EJERCICIOS PRÁCTICA 7

Ejercicio 3 (7.1)

3. Respon les següents qüestions sobre la traducció de l'**exercici 1** (traducció JSON → XML).

a. Com has manejat el valor *null* en l'element age de l'Anna en la traducció a XML? És aquesta la millor manera de representar la falta d'informació? Proposa totes les alternatives possibles.

Para definir un valor null en el elemento "age" de Anna utilizé una etiqueta vacía.

En mi opinión, la manera que emplee yo es la mejor ya que la etiqueta no se suprime ni se deja vacía, lo cual equivale a una mejor representación y comprensión del código pero aún así existen otras maneras:

- <age></age> → no poner nada dentro
- suprimir la etiqueta

b. Què haurem de tenir en compte quan tenim elements repetits com ara les mascotes o els amics? S'ha mantingut la consistència en la traducció?

Hemos de tener en cuenta que en XML no se puede repetir etiquetas.

Al realizar la traducción tuve que crear nuevas etiquetas para poder definir y representar todos los valores que eran repetidos.

Ejercicio 4 (7.1)

4. Respon les següents qüestions sobre la traducció de l'**exercici 2** (traducció XML→ JSON).

   a. Explica què s'ha convertit en objectes, i què en arrays i per què has pres aquestes decisions.

   Todo el contenido de la etiqueta <devices> se transformó en un array porque su contenido es una lista de diferentes dispositivos (objetos).

   Todo el contenido de cada etiqueta <device> se transformaron en objetos porque el contenido es información referente a un elemento concreto.

   Las opciones de la etiqueta <slogan>, la lista de <características> y los atributos y valores de la etiqueta <price> también se transformaron en objetos porque es información referente a un único elemento.

   b. Explica què has fet per tal de mantenir junta la informació del preu amb el tipus de moneda pagada. Com has transformat, en aquest cas, els atributs de l'XML a JSON i per què?

   Tanto los atributos como el valor de <price> los convertí en un objeto para poder mantenerlos juntos y porque es la manera más simple y clara para agrupar información de un elemento en JSON.

   c. Hi ha alguna etiqueta en l'XML que no s'ha traduït directament a JSON? Creus que això significa que s'ha perdut informació?

   No hay una traducción directa y exacta de XML a JSON, varias cosas cambian, por ejemplo la manera para representar los atributos y valores null. Sin embargo, esto no quiere decir que la información se pierda ya que JSON tiene otra manera para almacenar y estructurar la información.

   d. Com has gestionat els caràcters especials com les cometes dobles en la traducció? Com afecta això la llegibilitat del JSON?

   Los únicos carácteres especiales que tuve que gestionar fueron las contrabarras de la clave: "Internal Info" y lo que hice fue emplear un scape char → contrabarra (\).

   Los carácteres especiales pueden romper el código JSON y la única solución que hay es el empleo de los scape

chars. Sin embargo esto puede derivar a confusión a la hora de leer e interpretar el código a usuarios menos expertos.

e. Explica com has tractat els elements sense informació o amb dades opcionals. Has optat per deixar el camp buit, per fer servir el valor *null* o per ometre el camp? Explica quina creus que és la millor decisió i per qué.

A los elementos sin información les asigné el valor null.

En mi opinión la mejor manera para tratar este tipo de datos es asignar el valor null porque no se omite el atributo pero tampoco se deja vacío, lo cual conlleva a una mejor representación y comprensión del código.

Aunque creo que depende del caso.

f. Quina estructura de dades has utilitzat per representar les característiques de "P50 Pocket"? Explica si hi ha alternatives i per què has pres aquesta decisió.

Para representar las características de "P50 Pocket" utilizé un objeto JSON porque creo que es la opción más clara y simple para estructurar y representar información de un elemento lista.

Aunque también existe otra alternativa:

g. Si el JSON resultant no té el camp "items_count", creus que s'ha perdut informació? Creus que és útil tenir aquesta informació en un camp?

En mi opinión el campo "items_count" no es necesario en este código porque a simple vista se puede ver la cantidad de objetos que hay en la lista por lo tanto no se perdería información.
Este tipo de información es útil en casos por ejemplo donde haigan muchos objetos y en ese caso el valor de "items_count" te puede ayudar.

Ejercicio 6 (7.2)

6. **Proposa un pseudocodi** d'una funció per obtenir les dades que es demanen partint del JSON que acabes de generar. Fes servir la capçalera que se suggereix i recorda que l'objectiu d'aquest exercici és trobar l'estructura òptima del JSON. Per aquesta raó, la majoria d'aquests exercicis s'han de resoldre accedint a les de manera quasi directa a les dades.

   Considera que els **índexs de les llistes comencen en 0**. Tampoc **no cal tenir en compte els possibles errors**, com trobar-se llistes buides o elements inexistents.

   A tall d'exemple, considera una funció que retorni el nom del pokémon. Una possible solució seria

```
fun getPokemonName(pokemon) {
    return pokemon["name"]
}
```

Un exemple diferent pot ser una funció que retorni **el nom del primer moviment** del primer pokémon d'una llista de pokémons. La solució:

```
fun getMovimentPrimerPokemon(pokemonsList) {
    primerPokemon = pokemonsList[0]
    primerMoviment = primerPokemon["moviments"][0]
    return primerMoviment["nom"]
}
// solució alternativa:
// return pokemonsList[0]["moviments"][0]["nom"]
```

   a. Implementa una funció que retorna la **unitat** de mesura l'altura del pokémon. Si el pokémon mesura 0.8 m, la funció ha de retornar "m". Recorda que no cal processar les dades, sinó que és millor tenir-les ben estructurades.

```
fun getUnitatMesuraAltura(pokemon)
```

   b. La funció ha de retornar un booleà que indiqui si el segon moviment de la llista de moviments del pokémon **és de contacte** o no.

```
fun isSegonMovimentDeContacte(pokemon)
```

   c. Una funció que retorni la **suma de totes les estadístiques** del pokémon (velocitat, fortalesa, precisió, resistència)

```
fun getSumaEstadistiques(pokemon)
```

   d. La funció ha de retornar la **mitjana de totes les estadístiques** del pokémon.

```
fun getMitjanaEstadistiques(pokemon)
```

e. Donada una llista de 3 pokémons, la funció ha de retornar **la suma dels pesos** d'aquests pokémons.

```
fun getPes(llista3Pokemon)
```

f. Donat un pokémon i un nivell, la funció ha d'indicar si el nivell és igual o superior al nivell requerit per fer la **primera evolució** del pokémon.

Per exemple, si la funció rep nivell=4, i el nivell de la primera evolució és 3, la funció ha de retornar *true*.

```
fun isEvolucioPossible(pokemon, nivell)
```

g. Donada una llista de pokémons, la funció ha de retornar el pokémon amb la potència més alta. Pots fer una funció auxiliar que calculi la potència d'un pokémon que és la suma de les potències dels seus moviments (**opcional**)

```
fun getPotenciaMesAlta(pokemonsList)
```

```
//a.

fun getUnitatMesuraAltura(pokemon){
    pokemon_escogido = lista_pokemons["pokemon"]
    altura_pokemon = pokemon_escogido["altura"]
    return altura_pokemon["unidad"]
}

//b.

fun isSegonMovimentDeContacte(pokemon){
    pokemon_escogido = lista_pokemons["pokemon"]
    segundo_movimiento = pokemon_escogido["movimientos"][1]
    if segundo_movimiento["contacto"] == "Si":
        print("Si")
    else:
        print("No")
}

//c.

fun getSumaEstadistiques(pokemon){
    pokemon_escogido = lista_pokemons["pokemon"]
    estadisticas_pokemon = pokemon_escogido["estadísticas"]
    return estadisticas_pokemon["velocidad"] + estadisticas_pokemon["fortaleza"] + estadisticas_pokemon["precisión"] + estadisticas_pokemon["resistencia"]
}

//d.

fun getMitjanaEstadistiques(pokemon){
    pokemon_escogido = lista_pokemons["pokemon"]
    estadisticas_pokemon = pokemon_escogido["estadísticas"]
    return estadisticas_pokemon["velocidad"] + estadisticas_pokemon["fortaleza"] + estadisticas_pokemon["precisión"] + estadisticas_pokemon["resistencia"] / 4
}
```

```
//e.

fun getPes(llista3Pokemon){
    pokemon1 = lista_pokemons[llista3Pokemon[0]]
    peso_pokemon_1 = pokemon1["peso"]["peso_total"]

    pokemon2 = lista_pokemons[llista3Pokemon[1]]
    peso_pokemon_2 = pokemon2["peso"]["peso_total"]

    pokemon3 = lista_pokemons[llista3Pokemon[2]]
    peso_pokemon_3 = pokemon3["peso"]["peso_total"]

    return peso_pokemon_1 + peso_pokemon_2 + peso_pokemon_3
}

//f.

fun isEvolucioPossible(pokemon, nivell){
    pokemon_escogido = lista_pokemons["pokemon"]
    primera_evolucion = pokemon_escogido["evoluciones"][0]
    if nivell >= primera_evolucion["nivel"]:
        print("True")
    else:
        print("False")
}

//g.

fun getPotenciaMesAlta(pokemonsList){
    pokemon_escogido = lista_pokemons[pokemonsList[0]]
    movimiento1_pokemon1 = pokemon_escogido["movimientos"][0]
    movimiento2_pokemon1 = pokemon_escogido["movimientos"][1]
    suma_potencia_pokemon1 = movimiento1_pokemon1["potencia"] + movimiento2_pokemon1["potencia"]

    pokemon_escogido = lista_pokemons[pokemonsList[1]]
    movimiento1_pokemon2 = pokemon_escogido["movimientos"][0]
    movimiento2_pokemon2 = pokemon_escogido["movimientos"][1]
    suma_potencia_pokemon2 = movimiento1_pokemon2["potencia"] + movimiento2_pokemon2["potencia"]

    pokemon_escogido = lista_pokemons[pokemonsList[2]]
    movimiento1_pokemon3 = pokemon_escogido["movimientos"][0]
    movimiento2_pokemon3 = pokemon_escogido["movimientos"][1]
    suma_potencia_pokemon3 = movimiento1_pokemon3["potencia"] + movimiento2_pokemon3["potencia"]

    return max(suma_potencia_pokemon1, suma_potencia_pokemon2, suma_potencia_pokemon3)
}
```