sound

(why is it so imperative?)

sound

(why is it so imperative?)

sound

(why is it so imperative?)

sound

(why is it so imperative?)

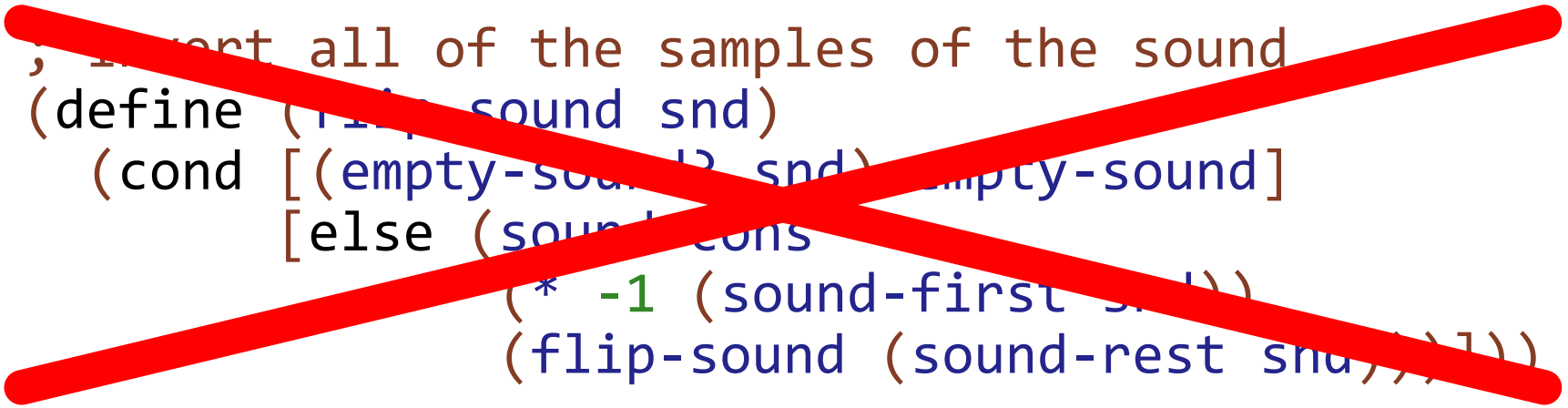
```
(require rsound
         rsound/piano-tones)

(define SAMPLE-RATE 44100)

(define s
 (assemble
  (for/list ([i 40])
    (list (piano-tone (+ 25 (random 30)))
          (* i (/ SAMPLE-RATE 10))))))
```

```
; invert all of the samples of the sound
(define (flip-sound snd)
  (cond [(empty-sound? snd) empty-sound]
        [else (sound-cons
               (* -1 (sound-first snd))
               (flip-sound (sound-rest snd)))]))
```

```
(cons
 0.0010376293221839045
 (cons
  0.0012207403790398877
  (cons
   0.0013428144169438765
   (cons
    0.0014648884548478652
    (cons
     0.0015564439832758568
     empty)))))
```


; Invert all of the samples of the sound
(define (flip-sound snd)
(cond [(empty-sound? snd) empty-sound]
[else (sound-cons
(* -1 (sound-first snd))
(flip-sound (sound-rest snd)))]))

16.6 ms


16.6 ms

22.7μs

```
; invert all of the samples of the sound
(define new-snd (silence (rs-frames snd)))

(for ([i (rs-frames snd)])
  (set-rs-ith/left! new-snd i
                    (* -1 (rs-ith/left snd i)))
  (set-rs-ith/right! new-snd i
                     (* -1 (rs-ith/right snd i))))
```

```
(define SR 44100)

(define (sine-tone f)
  (define freq 201)
  (* 0.2 (sin (* (/ (* 2 pi) SR) freq f))))

(play (fun->sound sine-tone (* 3 SR)))
```

```
(define SR 44100)

(define (sine-tone f)
  (define freq (* 201 (+ 1 (/ f (* 3 SR)))))
  (* 0.2 (sin (* (/ (* 2 pi) SR) freq f))))

(play (fun->sound sine-tone (* 3 SR)))
```

$$sin(t) = Re\{e^{i\omega t}\}$$

$$\frac{d}{dt}e^{i\omega t} = i\omega e^{i\omega t}$$

$$\left| \frac{d}{dt} e^{i\omega t} \right| = \left| i\omega e^{i\omega t} \right| = \omega$$

$$\left| \frac{d}{dt} e^{i(\omega + kt)t} \right|$$

$$|\frac{d}{dt}e^{i(\omega+kt)t}|$$

$$= |[\frac{d}{dt}(i(\omega+kt)t)]]e^{i(\omega+kt)t}|$$

$$= |\frac{d}{dt}(i(\omega+kt)t)]|$$

$$= |\frac{d}{dt}((\omega+kt)t)]|$$

$$= |\frac{d}{dt}(\omega t+ikt^2)|$$

$$= |(\omega+2kt)|$$

grrr!

```
(define ang 0.0)
(define pitch 400)
(define pitch-incr 0.01)
(loop
 (define cur-sample (sin angle))
 (set! pitch (+ pitch pitch-incr))
 (set! angle (+ angle (* 2 pi (/ pitch 44100))))
 cur-sample)
```

yay!

```
(define pitch-incr 0.01)
(network ()
  [cur-sample = (sin angle)]
  [pitch = (+ (prev pitch 400) pitch-incr)]
  [angle = (+ (prev angle 0.0) (* 2 pi (/ pitch 44100)))])
```

```
(define pitch-incr 0.01)
(network ()
  [cur-sample <= sine-wave pitch]
  [pitch = (+ (prev pitch 400) pitch-incr)])
```

```
(lambda ()
  (let* ([cur-sample1 #f]
         [pitch2 #f]
         [temp3 (network-init sine-wave)]
         [temp4 #f]
         [later-times-fun
          (lambda ()
            (let* ([cur-sample (temp3 pitch)]
                   [pitch (+ pitch2 pitch-incr)])
              (set! cur-sample1 cur-sample)
              (set! pitch2 pitch)
              pitch))]
         [first-time-fun
          (lambda ()
            (set! first-time-fun later-times-fun)
            (let* ([cur-sample (temp3 pitch)]
                   [pitch (+ 400 pitch-incr)])
              (set! cur-sample1 cur-sample)
              (set! pitch2 pitch)
              pitch))])
    (lambda () (first-time-fun))))
```

Parts II & III

- big-bang and `[on-sound ...]`
- creative exploration and test cases

```
raco pkg install rsound
```