CS 1101 A Term 2021 Exam 3 (150 points)

Pledge: _____ (2)

Question 1: _____ (30)

Question 2: _____ (30)

Question 3: _____ (30)

Question 4a: _____ (20)

Question 4b: _____ (20)

Question 5a: _____ (20)

Question 5b: _____ (8 Bonus)

TOTAL: _____ (150 or 160 with Bonus)

You have 90 minutes to complete this exam (plus 7 "Canvas" minutes–an extra bit of time for you to make your ultimate submission to Canvas). You do not need to show templates, but you may receive partial credit if you do. You also do not need to show test cases or examples of data definitions except in those cases where they are specifically requested, but you are free to develop them if they will help you write the desired programs.

Your programs may contain only the following BSL/ISL/ASL/Racket constructs:

define define-struct cond else if local begin max min

empty? cons? cons first rest list append length

\+ − ∗ / = < > <= >= **and or not**

string=? string-length string-append substring add1 sub1 error format

predicates for any defined data types

filter map andmap ormap foldr foldl build-list apply lambda

set! set-structure!

Plus, of course, any operators introduced by **define-struct** (including mutators)

Additionally, you may use any constants you find necessary, for example, empty, true, false, 0, etc.

Pledge (2 points):

I pledge on my honor that I am taking this Exam on my own, with help only from the provided starter file and DrRacket (if I have so chosen) and absolutely no one else, *sans* notes, book closed, search engines idled, Help Desk unconsulted. I further pledge not to discuss the Exam in any way with anyone until Friday, October 15, 2021 because I understand that there are students with circumstances that may cause them to take the Exam after I have completed it.

"My brain is open. . . ."

a.) `#true`
b.) `#false`

1. Higher-Order Functions (30 points)

a). (6 points) Create `(list -1 0 1 2 3)` with an expression involving at least two higher-order functions. (You may define a helper function for use in the expression if you wish, but it is not necessary.)

b). (6 points) Briefly explain why the lengths of the lists returned by

```
(map abs (filter positive? (list -1 0 1 2 3)))
```

and

```
(filter positive? (map abs (list -1 0 1 2 3)))
```

are not the same length. It is NOT enough merely to compute the lists or their lengths.

c). (6 points) The higher-order function `foldr` has the following signature:

```
foldr: (X  Y -> Y)  Y  ListOfX  ->  Y
```

Briefly explain why the isolated `Y`—the function's middle parameter—must match the `Y` that is the return value. That is, why must the components starred (*) below match?

```
foldr: (X  Y -> Y)  Y*  ListOfX  ->  Y*
```

d). (6 points) Briefly explain why a function whose return value comes from a `set!` call cannot adequately be tested with `check-expect`.

e). (6 points) The call `(newline)` takes no arguments and always returns `(void)`. Briefly explain how a function call that takes no arguments can return a value other than `(void)` and a different value on every call.

2. Higher-Order Functions on Lists of Structs (30 points)

Nomanisan Island is a bird watcher's paradise. The Island's forests echo with the songs and sounds of painted warblers and drowsy woodpeckers. The shores of the Island teem with birds found nowhere else in the world such as the streaked gargull and the paisley-footed booby. Birders flock from around the world, binoculars in hand, in hopes of spotting rarities only Nomanisan Island can offer them.

Here are some data definitions:

```
(define-struct sighting (spotter species flock# banded? eco-status))
;;a Sighting is a (make-sighting String String Natural Boolean String)
;;interp: represents a sighting where
;;        spotter is the name of the person reporting the sighting
;;        species is the bird's species
;;        flock# is the number of such birds spotted together simultaneously
;;        banded is #true if the spotter notes a biologist's band on any bird's leg
;;        eco-status is "least concern" or "threatened" or "endangered"

;;a ListOfSighting is one of:
;;  empty
;;  (cons Sighting ListOfSighting)
```

Using filter and/or map, write a function satisfying the following signature/purpose:

```
;;multi-endangered-spotters: Natural ListOfSighting -> ListOfString
;;consumes number representing a flock size and a list of sightings
;;returns a list of the names of spotters who report any endangered species
;;in a group of the given size or larger, that is, with flock# >= given number
```

Include a signature and purpose for any helper function(s) you write.

3. Accumulator-Style Programming (30 points)

Nomanisan Island is a bird watcher's paradise. The Island's forests echo with the songs and sounds of painted warblers and drowsy woodpeckers. The shores of the Island teem with birds found nowhere else in the world such at the gargull and the paisley-footed booby. Birders flock from around the world, binoculars in hand, in hopes of spotting rarities only Nomanisan Island can offer them.

Here are some data definitions:

```
(define-struct sighting (spotter species flock# banded? eco-status))
;;a Sighting is a (make-sighting String String Natural Boolean String)
;;interp: represents a sighting where
;;        spotter is the name of the person reporting the sighting
;;        species is the bird's species
;;        flock# is the number of such birds spotted together simultaneously
;;        banded is #true if the spotter notes a biologist's band on any bird's leg
;;        eco-status is "least concern" or "threatened" or "endangered"

;;a ListOfSighting is one of:
;;  empty
;;  (cons Sighting ListOfSighting)
```

Using accumulator-style recursion, write a function satisfying the following signature/purpose:

```
;;multi-endangered-spotters: Natural ListOfSighting -> ListOfString
;;consumes number representing a flock size and a list of sightings
;;returns a list of the names of spotters who report any endangered species
;;in a group of the given size or larger, that is, with flock# >= given number
```

Include a signature and purpose for any helper function(s) you write.

4a. Mutable Variables (20 points)

The summit of Mount Donne on Nomanisan Island is such a popular spot that local authorities have had to implement a reservation system to control traffic flow.

```
(define-struct reservation (name time-of-day duration vip?))
;;a Reservation is a (make-reservation String Natural Natural Boolean)
;;interp: represents an reservation for Mount Donne on Nomanisan Island where
;;        name is the name person making the reservation
;;        time-of-day is the hour of the reservation (0-23 military time)
;;        duration is the length of the reservation in minutes
;;        vip? is #true if the person making the reservation is a VIP
;;             (Very Important Person)

;;a ListOfReservation is one of:
;;  empty
;;  (cons Reservation ListOfReservation)

;;All daily reservations for Mount Donne are contained in the list MT.DONNE-REZ
```

Dignitaries from the Archipelago of Macronarnia are touring Nomanisan Island. The Grand Poobah of the Island wants to make their visit as relaxed as possible. To this end, the Grand Poobah has ordered that all VIP reservations for Mount Donne be extended by 60 minutes.

Write a function that satisfies the following signature/purpose:

```
;;vip+60:  ListOfReservation -> ListOfReservation
;;consumes a list of reservations
;;adds 60 minutes to the duration of all VIP reservations
;;EFFECT: Possibly changes some reservations in the list
```

Include a signature and purpose for any helper function(s) you write.

4b. Mutable Variables (20 points)
The summit of Mount Donne on Nomanisan Island is such a popular spot that local authorities have had to implement a reservation system to control traffic flow.

```
(define-struct reservation (name time-of-day duration vip?))
;;a Reservation is a (make-reservation String Natural Natural Boolean)
;;interp: represents an reservation for Mount Donne on Nomanisan Island where
;;        name is the name person making the reservation
;;        time-of-day is the hour of the reservation (0-23 military time)
;;        duration is the length of the reservation in minutes
;;        vip? is #true if the person making the reservation is a VIP
;;             (Very Important Person)

;;a ListOfReservation is one of:
;;  empty
;;  (cons Reservation ListOfReservation)

;;All daily reservations for Mount Donne are contained in the list MT.DONNE-REZ
```

Because of global climate change, the annual migration of the speckled sloth–expected next month–began at midnight. The sloths are expected to complete their crossing of the Mount Donne summit road later this morning. Local officials have decided to close the road until noon, and therefore all morning reservations must be cancelled.

Write a function that satisfies the following signature/purpose:

```
;;cancel-reservations: Natural -> (void)
;;consumes an integer representing a time of day and removes all reservations
;;scheduled before the given time from MT.DONNE-REZ and produces (void)
;;returns error "No Reservations Removed" if no reservations are cancelled
;;EFFECT: MT.DONNE-REZ may be shortened

;;HINT:  Name your parameter 'a-time'.  'time' is a reserved word you cannot overload.
```

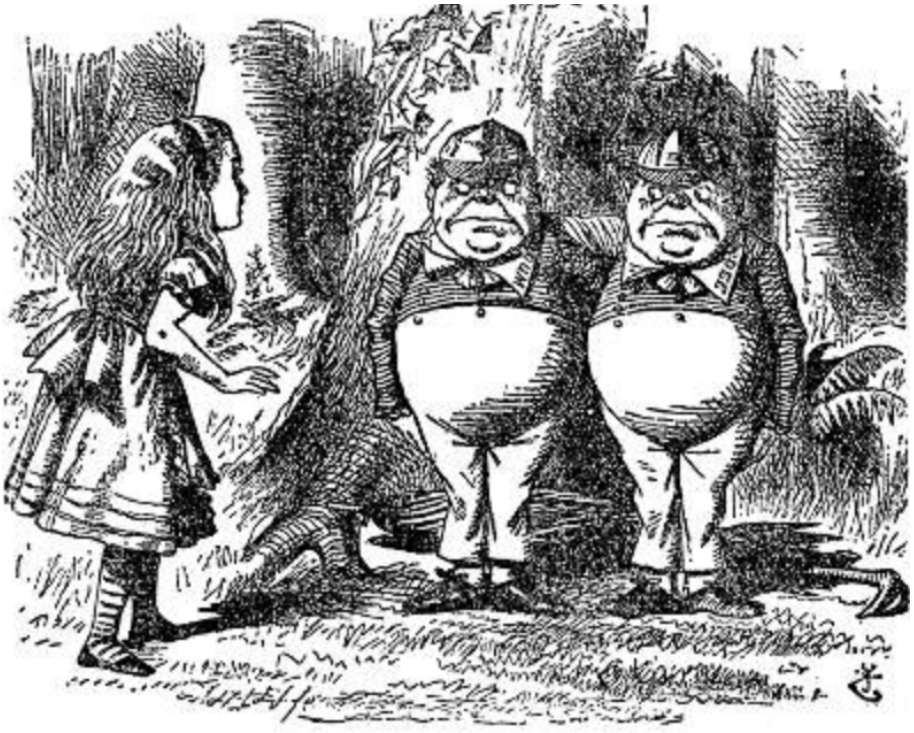
Be sure to include a test sequence to demonstrate that your function operates properly.

Include a signature and purpose for any helper function(s) you write.

5a. Data With Cycles (20 points)
Here are some data definitions:

```
(define-struct character (name stories buddy))
;;A Character is a (make-character String ListOfString Character)
;;interp: represents a (possibly fictional) character where
;;        name is the character's name
;;        stories is a list of books, movies, etc. the character appears in
;;        buddy is the character's best friend

;;a ListofString is one of:
;;  empty
;;  (cons String ListOfString)
```



Tweedledum and Tweedledee first appeared in John Byrom's "Nursery Rhymes" and then most famously in Lewis Carroll's Alice adventure "Through the Looking-Glass".

Write a sequence of expressions that correctly–and *mutually*–defines `TWEEDLEDUM` and `TWEEDLEDEE` as characters, each of whom is the other's best friend, under the data definition above.

5b. BONUS: Data With Cycles (8 points)
Here are some data definitions:

```
(define-struct character (name stories buddy))
;;A Character is a (make-character String ListOfString Character)
;;interp: represents a (possibly fictional) character where
;;        name is the character's name
;;        stories is a list of books, movies, etc. the character appears in
;;        buddy is the character's best friend

;;a ListofString is one of:
;;  empty
;;  (cons String ListOfString)
```

b). (8 Bonus Points)
SPOILER ALERT! In Mel Brooks' 1987 space opera parody "Spaceballs", John Candy plays Barf, the co-pilot of mercenary hero Lone Starr's spaceship *Eagle 5*.
As a Mog, half-man/half-dog, Barf is his own best friend.[1]

Write an expression or a sequence of expressions to correctly define `BARF` as a Character in terms of himself, or explain why such a definition is impossible.

[1] `https://www.youtube.com/watch?v=kJ26RApuQJo`