

PLEASE HAND IN

UNIVERSITY OF TORONTO
FACULTY OF ARTS AND SCIENCE

TERM TEST #2

CSC 104H

DURATION — 50 MINUTES

PLEASE HAND IN

LAST/FAMILY NAME: _____

FIRST/GIVEN NAME: _____

Do NOT turn this page until you have received the signal to start.
(In the meantime, please fill out the identification section above, and read the instructions below.)

This test consists of 4 questions on 6 pages (including this one).
When you receive the signal to start, please make sure that your copy of the test is complete.
Please answer questions in the space provided.
You will earn 20% for any question you leave blank or write "I cannot answer this question," on.

Good Luck!

QUESTION 1. [9 MARKS]**PART (A)** [3 MARKS]

Convert 77 to its binary representation, briefly showing your steps.

PART (B) [3 MARKS]

Convert the binary representation 1011010 to its decimal representation, briefly showing your steps.

PART (C) [3 MARKS]

Add the numbers whose binary representations 1110110 and 1010100 (without converting them to decimal notation). Show your work (i.e. when you “carry” a 1).

QUESTION 2. [5 MARKS]

Assume LLL is defined by:

```
(define LLL (list (list "cat" "dog")
                  "bird"
                  (list "snail")))
```

PART (A) [2 MARKS] Show the intermediate step, and the result value, for:

```
(map list? LLL)
```

PART (B) [1 MARK] Show the result value for: (filter list? LLL)

PART (C) [2 MARKS] Show the intermediate steps, and the result value, for:

```
(apply append (filter list? LLL))
```

QUESTION 3. [15 MARKS]

PART (A) [4 MARKS]


Define function `r-b` according to its 'check-expect', and fill in its contract.


; Design:

```
(check-expect (r-b (square 10 "solid" "green"))
              (beside (rotate 90 (square 10 "solid" "green"))
                      (rotate -90 (square 10 "solid" "green"))))
```

; r-b : ->

PART (B) [6 MARKS] Here are three 'check-expect's for a function 'tower':
 (check-expect (tower 0) \triangle) ; The width of the triangle is 10.

(check-expect (tower 1) )

(check-expect (tower 2) )

Complete the following three 'check-expect's WITHOUT drawing any images manually by hand.
 Use 'r-b' from Part (A) to help you.
 Use (tower 0) inside your second 'check-expect'.
 Use (tower 1) inside your third 'check-expect'.

(check-expect (tower 0)

)

(check-expect (tower 1)

)

(check-expect (tower 2)

)

PART (C) [5 MARKS]

Write the function 'tower'.

; tower : image -> image

QUESTION 4. [7 MARKS]

Assume the following function has been defined:

```
(define (f a-list)
  (cond [(= (length a-list) 1) a-list]
        [else (append (list (first a-list))
                        (f (rest a-list))
                        (list (first a-list)))])])
```

PART (A) [4 MARKS]

Show the result value of each of the following expressions:

```
(f (list "A"))
```

```
(f (list "A" "B"))
```

Show the result of the following expression, briefly showing your steps (include at least two illustrative intermediate steps):

```
(f (list "A" "B" "C"))
```

1: _____/ 9

2: _____/ 5

3: _____/15

4: _____/ 7

TOTAL: _____/36