```
;;;;;;;;;;;;;;;;;;; SI proj ;;;;;;;;;;;;;;;;;;
;;;                 Yohei Yasukawa           ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; missile
; Definitions for testing
(define mis-0-0 (make-posn 0 0))
(define mis-10-10 (make-posn 10 10))
(define mis-20-20 (make-posn 20 20))
(define mis-30-30 (make-posn 30 30))
(define mis-40-40 (make-posn 40 40))
(define mis-50-50 (make-posn 50 50))
(define mis-60-60 (make-posn 60 60))
(define mis-70-70 (make-posn 70 70))
(define mis-80-80 (make-posn 80 80))
(define mis-90-90 (make-posn 90 90))
(define mis-100-100 (make-posn 100 100))

; Definitions for initializing
(define DEFAULT_X 0)
(define DEFAULT_Y 0)
(define MISSILE_IMG  )
(define MISSILE_MOV 1)
(define BG_WIDTH 500)
(define BG_HEIGHT 500)
(define T_EDGE 0)
(define EMPTY_MISSILE  )
(define EMPTY_SCENE (empty-scene BG_WIDTH BG_HEIGHT))
(define barrage-init (cons mis-0-0
                           (cons mis-20-20
                                 (cons mis-40-40
                                       (cons mis-60-60
                                             (cons
mis-80-80
                                                   (cons
mis-100-100 empty)))))))
(define missile-init (make-posn 100 100))


;; missile-tick: MissileStatue -> MissileStatue
; calculates the state following the given state if only
time passes
(define (missile-tick current)
  (move-up current))
```

```
;; barrage-tick: BarrageStatue -> BarrageStatus
; calculates the state following the given state if only
time passes
(define (barrage-tick barrage)
  (cond
    [(empty? barrage) empty]
    [(cons? barrage)
     (if (touch-top-wall? (first barrage))
         (barrage-tick (rest barrage)) ; remove a needless
missile.
         (cons-barrage-tick barrage))]
    ))
(define (cons-barrage-tick barrage)
  (cons
   (missile-tick (first barrage))
   (barrage-tick (rest barrage))
   ))
(check-expect (barrage-tick (cons mis-10-10 empty))
              (cons (make-posn 10 9) empty))
(check-expect (barrage-tick (cons mis-0-0 empty)) empty)

;; touch-top-wall? : Posn -> boolean
; determine if a given object is touching a wall on the
top.
(define (touch-top-wall? posn)
  (cond
    [(<= (posn-y posn) T_EDGE) true]
    [else false]
    ))
(check-expect (touch-top-wall? (make-posn 0 T_EDGE)) true)
(check-expect (touch-top-wall? (make-posn 10 10)) false)

;; move-up : Status -> Status
; move a given object to the top in 1 px
(define (move-up current)
  (if (touch-top-wall? current)
      current
      (make-posn (posn-x current)
                 (- (posn-y current) MISSILE_MOV))
      ))
(check-expect (move-up (make-posn 0 10)) (make-posn 0 9))
(check-expect (move-up (make-posn 0 T_EDGE)) (make-posn 0
T_EDGE))
```

```
; barrage-render : List of Missile -> image
; constructs an image representing the given state
(define (barrage-render barrage)
  (place-image (cons-barrage-render barrage
                                    (make-posn DEFAULT_X
DEFAULT_Y))
               -5 -5 ; (-5,-5) eliminates a missile image
on the origin.
               EMPTY_SCENE))

(define (cons-barrage-render barrage prev-missile)
  (cond
    [(empty? barrage) MISSILE_IMG]
    [(cons? barrage)
     (overlay/xy MISSILE_IMG
                 (- (posn-x (first barrage)) (posn-x
prev-missile))
                 (- (posn-y (first barrage)) (posn-y
prev-missile))
                 (cons-barrage-render (rest barrage)
(first barrage))
                 )]
    ))


; main function for barrage
(define (main-barrage current)
  (big-bang current
            (on-tick barrage-tick)
            (to-draw barrage-render)
            ))

(main-barrage barrage-init)
```