```
(define WIDTH 800)
(define HEIGHT 600)
(define BACKGROUND (overlay/align "middle"
"bottom" (rectangle WIDTH 50 "solid" "green")
                                  (rectangle
WIDTH HEIGHT "solid" "lightblue")))
```

```
(define PLANE (flip-horizontal          ))

;draw-plane: posn --> image
;draw-plane consumes a posn structure and
produces an image
(define (draw-plane point)
  (place-image PLANE (posn-x point) (posn-y
point) BACKGROUND))

;fly-plane: posn --> posn
;fly-plane consumes a posn structure and
produces a posn
(define (fly-plane point)
  (cond
    [(> (posn-x point) WIDTH)
     (make-posn 0 (+ 5 (posn-y point)))]
    [else (make-posn (+ (posn-x point) 3)
(posn-y point))]))

(check-expect (fly-plane (make-posn 50 50))
```

```
(make-posn 53 50))
(check-expect (fly-plane (make-posn (+ WIDTH 3)
50)) (make-posn 0 55))

;move-plane: posn string --> posn
;move-plane consumes a posn  and a string (key)
and produces a posn
(define (move-plane point key)
  (cond
    [(string=? key "up") (make-posn (posn-x
point) (- (posn-y point) 3))]
    [(string=? key "down") (make-posn (posn-x
point) (+ (posn-y point) 3))]
    [(string-ci=? key "q") (stop-with point)]
    [else point]))

(check-expect (move-plane (make-posn 50 50)
"up") (make-posn 50 47))
(check-expect (move-plane (make-posn 50 50)
"down") (make-posn 50 53))
(check-expect (move-plane (make-posn 50 50)
"h") (make-posn 50 50))

;plane-lands: point --> boolean
;plane-lands consumes a posn and produces a
boolean
(define (plane-lands point)
  (>= (+ (posn-y point) (/ (image-height PLANE)
2)) (- HEIGHT 50)))

(check-expect (plane-lands (make-posn 50 800))
```

```
true)


(big-bang (make-posn (/ (image-width PLANE) 2)
(/ (image-height PLANE) 2))
          (check-with posn?)
          (to-draw draw-plane)
          (on-tick fly-plane)
          (on-key move-plane)
          (stop-when plane-lands)
          )
```