**7.1**

**Actividad 3**

a. En la traducción de JSON a XML proporcionada anteriormente, he manejado el valor null para la edad de Anna dejándolo como un elemento vacío: <Age/>. No existe un estándar único para representar null en XML, y la elección depende de cómo se va a utilizar y procesar el XML. A continuación indico algunas alternativas para representar null en XML:

1. Elemento vacío: <Age/>
2. Atributo xsi:nil (requiere incluir el namespace de XML Schema Instance): <Age xsi:nil="true"/>
3. Texto que indica nulo: <Age>NULL</Age> o <Age>null</Age>
4. Omitir completamente el elemento si la edad es nula.
5. Uso de un atributo para indicar que el valor es nulo, como <Age isNull="true"/>.

b. En la traducción a XML, los elementos repetidos se manejan mediante el uso de múltiples elementos dentro de un contenedor padre (por ejemplo, <Pets> contiene múltiples <Pet>, y <Friends> contiene múltiples <Friend>).

**Actividad 4**

a. En JSON, els elements amb múltiples instàncies es converteixen en arrays, mentre que els elements únics es converteixen en objectes.

b. Per mantenir junta la informació del preu amb el tipus de moneda, es podria crear un objecte price que contingui tant el valor com la moneda.

c. Algunes etiquetes XML podrien no traduir-se directament a JSON si no contenen informació útil o si la informació es pot inferir d’altres parts del JSON. d. Els caràcters especials com les cometes dobles es poden escapar en JSON utilitzant una barra invertida (\\). Això no afecta la llegibilitat del JSON si es visualitza amb una eina que suporti JSON.

**7.2.**

**Actividad 6**

1. def getUnitatMesuraAltura(pokemon):

return "m" if float(pokemon["medidas"]["altura"]) >= 0.8 else "cm"

1. def isSegonMovimentDeContacte(pokemon):

return pokemon["movimientos"][1]["contacto"] == "Sí"

1. def suma\_estadisticas(pokemon\_data):

estadisticas = pokemon\_data["estadisticas"]

suma\_total = sum([int(valor) for valor in estadisticas.values()])

return suma\_total

1. def media\_estadisticas(pokemon\_data):

estadisticas = pokemon\_data["estadisticas"]

valores\_estadisticas = [int(valor) for valor in estadisticas.values()]

suma\_total = sum(valores\_estadisticas)

media = suma\_total / len(valores\_estadisticas)

return media

1. def suma\_pesos\_pokemon(lista\_pokemon):

suma = 0

for pokemon in lista\_pokemon:

peso = pokemon["medidas"]["peso"]

peso\_numero = float(peso.replace("kg", "").replace(",", ""))

suma += peso\_numero

return suma

1. def nivel\_superior\_o\_igual\_a\_primera\_evolucion(pokemon\_data):

evoluciones = pokemon\_data.get("evoluciones", [])

if evoluciones:

primer\_evolucion = evoluciones[0].get("pre-evolucion", {})

nivel\_requerido = int(primer\_evolucion.get("nivel\_evolucion", 0))

nivel\_pokemon = int(pokemon\_data.get("nivel\_evolucion", 0))

return nivel\_pokemon >= nivel\_requerido

return False

1. def pokemon\_con\_potencia\_mas\_alta(lista\_pokemon):

if not lista\_pokemon:

return None

max\_potencia = -1

pokemon\_potencia\_mas\_alta = None

for pokemon in lista\_pokemon:

movimientos = pokemon.get("movimientos", [])

for movimiento in movimientos:

potencia = movimiento.get("potencia")

if potencia.isdigit():

potencia\_numero = int(potencia)

if potencia\_numero > max\_potencia:

max\_potencia = potencia\_numero

pokemon\_potencia\_mas\_alta = pokemon

return pokemon\_potencia\_mas\_alta