Práctica Inicial de Tuplas en Python

1. Objetivos

a. Comprender y discutir en grupo la lógica de cada reto.
b. Fomentar la robustez de las funciones implementadas con Python.
c. Comprender las separaciones de las responsabilidades en las funciones.
d. Crear algoritmos usando tuplas y otras estructuras según se requiera.
e. Documentar la solución.
f. Realizar código limpio y eliminar olores de software.
g. Cuide la correcta arquitectura el archivo a entregar, respete sus secciones.
h. Recuerde que por cada **def** debe tener al menos un **return** y su **respectiva documentación** en el lugar exacto enseñado por estrategia para la ayuda al programador.

2. Introducción

Trabaje en un solo archivo, llame secuencialmente todos los retos desde el programa principal mostrando: Nombre del reto, entrada y salida (para cada escenario allí mostrado).
Siga la siguiente estrategia:

1. Comprenda el problema y determine la estrategia de programación.
2. Programe únicamente las funciones de proceso para todos los retos.
3. Separando las funciones por responsabilidad, valide los retos según especifica cada reto.

3. Por Hacer

Reto 1. Desglose

. Hacer una función ***desglose*** que reciba un monto de dinero como un número entero y lo reparta en diferentes denominaciones. Se asumen las siguientes denominaciones: 1, 5, 10, 20, 50, 100, donde la denominación 1 corresponde a una moneda y el resto corresponden a billetes. Debe retornarse la menor cantidad posible de cada denominación. La función debe brindar resultados como los que se muestran a continuación:

```
>>> desglose(1427)
(14, 0, 1, 0, 1, 2)
```

Puede notarse que 1427 se descompone en 14 billetes de 100, 0 billetes de 50, 1 billete de 20, 0 billetes de 10, 1 billete de 5 y 2 monedas de 1.

Reto 2. Cuente palabras.

Haga la función **contarPalabras** que reciba dos tuplas: una donde cada elemento es una palabra y otra donde cada elemento es una frase. En la función debe imprimir cada palabra de la tupla de palabras junto con la cantidad de veces que aparece dicha palabra en la tupla de las frases.

Ejemplo:
>>> **contarPalabras** ((“calor”, “ayer”, “el”, “mañana”), (“ayer hizo bastante calor”, “en el laboratorio hace calor”))
Cantidad de veces que aparece cada palabra en las frases:
calor: 2
ayer: 1
el: 1
mañana: 0

Reto 3. Pares amigables.

En matemáticas un par de números *m* y *n* es llamado par amigable (o números amistosos), si la suma de todos los divisores de *m* (excluyendo a *m*) es igual al número *n*, y la suma de todos los divisores del número *n* (excluyendo a *n*) es igual a *m* (donde $m \neq n$).

Ejemplo: los números 220 y 284 son un par amigable. La explicación es la siguiente:
Los únicos números que dividen de forma exacta a 220 son 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 y 110, y la suma de ellos es: 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284
Los únicos números que dividen de forma exacta a 284 son 1, 2, 4, 71 y 142 , y la suma de ellos es: 1 + 2 + 4 + 71 + 142 = 220

Haga la función **esParAmigable** que reciba una tupla con dos números enteros (> 0) y retorne el valor booleano True si cumplen esa propiedad, de lo contrario retorne False. También debe retornar False si los números recibidos son iguales.
Valide que el valor recibido sea una tupla y que contenga exactamente dos enteros.

Ejemplos del funcionamiento:
>>> esParAmigable((220, 284))
True
>>> esParAmigable ((15, 18))
False
>>> esParAmigable ((1210, 1184))
True
>>> esParAmigable ((890, 890))
False

Reto 4. Mostrar Cercano

Desarrolle la función **mostrarCercano** que reciba cinco números enteros dentro de una tupla, mayores a cero y diferentes entre sí. Retorna cuál de los últimos cuatro números es el más cercano al primero. Ejemplo:
>>> mostrarCercano((8, 11, 4, 10, 100))
10
En caso de que existan dos números cercanos retorne el menor. Ejemplo:
>>> mostrarCercano ((8, 11, 10, 4, 6))
6

Haga la validación de las restricciones y cuando encuentre algún error retorne el mensaje respectivo. Las entradas deben ser: enteros, mayores a cero, diferentes entre sí.

Ejemplos del funcionamiento:
>>> mostrarCercano ([8, 11, 4, 8,10])
>>>Debe ingresar una tupla estrictamente.
>>> mostrarCercano ((8, 11, 4, 10))
>>>Debe ingresar exactamente 5 valores enteros.
>>> mostrarCercano((8, 11, "4", -10, 6))
>>>Las entradas deben ser enteros
>>> mostrarCercano ((8, 11, 4, -10, 6))
>>>Las entradas deben ser mayores a cero
>>> mostrarCercano ((8, 11, 4, 10, 4))
>>>Las entradas deben ser diferentes entre sí

Reto 5. Enfrentar.

Haga la función **enfrentar** que reciba una lista de tuplas, donde cada tupla tiene dos elementos: el código de un equipo y su nombre, y retorne una lista con las posibles combinaciones de pares entre sus elementos considerando únicamente los valores del código del equipo. Cada par formado va a ser una tupla de la lista retornada. Valide estas restricciones: que el dato recibido sea una lista y que esa lista tenga dos tuplas o más, de lo contrario realimente según corresponda.

Ejemplos del funcionamiento:
>>> **enfrentar** ([("CRC", "Costa Rica"), ("USA", "Estados Unidos"), ("MEX", "México"), ("PAN", "Panamá")])
[("CRC", "USA"), ("CRC", "MEX"), ("CRC", "PAN"), ("USA", "MEX"), ("USA", "PAN"), ("MEX", "PAN")]

>>> **enfrentar** (**(** ("CRC", "Costa Rica"), ("USA", "Estados Unidos") **)**)
Debe ingresar una lista para enfrentar.

>>> **enfrentar** ([("CRC", "Costa Rica")])
Debe ingresar más de un valor para poder enfrentar.

>>> **enfrentar** (**[** ("CRC", "Costa Rica"), **[**"USA", "Estados Unidos"**]** **]**)
Todos los elementos de la lista, deben ser tuplas.

Reto 6. Obtener diferencia.

Haga la función ***obtenerDiferencia*** que retorne la diferencia simétrica de dos números enteros (> 0). La diferencia simétrica es un número con los dígitos que pertenecen solamente a uno de los números recibidos, pero no a ambos. Los números recibidos pueden tener dígitos repetidos pero el resultado no. En caso de no haber diferencia simétrica retornar False. Recibe una tupla con los dos números.

Ejemplos del funcionamiento:

```
>>> obtenerDiferencia((1265, 42))
1654
>>> obtenerDiferencia ((8587, 71457))
814
>>> obtenerDiferencia ((542, 254))
False
>>> obtenerDiferencia ((2984, 48298))
False
>>> obtenerDiferencia ((12345, 67890))
1234567890
>>> obtenerDiferencia ((1, 1010))
0
```

```
>>> obtenerDiferencia ((-21, 1010))
Ambos valores deben ser mayores a 0.
>>> obtenerDiferencia ((1, 1010.6))
Ambos valores deben ser enteros.
>>> obtenerDiferencia ([12345, 67890])
Debe recibir una tupla para analizar los
valores.
```

Reto 7. "Ruta por la Costa del Caribe".

Ayude a los encargados de la Ruta a determinar quiénes fueron los excelentes atletas que lograron la meta, para ello se fue por etapas. Reciba una lista donde cada elemento es otra lista con la información de cada etapa. En cada etapa se tiene una tupla por cada corredor: (numero_corredor, tiempo_etapa), numero_corredor identifica a cada corredor con un entero cualquiera de 1 en adelante, tiempo_etapa es el tiempo que duró el corredor en completar la etapa, un entero de máximo 5 dígitos, siendo estos la totalidad de segundos de esfuerzo.
Desarrolle la función ***determinarPosicion*** que reciba una lista con n etapas estructuradas como se indicó anteriormente y retorne una lista de tuplas con las posiciones finales de la vuelta en orden ascendente por tiempo tal como muestra el ejemplo de funcionamiento.

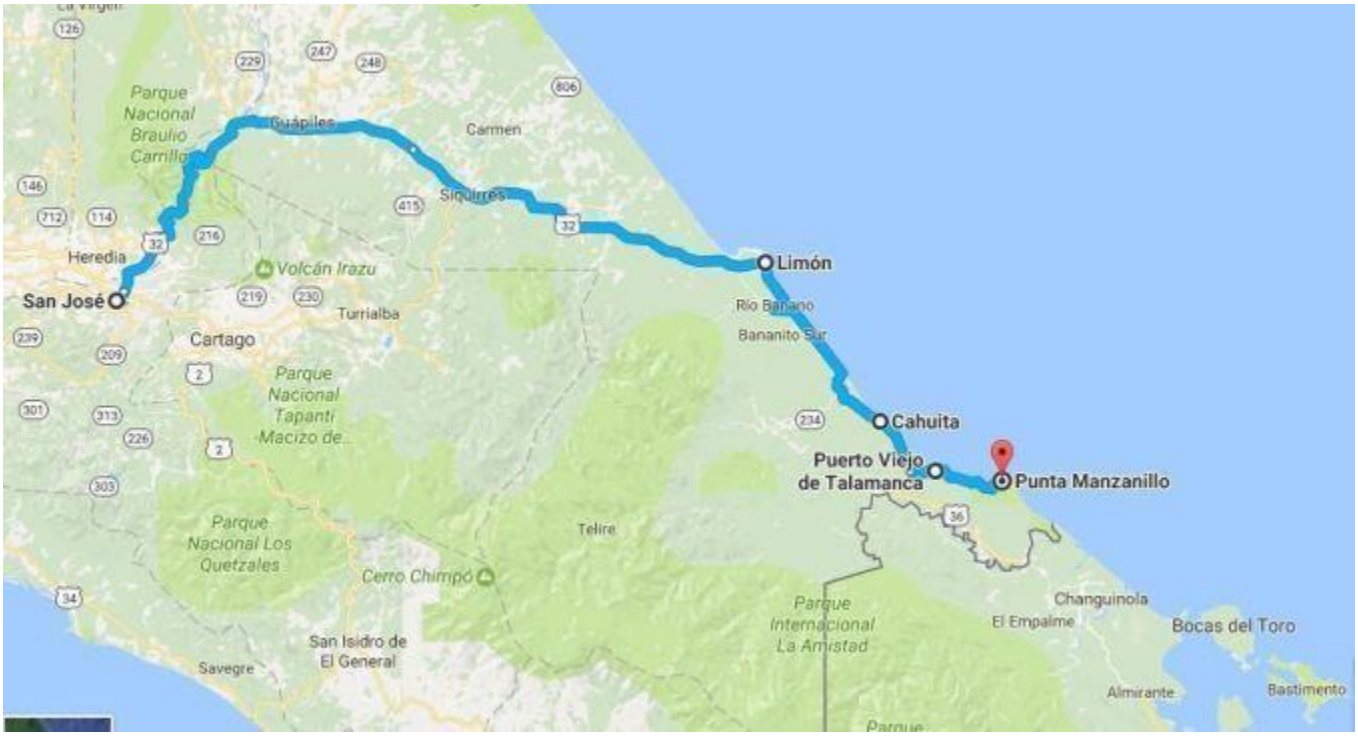
>>> ***determinarPosicion*** **([[**(100, ~~11512~~), (2, 15050), (130, 12320), (101, *12050*), (125, 11501), (115, 20000) **]**,

[(130, 14050), (100, ~~15050~~), (125, 14515), (2, 23000), (101, *15000*) **]**,

[(100, ~~4520~~), (101, *4720*), (130, 4520), (125, 4600) **]])**

[(125, 30616), (130, 30890), (100, ~~31082~~), (101, *31770*) **]**

El primer elemento de la lista resultante (índice 0) corresponde al corredor con el primer puesto de la vuelta, el segundo elemento (índice 1) corresponde al corredor con el segundo puesto y así sucesivamente.
Los corredores que no participan en todas las etapas son excluidos de la clasificación. Vea los casos de los ciclistas con los números 2 y 115.



Extraído de: Costa Rica – Ruta por la Costa del Caribe, https://elviajenotermina.com/costa-rica-caribe/