Four Preludes

computer music for electric guitar and tape

Nigel Morgan

SCOM score-file

```
;;; PRELUDES FOR ELECTRIC GUITAR AND COMPUTER

;  1. Spacious Subway & Secret Stations
;; part 1

(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                            a   b   c    d    e   f   g

(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))


(def-tonality
  synth (activate-tonality (original b 2))
  guitar (activate-tonality (original-g e 3)(original-g a 3)
                            (original-g b& 4)(original-g c# 5)))

(def-symbol
  synth (find-change (vector-to-symbol a g
                                    (gen-noise-white 128 0.1 0.191)))
  guitar (find-change (vector-to-symbol a i
                                         (gen-noise-white 128 0.2 0.193))))

(def-length
  synth '(1/16)
  guitar '(1/16))

(def-zone
  synth (* 128 (get-tick '1/16))
  guitar (build-list (/ (car (zone-of synth)) 4) 4))

(def-channel
 synth 1
 guitar 4)

(def-program gm-sound-set
 synth electric-piano-1
 guitar 29)

(def-tempo 90)

(compile-instrument-p "ccl;output:" "prelude1i"
  synth
 guitar
)
```

```
;  1. Spacious Subway & Secret Stations
;; part 2


;; material

(setq seed 0.191)
(setq value 128)

(setq rhy1 (append '(-1/2 -1/16 1/16 1/8)
                (symbol-shuffle (gen-repeat 4 '(1/8 1/16 1/16)) 0.1)
                   '(1/8 1/16 1/32 1/32)))

(setq rhy2 (append (symbol-shuffle
                     (gen-repeat 7 '(1/16 1/16 1/8)) 0.2)
                   '(1/8 1/16 1/32 1/32)))


(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                           a   b   c    d    e   f   g

(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))


(def-tonality
  default (activate-tonality (original b 2))
  guitar (activate-tonality (original-g e 6)(original-g e 5)
                            (original-g e 4)(original-g e 3))
)

(def-symbol
  synth1 (find-change
          (vector-to-symbol a g (gen-noise-white value 1.0 seed)))
  synth2 (filter-delete '(a b c) (get-symbols-of 'synth1))
  synth3 (filter-delete '(d e f g) (get-symbols-of 'synth1))
  guitar (find-change
       (vector-to-symbol a i (gen-noise-white value 0.7 0.193)))
)

(def-length
  default '(1/16)
  guitar  (list rhy1 rhy2 rhy2 (reverse rhy1))
)
```

```
(def-zone
  default (* value (get-tick '1/16))
  guitar  (build-list (/ (car (zone-of synth)) 4) 4)
)

(def-velocity
  synth1 (vector-round 64 110 (gen-noise-white value 1.0 seed))
  synth2 (vector-round 12 96 (gen-noise-white value 1.0 seed))
  synth3 (vector-round 96 12 (gen-noise-white value 1.0 seed))
  guitar (vector-round 116 54 (gen-noise-white value 0.7 0.193))
)

(def-channel
 synth1 1
 synth2 2
 synth3 3
 guitar 4
)

(def-program gm-sound-set
 synth1 electric-piano-1
 synth2 electric-piano-2
 synth3 synth-bass-2
 guitar 32
)

(def-controller gm-controllers
 (synth1  panning '((64)))
 (synth2  panning '((0)))
 (synth3  panning '((127)))
)

(def-tempo 90)

(compile-instrument-p "ccl;output:" "prelude1ii"
  synth1
  synth2
  synth3
  guitar
)
```

```
;  1. Spacious Subway & Secret Stations
;; part 3



(setq rhy1 (append '(-1/2 -1/16 1/16 1/8)
                (symbol-shuffle (gen-repeat 4 '(1/8 1/16 1/16)) 0.1)
                   '(1/8 1/16 1/32 1/32)))


(setq rhy2 (append
                     (gen-repeat 7 '(1/16 1/16 1/16 1/16)) '
              (1/16 1/16 1/32 1/32 1/32 1/32)))

(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                            a   b   c    d    e   f   g


(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))

(setq tonal (activate-tonality (original b 2)))

(setq gtone (symbol-shuffle
       (activate-tonality (original-g c# 5)(original-g e 5)
                          (original-g b& 4)(original-g e 3)) 0.1))

(setq seed 0.195)
(setq value 32)

(def-symbol
  synth1 (find-change
            (vector-to-symbol a g (gen-noise-white value 1.0 seed)))
  synth2 (gen-variants seed 3 '(a c e g bdf = = = = cef = = = g e a))
  synth3 (gen-variants seed 3 '(= a = a b b = b = d c b = = b b))
  guitar (find-change (vector-to-symbol a i
                                (gen-noise-white 128 0.9 0.194)))
)

(def-length
  default '(1/16)
  guitar (append rhy1 rhy2 (reverse rhy1) rhy2)
)
```

```
(def-velocity
  default (vector-round 32 110 (gen-noise-white value 1.0 seed))
  guitar (vector-round 72 120 (gen-noise-white value 0.7 0.193))
)

(def-channel
 synth1 1
 synth2 2
 synth3 3
 guitar 4
)

(def-program gm-sound-set
 synth1 electric-piano-1
 synth2 electric-piano-2
 synth3 synth-bass-2
 guitar 30
)

(def-controller gm-controllers
 (synth1  panning '((64)))
 (synth2  panning '((0)))
 (synth3  panning '((127)))
)

(def-tempo 90)

(compile-song-p "ccl;output:" 1/2  "prelude1iii"

chang  gtone " . .  . . .  .  "
synth1 tonal "----------------"
synth2 tonal "-   ---- - - ---"
synth3 tonal "-  --  -- --- --"
guitar chang "------ --- -----"

)

#|


synth1 - el-piano1, rnd pan 'dull'
synth2 - el-piano2, 64 pan, 'bright' max reverb, chorus
synth3 - synth-bass3, 64 pan, nil reverb, some chorus

|#
```

```
2. Symetrical Weather of Mountains
; part 1

(set-tonality duo
             d& 5  ; a  low bongo
             c 5   ; b  high bongo)

(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))

(create-tonality original-g1 '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4
                            ;  a   b   c   d   e   f    g
                              f 4 f# 4 a# 4 b 4 c 5 d 5 d# 5
                            ;  h  i    j    k   l   m   n
                              f# 5 g 5 b 5 c 6 c# 6 d# 6 e 6))
                            ;  o   p    q   r   s   t    u

(setq g-sym '(= ad = be fh
             i h k j m l = o
             pm = nk = li
             ae = = be = cg il ko ps
             = egjng = psm))

(setq rhy '(1/4 1/4 1/4 1/4 1/4 ; 5/4
          1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 ; 4/4
          1/4 1/8 1/8 1/8 1/8 ; 3/4
          1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 ; 9/8
          1/4 1/4 1/4 1/8) ; 7/8)

(setq dyn '(120 110 110 110 120
            54 32 64 32 32 74 32 84
            94 32 42 52 62
            84 32 42 74 32 42 62 32 42
            72 82 92 127))

(setq sym '(ab a a a ab
            b a b a a ba b ba
            a a b b b
            a b b ab b b = ab ab
            b ab b =))

(setq a '(1 5 1)
      b '(6 13 1)
      c '(14 18 1)
      d '(19 27 1)
      e '(28 31 1))
```

```
(setq structure (vector-to-symbol a e
                                  (gen-ramp 4.5 0.5 32 90
                                            (gen-sin 1.5 0.3 32))))
(def-tonality
 bongos duo
 guitar (activate-tonality (original-g1 e 3))
)

(def-symbol
 bongos (gen-loop  (mapcar 'eval structure) sym)
 guitar (gen-loop  (mapcar 'eval structure) g-sym)
)

(def-length
 bongos (gen-loop (mapcar 'eval structure) rhy)
 guitar (gen-loop (mapcar 'eval structure) rhy)
)

(def-zone
 default (make-zone (get-lengths-of 'bongos))
)

(def-velocity
 bongos (gen-loop (mapcar 'eval structure) dyn)
 guitar (gen-loop (mapcar 'eval structure) dyn)
)

(def-channel
 bongos 10
 guitar 4
)

(def-program
guitar 26
)

(def-tempo 150)

(compile-instrument-p "ccl;output:" "prelude 2i"
 bongos
 guitar
)
```

```
2. Symetrical Weather of Mountains
; part 2



(set-tonality duo
             d& 5  ; a  low bongo
             c 5   ; b  high bongo
             e 5   ; c  low conga
             e& 5  ; d  open high conga
             d 5   ; e  muted high conga
)

(create-tonality original-g1 '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4
                            ;  a   b   c   d   e   f    g
                              f 4 f# 4 a# 4 b 4 c 5 d 5 d# 5
                            ;  h  i    j    k   l   m   n
                              f# 5 g 5 b 5 c 6 c# 6 d# 6 e 6))
                            ;  o   p    q   r   s   t    u

(setq g-sym1 '(= a b nk c d e f gj
               = uq = s r p o
               pm = nk = li
               = pq = uq = n m kj
               i h gnpr = s t))

(setq g-rhy '(1/4 1/8 1/8 1/4 1/8 1/8 1/16 1/16 1/8 ; 5/4
              1/4 1/4 1/4 1/16 1/16 1/16 1/16 ; 4/4
              1/4 1/8 1/8 1/8 1/8; 3/4
              1/8 1/4 1/8 1/4 1/8 1/16 1/16 1/8 ; 9/8
              1/8 1/8 1/4 1/4 1/16 1/16 ); 7/8)

(setq rhy '(1/4 1/4 1/4 1/4 1/4 ; 5/4
           1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 ; 4/4
           1/4 1/8 1/8 1/8 1/8 ; 3/4
           1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 ; 9/8
           1/4 1/4 1/4 1/8) ; 7/8)

(setq g-dyn '(0 110 90 64 75 80 85 90 110
              0 120 0 84 70 73 76
              94 32 42 52 62
              0 110 0 127 0 96 76 96
              80 72 110 0 115 119))
```

```
(setq dyn '(120 110 110 110 120
            54 32 64 32 32 74 32 84
            94 32 42 52 62
            84 32 42 74 32 42 62 32 42
            72 82 92 127))

(setq sym-b '(ab a a a ab
            b a b a a ba b ba
            a a b b b
            a b b ab b b = ab ab
            b ab b =))

(setq sym-c '(= = c e dc
              = d c = = d e d
              c = e d e
              = = e d d d c = e
              = e d c))


(setq a '(1 5 1)
      b '(6 13 1)
      c '(14 18 1)
      d '(19 27 1)
      e '(28 31 1))

(setq f '(1 9 1)
      g '(10 16 1)
      h '(17 21 1)
      i '(22 29 1)
      j '(30 35 1))

(setq structure (vector-to-symbol a e
                                  (gen-ramp 4.5 0.5 32 90
                                            (gen-sin 1.5 0.3 32))))

(setq structure-g (vector-to-symbol f j
                                  (gen-ramp 4.5 0.5 32 90
                                            (gen-sin 1.5 0.3 32))))
```

```
(def-tonality
 default duo
 guitar (activate-tonality (original-g1 e 3))
)

(def-symbol
 bongos (gen-loop  (mapcar 'eval structure) sym-b)
 congas (gen-loop  (mapcar 'eval structure) sym-c)
 guitar (gen-loop  (mapcar 'eval structure-g) g-sym1)
)

(def-length
 default (gen-loop (mapcar 'eval structure) rhy)
 guitar (gen-loop (mapcar 'eval structure-g) g-rhy)
)

(def-zone
 default (make-zone (get-lengths-of 'bongos))
)

(def-velocity
 default (gen-loop (mapcar 'eval structure) dyn)
 guitar (gen-loop (mapcar 'eval structure-g) g-dyn)
)

(def-channel
 bongos 10
 congas 10
 guitar 4
)

(def-tempo 150)

(compile-instrument-p "ccl;output:" "prelude 2ii"
 bongos
 congas
 guitar
)
```

```
2. Symetrical Weather of Mountains
; part 3


(set-tonality duo
              d& 5  ; a  low bongo
              c 5   ; b  high bongo
              e 5   ; c  low conga
              e& 5  ; d  open high conga
              d 5   ; e  muted high conga
)

(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                           a   b   c    d    e   f   g

(create-tonality original-g1 '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4
                           ;  a   b   c   d   e   f    g
                             f 4 f# 4 a# 4 b 4 c 5 d 5 d# 5
                           ;  h  i    j    k   l   m   n
                             f# 5 g 5 b 5 c 6 c# 6 d# 6 e 6))
                           ;  o   p    q   r   s   t    u

(setq g-sym2 '(= = h i j k l m n o =
               c d e = = g f i h k j =
               pm = nk = li
               cijkp = be ad fh i l
               = u tn sl rk pj io))


(setq g-rhy1 '(1/4 1/4 1/16 1/16 1/16 1/16 1/16 1/16 1/16 1/16 1/4
              1/16 1/16 1/8 1/8 1/8 1/16 1/16 1/16 1/16 1/16 1/16 1/8
               1/4 1/8 1/8 1/8 1/8
               1/4. 1/4 1/8 1/8 1/8 1/16 1/16
               1/8 1/8 1/8 1/8 1/8 1/8 1/8))



(setq g-dyn1 '(0 0 55 57 59 60 62 64 70 74 0
               96 84 96 0 0 45 47 64 59 54 52 0
                94 52 42 52 62
                110 0 96 80 96 72 70
                0 120 110 100 96 90 84))
```

```
(setq rhy '(1/4 1/4 1/4 1/4 1/4 ; 5/4
          1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 ; 4/4
          1/4 1/8 1/8 1/8 1/8 ; 3/4
          1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8 ; 9/8
          1/4 1/4 1/4 1/8) ; 7/8
)

(setq dyn '(120 110 110 110 120
           54 32 64 32 32 74 32 84
           94 32 42 52 62
           84 32 42 74 32 42 62 32 42
           72 82 92 127)
)

(setq sym-b '(ab = = a ab
           b = b = = ba = ba
           = = b b b
           a = b ab = b = ab ab
           b ab b =)
)

(setq sym-c '(= = c e dc
             = d c = = d e d
             c = e d e
             = = e d d d c = e
             = e d c)
)

(setq a '(1 5 1)
     b '(6 13 1)
     c '(14 18 1)
     d '(19 27 1)
     e '(28 31 1)
)

(setq f '(1 11 1)
     g '(12 23 1)
     h '(24 28 1)
     i '(29 35 1)
     j '(36 42 1)
)
```

```
(setq structure (vector-to-symbol a e
                                  (gen-ramp 4.5 0.5 32 90
                                            (gen-sin 1.5 0.3 32))))

(setq structure-g (vector-to-symbol f j
                                  (gen-ramp 4.5 0.5 32 90
                                            (gen-sin 1.5 0.3 32))))



(def-tonality
 default duo
 synth (activate-tonality (original b 2))
 guitar (activate-tonality (original-g1 e 3))
)

(def-symbol
 bongos (gen-loop  (mapcar 'eval structure) sym-b)
 congas (gen-loop  (mapcar 'eval structure) sym-c)
 synth  (fill-rest (get-symbols-of 'congas)
                   (symbol-scale '(a g) structure))
 guitar (gen-loop  (mapcar 'eval structure-g) g-sym2)

)

(def-length
 default (gen-loop (mapcar 'eval structure) rhy)
 synth (get-lengths-of 'default)
 guitar (gen-loop (mapcar 'eval structure-g) g-rhy1)
)

(def-zone
 default (make-zone (get-lengths-of 'bongos))
)

(def-velocity
 default (gen-loop (mapcar 'eval structure) dyn)
 synth   (vector-round 32 120
         (gen-triangle 0.5 1 (length (get-velocities-of 'default))))
 guitar (gen-loop (mapcar 'eval structure-g) g-dyn1)
)
```

```
(def-channel
 bongos 10
 congas 10
 synth 1
 guitar 4
)


(def-tempo 150)

(compile-instrument-p "ccl;output:" "prelude2iii"
 bongos
 congas
 synth
 guitar
)
```

```
3. Focused imagination of Sleep
; part 1


(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                           a   b   c    d    e   f   g

(setq value 256)

(setq dynamics
      (vector-to-list (vector-round 12 110
                    (gen-ramp 4.5 0.5 value 0
                              (gen-sin 1.5 0.3 value)))))

(setq g-symbols
      (vector-to-symbol a n
                    (gen-ramp 4.5 0.7 (/ value 2) 0
                              (gen-sin 1.5 0.9 (/ value 2)))))



(setq g-symbols-o (ornament-higher-chromatic
              (ornament-lower 4 g-symbols)))

(setq g-lengths (length (get-timing '1/8 (find-change g-symbols-o))))

(setq g-dynamics
      (vector-to-list (vector-round 64 120
                    (gen-ramp 4.5 0.7 g-lengths 0
                              (gen-sin 1.5 0.9 g-lengths)))))

(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))


(def-tonality
  default (activate-tonality (original b 2))
  guitar (activate-tonality (original-g e 4))
)
```

```
(def-symbol
  s1  '(g)
  s2  '(f)
  s3  '(e)
  s4  '(d)
  s5  '(c)
  s6  '(b)
  s7  '(a)
 guitar (delete '= (find-change g-symbols-o))
)

(def-length
  default  '(1/16)
  guitar  (get-timing '1/8 (find-change g-symbols-o))
)

(def-zone
  default (* (get-tick '1/16) value)
)

(def-velocity
  s1 (symbol-scroll 32 dynamics)
  s2 (symbol-scroll 64 dynamics)
  s3 (symbol-scroll 96 dynamics)
  s4 (symbol-scroll 128 dynamics)
  s5 (symbol-scroll 160 dynamics)
  s6 (symbol-scroll 192 dynamics)
  s7 dynamics
  guitar g-dynamics

)

(def-channel
  s1  1
  s2  2
  s3  3
  s4  4
  s5  5
  s6  6
  s7  7
  guitar 8
)
```

```
(def-program gm-sound-set
  default fx-4-athmosphere
  guitar 27
)

(def-controller gm-controllers
  (s1 panning '((0)))
  (s2  panning '((24)))
  (s3  panning '((48)))
  (s4  panning '((64)))
  (s5  panning '((72)))
  (s6  panning '((96)))
  (s7  panning '((120)))
  (guitar panning '((64)))
)


(def-tempo 100)

(compile-instrument-p "ccl;output:" "prelude 3i"
  s1
  s2
  s3
  s4
  s5
  s6
  s7
 guitar
)
```

```
3. Focused imagination of Sleep
; part 2

(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                           a   b   c    d    e   f   g

(setq value 256)

(setq dynamics
      (vector-to-list (vector-round 12 110
                    (gen-ramp 4.5 0.5 value 0
                              (gen-sin 1.5 0.3 value)))))


; (setq g-symbols-1
      (vector-to-symbol -g n
                    (gen-ramp 4.5 0.7 (/ value 2) 0
                              (gen-sin 2.5 0.9 (/ value 2)))))


(setq g-symbols-1 (vector-to-symbol -g q
                           (gen-sin 4.5 0.5 (/ value 2) 0

(setq g-symbols-o (ornament-lower 1 g-symbols-1))

(setq g-lengths (length (get-timing '1/8 (find-change g-symbols-1))))

(setq g-dynamics
      (vector-to-list (vector-round 74 120
                    (gen-ramp 4.5 0.7 g-lengths 0
                              (gen-sin 2.5 0.75 g-lengths)))))

(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))


(def-tonality
  default (activate-tonality (original b 2))
  guitar (activate-tonality (original-g e 4))
)
```

```
(def-symbol
  s1  (vector-to-symbol a g (gen-sin 4.5 0.5 value 30))
  s2  (vector-to-symbol a g (gen-sin 4.5 0.5 value 60))
  s3  (vector-to-symbol a g (gen-sin 4.5 0.5 value 90))
  s4  (vector-to-symbol a g (gen-sin 4.5 0.5 value 120))
  s5  (vector-to-symbol a g (gen-sin 4.5 0.5 value 150))
  s6  (vector-to-symbol a g (gen-sin 4.5 0.5 value 180))
  s7  (vector-to-symbol a g (gen-sin 4.5 0.5 value 0))
  guitar (delete '= (find-change g-symbols-o))
)

(def-length
  default '(1/16)
  guitar (get-timing '1/8 (find-change g-symbols-o))
)


(def-zone
  default (* (get-tick '1/16) value)
)
(def-instrument-velocity
  s1 (symbol-scroll 32 dynamics)
  s2 (symbol-scroll 64 dynamics)
  s3 (symbol-scroll 96 dynamics)
  s4 (symbol-scroll 128 dynamics)
  s5 (symbol-scroll 160 dynamics)
  s6 (symbol-scroll 192 dynamics)
  s7 dynamics
  guitar g-dynamics

)

(def-channel
  s1  1
  s2  2
  s3  3
  s4  4
  s5  5
  s6  6
  s7  7
 guitar 8
)
```

```
(def-program gm-sound-set
  default fx-4-athmosphere
  guitar 27
)

(def-controller gm-controllers
  (s1 panning '((0)))
  (s2  panning '((24)))
  (s3  panning '((48)))
  (s4  panning '((64)))
  (s5  panning '((72)))
  (s6  panning '((96)))
  (s7  panning '((120)))
  (guitar panning '((64)))
)


(def-tempo 100)

(compile-instrument-p "ccl;output:" "prelude3ii"
  s1
  s2
  s3
  s4
  s5
  s6
  s7
 guitar
)
```

```
3. Focused imagination of Sleep
; part 3



(create-tonality original '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
;                           a   b   c    d    e   f   g

(setq value 256)

(setq dynamics
      (vector-to-list (vector-round 12 110
                    (gen-ramp 4.5 0.5 value 0
                              (gen-sin 1.5 0.3 value)))))

; (setq g-symbols-1
      (vector-to-symbol -g q
                    (gen-ramp 4.5 0.7 (/ value 2) 0
                              (gen-sin 2.5 0.9 (/ value 2)))))


(setq g-symbols-1 (vector-to-symbol h q
              (gen-sin 4.5 0.5 (/ value 2) 0
                     (gen-sin 2.5 0.75 (/ value 2) 240))))

(setq g-symbols-o (ornament-higher-chromatic  g-symbols-1))

(setq g-lengths (length (get-timing '1/8 (find-change g-symbols-1))))

(setq g-dynamics
      (vector-to-list (vector-round  120 74
                    (gen-ramp 4.5 0.5 g-lengths 0
                              (gen-sin 1.5 0.3 g-lengths)))))

(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))

(def-tonality
  default (activate-tonality (original b 2))
  guitar (activate-tonality (original-g e 4))
)
```

```
(def-symbol
  s1  (filter-lowcut 'd
                     (vector-to-symbol a g
                                       (gen-sin 4.5 0.5 value 30)))
  s2  (filter-midcut 'c 'e
                     (vector-to-symbol a g
                                       (gen-sin 4.5 0.5 value 60)))
  s3  (filter-highcut 'd
                      (vector-to-symbol a g
                                        (gen-sin 4.5 0.5 value 90)))
  s4  (filter-midpass 'c 'e
                      (vector-to-symbol a g
                                        (gen-sin 4.5 0.5 value 120)))
  s5  (filter-pass '(b d f)
                   (vector-to-symbol a g
                                     (gen-sin 4.5 0.5 value 150)))
  s6  (filter-delete '(a g)
                     (vector-to-symbol a g
                                       (gen-sin 4.5 0.5 value 180)))
  s7  (vector-to-symbol a g (gen-sin 4.5 0.5 value 0))
  guitar (delete '= (find-change g-symbols-o))
)

(def-length
  default  '(1/16)
  guitar (get-timing '1/8 (find-change g-symbols-o))
)

(def-zone
  default (* (get-tick '1/16) value)
)

(def-velocity
  s1 (symbol-scroll -32 dynamics)
  s2 (symbol-scroll -64 dynamics)
  s3 (symbol-scroll -96 dynamics)
  s4 (symbol-scroll -128 dynamics)
  s5 (symbol-scroll -160 dynamics)
  s6 (symbol-scroll -192 dynamics)
  s7 dynamics
 guitar g-dynamics

)
```

```
(def-channel
  s1  1
  s2  2
  s3  3
  s4  4
  s5  5
  s6  6
  s7  7
 guitar 8
)

(def-program gm-sound-set
  default fx-4-athmosphere
  guitar 27
)

(def-controller gm-controllers
  (s1 panning '((0)))
  (s2  panning '((24)))
  (s3  panning '((48)))
  (s4  panning '((64)))
  (s5  panning '((72)))
  (s6  panning '((96)))
  (s7  panning '((120)))
  (guitar  panning '((0)))
)


(def-tempo 100)

(compile-instrument-p "ccl;output:" "prelude3iii"
  s1
  s2
  s3
  s4
  s5
  s6
  s7
  guitar
)
```

```
4. Journey in a Logical Forest.
; part 1


(create-tonality original-g '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))

(setq g-ton-lis
  (symbol-transform
    from '((e 3) (f 3) (a 3) (b& 3) (b 3) (c# 4) (d 4))
    to '((e 2) (f 4) (a 4) (b& 4) (c# 5) (d 5) (b 6))
    order '(5 2 4 3 1 6 0)
    changes '(1)
    repeat '(1)
))


(setq ton-lis
  (symbol-transform
    from '((b 2) (d 4) (c# 5) (b& 4) (a 5) (f 5) (e 6))
    to '((e 2) (f 3) (a 4) (b& 5) (c# 4) (d 5) (b 6))
    order '(5 2 4 3 1 6 0)
    changes '(1)
    repeat '(1)
))

(setq tonal (list
            (flatten (subseq ton-lis 0 7))
            (flatten (subseq ton-lis 7 14))
            (flatten (subseq ton-lis 14 21))
            (flatten (subseq ton-lis 21 28))
            (flatten (subseq ton-lis 28 35))
            (flatten (subseq ton-lis 35 42))
            (flatten (subseq ton-lis 42 49))
))

(setq g-tonal (list
            (flatten (subseq g-ton-lis 0 7))
            (flatten (subseq g-ton-lis 7 14))
            (flatten (subseq g-ton-lis 14 21))
            (flatten (subseq g-ton-lis 21 28))
            (flatten (subseq g-ton-lis 28 35))
            (flatten (subseq g-ton-lis 35 42))
            (flatten (subseq g-ton-lis 42 49))
))
```

```
(defun symbol-to-dynamic (symbol)
 (cadr (assoc symbol '((a 72) (b 64) (c 54) (d 44)
             (e 60) (f 80) (g 96) (= 0)))))

(def-tonality
   default tonal
   guitar g-tonal
)

(def-symbol
   synth (find-change
          (vector-to-symbol a g (gen-noise-white (* 14 7))))
   echo1 '(abcdefg =)
   echo2 '(= abcdefg)
   guitar (flatten (cons '(= = = = = = =)
                  (gen-evolve 2 '(change-order nil x)
                       '(= a b c d e f g f e d c b =) :list 0.123)))
)

(def-length
   synth '(1/16)
   default '(7/8)
   guitar '(1/8 1/8 1/8 1/8 1/8 1/8 1/8))
)

(def-duration
 guitar '(7/8 6/8 5/8 4/8 3/8 2/8 1/8)
)

(def-zone
   default (build-list '7/8 (length tonal))
)

(def-velocity
  synth (mapcar 'symbol-to-dynamic (get-symbols-of 'synth))
  default '(32)
  guitar (append '(0 0 0 0 0 0 0)
       (gen-repeat 3 '(0 32 45 48 56 67 77 80 77 67 56 48 45 0)))
)
```

```
(def-channel
 synth 1
 echo1 4
 echo2 5
 guitar 6
)

(def-tempo 50)

(compile-instrument-p "ccl;output:" "prelude4i"
 synth
 echo1
 echo2
 guitar
)

4. Journey in a Logical Forest.
; part 2

(defun chord-list-div (pattern)
  (setq pattern (car pattern))
  (prog (out)
    loop
    (cond ((null pattern) (return out)))
    (setq out (append out
                      (list (list (car pattern) (cadr pattern)))))
    (setq pattern (cddr pattern))
    (go loop)))

(defun chords-list-div (num pattern)
  (prog (out)
    (dotimes (i (/ (length pattern) num))
      (setq out
            (append out
                    (list
                     (flatten
                      (subseq pattern (* i num) (* (+ i 1) num)))))))
    (return out)))

(create-tonality original1 '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
(create-tonality original2 '(e 2 f 4 a 5 b& 4 c# 5 d 5 b 6))

(create-tonality original-g1 '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))
(create-tonality original-g2 '(e 2 f 4 a 4 b& 4 c# 5 d 5 b 6))
```

```
(setq g-ton-lis
 (symbol-transform
  from (chord-list-div (activate-tonality (original-g1 e 3)))
  to (chord-list-div (activate-tonality (original-g2 e 2)))
  order '(5 2 4 3 1 6 0)
  changes '(1)
  repeats '(1)
))


(setq ton-lis
 (symbol-transform
  from (chord-list-div (activate-tonality (original1 b 2)))
  to (chord-list-div (activate-tonality (original2 e 2)))
  order '(5 2 4 3 1 6 0)
  changes '(1)
  repeats '(1)
))

(def-tonality
   default (gen-palindrome (chords-list-div 7 ton-lis))
   guitar (gen-palindrome (chords-list-div 7 g-ton-lis))
)

(def-symbol
   synth (find-change
            (vector-to-symbol a g
              (gen-noise-white
                 (* (length (get-chords-of 'synth)) 14))))
 echo1 '(abcdefg =)
 echo2 '(= abcdefg)
 guitar (flatten (cons '(= a = bd = ef =)
                  (gen-evolve 5 '(change-order nil x)
                    '(= a b c d e f g f e d c b =) :list 0.123)))
)

(def-length
   synth '(1/16)
   default '(7/8)
   guitar '(1/8 1/8 1/8 1/8 1/8 1/8 1/8))
)
```

```
(def-duration
 guitar '(7/8 6/8 5/8 4/8 3/8 2/8 1/8)
)

(def-zone
   default (build-list '7/8 (length (get-chords-of 'synth)))
)

(def-velocity
 synth (gen-accent 32 14
           (gen-cresc-dim 32 96 (length (get-symbols-of 'synth))))
 default '(32)
 guitar (append '(0 40 0 50 0 65 0)
       (gen-repeat 12 '(32 32 45 48 56 67 77 80 77 67 56 48 45 32)))
)

(def-channel
 synth 1
 echo1 4
 echo2 5
 guitar 6
)

(def-tempo 50)

(compile-instrument-p "ccl;output:" "prelude4ii"
  synth
  echo1
  echo2
  guitar
)

4. Journey in a Logical Forest.
; part 3


(create-tonality original1 '(b 2 d 4 c# 5 b& 4 a 5 f 5 e 6))
(create-tonality original2 '(e 2 f 4 a 5 b& 4 c# 5 d 5 b 6))
(create-tonality original3 '(f# 2 c# 4 e 5 b 4 g# 5 a# 5 d# 6))

(create-tonality original-g1 '(e 3 f 3 a 3 b& 3 b 3 c# 4 d 4))
(create-tonality original-g2 '(e 2 f 4 a 4 b& 4 c# 5 d 5 b 6))
(create-tonality original-g3 '(e 2 a# 3 b 3 c# 4 f# 4 g# 5 d# 6))
```

```
(setq g-ton-lis
 (symbol-transform
  from (chord-list-div (activate-tonality (original-g1 e 3)))
  to (chord-list-div (activate-tonality (original-g3 e 2)))
  order '(5 2 4 3 1 6 0)
  changes '(1)
  repeats '(1)
))

(setq ton-lis
 (symbol-transform
  from (chord-list-div (activate-tonality (original1 b 2)))
  to (chord-list-div (activate-tonality (original3 f# 2)))
  order '(5 2 4 3 1 6 0)
  changes '(1)
  repeat '(1)
))

(def-tonality
   default (gen-palindrome (chords-list-div 7 ton-lis))
   guitar (gen-palindrome (chords-list-div 7 g-ton-lis))
)

(def-symbol
   synth (flatten
            (gen-palindrome
               (symbol-divide 14 nil nil
                 (find-change
                   (vector-to-symbol a g
                     (gen-noise-white (* 14 7)))))))
  echo1 '(abcdefg =)
  echo2 '(= abcdefg)
  guitar (flatten (cons '(= a = bd = ec =)
                  (gen-evolve 5 '(change-order nil x)
             '(= a b c d e f g f e d c b =) :list 0.123)))
)

(def-length
   synth '(1/16)
   default '(7/8)
   guitar '(1/8 1/8 1/8 1/8 1/8 1/8 1/8))
)
```

```
(def-duration
 guitar '(7/8 6/8 5/8 4/8 3/8 2/8 1/8)
)

(def-zone
   default (build-list '7/8 (length (get-chords-of 'synth)))
)

(def-velocity
 synth (gen-accent 32 14
           (gen-cresc-dim 32 96 (length (get-symbols-of 'synth))))
 default '(32)
 guitar (append '(0 40 0 50 0 65 0)
        (gen-repeat 12 '(32 32 45 48 56 67 77 80 77 67 56 48 45 32)))
)

(def-channel
 synth 1
 echo1 4
 echo2 5
 guitar 6
)

(def-tempo 50)

(compile-instrument-p "ccl;output:" "prelude4iii"
 synth
 echo1
 echo2
 guitar
)
```