CSE 341 | Section 6

Racket: Basics, Lists, and Delayed Evaluation

Q1 (Scope Review): Consider the following Racket code:

```
(define x 3)
(define f1
  (lambda (x)
    (let ([y (+ x 1)])
      (+ y x))))
```

```
(define x 3)
(define f2
  (let ([y (+ x 1)])
    (lambda (x)
      (+ y x)))))
```

What is \((f1\ 2)\) bound to? ______________

What is \((f2\ 2)\) bound to? ______________

Q2 (Functions): Define a function named `digit-sum` that accepts an integer \(n\) and returns the sum of its digits. For example, the call \((\text{digit-sum}\ 341)\) should return 8 (since \(3 + 4 + 1 = 8\)). If passed a negative parameter, the function should return the negative sum of the digits. For example, the call \((\text{digit-sum}\ -341)\) should return -8 (since \(-(3 + 4 + 1) = -8\)). You may assume that only integers are passed as arguments. Use racket functions `remainder` and `quotient`.

Q3 (Functions): Write a Racket function named `star-string` that accepts an integer argument \(n\) and returns a string of stars (asterisks) \(2n\) long (i.e., \(2\) to the \(n\)th power). Use the racket `string-append` function. For example:

<table>
<thead>
<tr>
<th>Call</th>
<th>Output</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>(star-string 0)</td>
<td>&quot;*&quot;</td>
<td>(2^0 = 1)</td>
</tr>
<tr>
<td>(star-string 1)</td>
<td>&quot;**&quot;</td>
<td>(2^1 = 2)</td>
</tr>
<tr>
<td>(star-string 2)</td>
<td>&quot;****&quot;</td>
<td>(2^2 = 4)</td>
</tr>
<tr>
<td>(star-string 3)</td>
<td>&quot;********&quot;</td>
<td>(2^3 = 8)</td>
</tr>
</tbody>
</table>
Q4 (Lists): Define a function `count-in-range` that takes a list of numbers and two numbers `lo` and `hi` and returns the count of elements in the list that are between `lo` and `hi` (inclusive).

Q5 (Lists): Define a function `partition-parity` that takes a list as an argument and returns a pair of lists such that the first list holds all even values of the argument and the second list holds all odd (maintaining original order). Use the racket function `even?` or `odd?`. 