Ejercicios Complementarios:

Conceptos a aplicar

Resolver la práctica aplicando los siguientes conceptos cuando sea necesario:

• Modularización y definición adecuada de argumentos
• Estructuras de control
• Pasaje de parámetros
• Condicionales simples y compuestos |

Ejercicio 1

a) Realice una función que procesa la información de alumnos de la UNAB. De cada alumno se conoce legajo, nombre, apellido, contraseña. El procesamiento termina cuando se ingresa el número de legajo 0. La función deberá retornar una lista con la información procesada.

b) Realice una función llamada imprimir_alumno que recibe como parámetro una lista con los datos de un alumno (Legajo, nombre, apellido, contraseña), los datos del alumno serán mostrados por pantalla con la forma:

- Legajo:XXXX
- Nombre:XXXX
- Apellido:XXXX
- Contraseña:XXXXX

c) Realice un función llamada legajo_menor que recibe como parámetro una lista de alumnos, de cada alumno se conoce la información (legajo, nombre, apellido, contraseña), la función debe buscar cuál es el alumno con el menor legajo dentro de la lista y retornarlo.

d) Realice un función llamada nombre_mas_largo que recibe como parámetro una lista de alumnos. La función debe buscar cuál es el alumno con el nombre más largo dentro de la lista y retornarlo.

e) Realice un función llamada controlar_clave que recibe como parámetro un alumno,(legajo, nombre, apellido, contraseña). La función debe controlar si la contraseña es mayor a 6 caracteres y termina con un número, deberá imprimir un mensaje especificando el error cometido en caso de no cumplir las condiciones o bien imprimir los datos del alumno si la clave cumple con todas las condiciones.

f) Realice una función llamada verificar_claves que recibe como parámetro una lista de alumnos, la función deberá controlar por cada alumno si la contraseña que usa cumple con: ser mayor a 6 caracteres y terminar con un número.
Construir un menú, el menú deberá permitir ingresar 5 opciones, La opción 0 permite salir del menú, el resto de las opciones permiten:

- imprimir los datos de todos los alumnos con el formato pedido en el punto a)
- imprimir los datos del alumno que tiene el legajo más chico.

- imprimir los datos del alumno que tiene el nombre más largo.
- Imprimir si las contraseñas de cada alumno cumplen con un tamaño mayor a 6 caracteres y terminan con un número.

Ejercicio resuelto por ustedes:

```
class Alumnos():
    def __init__(self, legajo=None, nombre=None , apellido=None, clave=None):
        self.legajo = legajo
        self.nombre = nombre
        self.apellido = apellido
        self.clave = clave
        self.lista = []
    def procesar_alumnos(self):
        self.legajo=int(input("escriba el legajo(0 para salir)"))
        while self.legajo!=0:
            self.nombre = input("ingrese nombre: ")
            self.apellido = input("ingrese apellido: ")
            self.clave = input("ingrese clave: ")
            self.lista.append(Alumnos(self.legajo, self.nombre, self.apellido,
self.clave))
            self.legajo=int(input("escriba el legajo(0 para salir)"))

    def legajo_menor(self):
        # Encontrar el alumno con el menor legajo
        alumno_con_menor_legajo = min(self.lista, key=lambda alumno:
alumno.legajo)
        print(f"El alumno con el menor legajo es:
{alumno_con_menor_legajo.nombre} con legajo {alumno_con_menor_legajo.legajo}")

    def nombre_mas_largo(self):
        # Encontrar el alumno con el nombre más largo
        alumno_con_nombre_mas_largo = max(self.lista, key=lambda alumno:
len(alumno.nombre))
        print(f"El alumno con el nombre más largo es:
{alumno_con_nombre_mas_largo.nombre} con legajo
{alumno_con_nombre_mas_largo.legajo}")

    def controlar_clave(self):
        # Controlar la clave del alumno
        alumno = self.lista[int(input("Ingrese el legajo del alumno: "))-1] #
buscamos el alumnos por legajo, restmos uno ya que los indices empiezan con 0
pero el legajo no puede. (si el legajo no va en orden, bueno, jaja saludos)
        if len(alumno.clave) > 6 and alumno.clave[-1].isdigit():    #utilizamos
el metodo -1 para acceder al ultimo elemento de en este caso el string "clave"
y hacemos las validaciones requeridas en el ejercicio
            print(f"Los datos del alumno con legajo {alumno.legajo} son:
{alumno}")
        else:
            print("La contraseña no cumple con las condiciones")
```

```
    """copy-paste del metodo anterior."""
    def verificar_claves(self):
        # Controlar la clave de todos los alumnos
        for alumno in alumnos.lista:
            if len(alumno.clave) > 6 and alumno.clave[-1].isdigit():
                print(f"Los datos del alumno con legajo {alumno.legajo} son:
{alumno}")
            else:
                print("La contraseña no cumple con las condiciones")


    def __str__(self) -> str:
        return f"Legajo: {self.legajo}\nNombre: {self.nombre}\nApellido:
{self.apellido}\nContraseña: {self.clave}"




alumnos=Alumnos()

alumnos.procesar_alumnos()

def mostrar_alumnos():
    for x in range(len(alumnos.lista)):
        print(f"{alumnos.lista[x]}")


def menu():
    while True:
        print("""\nMenú:
        \n1. Imprimir los datos de todos los alumnos.
        \n2. Imprimir los datos del alumno con el legajo más chico.
        \n3. Imprimir los datos del alumno con el nombre más largo.
        \n4. Controlar la clave de un alumno.
        \n5. Controlar la clave de todos los alumnos.
        \n0. Salir.""")
        opcion = int(input("Seleccione una opción: "))
        if opcion == 1:
            mostrar_alumnos()
        elif opcion == 2:
            alumnos.legajo_menor()
        elif opcion == 3:
            alumnos.nombre_mas_largo()
        elif opcion == 4:
            alumnos.controlar_clave()
        elif opcion == 5:
            alumnos.verificar_claves()
        elif opcion == 0:
            break
        else:
            print("Opción no válida. Intente nuevamente.")
menu()
```

Dejo otra manera, en este caso sin usar clases, pero usando manejo de archivos y excepciones

```
import os

ALUMNOS_FILE = "alumnos.txt"

def cargar_alumnos():
    alumnos = []
    try:
        if os.path.exists(ALUMNOS_FILE):
            with open(ALUMNOS_FILE, 'r') as file:
                for line in file:
                    try:
                        legajo, nombre, apellido, contraseña =
line.strip().split(',')
                        alumnos.append((int(legajo), nombre, apellido,
contraseña))
                    except ValueError:
                        print("Error al procesar la línea del archivo: " +
line)
    except IOError:
        print("Error al intentar leer el archivo.")
    return alumnos

def guardar_alumnos(alumnos):
    try:
        with open(ALUMNOS_FILE, 'w') as file:
            for alumno in alumnos:
                try:
                    file.write("{},{},{},{}\n".format(alumno[0], alumno[1],
alumno[2], alumno[3]))
                except Exception as e:
                    print("Error al escribir los datos del alumno " +
str(alumno) + ": " + str(e))
    except IOError:
        print("Error al intentar escribir en el archivo.")

def procesar_alumnos():
    alumnos = cargar_alumnos()
    while True:
        try:
            legajo = int(input("Ingrese el legajo (0 para terminar): "))
        except ValueError:
            print("Error: Legajo debe ser un número.")
            continue

        if legajo == 0:
            break

        nombre = input("Ingrese el nombre: ")
        apellido = input("Ingrese el apellido: ")
```

```
            contraseña = input("Ingrese la contraseña: ")
            alumnos.append((legajo, nombre, apellido, contraseña))

        guardar_alumnos(alumnos)
        return alumnos

    def imprimir_alumno(alumno):
        legajo, nombre, apellido, contraseña = alumno
        print("- Legajo: " + str(legajo))
        print("- Nombre: " + nombre)
        print("- Apellido: " + apellido)
        print("- Contraseña: " + contraseña)

    def legajo_menor(alumnos):
        if not alumnos:
            return None
        menor = min(alumnos, key=lambda x: x[0])
        return menor

    def nombre_mas_largo(alumnos):
        if not alumnos:
            return None
        mas_largo = max(alumnos, key=lambda x: len(x[1]))
        return mas_largo

    def controlar_clave(alumno):
        legajo, nombre, apellido, contraseña = alumno
        if len(contraseña) <= 6:
            print("Error: La contraseña del alumno " + nombre + " es menor a 6
    caracteres.")
        elif not contraseña[-1].isdigit():
            print("Error: La contraseña del alumno " + nombre + " no termina en un
    número.")
        else:
            imprimir_alumno(alumno)

    def verificar_claves(alumnos):
        for alumno in alumnos:
            controlar_clave(alumno)

    def menu():
        alumnos = procesar_alumnos()

        while True:
            print("\nMenú:")
            print("1. Imprimir datos de todos los alumnos")
            print("2. Imprimir alumno con el legajo más chico")
            print("3. Imprimir alumno con el nombre más largo")
            print("4. Verificar contraseñas de los alumnos")
            print("0. Salir")

            try:
                opcion = int(input("Seleccione una opción: "))
```

```
        except ValueError:
            print("Error: La opción debe ser un número.")
            continue

        if opcion == 0:
            break
        elif opcion == 1:
            for alumno in alumnos:
                imprimir_alumno(alumno)
        elif opcion == 2:
            menor_alumno = legajo_menor(alumnos)
            if menor_alumno:
                imprimir_alumno(menor_alumno)
            else:
                print("No hay alumnos en la lista.")
        elif opcion == 3:
            largo_alumno = nombre_mas_largo(alumnos)
            if largo_alumno:
                imprimir_alumno(largo_alumno)
            else:
                print("No hay alumnos en la lista.")
        elif opcion == 4:
            verificar_claves(alumnos)
        else:
            print("Opción no válida. Intente de nuevo.")

# Ejecutar el menú
menu()
```

Ejercicio 2

Nos contratan para realizar un sistema para una editorial. Se recibe un texto por teclado y se desea obtener la siguiente información, para esto deberá construir un menú que permita ingresar las diferentes opciones: (RESOLVER CADA PUNTO CON UNA FUNCION):

- La longitud total del texto.
- La cantidad de palabras componen el texto.
- La cantidad de oraciones que componen el texto.
- La cantidad de palabras que comienzan con vocal o con consonante, dependiendo del valor ingresado por el usuario.
- Buscar una palabra ingresada por el usuario y retornar la cantidad de veces que se encuentra en el texto.
- La cantidad de palabras que comienzan con mayúscula.
- La cantidad de caracteres que son números.
- La cantidad de palabras que comienzan con vocal y la cantidad de palabras que comienzan con consonante.
- Imprimir todas las palabras que terminan en infinitivo (terminadas en ar er o ir).

Resuelto por ustedes:

```
import  re
from collections import Counter

def longitud_total (texto):
    return len(texto)

#re.findall es la funcion del import re para manejar grandes cadenas de texto

def cantidad_palabras(texto):
    palabras= re.findall(r'\b\w+\b', texto)
    return len(palabras)
    #funcion que devuelve la longitud total del texto

def cantidad_oraciones(texto):
    oraciones = re.split(r'[.!?]', texto)
    return len([s for s in oraciones if s.strip()])
    #s strip filtra las oraciones vacias o solo con espacios
    #split  divide al texto segun los signos de puntuacion encerrados en
corchetes

def cantidad_palabras_vocal_consonante(texto, tipo = None):
    #con r'\b\w+\b' se delimita el espacio del texto a traves de la primer y
ultima b y de esa forma se encuentran las palabras
    #el w+ busca cualquier caracter en el string para contarlo tambien incluido
numero, letra y caracteres especiales
    palabras = re.findall(r'\b\w+\b', texto)
    if tipo == 'vocal':
        return sum(1 for palabra in palabras if palabra[0].lower() not in
'aeiou')
    elif tipo == 'consonante':
        return sum(1 for palabra in palabras if palabra [0].lower not in
'aeiou')
    return(sum(1 for palabra in palabras if palabra[0].lower in 'aeiou'), sum(1
for palabra in palabras if palabra[0].lower() not in 'aeiou'))
    #Se suman vocales y consonantes en caso de que no se especifique bien

def buscar_palabra(texto, palabra_buscada):
    palabras= re.findall(r'\b\w+\b', texto.lower())
    return palabras.count(palabra_buscada.lower())
    #se cuenta cuantas veces aparece la palabra buscada sin distincion de
mayuscula o minuscula, por eso el .lower()

def cantidad_palabras_mayus(texto):
    palabras = re.findall(r'\b\+w\b', texto)
    return sum(1 for palabra in palabras if palabra[0].isupper())
    #Aca se cuenta la cantidad de palabras que empiezan con mayuscula en el
texto .isupper()

def cantidad_numeros(texto):
    return sum(c.isdigit()for c in texto)
```

```
    #
def palabras_infinitivo(texto):
    infinitivo = re.findall(r'\b\w+(?: ar , er, ir)\b', texto)
    return infinitivo
    #Se delimita el texto con re.findall con las b y se buscan las
terminaciones especificadas dentro del parentesis

def menu():
    texto = input("Ingrese el texto: ")

    while True:
        print   ("\nMenú: ")
        print   ("1. Longitud total del texto")
        print   ("2. Cantidad de palabras")
        print   ("3. Cantidad de oraciones")
        print   ("4. Cantidad de palabras que comienzan con vocal o
consonante")
        print   ("5. Buscar palabra y contar ocurrencias")
        print   ("6. Cantidad de palabras que cominenzan con mayuscula")
        print   ("7. Cantidad de caracteres numericos")
        print   ("8. Cantidad de palabras que comienzan con vocal y
consonante")
        print   ("9. Imprimir palabras terminadas en infinitivo")
        print   ("0. Salir")

        opcion = input ("Elija una opción: ")

        if opcion == '0':
            break
        elif opcion == '1':
            print ("Longitud total del texto: ", longitud_total(texto))
        elif opcion == '2':
            print("Cantidad de palabras:", cantidad_palabras(texto))
        elif opcion == '3':
            print ("Cantidad de oraciones:", cantidad_oraciones(texto))
        elif opcion == '4':
            tipo = input ("Ingrese 'vocal' para palabras que comienzan con
vocal o 'consonante' para las palabras que cominenzan con consonante: ")
            print(f"Cantidad de palabras que cominenzan con {tipo}:",
cantidad_palabras_vocal_consonante(texto, tipo))
        elif opcion == '5':
            palabra_buscada = input("Ingrese la palabra a buscar: ")
            print("Cantidad de veces que aparece:", buscar_palabra(texto,
palabra_buscada))
        elif opcion == '6':
            print ("Cantidad de palabras que cominenzan con mayuscula:",
cantidad_palabras_mayus(texto))
        elif opcion == '7':
            print ("Cantidad de caracteres numericos:",
cantidad_numeros(texto))
        elif opcion == '8':
            vocales, consonantes = cantidad_palabras_vocal_consonante(texto)
            print ("Cantidad de palabras que comienzan con vocal:", vocales)
```

```
            print ("Cantidad de palabras que comienzan con consonantes:")
        elif opcion == '9':
            print ("Palabras terminadas en infinitivo",
palabras_infinitivo(texto))
        else:
            print ("Opcion no valida, intente nuevamente.")


menu()
```

Otra manera:

```
import re

# Función para obtener la longitud total del texto
def longitud_texto(texto):
    return len(texto)

# Función para contar la cantidad de palabras (ignorando la puntuación)
def contar_palabras(texto):
    palabras = re.findall(r'\b\w+\b', texto)
    return len(palabras)

# Función para contar la cantidad de oraciones (considerando más signos de
puntuación)
def contar_oraciones(texto):
    oraciones = re.split(r'[.!?;:]', texto)
    return len([oracion for oracion in oraciones if oracion.strip()])

# Función para contar palabras que empiezan con vocal o consonante, con soporte
para acentos
def contar_palabras_vocal_consonante(texto, tipo):
    palabras = re.findall(r'\b\w+\b', texto)
    vocales = "aeiouáéíóúAEIOUÁÉÍÓÚ"
    contador = 0
    if tipo == 'vocal':
        for palabra in palabras:
            if palabra[0] in vocales:
                contador += 1
    elif tipo == 'consonante':
        for palabra in palabras:
            if palabra[0] not in vocales and palabra[0].isalpha():
                contador += 1
    return contador

# Función para buscar una palabra en el texto (ignorando mayúsculas y
minúsculas)
def buscar_palabra(texto, palabra):
    palabras = re.findall(r'\b\w+\b', texto.lower())
    return palabras.count(palabra.lower())
```

```python
# Función para contar palabras que empiezan con mayúscula
def contar_mayusculas(texto):
    palabras = re.findall(r'\b\w+\b', texto)
    contador = 0
    for palabra in palabras:
        if palabra[0].isupper():
            contador += 1
    return contador

# Función para contar caracteres que son números
def contar_numeros(texto):
    return sum(c.isdigit() for c in texto)

# Función para contar palabras que empiezan con vocal y consonante
def contar_vocales_consonantes(texto):
    vocales = "aeiouáéíóúAEIOUÁÉÍÓÚ"
    palabras = re.findall(r'\b\w+\b', texto)
    contador_vocal = 0
    contador_consonante = 0
    for palabra in palabras:
        if palabra[0] in vocales:
            contador_vocal += 1
        elif palabra[0].isalpha():
            contador_consonante += 1
    return contador_vocal, contador_consonante

# Función para imprimir palabras que terminan en infinitivo
def palabras_infinitivo(texto):
    palabras = re.findall(r'\b\w+\b', texto)
    infinitivos = [palabra for palabra in palabras if palabra.endswith(('ar',
'er', 'ir'))]
    print("Palabras que terminan en infinitivo:", infinitivos)

# Función del menú interactivo
def menu():
    texto = input("Ingrese el texto: ")

    while True:
        print("\nMenú de opciones:")
        print("1. Mostrar la longitud total del texto")
        print("2. Mostrar la cantidad de palabras")
        print("3. Mostrar la cantidad de oraciones")
        print("4. Contar palabras que empiezan con vocal o consonante")
        print("5. Buscar una palabra en el texto")
        print("6. Mostrar la cantidad de palabras que empiezan con mayúscula")
        print("7. Mostrar la cantidad de caracteres que son números")
        print("8. Mostrar la cantidad de palabras que comienzan con vocal y con
consonante")
        print("9. Imprimir palabras que terminan en infinitivo (ar, er, ir)")
        print("0. Salir")

        opcion = input("Seleccione una opción: ")
```

```
        if opcion == "1":
            print("Longitud del texto:", longitud_texto(texto))
        elif opcion == "2":
            print("Cantidad de palabras:", contar_palabras(texto))
        elif opcion == "3":
            print("Cantidad de oraciones:", contar_oraciones(texto))
        elif opcion == "4":
            tipo = input("¿Contar palabras que empiezan con vocal o consonante?
(vocal/consonante): ").strip().lower()
            if tipo in ['vocal', 'consonante']:
                print(f"Cantidad de palabras que empiezan con {tipo}:
{contar_palabras_vocal_consonante(texto, tipo)}")
            else:
                print("Opción inválida.")
        elif opcion == "5":
            palabra = input("Ingrese la palabra a buscar: ")
            print(f"La palabra '{palabra}' aparece {buscar_palabra(texto,
palabra)} veces.")
        elif opcion == "6":
            print("Cantidad de palabras que empiezan con mayúscula:",
contar_mayusculas(texto))
        elif opcion == "7":
            print("Cantidad de caracteres que son números:",
contar_numeros(texto))
        elif opcion == "8":
            vocales, consonantes = contar_vocales_consonantes(texto)
            print(f"Palabras que comienzan con vocal: {vocales}, con
consonante: {consonantes}")
        elif opcion == "9":
            palabras_infinitivo(texto)
        elif opcion == "0":
            print("Saliendo del programa...")
            break
        else:
            print("Opción no válida, intente de nuevo.")

# Ejecutar el menú
menu()
```