Name:_____

Question 0 (20 points)	
Question 1 (10 points)	
Question 2 (20 points)	
Question 3 (20 points)	
Question 4 (15 points)	
Question 5 (15 points)	
Total	

You have 75 minutes to complete the exam. You may have with you a single piece of paper with notes on both sides. You may have *only* your sheet of notes, a single writing implement, and the exam on your desk (e.g., no cell phones, laptops, water bottles, etc.).

You do not need to show contracts, signatures, purposes, templates, or test cases unless specifically asked. However, they may help you earn partial credit if you can not provide a complete solution.

Question 0: Terminology (20 points, 1 each)

Match the term to the *BEST* letter. Some letters may not be used (choose "Z: None of the above"). Some letters may be used more than once (choose any one).

```
;; a HealthNumber is a number between [1, 10]
;; where 1 is dead and 10 is excellent health.

(define-struct loc (lat lon))
;; make-loc: number number -> Loc
;; lat and lon are the latitude and longitude
;; GPS coordinates, respectively.
(define LOC1 (make-loc 39.68 75.75))

(define-struct fish (loc health size))
;; make-fish: Loc HealthNumber number -> Fish
;; loc is the fish's current GPS coordinate
;; health is fish health, size is fish weight in oz.
(define FISH1 (make-fish LOC1 10 400))

;; a BirdLifeString is one of
;; "juvenile", "egg-laying", "foraging", "geriatric"

(define-struct bird (loc health male? cycle))
;; make-bird: Loc HealthNumber boolean
;;            BirdLifeString -> Bird
;; loc is the bird's current GPS coordinate
;; health is bird health, male? is true if male,
;; otherwise female/false and cycle is the life
;; cycle state.
(define BIRD1 (make-bird LOC1 8 false "egg-laying"))

;; a MarineAnimal is
;; -- a fish, or
;; -- a bird.
#;
(define (marine-animal-fun ama)
  (cond [(fish? ama) (fish-fun ama)]
        [(bird? ama) (bird-fun ama)]))

;; ma-loc: MarineAnimal --> Loc
(check-expect (ma-loc FISH1) LOC1)

(define (ma-loc ama)
  (cond [(fish? ama) (fish-loc ama)]
        [(bird? ama) (bird-loc ama)]))
```

A B C D E F G H I J K L M N O P Q

_____ comment

_____ interval

_____ enumeration

_____ itemization

_____ parameter

_____ argument

_____ template

_____ clause

_____ constant

_____ unit test

_____ MarineAnimal

_____ predicate

_____ constructor

_____ selector

_____ structure definition

_____ constant definition

_____ function definition

_____ signature

_____ function call

_____ boolean

Question 1: Simple Function (10 points)

Design a function `celsius->fahrenheit` that converts temperatures in degrees Celsius to temperatures in degrees Fahrenheit. For reference, a temperature T in degrees Fahrenheit (°F) is equal to the temperature T in degrees Celsius (°C) times 9/5 plus 32.

Show the signature/contract, purpose, test(s), and definition for the function.

Question 2: Cond (20 points)

Beats per minute (BPM) is a unit typically used as a measure of tempo in music. For ease of use, composers often use a set of basic tempo markings to indicate how quickly a piece of music should be played.

Develop a function `classify` that consumes the BPM for a song and produces a corresponding tempo marking according to the following chart:

BPM	Tempo Marking
<76	Largo
76–120	Andante
121–168	Allegro
>168	Presto

Show the signature, test(s), and function definition, including any additional data definitions you require.

Question 3: Data Definitions and Templates (20 points)

Consider a Reminders app that allows users to set a reminder to go off either on a specific date and time or when arriving or leaving a location. All reminders are handled in a single unified interface. A time reminder records the reminder text, the date, and the time of the reminder. A location reminder records the GPS location (as two numbers, latitude and longitude), the reminder text, and whether to remind when you arrive or when you leave the location.

Develop data definitions for `Reminder` and **either** `TimeReminder` or `LocationReminder`. Provide additional data definitions for any other types that are necessary.

Question 4: Structures (15 points)

Consider the following definitions for a date:

```
(define-struct date (day month year))
;; make-date: DayNumber MonthNumber YearNumber -> Date
;; interp: day   is the day [0, 31]
;;         month is the month [1, 12]
;;         year  is the year [>= 0]
```

a position:

```
(define-struct location (latitude longitude))
;; make-posn Number Number -> Location
;; interp: latitude  is the latitude component
;;         longitude is the longitude component
```

and an event:

```
(define-struct event (when where))
;; make-event: Date Location -> Event
;; interp: when  is date of the event
;;         where is the location of the event
```

Design a function `conflict?` that consumes two events and produces `true` if the events have the same date and `false` otherwise.

Question 5: Lists (15 points)

(1) **[2 pts]** Is the following a correct (well-formed) recursive data definition? Circle your answer.

```
;; a ListOfBoolean is
;; -- empty
```

a. No, it does not have a self-referential case
b. No, it does not have a base case
c. No, it does not have a self-referential case or a base case
d. Yes, it is a correct recursive data definition

(2) **[2 pts]** Is the following a correct (well-formed) recursive data definition? Circle your answer.

```
;; a ListOfWidget is
;; -- empty
;; -- (cons Widget ListOfWidget)
```

a. No, it does not have a self-referential case
b. No, it does not have a base case
c. No, it does not have a self-referential case or a base case
d. Yes, it is a correct recursive data definition

(3) **[11 pts]** Design a function called `count-evens` that consumes a list of numbers and counts how many even numbers are in the given list. You may use the function `even?` to check whether a given number is even.