Exercise 1

Imagine that you, as a web developer, have been asked to develop a web platform that will track students' progress in learning web development.

First you need to create a student object, which will have 3 properties:

stack. An array of strings, where each string is a technology. Initially, the stack array must be equal to ['HTML'].

level. data type number. level is responsible for the current level of the student and is initially equal to 1.

improveLevel. A function that first increments the student's level property by one. If level is 2nd, then you need to add 'CSS' value to the end of the stack array, if 3rd - add 'JavaScript' value to the end of stack, 4th - 'React', 5th - 'NodeJS'. If the level value has become more than 5, then you need to display in the modal window via alert the message “The student has learned all technologies!”. Also, the improveLevel function must return the updated student object at the very end.

As a result, the value of the stack property after executing the code below should be equal to ["HTML", "CSS", "JavaScript", "React", "NodeJS"].

student

.improveLevel()

.improveLevel()

.improveLevel()

.improveLevel()

.improveLevel()

Note: The chain of consecutive impoveLevel functions works this way because improveLevel returns a student object, and the student object has an impoveLevel method.

Look at an example:

const student = {

improveLevel() {

return this;

}

};

student

.improveLevel()

.improveLevel()

.improveLevel();

Task 2

You have 2 objects dog and bird that contain their description.

const dog = {

name: 'Charlie',

type: 'Dog',

makeSound() {

return 'Wow-Wow'

}

}

const bird = {

name: 'Petya',

type: 'Sparrow',

makeSound() {

return 'Cheek-chirp';

}

}

You need to create a makeDomestic function that will print all information about the animal to the console with the message “type by name name says sound”, where name is the name of the animal, type is its type, and sound is the result of calling the makeSound function.

Also, the makeDomestic function must return a new animal object, which will contain all the parameters that were before (name, type, makeSound), as well as a new isDomestic parameter, which is a boolean data type and indicates whether the animal is domestic or not.

To specify a value for the isDomestic property (true or false) you need to pass it as a parameter to the makeDomestic function.

In addition, inside the makeDomestic function, only this should be used. Those. you are only allowed to access the properties of an animal this.name or this.makeSound, not bird.name or bird.makeSound.

You need to solve this problem in 3 ways: through bind, call and apply. Call the makeDomestic function 3 times using these methods. When using bind, the makeDomestic function must work on the dog object, when using apply and call, on the bird object. Don't forget to pass the isDomestic parameter to the function when calling it.

Task 3

Given a footballer object that has some information about the player and the actions he can perform.

const footballer = {

fullName: 'Cristiano Ronaldo',

attack: () => {

console.log(`${this.fullName} is now with the ball and starting to attack!`);

},

scoreGoal(sound) {

console.log(`${this.fullName} scored a goal! Wow!`);

this.celebrate(sound);

},

celebrate(sound) {

console log(sound);

},

goToSubstitution: function(newPlayer) {

console.log(`${this.fullName} goes to replace. ${newPlayer}` enters the field);

}

}

const attack = footballer.attack;

const score = footballer.scoreGoal;

const substitute = footballer.goToSubstitution;

attack()

score('Siiiiii');

substitute('Paulo Dibala')

Unfortunately, this code does not work correctly. In some cases it prints undefined, and in others it gives an error altogether.

You need to fix this code using bind, call and apply. The attack function must be fixed with bind, score with call, substitute with apply.

Clue:

don't forget that arrow functions don't have their this.