ITICBCN
Institut TIC de Barcelona
Consorci d'Educació de Barcelona
Generalitat de Catalunya
Ajuntament de Barcelona

Pràctica 7: JSON **(part 2)**

Objectius

Els objectius d'aquesta part de la pràctica són reforçar els coneixements apresos en la pràctica 7.1. D'altra banda, els següents exercicis tenen com a objectius principals l'aprenentatge de l'estructuració òptima d'un JSON per emmagatzemar detalls específics sobre pokémons.

Mitjançant funcions d'accés a les dades, es busca reforçar la pràctica en la manipulació de documents JSON. A més, els exercicis finals permetran reflexionar sobre la qualitat de les decisions preses en el primer exercici d'aquesta part.

Lliuraments

L'entrega ha de contenir els exercicis d'aquesta part de la pràctica **juntament amb els de la part 7.1.**

Mitjançant GIT, has d'entregar els següents fitxers:

- Exercici 1 (7.1): un fitxer XML: *peterJones.xml*
- Exercici 2 (7.1): un fitxer JSON: *dediamarkt*.json
- Exercici 5 (7.2): un fitxer JSON: *pokemons.json*

Mitjançant l'entrega de documents o GIT:

- 1 únic document PDF amb els exercicis
  - Exercici 3 (7.1)
  - Exercici 4 (7.1)
  - Exercici 6 (7.2)

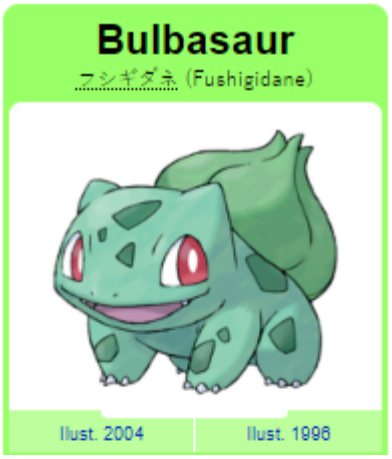
*Si el lliurament no compleix els requisits d'entrega, la nota serà un *no entregat*.

PART 2

L'objectiu de la segona part de la pràctica és ser capaç de dissenyar un document JSON que emmagatzemi unes certes dades. En els següents exercicis es demana generar una estructura JSON que contingui informació sobre pokémons.

De vegades, per dissenyar una estructura de dades cal conèixer les dades que estàs tractant. Per aquesta raó, si tens qualsevol dubte pots consultar la Wikidex (enciclopèdia Pokémon) que és on s'ha basat aquesta pràctica.

https://www.wikidex.net/wiki/Bulbasaur



Generación	Primera
Categoría	Semilla
Tipos	PLANTA VENENO
Habilidad	Espesura
Hab. oculta	Clorofila
Peso	6,9 kg
Altura	0,7 m
Grupos de huevo	Monstruo Planta
Sexo	♂ 87,5% ♀ 12,5%
Color	Verde
Figura	
Grito	

Ver más imágenes de Bulbasaur

Estadísticas del Pokéathlon

Velocidad	★★★	3
Fortaleza	★★★	3
Precisión	★★★★	4
Resistencia	★★★★	4
Salto	★★★	3
Total — 17	Media — 3.4	

5. Crea un JSON que et permeti emmagatzemar una **llista de pokémons**. Aquesta llista ha de contenir **2 pokémons**, com a mínim. Has de tenir en compte que cada pokémon ha de contenir la següent informació.

a. Cada pokémon ha de tenir un **identificador (id)** únic.

b. Cada pokémon ha de tenir els següents valors. Escull el tipus de dades més convenient
   i. nom
   ii. tipus (el pokémon pot ser de més d'un tipus)
   iii. pes (amb la unitat de mesura, per exemple kg o g)
   iv. altura (amb la unitat de mesura, per exemple metres o cm)
   v. estadístiques (velocitat, fortalesa, precisió, resistència)

c. Cada pokémon pot tenir un o més **moviments**[1]

d. Cada moviment ha de tenir els següents valors. Escull el tipus de dades més convenient.
   i. nom
   ii. descripció
   iii. potència (https://www.wikidex.net/wiki/Movimiento#Potencia)
   iv. contacte (https://www.wikidex.net/wiki/Movimiento#Contacto)

e. Cada pokémon ha de tenir una **imatge**. El recurs ha de ser accessible i pot ser a internet.

f. Afegiu un camp a cada pokémon anomenat "**evolucions**" que contingui una llista amb els identificadors dels Pokémon als quals pot evolucionar. També ha d'haver-hi la informació del nivell necessari per evolucionar.

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)
```