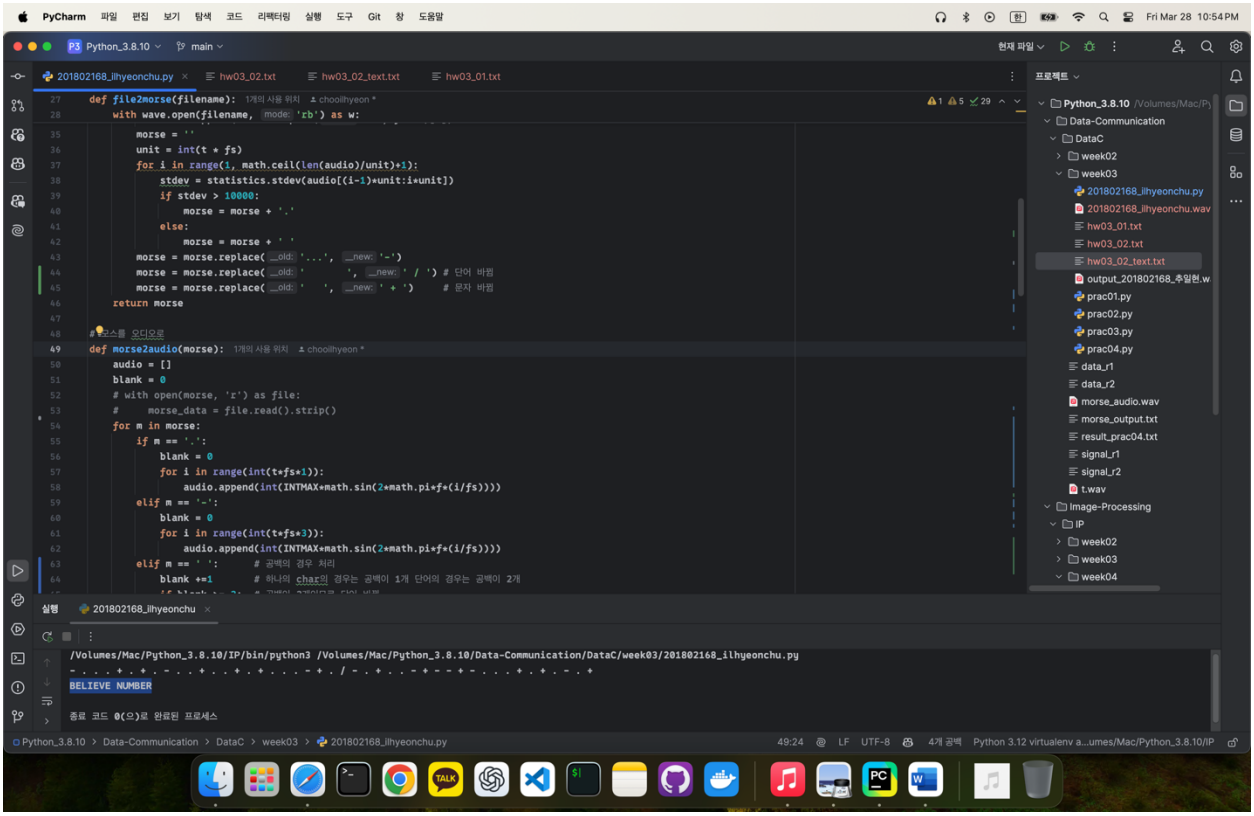
데이터 통신 3 주차 보고서

201802168 추일현



결과 사진

전체적인 과정을 정리하면

1. 주어진 .wav 파일을 모스로 변환
2. 모스를 우리가 사용하는 문자로 변환

이렇게 2 가지 단계로 나뉜다.

결과는 BELIEVE NUMBER 이다.

작성 또는 수정하여 사용한 함수는 2 가지이다.

```
def file2morse(filename):
    with wave.open(filename, 'rb') as w:
        audio = []
        framerate = w.getframerate()
        frames = w. getnframes()
        for i in range(frames):
            frame = w.readframes(1)
            audio.append(struct.unpack('<i', frame)[0])
        morse = ''
        unit = int(t * fs)
        for i in range(1, math.ceil(len(audio)/unit)+1):
            stdev = statistics.stdev(audio[(i-1)*unit:i*unit])
            if stdev > 10000:
                morse = morse + '.'
            else:
                morse = morse + ' '
        morse = morse.replace('...', '-')
        morse = morse.replace('        ', ' / ') # 단어 바뀜
        morse = morse.replace('    ', ' + ')     # 문자 바뀜
    return morse
```

1 번 과정인 .wav 파일을 모스로 변환하는 함수

위쪽 부분은 실습에서 주어진 코드를 그대로 사용하였고

아래쪽에 변환된 모스 코드에서 공백 부분들을 어떻게 처리할지에 대한 코드들을 추가하였다.

우선 주어진 코드는 모든 신호를 '.' 과 ' ' 으로 바꾼 후 '.'이 3 개 연속일 경우 '-' 로 바꾼다.

이 경우 여러 문자나 단어가 오면 제대로 변환이 이루어지지 않으므로 해당 부분에 대해 문자나 단어가 바뀔 때 ' ' 이 연속으로 나타나므로 그것을 이용했다.

7 개의 공백은 7unit 으로 단어가 바뀌는 것이고 3 개의 공백은 문자가 바뀌는 것이므로 코드를 맞춰서 작성했다.

추가로 해당 코드의 순서를 바꿀 경우 단어가 바뀌는 7 개의 공백이 2 개의 ' + ' 와 하나의 ' ' 로 바뀌므로 순서에 주의해야한다.

```
def morse2text(morse):
    text = ''
    word = ''
    morse_to_text = {v: k for k, v in english.items()}
    morse_to_text.update({v: k for k, v in number.items()})

    print(morse)
    for code in morse.split(' '):   # 모스코드 단어 단위로 나누기
        if code == '+':     # 문자 바뀜 word 에 모여있는 모스 코드들을 문자로 변환 후 text 에 추가
            if word in morse_to_text:
                text += morse_to_text[word]
            word = ''
        elif code == '/':   # 단어 바뀜 word 에 모여있는 모스 코드들을 문자로 변환 후 text 에 추가
            if word in morse_to_text:
                text += morse_to_text[word]
            text += ' '
            word = ''
        else :              # 모스 코드를 word 에 추가시켜 나중에 문자단위로 변환
            word += code
    print (text)
    return text
```

2 번째 과정인 모스 코드를 우리가 사용하는 문자열로 바꾸는 함수다.

morse_to_text 부분은 계속 결과가 안나와 찾아본 결과 기본적으로 주어진 english 와 number 가 '문자' : '모스부호' 로 되어있어 제대로 작동을 안하는것을 알게되었고 key 와 value 를 바꿔주는 함수를 이용해 morse_to_text 라는 매핑을 만들어 추가했다.

우린 위에서 문자나 단어가 바뀌는 것은 '+' 와 '/'를 이용해 구별하였고 모든 '.', '-', '+', '/' 은 ' '을 두어 구별했으니 morse.split(' ') 을 통해 나누면서 반복문에 넣어 모스 부호를 문자로 바꿨다.

우선 '+' 의 경우는 문자가 바뀌는 경우로 (ex. BE 에서 B 를 나타내는 모스 부호가 끝난 경우) 지금까지 word 에 모아둔 모스 부호를 morse_to_text 에서 찾아서 변환한 후 text 문자열에 추가한다.

이후 word 를 초기화한다.

'/'의 경우는 단어가 바뀌는 경우로 (ex. BELIEVE NUMBER 에서 E 까지 끝나고 N 이 시작) 이때까지 word 에 모둔 모스 부호를 '+' 경우와 같이 추가한 후 ' ' 을 하나 더 추가해준다.

나머지 경우는 '.', '-' 이 오는 경우로 이때는 아직 하나의 문자조차 다 모이지 않은 상태이므로 그냥 word 에 추가해준다.

중간에 있는 print() 의 경우 디버깅 과정을 수월하게 하기위해 넣어둔 코드로 과제와 상관없다.

하지만 결과에 영향을 주지 않고 2 번째 print 의 경우는 해석된 문장을 보여주므로 편하기 때문에 그냥 지우지 않고 내버려두었다.

해석문장은 BELIEVE NUMBER 이다.