Exercise 1

You are given an array of users users. Each of them has a status property, which can be either "online" or "offline".

const users = [

{

username: 'David',

status: 'online'

lastActivity: 10

}, {

username: 'Lucy',

status: 'offline'

lastActivity: 22

}, {

username: 'Bob'

status: 'online'

lastActivity: 104

}

]

You need to create a new array onlineUsers that will contain objects of only those users whose status is "online".

After that, display the message “The following users are online now: usersOnlineNames” via alert, where usersOnlineNames is a string in which user names are displayed separated by commas.

For the code above, the result should be: “The following users are currently online: David, Bob”.

Task 2

You need to create a giveJobToStudent function that takes 2 parameters:

student. An object containing information about the student.

jobName. The title of the student's new job.

The giveJobToStudent function displays the message “Congratulations! Student fullName has a new job! It is now jobName”, where fullName is the student's name and jobName is the name of the student's new job.

The giveJobToStudent function should return a new student object that will contain all the keys from the student object and also have a new job key with the value of the jobName parameter.

To test the giveJobToStudent function, use the following code:

const student = {

fullName: 'Maxim',

experienceInMonths: 12,

stack: ['HTML', 'CSS', 'JavaScript', 'React'],

}

const updatedStudent = giveJobToStudent(student, 'web developer');

Task 3

You need to create a function that will work with objects. Name it handleObject.

handleObject takes 3 parameters:

obj. The object that the function will operate on.

key. object key.

action. The action that we will perform on the object.

The action parameter can have 4 values:

'get'. If action is 'get', then the handleObject function must return the value of key in obj.

'add'. If action is 'add', then the handleObject function must add a new key key to the object object and assign the value of the empty string "". It is also necessary to return the updated obj object from the function.

'delete'. If action is 'delete', then the handleObject function must remove the key property from obj and return an updated object.

If action is any other value, then the handleObject function must return the obj object.

Test the function on this code:

const student = {

name: 'Maxim',

programmingLanguage: 'JavaScript',

}

const result = handleObjects(student, 'programmingLanguage', 'delete');

console.log('result', result);

Task 4

Imagine there is an investigation going on at the police station. Several crimes have been committed. The police have other higher priority tasks, so they asked you to write a program that will calculate the criminal from the data already known.

Your task is to create a getKiller function. getKiller takes 2 parameters:

suspectInfo. This is an object where the keys are the people suspected of the crime and the values are arrays of the people the suspects saw on the day of the murder.

deadPeople. This is an array with the names of the people the perpetrator killed.

The criminal is the one who saw all the people killed on the day of the murder. The getKiller function must return the name of the killer.

Examples of getKiller function results:

getKiller(

{

'James': ['Jacob', 'Bill', 'Lucas'],

'Johnny': ['David', 'Kyle', 'Lucas'],

'Peter': ['Lucy', 'Kyle'],

}, ['Lucas', 'Bill']

); // Killer James

getKiller(

{

'Brad': [],

'Megan': ['Ben', 'Kevin'],

'Finn': [],

},

['Ben']

); // Killer Megan

Task 5

Have you ever played the lottery? A lottery is a game in which a winner is randomly determined and given a prize. Now your task will be to develop logic for such a game.

You need to create a getWinner function that takes 2 parameters:

applicants. An object where the keys are the numbers of the people to be randomly selected, and the values are the objects of the candidates to win the lottery.

winnerObject. This is an object that stores only 1 prize key, which stores the values of the lottery prize.

You need to randomly select a winning number (a random key in the applicants object) and return an object from the getWinner function that will store the properties from the winnerObject and the winner object.

To get a random value in a range, use the following function:

function getRandomNumberInRange(min, max) {

return Math.floor(Math.random() \* (max - min)) + min;

}

An example of the return result of the getWinner function:

const todaysWinner = {

prize: '$10,000',

}

const winnerApplicants = {

'001': {

name: 'Max',

age: 25

},

'201': {

name: 'Svetlana',

age: 20

},

'304': {

name: 'Catherine',

age: 35

},

}

const resultWinner = getWinner(winnerApplicants, todaysWinner);

console.log('resultWinner', resultWinner); // { prize: '$10,000', name: 'Maxim', age: 25 }

Task 6

Imagine that you are developing a voucher program for a local hospital. Previously, coupons were issued manually and you need to optimize this task.

You need to create a giveTalonsInOrder function that takes 2 parameters:

patients. An array of objects, each of which stores information about the patient's name and its unique id number.

orders. An array of unique id numbers that specifies the order in which patients should queue.

The function should return a new array in which the objects from the patients array will be sorted by id from the orders array.

Look at the possible result of the giveTalonsInOrder function:

const ordersArr = [4, 2, 1, 3];

const people = [

{ id: 1, name: "Maxim" },

{ id: 2, name: "Nicholas" },

{ id: 3, name: "Angelina" },

{ id: 4, name: "Vitaly" },

];

const result = giveTalonsInOrder(people, ordersArr);

console.log('result', result);

/\* Returns an array

[

{ id: 4, name: 'Vitaly' },

{ id: 2, name: 'Nicholas' },

{ id: 1, name: 'Maxim' },

{ id: 3, name: 'Angelina' }

]

\*/