```
;;;;;;;;;;;;;;;;;; VZ proj ;;;;;;;;;;;;;;;;;;
;;;                Yohei Yasukawa             ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;; Global Definitions
(define              )
(define               )
(define                                              )
(define                                  )


;;;;;;;;;;;;
;; Turtle ;;
;;;;;;;;;;;;
;; Definitions
(define                )
```

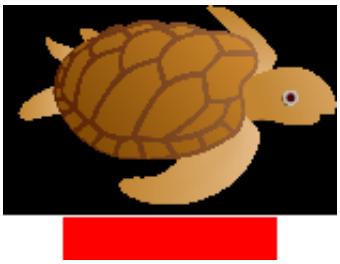
```
(define                          )
```

```
(define                                 )

(define                                          )
(define                                                  )


;; Definitions for examples
(define                                        )
(define                                        )
(define                                    )
(define                                  )


;; create-t-fmeter : ftime -> image
; create an image of feed meter by a given time.


;Examples
```

```
; turtle-tick : TurtleStatus -> TurtleStatus
; calculates the state following the given state if only
time passes
```

```
; turtle-key : TurtleStatus KeyEvent -> TurtleStatus
; calculates the state following the given state if given
key is pressed
```

```
; turtle-render : TurtleStatus -> image
; constructs an turtle image representing the given state
```



```
;;;;;;;;;;;;;;;;;;;
;; Lightning Bug ;;
;;;;;;;;;;;;;;;;;;;

; Definitions for lbug
```

```
(define                          )
```

```
(define                          )
```

```
(define                          )
```

```
(define                           )

(define                                                  )
(define                                                    )
;(define INIT_LBUG (make-lbug 50 50 "right-up" true))   ;
For further testing
;(define INIT_LBUG-1 (make-lbug 51 49 "right-up" true)) ;
For further testing

; Definitions for lbug examples
(define                 )
(define                )
```

```
(define                                        )
(define                )
(define                                    )
(define                                                  )
(define                                                )
(define
                    )
(define
                     )
(define                                                  )
(define                                                  )
(define
                     )
(define
                    )
(define                                                )
(define                                                 )
(define
                  )
(define
                   )
(define                                                    )
(define lbug-ru-11-9 (make-lbug 11 9 "right-up" true))
(define
                   )
(define
                  )

; lbug-tick : LBugStatus -> LBugStatus
; calculates the state following the given state if only
time passes




                                        (flip-vertically
current)
```

```
                                        (flip-vertically
current)


     (string=? (lbug-dir current) "left-up")
     (cond
       [(touch-left-wall? current) (flip-horizontally
current)]
       [(touch-top-wall? current) (flip-vertically
current)]
       [else (move-left-up current)]
       )
     (string=? (lbug-dir current) "right-up")
     (cond
       [(touch-right-wall? current) (flip-horizontally
current)]
       [(touch-top-wall? current) (flip-vertically
current)]
       [else (move-right-up current)]
       )
```

```
;; touch-left-wall? : LBugStatus -> boolean
; determine if a given lightning bug is touching a wall on
the left
```

```
;; touch-right-wall? : LBugStatus -> boolean
; determine if a given lightning bug is touching a wall on
the right
```

```
;; touch-top-wall? : LBugStatus -> boolean
; determine if a given lightning bug is touching a wall at
the top
```

```
;; touch-bottom-wall? : LBugStatus -> boolean
; determine if a given lightning bug is touching a wall at
the bottom
```

```
;; move-left-down : LBugStatus -> LBugStatus
; move a given lightning bug to the left down in 1 px
```

```
;; move-right-down : LBugStatus -> LBugStatus
; move a given lightning bug to the right down in 1 px
```

```
;; move-left-up : LBugStatus -> LBugStatus
; move a given lightning bug to the left up in 1 px
```

```
;; move-right-up : LBugStatus -> LBugStatus
; move a given lightning bug to the right in 1 px
```

```
;; flip-horizontally : LBugStatus -> LBugStatus
; make a given lightning bug face toward an opposite
direction.
```

```
     (string=? (lbug-dir current) "left-up")
     (light-random (make-lbug (lbug-posx current)
(lbug-posy current)
                              "right-up" (lbug-lighton?
current)))
     (string=? (lbug-dir current) "right-up")
     (light-random (make-lbug (lbug-posx current)
(lbug-posy current)
                               "left-up" (lbug-lighton?
current)))
```

```
;; flip-vertically : LBugStatus -> LBugStatus
; make a given lightning bug face toward an opposite
direction.
```

```
     (light-random (make-lbug (lbug-posx current)
(lbug-posy current)
                              "left-down" (lbug-lighton?
current)))
```

```
     (light-random (make-lbug (lbug-posx current)
(lbug-posy current)
                              "left-up" (lbug-lighton?
current)))
```

```
;; light-random : LBugStatus -> LBugStatus
; determine if a given lightning bug turns on or off at
random
```

```
     (make-lbug (lbug-posx current) (lbug-posy current)
(lbug-dir current) false)

; omitting check-expects due to random results
```

```
; lbug-render : LBugStatus -> image
; constructs an lightning bug image representing the given
state
```

```
          (string=? (lbug-dir current) "right-up")
```

```
     (place-image LB_LEFT_OFF_IMG
                  (+ (lbug-posx current) 50)
```

```
                    (+ (lbug-posy current) 50)
                    BACKGROUND)



      (place-image LB_RIGHT_OFF_IMG
                    (+ (lbug-posx current) 50)
                    (+ (lbug-posy current) 50)
                    BACKGROUND)


;;;;;;;;;
;; Dog ;;
;;;;;;;;;

; Definitions
```

```
(define                ) (define
```

```
)
```

```
(define                ) (define
```


)

(define ) (define


)

(define ) (define


)

(define ) (define


)

```
(define                  )
(define                    )
(define                    )


(define                 )
(define                 )
(define                   )
(define                   )
(define                      )
(define                   )

(define


                                                    )
(define


                                                                )

; Definitions for examples
(define
  )
(define                                                        )
(define                                                            )
(define                                                  )
(define                                                     )
(define                                                   )
(define                                                       )
(define                                                      )
(define                                                     )
(define                                                    )
(define                                                  )
(define                                                    )
(define                                                    )
(define                                                    )
(define                                                    )
(define                                                  )
```

```
(define                                                    )
(define                                                    )
(define                                                      )
(define                                                     )
(define                                                  )
(define                                                   )
(define                                                  )
(define                                                     )

; dog-tick : DogStatus -> DogStatus
; calculates the state following the given state if only
time passes

     current

; Examples

;; taildown : DogStatus -> DogStatus
; move tail position to down
```

```
;; tailup : DogStatus -> DogStatus
; move tail position to up
```

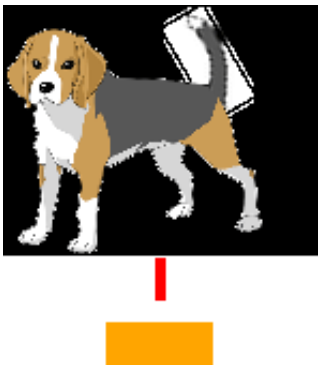
```
;; decr-fullness : DogStatus -> dog-fullness
; decrement a fullness by a given status
```

```
;; decr-happiness : DogStatus -> dog-happiness
; decrement a happiness by a given status
```

```
; dog-key : DogStatus KeyEvent -> DogStatus
; calculates the state following the given state if given
key is pressed

          current    ; This code won't be executed.
; Examples

;; feed-dog : DogStatus -> DogStatus
; calculate how much fullness a dog gets in one feed by a
given status
```

```
;; pet-dog : DogStatus -> DogStatus
; calculate how much happiness a dog gets in one pet by a
given status
```

```
; dog-render : DogStatus -> image
; constructs an image representing the given state
```



```
;; create-dog-image : current -> image
; create an dog image with tail down or tail up by a given
status.
```

```
;; create-meters : DogStatus -> image
; create/disappear a feed meter and happiness meter by a
given status,
; and put them into one image.
```

```
;; create-d-fmeter : ftime -> image
; create an image of feed meter by a given time.

;Examples
```

```
;; create-hmeter : htime -> image
; create an image of happiness meter by a given time.

;Examples
```

```
;;;;;;;;;;
;; Main ;;
;;;;;;;;;;
```

```
;; Definitions for examples
(define
                                                    )
(define
                                                          )


;; main-tick : Status -> Status
; calculates the state following the given state if only
time passes









;; main-key : Status -> KeyEvent
; calculates the state following the given state if given
key is pressed


          current


;; main-render : Status -> Status
; constructs an whole image representing the given state
```