CSC 324H5 F 2018 Midterm
Duration — 50 minutes
Aids allowed: none

**Student Number:** |_|_|_|_|_|_|_|_|_|_|_|_|_|

**Last Name:** _____ **First Name:** _____

Lecture Section: L0102 Test Version: B Instructor: Lisa Zhang

---

*Do **not** turn this page until you have received the signal to start.*
(Please fill out the identification section above, **write your name on the back of the test**, and read the instructions below.)
*Good Luck!*

---

This test consists of 4 questions on 10 pages (including this page). *When you receive the signal to start, please make sure that your copy is complete.* Comments are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code.
If you use any space for rough work, indicate clearly what you want marked.

# 1: _____/10
# 2: _____/ 5
# 3: _____/ 4
# 4: _____/ 6

TOTAL: _____/25

---

Total Pages = 10

## Question 1. [10 MARKS]

Answer each of the following multiple choice questions by circling all answers that apply. There may be multiple correct answers per question, or no correct answers at all.

1. Which of the following expressions match the macro pattern `(<a> <b> <c> ...)`?

   (a) `(list)`

   (b) `(a b)`

   (c) `(a ((b) (c)) ((d)))`

   (d) `(a) (b) (c) (d)`

2. Which of the following definitions are tail recursive?

   (a)
   ```
   (define (f lst acc)
     (f lst acc))
   ```

   (b)
   ```
   (define (f lst acc)
     (if (empty? lst)
         (f '() acc)
         (f (rest lst) (cons (first lst) acc))))
   ```

   (c)
   ```
   (define (f lst acc)
     (if (empty? lst)
         (reverse acc)
         (f (rest lst) (cons (first lst) acc))))
   ```

   (d)
   ```
   (define (f lst acc)
     (if (empty? lst) lst acc))
   ```

3. Consider the following Haskell code:

   ```
   g1 x y = x + y
   g2 x y z = x + (y * z)
   ```

   Which of the following expressions will result in an error?

   (a) `q = g1 3`

   (b) `q = g1 3 4 5`

   (c) `q = (g2 3 4 5)`

   (d) `q = (((g2 3) 4) 5)`

4. Which of the following are valid expressions of the below grammar?

```
<expr> = INTEGER
       | (* INTEGER <expr>)
       | (+ INTEGER <expr>)
```

   (a) `+ 2 3`

   (b) `(+ (/ 1 0) (* 3 (- 6 5)))`

   (c) `(+ 1 (+ 0 0))`

   (d) `(+ (+ 0 0) 1)`

5. Consider the following macro:

```
(define-syntax my-macro
  (syntax-rules ()
    [(my-macro ((<x> <y>) ...))
     (list 'a <x> ...)]
    [(my-macro (<x> ...))
     (list 'b <x> ...)]
    [(my-macro ((<x> <y> ...)))
     (list 'c <y> ...)]))
```

   Which macro expansions below are correct?

   (a) `(my-macro ((1 2)))` expands to `(list 'a 1)`

   (b) `(my-macro (()))` expands to `(list 'a)`

   (c) `(my-macro (1 2))` expands to `(list 'b 1 2)`

   (d) `(my-macro ((1)))` expands to `(list 'c)`

## Question 2. [5 MARKS]

Write a **tail-recursive** function `my-map` using Racket. Like the built-in function `map`, `my-map` takes as arguments a function `f` and a list `lst`, and returns a new list that consists of each element of `lst` applied to `f`.

```
>>> (my-map (lambda (x) (* 2 x)) (list 1 2 3))
'(2 4 6)
>>> (my-map (lambda (x) (* 2 x)) (list))
'()
```

Do **not** call `map`. Use recursion directly, and design your function to be tail-recursive. You may use the list function `reverse`.

## Question 3. [4 MARKS]

The following Haskell data type encodes a binary tree of integers.

```
data BTree = BLeaf Int
           | BNode Int BTree BTree
```

Here are some examples of `BTree` objects:

```
BLeaf 3
BNode 1 (BLeaf 3) (BLeaf 5)
BNode 1 (BNode 2 (BLeaf 3) (BLeaf 4)) (BLeaf 5)
```

Write a function `numNodes` that counts the number of `BNode`s (i.e. non-leaf nodes) in a binary tree.

```
numNodes:: BTree -> Int
```

## Question 4. [6 MARKS]

### Part (a) [3 MARKS]

Write a Racket class `Junk` whose objects are constructed by providing one non-empty list, like this:

```
(Junk '(10 20 30))
```

Objects of the class should have two methods: `first` that returns the first element of the provided list, and `prepend` that takes an argument and adds it to the beginning of the list.

### Part (b) [3 MARKS]

Suppose we use two attribute hash tables in a class macro: one for the object and one for the class. (1) What would be stored in each hash table? (2) When looking up an attribute, in which order should the hash tables be checked to find the attribute? Explain.

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*