1. The first notable obstacle I had was just getting started programming the functions. It was hard to figure out how large of a scope I wanted to create the program. I didn’t know how far to plan in terms of cases so I mapped out a pseudo code for both functions. Then, I developed the program incrementally, constantly testing multiple cases until I finally got the whole function to work properly. Then, I moved on to the next function and succeeded in the same way. Another obstacle I faced was using the encodeNote function. The function takes in an int octave variable, which was a slight problem because the song string only returned chars, even for digits. But I solved this problem because I realized that if you store a char into an int, it stores the ASCII value of the digit. Then, I just subtracted the value by 48 in order to get the real int value. Finally, I realized that the encodeSong function wasn’t supposed to make changes to the instructions string unless the song string was playable. I fixed this problem by using a tempSong string to hold the song and if the song was playable, I would set instructions to tempSong.
2. hasCorrectSyntax

repeatedly through each character of string:

if beat ending:

break

if note:

check next character

if beat ending or note:

break

if accidental:

check next character

if beat ending or note:

break

if digit:

check next character

if beat ending or note:

break

else:

return false

else:

return false

if digit:

check next character

if beat ending or note:

break

else:

return false

else:

return false

else:

return false

encodeSong

if song has correct syntax:

repeatedly through song string:

if note:

set note

check next character

if digit:

set octave

elif accidental:

set accidental

check next character

if digit:

set octave

elif beat ending:

if next beat is also beat:

add space to instructions

continue to next iteration

encode note using octave, accidental, and note

if blank it is unplayable:

set badBeat

return 2

add encoded note to instructions

return 0

else

return 1

* Zero beats(“”)
* One beat with no notes(“/”)
* One beat with one note(“A/”)
* One beat with one accidental note(“A#/”)
* One beat with one octave note(“A6/”)
* One beat with one accidental and octave note(“Ab3/”)
* String with multiple empty beats(“A#6/B4///Db6//G3/”)
* Unplayable string with notes outside octave(“C0C0DC0DD/E#FbB#Cb/B#9/”)
* Multiple empty beats string(“///”)
* No beat ending(“D3/D4/D5”)
* Lower case letters(“abc/”)
* Random symbols(“$@#!$/”)
* Two accidentals in a row(“A##/”)
* Same notes in one beat("DADDA/”)