CS 270 - Lab 7

M. Boady, B. Char, J. Johnson, G. Long, S. Earth

1 Introduction

You may work in teams of **one** or **two** students. Submit one copy for the entire group.

Write your answers on this lab sheet. Only what is written on this lab sheet will be graded.

This lab is due at the end of the class period. You may not continue to work on it once class has ended.

This lab contains 4 questions.

Grading

- 25 points - Putting everyones names on this page
- 20 points - Earned for each correct question (Answer is fully correct)
- 5 points - Earned for **partial credit** any question

No additional point amounts can be earned. You cannot earn 7 points on a question for example.

The maximum score for a lab is 100. If you get everything correct, that adds up to 105 points but will be reduced to 100.

A question will be marked correct as long as it covers all requirements of the question. It does not need to be perfect, but must be fully correct. A single typo or very minor issue where the intention is clear and all requirements are met would still earn full points.

We want you to complete questions fully, not try to earn partial credit on multiple questions. You may ask your Professor/Course assistant questions during lab.

Labs must be done in the presence of an instructor and/or course assistant or credit will not be given.

Partners should alternate each class day which person is physically typing and submitting the lab.

Do not split up the problems or you risk not finishing on time due to the cumulative nature of the questions.

Enter the name of the student in the group

Member 1 (submitter): Elan Rubin

Member 2: Josh Koo

Question 1 :

We can create lists using only integers, the **cons**, and **null** commands.

For example, (1 (2 3)) is created from (cons 1 (cons (cons 2 (cons 3 null)) null))

Write a Racket Command, using only **cons**, **null**, and **integers** to generate each of the following lists.

Spacing between parenthesis is just for readability. Single Quotes at the front of lists are not shown because the PDF does not always render them correctly. Racket would display these with single quotes.

(a) (1 2 3)

```
(cons 1 (cons 2 (cons 3 null)))
```

(b) ((1) (2))

```
(cons (cons 1 null) (cons (cons 2 null) null))
```

(c) (((1)))

```
(cons (cons (cons 1 null) null ) null)
```

(d) (() ())

```
(cons null (cons null null))
```

(e) ((1 2) (3))

```
(cons (cons 1 (cons 2 null)) (cons (cons 3 null) null))
```

Question 2 :

Define a recursive racket function (seq n) that returns a list of integers from n down to 0. If the input is negative, return the null list. This is useful because the last number you want to work on is 0, all negative numbers returning the empty list is a simple starting point. Your solution must use recursion.

For example:

```
(seq -1) ;Returns '()
(seq 0) ;Returns '(0)
(seq 1) ;Returns '(1 0)
(seq 2) ;Returns '(2 1 0)
(seq 3) ;Returns '(3 2 1 0)
(seq 4) ;Returns '(4 3 2 1 0)
```

(a) Provide the Input and Output Contract for this function

```
; input-contract: positive or negative integer
; output-contract: list of integers that contains n to 0, or null list
```

(b) Write your function using valid Racket Syntax below.

```
#lang racket

(define (seq n)
  (if (< n 0)
      null
  (cons n (seq(- n 1)))))
```

Question 3 :

Define a recursive racket function (mult5 n k) that returns a list of n integers that are multiples of 5 starting with k. Your solution must use recursion.

You may **assume** that k will always be a multiple of 5.

For example:

```
(mult5 0 10) ;Returns '()
(mult5 1 10) ;Returns '(10)
(mult5 2 10) ;Returns '(10 15)
(mult5 3 10) ;Returns '(10 15 20)
(mult5 4 10) ;Returns '(10 15 20 25)
(mult5 5 10) ;Returns '(10 15 20 25 30)
(mult5 6 90) ;Returns '(90 95 100 105 110 115)
(mult5 7 25) ;Returns '(25 30 35 40 45 50 55)
(mult5 8 5) ;Returns '(5 10 15 20 25 30 35 40)
```

(a) Provide the Input and Output Contract for this function

```
; input-contract: two integers, the first one representing the starting number, and the second representing the starting point
; output-contract: list of integers that are multiples of 5, starging at the second input
```

(b) Write your function using valid Racket Syntax below.

```
(define (mult5 n k)
  (if (equal? n 0)
      null
  (cons k (mult5 (- n 1) (+ k 5)))))
```

Question 4 :

Define a recursive racket function (largest L) finds the largest number in an unsorted list L. Your solution **must** use recursion.

You may **assume** that L will be null or only contain positive integers.

For example:

```
(largest null) ;Returns 0
(largest '(1 2)) ;Returns 2
(largest '(2 1)) ;Returns 2
(largest '(192837465));Returns 9
(largest '(1 2 3 4 5 6 7 8 9)) ;Returns 9
(largest '(987654321));Returns 9
(largest '(10 20 100 50 92 72 99)) ;Returns 100
```

(a) Provide the Input and Output Contract for this function

```
;input-contract: the input is a list of integers
;output-contract: the output is the largest integer in the list
```

(b) Write your function using valid Racket Syntax below.

```
(define (largest L)
  (cond
    [(null? L) 0]
    [(null? (rest L)) (first L)]
    [else
     (if (> (first L) (largest (rest L)))
         (first L)
         (largest (rest L)))]))
```