Assignment :)

Language Level: Intermediate Student with Lambda

Question List:

Q1) Normal List

Q2) List w/ inexact output

Q3) Unknown Data Type (set-conversion)

Q4) Custom Output

Q5) Forbidden Functions

Q6) Function Consumes More Than 1 Parameter

Q7) Require Teachpack/Files

Q1: (Normal List)

Write a function called sum1, which consumes a list of Nat, and produces the sum of all the elements in that list. (submit “sum1.rkt”)

Eg. (sum1 (list 1 2 3)) => 6 (testgen mode: 'list)

Q2: (List w/ Inexact Output)

Write a function called sum2, which consumes a list of Floating Num, and produces the sum of all the elements in that list. Hint: use check-within to test. (submit “sum2.rkt”)

Eg. (sum2 (list -1.1 1.1 2.45)) => 2.45 (testgen mode: 'list with inexact output)

Q3: (Unknown Data Type(set-conversion))

Write a function called tree-copy, which cosumes a BT, and produces a copy of that BT ;;A BT is either empty, or (make-btnode num BT BT) (submit “tree-copy.rkt”)

Eg. (tree-copy (make-btnode 1 empty empty)) => (make-btnode 1 empty empty) (testgen mode: 'non-list)

Q4: (Custom Output)

Write a function called subsets1, which consumes a list of Nat and produces a list of all of its subsets. (submit “subset1.rkt”)

Eg. (subsets1 ’(1 2)) => (list ’(1 2) ’(1) ’(2) ’()). (testgen mode: 'custom) (from Fall2013CS135 A10 BONUS)

Q5: (Forbidden Functions)

Write a function called my-reverse, which consumes a list of any value, and produce that list in reverse order. You cannot use reverse.(submit “my-reverse.rkt”)

Eg. (my-reverse (list 1 2 3)) => (list 3 2 1) (set up forbidden functions)

Q6: (Function Consumes More Than 1 Parameter)

Consider the following predicate function that consumes three Booleans and produces a Boolean:

```
(define (cond-mystery? a b c)
        (cond
            [(not a) c]
            [else b]))
```

Write the scheme function bool-mystery? so that it is equivalent to cond-mystery? Except that it uses only a Boolean expression (i.e.: it does not have a cond expression). (submit “bool-mystery.rkt”).

Q7: (Require Teachpack/Files)

Write a function called lcm (least common multiple) which consuems 2 parameter and produce the least common multiple. Note: (lcm m n) = (/ (* m n) (gcd m n)). gcd will provide to you in “gcd.rkt”.