*#!/usr/bin/env python3*

*# -\*- encoding: utf-8 ID:U1910213 U1910090 U1910001 U1910008-\*-*

*import wave*

*import contextlib*

*import numpy as np*

*from scipy import signal*

*from scipy.io import wavfile*

*from scipy.fft import rfft, rfftfreq*

*# Ask for the file name*

*# if the file doesn't exist, ask again*

*while True:*

*file\_name = input("Enter the file name: ")*

*try:*

*wavfile.read(file\_name)*

*break*

*except FileNotFoundError:*

*print("File not found, please try again.”)*

*FILE = file\_name*

*SAMPLE\_RATE = 44100*

*LOW\_CUT = 200*

*HIGH\_CUT = 600*

*def butter\_bandpass(lowcut, highcut, fs, order=5):*

*nyq = 0.5 \* fs*

*low = lowcut / nyq*

*high = highcut / nyq*

*b, a = signal.butter(order, [low, high], btype='band')*

*return b, a*

*def butter\_bandpass\_filter(data, lowcut, highcut, fs, order=5):*

*b, a = butter\_bandpass(lowcut, highcut, fs, order=order)*

*y = signal.lfilter(b, a, data)*

*return y*

*def main():*

*with contextlib.closing(wave.open(FILE, 'r')) as f:*

*frames = f.getnframes()*

*rate = f.getframerate()*

*duration = frames / float(rate)*

*durations = int(duration)*

*audio\_freq\_sample, audio\_in = wavfile.read(FILE)*

*normalized = np.int16((audio\_in / audio\_in.max()) \* 32767)*

*sample\_number = SAMPLE\_RATE \* durations*

*yf = rfft(normalized)*

*xf = rfftfreq(sample\_number, 1 / SAMPLE\_RATE)*

*# Print the frequency of the signal*

*print("The frequency of the signal is: ", xf[np.argmax(np.abs(yf))])*

*y = butter\_bandpass\_filter(normalized, LOW\_CUT, HIGH\_CUT, SAMPLE\_RATE, order=5)*

*output = np.int16(y \* (32767 / y.max()))*

*wavfile.write('out\_' + FILE, SAMPLE\_RATE, output)*

*if \_\_name\_\_ == '\_\_main\_\_':*

*main()*