CSc 372 Quiz 4: SML Concepts from SA2

Name	
UA Email	

Keep your eyes on your own quiz or exam and study guide. Do not access the internet, any other person, or AI to help you during the quiz. When you are done, raise your hand so someone can pick up your quiz and study guide. Once your quiz has been picked up, you can quietly use any devices, but do not communicate with others who are still taking the quiz.
Put a `?` for questions you have no idea on.

Question 1 [2 pts]. Write a recursive function in SML that sums the squares of even numbers in a list?

Question 2 [2 pts]. Write a function in SML that sums the squares of even numbers in a list using foldl?

Question 3 [1pt]. True or False …………………………………………………........
A value's type constrains the way it may be used in a program.

Question 4 [1 pt]. Fill box with answer or a ? …………………… ☐

What is Lisp lacking such that it has to check if x is null to see if the list is empty versus the SML example shown?

```
fun length [] = 0
  | length (x::xs) = 1 + length xs;

(define (length x)
   (if (null? x)
        0
        (+ 1 (length (cdr x)))))
```

Question 5 [1 pt]. Put the letter for the best answer or a ? in this box …………………...... ☐

Which of these patterns will match a list of characters with the first character being an 'a' and zero or more characters in the rest of the list?

```
A.(#"a" :: [])     B.(#"a" :: _)     C.[#"a", _]    D.[a, xs]
```

Question 6 [1pt]. True or False ……………………………………………....... ☐

The expression `Char.ord c >= Char.ord #"A"` evaluates to `true` only when a capital letter value is bound to `c`.

Question 7 [1 pt]. Put the letter for the best answer or a ? in this box …………………..... ☐

Assuming the partition function you implemented for SA2, `partition (fn x => x mod 2 = 0) [1, 2, 3, 4, 5]` should return

   A. `[2, 4, 1, 3, 5]`
   B. `([2, 4])`
   C. `[1, 3, 5]`
   D. `([2, 4], [1, 3, 5])`

Question 8 [1 pt]. Put the letter for the best answer or a ? in this box …………………..... ☐

`partition (fn x => x = 3) [1, 2, 3, 4, 5]` should return

   A. `[3, 3, 3, 3, 3]`
   B. `[3, 1, 2, 4, 5]`
   C. `([3], [])`
   D. `([3], [1, 2, 4, 5])`