(require 2htdp/image)

(require 2htdp/universe)

**Struct and template**

(define-struct passenger [name seat])

;; A Passenger is a (make-passenger String String)

;; - where name is the passenger's full name

;; - and the seat is the passenger's seat number

(define P1 (make-passenger "Scott MacKenzie" "3A"))

(define P2 (make-passenger "TJ Porter" "13C"))

;; passenger-temp : Passenger -> ???

#;

(define (passenger-temp p)

(... (passenger-name p) ... (passenger-seat p) ...))

**Union struct and template**

(define-struct monkey [name c others])

; A MonkeyChain is one of:

; - "barrel"

; - (make-monkey String String MonkeyChain)

;

; Interpretation: A collection of monkeys

; - "barrel" is an empty barrel

; - make-monkey

; - name is the name of this monkey

; - c is the color of this monkey

; - others is the other monkeys (or barrel) it is attached to

(define MONKEYCHAIN-0 "barrel")

(define MONKEYCHAIN-1 (make-monkey "Carol" "purple" MONKEYCHAIN-0))

(define MONKEYCHAIN-2 (make-monkey "Bob" "purple" MONKEYCHAIN-1))

#;

(define (monkey-temp m)

(cond

[(string? m) ...]

[(monkey? m)

(... (monkey-name m) ... (monkey-c m) ...

(monkey-temp (monkey-others m)) ...)]))

**Data definition and template**

;; An MaybeID is one of:

;; - #false

;; - PositiveInteger

;; representing either a student's NUID, or #false if it is unknown

(define MAYBE-ID1 #false)

(define MAYBE-ID2 123456789)

;; maybe-id-template : MaybeID -> ???

#;

(define (maybe-id-template maybe-id)

(cond [(false? maybe-id) ...]

[(number? maybe-id) ...]))

**常用公式**

(define BACKGROUND (empty-scene 500 500))

(define BACKGROUND (empty-image))

(boolean? Input)

(boolean=? Input #false / #true)

(string? input)

(string=? input “god”)

(number? input)

(= input 5)

(max/min 5 10 15)

(not A) ; if the output of A is true, then not A is false

(or A B C)

(and A B C)

(\* num 8)

(+/- num 8)

(sqr 3). 9

(expt 2 3) 8

(modulo 9 5) 4

(sqrt 9). 3

(string-append “hollo” “world”) “holloworld”

(string-append “hollo” “ ” “world”) “hollo world”

(string-length “hello world) 11

(number->string 43) “43”

(string->number “43”) 43

(string->number “hollo world) #false

(circle 10 “solid” “red)

(rectangle 30 20 “outline” “blue)

(above A B C D)

(overlay (square 50 “solid” “blue) (circle 10 “solid” “red))

(if A B C) ; if A is correct then B, If A is wrong then C

(place-image (circle 5 “solid””green”) 50 80 (empty-scene 100 100))

(string-upcase “cAt”) “CAT”

(substring “hello world” 1 7). “ello w”

(substring “hello world” 4). “o world”

(string-ith “hello world” 2). “l”

(text “hello” 24 “orange”)

1. 运行得出output题

(cond

;[#false (answer-to EX1)]

[#true "Greetings"]

[else (string-append "Hello, "

(answer-to EX1))])

; => "Greetings"

; Looking at (answer-question EX1) separately:

(string=? "CS2500"(string-append "2500" "CS"))

(string=? "CS2500" "2500CS") ;=> #false

2. 找bug要用check-expect 然后解释为什么运行不了

(max-earner (make-record "annie" 100)

(make-record "suzanne" 150))

;; this check-expect would fail with this function because of error in comparison

;; the actual output is "annie"

;; it should have been "suzanne"

(max-earner (make-record "alice" 350)

(make-record "brian" (make-bonus 350 100)))

;; this check-expect would fail with this function due to the error of not handling bonus

;; the actual output is "error"

;; it should have been “brian"