Problemario: Programación lógica

Instrucciones

Utilizando la implementación del lenguaje embebido miniKanren indicada por tu profesor, resuelve los problemas de programación lógica que se presentan a continuación. Coloca tu código en un solo archivo. Cada función debe incluir un comentario con una breve descripción de lo que hace.

1. `(lasto lst x)`: Función lógica que tiene éxito si `x` es el último elemento de `lst`.

   Ejemplos:

   ```
   (run 1 (q) (lasto '(1 2 3 4) q))
   ⇒ (4)

   (run 1 (q) (lasto () q))
   ⇒ ()

   (run 5 (q) (lasto q 'a))
   ⇒ ((a) (_.0 a) (_.0 _.1 a) (_.0 _.1 _.2 a) (_.0 _.1 _.2 _.3 a))
   ```

2. `(butlasto lst result)`: Función lógica que tiene éxito si `result` contiene los mismos elementos que `lst` excepto el último.

   Ejemplos:

   ```
   (run 1 (q) (butlasto '(1 2 3 4) q))
   ⇒ ((1 2 3))

   (run 1 (q) (butlasto q '(1 2 3 4)))
   ⇒ ((1 2 3 4 _.0))

   (run 1 (q) (butlasto '(1 2 3 4) '(1 2 3 4)))
   ⇒ ()

   (run 3 (p q) (butlasto p q))
   ⇒ (((_.0) ()) ((_.0 _.1) (_.0)) ((_.0 _.1 _.2) (_.0 _.1)))
   ```

3. `(enlisto lst result)`: Función lógica que tiene éxito si `result` contiene los mismos elementos que `lst` pero cada uno colacado dentro de una lista.

   Ejemplos:

   ```
   (run 1 (q) (enlisto '(a b c d e) q))
   ⇒ (((a) (b) (c) (d) (e)))

   (run 1 (q) (enlisto q '(a b c d e)))
   ⇒ ()

   (run 1 (q) (enlisto q '((a) (b) (c) (d) (e))))
   ⇒ ((a b c d e))

   (run 3 (p q) (enlisto p q))
   ⇒ ((() ()) ((_.0) ((_.0))) ((_.0 _.1) ((_.0) (_.1))))
   ```

4. `(duplicateo lst result)`: Función lógica que tiene éxito si cada elemento en `lst` aparece duplicado en `result`.

   Ejemplos:

```
(run 1 (q) (duplicateo '(1 2 3 4) q))
⇒ ((1 1 2 2 3 3 4 4))

(run 1 (q) (duplicateo q '(a a b b c c)))
⇒ ((a b c))

(run 1 (q) (duplicateo q '(a a b b c c d)))
⇒ ()

(run 3 (p q) (duplicateo p q))
⇒ ((() ()) ((_.0) (_.0 _.0)) ((_.0 _.1) (_.0 _.0 _.1 _.1)))
```

5. `(removeo x lst result)`: Función lógica que tiene éxito si se puede eliminar la primera ocurrencia de `x` en `lst` obteniendo `result`.

   Ejemplos:

```
(run 1 (q) (removeo 3 '(1 2 3 4) q))
⇒ ((1 2 4))

(run 1 (q) (removeo 5 '(1 2 3 4) q))
⇒ ()

(run 1 (q) (removeo q '(1 2 3 4) '(1 2 4)))
⇒ (3)

(run 5 (q) (removeo 0 q '(1 2 3 4)))
⇒ ((0 1 2 3 4) (1 0 2 3 4) (1 2 0 3 4) (1 2 3 0 4) (1 2 3 4 0))

(run* (p q) (removeo p '(1 2 3 4) q))
⇒ ((1 (2 3 4)) (2 (1 3 4)) (3 (1 2 4)) (4 (1 2 3)))
```

6. `(reverseo lst result)`: Función lógica que tiene éxito si `result` es la reversa de `lst`.

   Ejemplos:

```
(run 1 (q) (reverseo '(a b c d) q))
⇒ ((d c b a))

(run 1 (q) (reverseo q '(a b c d)))
⇒ ((d c b a))

(run 1 (q) (reverseo '(a b c d) '(e d c b a)))
⇒ ()
```

7. `(palindromeo lst)`: Función lógica que tiene éxito si `lst` es un palíndromo (se lee igual de izquierda a derecha que de derecha a izquierda).

   Ejemplos:

   ```
   (run 1 (q) (palindromeo '(a b c d c b a)) (== q 'yes))
   ⇒ (yes)

   (run 1 (q) (palindromeo '(a b c d e f g)) (== q 'yes))
   ⇒ ()

   (run 1 (q) (palindromeo '()) (== q 'yes))
   ⇒ (yes)
   ```

8. `(rotateo lst result)`: Función lógica que tiene éxito cuando `result` es el resultado de girar `lst` hacia la izquierda una posición. En otras palabras, el primer elemento de `lst` se convierte en el último elemento de `result`.

   Ejemplos:

   ```
   (run 1 (q) (rotateo '(a b c d e) q))
   ⇒ ((b c d e a))

   (run 1 (q) (rotateo q '(a b c d e)))
   ⇒ ((e a b c d))

   (run 1 (q) (rotateo '(a b c d e) '(a b c d e)))
   ⇒ ()
   ```

9. `(evensizeo lst)` y `(oddsizeo lst)`: Estas dos funciones lógicas deben definirse de manera mutuamente recursiva. Es decir, cada una debe definirse en términos de la otra. Estas funciones tienen éxito si el número de elementos en `lst` es par o impar, respectivamente.

   Ejemplos:

   ```
   (run 1 (q) (evensizeo '(a b c d)) (== q 'yes))
   ⇒ (yes)

   (run 1 (q) (oddsizeo '(a b c)) (== q 'yes))
   ⇒ (yes)

   (run 1 (q) (oddsizeo '(a b c d)) (== q 'yes))
   ⇒ ()

   (run 4 (q) (evensizeo q))
   ⇒ (() (_.0 _.1) (_.0 _.1 _.2 _.3) (_.0 _.1 _.2 _.3 _.4 _.5))
   ```

10. `(splito lst a b)`: Función lógica que tiene éxito cuando al dividir `lst` se obtiene `a` y `b`. Los elementos primero, tercero, quinto, etc. de `lst` van en `a`, mientras que los elementos segundo, cuarto, sexto, etc. van en `b`.

    Ejemplos:

    ```
    (run 1 (p q) (splito '(a 1 b 2 c 3 d 4 e) p q))
    ⇒ (((a b c d e) (1 2 3 4)))

    (run 1 (q) (splito q '(a b c d e) '(1 2 3 4)))
    ⇒ ((a 1 b 2 c 3 d 4 e))

    (run 1 (q) (splito '(a b c) '(a b c) q))
    ⇒ ()

    (run 1 (q) (splito '(a b c) '(a c) q))
    ⇒ ((b))
    ```

11. `(swappero a b lst result)`: Función lógica que tiene éxito solo si `result` contiene los mismos elementos que `lst` excepto que cada ocurrencia de `a` se intercambia por `b`, y viceversa.

    Ejemplos:

    ```
    (run 1 (q) (swappero 'a 'b '(a b a b b b a) q))
    ⇒ ((b a b a a a b))

    (run 1 (q) (swappero 'a 'b q '()))
    ⇒ (())

    (run 1 (q) (swappero 'purr
                         'kitty
                         '(soft kitty warm kitty little ball of fur
                           happy kitty sleepy kitty purr purr purr) q))
    ⇒ ((soft purr warm purr little ball of fur happy purr sleepy purr kitty kitty kitty))
    ```

12. `(equalo lst)`: Función lógica que tiene éxito solo si todos los elementos contenidos en `lst` se unifican con el mismo valor. La función siempre debe tener éxito si `lst` está vacía o tiene un solo elemento.

    Ejemplos:

    ```
    (run* (q) (equalo '()) (== q 'yes))
    ⇒ (yes)

    (run 1 (q) (equalo '(a a a a a a)) (== q 'yes))
    ⇒ (yes)

    (run 1 (q) (equalo '(a a a a b a)) (== q 'yes))
    ⇒ ()

    (run 5 (q) (equalo q))
    ⇒ (() (_.0) (_.0 _.0) (_.0 _.0 _.0) (_.0 _.0 _.0 _.0))
    ```

13. `(subseto a b)`: Función lógica que tiene éxito si todos los elementos de la lista `a` son miembros a su vez también de la lista `b`.

    Ejemplos:

    ```
    (run 1 (q) (subseto '(b d) '(a b c d e)) (== q 'yes))
    ⇒ (yes)

    (run 1 (q) (subseto '(a b c d) '(a b c d)) (== q 'yes))
    ⇒ (yes)

    (run 1 (q) (subseto '(a b c d e) '(a b c d)) (== q 'yes))
    ⇒ ()

    (run 3 (q) (subseto q '(a b c d)))
    ⇒ (() (a) (b))
    ```

14. `(compresso lst result)`: Función lógica que tiene éxito si `result` tiene los mismos elementos que `lst` excepto que si algunos de estos están repetidos de manera consecutivos se reemplazan por una sola instancia. El orden de los elementos no debe modificarse.

    Ejemplos:

    ```
    (run 1 (q) (compress '(a a a a b c c a a d e e e e) q))
    ⇒ ((a b c a d e))

    (run 1 (q) (compress '(a b c d) q))
    ⇒ ((a b c d))

    (run 1 (q) (compress '(a a a a a a a a a a) q))
    ⇒ ((a))

    (run 5 (q) (compress q '(a b)))
    ⇒ ((a b) (a a b) (a b b) (a a a b) (a a b b))
    ```