; CSC104 2017 Winter Test 2.

; LAST/FAMILY NAME:

; FIRST/GIVEN NAME:

; UTORID:

; TOTAL MARKS : 12 + 8 + 3 + 15 = 38
; Question 1. [12 Marks]

; Reminders.

(check-expect (first (list 2 0 1 7)) 2)
(check-expect (reverse (list 2 0 1 7)) (list 7 1 0 2))

; [4 Marks]

; Assume the following list has been defined:

(define L (list (list 1 2) 3 4 5 (list 6) 7 8))

; Show the final result value of each of the following expressions:

(length L)
(first L)
(reverse L)
(filter number? L)

; [8 Marks]

; Assume the following list has been defined:

(define LOL (list (list 1 2 3) (list 4) (list 5 6)))

; Show the intermediate steps and the final result value of each of the following expressions:

(map length LOL)

(map first LOL)
(map reverse LOL)

(apply append LOL)

; Question 2. [8 Marks]

; Assume function ‘r’ below has been defined:

(define (r L)
  (cond [(= (length L) 1) L]
        [else (list* (first L) (r (reverse (rest L)))))])

; Reminders.

(check-expect (rest (list 2 0 1 7)) (list 0 1 7))

(check-expect (list* 2 (list 0 1 7)) (list 2 0 1 7))

; [2 Marks] Show the the final result value of the following expression:
(r (list 1))

; [6 Marks] For each of the following expressions, show [at least] one ‘list*’ intermediate step
; and the final result value:
(r (list 1 2))

(r (list 1 2 3))

(r (list 1 2 3 4))
Question 3. [3 Marks]
 Convert the binary representation 1001101 to its decimal representation,
 briefly showing your steps.

Question 4. [15 Marks]

(release picturing-programs)

Assume the following has been defined:

(define E (input))

[4 Marks] Based on the 'check-expect's below, define the function 'sandwich'.

Documentation/Testing.

(check-expect (sandwich (image 1/2 (rotate-ccw (image 1/2 (rotate-cw)))))

(check-expect (sandwich (image 1/2 (rotate-ccw (image 1/2 (rotate-cw)))))

Design.

(check-expect (sandwich (image 1/2 (rotate-ccw (image 1/2 (rotate-cw))))))

(beside (scale 1/2 (rotate-ccw (image 1/2 (rotate-cw)))) (scale 1/2 (rotate-cw (image 1/2 (rotate-cw)))))

Define 'sandwich' here, including its type contract:

[6 Marks] Complete the three full design 'check-expects' below without drawing any images.
Use 'sandwich' to help.
Inside the 'check-expect' for '(branch 1)' use the expression '(branch 0)'.
Inside the 'check-expect' for '(branch 2)' use the expression '(branch 1)'.

Documentation/Testing.

(check-expect (branch 0))
(check-expect (branch 1) )

(check-expect (branch 2) )

; Full Design.

(check-expect (branch 0) )

)

(check-expect (branch 1) )

)

(check-expect (branch 2) )

)

; [5 Marks] Define the function ‘branch’.