Mega ayudantía - Primer parcial

> Realizado por: Gabriel Salas - Ing. en Telemática

Lecturas de código

¿Qué imprime el siguiente código?

```
lnumeros = [23, 4, 14, 49, 72, 89, 100, 139, 233]
ele = lnumeros[ : ]
ele.reverse()
ele.sort(reverse = True)

if max(lnumeros) == ele[0]:
    print("Son números iguales")
else:
    print("No son números iguales")
```

¿Qué imprime el siguiente código?

```
def listarReverso(lpalabras, caracter):
    lretornar = []
    for palabra in lpalabras:
        if caracter in palabra.lower():
            lretornar.append(palabra)
    return lretornar.reverse()

latam = ["Ecuador", "Colombia", "Bolivia", "Chile", "Argentina", "Perú"]
print(listarReverso(latam, "a"))
```

¿Qué imprime el siguiente código?

```
lista = [89, 45, 23, 17, 55, 95, 13, 41, 28, 11]
lista.sort()
promedio = sum(lista) // len(lista)
print(promedio)
menores = []
i = 0
while lista[i] < promedio:
    menores.append(lista[i])
    i += 1
print(menores)
```

Ejercicio 1

Defina la función `listarJugadores(ldatos, pais)` que recibe una lista con elementos con el siguiente formato: `"Nombre del Jugador|Nacionalidad"`. La función **debe retornar una lista** con los nombres de los jugadores que pertenecen a dicho país.

Ejercicio 2

Defina la función `mostrarTopAlmacenamiento(lapps, entero)` que recibe una lista de aplicaciones con elementos con el siguiente formato: `"Aplicación-Espacio"` y un número entero positivo menor o igual a la cantidad de elementos que contiene la lista. La función **debe motrar por pantalla** un top definido por el número recibido con las aplicaciones con mayor consumo de almacenamiento de un dispositivo.

En su programa principal, **asuma que ya tine la lista de aplicaciones**. Implemente un código que genere un número aleatorio entre 3 y un número menor a la cantidad de elementos de la lista de aplicaciones e implemente la función definida. Ejemplo:

```
apps = ["Microsoft Teams-520MB", "Espol Académico-38MB", "Facebook-366MB", "WhatsApp-1500MB", "Tik tok-1200MB", ...]
```

> Recuerde que la longitud y los elementos de la lista pueden variar.

Salida (con n = 3):

```
Top 3 aplicaciones con mayor consumo de espacio:
1. WhatsApp: 1500 MB
2. Tik tok: 1200 MB
3. Microsoft Teams: 520 MB
```

Ejercicio 3

Defina la función `siglas(frase, palabras_comunes)` que recibe una cadena y una lista con palabras repetidas dentro de esa cadena. **Asuma que la cadena contiene las palabras en minúscula y separadas por espacio, que no contiene comas ni puntos**. La lista también contiene cadenas en minúsculas. La función **debe retornar una cadena** con las iniciales en mayúscula de cada palabra que no sea común en la lista, a excepción de la primera y la última.

Ejemplo:

```
frase = "para todo epsilon mayor a cero existe un delta mayor a cero"
palabras_com = ["el", "de", "a", "un", "para"]
```

Salida:

```
PTEMCEDMC
```

Ejercicio 4

Escriba un programa que realice lo siguiente:

1. Genere un número aleatorio entre 20 y 150.
2. Genere una lista con la cantidad de elementos igual al número generado anteriormente con números aleatorios entre 100 y 5500.
3. Pida al usuario un número. Mientras no ingrese un número entre el menor y el mayor de los números de la lista generada, continúe solicitando.
4. Muestre por pantalla los números de la lista que son divisibles para el número ingresado por el usuario.

Ejercicio 5

Defina la función `crearCarta(lsimbolos, lvalores)` que recibe 2 listas: una lista que representa los símbolos de los naipes, y una lista con los valores de los naipes. La función **debe retornar una lista con 5 cartas únicas aleatorias con formato** `"valor-símbolo"`.

En un programa principal, **suponga que ya tiene las listas de símbolos y valores**. Implemente la función y muestre la lista desordenada. Escoja una carta aleatoria y solicite al usuario que escoja una de las cartas ingresando valor-símbolo. Si el usuario adivia la carta escogida, muestre un mensaje de felicitaciones, caso contrario, muestre un mensaje de que perdió.

Las listas que debe suponer que posee son:

```
lsimb = ["T", "E", "C", "D"]
lval = ["A", "2", "3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K"]
```

Ejemplo de consola:

```
["A-T", "5-E", "J-C", "6-C", "6-D"]
Adivine la carta: 5-E
¡Felicidades, adivinaste!
```