*Cthar* Segment.

import abjad

import itertools

import os

import pathlib

import time

import abjadext.rmakers

from MusicMaker import MusicMaker

from AttachmentHandler import AttachmentHandler

from random import random

from random import seed

print('Interpreting file ...')

time\_signatures = [

abjad.TimeSignature(pair) for pair in [

(4, 4), (5, 4), (3, 4), (3, 4), (5, 4), (3, 4),

(4, 4), (4, 4), (5, 4), (4, 4), (4, 4), (4, 4),

(3, 4), (5, 4), (4, 4), (4, 4), (4, 4), (4, 4),

(4, 4), (5, 4), (3, 4), (4, 4), (3, 4), (3, 4),

(5, 4), (4, 4), (3, 4), (4, 4), (4, 4), (5, 4),

(3, 4), (5, 4), (5, 4), (4, 4), (3, 4), (3, 4),

(4, 4), (4, 4), (4, 4), (4, 4), (5, 4), (4, 4),

(5, 4), (4, 4), (4, 4), (5, 4), (5, 4),

]

]

bounds = abjad.mathtools.cumulative\_sums([\_.duration for \_ in time\_signatures])

def reduceMod7(rw):

return [(x % 8) for x in rw]

def reduceMod9(rw):

return [(x % 10) for x in rw]

def reduceMod17(rw):

return [(x % 18) for x in rw]

def reduceMod21(rw):

return [(x % 22) for x in rw]

def reduceMod47(rw):

return [(x % 48) for x in rw]

def cyc(lst):

count = 0

while True:

yield lst[count%len(lst)]

count += 1

def grouper(lst1, lst2):

def cyc(lst):

c = 0

while True:

yield lst[c%len(lst)]

c += 1

lst1 = cyc(lst1)

return [next(lst1) if i == 1 else [next(lst1) for \_ in range(i)] for i in lst2]

seed(1)

cello\_random\_walk\_one = []

cello\_random\_walk\_one.append(-1 if random() < 0.5 else 1)

for i in range(1, 1000):

movement = -1 if random() < 0.5 else 1

value = cello\_random\_walk\_one[i-1] + movement

cello\_random\_walk\_one.append(value)

cello\_random\_walk\_one = [abs(x) for x in cello\_random\_walk\_one]

cello\_chord\_one = [-12, -11.5, -11, -10.5, -10, -9.5, -9, -8.5, -8, -7.5, -7, -6.5, -6, -5.5, -5, -4.5, -4, -3.5, -3, -2.5, -2, -1.5, -1, -0.5, 0, -0.5, -1, -1.5, -2, -2.5, -3, -3.5, -4, -4.5, -5, -5.5, -6, -6.5, -7, -7.5, -8, -8.5, -9, -9.5, -10, -10.5, -11, -11.5, ]

cello\_notes\_one = [cello\_chord\_one[x] for x in reduceMod47(cello\_random\_walk\_one)]

seed(2)

cello\_random\_walk\_two = []

cello\_random\_walk\_two.append(-1 if random() < 0.5 else 1)

for i in range(1, 1000):

movement = -1 if random() < 0.5 else 1

value = cello\_random\_walk\_two[i-1] + movement

cello\_random\_walk\_two.append(value)

cello\_random\_walk\_two = [abs(x) for x in cello\_random\_walk\_two]

cello\_chord\_two = [-24, -11, -20, -6, -12, -6, 0, -11, -6, 4, 0, 6, 0, -11, -6, -24, -8, 0, ]

cello\_notes\_two\_walk = [cello\_chord\_two[x] for x in reduceMod17(cello\_random\_walk\_two)]

map\_1 = [1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 2, 1, ]

cello\_notes\_two = grouper(cello\_notes\_two\_walk, map\_1)

seed(3)

cello\_random\_walk\_three = []

cello\_random\_walk\_three.append(-1 if random() < 0.5 else 1)

for i in range(1, 1000):

movement = -1 if random() < 0.5 else 1

value = cello\_random\_walk\_three[i-1] + movement

cello\_random\_walk\_three.append(value)

cello\_random\_walk\_three = [abs(x) for x in cello\_random\_walk\_three]

cello\_chord\_three = [-24, -20, -15, -14, -4, 5, 11, 19, 26, 37, 39, 42, 39, 37, 26, 19, 11, 5, -4, -14, -15, -20, ]

cello\_notes\_three = [cello\_chord\_three[x] for x in reduceMod21(cello\_random\_walk\_three)]

seed(4)

cello\_random\_walk\_four = []

cello\_random\_walk\_four.append(-1 if random() < 0.5 else 1)

for i in range(1, 2000):

movement = -1 if random() < 0.5 else 1

value = cello\_random\_walk\_four[i-1] + movement

cello\_random\_walk\_four.append(value)

cello\_random\_walk\_four = [abs(x) for x in cello\_random\_walk\_four]

cello\_chord\_four = [-17, -8, -13, -5, 5, -5, -13, -8, ]

map\_2 = [2, 1, 2, 1, 2, 2, 1, 2, 1, 2, 1, 1, 1, 2, 1, 2, 1, ]

cello\_notes\_four\_walk = [cello\_chord\_four[x] for x in reduceMod7(cello\_random\_walk\_four)]

cello\_notes\_four = grouper(cello\_notes\_four\_walk, map\_2)

rmaker\_one = abjadext.rmakers.TaleaRhythmMaker(

talea=abjadext.rmakers.Talea(

counts=[7, 4, 6, 3, 5, 3, 5, 3, 6, 4],

denominator=32,

),

beam\_specifier=abjadext.rmakers.BeamSpecifier(

beam\_divisions\_together=True,

beam\_rests=False,

),

extra\_counts\_per\_division=[0, 1, 0, -1],

tuplet\_specifier=abjadext.rmakers.TupletSpecifier(

trivialize=True,

extract\_trivial=True,

rewrite\_rest\_filled=True,

),

)

rmaker\_two = abjadext.rmakers.TaleaRhythmMaker(

talea=abjadext.rmakers.Talea(

counts=[1, 1, 1, 2, 1, 3, 1, 2, 3],

denominator=16,

),

beam\_specifier=abjadext.rmakers.BeamSpecifier(

beam\_divisions\_together=True,

beam\_rests=False,

),

extra\_counts\_per\_division=[1, 0, -1, 0, 1],

tuplet\_specifier=abjadext.rmakers.TupletSpecifier(

trivialize=True,

extract\_trivial=True,

rewrite\_rest\_filled=True,

),

)

rmaker\_three = abjadext.rmakers.EvenDivisionRhythmMaker(

denominators=[8, 8, 16, 8, 8, 16],

extra\_counts\_per\_division=[0, 1, 0, 0, -1, 0, 1, -1],

tuplet\_specifier=abjadext.rmakers.TupletSpecifier(

trivialize=True,

extract\_trivial=True,

rewrite\_rest\_filled=True,

),

)

attachment\_handler\_one = AttachmentHandler(

starting\_dynamic='p',

ending\_dynamic='mp',

hairpin\_indicator='--',

articulation='accent',

)

attachment\_handler\_two = AttachmentHandler(

starting\_dynamic='fff',

ending\_dynamic='mf',

hairpin\_indicator='>',

articulation='tenuto',

)

attachment\_handler\_three = AttachmentHandler(

starting\_dynamic='mp',

ending\_dynamic='ff',

hairpin\_indicator='<|',

articulation='',

)

#####cello#####

cellomusicmaker\_one = MusicMaker(

rmaker=rmaker\_one,

pitches=cello\_notes\_one,

continuous=True,

attachment\_handler=attachment\_handler\_one,

)

cellomusicmaker\_two = MusicMaker(

rmaker=rmaker\_two,

pitches=cello\_notes\_two,

continuous=True,

attachment\_handler=attachment\_handler\_two,

)

cellomusicmaker\_three = MusicMaker(

rmaker=rmaker\_three,

pitches=cello\_notes\_three,

continuous=True,

attachment\_handler=attachment\_handler\_three,

)

cellomusicmaker\_four = MusicMaker(

rmaker=rmaker\_two,

pitches=cello\_notes\_four,

continuous=True,

attachment\_handler=attachment\_handler\_three,

)

silence\_maker = abjadext.rmakers.NoteRhythmMaker(

division\_masks=[

abjadext.rmakers.SilenceMask(

pattern=abjad.index([0], 1),

),

],

)

bowmaker = MusicMaker(

pitches=[33, ],

rmaker=rmaker\_two,

continuous=True,

)

class MusicSpecifier:

def \_\_init\_\_(self, music\_maker, voice\_name):

self.music\_maker = music\_maker

self.voice\_name = voice\_name

print('Collecting timespans and rmakers ...')

###group one###

voice\_1\_timespan\_list = abjad.TimespanList([

abjad.AnnotatedTimespan(

start\_offset=start\_offset,

stop\_offset=stop\_offset,

annotation=MusicSpecifier(

music\_maker=music\_maker,

voice\_name='Voice 1',

),

)

for start\_offset, stop\_offset, music\_maker in [

[(0, 4), (4, 4), bowmaker],

[(4, 4), (7, 4), bowmaker],

[(12, 4), (15, 4), bowmaker],

[(15, 4), (17, 4), bowmaker],

[(17, 4), (20, 4), bowmaker],

[(23, 4), (25, 4), bowmaker],

[(25, 4), (27, 4), bowmaker],

[(27, 4), (30, 4), bowmaker],

[(32, 4), (36, 4), bowmaker],

[(43, 4), (44, 4), bowmaker],

[(44, 4), (48, 4), bowmaker],

[(48, 4), (51, 4), bowmaker],

[(52, 4), (56, 4), bowmaker],

[(56, 4), (58, 4), bowmaker],

[(62, 4), (64, 4), bowmaker],

[(68, 4), (72, 4), bowmaker],

[(72, 4), (76, 4), bowmaker],

[(76, 4), (78, 4), bowmaker],

[(78, 4), (81, 4), bowmaker],

[(82, 4), (84, 4), bowmaker],

[(84, 4), (87, 4), bowmaker],

[(88, 4), (91, 4), bowmaker],

[(91, 4), (93, 4), bowmaker],

[(94, 4), (99, 4), bowmaker],

[(100, 4), (103, 4), bowmaker],

[(103, 4), (105, 4), bowmaker],

[(106, 4), (110, 4), bowmaker],

[(110, 4), (111, 4), bowmaker],

[(112, 4), (114, 4), bowmaker],

[(114, 4), (119, 4), bowmaker],

[(122, 4), (126, 4), bowmaker],

[(128, 4), (131, 4), bowmaker],

[(132, 4), (134, 4), bowmaker],

[(139, 4), (140, 4), bowmaker],

[(144, 4), (146, 4), bowmaker],

[(146, 4), (149, 4), bowmaker],

[(150, 4), (153, 4), bowmaker],

[(157, 4), (158, 4), bowmaker],

[(158, 4), (162, 4), bowmaker],

[(165, 4), (167, 4), bowmaker],

[(167, 4), (169, 4), bowmaker],

[(174, 4), (176, 4), bowmaker],

[(176, 4), (177, 4), bowmaker],

[(181, 4), (185, 4), bowmaker],

[(185, 4), (186, 4), bowmaker],

]

])

voice\_2\_timespan\_list = abjad.TimespanList([

abjad.AnnotatedTimespan(

start\_offset=start\_offset,

stop\_offset=stop\_offset,

annotation=MusicSpecifier(

music\_maker=music\_maker,

voice\_name='Voice 2',

),

)

for start\_offset, stop\_offset, music\_maker in [

[(0, 4), (4, 4), cellomusicmaker\_two],

[(4, 4), (7, 4), cellomusicmaker\_one],

[(12, 4), (15, 4), cellomusicmaker\_two],

[(15, 4), (17, 4), cellomusicmaker\_one],

[(17, 4), (20, 4), cellomusicmaker\_two],

[(23, 4), (25, 4), cellomusicmaker\_two],

[(25, 4), (27, 4), cellomusicmaker\_one],

[(27, 4), (30, 4), cellomusicmaker\_two],

[(32, 4), (36, 4), cellomusicmaker\_three],

[(43, 4), (44, 4), cellomusicmaker\_two],

[(44, 4), (48, 4), cellomusicmaker\_two],

[(48, 4), (51, 4), cellomusicmaker\_one],

[(52, 4), (56, 4), cellomusicmaker\_one],

[(56, 4), (58, 4), cellomusicmaker\_two],

[(62, 4), (64, 4), cellomusicmaker\_two],

[(68, 4), (72, 4), cellomusicmaker\_three],

[(72, 4), (76, 4), cellomusicmaker\_two],

[(76, 4), (78, 4), cellomusicmaker\_three],

[(78, 4), (81, 4), cellomusicmaker\_two],

[(82, 4), (84, 4), cellomusicmaker\_two],

[(84, 4), (87, 4), cellomusicmaker\_four],#

[(88, 4), (91, 4), cellomusicmaker\_four],

[(91, 4), (93, 4), cellomusicmaker\_one],

[(94, 4), (99, 4), cellomusicmaker\_three],

[(100, 4), (103, 4), cellomusicmaker\_one],

[(103, 4), (105, 4), cellomusicmaker\_one],

[(106, 4), (110, 4), cellomusicmaker\_four],

[(110, 4), (111, 4), cellomusicmaker\_four],

[(112, 4), (114, 4), cellomusicmaker\_three],

[(114, 4), (119, 4), cellomusicmaker\_three],

[(122, 4), (126, 4), cellomusicmaker\_one],

[(128, 4), (131, 4), cellomusicmaker\_three],

[(132, 4), (134, 4), cellomusicmaker\_four],

[(139, 4), (140, 4), cellomusicmaker\_four],

[(144, 4), (146, 4), cellomusicmaker\_four],

[(146, 4), (149, 4), cellomusicmaker\_four],

[(150, 4), (153, 4), cellomusicmaker\_four],#

[(157, 4), (158, 4), cellomusicmaker\_two],

[(158, 4), (162, 4), cellomusicmaker\_three],

[(165, 4), (167, 4), cellomusicmaker\_two],

[(167, 4), (169, 4), cellomusicmaker\_two],

[(174, 4), (176, 4), cellomusicmaker\_three],

[(176, 4), (177, 4), cellomusicmaker\_one],

[(181, 4), (185, 4), cellomusicmaker\_two],

[(185, 4), (186, 4), cellomusicmaker\_three],

]

])

###group two###

voice\_3\_timespan\_list = abjad.TimespanList([

abjad.AnnotatedTimespan(

start\_offset=start\_offset,

stop\_offset=stop\_offset,

annotation=MusicSpecifier(

music\_maker=music\_maker,

voice\_name='Voice 3',

),

)

for start\_offset, stop\_offset, music\_maker in [

[(0, 4), (3, 4), bowmaker],

[(3, 4), (4, 4), bowmaker],

[(4, 4), (5, 4), bowmaker],

[(8, 4), (9, 4), bowmaker],

[(9, 4), (12, 4), bowmaker],

[(12, 4), (15, 4), bowmaker],

[(20, 4), (23, 4), bowmaker],

[(25, 4), (27, 4), bowmaker],

[(27, 4), (29, 4), bowmaker],

[(34, 4), (36, 4), bowmaker],

[(36, 4), (40, 4), bowmaker],

[(40, 4), (43, 4), bowmaker],

[(48, 4), (51, 4), bowmaker],

[(52, 4), (56, 4), bowmaker],

[(58, 4), (60, 4), bowmaker],

[(60, 4), (64, 4), bowmaker],

[(64, 4), (66, 4), bowmaker],

[(72, 4), (76, 4), bowmaker],

[(76, 4), (79, 4), bowmaker],

[(79, 4), (81, 4), bowmaker],

[(81, 4), (82, 4), bowmaker],

[(83, 4), (84, 4), bowmaker],

[(84, 4), (88, 4), bowmaker],

[(88, 4), (89, 4), bowmaker],

[(90, 4), (91, 4), bowmaker],

[(91, 4), (94, 4), bowmaker],

[(94, 4), (96, 4), bowmaker],

[(97, 4), (99, 4), bowmaker],

[(99, 4), (103, 4), bowmaker],

[(104, 4), (106, 4), bowmaker],

[(106, 4), (110, 4), bowmaker],

[(111, 4), (114, 4), bowmaker],

[(115, 4), (117, 4), bowmaker],

[(119, 4), (122, 4), bowmaker],

[(125, 4), (127, 4), bowmaker],

[(127, 4), (129, 4), bowmaker],

[(133, 4), (136, 4), bowmaker],

[(136, 4), (138, 4), bowmaker],

[(143, 4), (146, 4), bowmaker],

[(146, 4), (150, 4), bowmaker],

[(150, 4), (154, 4), bowmaker],

[(154, 4), (155, 4), bowmaker],

[(157, 4), (158, 4), bowmaker],

[(158, 4), (160, 4), bowmaker],

[(164, 4), (167, 4), bowmaker],

[(167, 4), (169, 4), bowmaker],

[(171, 4), (172, 4), bowmaker],

[(172, 4), (174, 4), bowmaker],

[(178, 4), (180, 4), bowmaker],

[(180, 4), (183, 4), bowmaker],

[(185, 4), (189, 4), bowmaker],

]

])

voice\_4\_timespan\_list = abjad.TimespanList([

abjad.AnnotatedTimespan(

start\_offset=start\_offset,

stop\_offset=stop\_offset,

annotation=MusicSpecifier(

music\_maker=music\_maker,

voice\_name='Voice 4',

),

)

for start\_offset, stop\_offset, music\_maker in [

[(0, 4), (3, 4), cellomusicmaker\_one],

[(3, 4), (4, 4), cellomusicmaker\_two],

[(4, 4), (5, 4), cellomusicmaker\_one],

[(8, 4), (9, 4), cellomusicmaker\_one],

[(9, 4), (12, 4), cellomusicmaker\_three],

[(12, 4), (15, 4), cellomusicmaker\_one],

[(20, 4), (23, 4), cellomusicmaker\_two],

[(25, 4), (27, 4), cellomusicmaker\_one],

[(27, 4), (29, 4), cellomusicmaker\_one],

[(34, 4), (36, 4), cellomusicmaker\_two],

[(36, 4), (40, 4), cellomusicmaker\_one],

[(40, 4), (43, 4), cellomusicmaker\_two],

[(48, 4), (51, 4), cellomusicmaker\_two],

[(52, 4), (56, 4), cellomusicmaker\_two],

[(58, 4), (60, 4), cellomusicmaker\_one],

[(60, 4), (64, 4), cellomusicmaker\_one],

[(64, 4), (66, 4), cellomusicmaker\_three],

[(72, 4), (76, 4), cellomusicmaker\_two],

[(76, 4), (79, 4), cellomusicmaker\_one],

[(79, 4), (81, 4), cellomusicmaker\_one],

[(81, 4), (82, 4), cellomusicmaker\_three],

[(83, 4), (84, 4), cellomusicmaker\_two],

[(84, 4), (88, 4), cellomusicmaker\_two],

[(88, 4), (89, 4), cellomusicmaker\_one],

[(90, 4), (91, 4), cellomusicmaker\_one],

[(91, 4), (94, 4), cellomusicmaker\_three],

[(94, 4), (96, 4), cellomusicmaker\_two],

[(97, 4), (99, 4), cellomusicmaker\_two],

[(99, 4), (103, 4), cellomusicmaker\_one],

[(104, 4), (106, 4), cellomusicmaker\_one],

[(106, 4), (110, 4), cellomusicmaker\_three],

[(111, 4), (114, 4), cellomusicmaker\_two],

[(115, 4), (117, 4), cellomusicmaker\_four],#

[(119, 4), (122, 4), cellomusicmaker\_four],

[(125, 4), (127, 4), cellomusicmaker\_four],

[(127, 4), (129, 4), cellomusicmaker\_four],

[(133, 4), (136, 4), cellomusicmaker\_four],

[(136, 4), (138, 4), cellomusicmaker\_four],

[(143, 4), (146, 4), cellomusicmaker\_four],

[(146, 4), (150, 4), cellomusicmaker\_four],

[(150, 4), (154, 4), cellomusicmaker\_four],#

[(154, 4), (155, 4), cellomusicmaker\_one],

[(157, 4), (158, 4), cellomusicmaker\_three],

[(158, 4), (160, 4), cellomusicmaker\_three],

[(164, 4), (167, 4), cellomusicmaker\_two],

[(167, 4), (169, 4), cellomusicmaker\_two],

[(171, 4), (172, 4), cellomusicmaker\_three],

[(172, 4), (174, 4), cellomusicmaker\_one],

[(178, 4), (180, 4), cellomusicmaker\_one],

[(180, 4), (183, 4), cellomusicmaker\_two],

[(185, 4), (189, 4), cellomusicmaker\_two],

[(189, 4), (190, 4), silence\_maker],

]

])

all\_timespan\_lists = {

'Voice 1': voice\_1\_timespan\_list,

'Voice 2': voice\_2\_timespan\_list,

'Voice 3': voice\_3\_timespan\_list,

'Voice 4': voice\_4\_timespan\_list,

}

global\_timespan = abjad.Timespan(

start\_offset=0,

stop\_offset=max(\_.stop\_offset for \_ in all\_timespan\_lists.values())

)

for voice\_name, timespan\_list in all\_timespan\_lists.items():

silences = abjad.TimespanList([global\_timespan])

silences.extend(timespan\_list)

silences.sort()

silences.compute\_logical\_xor()

for silence\_timespan in silences:

timespan\_list.append(

abjad.AnnotatedTimespan(

start\_offset=silence\_timespan.start\_offset,

stop\_offset=silence\_timespan.stop\_offset,

annotation=MusicSpecifier(

music\_maker=None,

voice\_name=voice\_name,

),

)

)

timespan\_list.sort()

for voice\_name, timespan\_list in all\_timespan\_lists.items():

shards = timespan\_list.split\_at\_offsets(bounds)

split\_timespan\_list = abjad.TimespanList()

for shard in shards:

split\_timespan\_list.extend(shard)

split\_timespan\_list.sort()

all\_timespan\_lists[voice\_name] = timespan\_list

score = abjad.Score([

abjad.Staff(lilypond\_type='TimeSignatureContext', name='Global Context'),

abjad.StaffGroup(

[

abjad.Staff([abjad.Voice(name='Voice 1')],name='Staff 1', lilypond\_type='BowStaff',),

abjad.Staff([abjad.Voice(name='Voice 5')],name='Staff 5', lilypond\_type='BeamStaff',),

abjad.Staff([abjad.Voice(name='Voice 2')],name='Staff 2', lilypond\_type='Staff',),

],

name='Staff Group 1',

),

abjad.StaffGroup(

[

abjad.Staff([abjad.Voice(name='Voice 3')],name='Staff 3', lilypond\_type='BowStaff',),

abjad.Staff([abjad.Voice(name='Voice 6')],name='Staff 6', lilypond\_type='BeamStaff',),

abjad.Staff([abjad.Voice(name='Voice 4')],name='Staff 4', lilypond\_type='Staff',),

],

name='Staff Group 2',

)

],

)

for time\_signature in time\_signatures:

skip = abjad.Skip(1, multiplier=(time\_signature))

abjad.attach(time\_signature, skip)

score['Global Context'].append(skip)

print('Making containers ...')

def make\_container(music\_maker, durations):

selections = music\_maker(durations)

container = abjad.Container([])

container.extend(selections)

return container

def key\_function(timespan):

"""

Get the timespan's annotation's rhythm-maker.

If the annotation's rhythm-maker is None, return the silence maker.

"""

return timespan.annotation.music\_maker or silence\_maker

for voice\_name, timespan\_list in all\_timespan\_lists.items():

for music\_maker, grouper in itertools.groupby(

timespan\_list,

key=key\_function,

):

durations = [timespan.duration for timespan in grouper]

container = make\_container(music\_maker, durations)

voice = score[voice\_name]

voice.append(container)

print('Adding Beam Staff ...')

voice\_1\_copy = abjad.mutate(score['Voice 1']).copy()

score['Voice 5'].extend([voice\_1\_copy[:]])

voice\_3\_copy = abjad.mutate(score['Voice 3']).copy()

score['Voice 6'].extend([voice\_3\_copy[:]])

print('Splitting and rewriting ...')

# split and rewite meters

for voice in abjad.iterate(score['Staff Group 1']).components(abjad.Voice):

for i , shard in enumerate(abjad.mutate(voice[:]).split(time\_signatures)):

time\_signature = time\_signatures[i]

abjad.mutate(shard).rewrite\_meter(time\_signature)

for voice in abjad.iterate(score['Staff Group 2']).components(abjad.Voice):

for i , shard in enumerate(abjad.mutate(voice[:]).split(time\_signatures)):

time\_signature = time\_signatures[i]

abjad.mutate(shard).rewrite\_meter(time\_signature)

print('Beaming runs ...')

for voice in abjad.select(score).components(abjad.Voice):

for run in abjad.select(voice).runs():

if 1 < len(run):

specifier = abjadext.rmakers.BeamSpecifier(

beam\_each\_division=True,

)

specifier(abjad.select(run))

abjad.attach(abjad.StartBeam(), run[0])

abjad.attach(abjad.StopBeam(), run[-1])

print('Stopping Hairpins ...')

for staff in abjad.iterate(score['Staff Group 1']).components(abjad.Staff):

for rest in abjad.iterate(staff).components(abjad.Rest):

previous\_leaf = abjad.inspect(rest).leaf(-1)

if isinstance(previous\_leaf, abjad.Note):

abjad.attach(abjad.StopHairpin(), rest)

elif isinstance(previous\_leaf, abjad.Chord):

abjad.attach(abjad.StopHairpin(), rest)

elif isinstance(previous\_leaf, abjad.Rest):

pass

for staff in abjad.iterate(score['Staff Group 2']).components(abjad.Staff):

for rest in abjad.iterate(staff).components(abjad.Rest):

previous\_leaf = abjad.inspect(rest).leaf(-1)

if isinstance(previous\_leaf, abjad.Note):

abjad.attach(abjad.StopHairpin(), rest)

elif isinstance(previous\_leaf, abjad.Chord):

abjad.attach(abjad.StopHairpin(), rest)

elif isinstance(previous\_leaf, abjad.Rest):

pass

#attach instruments and clefs

print('Adding attachments ...')

bar\_line = abjad.BarLine('|.')

section\_bar\_line = abjad.BarLine('||')

metro = abjad.MetronomeMark((1, 8), 60)

markup1 = abjad.Markup(r'\bold { A }')

markup2 = abjad.Markup(r'\bold { B }')

markup3 = abjad.Markup(r'\bold { C }')

markup4 = abjad.Markup(r'\bold { D }')

markup5 = abjad.Markup(r'\bold { E }')

markup6 = abjad.Markup(r'\bold { F }')

mark1 = abjad.RehearsalMark(markup=markup1)

mark2 = abjad.RehearsalMark(markup=markup2)

mark3 = abjad.RehearsalMark(markup=markup3)

mark4 = abjad.RehearsalMark(markup=markup4)

mark5 = abjad.RehearsalMark(markup=markup5)

mark6 = abjad.RehearsalMark(markup=markup6)

def \_apply\_numerators\_and\_tech(staff, nums, tech):

numerators = cyc(nums)

techs = cyc(tech)

for logical\_tie in abjad.select(staff).logical\_ties(pitched=True):

tech = next(techs)

numerator = next(numerators)

bcp = abjad.BowContactPoint((numerator, 5))

technis = abjad.BowMotionTechnique(tech)

for note in logical\_tie:

abjad.attach(bcp, note)

abjad.attach(technis, note)

for run in abjad.select(staff).runs():

abjad.bow\_contact\_spanner(run, omit\_bow\_changes=False)

for voice in abjad.select(score['Voice 1']).components(abjad.Voice):

seed(4)

nums\_random\_walk = []

nums\_random\_walk.append(-1 if random() < 0.5 else 1)

for i in range(1, 1000):

movement = -1 if random() < 0.5 else 1

value = nums\_random\_walk[i-1] + movement

nums\_random\_walk.append(value)

nums\_random\_walk = [abs(x) for x in nums\_random\_walk]

nums\_chord = [0, 5, 3, 1, 4, 2, 5, 4, 3, 2]

num\_list = [nums\_chord[x] for x in reduceMod9(nums\_random\_walk)]

tech\_list = ['ordinario', 'ordinario', 'ordinario', 'ordinario', 'circular', 'circular', 'ordinario', 'ordinario', 'ordinario', 'jete', 'ordinario', 'ordinario', 'ordinario', 'ordinario', 'ordinario', 'jete', 'jete', 'jete', 'jete',]

\_apply\_numerators\_and\_tech(staff=voice, nums=num\_list, tech=tech\_list)

for voice in abjad.select(score['Voice 3']).components(abjad.Voice):

seed(5)

nums\_random\_walk = []

nums\_random\_walk.append(-1 if random() < 0.5 else 1)

for i in range(1, 1000):

movement = -1 if random() < 0.5 else 1

value = nums\_random\_walk[i-1] + movement

nums\_random\_walk.append(value)

nums\_random\_walk = [abs(x) for x in nums\_random\_walk]

nums\_chord = [0, 1, 2, 3, 4, 5, 4, 3, 2, 1]

num\_list = [nums\_chord[x] for x in reduceMod9(nums\_random\_walk)]

tech\_list = ['ordinario', 'ordinario', 'ordinario', 'ordinario', 'circular', 'circular', 'ordinario', 'ordinario', 'ordinario', 'jete', 'ordinario', 'ordinario', 'ordinario', 'ordinario', 'ordinario', 'jete', 'jete', 'jete', 'jete',]

\_apply\_numerators\_and\_tech(staff=voice, nums=num\_list, tech=tech\_list)

def \_apply\_position\_and\_span(staff, poses):

positions = cyc(poses)

for run in abjad.select(staff).runs():

span = abjad.StartTextSpan(

left\_text=abjad.Markup(next(positions)).upright(),

right\_text=abjad.Markup(next(positions)).upright(),

style='dashed-line-with-arrow',

)

abjad.attach(span, run[0])

abjad.attach(abjad.StopTextSpan(), run[-1])

abjad.override(staff).text\_spanner.staff\_padding = 0

for voice in abjad.select(score['Voice 5']).components(abjad.Voice):

pos\_list\_1 = ['st.', 'ord.', 'sp.', 'msp.', 'ord.',]

\_apply\_position\_and\_span(staff=voice, poses=pos\_list\_1)

for voice in abjad.select(score['Voice 6']).components(abjad.Voice):

pos\_list\_2 = ['sp.', 'msp.', 'ord.', 'st.', 'ord.',]

\_apply\_position\_and\_span(staff=voice, poses=pos\_list\_2)

for voice in abjad.select(score['Voice 1']).components(abjad.Voice):

for run in abjad.select(voice).runs():

specifier = abjadext.rmakers.BeamSpecifier(

beam\_each\_division=False,

)

specifier(run)

for voice in abjad.select(score['Voice 3']).components(abjad.Voice):

for run in abjad.select(voice).runs():

specifier = abjadext.rmakers.BeamSpecifier(

beam\_each\_division=False,

)

specifier(run)

instruments1 = cyc([

abjad.Cello(),

])

instruments2 = cyc([

abjad.Cello(),

])

clefs1 = cyc([

abjad.Clef('percussion'),

abjad.Clef('percussion'),

abjad.Clef('bass'),

])

clefs2 = cyc([

abjad.Clef('percussion'),

abjad.Clef('percussion'),

abjad.Clef('bass'),

])

abbreviations1 = cyc([

abjad.MarginMarkup(markup=abjad.Markup('B.H.'),),

abjad.MarginMarkup(markup=abjad.Markup('vc.I'),),

abjad.MarginMarkup(markup=abjad.Markup('L.H.'),),

])

abbreviations2 = cyc([

abjad.MarginMarkup(markup=abjad.Markup('B.H.'),),

abjad.MarginMarkup(markup=abjad.Markup('vc.II'),),

abjad.MarginMarkup(markup=abjad.Markup('L.H.'),),

])

names1 = cyc([

abjad.StartMarkup(markup=abjad.Markup('Bow Hand'),),

abjad.StartMarkup(markup=abjad.Markup('Violoncello I'),),

abjad.StartMarkup(markup=abjad.Markup('Left Hand'),),

])

names2 = cyc([

abjad.StartMarkup(markup=abjad.Markup('Bow Hand'),),

abjad.StartMarkup(markup=abjad.Markup('Violoncello II'),),

abjad.StartMarkup(markup=abjad.Markup('Left Hand'),),

])

for staff in abjad.iterate(score['Staff Group 1']).components(abjad.Staff):

leaf1 = abjad.select(staff).leaves()[0]

abjad.attach(next(instruments1), leaf1)

abjad.attach(next(abbreviations1), leaf1)

abjad.attach(next(names1), leaf1)

abjad.attach(next(clefs1), leaf1)

for staff in abjad.iterate(score['Staff Group 2']).components(abjad.Staff):

leaf1 = abjad.select(staff).leaves()[0]

abjad.attach(next(instruments2), leaf1)

abjad.attach(next(abbreviations2), leaf1)

abjad.attach(next(names2), leaf1)

abjad.attach(next(clefs2), leaf1)

for staff in abjad.select(score['Staff Group 1']).components(abjad.Staff)[0]:

leaf1 = abjad.select(staff).leaves()[0]

last\_leaf = abjad.select(staff).leaves()[-1]

abjad.attach(metro, leaf1)

abjad.attach(bar\_line, last\_leaf)

for staff in abjad.select(score['Staff Group 2']).components(abjad.Staff)[0]:

leaf1 = abjad.select(staff).leaves()[0]

last\_leaf = abjad.select(staff).leaves()[-1]

abjad.attach(metro, leaf1)

abjad.attach(bar\_line, last\_leaf)

for staff in abjad.iterate(score['Global Context']).components(abjad.Staff):

leaf1\_start = abjad.select(staff).leaves()[7]

leaf1 = abjad.select(staff).leaves()[8]

abjad.attach(mark1, leaf1)

abjad.attach(section\_bar\_line, leaf1\_start)

for staff in abjad.iterate(score['Global Context']).components(abjad.Staff):

leaf2\_start = abjad.select(staff).leaves()[15]

leaf2 = abjad.select(staff).leaves()[16]

abjad.attach(mark2, leaf2)

abjad.attach(section\_bar\_line, leaf2\_start)

for staff in abjad.iterate(score['Global Context']).components(abjad.Staff):

leaf3\_start = abjad.select(staff).leaves()[23]

leaf3 = abjad.select(staff).leaves()[24]

abjad.attach(mark3, leaf3)

abjad.attach(section\_bar\_line, leaf3\_start)

for staff in abjad.iterate(score['Global Context']).components(abjad.Staff):

leaf4\_start = abjad.select(staff).leaves()[31]

leaf4 = abjad.select(staff).leaves()[32]

abjad.attach(mark4, leaf4)

abjad.attach(section\_bar\_line, leaf4\_start)

for staff in abjad.iterate(score['Global Context']).components(abjad.Staff):

leaf5\_start = abjad.select(staff).leaves()[38]

leaf5 = abjad.select(staff).leaves()[39]

abjad.attach(mark5, leaf5)

abjad.attach(section\_bar\_line, leaf5\_start)

score\_file = abjad.LilyPondFile.new(

score,

includes=['first\_stylesheet.ily', '/Users/evansdsg2/abjad/docs/source/\_stylesheets/abjad.ily'],

)

abjad.SegmentMaker.comment\_measure\_numbers(score)

###################

directory = '/Users/evansdsg2/Scores/cthar/cthar/Segments/Segment\_I'

pdf\_path = f'{directory}/Segment\_I.pdf'

path = pathlib.Path('Segment\_I.pdf')

if path.exists():

print(f'Removing {pdf\_path} ...')

path.unlink()

time\_1 = time.time()

print(f'Persisting {pdf\_path} ...')

result = abjad.persist(score\_file).as\_pdf(pdf\_path)

print(result[0])

print(result[1])

print(result[2])

success = result[3]

if success is False:

print('LilyPond failed!')

time\_2 = time.time()

total\_time = time\_2 - time\_1

print(f'Total time: {total\_time} seconds')

if path.exists():

print(f'Opening {pdf\_path} ...')

os.system(f'open {pdf\_path}')