Auxiliar 9

P1) Cree la función quitar(a,b), que recibe como parámetros dos números enteros a y b, que indican dos índices de una lista de python (a <=b). La función debe modificar la lista global L, eliminando todos los elementos ubicados entre los índices a y b (inclusive). La función entrega como resultado,la cantidad de elementos que fueron borrados de la lista. Si alguno de los índices se encuentra fuera del rango de la lista, la función solo elimina los valores que estén dentro del rango. Ejemplos:

```
L = [79, 45, 18, 64, 17, 24, 51]

quitar(7,9)-> entrega 0, y L queda como: L= [79,45, 18, 64, 17, 24, 51]

quitar(3,5)-> entrega 3, y L queda como: L= [79,45,18,51]


L = [79,45,18,64,17,24,51]

quitar(5,8)-> entrega 2, y L queda como: L= [79. 45,18,64,17]

quitar(2,2)-> entrega 1, y L queda como: L= [79,45,64,17]
```

Double-click (or enter) to edit

```
# quitarIterativo: int, int -> None
# funcion que quita los elementos en el intervalo de indices [a,b]
# quitarIterativo(3,5) entrega 3 y L queda como [79, 45, 18, 51]
def quitarIterativo(a,b):
  assert type(a) == int and type(b) == int
  global L
  # creamos una lista donde agregaremos los elementos que caigan dentro
  # del ramgo
  largo = len(L)
  eliminados = 0
  Laux = []
  # para cada elemento de la lista, si está en el rango, la agregamos, si no, la ignoramos
  for i in range(largo):
    if a <= i <= b:
      eliminados += 1
    else:
      Laux.append(L[i])
```

```
      Laux.append(L[i])
  # reemplazamos la lista global L, por la nueva lista que creamos
  L = Laux
  return eliminados
# Test
L = [79, 45, 18, 64, 17, 24, 51]
assert quitarIterativo(7,9) == 0
assert L == [79, 45, 18, 64, 17, 24, 51]
assert quitarIterativo(3,5) == 3
assert L == [79, 45, 18, 51]
L = [79, 45, 18, 64, 17, 24, 51]
assert quitarIterativo(5,8) == 2
assert L == [79, 45, 18, 64, 17]
assert quitarIterativo(2,2) == 1
assert L == [79, 45, 64, 17]
```

P2) Se le pide crear un sistema de notas interactivos para el curso de programación con el cual debe poder agregar notas de alumnos, quitar, consultar notas por alumno, mostrar todas las notas del curso o sacar el promedio. Para esto debe manejar dos listas (una de los nombres y otra de las notas). Programe las siguientes funciones:

insertarNota(alumno, nota): Inserta el nombre del alumno con su respectiva nota.

quitarNota(nombre): Quita la nota y al alumno, retornando que nota fue retirada

consultarNota(nombre): Entrega la nota correspondiente al alumno.

mostrarNotas(): Imprime en pantalla todas las notas del curso

promedio(): Entrega el promedio del curso

Las listas son del tipo global

Luego cree un programa interactivo de sistema de notas que se comporte de la siguiente manera:

```
Sistema de notas
Opciones: 1-Agregar nota, 2-Quitar nota, 3-Consultar nota, 4-Mostrar notas, 5-Promedio
Eliga que opcion quiere ejecutar: 1
Inserte nombre del alumno; Pepe
Ingrese nota del alumno: 50
Opciones: 1-Agregar nota, 2-Quitar nota, 3-Consultar nota, 4-Mostrar notas, 5-Promedio
Eliga que opcion quiere ejecutar: 1
Inserte nombre del alumno: Juan
Ingrese nota del alumno: 30
Opciones: 1-Agregar nota, 2-Quitar nota, 3-Consultar nota, 4-Mostrar notas, 5-Promedio
Eliga que opcion quiere ejecutar: 4
Alumno: Pepe Nota: 50.0
Alumno: Juan Nota: 30.0
```

```
Opciones: 1-Agregar nota, 2-Quitar nota, 3-Consultar nota, 4-Mostrar notas, 5-Promedio
Eliga que opcion quiere ejecutar: 5
40.0
Opciones: 1-Agregar nota, 2-Quitar nota, 3-Consultar nota, 4-Mostrar notas, 5-Promedio

Eliga que opcion quiere ejecutar
```

```
# insertarNota: str float -> None
# Inserta la nota de alumno en las listas correspondientes
# Ejemplo: insertarNota("Juan", 50) debe dejar l_nombres = ["Juan"] y l_notas = [50]

def insertarNota(alumno, nota):

  global l_nombres, l_notas

  l_nombres.append(alumno)
  l_notas.append(nota)


# Test
l_nombres = []
l_notas = []
assert l_nombres == [] and l_notas == []

insertarNota("Juan", 50)
assert l_nombres == ["Juan"] and l_notas == [50]

insertarNota("Pedro", 40)
assert l_nombres == ["Juan", "Pedro"] and l_notas == [50, 40]


# quitarNota: str -> float
# Quita la nota del alumno de las listas y retorna esta
# Ejemplo: Si tenemos l_nombres = ["Juan"] y l_notas = [50] y hacemos quitarNota("Juan")
# Nos debe dejar l_notas y l_nombres vacio

def quitarNota(nombre):

  global l_nombres, l_notas

  nota = None
  largo_lista = len(l_nombres)
  for i in range(largo_lista):

    if l_nombres[i] == nombre:

      l_nombres.remove(nombre)
      nota = l_notas.pop(i)
```

```
  return nota

# Test
assert l_nombres == ["Juan", "Pedro"] and l_notas == [50, 40]

quitarNota("Pedro")
assert l_nombres == ["Juan"] and l_notas == [50]

quitarNota("Diego")
assert l_nombres == ["Juan"] and l_notas == [50]

# consultarNota: str -> float / str
# Consulta la nota del alumno en la lista
# Ejemplo: consultarNota("Juan") debe entregar un 50 si l_notas = [50] y l_nombres = ["Juan"]
def consultarNota(nombre):

  global l_nombres, l_notas

  largo_lista = len(l_nombres)
  for i in range(largo_lista):

    if l_nombres[i] == nombre:
      nota = l_notas[i]
      return nota

  return "El nombre no existe"

# Test
assert consultarNota("Juan") == 50
assert consultarNota("Diego") == "El nombre no existe"


# mostrarNotas: None -> None
# Imprime en pantalla las notas de los alumnos
def mostrarNotas():

  largo_lista = len(l_nombres)
  for i in range(largo_lista):
    print("Alumno: " + l_nombres[i] + " Nota: " + str(l_notas[i]))

# promedio: None -> float
# Entrega el promedio de notas de los alumnos
# Ejemplo si estan Juan, Pedro y Diego con notas 50, 60 y 40 respectivamente
# Debe entregar 50.0
def promedio():

  suma_notas = sum(l_notas)
  cantidad_notas = len(l_notas)
  return suma_notas/cantidad_notas
```

```
#Test
insertarNota("Pedro", 60)
insertarNota("Diego", 40)
assert promedio() == 50
```

```
print("Sistema de notas")
l_notas = []
l_nombres = []

while True:
  print("Opciones: 1-Agregar nota, 2-Quitar nota, 3-Consultar nota, 4-Mostrar notas, 5-Promed


  input1 = input("Eliga que opcion quiere ejecutar ")
  if input1 == "1":
    nombre = input("Inserte nombre del alumno ")
    nota = float(input("Ingrese nota del alumno "))
    insertarNota(nombre, nota)

  elif input1 == "2":

    nombre = input("Ingrese nombre del alumno ")
    nota = quitarNota(nombre)
    print("La nota eliminada fue: ", nota)

  elif input1 == "3":

    nombre = input("Ingrese nombre del alumno ")
    consultarNota(nombre)

  elif input1 == "4":

    mostrarNotas()

  elif input1 == "5":

    print(promedio())

  else:
    break
```

P3) Un palíndromo, es una palabra o frase que se lee igual en un sentido que en otro. Cree una función esPalindromo que verifique si un string es un palindromo o no. Para esto no puede crear un

nuevo string, debe manejar dos variables, una que lea desde el inicio y otra desde el final.

```
# esPalindromo : str -> bool
# Revisa si un string es un palindromo o no
# Ejemplo: esPalindromo("uwu") retorna True, esPalindromo("ojo") retorna False
def esPalindromo(string):

  inicial = 0
  final = len(string) - 1
  while final >=0 and inicial <= len(string):

    if string[inicial] != string[final]:

      return False

    inicial += 1
    final -= 1

  return True


# Test

assert esPalindromo("uwu")
assert not esPalindromo("Conejo")
assert not esPalindromo("programacion")
assert esPalindromo("radar")
assert esPalindromo("ojo")
assert esPalindromo("sometemos")
assert not esPalindromo("hola")
```