Daniel Sanchez Tapia Arman Mohammed Akther Oriol González Farssac

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 'Anna en la traducció a XML? És aquesta la millor manera de representar la falta d'informació? Proposa totes les alternatives possibles
Hem optat per deixar l'etiqueta buida degut a que al llenguatge no existeix el valor "null" que hi ha a Json. Una altra manera de fer-ho podria ser creant un atribut buit.

b. Que haurem de tenir en compte quan tenim elements repetits com ara les mascotes o els amics? S'ha mantingut la consistència en la traducció?
En aquests casos on tenim array al document Json haurem de crear una etiqueta filla (la quantitat serà els espais de l'array) que tindrà el nom que té l'array però en singular. La traducció ha sigut consistent perquè no s'ha perdut informació posant tots els valors com a etiquetes dins d'una etiqueta propia.

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

a. Explica que s'ha convertit en objectes, i que en arrays i perquè has pres aquestes decisions.
El dispositiu s'han convertit en un objecte dins d'un array de dispositius, perquè és la manera més correcta d'organitzar l'estructura, i que s'entengui bé, de manera que puguem tenir infinits dispositius en una llista.

b. Explica que 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 d'XML a JSON i per què?
Hem fet que el preu sigui un objecte que contingui un valor, com per exemple 20, i una moneda $.

c. Hi ha alguna etiqueta en 'XML que no s'ha traduït directament a JSON? Creus que això significa que s'ha perdut informació?
No es perd la informació, però no és tan específic com es guarda la dada.

d. Com has gestionat els caràcters especials com les cometes dobles en la traducció? Com afecta això la llegibilitat del JSON?
En el cas de la " \ " la manera en la qual es traduïa és posant dos " \\ " en l'exemple C:\\Documents\\files\\matebook.json es pot veure

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 què.
Ho hem deixat amb el valor en blanc perquè “ometre deixarlo en blanc” s'hauria de referir a una secció de memòria que no conté RES. I NULL es refereix a una adreça de memòria NO VÀLIDA.

f. Quina estructura de dades has utilitzat per representar les característiques de ”P50 Pocket"? Explica si hi ha alternatives i perquè has pres aquesta decisió.
Dins de l’objecte hem fet un array de strings que es digués característiques, que contindria diferents strings amb una caracteristica cada un.

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?
Crec que no és important, ja que en tenir un array de “ítems”, ja ho podem saber.


```
{
  "pokemons": [
    {
      "id": 1,
      "nom": "Bulbasaur",
      "tipus": ["Planta", "Venen"],
      "pes": {"valor": 6.9, "unitat": "kg"},
      "altura": {"valor": 0.7, "unitat": "m"},
      "estadistiques": {
        "velocitat": 45,
        "fortalesa": 49,
        "precisio": 65,
        "resistencia": 65
      },
      "moviments": [
        {
          "nom": "Llavors",
          "descripcio": "Llança llavors a l'enemic",
          "potencia": 40,
          "contacte": true
        },
        {
          "nom": "Gruixuda",
          "descripcio": "Augmenta la defensa del pokémon",
          "potencia": 0,
```

```
            "contacte": false
          }
        ],
        "imatge": "bulbasaur.png",
        "evolucions": [
          {
            "id": 2,
            "nivell": 16
          },
          {
            "id": 3,
            "nivell": 32
          }
        ]
      },
      {
        "id": 2,
        "nom": "Ivysaur",
        "tipus": ["Planta", "Venen"],
        "pes": {"valor": 13.0, "unitat": "kg"},
        "altura": {"valor": 1.0, "unitat": "m"},
        "estadistiques": {
          "velocitat": 60,
          "fortalesa": 62,
          "precisio": 80,
          "resistencia": 80
        },
        "moviments": [
          {
            "nom": "Llavors",
            "descripcio": "Llança llavors a l'enemic",
            "potencia": 40,
            "contacte": true
          },
          {
            "nom": "Gruixuda",
            "descripcio": "Augmenta la defensa del pokémon",
            "potencia": 0,
            "contacte": false
          },
```

```
          {
            "nom": "Latigazo",
            "descripcio": "Golpeja l'enemic amb vinclaments forts",
            "potencia": 55,
            "contacte": true
          }
        ],
        "imatge": "ivysaur.png",
        "evolucions": [
          {
            "id": 3,
            "nivell": 32
          }
        ]
      }
    ]
}
```

6. Proposa un pseudocodi d'una funció per obtenir les dades que es demanen partint del JSON que acabes de generar. Fes servir la capcalera que se suggereix ¡ recorda que l'objectiu d'aquest exercici és trobar l'estructura óptima 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)
```

```
fun getUnitatMesuraAltura(pokemon){
    return pokemon["altura"]["unitat"]
}
```

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)
```

```
fun isSegonMovimentDeContacte(pokemon){
    return pokemon["moviments"][1]["contacte"]
}
```

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

```
fun getSumaEstadistiques(pokemon)
```

```
fun getSumaEstadistiques(pokemon){
        return (pokemon[“estadistiques”][“velocitat”] +
        pokemon[“estadistiques”][“fortalesa”] + pokemon[“estadistiques”][“precisió”] +
        pokemon[“estadistiques”][“resistència”])
}
```

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

```
fun getMitjanaEstadistiques(pokemon)
```

```
fun getMitjanaEstadistiques(pokemon){
        sumaEstadistiques = pokemon[“estadistiques”][“velocitat”] +
pokemon[“estadistiques”][“fortalesa”] + pokemon[“estadistiques”][“precisió”] +
pokemon[“estadistiques”][“resistència”]
        return sumaEstadistiques / 4
}
```

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

```
fun getPes(llista3Pokemon)
```

```
fun getPes(llista3Pokemon){
        return llista3Pokemon[“pokemon1”][“pes”][“valor”] +
        llista3Pokemon[“pokemon2”][“pes”][“valor”] + llista3Pokemon[“pokemon3”][“pes”]
}
```

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)
```

```
fun isEvolucioPossible(pokemon, nivell){
        boolean evolucionara = false;
        If (nivell>= pokemon[“evolucio”][0][“nivell”]){
        evolucionara = true;
        }
        return evolucionara;
}
```