The assignment is to be able implement functions that will ultimately read in a song string and play the song string. Playing the song string for part one only involves printing the note letters and their durations. The song will be coded directly into the main function. My code outputs the note letter and duration in RealTerm via UART serial connection. The notes are represented by values A through G and R will represent a rest in the song. The durations of the notes or rests will be between 0 seconds and 31 seconds.

The code follows a specific algorithm for taking in song strings and "playing" them. The general algorithm is as follows:

1. Pass a song string to StoreSong() function
    a) For every character, check if they are null. If it is, stop and append a R0 note to the song.
    b) If able, store three values as the note letter, and possibly two duration characters. There is a chance that the duration is double digits.
    c) If the value stored in the 2nd digit's place is not actually an integer, the first duration value is the entire duration.
    d) If the value stored in the 2nd digit's place is an integer, the duration has a tens and a ones place.
    e) Pack the notes (see step 2)
    f) Continue steps 1.a. through 1.d. until an "R0" is found or a "\0" null character is found. Jump to step 3.

2. Pass note characters and durations to the PackNote() function
    a) If the character is 'R' or 'r', store 0b00000111 into a binary value holder
    b) Assuming the input is always valid, subtract 65 from the ASCII value because 65 is the offset of ASCII value to the binary bit representation.
    c) Take the result from step 2.a. or 2.b. and shift it to the left by 5 bit places.
    d) Add on the duration value. Return to step 1.f.

3. Play the song using the PlaySong() function.
    a) For every binary representation of a note, unpack the note letter by masking it with 0b11100000 then shifting the result to the right by 5, unless the value is 0b11100000 after masking. This means that the value is simply 'R'
    b) For every binary representation
    c) Play the note by printing the note letter and duration in the following format until there are no more note letter and note duration pairs:

[Note Letter] [Duration]

A few obstacles I faced while coding this part of the homework were that there were some issues using the sizeof() function where I tried using it to obtain the number of notes needed to be

played but it only allowed me to play two notes. This was resolved by avoiding the use of sizeof() and instead using a while loop. Since every song has an “R0” note to resemble the end of the song, I will play the song until the “R0” value is found. Another issue I faced was that I had issues printing via UART. I continued to get errors trying to use printf_P(PSTR()) functions but found that using printf() was sufficient. I do not completely understand why that is.