# کتابخانه‌های کد:

import numpy as np

import scipy.io.wavfile as wav

یکی از زیر مجموعه‌های کتابخانه scipy برای خواندن و نوشتن و کار با فایل‌های صوتی با فرمت wav.

import matplotlib.pyplot as plt

import sounddevice as sd

از این کتابخانه هم برای پخش صوت استفاده کردیم.

sampling\_rate, data = wav.read('../resource/voice1.wav')

داده‌های فایل صوتی voice1 رو با تابع read در کتابخانه wav خوندیم و مقدار نرخ نمونه گیری (تعداد نمونه گیری در ثانیه) در sampling\_rate و خود داده‌های صوتی فایل در data خروجی و ریخته می‌شود.

print('sampling rate:', sampling\_rate)

print('data type:', data.dtype)

print('data shape:', data.shape)

N, no\_channels = data.shape

print('signal length:', N)

چاپ مقادیر که به ترتیب موارد زیر را نمایش میدهند:

نرخ نمونه برداری

نوع مقادیر آرایه data که int16 است.

چند در چند بودن آرایه data که نشان دهنده دو کاناله بودن و طول کل نمونه‌های گرفته شده، می‌باشد که در ادامه در N و no\_channels ریخته می‌شوند.

channel0 = data[:, 0]

channel1 = data[:, 1]

ریختن مقادیرکانال اول در channel0 و دوم در channel1

def save\_wav(filename, data, samplerate):

wav.write(filename, samplerate, data)

داده‌های صوتی (data) رو با نرخ (samplerate) داده شده با نام filename ذخیره می‌کند.

def play\_audio(data, samplerate):

sd.play(data, samplerate)

sd.wait()

داده‌های صوتی رو با نرخ گرفته شده، پخش می‌کند.

def main():

linear\_data = data

save\_wav('linear\_pcm.wav', linear\_data, sampling\_rate)

play\_audio(linear\_data, sampling\_rate)

print('Playing Linear PCM')

تابع اصلی برای اجرا که توابع توضیح داده شده بالا را اجرا می‌کند البته با اجرا شدن قسمت پایینی.

if \_\_name\_\_ == "\_\_main\_\_":

main()

چک می کند که فایل فعلی همان فایل اجرایی است یا خیر که در صورت درست بودن شرط تابع main را اجرا می‌کند.