## Task 1 - Description

Task 1 generates three melodies – primary melody, alternate melody, and bassline melody using a simple constraint system dealing with octaves, duration, key, and harmonization.

## Infrastructure

A global variable, called **beat-total** defines the number of "beats" a user wants to be generated. This is constrained to 4/4 time, where 4 beats make up a measure.

A single note is represented by the *Note* object. The *Note* object has the following fields:

1. Pitch - The pitch is the sound of a note. It can be thought of in numeric form as a number based on its position on the scale. Currently, the program constructs melodies using only pitches within the Cmajor scale: C D E F G A B.
2. Octave - This one is harder to define. According to Encyclopedia Britannica, an octave is "an interval whose higher note has a sound-wave frequency of vibration twice that of its lower note," (https://www.britannica.com/art/octave-music). I think about octaves more instrumentally since that is how I learned music. On an instrument – say piano, for simplicity – you can have multiple C notes across the keys, with distinction generally noted as low, high, or middle C. This project uses 3 different octaves to aid in distinguishing the bassline from the other two melodies.
3. Duration - The duration is the length, measured in beats, of a note. Whole notes, half notes, quarter notes, and eighth notes are used in this project.
4. Str-representation - This is the string representation of a note in terms of ABC Notation. This representation can be copied over to EasyABC and played.

The constraint knowledge base consists of the following:

- The variable **CMAJOR** is assigned to the list '( C D E F G A B ).

- The variable **melody-durations** is assigned to the list '( 2 1 0.5 ), which represents the beats of a half note, quarter note, and eighth note, respectively. These durations are used for the 1st two melodies only.
- The variable **bassline-durations** is assigned to the list '( 4 2 1 ), which represents the beats of a whole note, half note, and quarter note, respectively. This is only used for the third melody.
- The variable **melody-octaves** is assigned to the list '( 2 3 ), which represents the octaves of the 1st two melodies only.
- The variable **bassline-octave** is assigned to the list '( 1 ), which represents the octave of the third melody (bassline) only.

## High-level Melody Generation

All melodies are generated in a similar manner (except for melody 2, which differs slightly from the others). The steps are the following.

1. A list of durations are generated based on the **beat-total** limit. The list is generated until the sum of the durations is equal to or greater than the **beat-total**. If greater than, the list is cleared and regenerated until it is equal to the **beat-total**. This process is accomplished using the *generate-durations* method.
    a. If melody 1 or 2 is being generated, durations are randomly chosen from the **melody-durations** list.
    b. If melody 3 is being generated, durations are randomly chosen from the **bassline-durations** list.
2. Using the list of durations, pitches are generated for each duration. This process differs between the melodies.
    a. **<u>For Melodies 1 and 3:</u>**
        i. The pitches are selected randomly from **CMAJOR** and added to a pitch list until the length of the pitch list matches the length of the duration list.

b. **<u>For Melody 2:</u>**

   i. For melody 2, one of three outcomes can happen: (1) a harmonization is generated based on melody 1, (2) a permutation is generated based on melody 1 or (3) a random melody is generated. The choice is randomly selected.

      1. Harmonization
         a. Copy melody 1's duration list.
         b. Get the position of the first pitch in melody 1's pitch list using **CMAJOR**. Then, add 2 to the position if it does not exceed the length of the list. Otherwise, subtract 2 from the position. Use the new position to get the harmony pitch from **CMAJOR**.
         c. Add the new pitch to melody 2's list of pitches.
         d. Continue this process until melody 2's list of pitches matches the length of melody 1's list of pitches.
      2. Permutation
         a. Copy melody 1's duration list.
         b. Randomly select a pitch from melody 1's pitch list.
         c. Add that pitch to melody 2's pitch list.
         d. Remove that pitch from melody 1's pitch list.
         e. Repeat until there are no pitches left in melody 1's pitch list.
      3. Random - The process is the same as melody 1 and 3.

3. An octave is randomly selected. If melody 1 or 2 is being generated, the octave is randomly selected from the **melody-octaves** list. If melody 3 is being generated, the octave is selected form the **bassline-octave** list.
4. Create a note object and populate with the first list element from the pitch-list and duration-list for their respective fields. Populate the octave field with the aforementioned octave. Generate an EasyABC string representation of the note and initialize to the *str-representation* field. Repeat this process with the *cdr* of the lists until the lists are empty.

5. The list of note objects is generated (yay). This is your melody!