correspondientes

Desafio Stark

Industrias Stark es principalmente una empresa de defensa que desarrolla y fabrica armas avanzadas y tecnologías militares.

Recientemente ha decidido ampliar su departamento de IT y para acceder a las entrevistas es necesario completar el siguiente desafío, el cual estará dividido en etapas. Cada semana recibiremos un nuevo pedido de parte de la empresa.

La empresa compartió con todos los participantes cierta información confidencial de un grupo de superhéroes. Y semana a semana enviará una lista con los nuevos requerimientos. Quien supere todas las etapas accedera a una entrevista con el director para de la compañía.

Set de datos

La información a ser analizada se encuentra en el archivo data_stark.py, por tratarse de una lista, bastará con incluir dicho archivo en el proyecto de la siguiente manera:

```
from data_stark import lista_personajes
```

Formato de los datos recibidos

```
lista_heroes = [
    {
```

correspondientes


```
        "nombre": "Howard the Duck",
        "identidad": "Howard (Last name unrevealed)",
        "empresa": "Marvel Comics",
        "altura": "79.349999999999994",
        "peso": "18.449999999999999",
        "genero": "M",
        "color_ojos": "Brown",
        "color_pelo": "Yellow",
        "fuerza": "2",
        "inteligencia": "average"
    },
    {
        "nombre": "Rocket Raccoon",
        "identidad": "Rocket Raccoon",
        "empresa": "Marvel Comics",
        "altura": "122.77",
        "peso": "25.73",
        "genero": "M",
        "color_ojos": "Brown",
        "color_pelo": "Brown",
        "fuerza": "5",
        "inteligencia": "average"
    }
]
```

correspondientes

Desafío #02: (con biblioteca propia: stark_biblioteca)

En base a lo resuelto en el desafío #00, deberás mejorar la calidad de tus funciones. Es por ello que te proponemos lo siguiente:

IMPORTANTE: ***Para todas y cada una de las funciones creadas, documentarlas escribiendo que es lo que hacen, que son los parámetros que reciben (si es una lista, un string, etc y que contendrá) y que es lo que retorna la función!***

0. Crear la función '*stark_normalizar_datos*' la cual recibirá por parámetro la lista de héroes. La función deberá:
    - Recorrer la lista y convertir al tipo de dato correcto las keys (solo con las keys que representan datos numéricos)
    - Validar primero que el tipo de dato no sea del tipo al cual será casteado. Por ejemplo si una key debería ser entero (ejemplo edad) verificar antes que no se encuentre ya en ese tipo de dato.
    - Si al menos un dato fue modificado, la función deberá imprimir como mensaje 'Datos normalizados', caso contrario no imprimirá nada.
    - Validar que la lista de héroes no esté vacía para realizar sus acciones, caso contrario imprimirá el mensaje: "Error: Lista de héroes vacía"

1.1. Crear la función *'obtener_nombre'* la cual recibirá por parámetro un diccionario el cual representara a un héroe y devolverá un string el cual contenga su nombre formateado de la siguiente manera:

Nombre: Howard the Duck

1.2. Crear la función *'imprimir_dato'* la cual recibirá por parámetro un string y deberá imprimirlo en la consola. La función no tendrá retorno.

1.3. Crear la función *'stark_imprimir_nombres_heroes'* la cual recibirá por parámetro la lista de héroes y deberá imprimirla en la consola. Reutilizar las funciones hechas en los puntos 1.1 y 1.2. Validar que la lista no esté vacía

para realizar sus acciones, caso contrario no hará nada y retornara -1.

Con este se resuelve el Ej 1 del desafío #00

2. Crear la función *'obtener_nombre_y_dato'* la misma recibirá por parámetro un diccionario el cual representara a un héroe y una key (string) la cual representará el dato que se desea obtener.

    - La función deberá devolver un string el cual contenga el nombre y dato (key) del héroe a imprimir. El dato puede ser 'altura', 'fuerza', 'peso' O CUALQUIER OTRO DATO.

    - El string resultante debe estar formateado de la siguiente manera: (suponiendo que la key es fuerza)
      Nombre: Venom | fuerza: 500

3. Crear la función *'stark_imprimir_nombres_alturas'* la cual recibirá por parámetro la lista de héroes, la cual deberá iterar e imprimir sus nombres y alturas USANDO la función creada en el punto 2. Validar que la lista de héroes no esté vacía para realizar sus acciones, caso contrario no hará nada y retornara -1.
   Con este se resuelve el Ej 2 del desafío #00

4.1. Crear la función *'calcular_max'* la cual recibirá por parámetro la lista de héroes y una key (string) la cual representará el dato que deberá ser evaluado a efectos de determinar cuál es el máximo de la lista. Por ejemplo la función deberá poder calcular: el peso, la altura o fuerza máximas y retornar el héroe que tenga el dato más alto.

Ejemplo de llamada:
calcular_max(lista, 'edad')

4.2. Crear la función *'calcular_min'* la cual recibirá por parámetro la lista de héroes y una key (string) la cual representará el dato que deberá ser evaluado a efectos de determinar cuál es el mínimo de la lista. Por ejemplo la función deberá poder calcular: el peso, la altura o fuerza máximas y retornar el héroe que tenga el dato más bajo.

correspondientes

Ejemplo de llamada:
calcular_min(lista, 'edad')

4.3. Crear la funcion *'calcular_max_min_dato'* la cual recibira tres parámetros:

- La lista de héroes
- El tipo de cálculo a realizar: es un valor string que puede tomar los valores *'maximo'* o *'minimo'*
- Un string que representa la key del dato a calcular, por ejemplo: '*altura*', '*peso*', '*edad*', etc.

La función deberá retornar el héroe que cumpla con la condición pedida. Reutilizar las funciones hechas en los puntos 4.1 y 4.2

Ejemplo de llamada:
calcular_max_min_dato(lista, "maximo", "edad")

4.4. Crear la función ***'stark_calcular_imprimir_heroe'*** la cual recibirá tres parámetros:

- La lista de héroes
- El tipo de cálculo a realizar: es un valor string que puede tomar los valores *'maximo'* o *'minimo'*
- Un string que representa la key del dato a calcular, por ejemplo: '*altura*', '*peso*', '*edad*', etc.

Con este se resuelve el Ej 3, Ej 4, Ej 6 y Ej 7 del desafío #00

La función deberá obtener el héroe que cumpla dichas condiciones, imprimir su nombre y el valor calculado. Reutilizar las funciones de los puntos:
punto 1.2, punto: 2 y punto 4.3

Validar que la lista de héroes no esté vacía para realizar sus acciones, caso contrario no hará nada y retornara -1.

Ejemplo de llamada:

stark_calcular_imprimir_heroe (lista, "maximo", "edad")

Ejemplo de salida:

Mayor altura: Nombre: Howard the Duck | altura: 79.34

correspondientes

5.1. Crear la función *'sumar_dato_heroe'* la cual recibirá como parámetro una lista de héroes y un string que representara el dato/key de los héroes que se requiere sumar. Validar que cada héroe sea tipo diccionario y que no sea un diccionario vacío antes de hacer la suma. La función deberá retorna la suma de todos los datos según la key pasada por parámetro

5.2. Crear la función *'dividir'* la cual recibirá como parámetro dos números (dividendo y divisor). Se debe verificar si el divisor es 0, en caso de serlo, retornar 0, caso contrario realizar la división entre los parámetros y retornar el resultado

5.3. Crear la función *'calcular_promedio'* la cual recibirá como parámetro una lista de héroes y un string que representa el dato del héroe que se requiere calcular el promedio. La función debe retornar el promedio del dato pasado por parámetro

IMPORTANTE: hacer uso de las las funciones creadas en los puntos 5.1 y 5.2

5.4. Crear la función ***'stark_calcular_imprimir_promedio_altura'*** la cual recibirá una lista de héroes y utilizando la función del punto 5.3 calcula y mostrará la altura promedio. Validar que la lista de héroes no esté vacía para realizar sus acciones, caso contrario no hará nada y retornara -1.

IMPORTANTE: hacer uso de las las funciones creadas en los puntos 1.2, 5.1 y 5.3

Con este se resuelve el Ej 5 del desafío #00

6.1. Crear la función "*imprimir_menu*" que imprima el menú de opciones por pantalla, el cual permite utilizar toda la funcionalidad ya programada. Se deberá reutilizar la función antes creada encargada de imprimir un string (1.2)

6.2. Crear la función "*validar_entero*" la cual recibirá por parámetro un string de número el cual deberá verificar que sea sea un string conformado únicamente por dígitos. Retornara True en caso de serlo, False caso contrario

6.3. Crear la función *'stark_menu_principal'* la cual se encargará de imprimir el menú de opciones y le pedirá al usuario que ingrese el número de una de las opciones elegidas y deberá validarlo. En caso de ser correcto dicho número, lo retornara casteado a entero, caso contrario retorna -1. Reutilizar las funciones del ejercicio 6.1 y 6.2

7. Crear la función *'stark_marvel_app'* la cual recibirá por parámetro la lista de héroes y se encargará de la ejecución principal de nuestro programa.
   Utilizar if/elif o match según prefiera (match solo para los que cuentan con

python 3.10+). Debe informar por consola en caso de seleccionar una opción incorrecta y volver a pedir el dato al usuario. Reutilizar las funciones con prefijo 'stark_' donde crea correspondiente.