Instituto Tecnológico de Costa Rica

Escuela de Computación

Introducción a la Programación / Taller de programación.

Profesor: Jeff Schmidt Peralta

EJERCICIOS PARA II EXAMEN PARCIAL.

1. En una lista se reciben los salarios de n empleados de una empresa. Escriba una función (recursiva e iterativa) `sueldos(lista)` que obtenga:

   a) Cuántos empleados ganan más de 1 millón.

   b) Cuántos empleados ganan menos de 1 millón.

   c) Cuál es el salario más alto.

   d) El promedio de salarios.

```
>>> sueldos([482400,1975600,250000,2675000])
[2, 2, 2675000, 1345750]
```

2. Escriba una función por medio de iteración `mas_par(num)` que recibe una lista y retorna un valor boolean que indica si la lista dada tiene más números pares que impares. La función debe mostrar un comportamiento similar a los siguientes ejemplos:

```
>>> mas_par([2,3,8,149,34])
True
```

```
>>> mas_par([22,3,43])
False
```

3. Escriba una función a llamar `iota(num)` que reciba un número natural y obtenga una lista en orden ascendente de números naturales menores al número dado.

```
>>> iota(5)
[0, 1, 2, 3, 4]
>>> iota(1)
[0]
```

4. Escriba una función `cuales` que reciba un número (que debe ser entero) y retorne una lista con los dígitos entre 0 y 5.

```
>>> cuales(482401)
([4,2,4,0,1])
>>> cuales(97)
([])
```

```
>>> cuales(0 )
([0])
```

5. Escriba una función iterativa `ultimo_par(vector)` que recibe un vector y retorna el último número par del vector dado. No puede invertir ni destruir el vector. No puede usar índices negativos.

```
>>> ultimo_par([23, 72, 149, 34])
34
>>> ultimo_par([12,5,21,3])      >>>ultimo_par([1,3,5,7])
12                               No hay ningun par
```

6. Escriba una función **iterativa** `cambie_repetidos(vec)` que reciba un vector de números enteros y sustituya todos los dígitos repetidos por 0.

```
>>> cambie_repetidos([1,2,5,3,1,5])
[0,2,0,3,0,0]
>>> cambie_repetidos([1,2,5,3,4,6,7])
[1,2,5,3,4,6,7]
```

7. Escriba una función llamada `partir(lista)` que reciba una lista y produce una lista de listas, donde cada sublista se forma con los números hasta encontrar la aparición de ceros, es decir, el cero va a ser el punto de corte para formar los números.
   a. Utilizando recursividad de cola
   b. Utilizando iteración

```
>>> partir([1,23,0,29,2,0,1,0,34])
[[1, 23], [29, 2], [1], [34]]
>>> partir([1,2,0,0,0,0,0])
[[1, 2]]
```

8. Escriba una función **iterativa** `elimine_todos(ele, lista)` que reciba un elemento y una lista y elimine todas las apariciones del elemento en la lista, aún si el elemento se encuentra en una sublista (a cualquier nivel).

   Sugerencias:
   a. hacer una representación intermedia de la lista original que no tenga el concepto de n niveles
   b. hacer una mezcla de iteración con recursividad.

```
>>> elimine_todos(4, [4, 5, [[[3, 4], 5], [3]], 4, 8])
[5, [[[3], 5], [3]], 8]
```