output_value1 = int(clip16(gain * input_tuple[1]))

   # Convert output value to binary string
                    output_string = struct.pack('hh', output_value0, output_value1)

   # Write output value to audio stream
                    stream.write(output_string)

   # Get next frame
```

```
                input_string = wf.readframes(1)

        print('* Finished')

stream.stop_stream()
stream.close()
p.terminate()
```

**Comments:**

I wrote this code with if loop and determined the channels with getnchannels() so that the program could run either mono or stereo.
When the wave file was ‘author.wav’, “num_channels    = wf.getnchannels()”  would get a value of 1, since it was mono. The program would go to “if num_channels == 1” and run the loop with one output value.
When the wave file was 'sin01_stereo.wav', “num_channels    = wf.getnchannels()”  would get a value of 2, since it was stereo. The program would go to “elif num_channels == 2” and run the loop with two output values.