

to kill a monarch

Preferably played in a dark or dim setting (e.g. with the least light needed by the performers).

michael winter  
(berlin, de; 2021)

notes . . . . . 1

musical score . . . . . 4

appendix 2 - SuperCollider code and Lilypond templates . . . . . 70

instrumentation and dedication

This piece was originally written for quartertone bass flute and three strings. There are also two electronic accompaniment parts (synthesized by a custom computer program written in SuperCollider).

However, the instrumentation is flexible such that the part labeled in the score by an asterisks (the part originally intended for flute; referred to here as part ‘star’) has a distinct timbre from the other three parts which are labeled by Roman numerals and should be rather homogeneous in timbre within themselves (the parts originally intended for strings, referred to here as the ‘candidate’ parts). These three parts may also be synthesized and played back with the electronic accompaniments (individually referred to as ‘electronic accompaniment’ I and II). Conversely, the part star can be synthesized, while the candidate parts are played on acoustic instruments.

That is, the piece may be played as a solo, trio, or quartet. The reason for this variability stems from a sort of economy-of-means. After moving to Berlin, I started discussing with Rebecca Lane writing a piece for her to play on microtonal-bass flute. At the same time, three mutual friends of ours, all cellists, Deborah Walker, Judith Hammann, and Lucy Railton, were entertaining the idea of starting a cello trio as the former two had also recently moved to Berlin. This flexible instrumentation gives the option for each of them to play the piece as a solo or together in different configurations. There is a certain pragmatism, however, that makes the part star playable on a microtonal bass flute and the candidate parts specifically suited for strings. These idiosyncrasies are given in more detail in the part descriptions below.

I would like to extend a special thanks to Rebecca Lane who compelled the piece. For encouraging me to write it and more specifically, for her suggestion early on to be fastidious about notating all the interrelations among and within the parts. Doing so led to a comprehensive analysis and better understanding of the underlying process which ultimately made the piece possible.

process and structure (the first paragraph may be used as a short program note)

The first half of the piece is a sort of extended prelude. The second half is a series of modulations / interpolations where each modulation goes from one mode, always a gamut of 7 pitches built upon a given root / fundamental, to another. The interpolation is governed by an algorithm that models the phenomenon where the rich-get-richer such that the more pitches that have been selected from one of a set of potential ‘candidate’ modes, the more likely that candidate will eventually become the destination mode. Once a mode ‘wins’ (becomes overwhelming rich), its wealth is effectively stripped and it can no longer persist.

The current state of the interpolation is articulated by part star and electronic accompaniment I while the candidate parts articulate candidate modes that *may eventually become* the destination mode. That is, at any point in time, the pitch gamut of part star and electronic accompaniment I is comprised of subsets of pitches from the candidate modes articulated. As such, part star and electronic accompaniment I are typically multimodal (built upon 2 or more roots) while, individually, the candidate parts are always unimodal (each built upon one root). The exception being when a destination mode is reached and all parts play the same gamut of pitches built upon a single root.

The score is divided into sections and subsections. The first section is the extended prelude. Starting from section 2, each section is one full modulation of the above-described process where each subsection can be considered as a discrete point in the interpolation. The destination mode is always reached by the ultimate subsection of each section.

## notation

At the beginning of section 1 and each subsection from section 2 onward, a key is provided above the staff that indicates the current root for each mode articulated by each of the candidate parts, respectively; the relationships among the roots; and the relationship of the previous root to the current root within each part. The relationships among the roots are given by frequency ratios written above lines that connect the part numbers (in Roman numerals). The relationship of the previous root to the current root within each part is given by a frequency ratio written below the note of the current root. The note indicating the pitch of the current root is preceded by a note (given in gray) indicating the pitch of the previous root unless the root has not changed.

For the candidate parts, each note indicates the closest pitch in twelve-tone equal temperament with a cent-deviation (100th of a tempered semitone) written above and a frequency ratio from the current root written below. Part star is the same except that the frequency ratio may be written as a superscript of a Roman numeral that indicates which root (of the mode from candidate part I, II, or III) the frequency ratio is referencing. If no Roman numeral is given, the last one is assumed.

All frequency ratios are given in a *collapsed* form as if the pitches were within one octave above the same arbitrary C and always in the form where the numerator is greater than the denominator.

## candidate parts

These parts are designed such that within each subsection, each part only sounds tones with pitches from one mode. A mode is always a 7 pitch gamut in the following form (given by frequency ratios and cents from the 1/1):

		5/4 (386¢)	11/8 (551¢)		13/8 (841¢)	7/4 (969¢)
1/1	9/8 (204¢)			3/2 (702¢)		
		6/5 (316¢)	4/3 (498¢)		8/5 (813¢)	15/8 (1088¢)

Horizontally aligned frequency ratios indicate that one or the other pitch may be used in the mode (most likely the top one). Therefore, only a handful of modes are possible even though the root progresses / changes throughout. Since the mode structure is rather limited, the players need to be able to transpose the possible modes arbitrarily. This is why these parts are specifically suited for strings. Basically the intervals within the modes stay the same or similar, but the position on the instrument changes.

## part star

The pitch gamut of this part often comprises notes derived from several candidate modes at once. As mentioned above, a Roman numeral indicates the part that is articulating the candidate mode from which the note is drawn from. When the gamut is multimodal, the part generally has a lower temporal density and the sequence of pitches is always rising: each pitch is followed by the next highest pitch in the gamut until an upper limit is reached. This should make playing the part more feasible for a wind instrument like bass flute despite the complexity. Throughout the first section and in each ultimate subsection from section 2 onward, the part comprises arbitrary sequences of notes like the other parts. Therefore, there is a distinct shift starting at section 2, where the rising sequences begin. This should be made as clear as possible. Similarly, the interruption of the rising scale in the ultimate subsection of each section should also be made as clear as possible. If necessary, this part can be transposed up or down an octave. The part is written assuming bass flute hence the octavation marking below the treble clef; i.e. sounding an octave lower.

## electronic accompaniments I and II

Electronic Accompaniment I articulates the current state of the interpolation as mentioned previously. Electronic Accompaniment II articulates the overall section-per-section form by swelling throughout each section on a tone with a pitch that is a perfect 5th above or a perfect 4th below the root of the destination mode which then cadences to the root itself in the ultimate subsection of each section.

## dynamics

Each section should generally have a dynamic crescendo that peaks in the ultimate subsection. This can be executed by following the dynamic envelope of electronic accompaniment I where the candidate parts sound within / equal to the electronics and part star sounds slightly above / in the foreground. Generally, the sound should be rather present; filling the room more and more throughout each crescendo. However, the beginning of the crescendo need not be extremely quiet nor the peak excessively loud. With that said, the cadential peak of electronic accompaniment II in each ultimate subsection should briefly overwhelm all the other parts.

Within each subsection, each part often has a small flourish of activities which should be articulated as sub-swells within the larger dynamic profile with peaks based on the temporal density.

tempo

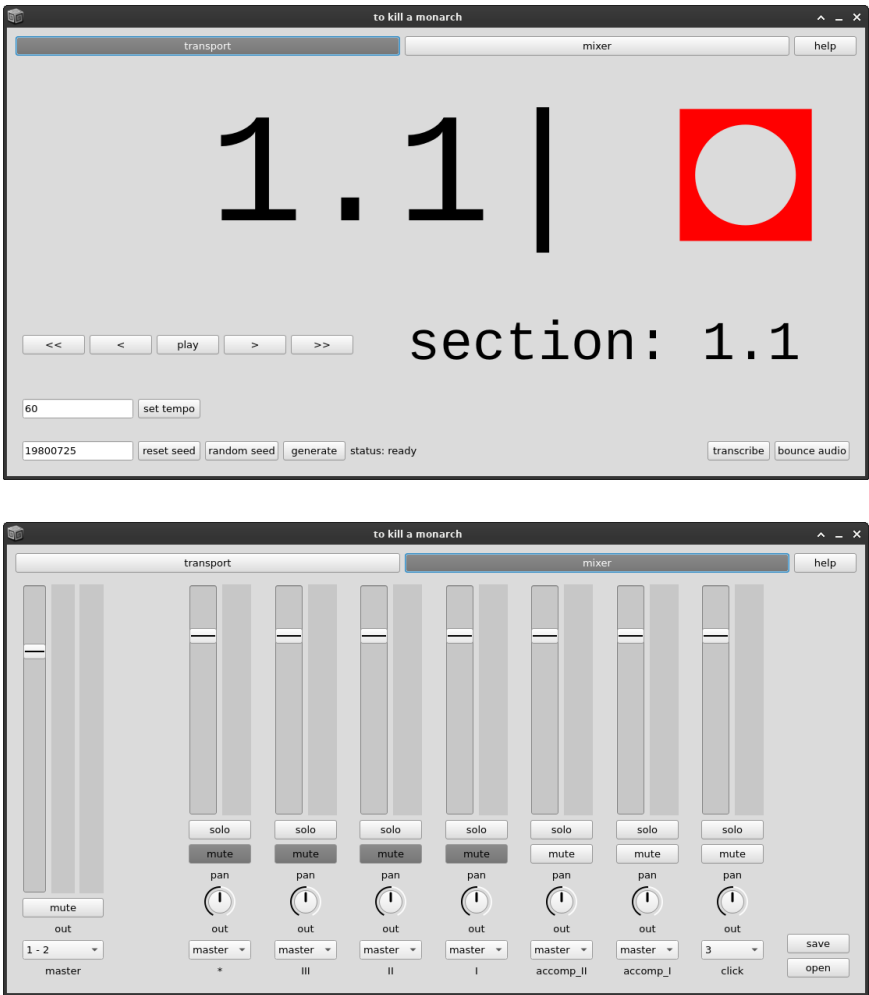
The score is written in a cut time with a tempo where the half note equals 60 beats per minute. The piece maybe be played at a slower tempo, but no less than the half note at 50 beats per minute.

SuperCollider program

A custom program written in the SuperCollider language can synthesize the parts that can be played by acoustic instruments and the strictly electronic accompaniments. A version of the application source code is appended at the end of this score. However, it is recommended to ensure that the most recent version of the code is being used which can be downloadable from a git repository at: [https://gitea.unboundedpress.org/mwinter/to\\_kill\\_a\\_monarch](https://gitea.unboundedpress.org/mwinter/to_kill_a_monarch)

The application provides a transport window to control playback and set variables as well as a basic mixing console to control the levels of the various sonic elements of the piece. The program also allows new versions of the piece to be generated, transcribed, and rendered to separate audio files for use with other playback systems. Note that while most of the code facilitates usability, playback, and transcription, the music of the piece is completely generated by the algorithm in `tkam_musical_data_generator.scd`. A help / readme file is included with the application documenting its functionality and use. To launch the application, execute `tkam_main.scd` in SuperCollider (on Linux, this is achieved by pressing cmd+enter with the cursor anywhere within the code block).

The generation of this document (using LaTeX) contains a version date at the bottom of this page in order to help track changes and the git repository will also detail commit changes. The piece was written using SuperCollider version 3.11.2 and Lilypond version 2.22.0.



application user interface

to kill a monarch

seed: 19800725

michael winter  
(berlin, germany; 2021)

1.1

III 1/1 II 1/1 I

+0 +0 +0

1/1 1/1 1/1

$\text{♩} = 60$

I

II

III

\*

+41 +16 +4 +0

13/8 6/5 9/8 1/1

-31 +2 -31 +0

III 7/4 3/2 7/4 1/1

⑤

+2 -31

3/2 7/4

1.2

+2

3/2

+4 -49 +2

11/8 9/8 3/2

+16 -49 -31 +41 +2 +4

6/5 11/8 7/4 13/8 3/2 9/8

⑨

-49 +0

11/8 1/1

+0 -49

1/1 11/8

-31 +16 +4 +2 +0

7/4 6/5 9/8 3/2 1/1

+0 -49

III 1/1 11/8 -1.

**1.3**

Interval numbers for section 1.3:

- Staff I: +4, 9/8, 3/2, +2, 1/1, +0
- Staff II: +0, 1/1, 7/4, -31, 9/8, 13/8, 3/2, 1/1, +2, +41, +2, -31, +41, +16, -49, +16, +0
- Staff III: +16, 6/5, 7/4, -31, 13/8, 1/1, 11/8, +41, +0, +2, 3/2, +4, 9/8
- Staff \*: -31, III 7/4, 1/1, +0

**1.4**

Interval numbers for section 1.4:

- Staff I: (Measure 17 is a whole rest)
- Staff II: +41, 13/8, +4, 9/8
- Staff III: -49, 11/8
- Staff \*: (Measure 17 is a whole rest)

**1.5**

Interval numbers for section 1.5:

- Staff I: -49, 11/8, +16, 6/5, -31, 7/4
- Staff II: +2, 3/2, -49, 11/8, +0, 1/1, -31, 7/4, +2, 1/1, -49, 11/8, +4, 9/8
- Staff III: -31, 7/4, +41, 13/8, +4, 9/8, +0, 1/1, +2, 3/2, +16, 6/5, +0, 1/1
- Staff \*: +41, 13/8, +2, 3/2, -31, 7/4

(25)

Staff I: +41, +2, +0, +4  
Intervals: 13/8, 3/2, 1/1, 9/8

Staff II: +16, +2, -31, +0, +41, -31, +16, -49  
Intervals: 6/5, 3/2, 7/4, 1/1, 13/8, 7/4, 6/5, 11/8

Staff III: -49, +0, -31  
Intervals: 11/8, 1/1, 7/4

Staff \*: -49, +41, +16  
Intervals: 11/8, 13/8, 6/5

(29) 1.6

Staff I: +0, +16, +0, +2, +0, -49, -31, +0  
Intervals: 1/1, 6/5, 1/1, 3/2, 1/1, 11/8, 7/4, 1/1

Staff II: (Whole notes)

Staff III: +41, +0  
Intervals: 13/8, 1/1

Staff \*: +4, +2  
Intervals: 9/8, 3/2

(33) 1.7

Staff I: +4, +41, +41, -31, +16  
Intervals: 13/8, 9/8, 13/8, 7/4, 6/5

Staff II: +41, +0  
Intervals: 13/8, 1/1

Staff III: +2, +16  
Intervals: 3/2, 6/5

Staff \*: -49, -31, +0  
Intervals: 11/8, 7/4, 1/1

37

Staff I: Measures 37-40. Time signatures: 3/2, 7/4, 1/1. Accidentals: +2, +0, -31.

Staff II: Measures 37-40. Time signatures: 9/8, 7/4. Accidentals: +4, -31.

Staff III: Measures 37-40. Time signatures: 3/2, 1/1, 11/8. Accidentals: +2, +0, -49.

Staff \*: Measures 37-40. Time signatures: 9/8, 13/8. Accidentals: +4, +41.

41

1.8

Staff I: Measures 41-44. Time signatures: 3/2, 11/8, 3/2. Accidentals: +2, -49, +2.

Staff II: Measures 41-44. Time signatures: 7/4. Accidental: -31.

Staff III: Measures 41-44. Time signatures: 3/2, 7/4, 9/8. Accidentals: +2, -31, +4.

45

Staff I: Measures 45-48. Time signatures: 11/8, 6/5, 13/8, 1/1, 3/2, 7/4, 9/8, 3/2, 11/8. Accidentals: -49, +16, +41, +0, +2, -31, +4, +2, -49.

Staff II: Measures 45-48. Time signatures: 1/1, 13/8. Accidentals: +0, +41.

Staff III: Measures 45-48. Time signatures: 3/2, 13/8, 6/5. Accidentals: +2, +41, +16.

Staff \*: Measures 45-48. Time signatures: 6/5, 11/8, 3/2, 6/5. Accidentals: +16, -49, +2, +16.

**1.9**

49

I

II

III

\*

3/2

7/4 9/8 11/8 3/2 1/1 6/5 13/8 11/8

+2

-31 +4 -49 +2 +0 +16 +41 -49

+0

1/1

-31 +2 +0

III 7/4 3/2 1/1 13/8 11/8

**1.10**

53

I

II

III

\*

6/5 13/8 1/1

1/1 7/4 3/2 9/8 6/5

11/8 9/8 7/4

9/8

7/4 1/1 3/2 7/4

1/1 3/2

III 3/2

+16 +41 +0

+0 -31 +2 +4

-49 +4 -31

+4

-31 +0 +2

+2

**1.11**

57

I

II

III

\*

7/4 11/8 9/8

7/4 6/5 13/8 7/4 11/8 1/1 1/1 3/2 7/4

7/4 1/1

-31 -49 +4

-31 +16 +41 -31 -49 +0 +0 +2

-31 +0

-31 +16

+4 +0



61 1.12

Staff I: +0, -31, 7/4, 1/1

Staff II: +41, -49, +0, +16, 13/8, 11/8, 1/1, 6/5

Staff III: -31, +16, +2, +4, +0, 7/4, 6/5, 3/2, 9/8, 1/1

Staff \*: +16, +2, +0, -31, III 6/5, 3/2, 1/1, 7/4

65 1.13

Staff I: -49, +2, +0, +4, 11/8, 3/2, 1/1, 9/8

Staff II: -31, +41, +0, +41, -31, +16, 7/4, 13/8, 1/1, 13/8, 7/4, 6/5

Staff III: +16, -31, -49, +41, +4, 6/5, 7/4, 11/8, 13/8, 9/8

Staff \*: -49, +4, +41, III 11/8, 9/8, 13/8

69

Staff I: +16, +41, -31, -49, +2, +16, +0, 6/5, 13/8, 7/4, 11/8, 3/2, 6/5, 1/1

Staff II: +2, +4, +0, +2, 3/2, 9/8, 1/1, 3/2

Staff III: +2, +16, 3/2, 6/5

Staff \*: +0, III 1/1

73 1.14

73 1.14

II +16 6/5

III

\* +16 6/5

I +2 3/2

II -49 -31 +41 11/8 7/4 13/8 11/8

77 1.15

77 1.15

II +0 +4 -31 -49 +0 +16 +2 3/2 7/4 11/8 1/1 13/8 9/8 7/4 1/1 3/2 +16 6/5

III +2 3/2

\* -31 +2 -49 +0 +41 +4 +0 1/1 13/8 9/8 1/1

81

81

II +41 +4 -31 -49 9/8 13/8 7/4 11/8

III -31 7/4 -49 +41 +0 -31 +16 +4 +2 +0 +4 13/8 11/8 1/1 13/8 7/4 9/8 3/2 1/1 9/8

\* -31 11/8 -49 +2 3/2

85 **1.16**

I II III \*

-49 11/8 -31 7/4 +41 13/8 -49 11/8 +0 1/1 +2 3/2

89

I II III \*

+0 1/1 +4 9/8 +41 13/8 +0 1/1 +41 13/8 +0 1/1 -31 7/4 +16 6/5 -49 11/8 +41 13/8 -31 7/4

**1.17** 93

I II III \*

+2 3/2 +0 1/1 +41 13/8 +4 9/8 -49 11/8 -31 7/4 +16 6/5 +2 3/2 +16 6/5 -49 11/8 -31 7/4 +16 6/5 +41 13/8 +0 1/1 +4 9/8 +2 3/2

97 1.18

I  $\frac{13}{8}$

II  $\frac{1}{1}$   $\frac{3}{2}$   $\frac{9}{8}$

III

\*

$\frac{13}{8}$   $\frac{1}{1}$   $\frac{9}{8}$

+41 +0 +2 +0 +4

101 1.19

I  $\frac{1}{1}$   $\frac{7}{4}$   $\frac{3}{2}$

II

III  $\frac{3}{2}$   $\frac{11}{8}$   $\frac{9}{8}$   $\frac{13}{8}$   $\frac{7}{4}$   $\frac{1}{1}$

\*

$\frac{6}{5}$   $\frac{9}{8}$   $\frac{1}{1}$   $\frac{7}{4}$   $\frac{11}{8}$

+0 -31 +2 +41 -31 +0 +2 -49 +4 +16 +0 -31 -49

105

I  $\frac{9}{8}$   $\frac{6}{5}$   $\frac{1}{1}$   $\frac{3}{2}$   $\frac{11}{8}$   $\frac{7}{4}$

II  $\frac{7}{4}$   $\frac{13}{8}$

III  $\frac{6}{5}$   $\frac{11}{8}$   $\frac{9}{8}$

\*

$\frac{11}{8}$   $\frac{3}{2}$   $\frac{7}{4}$

-49 +4 +16 +0 +2 -31 +41 +16 -49 +4

109 1.20

11/8 3/2 11/8 6/5 1/1 7/4 9/8

3/2 11/8

6/5 11/8 III 1/1

113 1.21

6/5 3/2 11/8 13/8 6/5 3/2 1/1 13/8 9/8 1/1 7/4

13/8

13/8 1/1

9/8 13/8 6/5 3/2 7/4 11/8

117 1.22

11/8 6/5

1/1

7/4 6/5

III 3/2 1/1 6/5 3/2 9/8

*to kill a monarch* (seed:19800725)

The musical score is for the song "The Rose Tree" in G major. It consists of four staves. The first three staves are labeled I, II, and III, and the fourth staff is marked with an asterisk (\*). The key signature has one sharp (F#), and the time signature is 3/8. The score includes various musical notations such as notes, rests, and accidentals, along with fret numbers and interval markings above the notes.

**Staff I:** The melody starts with a half note G (fret 2), followed by a quarter rest, then a quarter note A (fret 3), a quarter note G (fret 2), and a quarter note F# (fret 1). The interval markings are +2, +0, -49, +16, +0, and +2.

**Staff II:** The accompaniment starts with a half note G (fret 2), followed by a quarter rest, then a quarter note A (fret 3), a quarter note G (fret 2), and a quarter note F# (fret 1). The interval markings are +16, -31, +41, and +2.

**Staff III:** The accompaniment starts with a half note G (fret 2), followed by a quarter rest, then a quarter note A (fret 3), a quarter note G (fret 2), and a quarter note F# (fret 1). The interval markings are +2, -49, +0, +4, +41, and +16.

**Staff IV (\*):** The solo part starts with a half note G (fret 2), followed by a quarter rest, then a quarter note A (fret 3), a quarter note G (fret 2), and a quarter note F# (fret 1). The interval markings are -31, +16, and +41.

125

1.23

I

II

III

\*

7/4

III<sup>3/2</sup>

The image displays a musical score for the song "The Rose Tree". It consists of four staves, labeled I, II, III, and \*. Staff I is in treble clef with a key signature of one flat (Bb). Staff II is in alto clef with a key signature of one flat (Bb). Staff III is in bass clef with a key signature of one flat (Bb). Staff \* is in treble clef with a key signature of one flat (Bb). The score includes various musical notations such as notes, rests, and accidentals. Time signatures are indicated below the staves: 11/8, 9/8, 1/1, 13/8, 3/2, 7/4, 13/8, 1/1, 7/4, 9/8, 11/8, 9/8, 13/8, 3/2, 11/8, 6/5, 7/4, and 9/8. A box labeled "1.24" is positioned above staff I. A circled number "129" is located at the top left of the page.

133

Staff I: +16 +41 -49 +4 +0  
6/5 13/8 11/8 9/8 1/1

Staff II: +0 +2 -31 +0  
1/1 3/2 7/4 1/1

Staff III: -31 +41 +16 +0 +2 +4 -49 -31  
7/4 6/5 13/8 1/1 3/2 11/8 7/4

Staff \*: +2 +0 +41 +0 -49 +16  
III 3/2 1/1 13/8 1/1 11/8 6/5

1.25

137

Staff I: +41 +2  
13/8 3/2

Staff II: +41  
13/8

Staff III: +41 +2 +0 +16  
13/8 3/2 1/1 6/5

Staff \*: -31  
III 7/4

141

1.26

Staff I: +16 -31 +4 +2  
6/5 7/4 9/8 1/1 3/2

Staff II: -31 +4  
7/4 9/8

Staff III: +41 +2 -49 +4  
13/8 3/2 11/8 III 9/8

145

Staff I: Treble clef, key of B-flat major. Measures 145-148.

Staff II: Bass clef, key of B-flat major. Measures 145-148. Interval markings: +4, +16, +2.

Staff III: Bass clef, key of B-flat major. Measures 145-148. Interval markings: +0, +41, +2, +4, +16, +0, -49, +2, +41.

Staff \*: Treble clef, key of B-flat major. Measures 145-148. Interval markings: +0, +4, +2, -31.

Measure 145: 1/1, 3/2, 13/8, 9/8, 6/5, 3/2, 1/1, 11/8, 13/8, 1/1, 9/8, 3/2, 7/4.

1.27

149

Staff I: Treble clef, key of B-flat major. Measures 149-152. Interval markings: -49, +41, +16.

Staff II: Bass clef, key of B-flat major. Measures 149-152. Interval markings: -49, +0, +2.

Staff III: Bass clef, key of B-flat major. Measures 149-152. Interval markings: +0.

Staff \*: Treble clef, key of B-flat major. Measures 149-152. Interval markings: +0, +4, +16.

Measure 149: 11/8, 13/8, 6/5, 11/8, 1/1, 3/2, 1/1, 9/8, 6/5, 11/8, 1/1, 3/2, 1/1.

1.28

153

Staff I: Treble clef, key of B-flat major. Measures 153-156. Interval markings: +16, +2, +41.

Staff II: Bass clef, key of B-flat major. Measures 153-156. Interval markings: -31, +2, +41.

Staff III: Bass clef, key of B-flat major. Measures 153-156. Interval markings: +2, +4, +0, -31.

Staff \*: Treble clef, key of B-flat major. Measures 153-156. Interval markings: -49, +41, +0, -31, +0.

Measure 153: 7/4, 6/5, 3/2, 13/8, 3/2, 9/8, 1/1, 7/4, 11/8, 13/8, 1/1, 7/4, 1/1.



*to kill a monarch* (seed:19800725)

[illegible]

161

I

II

III

\*

+41

+2

+4

-49

+0

13/8

9/8

3/2

11/8

1/1

+41

+4

13/8

9/8

+2

+41

-49

+16

+0

3/2

11/8

13/8

6/5

1/1

1.30

I

II

III

\*

165

6/5

6/5

III  $7/4$

3/2

9/8

+16

+2

+4

-31

8

169 1.31

Staff I: -31 +16  
7/4 6/5

Staff II: -31 +41 +4  
7/4 13/8 9/8

Staff III: +2 +0 -31 +41 -49 +16  
3/2 1/1 13/8 11/8 6/5

Staff \*: -31 -49 +2 +16 +4 +0  
7/4 11/8 3/2 6/5 9/8 1/1

173

Staff I: +0 -31  
1/1 7/4

Staff II: +0 -49 +2 +41 +0 -31 +16 +2 +16  
1/1 11/8 3/2 13/8 1/1 7/4 6/5 3/2 6/5

Staff III: +0 +2 -49 +0 +41 +4 -49 +0 +2 +0  
1/1 3/2 11/8 1/1 13/8 9/8 11/8 1/1 3/2 1/1

Staff \*: -31 -49 +4 +41 +2  
III 11/8 9/8 13/8 3/2  
7/4

1.32 177

Staff I: -49 +16 +41 +0 +16  
11/8 6/5 13/8 1/1 6/5

Staff II: +41 +4 +0 +2 +41 -49 +0  
13/8 9/8 1/1 13/8 11/8 1/1

Staff III: -31 +16 +41  
7/4 6/5 13/8

Staff \*: +4 -31 +0 +2 +16 -31 +41 +2 +16  
III 9/8 7/4 1/1 9/8 3/2 6/5 7/4 13/8 3/2 6/5

**1.33**

181

II: +16, 6/5, +2, +4, -31, -49, +4, +16, 3/2, 9/8, 7/4, 11/8, 9/8, 6/5

III: -49, +4, +0, 11/8, 9/8, 1/1

\*: +0, -49, +4, +41, +16, III<sup>1/1</sup>, 11/8, 9/8, 13/8, 6/5

**1.34**

185

I: +0, +2, -49, +41, +2, 1/1, 3/2, 13/8, 3/2, 11/8, +0, 1/1, +16, 6/5

II: -31, +4, +0, 7/4, 9/8, 1/1

III: -49, 11/8

\*: -49, III<sup>11/8</sup>

**1.35**

189

I: +0, 1/1

II: +2, +41, -49, +4, 3/2, 13/8, 11/8, 9/8

III: -31, 7/4

\*: +0, -31, +16, +2, 1/1, 7/4, 6/5, 3/2

193

1.36

Staff I: -31 +16 -49 +0 +4 +2 +16 -49 +0 +41 +2 +0  
7/4 6/5 11/8 1/1 9/8 3/2 6/5 11/8 1/1 13/8 3/2 1/1

Staff II: +41 +2 -31 +0 +16  
13/8 3/2 7/4 1/1 6/5

Staff III: +41 +2 +16 -49 -31 +4 +2 +0 +16 +4 +16 +41 +2  
3/2 6/5 11/8 7/4 9/8 3/2 1/1 6/5 9/8 6/5 13/8 3/2

Staff \*: +4 +16 +0  
9/8 6/5 1/1

197

Staff I: -49 -31 +4 +0 +41 +2 +16  
7/4 9/8 11/8 1/1 13/8 3/2 6/5

Staff II: +2 -49  
3/2 11/8

Staff III: -31 +16 +0 +4  
7/4 6/5 1/1 9/8

Staff \*: +2  
III<sup>3/2</sup>

201

1.37

Staff I: -31 +0 +16 -49 +4  
7/4 6/5 1/1 9/8 11/8

Staff II: -31 +2 +0 +4  
7/4 3/2 1/1 9/8

Staff III: -49 +0  
11/8 1/1

Staff \*: +0  
III<sup>1/1</sup>

205 1.38

I:  $\frac{3}{2}$   $\frac{9}{8}$   $\frac{1}{1}$   $\frac{7}{4}$   $\frac{1}{1}$

II:  $\frac{7}{4}$   $\frac{6}{5}$   $\frac{13}{8}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{11}{8}$   $\frac{3}{2}$

III:  $\frac{7}{4}$   $\frac{3}{2}$   $\frac{11}{8}$   $\frac{1}{1}$   $\frac{3}{2}$   $\frac{6}{5}$

\*:  $\frac{13}{8}$   $\frac{3}{2}$   $\frac{11}{8}$   $\frac{9}{8}$   $\frac{1}{1}$

209

I:  $\frac{6}{5}$   $\frac{13}{8}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{9}{8}$

II:  $\frac{1}{1}$   $\frac{11}{8}$   $\frac{6}{5}$

III:  $\frac{13}{8}$   $\frac{1}{1}$   $\frac{11}{8}$   $\frac{9}{8}$   $\frac{6}{5}$

\*:  $\text{III} \frac{6}{5}$   $\frac{7}{4}$

213 1.39

I:  $\frac{3}{2}$   $\frac{1}{1}$

II:  $\frac{3}{2}$

III:  $\frac{3}{2}$

\*:  $\frac{3}{2}$   $\frac{1}{1}$   $\text{III} \frac{3}{2}$

217 1.40

Staff I:  $+2$ ,  $3/2$

Staff II:  $+2$ ,  $+4$ ,  $-31$ ,  $+0$ ,  $+2$ ,  $+4$ ,  $3/2$ ,  $9/8$ ,  $7/4$ ,  $1/1$ ,  $3/2$ ,  $9/8$

Staff III:  $+4$ ,  $-31$ ,  $+0$ ,  $-49$ ,  $9/8$ ,  $7/4$ ,  $1/1$ ,  $11/8$

Staff \*:  $+16$ ,  $6/5$

221

Staff I:  $-49$ ,  $+0$ ,  $-31$ ,  $+41$ ,  $+16$ ,  $-49$ ,  $11/8$ ,  $1/1$ ,  $7/4$ ,  $13/8$ ,  $6/5$ ,  $11/8$

Staff II:  $+41$ ,  $+0$ ,  $+16$ ,  $13/8$ ,  $1/1$ ,  $6/5$

Staff III:  $+41$ ,  $+0$ ,  $+2$ ,  $13/8$ ,  $1/1$ ,  $3/2$

Staff \*:  $-49$ ,  $11/8$

225 1.41

Staff I:  $+16$ ,  $6/5$

Staff II:  $-49$ ,  $11/8$

Staff III:  $-31$ ,  $+0$ ,  $+4$ ,  $7/4$ ,  $1/1$ ,  $9/8$

Staff \*:  $+2$ ,  $3/2$

229 1.42

Staff I: -31, +0, +4, +41, -31  
7/4, 1/1, 9/8, 13/8, 7/4

Staff II: +41, +16, +0  
13/8, 6/5, 1/1

Staff III: -31, +2  
7/4, 3/2

Staff \*: +41, +4, +0, -31, +4, +0, +4, -49, +2  
III 7/4, 13/8, 9/8, 1/1, 7/4, 9/8, 1/1, 9/8, 11/8, 3/2

233 1.43

Staff I: +2, +16  
3/2, 6/5

Staff II: +16, -31  
6/5, 7/4

Staff III: +16, -49  
6/5, 11/8

Staff \*: +0, +41, +16, +41, +2  
III 1/1, 13/8, 6/5, 13/8, 3/2

237

Staff I: +0, +41, +4  
1/1, 13/8, 9/8

Staff II: +2, +4  
3/2, 9/8

Staff III: +2, +0, +41, -31, +16, -49, +0, +4, +16  
3/2, 1/1, 13/8, 7/4, 6/5, 11/8, 1/1, 9/8, 6/5

Staff \*: +0, -31, -49, +0  
III 1/1, 7/4, 11/8, 1/1

**2.1**

III 9/8 II 3/2 I

+0 +4 +0 +0 +2

9/8 1/1 3/2

(241)

I

7/4 11/8 3/2 1/1 9/8 3/2 1/1

II

13/8

III

7/4 11/8 13/8

\*

+0 +4 +8

II<sup>1/1</sup> I<sup>3/2</sup> III<sup>9/8</sup>

-29 -47 +4 +2 +6 +4 +2

**2.2**

III 9/8 II 3/2 I

+4 +0 +2 +2 +0

1/1 3/2 3/2

(245)

I

6/5 13/8 3/2

7/4 6/5 13/8

II

III

1/1 9/8 3/2 7/4

\*

II<sup>11/8</sup> I<sup>13/8</sup>

+18 +42 +4 -31 +16

+41

**2.3**

III 3/2 II 9/8 I

+4 +2 +2 +0 +0 +4

4/3 3/2 9/8

(249)

I

1/1 3/2 11/8 7/4 1/1

3/2

II

1/1

III

3/2

\*

II<sup>9/8</sup> I<sup>1/1</sup> II<sup>3/2</sup> III<sup>9/8</sup> II<sup>11/8</sup>

+2 -49 -31 +4 +0 +6 +8 -49



2.4

1/1 3/2 1/1 3/2 I

III +2 0 +0 +4 +2

1/1 1/1 4/3

253

I

II

III

\*

13/8 9/8

-49 +0 -31 +0 +4 -49 +2

11/8 1/1 7/4 1/1 9/8 11/8 3/2

+18 +6 +42

6/5 9/8 13/8

+6

III 9/8

257

I

II

III

\*

11/8 3/2

+16 +2

6/5 3/2

-29 +4 +2

7/4 3/2 1/1

+18 -47

III 6/5 11/8

2.5

11/8 3/2 12/11 I

III +2 0 +0 +2 -49

3/2 3/2 12/11

261

I

II

III

\*

1/1 6/5 9/8 1/1

-49 -33 -45 -49

-47 +4 -29 +6

11/8 3/2 7/4 9/8

II 9/8

265

2.6

III 1/1 II 1/1 I

+0 -49 +2 -49 -49

11/8 12/11 1/1

I

-8 -49 -45

3/2 6/5 9/8

13/8 1/1

II

-29 +42 +18

7/4 13/8 6/5

III

-49 +2

11/8 3/2

\*

+18 +3

6/5 11/8

269

I

-33 +20 -49 -47 -49 +3 -47

7/4 1/1 3/2 1/1 11/8 3/2

6/5 11/8

II

-49 -45 -33 +3 -49 -8 +3 -49 -47

1/1 9/8 6/5 11/8 1/1 11/8 1/1 3/2

13/8

III

-49 -33 -49 -47 +20 -33 -47 +20 -49 +3

1/1 6/5 9/8 1/1 3/2 7/4 6/5 3/2 7/4 1/1 11/8

\*

-45 -49

III 9/8 1/1

273

I

-33 -49

6/5 1/1

II

-33 +20 +3 -8 -45 -49

6/5 7/4 11/8 13/8 9/8 1/1

III

+20 -45

7/4 9/8

\*

-8 -47 +3 -8 -45 +20 -47

6/5 13/8 3/2 11/8 9/8 7/4 3/2

-23-

**3.1**

III  $\frac{16}{13}$  II  $\frac{13}{12}$  I

-49 -49 -8 -49 -47

1/1 16/13 4/3

(277)

I  $\frac{7}{4}$   $\frac{1}{1}$   $\frac{9}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{11}{8}$

II  $\frac{11}{8}$

III  $\frac{6}{5}$   $\frac{3}{2}$

\*

+22 -47 -43 -45 -47 +5

+43

**3.2**

III  $\frac{13}{12}$  II  $\frac{13}{12}$  I

-49 -47 -8 -47

4/3 1/1 1/1

(281)

I  $\frac{13}{8}$

II  $\frac{1}{1}$

III  $\frac{13}{8}$   $\frac{7}{4}$

\*

-6 -8 +20 -8 +20 +40 +5

III  $\frac{13}{8}$   $\frac{7}{4}$  I  $\frac{5}{4}$   $\frac{11}{8}$

(285)

I  $\frac{1}{1}$   $\frac{5}{4}$   $\frac{11}{8}$   $\frac{1}{1}$   $\frac{7}{4}$   $\frac{3}{2}$

II  $\frac{3}{2}$   $\frac{1}{1}$   $\frac{7}{4}$

III  $\frac{1}{1}$   $\frac{7}{4}$

\*

-47 -6 -8 -39 -47 +22 -45

-6 +32 +40 +40 +5 -45

III  $\frac{13}{8}$  II  $\frac{13}{8}$  III  $\frac{5}{4}$   $\frac{5}{4}$   $\frac{11}{8}$   $\frac{3}{2}$

-24-

**3.3**

III 1/1 II 13/12 I

-47 -8 -47 -47 -8

1/1 13/12 13/12

(289)

I

+5

11/8

II

-6

-47 -43 +5 -47 -43 -45 +22

1/1 9/8 11/8 1/1 9/8 3/2 7/4

III

+5

11/8

\*

-6 +22 +32 -4

III 13/8 7/4 I 13/8 9/8

**3.4**

III 1/1 II 13/12 I

-47 -47 -8 -8 -47

1/1 13/12 13/12

(293)

I

-4 +32 -4 -22 -8

9/8 13/8 3/2 5/4 1/1

II

III

-6 +40 -43 -45

13/8 5/4 9/8 3/2

\*

+5 -45 -6

III 11/8 3/2 III 13/8

(297)

I

-45

3/2

II

-8

-22 +32 -39 -8

1/1 13/8 5/4 7/4 1/1

III

+22 -47

7/4 1/1

\*

+22 +32 +40 +40

7/4 II 13/8 III 5/4 5/4

**3.5**

III 13/12 1/1 II 13/12 I

-47 -8 -47 -47 -8

13/12 13/12

301

I

13/8 3/2 9/8 1/1 7/4 11/8 13/8 5/4

II

3/2 5/4 3/2

III

7/4 1/1

\*

15/4 III 11/8 3/2

+32 -22 -6 -4 -8 -39 +43 +32 -22

+40 -45 -45

+22 -47

-22 +5 -45

**3.6**

III 13/12 1/1 II 13/12 I

-47 -8 -47 -8 -47

13/12 1/1 13/12

305

I

II

III

5/4

\*

13/8 I 13/8 III 5/4 I 5/4

+40 -6 +32 +40 -22

309

I

1/1 13/8 9/8 3/2 1/1 5/4 1/1 11/8

II

11/8 9/8 3/2 1/1 13/8 7/4 3/2

III

13/8 3/2

\*

II 11/8 3/2

-47 -6 -43 -45 -47 +40 -47 +5

+5 -43 -47 +5 -6 +22 -45

+32 -6

+5 -45

**3.7**

III 1/1 II 1/1 I

-8 -47 -47 -47

13/12 1/1 1/1

313

I

II

III

\*

+40 -47 -43 -6 +22 -43 +5 -45

1/1 9/8 13/8 7/4 11/8 3/2

5/4

III 9/8

317

I

II

III

\*

+22 -43 -6 +40 -47 -45 +22

7/4 9/8 13/8 5/4 1/1 3/2 7/4

-47 1/1 +40 5/4 +22 7/4

-43 +22 +40 -47 -43 -6 +5

9/8 7/4 5/4 1/1 9/8 13/8 11/8

+5 +22 -47 +40 -47 -6 -45 +22 -6 -47

11/8 7/4 1/1 5/4 13/8 1/1 13/8 7/4 13/8 3/2 1/1

**4.1**

III 1/1 II 5/4 I

-47 -47 +40 -47

1/1 5/4 1/1

321

I

II

III

\*

+40 -47 +40

5/4 1/1 5/4

+22 7/4

+5 11/8

4.2

III 13/10 II 5/4 I 13/8

-47 +40 +40 -6 -47

5/4 13/10 1/1

325

I 11/8 +5

II 1/1 +40 -20 13/8

III 13/8 -6 +5 11/8

\* III 13/8 -6 +22 +8 7/4 II 7/4 +8 7/4

329

I 7/4 +22 -47 -43 1/1 9/8

II 3/2 6/5 -4 +9 +45 11/8

III 5/4 3/2 1/1 13/8 9/8 +42 +26 +40 -20 +43 +8 +42 +8 7/4 3/2 7/4

\* I 11/8 +5 -45 3/2 +26 III 5/4 -6 13/8 +22 +9 7/4 II 6/5

4.3

III 5/4 II 13/10 I 13/8

+40 -6 -47 -47 -6

1/1 13/8 13/8

333

I 3/2 -4

II 9/8 -43 -47 1/1

III

\* III 7/4 +8

337

Staff I: -2, +34, +9, 9/8, 13/8, 6/5

Staff II: -45, +22, -47, 3/2, 7/4, 1/1

Staff III: +26, +40, +8, 5/4, 1/1, 7/4

Staff \*: -4, +34, 1 3/2, 13/8

4.4

Staff I: 13/10, 5/4, 13/8, 1/1, -47, +22, +5, 7/4, 11/8

Staff II: +34, +9, -6, +34, -4, -6, +45, -37, 11/8, 13/8, 7/4, 6/5, 1/1, 13/8, 3/2, 1/1

Staff III: -20, +42, +8, 13/8, 3/2, 7/4

Staff \*: -37, +22, II 7/4, I 7/4

4.5

Staff I: 13/10, 1/1, 1/1, 1/1, -4, 3/2

Staff II: -37, 7/4

Staff III: -6, 1/1

Staff \*: +45, -4, -6, -4, +9, +34, -2, +34, -37, +9, III 11/8, 3/2, 1/1, 3/2, 6/5, 9/8, 13/8, 7/4, 6/5



349

I

II

III

\*

5.1

III 16/11 II 14/11 I

-6 -6 +45 -6 -37

1/1 16/11 8/7

353

I

II

III

\*

III<sup>6/5</sup>

357

I

II

III

\*

**5.2**

III 8/7 II 16/11 I

-6 -37 +45 -6 -37 +45

(361) 8/7 16/11 14/11

I

3/2 5/4 9/8

II

9/8 7/4 3/2

III

7/4 3/2 1/1 13/8 11/8 9/8 6/5 7/4 1/1

\*

I<sup>11/8</sup> II<sup>9/8</sup>

+47 +31 +49

-2 -37 -4

+31 -37 -35 +3 -33 +14 -22 +31 -37

-4 -2

**5.3**

III 8/7 II 1/1 I

-37 -6 -6 -37 +45 -37

(365) 8/7 8/7 14/11

I

1/1

II

1/1 9/8

III

9/8

\*

III<sup>3/2</sup> 13/8 II<sup>7/4</sup>

+45

-37 -33

-2

-35 +3 +31

(369)

I

6/5 7/4 3/2 1/1 7/4 11/8 13/8

II

III

7/4 3/2 7/4 9/8 3/2

\*

9/8 II<sup>9/8</sup>

-35 +14 -22 +31 -35 -37 +31 +14

-37 -4 -37 -2 -4

-33 -2

**5.4**

III 1/1 II 1/1 I

-6 -37 -37 -37

8/7 1/1 1/1

(373)

I

II

III

\*

-22 +14 -35 +3 -37 -22 -33 +14

6/5 11/8 7/4 13/8 1/1 6/5 9/8 11/8

+3 -37 -33 -22 -37

13/8 1/1 9/8 6/5 1/1

-35 -37 +31 +31 -33

III 3/2 1/1 7/4 7/4 9/8

(377)

I

II

III

\*

-33 +31 -37 -22 +14 -33 +3 -37 -35 -22

9/8 7/4 1/1 6/5 9/8 13/8 1/1 3/2 6/5

-35 +3

3/2 13/8

+31

7/4

-37 +3 -22 -35 +14 -33 +31

1/1 13/8 6/5 11/8 9/8 7/4 13/8

**6.1**

III 4/3 II 1/1 I

-37 -35 -37 -37

4/3 1/1 1/1

(381)

I

II

III

\*

-37 +14

1/1 11/8

-22

6/5

+16

11/8

6.2

III 8/5 II 4/3 I

-35 -22 -37 -35

8/5 4/3 1/1

385

I

-22 -35 +14 -37 -33 -35

6/5 3/2 11/8 1/1 9/8 3/2

II

-35 -37 -33 +14 +3 -37 -35

3/2 1/1 9/8 11/8 13/8 1/1 3/2

III

-49 -33 -35

5/4 3/2 1/1

\*

+14 +14

II<sup>11/8</sup> 11/8

389

I

-33 -22

9/8 6/5

II

+33

7/4

III

+47 -35 -20

7/4 5/4 3/2

\*

+3 -32 +47 -33 -22 +14

I<sup>13/8</sup> II<sup>9/8</sup> III<sup>7/4</sup> II<sup>3/2</sup> I<sup>6/5</sup> 11/8

6.3

III 8/5 II 6/5 I

-22 -35 -35 -37 -37 -22

8/5 4/3 4/3 6/5

393

I

-22

1/1

II

-37 -22 -35 +14 +3 -22 +31

1/1 3/2 11/8 13/8 6/5 7/4

III

-22 +30 -32 +5

1/1 11/8 9/8 13/8

\*

+3 -32

13/8 III<sup>9/8</sup>

-33-

**6.4**

III 6/5 II 8/5 I  
-35 -37 -37 -22 -22 -35  
4/3 6/5 8/5

397 +47 -22  
7/4 1/1

-35 +47  
-22 +30  
1/1 11/8  
5/4 7/4

-33 +14 -33  
3/2 1/1 9/8  
9/8 11/8

+47 -33  
II 7/4 I 3/2

**6.5**

III 1/1 II 4/3 I  
-37 -35 -22 -35 -35 -37  
4/3 8/5 4/3

401 +5 +16 -35 -32 +5  
13/8 11/8 1/1 9/8 13/8

+47 -35 -18  
5/4 7/4 9/8

-22  
6/5

+5 -35  
13/8 1/1

+5 +14 +14 +30 -49  
13/8 III 11/8 11/8 II 11/8 III 5/4

**6.6**

III 16/11 II 4/3 I  
-35 +14 -35 -37 -37 -35  
12/11 4/3 4/3

405 -37 -33 +14 +31 -35  
1/1 9/8 11/8 7/4 3/2  
13/8

-22 +14 -35 -37  
6/5 11/8 3/2 1/1

+16 -33 +5 -49  
11/8 3/2 13/8 5/4

-33 -32 +33 -49 -35 +5 -35  
3/2 9/8 7/4 5/4 1/1 13/8 5/4 1/1

+16 -33  
11/8 3/2

409

6.7

12/11

III 12/11 II 1/1 I

+14 -35 -37 +14 -35 +14

12/11 16/11 12/11

I

+5 +16 +33 -49 +33 -35 +16

13/8 11/8 7/4 5/4 7/4 1/1 11/8

II

+14

11/8

III

-17 +16 +14

7/4 3/2 1/1

\*

+33 -17 +14 +0 -49

III 7/4 I 7/4 III 5/4 I 5/4

II 11/8

413

I

+16 +0 +18 +14 +16

5/4 9/8 13/8 3/2 1/1

II

+0 -46 +14 +18

13/8 5/4 1/1 9/8

III

-32 +33 +16 +5 +33 -32 -49 +16 -32

9/8 11/8 13/8 7/4 9/8 5/4 9/8 11/8

\*

-35 -46 -17 +33 +18 +18 +0

II 11/8 13/8 7/4 III 7/4 II 9/8 9/8 5/4

417

6.8

1/1

III 1/1 II 1/1 I

-35 +14 +14 +14

12/11 1/1 1/1

I

-17

7/4

II

-46 +16 +0 -35

13/8 1/1 7/4 11/8

III

-35 +16 -17 -35 +14 +18 -46

11/8 III 3/2 7/4 11/8 1/1 9/8 13/8

\*

-35 -35 -35 -35

11/8 7/4 11/8 1/1 9/8 13/8

-35-

421

I

-35 +0 -46 -17 +18 +16 -17 +14

11/8 5/4 13/8 7/4 9/8 3/2 7/4 1/1

II

+16 +18 +14 -35 +0 +14 +18

1/1 11/8 5/4 1/1 9/8

3/2 9/8

III

+14 +18 -17 +16 +14 -35

1/1 9/8 7/4 3/2 1/1 11/8

\*

-35

11/8

7.1

III 3/2 II 3/2 I

+14 +14 +16 +14

1/1 3/2 1/1

425

I

-35 +16

11/8 3/2

II

-33 +16 -15 +2

11/8 1/1 13/8 5/4 7/4

III

-46 +16 +0 +18 +0

13/8 3/2 5/4 9/8 1/1 5/4

\*

-46 -17

III 13/8 7/4

7.2

III 13/8 II 13/12 I

+14 +16 -46 +14 +16

1/1 13/12 3/2

429

I

+2

5/4

II

III

-17

7/4

\*

-33 +6

II 11/8 II 11/8

433

1/1

-44

3/2

-46

1/1

-35 +0 +16 +14 -46

11/8 5/4 3/2 5/4 1/1

+18

13/8 9/8

-35

+0

III 5/4

11/8

-46

13/8

7.3

13/12

III 3/2 II 13/8 I

+14 +16 -46 +14 +16 -46

3/2 13/8 13/12

437

-46

13/8

-35

11/8

-33

11/8

+2

III 5/4

441

-42 -5

9/8 13/8

+0 -46 +14 +18 +16

5/4 13/8 1/1 9/8 3/2

+16

1/1

7.4

3/2

III 12/11 II 11/8 I

+16 +14 -35 -46 +14

1/1 11/8 13/8



445

-35

11/8

+6 -35 +17 -35 -31 -31 -48 -33

1/1 13/8 1/1 9/8 9/8 11/8 1/1 5/4 3/2

+18 +2

3/2 5/4

-33

III 11/8

7.5

12/11 1/1 12/11 I

+16 -35 12/11 1/1 12/11 3/2

-17

7/4

+16 +2 -33

5/4 1/1 11/8

+6

13/8

-35 +34 -31

1/1 7/4 9/8

-33

11/8

7.6

1/1 1/1 1/1 I

-35 1/1 -35 1/1 +16 -35 12/11

+18

3/2

+16

1/1

-33 -31 -35 -33 -31

3/2 9/8 5/4 1/1 3/2 9/8

-44 -15 -31 -48

1 13/8 7/4 III 9/8 5/4

457

I

II

III

\*

III<sup>9/8</sup>

8.1

III 11/8 II 16/13 I

-35 +17 -35 -35 +6

11/8 1/1 16/13

461

I

II

III

\*

11/8

1/1 13/8 1/1 9/8 7/4 11/8

1/1 9/8 5/4 3/2

465

I

II

III

\*

3/2 9/8 7/4 1/1 13/8

13/8 11/8 7/4 9/8 3/2 1/1 5/4 13/8 3/2 7/4

1/1

13/8 11/8 3/2

## 8.2

8.3 16/13 1/1 16/13 I

477

I

II

III

\*

III<sup>13/8</sup> 7/4

**8.4**

III  $\frac{1}{1}$  II  $\frac{1}{1}$  I  
+6 +6 -35 +6  
1/1 1/1 16/13

(481)

I -35 -33 1/1 3/2 6/5 21 -25 +46 7/4 13/8

II

III -43 11/8 +10 9/8 -25 +6 7/4 11/8 1/1 -43

\* -33 1 3/2 +6 7 1/1 III 1/1

(485)

I -43 11/8 +10 9/8

II +21 6/5 -43 11/8 +10 1/1 9/8 +6 +8 -25 3/2 7/4 1/1

III +21 6/5 +10 9/8 +46 3/2 13/8 +21 6/5 +6 1/1 -25 7/4 +8 3/2

\* +21 6/5 -25 7/4 +8 3/2 +21 6/5

**9.1**

III  $\frac{3}{2}$  II  $\frac{3}{2}$  I  
+6 +6 +8 +6  
1/1 3/2 1/1

(489)

I +8 3/2 +6 1/1

II +21 -43 -25 1/1 +10 9/8 +21 6/5 +46 13/8

III

\* +46 13/8 +10 9/8

The musical score for "The Rose Tree" is presented in four staves, labeled I, II, III, and \*. The key signature is one flat (B-flat), and the time signature is common time (C). The notation includes various musical symbols such as notes, rests, and accidentals. The score is written in a style typical of early 20th-century musical manuscripts.

**Staff I:** The first staff begins with a treble clef and a key signature of one flat. It contains a series of notes with various accidentals and rests. The notation includes a 7/4 time signature, a 6/5 time signature, and a 9/8 time signature. The staff ends with a double bar line.

**Staff II:** The second staff begins with a treble clef and a key signature of one flat. It contains a series of notes with various accidentals and rests. The notation includes a 11/8 time signature, a 9/8 time signature, a 7/4 time signature, a 3/2 time signature, a 13/8 time signature, a 1/1 time signature, a 3/2 time signature, a 11/8 time signature, a 9/8 time signature, and a 1/1 time signature. The staff ends with a double bar line.

**Staff III:** The third staff begins with a bass clef and a key signature of one flat. It contains a series of notes with various accidentals and rests. The notation includes a 6/5 time signature, a 9/8 time signature, a 1/1 time signature, a 13/8 time signature, a 6/5 time signature, a 3/2 time signature, a 11/8 time signature, a 9/8 time signature, a 7/4 time signature, a 13/8 time signature, a 1/1 time signature, and a 6/5 time signature. The staff ends with a double bar line.

**Staff \*:** The fourth staff begins with a treble clef and a key signature of one flat. It contains a series of notes with various accidentals and rests. The notation includes a 9/8 time signature and a 6/5 time signature. The staff ends with a double bar line.

9.4

III 1/1 II 1/1 I  
+10 +8 +10 +6 +10  
1/1 4/3 9/8

505

I -25 7/4 +46 13/8 -43 11/8

II -23 +23 +12 -41 +8 +12 6/5 7/4 9/8 11/8 1/1 3/2 9/8

III -4 +10 -21 5/4 1/1 7/4

\* -39 11/8 +46 13/8

509

I +10 1/1 -21 5/4 +12 +10 1/1 +23 8/5 -21 7/4 +14 9/8 +10 1/1 -39 11/8

II -39 11/8 +10 1/1 +12 3/2 +14 9/8

III -39 +12 11/8 +10 3/2 +23 1/1 +10 8/5 +12 1/1 -21 3/2 +14 7/4 -4 9/8 -39 5/4 11/8

\* -4 11/8 -21 7/4 +12 +14 3/2 -39 9/8 +10 1/1 -21 7/4 -4 5/4 +10 1/1 +23 8/5 -21 7/4 +14 9/8 3/2

10.1

III 3/2 II 12/11 I  
+10 +10 +12 +10 -39  
1/1 3/2 11/8

513

I -4 5/4 +12 11/8

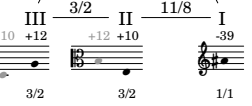
II +10 1/1 +23 8/5

III +10 1/1

\* -4 5/4 +10 1/1 +12 3/2 -39 11/8 -43- 9/8

10.2

12/11



517

I

II

III

\*

III 8/5

8/5

7/4

II 5/4

521

I

II

III

\*

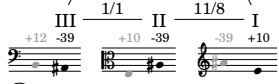
I 11/8

II 9/8

I 7/4

10.3

11/8



525

I

II

III

\*

III 9/8

I 8/5

III 5/4

10.4

1/1

III 55/32 II 55/32 I

-39 -39 +23 +10 -39

1/1 55/32 11/8

529

I

II

III

\*

+12 -37

11/8 3/2

533

I

II

III

\*

-37 +47 +30 -37

5/4 7/4 3/2

3/2

-8 +23 +27 -36 +23 +25 -8 +23 +10 -36 +23

7/4 1/1 9/8 13/8 1/1 3/2 7/4 1/1 5/4 13/8 1/1

+47 -39 +47

5/4 1/1 5/4

-36 -8 +47

II<sup>13/8</sup> 7/4 III<sup>5/4</sup>

5/4

10.5

55/32

1/1

II

55/32

I

-39

+23

1/1

1/1

1/1

537

I

II

III

\*

+2 +12 -39 +30

13/8 11/8 1/1 7/4

11/8 3/2

+47 +12 -37

5/4 11/8 3/2

+25

-25 +10

3/2 11/8 5/4

+27

9/8

+12 -25 +25 +37 -36 -8

I<sup>11/8</sup> III<sup>11/8</sup> 3/2 8/5 13/8 7/4

-45-



10.6

III  $\frac{1}{1}$  II  $\frac{1}{1}$  I  
+23 +23 -39 +23  
1/1 1/1 55/32

(541)

I  $\frac{1}{1}$   $\frac{13}{8}$   $\frac{9}{8}$  +23 -36

II  $\frac{9}{8}$  +27 -8 +25 +23  $\frac{3}{2}$   $\frac{7}{4}$   $\frac{1}{1}$

III  $\frac{3}{2}$  +25 -36  $\frac{13}{8}$  -8 +23 +25  $\frac{7}{4}$   $\frac{1}{1}$   $\frac{3}{2}$

\*  $\frac{1}{5/4}$  +47 +12  $\frac{11}{8}$  -8  $\text{III } \frac{7}{4}$  +23  $\frac{1}{1}$

(545)

I -8 +10 +23 +27 +25  $\frac{7}{4}$   $\frac{5}{4}$   $\frac{1}{1}$   $\frac{9}{8}$   $\frac{3}{2}$

II +27 -36 -8 +23 -25 +10 +25 -8 -36 -25  $\frac{9}{8}$   $\frac{13}{8}$   $\frac{7}{4}$   $\frac{1}{1}$   $\frac{11}{8}$   $\frac{5}{4}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{13}{8}$   $\frac{11}{8}$

III -25 +10 -8  $\frac{11}{8}$   $\frac{5}{4}$   $\frac{7}{4}$

\* -25 +23 -8 +27 +25 +10 -36 -8 -25 +25 +23  $\frac{11}{8}$   $\frac{1}{1}$   $\frac{7}{4}$   $\frac{9}{8}$   $\frac{3}{2}$   $\frac{5}{4}$   $\frac{7}{4}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{13}{8}$

11.1

III  $\frac{13}{8}$  II  $\frac{1}{1}$  I  
+23 -36 +23 +23  
13/8 1/1 1/1

(549)

I +10 -25 -8 -36  $\frac{5}{4}$   $\frac{11}{8}$   $\frac{7}{4}$   $\frac{13}{8}$

II +27 -25  $\frac{9}{8}$   $\frac{3}{2}$   $\frac{11}{8}$

III -34 -32 +4 -36  $\frac{13}{8}$   $\frac{1}{1}$   $\frac{3}{2}$   $\frac{9}{8}$

\* +10 -36 +10 -25  $\frac{5}{4}$   $\frac{13}{8}$   $\text{II } \frac{5}{4}$   $\frac{11}{8}$

to kill a monarch (seed:19800725)

**11.2**

III 13/8 II 13/8 I

-36 +23 -36 +23

1/1 13/8 1/1

(553)

I +10 -25 11/8 5/4 +23 +25 1/1 3/2

II +10 -8 5/4 7/4 -36 +4 1/1 13/8

III +33 7/4

\* +25 -36 3/2 13/8

(557)

I +10 +27 -36 -8 +25 +23 +25 -25 5/4 9/8 13/8 7/4 1/1 3/2 11/8 3/2

II -36 -50 -32 -36 +4 1/1 5/4 9/8 1/1 13/8

III -32 -36 9/8 1/1

\* -32 -50 III 9/8 5/4

**11.3**

III 13/8 II 5/4 I

-36 -36 +23 +23 +10

1/1 13/8 5/4

(561)

I -50 13/8

II +25 +10 -8 -25 +23 +27 +25 3/2 5/4 7/4 11/8 5/4 9/8 5/4 3/2

III -34 -32 -50 -34 +33 +4 -36 3/2 9/8 5/4 3/2 7/4 13/8 1/1

\* -50 +27 +10 +4 7 5/4 II 9/8 5/4 III 13/8

-47-

11.4

13/10

III

5/4

II

13/8

I

-36 +10

+23

+10 -36

13/10

1/1

13/10

565

I

II

III

\*

-8

7/4

+15

11/8

+25

-39

II 3/2

I 11/8

-25

11/8

+14

-21

9/8

7/4

11.5

13/10

III

13/10

II

1/1

I

+10 -36

+23 +10

-36 +10

13/10

5/4

13/10

569

I

II

III

\*

-50 -32 +15 -36

5/4 9/8 11/8 1/1

-50 -36 -34 -32 +33

5/4 1/1 3/2 9/8 7/4

+10

+25

3/2 5/4

+23

+27

+25

1/1 9/8 3/2

+12

-21

+10

-39

+14

-4

-21

3/2 7/4 1/1 11/8 9/8 5/4 7/4

+10

1/1

-32

9/8

-50

5/4

-21

III 7/4

573

I

II

III

\*

-39

11/8

+10

1/1

-50

13/8

-34

+4

+33

-50

3/2 13/8 7/4 5/4

+14

-4

-39

-32

-50

II 9/8

5/4

11/8

III 9/8

5/4

-48-

**11.6**

III  $\frac{13}{10}$  II  $\frac{13}{10}$  I

-36 +10 +10 -36 +10

13/10 13/10 1/1

(577)

I

II

III

\*

-50

$\text{II}^{\frac{5}{4}}$

-36 -50 -32

1/1  $\frac{5}{4}$   $\frac{9}{8}$

+12 +10 -4

1/1  $\frac{5}{4}$   $\frac{3}{2}$

**11.7**

III  $\frac{1}{1}$  II  $\frac{1}{1}$  I

+10 -36 +10 +10

1/1  $\frac{13}{10}$  1/1

(581)

I

II

III

\*

+14 -4 +10 +12 -4

$\frac{9}{8}$   $\frac{5}{4}$   $\frac{1}{1}$   $\frac{3}{2}$   $\frac{5}{4}$

-50

$\frac{13}{8}$

-21 +14

$\text{III}^{\frac{7}{4}}$   $\frac{9}{8}$

(585)

I

II

III

\*

+14 +10 +12 -50 -21 -39 -4 -21 -50 +14 -39 +12 -4

$\frac{13}{8}$   $\frac{7}{4}$   $\frac{11}{8}$   $\frac{5}{4}$   $\frac{7}{4}$   $\frac{13}{8}$   $\frac{9}{8}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{5}{4}$

-21 +10 -39 -4 +12 +10 -4 -39 +14 -50

$\frac{7}{4}$   $\frac{1}{1}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{5}{4}$   $\frac{11}{8}$   $\frac{9}{8}$   $\frac{13}{8}$

+10 -39 -21 +12 +10 +12

$\frac{1}{1}$   $\frac{7}{4}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{3}{2}$

+12 -39 +10 -4 -21 -39 +14 -50 -4 +10 -50 -21

$\text{III}^{\frac{11}{8}}$   $\frac{3}{2}$   $\frac{5}{4}$   $\frac{7}{4}$   $\frac{11}{8}$   $\frac{9}{8}$   $\frac{13}{8}$   $\frac{5}{4}$   $\frac{1}{1}$   $\frac{13}{8}$   $\frac{7}{4}$

1/1

589

I

II

III

\*

+10

1/1

-50

13/8

+12

-39

-4

+14

3/2

11/8

5/4

9/8

12.1

18/13

9/8

16/13

I

III

II

+10

+14

+10

+10

-50

9/8

1/1

16/13

593

I

II

III

\*

+19

7/4

-21

-4

+10

+12

5/4

7/4

1/1

3/2

+14

1/1

-4

11/5/4

12.2

9/8

18/13

16/13

I

III

II

+14

+10

-50

-50

+10

1/1

16/13

16/13

597

I

II

III

\*

+18

9/8

+18

-9

+12

11/9/8

11/13/8

11/3/2

-50

1/1

-9

13/8

-48

3/2

601

I  $\frac{1}{1}$   $\frac{9}{8}$  +10 +14

II  $\frac{7}{4}$   $\frac{11}{8}$   $\frac{7}{4}$  +19 +2

III  $\frac{5}{4}$   $\frac{3}{2}$  +0 +16

\*  $\frac{7}{4}$   $\frac{11}{8}$   $\frac{5}{4}$   $\frac{9}{8}$   $\frac{13}{8}$  -21 +2 -4 +18 -9

12.3

I  $\frac{1}{1}$   $\frac{18}{13}$   $\frac{18}{13}$   $\frac{16}{13}$  +14 -50 -50 +14 +10 -50

605

I  $\frac{1}{1}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{11}{8}$  +10 +12 +19 +2

II  $\frac{13}{8}$  -46

III  $\frac{11}{8}$  +2

\*  $\frac{7}{4}$   $\frac{11}{8}$  -46

12.4

I  $\frac{1}{1}$   $\frac{18}{13}$   $\frac{18}{13}$   $\frac{16}{13}$  -50 +14 -50 -50 +14 +10 -50

609

I  $\frac{5}{4}$   $\frac{13}{8}$   $\frac{9}{8}$   $\frac{1}{1}$   $\frac{3}{2}$   $\frac{9}{8}$   $\frac{7}{4}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{9}{8}$  +37 -9 -46 -50 -48 -46 +19 -48 +19

II  $\frac{1}{1}$   $\frac{5}{4}$   $\frac{9}{8}$  +18 +14 +0

III  $\frac{7}{4}$   $\frac{13}{8}$   $\frac{1}{1}$   $\frac{3}{2}$  +19 -9 -50 -48

\*  $\frac{9}{8}$   $\frac{5}{4}$   $\frac{11}{8}$  -46 +37 +2

613

I

II

III

\*

1/1 5/4 3/2 7/4 13/8

1/1 7/4 1/1 3/2 5/4

9/8 7/4 3/2 5/4 1/1 11/8 7/4 1/1 13/8 3/2 1/1

III<sup>1/1</sup> 5/4 3/2 9/8 1/1 5/4 7/4 9/8

-50 +37 -48 +19 -9

-50 +19 -50 -48 +37

-46 +19 -48 +37 -50 +2 -50 -9 -48 -50

-50 +37 -48 -46 -50 +37 +19 -46

13.1

5/4 1/1 I

III 5/4 II

-50 +37 -50 -50

5/4 1/1 1/1

617

I

II

III

\*

5/4

11/8 7/4

7/4 11/8 3/2 9/8

3/2 1/1 9/8 11/8 3/2 1/1 5/4 II<sup>3/2</sup>

+2 +19

+19 +2 +38 +40

-48 -50 -46 +2 -48 -50 +37 -48

13.2

10/9 9/8 I

III 5/4 II

+37 -50 -50 -46

1/1 1/1 9/8

621

I

II

III

\*

3/2 1/1 5/4 13/8 1/1

7/4 1/1 13/8

13/8 III<sup>3/2</sup> II<sup>9/8</sup> 5/4

-48 -50 +37 -9 -50

+5 +37 -23

-9 +38 -46 +37

**13.3**

III 9/8 II 5/4 I

+37 -46 -50 -46 +37

10/9 1/1 10/9

625

I

-44 -46 -44 -46 -42 -46 -40 -44

3/2 1/1 3/2 1/1 9/8 7/4 1/1 5/4 3/2

II

+37 -48 -46 -48 -50 +2

5/4 3/2 9/8 3/2 1/1 11/8

III

-48 +37 +38

6/5 1/1 3/2

\*

+2 -48 -44 +38

II 11/8 3/2 I 3/2 III 3/2

629

I

-12 +38 -23 -48 +37 +37 +5 +38 -23 -12 -48 +37

11/8 3/2 13/8 6/5 1/1 9/8 1/1 7/4 3/2 13/8 11/8 6/5 1/1

II

III

+40 -46 -44

5/4 1/1 3/2

\*

-23

I 13/8

**13.4**

III 10/9 II 10/9 I

-46 -50 +37 +37 -46

1/1 5/4 10/9

633

I

+5

7/4

II

+37 -48 -46 +5

5/4 3/2 9/8 7/4

III

+23 -44

7/4 3/2

\*

-23 -46 +37

13/8 II 9/8 5/4

-53-



637

13.5

III 10/9 II 10/9 I

-46 +37 -46

1/1 1/1 1/1

I

9/8 3/2 9/8 1/1

-42 -44 -42 -46

II

13/8 1/1 6/5 3/2

-23 +37 -48 +38

III

9/8 1/1 13/8 5/4

-42 -46 -5 +40

\*

II<sup>9/8</sup> III<sup>11/8</sup> 3/2 II<sup>3/2</sup> 13/8

+40 +5 -44 +38 -23

641

13.6

III 1/1 II 1/1 I

-46 +37 -46 -46

1/1 10/9 1/1

I

13/8 5/4 9/8 7/4

-5 +40 -42 +23

II

1/1

+37

III

11/8 3/2 1/1 7/4 13/8 11/8 13/8 1/1 11/8

+5 -44 -46 +23 -5 +5 -5 -44

\*

III<sup>9/8</sup> III<sup>11/8</sup> 3/2 13/8 II<sup>13/8</sup>

-42 +40 +5 -44 -5 -23

645

I

3/2 1/1 11/8 1/1 3/2 13/8 7/4 11/8 5/4

-44 -46 +5 -46 -5 +23 +5 +40

II

III

5/4 7/4 9/8 1/1 9/8 7/4 13/8 9/8 1/1 5/4 7/4 3/2

+40 +23 -42 -46 -42 +23 -5 -42 -46 +40 +23 -44

\*

III<sup>3/2</sup> 13/8 7/4 1/1 11/8 11/8 3/2 7/4

-44 -5 +23 -46 -42 +5 +5 -44 +23

649 -42

I 9/8

II -5 +5 -46 -42 +23 -44 -5 -44 +40 -46

13/8 11/8 1/1 7/4 3/2 13/8 1/1 3/2 5/4

III -46 -5 +5

1/1 13/8 11/8

\* -46 +40 -5

1/1 5/4 13/8

14.1

III 3/2 II 3/2 I

-46 -46 -44 -46

1/1 3/2 1/1

653

I -46

1/1

II +7 -3 -42 -44 +42 +25 -40 -42

11/8 3/2 1/1 5/4 7/4 9/8 3/2

III -42

9/8

\* -44 +23

3/2 III 7/4

14.2

III 4/3 II 3/2 I

-46 -42 -44 -46

9/8 1/1 1/1

657

I -44 +5 +40

3/2 11/8 5/4

II +7 +25 +42

11/8 7/4 5/4

III -44

3/2

\* +42 -42

II 5/4 III 9/8

14.3

III 4/3 II 1/1 I  
-42 -44 -46 -44  
1/1 1/1 3/2

(661)

I -46 -5 -44 +5  
1/1 13/8 11/8 3/2

II -44  
1/1

III +44 -38 +9 -1 -42 -40  
5/4 9/8 11/8 13/8 1/1 3/2

\* +25 +44  
II 7/4 III 5/4

(665)

I -44 +42 -40 -44 -42  
1/1 5/4 9/8 1/1 3/2

II +7 -40  
11/8 9/8

III +27  
7/4

\* -40 +42 +7 -42  
II 9/8 5/4 11/8 3/2

14.4

III 1/1 II 1/1 I  
-42 -44 -44 -44  
4/3 1/1 1/1

(669)

I -3 +25 +7 +42 -44 -40 -44 -3 +42  
13/8 7/4 11/8 5/4 1/1 9/8 1/1 13/8 5/4

II -42  
3/2

III +9  
11/8

\* +25 -3  
7/4 III 13/8

673

-44 +25

7/4 1/1

-44 -3 -42 +42 -44

1/1 13/8 5/4 3/2 1/1

+42 -40

5/4 9/8

+7 -44 +42 -3 -44 +25 +42

11/8 1/1 5/4 3/2 1/1 7/4 5/4

13/8

677

-42 +7

3/2 11/8

-44

1/1

-44 +25 -42 -3 -42

1/1 7/4 3/2 13/8 3/2

15.1

16/11

III 4/3 II 12/11 I

-44 -44 -42 -44 +7

1/1 4/3 16/11

681

+7 +11 +7 -24 +9 +7

1/1 9/8 1/1 7/4 11/8 3/2 1/1

+44 -1 +27 -40 -42 -1

13/8 5/4 7/4 3/2 1/1 13/8

+25 +7

7/4 11/8

+44 -40 +42 +42 +7

II<sup>5/4</sup> III<sup>9/8</sup> 5/4 5/4 11/8

**15.2**

III  $\frac{12}{11}$  II  $\frac{4}{3}$  I

-44 +7 -42 +7 -44

16/11 1/1 16/11

685

I

7/4 5/4 9/8 11/8 7/4 3/2 5/4 3/2 7/4 11/8 13/8

II

5/4 11/8 3/2 1/1 7/4 5/4

III

3/2 13/8 1/1 3/2 7/4 6/5 1/1 9/8 13/8

\*

III  $\frac{6}{5}$  I  $\frac{7}{4}$  II  $\frac{5}{4}$

+25 +42 -40 +7 +25 -42 -40 +25 +7 -3

+44 +9 -40 +27 -42 +44

+9 +48 +7 +9 -24 +23 +7 +9 +11 +48

+23 +25 +44

**15.3**

III  $\frac{1}{1}$  II  $\frac{14}{11}$  I

+7 -42 +7 -44 +25

1/1 12/11 8/7

689

I

5/4 11/8 1/1

II

1/1 3/2 13/8 6/5

III

III  $\frac{11}{8}$  3/2 7/4 I  $\frac{7}{4}$

-41 +9 -24 -6

+42 +25 -24 +25

+7 +48 +23

**15.4**

III  $\frac{14}{11}$  II  $\frac{14}{11}$  I

+7 +7 +25 +25 +7

1/1 14/11 14/11

693

I

9/8 6/5 13/8 9/8

II

7/4 9/8 13/8 1/1

III

6/5 3/2 1/1 9/8 13/8 1/1

\*

III  $\frac{9}{8}$  6/5 II  $\frac{9}{8}$

+29 -24 +7 +23 +9 +11 +48 +7 +29

+29 +23 +11

697

15.5

III 14/11 1/1 II 14/11 I 1/1

+7 +25 +25 +7

14/11 1/1 1/1

I

II

III

\*

+27 +25 -35 -6 -24 +25 +29

3/2 1/1 13/8 7/4 11/8 1/1 9/8

-24 +23 +9 +11

7/4 13/8 6/5 9/8

+48 -24

III 13/8 7/4

701

I

II

III

\*

+9 -24 +23 +7

3/2 7/4 13/8 6/5 1/1

+27 -35 +25 +11 +29

3/2 13/8 1/1 5/4 9/8

-24 -35 +25

11/8 13/8 1/1

+27 -24 -35 -6 +23 +29

7/4 13/8 7/4 6/5 III 9/8

705

15.6

III 14/11 1/1 II 1/1 I 14/11

+25 +7 +25 +7 +25

14/11 1/1 14/11

I

II

III

\*

-35 -6 -24 -35 +27

7/4 11/8 13/8 3/2

+9

3/2

+48

I 13/8

**15.7**

III  $\frac{1}{1}$  II  $\frac{14}{11}$  I

+7 +25 +25 +25 +7

14/11 1/1 14/11

709

I  $\frac{11}{8}$   $\frac{3}{2}$  -24 +27

II +11 +25  $\frac{5}{4}$   $\frac{1}{1}$

III -24 +23 +7 +48 +11 +23 -24  $\frac{7}{4}$   $\frac{6}{5}$   $\frac{1}{1}$   $\frac{13}{8}$   $\frac{9}{8}$   $\frac{7}{4}$   $\frac{6}{5}$

\* -24 +27  $\text{III} \frac{7}{4}$   $\text{II} \frac{3}{2}$

713

I +7 -24 +48 +11 -24 +7 +9 +23 -41  $\frac{1}{1}$   $\frac{7}{4}$   $\frac{9}{8}$   $\frac{13}{8}$   $\frac{7}{4}$   $\frac{1}{1}$   $\frac{6}{5}$   $\frac{3}{2}$   $\frac{11}{8}$

II +27  $\frac{3}{2}$

III -24  $\frac{11}{8}$

\* -35 -6 +29 +11  $\text{III} \frac{13}{8}$   $\frac{7}{4}$   $\frac{9}{8}$   $\frac{5}{4}$

**15.8**

III  $\frac{1}{1}$  II  $\frac{1}{1}$  I

+25 +25 +7 +25

1/1 1/1 14/11

717

I +11 +25 -35 +29 -24 +11 +27 +29  $\frac{5}{4}$   $\frac{1}{1}$   $\frac{13}{8}$   $\frac{9}{8}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{9}{8}$   $\frac{5}{4}$

II

III -35 +25 +27 +11 +29 +25 +27 -6 +11 +25 +29 -24 +27 +25 -35  $\frac{1}{1}$   $\frac{3}{2}$   $\frac{5}{4}$   $\frac{9}{8}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{5}{4}$   $\frac{1}{1}$   $\frac{9}{8}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{13}{8}$

\* -35 +11  $\text{III} \frac{13}{8}$   $\frac{5}{4}$

721

I  $\frac{5}{4}$   $\frac{1}{1}$

II  $\frac{1}{1}$   $\frac{7}{4}$   $\frac{3}{2}$   $\frac{11}{8}$   $\frac{1}{1}$   $\frac{13}{8}$   $\frac{5}{4}$   $\frac{1}{1}$   $\frac{11}{8}$   $\frac{3}{2}$

III  $\frac{5}{4}$   $\frac{9}{8}$

\*  $\frac{9}{8}$   $\frac{11}{8}$   $\frac{1}{1}$   $\frac{7}{4}$   $\frac{9}{8}$

+11 +25 +29 -35 +11 +27 +25 -24

16.1

III  $\frac{9}{8}$  II  $\frac{1}{1}$  I

+25 +29 +25 +25

$\frac{9}{8}$   $\frac{1}{1}$   $\frac{1}{1}$

725

I  $\frac{13}{8}$

II  $\frac{5}{4}$

III  $\frac{1}{1}$   $\frac{7}{4}$

\*  $\frac{3}{2}$   $\frac{5}{4}$   $\text{III}^{\frac{6}{5}}$   $\frac{6}{5}$   $\text{II}^{\frac{3}{2}}$

+27 +11 +44 +27

-35 +11 -2

16.2

III  $\frac{9}{8}$  II  $\frac{9}{8}$  I

+29 +25 +25 +29 +25

$\frac{9}{8}$   $\frac{9}{8}$   $\frac{1}{1}$

729

I  $\frac{3}{2}$   $\frac{7}{4}$   $\frac{1}{1}$

II  $\frac{7}{4}$   $\frac{9}{8}$   $\frac{1}{1}$   $\frac{13}{8}$   $\frac{5}{4}$

III  $\frac{13}{8}$   $\frac{1}{1}$   $\frac{9}{8}$

\*  $\frac{13}{8}$   $\frac{7}{4}$

+27 -6 +25 -31 +29 +33

-6 -35



733

5/4 13/8 7/4 3/2 11/8 1/1 13/8 7/4

11/8 3/2 1/1 7/4 13/8 11/8 7/4 3/2

11/8

III 9/8 II 6/5 III 13/8 7/4 9/8 5/4 II 6/5

16.3

9/8 9/8 1/1

III 9/8 II 1/1 I

+25 +29 +29 +25 +25

737

9/8 5/4 1/1

5/4 13/8 1/1

6/5 1/1

11/8 II 13/8

741

9/8 5/4 1/1

5/4 13/8 1/1

6/5 1/1

11/8 II 13/8

16.4

18/13

III 9/8 II 16/13 I

+29 +25 +25 -35

1/1 1/1 16/13

745

I

II

III

\*

7/4 3/2 13/8 9/8

5/4

7/4 III 7/4 II 5/4

749

I

II

III

\*

1/1 11/8

11/8 9/8

3/2 1/1

III 9/8 I 13/8 III 6/5 11/8 3/2

16.5

9/8

III 18/13 II 16/13 I

+29 +25 -35 -35 +25

1/1 16/13 16/13

753

I

II

III

\*

5/4 7/4 3/2

1/1

7/4 13/8 9/8

III 7/4 II 11/8 I 5/4 II 13/8

16.6

18/13

III 18/13 II 1/1 I

+29 -35 -35 +29 +25 +29

18/13 18/13 9/8

(757)

I

II

III

\*

-31

9/8

+44 -2 -29

6/5 11/8 1/1 7/4

+29

1/1

-20

11/8

-2

-20

11/8 7/4

+31

+44

-31

13/8

-33

3/2

-31

3/2 9/8

+6

III 13/8

+44

II 6/5

16.7

18/13

III 18/13 II 1/1 I

-35 +29 +29 -35 +29 -35

18/13 18/13 18/13

(761)

I

II

III

\*

-2

7/4

+6

13/8

+17

-35

1/1

11/8

-48

-35

1/1

5/4

+31

+44

6/5

-2

-20

+33

11/8 9/8 7/4

+29

1/1

-20

+31

3/2

-2

III 7/4

(765)

I

II

III

\*

+34

7/4

-31

9/8

-35

1/1

+6

13/8

+17

-33

II 11/8 3/2

+6

13/8

**16.8** 1/1

III 18/13 II 18/13 I  
+29 -35 -35 +29 -35

769 18/13 18/13 1/1

I  $\frac{3}{2}$   $\frac{9}{8}$   $\frac{11}{8}$   $\frac{1}{1}$

II  $\frac{3}{2}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{9}{8}$

III  $\frac{13}{8}$   $\frac{11}{8}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{5}{4}$

\*  $\text{III}^{7/4}$

**16.9** 1/1

III 18/13 II 18/13 I  
-35 +29 -35

773 1/1 1/1 1/1

I

II  $\frac{11}{8}$   $\frac{3}{2}$   $\frac{7}{4}$   $\frac{11}{8}$   $\frac{6}{5}$   $\frac{13}{8}$

III  $\frac{3}{2}$   $\frac{11}{8}$   $\frac{9}{8}$   $\frac{1}{1}$

\*  $\text{II}^{11/8}$   $\frac{3}{2}$   $\text{III}^{5/4}$

**16.10** 1/1

III 1/1 II 1/1 I  
-35 +29 -35 -35

777 1/1 18/13 1/1

I  $\frac{13}{8}$   $\frac{7}{4}$   $\frac{3}{2}$   $\frac{9}{8}$

II  $\frac{3}{2}$   $\frac{1}{1}$   $\frac{7}{4}$   $\frac{9}{8}$   $\frac{3}{2}$   $\frac{1}{1}$   $\frac{11}{8}$

III

\*  $\frac{11}{8}$   $\frac{3}{2}$

to kill a monarch (seed:19800725)

781

I

II

III

\*

1/1 7/4 5/4 11/8 3/2 1/1 9/8

9/8 3/2 1/1

7/4 13/8 7/4 3/2 11/8 5/4 3/2 13/8

III 9/8 11/8

785

I

II

III

\*

13/8 7/4

5/4 1/1 13/8 9/8 11/8

1/1 5/4 13/8 1/1 3/2 9/8

789

I

II

III

\*

## appendix - SuperCollider code and Lilypond template

### tkam\_readme.scd

```
1  /*
2  ----execute
3  Execute tkam_main.scd to run.
4
5
6  ----transport tab
7  The play button will always start from the beginning of the current section.
8
9  The transport buttons allow you to advance by subsection (<,>) and section (<<,>>).
10
11 Tempo change will only go into effect once the enter key or "set tempo" button is pressed.
12
13 The default seed given in the application will generate the first version of the music and score (as provided). Changing the seed will generate a new version with that
14 seed once the "generate" button is pressed. After the new version is generated, new Lilypond files can be generated by pressing the "transcribe" button. This will
15 create a tkam_score.ly file in a folder labeled "seed.[number]" which can be rendered by Lilypond. Note that the file must be rendered from that location as it
16 depends on files in that folder and the "includes" subfolder.
17
18 ----mixer tab
19 This allow individual control of each of the sonic elements. The three parts that can be played on acoustic instruments are automatically muted. The outputs will go out to
20 whatever sound card is being used by the system.
21
22 */
```

### tkam\_main.scd

```
1  (
2  // MAIN LAUNCH (loads necessary files and definitions)
3
4  var appEnvironment;
5
6  //push new environment
7  appEnvironment = Environment.make;
8  appEnvironment.push;
9
10 s.waitForBoot({
11     var preampBusses, accompBusses, postampBusses;
12
13     `hash = Date.getDate.hash.asString;
14     `cRes = 1;
15
16     // load all files
17     "tkam.musical.data.generator.scd".loadRelative;
18     "tkam.sonifier.scd".loadRelative;
19     "tkam.gui.scd".loadRelative;
20     "tkam.transcriber.scd".loadRelative;
21
22     # preampBusses, accompBusses, postampBusses = `allocBusses.value(s);
23     `defineSynths.value(s, preampBusses, accompBusses, postampBusses);
24
25     `genAll = {arg seed;
26         `dUnit = 0.reciprocal;
27         `musicData = `genMusicData.value(seed);
28         `scoreData = `genScoreData.value("musicData[0]");
29         `sectionData = "musicData[4]";
30         `patterns = `genPatterns.value("musicData[0]", "musicData[1]", "musicData[2]", "musicData[3]", `sectionData,
31             preampBusses, accompBusses, postampBusses);
32         `sectionNavDict = "musicData[5]";
33         `isPlaying = false;
34     };
35
36     `patternProxy = EventPatternProxy.new;
37
38     `tempoClock = TempoClock.new(1);
39     `dir = thisProcess.nowExecutingPath.dirname;
40     "loading app".postln;
41     `genAll.value(19800725);
42     `play = Synth.new(\masterPlayerControl. ++ `hash);
43     4.collect({arg p; Synth.new(\clip. ++ `hash, {\bin, accompBusses[p].index, \bus, postampBusses[5].index}});
44     `generateGUI.value(preampBusses, accompBusses, postampBusses);
45     "ready".postln;
46 });
47 appEnvironment.pop;
48 )
```

### tkam\_musical\_data\_generator.scd

```
1  (
2  // DATA GENERATOR - this file IS the piece
3  var frAdd, frDiff, frToFloat, frNearestInList, frCollapse, harmonicDistance,
4  genMode, hdChoose, wchooseDict, collectRoots,
5  initModeState, advanceMode,
6  initTemporalState, genTemporalData,
7  initPartStates, distributeRoots,
8  genEnsemblePart, genAccompPart, genBassPart, genAmpCurve, genMusicData, genScoreData, genPatterns;
9
10
11  //-----FREQUENCY RATIO MATH FUNCTIONS-----
12  //for frequency ratios in the form [numerator.factors, denominator.factors]
13  //we use arrays of factors in order to represent very complex ratios
14
15  //add
16  frAdd = {arg fr0, fr1;
17      var num, den;
18      num = fr0[0] ++ fr1[0];
19      den = fr0[1] ++ fr1[1];
20      [[1] ++ num.difference(den).sort, [1] ++ den.difference(num).sort]
21  };
22
23  //difference
24  frDiff = {arg fr0, fr1;
25      var res;
26      res = frAdd.value(fr0, fr1.reverse);
27      if(frToFloat.value(res) < 1, {res = res.reverse});
28      res
29  };
30
31  //convert to float
32  frToFloat = {arg fr; fr[0].asFloat.product / fr[1].asFloat.product};
33
34  //find nearest in list (not sharing the same root or itself)
35  frNearestInList = {arg frComp, frDict;
36      var frNearest, diffNearest, sub;
37      frNearest = nil;
38      diffNearest = 1000;
39      frDict.reject({arg item;
```

```

40     (item[\root][0] == frComp[1]) ||
41     (item[\fr] == frComp[1])
42   }).keys.asList.sort({arg a, b; harmonicDistance.value(a) < harmonicDistance.value(b)}).do({arg fr;
43     var diff = abs(frToFloat.value(fr) - frToFloat.value(frComp[0]));
44     if(diff < diffNearest, {diffNearest = diff; frNearest = fr});
45   });
46   frNearest
47 };
48
49 //collapse into one octave
50 frCollapse = {arg fr;
51   var res = fr;
52   while({frToFloat.value(res) >= 2}, {res = frAdd.value(res, [1, 2])});
53   while({frToFloat.value(res) < 1}, {res = frAdd.value(res, [2, 1])});
54   res
55 };
56
57 //harmonic distance
58 harmonicDistance = {arg fr; log2(fr[0].asFloat.product * fr[1].asFloat.product)};
59
60
61 //-----GENERATE MODE-----
62 genMode = {arg forceHS = false;
63   var mode, alternateProb;
64   alternateProb = [1, 0].wchoose([if(forceHS, {0}, {1}), 4].normalizeSum);
65   mode = [
66     [1, 1],
67     [9, 8],
68     [[5, 4], [6, 5]].wchoose([3, if(forceHS, {0}, {1})].normalizeSum),
69     [[4, 3], [11, 8]].wchoose([alternateProb, 1].normalizeSum),
70     [3, 2],
71     [[8, 5], [13, 8]].wchoose([alternateProb, 1].normalizeSum),
72     [[15, 8], [7, 4]].wchoose([alternateProb, 1].normalizeSum)
73   ];
74   mode.collect({arg fr; [[1] ++ fr[0].factors, [1] ++ fr[1].factors]});
75 };
76
77 //-----CHOOSE AND COLLECT FUNCTIONS-----
78 hdChoose = {arg mode, exp = 1, weights = [1, 1, 1, 1, 1, 1, 1];
79   var probs;
80   probs = pow((1 / mode.collect({arg fr;
81     harmonicDistance.value(if(fr == [[1], [1]], {[[2], [1]]}, {fr}))))), exp) * weights;
82   mode.wchoose(probs.normalizeSum)
83 };
84
85 wchooseDict = {arg dict, exp = 1, limit = 0, isFR = true;
86   var keyList, probs;
87   keyList = if(isFR, {
88     dict.keys.asList.sort({arg a, b; harmonicDistance.value(a) < harmonicDistance.value(b)});
89   }, {
90     dict.keys.asList.sort({arg a, b; a.convertDigits(2) < b.convertDigits(2)});
91   });
92   probs = keyList.collect({arg key;
93     var count = dict[key][\count];
94     if(count < limit, {0}, {count})
95   });
96   probs = pow(probs, exp);
97   keyList.wchoose(probs.normalizeSum)
98 };
99
100 collectRoots = {arg dict; dict.keys.collect({arg fr;
101   dict[fr][\root][0]}.asList.sort({arg a, b; harmonicDistance.value(a) < harmonicDistance.value(b)});
102 });
103
104 //-----GENERATE MODE SEQUENCE-----
105
106 initModeState = {
107   var curModeState, frSet;
108   curModeState = Dictionary.new;
109   frSet = [[1, 1], [9, 8], [6, 5], [11, 8], [3, 2], [13, 8], [7, 4]].collect({arg fr;
110     [1] ++ fr[0].factors, [1] ++ fr[1].factors});
111   frSet.do({arg fr;
112     var mode, count;
113     mode = genMode.value;
114     count = if(fr == [[1], [1]], {10}, {1});
115     curModeState.add(fr->
116       Dictionary.with(*(\count->count, \mode->frSet, \root->[[[1], [1]], frSet], \mult->fr, \fr->fr))
117   });
118   curModeState;
119 };
120
121 advanceMode = {arg lastModeState, lastCadenceState, forceHS = false;
122   var curModeState, curRoots, lastRoots, lastCadenceRoot, changeCount, modSpeed;
123
124   curModeState = lastModeState.deepCopy;
125   curRoots = collectRoots.value(curModeState);
126   lastRoots = collectRoots.value(lastModeState);
127   lastCadenceRoot = collectRoots.value(lastCadenceState).asList[0];
128   changeCount = 0;
129   modSpeed = if(forceHS, {1}, {[1, 2, 3].wchoose([2, 3, 1].normalizeSum)});
130
131   while({
132     ((curRoots == lastRoots) && (changeCount < modSpeed)) ||
133     ((changeCount < modSpeed) && (curRoots.size > 1))
134   }, {
135     var roots, rootSel, mults, multProbs, multSel, new;
136
137     //bump for length of time its been around
138     curModeState.keysValuesDo({arg key, val;
139       val[\count] = val[\count] + 1;
140       if(val[\count] > 100, {val[\count] = 1});
141     });
142
143     // max 3 roots that were not the last cadencial root and only in the mode of the last cadence
144     roots = curModeState.reject({arg val;
145       (val[\fr] == lastCadenceRoot) ||
146       lastCadenceState.includesKey(val[\fr]).not || //consider two steps out?
147       ((curRoots.size >= 3) && curRoots.includes(val[\fr]).not)
148     });
149
150     rootSel = wchooseDict.value(roots, 1, 2);
151     mults = curModeState[rootSel][\mode];
152     multProbs = mults.collect({arg fr;
153       if(curModeState.keys.includes(frCollapse.value(frAdd.value(rootSel, fr))), {1}, {2})});
154     multSel = hdChoose.value(mults, 0.5, multProbs);
155     new = frCollapse.value(frAdd.value(rootSel, multSel));
156
157     curModeState[rootSel][\count] = curModeState[rootSel][\count] + 2; //bump if gets chosen as a root
158
159     if(curModeState.includesKey(new), {
160       //bump if it gets chosen again
161       curModeState[new][\count] = curModeState[new][\count] + 1;
162       if((curModeState[new][\count] >= 20), {
163         curModeState[new][\root] = [rootSel, curModeState[rootSel][\mode]]
164       }

```

```

165     });
166   }, {
167     var old;
168     //calculate nearest in list that does not share the same root
169     old = frNearestInList.value([new, rootSel], curModeState);
170     if(curModeState[old][\count] >= 20, {
171       var mode, root;
172       root = [rootSel, curModeState[rootSel][\mode]];
173       curModeState.add(new ->
174         Dictionary.with(*[\count->1, \mode->genMode.value(forceHS), \root->root, \mult->multSel, \fr->new]));
175       curModeState.removeAt(old);
176       changeCount = changeCount + 1;
177     })
178   });
179   curRoots = collectRoots.value(curModeState);
180 });
181 curModeState
182 };
183
184
185 //-----GENERATE TEMPORAL FRAMEWORK-----
186 initTemporalState = {
187   Dictionary.with(*[[0, 1], [0, 1], [0, 1]].allTuples.collect({arg tuple;
188     tuple->Dictionary.with(*[\count->1]))))
189 };
190
191 genTemporalData = {arg lastTupleState, modeState, cadenceOverride, noParts = 4;
192   var cadence, curTupleState, timeToNextEvent, tuple, temporalData;
193
194   cadence = if(collectRoots.value(modeState).size == 1, {cadenceOverride.not}, {false});
195   curTupleState = lastTupleState.deepCopy;
196   timeToNextEvent = (64 + 50.rand + if(cadence, {50}, {0})).round(16);
197   tuple = wchooseDict.value(curTupleState, isFR: false);
198   if(cadence, {tuple = [1, 1, 1]});
199   curTupleState = curTupleState.keysValuesDo({arg key, val;
200     curTupleState[key][\count] = val[\count] + 1});
201   curTupleState[tuple][\count] = 0;
202   tuple = if(cadence, {[1, 1, 1, 1]}, {[0] ++ tuple});
203
204   temporalData = noParts.collect({arg p;
205     var flourishDensity, genDensity, flourish, beforeLen, before, after, buffer;
206     flourishDensity = if(tuple[p] == 1, {0.125 + 0.5.rand}, {3});
207     if((p == 0) && cadence.not, {flourishDensity = 3});
208     genDensity = if(p == 0, {5}, {20});
209
210     flourish = (if(cadence, {16}, {8}) + 32.rand).collect({[0, 1].wchoose([flourishDensity, 1].normalizeSum)});
211     buffer = 16.collect({0});
212     beforeLen = ((timeToNextEvent - flourish.size - buffer.size) / if(cadence, {1.25}, {1})).asInteger.rand;
213     before = beforeLen.collect({arg i; [0, 1].wchoose([genDensity, 0.25].normalizeSum)});
214     after = (timeToNextEvent - before.size - flourish.size - buffer.size).collect({[0, 1].wchoose([genDensity, 1].normalizeSum)});
215     flourish = before ++ flourish ++ after;
216     if(flourish.sum == 0, {flourish[flourish.size.rand] = 1});
217     flourish = buffer ++ flourish;
218   });
219
220   [temporalData, curTupleState]
221 };
222
223
224 //-----GENERATE ENSEMBLE PARTS-----
225 initPartStates = {
226   var allRatios;
227   allRatios = [[1, 1], [9, 8], [5, 4], [6, 5], [4, 3], [11, 8], [3, 2], [8, 5], [13, 8], [15, 8], [7, 4]];
228   Dictionary.with(*
229     4.collect({arg part;
230       part->Dictionary.with(*[\multCounts->
231         Dictionary.with(*allRatios.collect({arg fr;
232           [[1] ++ fr[0].factors, [1] ++ fr[1].factors->1])),
233           \noteCount->0, \index->part, \lastFreq->0, \lastFreqRatio->[[1], [1]], \lastDur->0
234         })
235       })
236   )
237 };
238
239 //this is how roots are distributed to the parts
240 distributeRoots = {arg modeState, lastRoots;
241   var roots;
242   roots = modeState.keys.asList.collect({arg fr; modeState[fr][\root]});
243   roots = roots.asBag.contents.asPairs.reverse.clump(2);
244   roots = roots = roots.sort({arg a, b;
245     if(a[0] != b[0], {a[0] > b[0]}, {frToFloat.value(a[1][0]) > frToFloat.value(b[1][0])})}).collect({arg item; item[1]}).wrapExtend(4);
246
247   roots = [roots[0]] ++ roots[..2].scramble;
248   roots = 4.collect({arg part;
249     var root, rootMod, rootFreq, mode;
250     root = roots[part];
251     rootMod = frDiff.value(root[0], lastRoots[part]);
252     rootFreq = 36.midicps * pow(2, [1, 0, 1, 2][part]) * frToFloat.value(root[0]);
253     [root[0], root[1], rootMod, rootFreq]
254   });
255   roots = roots.collect({arg root, r;
256     var rootRels;
257     rootRels = 4.collect({arg p; frDiff.value(root[0], roots[p][0])});
258     rootRels.removeAt(r);
259     root.add(rootRels)
260   });
261   roots
262 };
263
264 genEnsemblePart = {arg partState, modeState, temporalData, roots, part, offset;
265   var trans, root, mults, rootMod, amp, firstChange, cadence, lastInsRef, ensData;
266
267   trans = pow(2, [1, 0, 1, 2][partState[\index]]);
268   # root, mults, rootMod = roots[part];
269   amp = [0, 1, 2, 3].wchoose([0, 2, 2, 2].normalizeSum);
270   firstChange = false;
271   cadence = if(collectRoots.value(modeState).size == 1, {true}, {false});
272   lastInsRef = nil;
273
274   ensData = [];
275   temporalData.do({arg val, ts;
276     var timeStamp, comp, change;
277
278     partState[\lastDur] = partState[\lastDur] + 1;
279     timeStamp = ts + offset;
280     change = [val == 1, (val == 1) && firstChange.not].wchoose([1, 2].normalizeSum);
281     if(
282       (partState[\index] == 0) &&
283       (frToFloat.value(partState[\lastFreqRatio]) >= 4.0) &&
284       (partState[\lastDur] < 16) && cadence.not,
285       {change = false}
286     );
287
288     if(change, {
289       var mult, multWeights, freq, rootFreq, insRef;

```



```

290
291 //this weights notes that are richer and mixes with the DCA algorithm
292 multWeights = multis.collect({arg fr;
293     var comp = frCollapse.value(frAdd.value(root, fr));
294     if(modeState.keys.includes(comp), {3}, {1}) * pow(partState[\multCounts][fr], 1);
295 });
296
297 mult = hdChoose.value(mults, 0.5, multWeights);
298 multis.do({arg fr; partState[\multCounts][fr] = partState[\multCounts][fr] + 1});
299 partState[\multCounts][mult] = 0;
300
301 freq = 36.midicps * trans * frToFloat.value(frAdd.value(root, mult));
302
303 //flute special case
304 if((partState[\index] == 0) && cadence.not, {
305     var mode, continue, freqRatio;
306     mode = modeState.keys.asList.collect({arg fr;
307         [
308             frCollapse.value(frAdd.value(modeState[fr][\root][0], modeState[fr][\mult])),
309             modeState[fr][\root][0], modeState[fr][\mult]
310         ]
311     });
312     mode = mode.sort({arg a, b;
313         case
314             {frToFloat.value(a[0]) != frToFloat.value(b[0])} {frToFloat.value(a[0]) < frToFloat.value(b[0])}
315             {frToFloat.value(a[1]) != frToFloat.value(b[1])} {frToFloat.value(a[1]) < frToFloat.value(b[1])}
316             {true} {frToFloat.value(a[2]) < frToFloat.value(b[2])};
317     });
318     mode = mode ++ mode.collect({arg fr; [frAdd.value(fr[0], [[2], [1]]), fr[1], fr[2]]});
319     mode = mode ++ mode.collect({arg fr; [frAdd.value(fr[0], [[4], [1]]), fr[1], fr[2]]});
320     continue = true;
321     while({continue}, {
322         # freqRatio, root, mult = mode[partState[\noteCount] % 15];
323         freq = 36.midicps * trans * frToFloat.value(frAdd.value([1], [1]), freqRatio));
324         continue = (freq <= partState[\lastFreq]) && (partState[\noteCount] % 15) != 0;
325         partState[\noteCount] = partState[\noteCount] + 1;
326     });
327     partState[\lastFreq] = freq;
328     partState[\lastFreqRatio] = freqRatio;
329     insRef = roots.slice(nil, 0).deepCopy.drop(1).indexOfEqual(root) + 1;
330     insRef = if(lastInsRef != insRef, {lastInsRef = insRef; insRef}, {lastInsRef = insRef; nil});
331 });
332 if((partState[\index] == 0) && cadence, {
333     insRef = if(firstChange.not, {1}, {nil});
334 });
335
336 rootFreq = 36.midicps * trans * frToFloat.value(root);
337
338 if((partState[\index] == 0) && ((partState[\noteCount] % 15) == 1) && cadence.not, {ensData = ensData.add([0, timeStamp - 8, 0, 0, 0])});
339 ensData = ensData.add([freq, timeStamp, amp, mult, insRef]);
340 firstChange = true;
341 partState[\lastDur] = 0;
342 if((partState[\index] == 0) && cadence, {partState[\lastDur] = 32});
343 });
344 };
345 ensData = [[0, ensData[0][1] - 4, 0, 0, 0]] ++ ensData;
346 [ensData, partState]
347 };
348
349
350 //-----GENERATE ELECTRONIC ACCOMPANIMENT-----
351 genAccompPart = {arg modeState, temporalData, offset, trans, part, register;
352     var firstChange, accompData;
353     firstChange = false;
354     accompData = [];
355     temporalData.do({arg val, tS;
356         var change;
357         change = (val == 1, (val == 1) && firstChange.not).wchoose([1, if(part == 0, {5}, {3})].normalizeSum);
358         if(change, {
359             var sel, freq, amp;
360             sel = wchooseDict.value(modeState, 0.1);
361             freq = 48.midicps * trans * frToFloat.value(sel);
362             amp = [0, 1, 2, 3].wchoose([2, 2, 1, 1].normalizeSum);
363
364             accompData = accompData.add([freq, tS + offset, amp, part]);
365             firstChange = true;
366         });
367     });
368     accompData
369 };
370
371
372 //-----GENERATE ELECTRONIC BASS-----
373 genBassPart = {arg root, ampCurve, hi;
374     var freq;
375     freq = if(hi,
376         {36.midicps * frToFloat.value(frCollapse.value(frAdd.value(root, [[3], [2]]))},
377         {36.midicps * frToFloat.value(root)});
378     ampCurve.collect({arg sec, iter; [freq, sec[1]]})
379 };
380
381
382 //-----GENERATE AMP CURVES-----
383 genAmpCurve = {arg temporalData1, temporalData2, offset1, offset2, type;
384     var firsts1, firsts2, delay, attack, decay, release, min, max, env;
385     firsts1 = temporalData1.collect({arg ptd; ptd.indexOf(1)});
386     firsts2 = temporalData2.collect({arg ptd; ptd.indexOf(1)});
387     delay = switch(type)
388     {0} {0}
389     {1} {0}
390     {2} {firsts1.minItem};
391     attack = switch(type)
392     {0} {offset2 - offset1}
393     {1} {offset2 - temporalData2[0].size + firsts2.minItem - offset1}
394     {2} {firsts1.maxItem - firsts1.minItem};
395     decay = switch(type)
396     {0} {0}
397     {1} {firsts2.maxItem - firsts2.minItem}
398     {2} {temporalData1[0].size - firsts1.maxItem};
399     release = switch(type)
400     {0} {0}
401     {1} {temporalData2[0].size - firsts2.maxItem}
402     {2} {(offset2 - temporalData2[0].size) - offset1};
403     min = switch(type)
404     {0} {0.15}
405     {1} {0}
406     {2} {0};
407     max = switch(type)
408     {0} {0.5}
409     {1} {0.65}
410     {2} {1};
411
412     env = Env.dadsr(delay, attack, decay, 0.25, release, curve: \cub).range(min, max);
413     ((delay + attack + decay + release) / 1).asInteger.collect({arg iter; [env.at(iter * 1), offset1 + (iter * 1)]})
414 };

```

```

415
416
417 //-----GENERATE ALL MUSIC DATA-----
418 ~genMusicData = {arg seed;
419   var minTotalDur, minSection1Dur, dUnit, curLen, cadence,
420   ultimateSubsection, ultimateSection, ultimateCadenceCount,
421   minTotalLen, minSection1Len,
422   modeState, temporalState, partStates,
423   lastCadenceTemporalData, lastCadenceState, lastSectionPoint,
424   ensData, accompData, bassData, ampData,
425   sectionData, sectionNavDict,
426   sectionCount, subsectionCount,
427   lastRoots, roots, ampDataTmp;
428
429   thisThread.randSeed = seed;
430
431   # minTotalDur, minSection1Dur, dUnit, curLen, cadence = [23 * 60, 8 * 60, 8.reciprocal, 0, false];
432   # ultimateSubsection, ultimateSection, ultimateCadenceCount = [false, false, 0];
433   # minTotalLen, minSection1Len = [(minTotalDur / dUnit).round(16), (minSection1Dur / dUnit).round(16)];
434   # modeState, temporalState, partStates = [initModeState.value, initTemporalState.value, initPartStates.value];
435   # lastCadenceTemporalData, lastCadenceState, lastSectionPoint = [nil, modeState.deepCopy, 0];
436   # ensData, accompData, bassData, ampData = [4.collect({[]}), 4.collect({6.collect({[]})}), 2.collect({[]}), 3.collect({[]})];
437   # sectionData, sectionNavDict = [Dictionary.new, Dictionary.new];
438   # sectionCount, subsectionCount = [1, 1];
439
440   while({(curLen < minTotalLen) || ((curLen >= minTotalLen) && cadence.not) || ultimateSection.not}, {
441     var temporalData;
442     # temporalData, temporalState = genTemporalData.value(temporalState, modeState, curLen <= minSection1Len);
443
444     collectRoots.value(modeState).collect({arg fr; [fr[0].asFloat.product, fr[1].asFloat.product]}).postln;
445     //modeState.keys.postln;
446     ("-- " ++ sectionCount ++ " " ++ subsectionCount ++ " ----").postln;
447
448     lastRoots = if(curLen == 0, {4.collect({[1], [1]}), {roots.slice(nil, 0)}};
449     roots = distributeRoots.value(modeState, lastRoots);
450
451     sectionData.add((curLen / 4).asInteger->[roots, lastRoots.collect({arg fr, part;
452       [fr, 36.midicps * pow(2, [1, 0, 1, 2][part]) * frToFloat.value(fr)]]}, sectionCount, subsectionCount, cadence, ultimateSubsection));
453     sectionNavDict.add([sectionCount, subsectionCount]->[(curLen / 16 + 1).asInteger]);
454
455     4.do({arg part;
456       var musicData, partState, accompRoutine;
457       # musicData, partState = genEnsemblePart.value(partStates[part], modeState, temporalData[part], roots, part, curLen);
458       ensData[part] = ensData[part] ++ musicData;
459       partStates[part] = partState;
460
461       //use an independent random number generator for the accompaniment
462       accompRoutine = Routine({
463         thisThread.randSeed = Date.seed;
464         6.do({arg register;
465           musicData = genAccompPart.value(modeState, temporalData[part], curLen, pow(2, part + register), part, register);
466           accompData[part][register] = accompData[part][register] ++ musicData;
467         });
468       });
469       accompRoutine.value;
470
471     });
472
473     subsectionCount = subsectionCount + 1;
474
475     if(curLen == 0, {
476       lastCadenceTemporalData = temporalData;
477       lastSectionPoint = curLen;
478     });
479
480     curLen = curLen + temporalData[0].size;
481
482     if(curLen > minSection1Len, {
483       if(collectRoots.value(modeState).size == 1, {
484
485         ampData[0] = ampData[0] ++ genAmpCurve.value(lastCadenceTemporalData, temporalData, lastSectionPoint, curLen, 0);
486         ampDataTmp = genAmpCurve.value(lastCadenceTemporalData, temporalData, lastSectionPoint, curLen, 1);
487         ampData[1] = ampData[1] ++ ampDataTmp;
488         bassData[0] = bassData[0] ++ genBassPart.value(collectRoots.value(modeState).asList[0], ampDataTmp, true);
489
490         if(sectionCount == 1, {
491           ampData[2] = ampData[2] ++ ((curLen - temporalData[0].size) / 1).asInteger.collect({arg iter; [0, iter * 1]});
492           bassData[1] = bassData[1] ++ ((curLen - temporalData[0].size) / 1).asInteger.collect({arg iter; [0, iter * 1]});
493         }, {
494           ampDataTmp = genAmpCurve.value(lastCadenceTemporalData, temporalData, lastSectionPoint, curLen, 2);
495           ampData[2] = ampData[2] ++ ampDataTmp;
496           bassData[1] = bassData[1] ++ genBassPart.value(collectRoots.value(lastCadenceState).asList[0], ampDataTmp, false);
497         });
498
499         sectionData.add((curLen - temporalData[0].size) / 4).asInteger->
500           sectionData[(curLen - temporalData[0].size) / 4].asInteger.put(5, true);
501         (subsectionCount - 1).do({arg subsectionIndex;
502           sectionNavDict.add([sectionCount, subsectionIndex + 1]->[sectionNavDict[[sectionCount, subsectionIndex + 1]][0], subsectionCount - 1]);
503         });
504         # sectionCount, subsectionCount = [sectionCount + 1, 1];
505         # lastCadenceTemporalData, lastCadenceState, lastSectionPoint = [temporalData, modeState, curLen];
506         cadence = true;
507
508         //this should ensure that the final cadence is a HS
509         if(curLen >= minTotalLen, {ultimateCadenceCount = ultimateCadenceCount + 1});
510         ultimateSection = ultimateCadenceCount > 1;
511       }, {
512         cadence = false
513       });
514       modeState = advanceMode.value(modeState, lastCadenceState, curLen >= minTotalLen);
515     });
516   });
517
518   ampDataTmp = genAmpCurve.value(lastCadenceTemporalData, [[1], [1], [1], [1]], lastSectionPoint, lastSectionPoint + 360, 2);
519   ampData[2] = ampData[2] ++ ampDataTmp;
520   bassData[1] = bassData[1] ++ genBassPart.value(collectRoots.value(lastCadenceState).asList[0], ampDataTmp, false);
521
522   [ensData, accompData, bassData, ampData, sectionData, sectionNavDict]
523 };
524 )

```

## tkam\_sonifier.scd

```

1 (
2   var formatPatternData;
3
4   //-----ALLOCATE BUSES-----
5   ~allocBuses = {arg server;
6     var preampBuses, accompBuses, postampBuses;
7     preampBuses = 3.collect({Bus.audio(server, 1)});
8     accompBuses = 4.collect({Bus.audio(server, 1)});
9     postampBuses = 7.collect({Bus.audio(server, 1)});
10    [preampBuses, accompBuses, postampBuses];

```

```

11 }
12
13 //-----DEFINE SYNTHS-----
14 "defineSynths = {arg server, preampBusses, accompBusses, postampBusses;
15   var sdPlayer, sdTransport, sdClick, sdAmpCurve, sdEns, sdAccomp, sdClip, sdBass, sdDiskOut, allSds;
16
17   sdPlayer = SynthDef(\masterPlayerControl. ++ `hash, {
18     var router, sigs, sigsPanned, masterSig, imp;
19
20     sigs = postampBusses.collect({arg bus, i; In.ar(bus) * NamedControl.kr(\vol. ++ i, 1, 0.1) * NamedControl.kr(\mute. ++ i, 1, 0.1)});
21     router = sigs.collect({arg sig, i; NamedControl.kr(\out. ++ i, 0, 0)});
22     sigs.collect({arg sig, i; Out.ar(router[i] - 1, sig * router[i].sign)});
23
24     sigsPanned = sigs.collect({arg sig, i; Pan2.ar(sig, NamedControl.kr(\pan. ++ i, 0, 0.1)});
25     masterSig = Mix.ar(sigsPanned.collect({arg sig, i; sig * abs(router[i].sign - 1)}));
26     masterSig = masterSig * NamedControl.kr(\masterVol, 1, 0.1) * NamedControl.kr(\masterMute, 1, 0.1);
27
28     Out.ar(NamedControl.kr(\masterOut, 0, 0), masterSig);
29
30     imp = Impulse.kr(10);
31     SendReply.kr(imp, '/masterLevels.' ++ `hash, values: [Amplitude.kr(masterSig)]);
32     sigs.collect({arg sig, i; SendReply.kr(imp, '/trackLevel.' ++ i ++ "-" ++ `hash, values: [Amplitude.kr(sig)])});
33   });
34
35   sdTransport = SynthDef(\transport. ++ `hash, {arg measure = 0, beat = 0, section = 0, subsection = 0, gate = 1, dur = 1;
36     SendReply.kr(Impulse.kr(0) * (measure > 0) * (beat > 0), '/measureClock.' ++ `hash, values: [measure, beat, section, subsection]);
37     EnvGen.kr(Env.sine(dur), gate, doneAction: 2);
38   });
39
40   sdClick = SynthDef(\click. ++ `hash, {arg beat = 0, gate = 1, dur = 1;
41     Out.ar(postampBusses[6], 10 * BPF.ar(WhiteNoise.ar * EnvGen.kr(Env.perc(0.01, 0.1), gate), 440 * ((beat <= 1) + 1), 0.02));
42     EnvGen.kr(Env.sine(dur), gate, doneAction: 2);
43   });
44
45   sdAmpCurve = SynthDef(\amp.curve. ++ `hash, {arg amp = 1, dur = 0.1, bus = 0;
46     Out.kr(bus, amp.lag)
47   });
48
49   sdEns = SynthDef(\ens. ++ `hash, {arg freq = 440, amp = 1, dur = 1, gate = 1, bus = 0, ampBus = 0, rel = 0.1;
50     Out.ar(bus, SinOsc.ar(freq, 2pi.rand, 0.1) * amp * Latch.kr(In.kr(ampBus), Impulse.kr(0)) * EnvGen.kr(Env.asr(0.1, 1, rel), gate, doneAction: 2));
51   });
52
53   sdAccomp = SynthDef(\accomp. ++ `hash, {arg freq = 440, amp = 1, sustain = 1, dur = 1, gate = 1, bout = 0, ampBus = 0, rel = 0.01;
54     Out.ar(bout, SinOsc.ar(freq, 2pi.rand, 1) * 0.01 * amp * Latch.kr(In.kr(ampBus), Impulse.kr(0)) * EnvGen.kr(Env.asr(sustain, 1, rel), gate, doneAction: 2));
55   });
56
57   sdClip = SynthDef(\clip. ++ `hash, {arg dur = 1, gate = 1, bin = 0, bus = 0;
58     Out.ar(bus, (In.ar(bin)).clip(0, 1) * 50)
59   });
60
61   sdBass = SynthDef(\bass_mono. ++ `hash, {arg freq = 440, ampBus = 0, bus = 0;
62     Out.ar(bus, (SinOsc.ar(freq) * 0.5 * In.kr(ampBus)))
63   });
64
65   sdDiskOut = SynthDef(\disk.out. ++ `hash, {arg bufnum, inbus;
66     DiskOut.ar(bufnum, In.ar(inbus));
67   });
68
69   allSds = [sdPlayer, sdTransport, sdClick, sdAmpCurve, sdEns, sdAccomp, sdClip, sdBass, sdDiskOut];
70   allSds.do({arg sd; sd.add});
71   allSds
72 };
73
74 // group data by measures for navigation
75 formatPatternData = {arg musData, measureLen, rel, print = false;
76   var dataLen,
77   dataLen = musData[0][0].size + 1;
78   musData.collect({arg partData;
79     var res;
80     res = partData;
81     res = res.collect({arg mData, index; mData.add(if(index != (res.size - 1), {rel}, {5.rand + 5})));
82     res = res.flop;
83     res = res.add(res[1]);
84     res[1] = (res[1].differentiate.drop(1) ++ [10]);
85     res = res.flop ++ measureLen.collect({arg measure; dataLen.collect({0}) ++ [measure * 16]});
86     res = res.sort({arg a, b; a.last < b.last}).flop;
87     res = res.insert(1, (res.last.differentiate.drop(1) ++ [10])).flop;
88     res = res.separate({arg a, b; (a.last / 16).trunc != (b.last / 16).trunc});
89     res.collect({arg measureData; measureData.flop})
90   }).flop
91 };
92
93 //-----GENERATE PATTERNS-----
94 //this generates patterns grouped by measures except for bass data and amp curve data which are much higher resolution
95 //these are used to make playable patterns
96 "genPatterns = {arg ensData, accompData, bassData, ampData, sectionData, preampBusses, accompBusses, postampBusses;
97   var measureLen, ensDataFormatted, accompDataFormatted, bassDataFormatted, ampDataFormatted,
98   dUnit, section, subsection, patterns;
99
100   measureLen = ((
101     ensData.collect({arg partData; partData.last[1]}) ++
102     accompData.flatten.collect({arg partData; partData.last[1]})
103   ).maxItem.ceil(16) / 16).asInteger + 1;
104
105   ensDataFormatted = formatPatternData.value(ensData, measureLen, 0.1, true);
106   accompDataFormatted = formatPatternData.value(accompData.flatten, measureLen, 0.01);
107   dUnit = 8.reciprocal;
108
109   patterns = measureLen.collect({arg measure;
110     if(sectionData[measure * 4] != nil, {
111       section = sectionData[measure * 4][2];
112       subsection = sectionData[measure * 4][3];
113     });
114     Ppar(
115       //check how amplitude is being handled
116       ensDataFormatted[measure].collect({arg musData, p;
117         Pbind(
118           \instrument, \ens. ++ `hash,
119           \freq, Pseq(musData[0].replace(0, Rest(0))),
120           \dur, Pseq(musData[1] * dUnit),
121           \sustain, Pseq(musData[2] * dUnit),
122           \amp, [1, 0.7, 0.5, 0.3][p],
123           \ampBus, preampBusses[0].index,
124           \bus, postampBusses[p].index,
125           \rel, Pseq(musData[6])
126         )
127       }) ++
128       //check how amplitude and attack are being handled
129       accompDataFormatted[measure].collect({arg musData;
130         Pbind(
131           \instrument, \accomp. ++ `hash,
132           \freq, Pseq(musData[0].replace(0, Rest(0))),
133           \dur, Pseq(musData[1] * dUnit),
134

```

```

136         \sustain, Pseq(musData[2] * dUnit),
137         \amp, Pseq(musData[3].collect({arg item; [0, 2, 4, 8][item]} * 0.0125 * 1),
138         \ampBus, preampBusses[0].index,
139         \bout, Pseq(musData[4].collect({arg index; accompBusses[index].index})),
140         \rel, Pseq(musData[5])
141     )
142 } ++
143 [
144     Pbind(
145         \instrument, \transport. ++ ^hash,
146         \measure, measure + 1,
147         \beat, Pseq([1, 2]),
148         \section, section,
149         \subsection, subsection,
150         \dur, 8 * dUnit
151     ),
152     Pbind(
153         \instrument, \click. ++ ^hash,
154         \beat, Pseq([1, 2]),
155         \dur, 8 * dUnit
156     )
157 ]
158 )
159 };
160 [patterns, bassData, ampData]
161 };
162
163
164 //this is a playable pattern based on start measure
165 ^genPlayablePatterns = {arg startMeasure, patterns, preampBusses, accompBusses, postampBusses;
166     Ppar(
167         [Pseq(patterns[0][startMeasure..], 1)] ++
168
169         patterns[2].collect({arg pattern, p;
170             Pmono(\amp.curve. ++ ^hash,
171                 \amp, Pseq(Pseq(pattern.slice(nil, 0)[(startMeasure * 16)..], 1), 1 * ^dUnit, \cub),
172                 \dur, 1 * ^dUnit,
173                 \bus, preampBusses[p].index
174             )
175         }) ++
176         patterns[1].collect({arg pattern, p;
177             Pmono(\bass.mono. ++ ^hash,
178                 \freq, Pseq(Pseq(pattern.slice(nil, 0)[(startMeasure * 16)..], 1),
179                 \dur, 1 * ^dUnit,
180                 \ampBus, preampBusses[p + 1].index,
181                 \bus, postampBusses[4].index
182             )
183         })
184     );
185 };
186
187
188 //-----BOUNCE AUDIO-----
189 //this bounces the audio for use in another DAW or for practice
190 ^bounceAudio = {arg seed;
191     var trackNames, basePath, server, buffers, recDur,
192     preampBusses, nextNode, accompBusses, postampBusses,
193     synths, prePatterns, playablePatterns, score;
194
195     trackNames = ["part.star", "part.III", "part.II", "part.I", "accomp.II", "accomp.I", "click"];
196
197     basePath = ^dir +/ "audio" +/ "seed." ++ seed;
198     basePath.mkdir;
199
200     server = Server(\nrt. ++ ^hash,
201         options: ServerOptions.new
202             .numOutputBusChannels(7)
203             .numInputBusChannels(0)
204     );
205
206     # preampBusses, accompBusses, postampBusses = ^allocBusses.value(s);
207     postampBusses = 7.collect({arg index; Bus.new(rate: 'audio', index: index, numChannels: 1, server: server)});
208     synths = ^defineSynths.value(s, preampBusses, accompBusses, postampBusses);
209
210     prePatterns = ^genPatterns.value("musicData[0]", "musicData[1]", "musicData[2]", "musicData[3]", ^sectionData,
211         preampBusses, accompBusses, postampBusses);
212
213     playablePatterns = ^genPlayablePatterns.value(0, prePatterns, preampBusses, accompBusses, postampBusses);
214
215     recDur = (prePatterns[2][0].size / 8) + 45;
216     score = playablePatterns.asScore(duration: recDur, timeOffset: 0.001);
217     nextNode = score.score.slice(nil, 1).select({arg msg; msg[0] == 9}).slice(nil, 2).maxItem + 1;
218
219     synths.do({arg synth; score.add([0.0, [\d.recv, synth.asBytes]]);});
220
221     4.collect({arg p;
222         score.add([0.0, [\s.new, \clip. ++ ^hash, nextNode, 1, 1, \bin, accompBusses[p].index, \bus, postampBusses[5].index]]);
223         nextNode = nextNode + 1;
224     });
225
226     buffers = 7.do({arg track;
227         score.add([0.0, [\b.alloc, track, 65536, 1]]);
228         score.add([0.0, [\b.write, track, basePath +/ "tkam." ++ trackNames[track] ++ ".wav".standardizePath, "WAV", "int16", 0, 0, 1]]);
229         score.add([0.0, [\s.new, \diskout. ++ ^hash, nextNode, 1, 1, \bufnum, track, \inbus, track]]);
230         score.add([recDur, [\n.free, nextNode]]);
231         score.add([recDur, [\b.close, track]]);
232         score.add([recDur, [\b.free, track]]);
233         nextNode = nextNode + 1;
234     });
235
236     score.sort;
237
238     score.recordNRT(
239         outputFilePath: basePath +/ "tkam.all" ++ ".wav".standardizePath,
240         sampleRate: 44100,
241         headerFormat: "WAV",
242         sampleFormat: "int16",
243         options: server.options,
244         duration: recDur
245     );
246
247     server.remove;
248 }
249 )

```

## tkam\_transcriber.scd

```

1 (
2   var formatMusicData, spellingDict, lyNoteNameStr, lyOctStr, lyFinalizeMusic, lyMeasureDef,
3   lyRelMark, lyRelMarkNote, lyHBracket, lyStaffDef, lyTie,
4   lyNoteName, lyCentDev, lyFreqRatio, lyDur, lyNote, lyBeamOpen, lyBeamClosed,
5   consolidateNotes, consolidateRests;
6

```





```

257 lastMusAtom = nil;
258 measureCount = 0;
259 spellingPref = \sharps;
260 tmpSectionData = nil;
261 part.clump(4).do({arg beat, i;
262   var gSum;
263   gSum = 0;
264   beat.separate({arg a, b;
265     ((a[0] != -1) || (b[0] != -1)) && (a != b)}.do({arg group, g;
266       var noteLength, curMusAtom, freq, freqRatioMult, ref, isSame, isRest, isFirst, isLast,
267       isTied, isMeasureBound, isBeamStart, isBeamEnd;
268
269       noteLength = group.size;
270       gSum = gSum + noteLength;
271       curMusAtom = group[0];
272       freq = curMusAtom[0];
273       freqRatioMult = curMusAtom[1];
274       ref = curMusAtom[2];
275       # isSame, isRest, isFirst, isLast = [curMusAtom == lastMusAtom, freq == -1, g == 0, gSum == 4];
276       # isTied, isMeasureBound = [isSame && isRest.not, isFirst && ((i % 4) == 0)];
277       # isBeamStart, isBeamEnd = [(noteLength != 4) && isFirst, (noteLength != 4) && isLast];
278
279       //add ties
280       if(isTied, {lyStr = lyStr + lyTie.value});
281
282       //add barline and ossia definition
283       if(isMeasureBound, {lyStr = lyStr + lyMeasureDef.value(sectionData[i], insNames[p], p, i)});
284
285       //add note data
286       if(sectionData[i] != nil, {
287         tmpSectionData = sectionData[i];
288       });
289       if(isTied.not, {
290         partLookup = if((p != 0) || [1, 2, 3].includes(ref).not, {p}, {ref});
291         pcRoot = ((tmpSectionData[0][partLookup][3].cpsmidi).round(1) % 12).asInteger;
292         quality = if(tmpSectionData[0][partLookup][1][2] == [[1, 5], [1, 2, 2]], {\major}, {\minor});
293         spellingPref = spellingDict[quality][pcRoot];
294         if(p == 0, {(i / 4).asInteger, partLookup, pcRoot, quality});
295       });
296
297       lyStr = lyStr + lyNote.value(freq, noteLength, freqRatioMult, ref, spellingPref, isSame.not && isRest.not);
298
299       //beam group
300       if(isBeamStart, {lyStr = lyStr + lyBeamOpen.value});
301       if(isBeamEnd, {lyStr = lyStr + lyBeamClosed.value});
302
303       lastMusAtom = curMusAtom;
304     });
305   });
306
307   //wrap music and add staff definitions
308   lyStr = lyFinalizeMusic.value(lyStr, p, insNames[p], insNamesShort[p], insMidi[p], insClef[p]);
309
310   //consolidate notes and rests
311   lyStr = consolidateNotes.value(lyStr, p);
312
313   //write file
314   lyFile.write(lyStr);
315   lyFile.close;
316 };
317
318
319
320 //-----GENERATE SCORE DATA-----
321 ^genScoreData = {arg ensData;
322   var res;
323   res = ensData.collect({arg partData;
324     partData.flop.collect({arg data, d; if(d == 1, {data.differentiate ++ [10]}, {[0] ++ data})});
325   });
326   res.collect({arg part; part.flop});
327 };
328 )

```

## tkam\_gui.scd

```

1 (
2   var clockStringFunc, metronomeStringFunc, metronomeColorFunc, updateTransport, updateSection, updateSubsection,
3   buildGenerator, buildMetronome, buildTransport, buildTempoControl, buildMasterFader, buildTrackFader,
4   buildMasterView, buildFaderView, buildHelpView, currentSection = 1, currentSubsection = 1;
5
6   // these funcs update the elements of the transport panel
7   clockStringFunc = {
8     arg measure, beat;
9     var measureString, beatString, leadSpace;
10    measureString = measure.asInteger.asString;
11    beatString = beat.asInteger.asString;
12    leadSpace = (3 - measureString.size).collect({" "}).join;
13    leadSpace ++ measureString ++ "." ++ beatString
14  };
15
16  // [-30, -105, -104] and [-30, -105, -113] are unicode inverse bullet and normal bullet, respectively
17  metronomeStringFunc = { arg beat; if(beat == 1,
18    {[-30, -105, -104].collect({arg int; int.asAscii}).as(String)},
19    {[-30, -105, -113].collect({arg int; int.asAscii}).as(String)});
20  metronomeColorFunc = { arg beat; if(beat == 1, {Color.red}, {Color.black});};
21
22  updateTransport = {arg clock, metronome, sectionDisplay, measure, beat, section, subsection;
23    sectionDisplay.string = "section: " ++ section.asInteger ++ "." ++ subsection.asInteger;
24    clock.string = clockStringFunc.value(measure, beat);
25    metronome.stringColor = metronomeColorFunc.value(beat);
26    metronome.string = metronomeStringFunc.value(beat);
27    {0.75.wait; {metronome.string = ""}.defer}.fork(`tempoClock, quant: 0);
28  }.inEnvir;
29
30  buildGenerator = {arg view;
31    var ranSeed;
32    HLayout(
33      ranSeed = TextField(view).string("19800725"),
34      Button(view).states([["reset seed"]]).action({ ranSeed.string = "19800725".inEnvir}),
35      Button(view).states([["random seed"]]).action({ ranSeed.string = 50000000.rand.asString.inEnvir}),
36      Button(view).states([["generate"]]).action({
37        {^genAll.value(ranSeed.string.asInteger); ^appStatus.string = "status: ready".fork(AppClock);
38        ^appStatus.string = "status: generating".inEnvir},
39        [^appStatus = StaticText(view).string("status: ready"), stretch: 1],
40        Button(view).states([["transcribe"]]).action({
41          {^transcribe.value("scoreData", "sectionData", ranSeed.value); ^appStatus.string = "status: ready".fork(AppClock);
42          ^appStatus.string = "status: transcribing".inEnvir},
43        Button(view).states([["bounce audio"]]).action({
44          {^bounceAudio.value(ranSeed.value); ^appStatus.string = "status: ready".fork(AppClock);
45          ^appStatus.string = "status: bouncing audio".inEnvir},
46        nil)
47      );
48  };

```

```

49 buildMetronome = {arg win;
50   var clock, metronome, layout;
51
52   clock = StaticText(win).string(" 1.1").font(Font("Liberation Mono", 200));
53   metronome = StaticText(win).string([-30, -105, -104].collect({arg int; int.asAscii})
54     .as(String)).font(Font("Liberation Mono", 300)).stringColor(Color.red);
55
56   layout = HLayout(
57     clock,
58     StaticText(win).string("|").font(Font("Liberation Mono", 200)),
59     metronome
60   );
61
62   [clock, metronome, layout]
63 };
64
65 updateSection = {arg mod, clock, metronome, sectionDisplay, refresh = true, indirect = false;
66   var changeSection;
67   case
68   { (currentSubsection > 1) && (mod < 0) } {
69     currentSubsection = 1;
70   }
71   { (currentSubsection <= 1) && (mod < 0) && (currentSection > 1) } {
72     currentSection = currentSection + mod;
73     if(indirect, {
74       currentSubsection = `sectionNavDict[[currentSection, 1]][1]
75     }, {
76       currentSubsection = 1;
77     })
78   }
79   { (mod > 0) && (`sectionNavDict[[currentSection + mod, 1]] != nil) } {
80     currentSection = currentSection + mod;
81     currentSubsection = 1;
82   };
83
84   if(refresh, {
85     updateTransport.value(clock, metronome, sectionDisplay,
86       `sectionNavDict[[currentSection, currentSubsection]][0], 1,
87       currentSection, currentSubsection
88     );
89   });
90 };
91
92 updateSubsection = {arg mod, clock, metronome, sectionDisplay, refresh = true;
93   if(`sectionNavDict[[currentSection, currentSubsection + mod]] != nil, {
94     currentSubsection = currentSubsection + mod;
95     if(refresh, {
96       updateTransport.value(clock, metronome, sectionDisplay,
97         `sectionNavDict[[currentSection, currentSubsection]][0], 1,
98         currentSection, currentSubsection
99       );
100     });
101   }, {
102     updateSection.value(mod, clock, metronome, sectionDisplay, refresh, true)
103   })
104 };
105
106 buildTransport = {arg win, view, clock, metronome, preampBusses, accompBusses, postampBusses;
107   var sec, subsec, sectionDisplay, layout, player;
108
109   sectionDisplay = StaticText(win).string("section: 1.1").font(Font("Liberation Mono", 70));
110
111   OSCFunc({ arg msg, time;
112     {
113       var measure, beat, section, subsection;
114       # measure, beat, section, subsection = msg[3..];
115       currentSection = sec = section.asInteger;
116       currentSubsection = subsec = subsection.asInteger;
117       updateTransport.value(clock, metronome, sectionDisplay, measure, beat, section, subsection);
118     }.inEnvir.defer;
119   }, 'measureClock.' ++ `hash, s.addr);
120
121   layout = HLayout(
122     Button(view).states([["<<", Color.black]]).action({arg pState; updateSection.value(-1, clock, metronome, sectionDisplay)}.inEnvir),
123     Button(view).states([["<", Color.black]]).action({arg pState; updateSubsection.value(-1, clock, metronome, sectionDisplay)}.inEnvir),
124     Button(view).states([["play", Color.black], ["stop", Color.black, Color.grey]]).action({arg pState;
125       if(pState.value == 1, {
126         player = {
127           var startMeasure = `sectionNavDict[[currentSection, currentSubsection]][0] - 1;
128           `patternProxy.source = `genPlayablePatterns.value(startMeasure, `patterns, preampBusses, accompBusses, postampBusses);
129           Pbind(\instrument, \click. ++ `hash, \beat, Pseq([1, 2, 1, 2]), \dur, 1).play(`tempoClock, quant: 0);
130           [1, 2, 1, 2].do({arg beat;
131             {
132               metronome.stringColor = metronomeColorFunc.value(beat);
133               metronome.string = metronomeStringFunc.value(beat);
134             }.defer;
135             0.75.wait;
136             {metronome.string = ""}.defer;
137             0.25.wait;
138           });
139           `patternProxy.play(`tempoClock, quant: 0)
140         }.fork(`tempoClock, quant: 0)
141       }, {
142         `patternProxy.pause;
143         //player.stop;
144         updateTransport.value(clock, metronome, sectionDisplay,
145           `sectionNavDict[[currentSection, currentSubsection]][0], 1,
146           currentSection, currentSubsection);
147       });
148     }.inEnvir),
149     Button(view).states([[">", Color.black]]).action({arg pState; updateSubsection.value(1, clock, metronome, sectionDisplay)}.inEnvir),
150     Button(view).states([[">>", Color.black]]).action({arg pState; updateSection.value(1, clock, metronome, sectionDisplay)}.inEnvir), nil,
151     sectionDisplay, nil);
152   [sectionDisplay, layout]
153 };
154
155 buildTempoControl = {arg view;
156   var layout, tempoField, address, updateSection;
157   layout = HLayout(
158     tempoField = TextField(view).string("60").action({arg v;
159       var tempo = v.value.asInteger; `tempoClock.tempo = tempo / 60}.inEnvir),
160     Button(view).states([["set tempo"]]).action({arg v; `tempoClock.tempo = tempoField.string.asInteger / 60}.inEnvir),
161     [StaticText(view).string(" "), stretch: 1]);
162   [layout, tempoField]
163 };
164
165 buildMasterFader = {arg view;
166   var trackIndicators, layout, volSlider, muteButton, outMenu;
167
168   trackIndicators = {LevelIndicator()} ! 2;
169
170   OSCFunc.new({arg msg;
171     {trackIndicators[0].value = msg[3].ampdb.linlin(-50, 0, 0, 1)}.defer;
172     {trackIndicators[1].value = msg[4].ampdb.linlin(-50, 0, 0, 1)}.defer
173   }, 'masterLevels.' ++ `hash, s.addr);

```





```

299     master[2].valueAction = settings["master.pan"];
300     master[3].valueAction = settings["master.out"];
301     settings["track.volumes"].do({arg val, v; tracks[v][1].valueAction = val});
302     settings["track.solos"].do({arg val, v; tracks[v][2].valueAction = val});
303     settings["track.mutes"].do({arg val, v; tracks[v][3].valueAction = val});
304     settings["track.pans"].do({arg val, v; tracks[v][4].valueAction = val});
305     settings["track.outs"].do({arg val, v; tracks[v][5].valueAction = val});
306   }, {}, false, basePath);
307 };
308
309 saveButton = Button(view).states-([["save", Color.black]]).action-({
310   Dialog.savePanel({ arg path;
311     var settings, file;
312     settings = "{}\n";
313     settings = settings ++ "\ntempo\": " ++ tempoField.string ++ "\n\n";
314     settings = settings ++ "\nmaster.volume\": " ++ master[1].value ++ "\n\n";
315     settings = settings ++ "\nmaster.mute\": " ++ master[2].value ++ "\n\n";
316     settings = settings ++ "\nmaster.out\": " ++ master[3].value ++ "\n\n";
317     settings = settings ++ "\ntrack.volumes\": [" ++ tracks.collect({arg track; track[1].value}).join(",") ++ "],\n\n";
318     settings = settings ++ "\ntrack.solos\": [" ++ tracks.collect({arg track; track[2].value}).join(",") ++ "],\n\n";
319     settings = settings ++ "\ntrack.mutes\": [" ++ tracks.collect({arg track; track[3].value}).join(",") ++ "],\n\n";
320     settings = settings ++ "\ntrack.pans\": [" ++ tracks.collect({arg track; track[4].value}).join(",") ++ "],\n\n";
321     settings = settings ++ "\ntrack.outs\": [" ++ tracks.collect({arg track; track[5].value}).join(",") ++ "],\n\n";
322     settings = settings ++ "};";
323     file = File(path, "w");
324     file.write(settings);
325     file.close;
326   }, {}, basePath);
327 });
328
329 view.layout.(HLayout(HLayout(master[0], nil, *tracks.slice(nil, 0)), VLayout(nil, saveButton, openButton)))
330 };
331
332 buildHelpView = {arg win;
333   TextView(win).string(File.readString("dir ++ "tkam.readme.scd")).editable(false);
334 };
335
336 ^generateGUI = {arg preampBusses, accompBusses, postampBusses;
337   var win, tabButtonReset, transportButton, mixerButton, helpButton, masterControl, tempoControl, masterView, faderView, helpView, tabs;
338   win = Window("to kill a monarch", Rect(500, 500, 1100, 575), false).front;
339   tabButtonReset = {transportButton.value = 1; mixerButton.value = 1; helpButton.value = 1};
340   masterControl = buildMasterView.value(win, preampBusses, accompBusses, postampBusses);
341   masterView = masterControl[0];
342   tempoControl = masterControl[1];
343   faderView = buildFaderView.value(win, tempoControl);
344   helpView = buildHelpView.value(win);
345
346   win.layout = VLayout(
347     HLayout(
348       HLayout(
349         [
350           transportButton = Button().states-([["transport", Color.white, Color.grey], ["transport", Color.black]]).action-({
351             tabButtonReset.value; transportButton.value = 0; tabs.index = 0 }.inEnvir).value(0), stretch: 1
352         ],
353         [
354           mixerButton = Button().states-([["mixer", Color.white, Color.grey], ["mixer", Color.black]]).action-({
355             tabButtonReset.value; mixerButton.value = 0; tabs.index = 1 }.inEnvir).value(1), stretch: 1
356         ]
357       ),
358       helpButton = Button().states-([["help", Color.white, Color.grey], ["help", Color.black]]).action-({
359         tabButtonReset.value; helpButton.value = 0; tabs.index = 2 }.inEnvir).value(1)
360     ),
361     tabs = StackLayout(masterView, faderView, helpView));
362 };

```

## tkam\_score\_template.ly

```

1 \version "2.19.83"
2
3 #(define (override-color-for-all-grobs color)
4   (lambda (context)
5     (let loop ((x all-grob-descriptions))
6       (if (not (null? x))
7         (let ((grob-name (caar x)))
8           (ly:context-pushpop-property context grob-name 'color color)
9           (loop (cdr x))))))
10
11 #(define-markup-command (relMark layout props mus) (ly:music?)
12   #:properties ((size -2))
13   (interpret-markup layout props
14     #{{
15       \markup {
16         \score {
17           \new Staff { $mus }
18           \layout {
19             \context {
20               \Staff
21               \remove TimeSignature.engraver
22               fontSize = #-2
23               \hide Stem
24               \override TextScript.outside-staff-priority = ##f
25               \override StaffSymbol.staff-space = #(magstep -2)
26               \override StaffSymbol.thickness = #(magstep -2)
27               \override TextScript.self-alignment-X = #-0.4
28               \override TextScript.staff-padding = #1
29             }
30             \context {
31               \Score
32               proportionalNotationDuration = #(ly:make-moment 1/16)
33               \remove "Separating.line.group.engraver"
34               \override SpacingSpanner.strict-note-spacing = ##t
35               \override RehearsalMark.self-alignment-X = #-1
36               \override RehearsalMark.Y-offset = #10
37               \override RehearsalMark.X-offset = #10
38             }
39             \context {
40               \Voice
41               \consists "HorizontalBracket.engraver"
42               \override HorizontalBracket.direction = #UP
43             }
44             indent = 0
45             line-width = 4\cm
46           }
47         }
48       }
49     })
50
51 \paper {
52   #(set-paper-size "a4" 'portrait)
53   top-margin = 1\cm
54   bottom-margin = 1\cm
55   left-margin = 2\cm
56

```

```

57 | ragged-bottom = ##t
58 |
59 | top-system-spacing =
60 | #'((basic-distance . 15 )
61 | (minimum-distance . 15 )
62 | (padding . 0 )
63 | (stretchability . 0))
64 |
65 | system-system-spacing =
66 | #'((basic-distance . 35 )
67 | (minimum-distance . 35 )
68 | (padding . 0 )
69 | (stretchability . 0))
70 |
71 | last-bottom-spacing =
72 | #'((basic-distance . 10 )
73 | (minimum-distance . 10 )
74 | (padding . 0 )
75 | (stretchability . 0))
76 |
77 | %systems-per-page = 3
78 | first-page-number = 1
79 | print-first-page-number = ##t
80 |
81 | print-page-number = ##t
82 | oddHeaderMarkup = \markup { \fill-line { \line { \on-the-fly #not-first-page {\pad-markup #2 { \concat {\italic {"to kill a monarch "}} (seed: xxx)}}}}}
83 | evenHeaderMarkup = \markup { \fill-line { \line { \on-the-fly #not-first-page {\pad-markup #2 { \concat {\italic {"to kill a monarch "}} (seed: xxx)}}}}}
84 | oddFooterMarkup = \markup { \fill-line {
85 |   \concat {
86 |     " "
87 |     \fontsize #1.5
88 |     \on-the-fly #print-page-number-check-first
89 |     \fromproperty #'page:page-number-string
90 |     " "}}}
91 | evenFooterMarkup = \markup { \fill-line {
92 |   \concat {
93 |     " "
94 |     \fontsize #1.5
95 |     \on-the-fly #print-page-number-check-first
96 |     \fromproperty #'page:page-number-string
97 |     " "}}}
98 | }
99 |
100 | \header {
101 |   title = \markup { \italic {to kill a monarch}}
102 |   composer = \markup { \right-column {"michael winter" "(berlin, germany; 2021)"}
103 |   poet = "seed: xxx"
104 |   tagline = ""
105 | }
106 |
107 | #(set-global-staff-size 11)
108 |
109 | \layout {
110 |   indent = 0.0\cm
111 |   line-width = 17.5\cm
112 |   ragged-last = ##f
113 |   ragged-right = ##f
114 |
115 |   \context {
116 |     \Score
117 |     \override BarNumber.stencil = #(make-stencil-circler 0.1 0.25 ly:text-interface::print)
118 |     \override Stem.stemlet-length = #0.75
119 |     \proportionalNotationDuration = #(ly:make-moment 1/16)
120 |     \remove "SeparatingLineGroupEngraver"
121 |     \override RehearsalMark.self-alignment-X = #-1
122 |     \override RehearsalMark.Y-offset = #10
123 |     \override RehearsalMark.X-offset = #-8
124 |     \override RehearsalMark.outside-staff-priority = #0
125 |   }
126 |   \context {
127 |     \Staff
128 |
129 |     \override VerticalAxisGroup.staff-staff-spacing =
130 |       #'((basic-distance . 20 )
131 |         (minimum-distance . 20 )
132 |         (padding . 0 )
133 |         (stretchability . 0))
134 |
135 |     \override VerticalAxisGroup.default-staff-staff-spacing =
136 |       #'((basic-distance . 20 )
137 |         (minimum-distance . 20 )
138 |         (padding . 0 )
139 |         (stretchability . 0))
140 |     \override TextScript.staff-padding = #2
141 |     \override TextScript.self-alignment-X = #0
142 |   }
143 |   \context {
144 |     \StaffGroup
145 |     \name "SemiStaffGroup"
146 |     \consists "SpanBarEngraver"
147 |     \override SpanBar.stencil =
148 |       #(lambda (grob)
149 |         (if (string=? (ly:grob-property grob 'glyph-name) "|")
150 |             (set! (ly:grob-property grob 'glyph-name) ""))
151 |         (ly:span-bar::print grob)))
152 |   }
153 |   \context {
154 |     \Score
155 |     \accepts SemiStaffGroup
156 |   }
157 | }
158 |
159 | \midi { }
160 |
161 |
162 | \score{
163 | \new Score
164 | <<
165 | \new SemiStaffGroup {
166 | <<
167 | \include "includes/part.I.ly"
168 | \include "includes/part.II.ly"
169 | \include "includes/part.III.ly"
170 | >>
171 | }
172 | \include "includes/part.star.ly"
173 | >>
174 |
175 | \layout{}
176 | \midi{}
177 | }

```