Práctica de Listas con recursividad

1. Construya una función en Python a llamar insert_d(ref, ele, lista), que reciba 2 elementos y una lista. La función debe generar una lista en la cual se inserte el ele después del elemento ref, ***cada vez que aparezca en la lista de entrada***.

```
>>> insert_d('x', 'y', ['a', 'b', 'c', 'x', 'd', 'e']]
['a', 'b', 'c', 'x', 'y', 'd', 'e']

>>> insert_d(4, 5, [1, 2, 3, 4, 6, 7, 4, 8])
[1, 2, 3, 4, 5, 6, 7, 4, 5, 8]
```

2. Escriba una función alterna(lista) que reciba una lista no nula y verifique si sus elementos son alternados entre pares e impares.

```
>>> alternada([2, 5, 8, 7, 10])
True

>>> alternada([1, 2, 3, 4, 6])
False

>>> alternada([1, 2, 3, 4, 5])
True
```

3. Se debe recibir una lista no nula y devolver otra compuesta por dos sublistas, la primera tendrá a los elementos que ocupan un lugar impar, mientras que la segunda contendrá a los que ocupan un lugar par. Escriba la función separa(lista) utilizando recursividad de pila, de forma que se obtengan resultados cómo los siguientes:

```
>>> separar(['a', 'b', 'c', 'd', 'e'])
[['b', 'd'], ['a', 'c', 'e']]
```

4. Escribir una función recursiva para replicar los elementos de una lista una cantidad `n` de veces.

```
replicar ([1, 3, 3, 7], 2)
>>> [1, 1, 3, 3, 3, 3, 7, 7]
```

5. Escriba una función **listaAscendente(lista)**, que recibe una lista y devuelve True si los elementos de la lista están en orden ascendiente y False en caso contrario.

```
listaAscendente([2,4,5,6,7])

>>> True

listaAscendente([8,7,2,4,5])

>>> False
```

6. Construya una función llamada **split(ele, lista).** Esta función toma una lista y la parte en sublistas usando como punto de corte el elemento recibido como argumento. A continuación se muestran ejemplos de cómo debe comportarse la función, asumo puede tener letras y números:

```
>>> split('x', [])
[]
>>> split('x', [1, 4, 7, 'x', 1, 2, 3])
[[1, 4, 7], [1, 2, 3]]
>>> split('x', [1, 4, 7, 'x', 'x', 1, 2, 3])
[[1, 4, 7],[],[1, 2, 3]]
>>> split('x', [1, 4, 7, 'x', 'x', 'x',1, 2, 3])
[[1, 4, 7],[], [], [1, 2, 3]]
>>> split('x', [1, 4, 7, 'x', 'x', 'x',1, 2, 3,'x'])
[[1, 4, 7],[], [], [1, 2, 3]]
```