Parcialito 03

Total de puntos 9/10

Apellido *

La Grotta

Nombre *

Thiago

✓ Cuando se trabaja con listas paralelas se debe utilizar * 1/1

- ○ solamente listas del mismo tipo
- ◉ el índice como forma de relacionarlas ✓
- ○ las dos anteriores son correctas
- ○ ninguna es correcta

✓ Cuando se quiere ordenar listas paralelas: * 1/1

- ◉ Se deben swappear todas las listas ✓
- ○ Sólo se debe hacer swap de la lista que contiene el criterio de ordenamiento
- ○ Las listas paralelas no pueden ordenarse
- ○ Ninguna es correcta

✗ Se desea hacer ordenamiento por doble criterio: apellido y nombre. Determine qué afirmación es correcta *0/1

- ○ Sólo se debe evaluar el nombre si los apellidos son coincidentes
- ◉ Se debe hacer el ordenamiento por apellido y luego hacer otro ordenamiento por nombre ✗
- ○ Sólo se debe ordenar por apellido, los nombres quedarán ordenados
- ○ No se puede hacer ordenamiento por doble criterio en listas paralelas

Respuesta correcta

- ◉ Sólo se debe evaluar el nombre si los apellidos son coincidentes

✓ Marcar la opción correcta para completar el siguiente código: * 1/1

```
nombres= ["Ana", "Juan", "Jose", "Sol", "Mario", "Ramiro", "Zoe", "Ana"]
edades = [  34,    45,     24,    65,     31,       55,      38,    23]
#Mostrar los datos de Sol
print(.........)
print(.........)
```

- ◉ print(nombres[3]) - print(edades[3]) ✓
- ○ print(nombres) - print(edades)
- ○ print(nombres[4]) - print(edades[4])
- ○ ninguna de las anteriores

✓ ¿Cuál es el nombre para llamar a una función dentro de la misma función? *1/1

- ○ subfunción
- ◉ recursión ✓
- ○ bucle infinito
- ○ inner call

✓ Marque la opción correcta según el siguiente código: * 1/1

```
nombres= ["Ana", "Juan", "Jose", "Sol", "Mario", "Ramiro", "Zoe", "Ana"]

for i in range(len(nombres)-1):
    for j in range(i+1, len(nombres)):
        if nombres[i] < nombres[j]:
            auxn = nombres[i]
            nombres[i] = nombres[j]
            nombres[j] = auxn

print(nombres)
```

- ○ Ordenará de la A-Z
- ◉ Ordenará de la Z-A ✓
- ○ La iteración de i es incorrecta
- ○ La iteración de j es incorrecta
- ○ La condición criterio es incorrecta

✓ Para funcionar correctamente el ordenamiento por burbujeo : * 1/1

- ○ El tipo de dato a ordenar debe ser entero
- ◉ Necesita una variable auxiliar para intercambiar valores ✓
- ○ Necesita solamente un ciclo for para recorrer lo que quiero ordenar

✓ Marcar el código correcto a completar en la línea 7 para ordenar la lista de manera descendente: *1/1

```
1
2   lista = ["Juan","Beto", "Pedro"]
3
4   #Ordenamiento de Burbujeo
5   for i in range(len(lista)-1):
6       for j in range(i+1,len(lista)):
7
8               aux = lista[i]
9               lista[i] = lista [j]
10              lista[j] = aux
11
12  for i in range(len(lista)):
13      print(lista[i])
14
```

- ○ if lista[i] > lista[j]:
- ◉ if lista[i] < lista[j]: ✓
- ○ ninguna de las anteriores

✓ Marcar el código correcto a completar en la línea 21 para ordenar la lista de manera ascendente: *1/1

```
11  contactos = [ ["Ana", 23,[123,345]] ,
12                ["Juan", 45, [456,567]] ,
13                ["Sol", 34, [678,987]] ,
14                ["Luis", 23,[356]] ,
15                ["Roberto", 46, [321]]
16               ]
17
18  #Ordenamos por edad
19  for i in range(len(contactos)-1):
20      for j in range(i+1, len(contactos)):
21
22              lista_aux = contactos[i]
23              contactos[i] = contactos[j]
24              contactos[j] = lista_aux
25
26  print("Ordenados")
27  mostrar()
```

- ○ if contactos[i][0] > contactos[j][0]:
- ◉ if contactos[i][1] > contactos[j][1]: ✓
- ○ ninguna de las anteriores

✓ ¿Cuál será la salida del siguiente programa?: * 1/1

```
lista = [1,2,3,4,5]

for i in range(len(lista)):
    print(i)
```

- ○ 1 2 3 4 5
- ◉ 0 1 2 3 4 ✓
- ○ ninguna de las anteriores falta el corchete