Estructuras Discretas 2022-1
Práctica 4: Listas

Facultad de Ciencias, UNAM

Fecha de entrega: 03 de diciembre del 2021

Resuelva los siguientes ejercicios, la práctica es en **parejas**. En el archivo `listas.py` se encuentra la firma correspondiente para cada función y donde tiene que completar con código. Después de haber resuelto todo, puede ejecutar dentro de la carpeta el comando `python3 -m unittest -v test_listas.py` para asegurarse que la salida de cada función es correcta, es importante mencionar que esto no asegura que esté bien hecho, por ejemplo, que no haya usado funciones ya definidas en `Python` que resuelvan directamente el problema pero podrá darse cuenta si alguna función es incorrecta.

Para esta práctica **todas las funciones deben ser recursivas**, de otro modo no se considerarán correctas.

Para la entrega, siga las especificaciones publicadas en `Classroom`.

1. Define la función recursiva `repr` que devuelve una representación en cadena de una lista.

   Ejemplo:

```
>>> l = Lista()
>>> repr(l)
    ()
>>> l = Lista([1,2 3])
>>> repr(l)
    (1 (2 (3 ())))
>>> l = Lista(['a','b' 'c'])
>>> repr(l)
    ('a' ('b' ('c' ())))
```

2. Define la función recursiva `agrega_principio` la cuál agrega un elemento al principio de la lista.

   Ejemplo:

```
>>> l1 = Lista()
>>> l1.agrega_principio(6)
>>> l1
    (6 ())
>>> l2 = Lista([3,5,4])
>>> l2.agrega_principio(8)
>>> l2
    (8 (3 (5 (4 ()))))
```

3. Define la función recursiva `agrega_final` la cuál agrega un elemento al final de la lista.

   Ejemplo:

```
    >>> l1 = Lista()
    >>> l1.agrega_final(6)
    >>> l1
    (6 ())
    >>> l2 = Lista([3,5,4])
    >>> l2.agrega_final(8)
    >>> l2
    (3 (5 (4 (8 ()))))
```

4. Define la función recursiva `longitud` que devuelve el número de elementos de una lista.

   Ejemplo:

```
>>> l = Lista()
>>> l.longitud()
    0
>>> l = Lista([1,2,3])
>>> l.longitud()
    3
```

5. Define la función recursiva `contiene`, que dado un elemento, devuelve `true` si el elemento se encuentra en la lista y `false` en otro caso.

   Ejemplo:

```
>>> l1 = Lista()
>>> l1.contiene(1)
    False

>>> l2 = Lista([1,6,10])
>>> l2.contiene(6)
    True

>>> l2.contiene(2)
    False
```

6. Define la función recursiva `copia`, que dada la lista original, devuelve una nueva lista idéntica a esta.

   Ejemplo:

```
    >>> l1 = Lista()
    >>> l2 = Lista([1,2,3])

    >>> l1.copia()
    ()

    >>> l2.copia()
    (1 (2 (3 ())))

    >>> l2 == l2.copia()
    True

    >>> l2 is l2.copia()
    False
```

7. Define la función recursiva `concatena`, que dada la lista original, recibe una lista como argumento y la concatena al final de la original.

   Ejemplo:

```
    >>> l1 = Lista()
    >>> l2 = Lista([1,2,3])
    >>> l3 = Lista([4,5,6])

    >>> l1.concatena(l2)
    (1 (2 (3 ())))

    >>> l2.concatena(l3)
    (1 (2 (3 (4 (5 (6 ()))))))
```

8. Define la función recursiva `reversa`, que devuelve una nueva lista con el orden invertido de sus elementos.

   Ejemplo:

```
>>> l = Lista([1,2,3,4])
>>> l.reversa()
    (4 (3 (2 (1 ()))))

>>> l = Lista(['a','d','i','k'])
>>> l.reversa()
    ('k' ('i' ('d' ('a' ()))))
```

9. Define la función recursiva `mapea`, que devuelve la lista resultante de aplicar una función f a cada elemento de la lista.

   Ejemplo:

```
>>> l = Lista([10,70,100])
>>> l.mapea(lambda x: x//10)
    (1 (7 (10 ())))

>>> l.mapea(lambda x: (x+10)//2)
    (10 (40 (55 ())))
```

10. Define la función recursiva `filtra` que devuelve una lista con los elementos que cumplen con la condición `f`.

```
>>> l = Lista([10,35,70,80,98,100])
>>> l.filtra(lambda x: x<80)
    (10 (35 (70 ())))

>>> l.filtra(lambda x: x%10 == 0)
    (10 (70 (80 (100 ()))))
```