Exercise 4

1. Select a starter Pokémon from the `pokemon` array. Remember, a starter Pokémon's `starter` property is true.

2. Add this Pokémon to the `game.party` array. Which array method will you use to add them?

Solve Exercise 4 here:

\*/

const starters = []

// Adds all availiable starters to the starters array

pokemon.forEach(element => {

if (element.starter) {

starters.push(element)

}

});

// Adds a random starter from starters array to players party. Can change to selection later

game.party.push(starters[Math.floor(Math.random() \* starters.length)])

console.log(`Greetings trainer! You have chosen ${game.party[0].name} to accompany you on your adventure!`)

/\*

Exercise 5

1. Choose three more Pokémon from the `pokemon` array and add them to your party.

2. Consider different attributes like 'type' or 'HP' for your selection. Which array method will you use to add them?

Solve Exercise 5 here:

\*/

console.log('We have decided to randomly give you 3 more Pokemon to help you on your way.')

const pokemonTypes = []

pokemon.forEach(element => {

if (!pokemonTypes.includes(element.type))

pokemonTypes.push(element.type)

})

console.log("We have the following types availiable: "+pokemonTypes.join(', '))

const partyPokemonTypes = []

game.party.forEach(element => {

if (!partyPokemonTypes.includes(element.type))

partyPokemonTypes.push(element.type)

})

console.log('You already have '+partyPokemonTypes.join(', ')+" so we'll pick "+(4 - game.party.length)+" different types for you.")

let availablePokemonTypes = pokemonTypes

partyPokemonTypes.forEach(element => {

for (let i=0; i<availablePokemonTypes.length; i++) {

if (availablePokemonTypes[i] === element) {

availablePokemonTypes.splice(i,1)

}

}

})

console.log('Your availiable types are: '+availablePokemonTypes)

while (partyPokemonTypes.length < 4) {

let newTypeIdx = Math.floor(Math.random() \* availablePokemonTypes.length)

partyPokemonTypes.push(availablePokemonTypes[newTypeIdx])

availablePokemonTypes.splice(newTypeIdx,1)

}

console.log("You're going to have "+partyPokemonTypes.join(', ')+' type pokemon!')

for (let i=1; i<partyPokemonTypes.length; i++) {

let possiblePartyMembers = []

pokemon.forEach(element => {

if (element.type === partyPokemonTypes[i]) {

possiblePartyMembers.push(element)

}

});

game.party.push(possiblePartyMembers[Math.floor(Math.random() \* possiblePartyMembers.length)])

}

const partyMembers = []

game.party.forEach(element => {

partyMembers.push(element.name)

})

console.log("You're party is complete! You're training with "+partyMembers.join(', ',"."))

/\*

Exercise 6

1. Set the `completed` property to true for gyms with a difficulty below 3.

2. Think about how you'd loop through the `gyms` array to check and update the `completed` property.

Solve Exercise 6 here:

\*/

console.log("Wow! You're already strong enough to defeat the first two gyms! LET'S GOOOOOOO!")

game.gyms.forEach(element => {

if (element.difficulty < 3) {

element.completed = true

console.log("You've completed the level", element.difficulty,"gym in", element.location+"! Congratulations!")

}

})

/\*

Exercise 7

1. Evolve the starter Pokémon you added to your party earlier. Each starter Pokémon evolves into a specific one.

2. How would you replace the current starter Pokémon in your party with its evolved form?

Hint:

- Pokemon 1: Bulbasaur evolves into Pokemon 2: Ivysaur

- Pokemon 4: Charmander evolves into Pokemon 5: Charmeleon

- Pokemon 7: Squirtle evolves into Pokemon 8: Wartortle

- Pokemon 25: Pikachu evolves into Pokemon 26: Raichu

More Hints: The existing starter Pokemon will be \*replaced\* in your party with the Pokemon it evolved into. When working

with an array of objects, the splice() array method is ideal for replacing one element with another.

Solve Exercise 7 here:

\*/

", let starter = game.party[0].name

console.log(starter, "is feeling weird. He's Evolving!!!")

for (let i=0; i<pokemon.length; i++) {

if (pokemon[i].number === game.party[0].number) {

game.party.shift()

game.party.unshift(pokemon[i+1])

break

}

}

console.log(starter, "has evolved intogame.party[0].name)

/\*

Exercise 8

1. Print the name of each Pokémon in your party.

2. Consider using a loop or an array method to access each Pokémon's name.

Solve Exercise 8 here:

\*/

while (partyMembers.length > 0) {

partyMembers.pop()

}

game.party.forEach(element => {

partyMembers.push(element.name)

})

console.log("You're team is getting stronger! You're now training with "+partyMembers.join(', ',"."))

/\*

Exercise 9

1. Can you print out all the starter Pokémon from the `pokemon` array?

2. Think about how you can identify a starter Pokémon and then log their names.

Solve Exercise 9 here:

\*/

const starterNames = []

starters.forEach(element => {

starterNames.push(element.name)

})

console.log("You've come a long way since you chose between", starterNames.join(', ')+".")

/\*

Exercise 10

Create a method called `catchPokemon` and add it to the `game` object. You should not need to edit the original game object directly. This method should:

- Accept an object as a parameter called `pokemonObj`

- Add the `pokemonObj` to the `game.party` array.

- not return anything

After writing this method, call it and pass in a Pokemon object of your choice from the `pokemon` data to catch it.

Solve Exercise 10 here:

\*/

game.catchPokemon = function(pokemonObj) {

game.party.push(pokemonObj)

console.log(`You caught a ${pokemonObj.name}!`)

}

game.catchPokemon(pokemon[Math.floor(Math.random() \* pokemon.length)])

/\*

Exercise 11

1. Copy the `catchPokemon` method that you just wrote above, and paste it below. Modify it so that it also decreases the number of pokeballs in your inventory each time you catch a Pokémon.

2. How will you find and update the quantity of pokeballs in the `game.items` array?

Tips:

For this exercise, it's okay to have a negative number of pokeballs.

After updating the method, call it and pass in a Pokemon object of your choice from the `pokemon` data to catch it.

Also, log the `game.items` array to confirm that the pokeball quantity is being decremented.

Solve Exercise 11 here:

\*/

game.catchPokemon = function(pokemonObj) {

game.party.push(pokemonObj)

game.items.forEach(item =>{

if (item.name === "pokeball") {

item.quantity--

console.log(`You caught a ${pokemonObj.name}! You have ${item.quantity} pokeballs left.`)

}

})

}

game.catchPokemon(pokemon[Math.floor(Math.random() \* pokemon.length)])

/\*

Exercise 12

1. Similar to Exercise 6, now complete gyms with a difficulty below 6. How will you approach this?

(change the value of `complete` in the qualifying objects from false to true).

Solve Exercise 12 here:

\*/

console.log("Holy Cow!! You've racked up enough power to defeat the gyms through level 5! LET'S GOOOOOOO!")

game.gyms.forEach(element => {

if (element.difficulty < 6 && !element.completed) {

element.completed = true

console.log("You've completed the level", element.difficulty,"gym in", element.location+"! Congratulations!")

}

})

/\*

Exercise 13

1. Create a `gymStatus` method in `game` to tally completed and incomplete gyms.

2. How will you iterate through the `gyms` array and update the tally? Remember to log the final tally.

This method should:

- Not accept any arguments.

- Initially create a constant `gymTally`, which is an object that has two

properties: `completed` and `incomplete`, both of which are initially set to 0.

- Iterate through the objects in the `game.gyms` array and update the

properties on `gymTally` as follows:

- `completed` should count how many gyms in the array have a value of `true`

for their `completed` property.

- `incomplete` should count how many gyms in the array have a value of

`false` for their `completed` property.

- Log the value of `gymTally`.

- The method should not return anything.

For example, if five gym objects have a value of `true` on their `completed` property and three gym objects have a value of `false` on their

`completed` property, the logged value would be: `{ completed: 5, incomplete: 3 }`.

Solve Exercise 13 here:

\*/

game.gymStatus = function () {

const gymTally = {

'completed' : 0,

'incomplete' : 0

}

game.gyms.forEach(gym => {

if (gym.completed) {

gymTally.completed++

}

})

console.log(gymTally)

}

game.gymStatus()

/\*

Exercise 14

1. Add a `partyCount` method to `game` that counts the number of Pokémon in your party.

This method should:

- Not accept any arguments.

- Count the number of Pokemon in the party.

- return the found number of Pokemon in the party.

Solve Exercise 14 here:

\*/

game.partyCount = function () {

const count = game.party.length

console.log(`You're party is growing! You're up to ${count} pokemon!`)

}

game.partyCount()

/\*

Exercise 15

1. Now, complete gyms with a difficulty below 8. Reflect on how this is similar to or different from the previous gym exercises.

(change the value of `complete` in the qualifying objects from false to true).

Solve Exercise 15 here:

\*/

game.gyms.forEach(element => {

if (element.difficulty < 8 && !element.completed) {

element.completed = true

console.log("You've completed the level", element.difficulty,"gym in", element.location+"! Congratulations!")

}

})