

Auxiliar Extra:

Todos los problemas deben ser resueltos en *Python*, utilizando estrictamente la Receta de Diseño entregada a lo largo del curso. Use nombres apropiados para funciones y variables, y testee cada vez que sea posible.

1. Listas

1.1. Rut

En este ejercicio, vamos a aprender como se calcula la clave del RUT, a decir la cifra verificadora que se encuentra despues de la raya. Esta cifra se calcula con una combinacion de multiplicaciones, sumas, y modulos de los numeros del RUT. El resultado es un digito entre 0 y 9, o la letra K. Por ejemplo, la clave del RUT 11.222.333 es 9. El calculo se hace de tal manera:

1. Convertir el RUT en una lista de cifras (1, 1, 2, 2, 2, 3, 3, 3).
2. Invertir la lista de cifras del RUT (3, 3, 3, 2, ...)
3. Multiplicar cada numero por un numero de la secuencia: 2, 3, 4, 5, 6, 7, 2, 3, ... E.g. en el ejemplo sera:
 $3 * 2, 3 * 3, 3 * 4, 2 * 5, 2 * 6, 2 * 7, 1 * 2, 1 * 3$.
4. Sumar los resultados del paso 3.
5. Calcular el resto de la division por 11 (usar el operador de modulo, %).
6. La cifra final es 11 menos el resultado del paso 5. Si el resultado es 10, devolver "K". Si es 11, devolver 0. Las cifras se convierten en string con `str()`.

Implementar, siguiendo la receta de diseño, las funciones necesarias para el calculo de la clave del RUT. Para convertir un numero en la lista de sus cifras, se puede ocupar la division entera para 10, y el modulo para 10.

Ahora suponemos que existe la función que calcular la clave correcta de un RUT descrita previamente. Dado esta función, definir una otra función, que toma una lista de personas, y devuelve la lista de las personas que tienen un RUT equivocado. Una persona es un dato compuesto:

```
struct.crear("persona", "nombre rut clave")  
# nombre y clave son string, rut es un num
```

2. Árboles

1. Cree una estructura *NodoABB*, que tenga dos hijos, izquierdo y derecho (ambos del tipo *NodoABB*) y un valor (int).
2. Cree una función que a partir de la estructura árbol que se puede crear con la estructura *NodoABB*, calcule el promedio de todos los valores presentes en el árbol. Para hacer esto, cree una función recursiva que recorra todo el árbol en *preorden*, es decir, primero recorre toma el valor del nodo, luego los de los nodos a la de izquierda y finalmente los de la derecha.
3. Ahora, dados dos enteros m y n , tal que $m \leq n$ entregue una lista con todos los elementos en el árbol que están en el intervalo $[m, n]$ (la lista debe estar ordenada)