Documentation

Elias Devadoss, Himal Pandey

0.1 BNF Grammar

```
<num> ::= positive integer
<song> ::= <tempo> <meter> <melody> <beat> <chordList>
<tempo> ::= <num> bpm
<meter> ::= <num> / <noteType>
<noteType> ::= 1 | 2 | 4 | 8 | 16 | 32
<melody> ::= <note>*
<note> ::= <pitch> <duration>
<duration> ::= <num> beats
<pitch> ::= <letter> <accidental> <octave> | e
<letter> ::= A | B | C | D | E | F | G
<accidental> ::= # | b | e
<octave> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8
<beat> ::= <percuss>*
<percuss> ::= <sound> <num>+
<sound> ::= kick | snare | hi-hat | crash | ride | china | splash
<chordList> ::=  <chord>*
<chord> ::= <pitch> <pitch> <pitch>+ <duration>
```

0.2 Syntax

Syntax: `l`
Abstract Syntax: `letter` of `char`
Type: `char`
Prec./Assoc.: n/a
Meaning: `l` is a primitive that we represent using the `char` data type. It represents the pitch of a given note.

Syntax: `octave`
Abstract Syntax: `num` of `int`
Type: `int`
Prec./Assoc.: n/a
Meaning: `octave` is a primitive that we represent using the `int` data type. It represents the octave of a given note.

Syntax: `duration`
Abstract Syntax: `num` of `int`
Type: `int`
Prec./Assoc.: n/a
Meaning: `duration` is a primitive that we represent using the `int` data type. It represents the number of beats a given note is held out.

Syntax: `lo d`
Abstract Syntax: `Note` of ((`letter` * `num`) * `num`)
Type: `char -> int -> int -> ((char * int) * int)`
Prec./Assoc.: n/a
Meaning: `Note` is a combining form that represents a note. It consits of a tuple of a tuple of a char and an int, and an int. This represents the notes pitch, octave, and duration, ex. C4 2