The aims of the exercise are to learn some skills working with objects and classes in JavaScript. However, the exercise does not elaborate on the rather intricate JavaScript prototypal inheritance. That is left for you as a subject of further studies.

Help to complete the tasks of this exercise can be found at least from the following book sources: From the subchapters “Date Type”, “The Math Object” of the chapter 5 “ Basic Reference Types”, from the subchapters “The Object Type”, “Iteration and Spread Operators” of the chapter 6 “Advanced Reference Types”, from different subchapters of the chapter 8 “Objects, Classes,…”, and from the chapter 20 “JSON” of the book “Professional JavaScript for Web Developers” (5th edition) by Matt Frisbie; and from the chapter 4 “Data Structures: Objects and Arrays” of the book “Eloquent JavaScript” (3rd edition) by Marijin Haverbeke.

Embed your theory answers, drawings, codes, and screenshots directly into this document. Always immediately after the relevant question. Return the document into your return box in itsLearning by the deadline.

*On each task and subtask, reference the primary information source you used to get help. You can use AI to gain understanding and to help solving problems, but it is not an acceptable primary source of information. I expect you also to use the original sources of information and to make appropriate references to them.*

Remember to give your own assessment when returning this document.

It’s also recommendable to use Internet sources to supplement the information provided by the course book.

The maximum number of points you can earn from this exercise is 10 + 2 = 12.

**Tasks:**

**1. Explain. (4 \* 0,25 = 1 point)**

a. What is the difference between having two references to the same object and having two different objects that contain the same properties?

Two references to the same object means that modifying one of the references leads to modifying both references, where as with two different objects with the same properties are independent of each other.

b. The keyword new.

Means creating a new object instance of a class, for example:

let my\_dog = new Dog(“Ressu”)

c. The keyword this.

The ‘this’ keyword functions similarly to the keyword ‘self’ in python. Which can be used inside a object or a class definition to refer to the object itself, its variables, or its methods.

d. A shallow copy of an object.

A shallow copy of an object creates a copy of an object with the same properties as the original object, but nested properties (i.e. the properties of properties) are just references.

**2. Object destructuring. (4 \* 0,25 = 1 point)**

You have the following object.

const musician = {

name: 'Sting',

realName: 'Gordon Matthew Thomas Sumner’,

instrument: {

type: 'bass'

}

};

Use object destructuring to do the following assignments.

a. Read the attributes name and instrument into the variables name and instrument.

b. Read the attributes name and instrument into the variables nameOfArtist and instumentOfArtist.

c. Read the type of the instrument into a variable instrumentTypeOfArtist.

d. Read the make of the instrument into a variable instrumentMakeOfArtist. If the attribute is missing from the current object, give it a default value “unknown”.

Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto

Tekoälyn generoima sisältö voi olla virheellistä.

**3. For..in loop, Object.entries, Object.keys, and Object.values. (4 \* 0,5 = 2 points)**

a. Explain for..in loop.

b. Explain static methods Object.entries, Object.keys, and Object.values.

c. Use for..in with the object musician above. Log the attribute names and attribute values on the console. For example, when it is the turn of the attribute realName, the following text should be printed:

realName = Gordon Matthew Thomas Sumner

d. Use the musician object from the task 2 again. Implement also another musician object. Put them both into an array. Give examples of the use of the static methods Object.entries, Object.keys, and Object.values with the object array you just created.

**4. Getters and setters (object accessors). (2 subtask answered gives 0,5 points, 3 subtasks answered gives 1 point)**

a. Create an object song. It has one attribute called name. It has a getter (a virtual attribute) called duration, and a setter that is also called duration. The getter returns the duration of the in minutes and seconds, and the setter can be used to set it.

b. Invoke the setter and the getter.

c. Explain the differences between normal object methods and these getters and setters.

**5. Working with JSON. (4 \* 0,5 = 2 points)**

a. What are the purposes of JSON?

b. There are few differences between JavaScript objects and JSON. List and explain them.

c. Serialize the object musician given in task 2 to a string containing a JSON object literal.

d. Deserialize the JSON object literal back to another JavaScript object.

**6. Working with some common JavaScript library objects. (2 \* 0,5 = 1 point)**

Use JavaScript library standard objects, like Date and/or Math to solve the following tasks.

a. Create a function getRandomIntegerFromRange. It accepts two arguments. The argument startRange should be an integer and it sets the start of the Range. The argument endRange should also be an integer and it sets the end of the Range. The function returns a random integer that is greater or equal to the startRange abd less or equal to the endRange.

b. Create a function getTimeDifferenceInFullDays that returns the number of full days between to dates. It accepts two arguments. The argument startDate is the start date of the period. The function endDate is the end date of the period. Use Date and Math objects.

**7. Creating objects with a JavaScript class. (2 points)**

a. What is a class in object-oriented programming? (0,5 points)

b. Create a class SimpleLotteryMachine. Utilize JavaScript Set object and initialize it in the constructor. When creating an object of this class, the user should be able to give the arguments to set the minimum and the maximum numbers that are used in the lottery draw. The set must include all the integers between and including these numbers. Same number cannot be drawn from the numbers twice. The class must implement a function that can be used to draw one number at a time. (1 point)

c. Use the SimpleLotteryMachine class to create 2 different objects. Use the first object to draw 7 numbers plus 2 additional numbers from 42 numbers. Use the second object to draw 5 numbers plus 2 additional numbers from 42 numbers. (0,5 points)

**8. Creating a utility with a JavaScript class. (2 \* 0,5 = 1 point)**

Here is a list of some valid zip codes: 00100, 87400, and 50010.

Here is a list of some invalid zip codes: 100, 874000, A4500.

a. Create a class ZipValidator. It has 3 static methods: isZipValid, isZipFixable, and fixZip. The first static methods accept has a parameter zipCode and checks that the given argument contains only numbers, and that it contains exactly five numbers. It returns true or false. The second static method has the same parameter, and returns true, if the given argument has a length less than or equal to 5, but doesn’t contain any other characters than numbers from 0 to 9. Otherwise, it returns false. The third static method has also a parameter zipCode. If the given argument has a length less than five characters, the method prefixes it with leading zeros. The method returns a valid zipCode. Use the class and call the static methods.

b. What is the difference between static methods and normal (instance) methods? What are the use cases for static methods?

**9. Extending JavaScript classes. (2 \* 0,5 = 1 point)**

a. Create a class ComplexLotteryMachine. This class extends the class SimpleLotteryMachine. The main differences to the simple class are that this class maintains 2 separate sets of numbers, one for the (normal) numbers and one for the additional numbers. They can have different minimum and maximum numbers. Same number cannot be drawn from the same set of numbers twice, but same number can be drawn from the different sets. The class must also implement a function that can be used to draw one additional number at a time.

b. Use the ComplexLotteryMachine class to create 2 different objects. Use the first object to draw 7 numbers plus 2 additional numbers from 42 numbers. Use the second object to draw 5 numbers plus 2 additional numbers from 42 numbers. Take care that the additional numbers are from a different set than the (normal) numbers. (0,5 points)

**Alternative task 10. Revealing Module pattern and IEFE. (Please note that even though this task is worth 4 \* 0,5 = 2 points, the maximum points available from the exercise are 12. No returned exercises will be rewarded with more points than 12)**

const greeter = (function () {

  let greeting = 'Hello';

  const exclaim = msg => `${msg}!`;

  const greet = name => exclaim(`${greeting} ${name}`);

  const salutation = (newGreeting) => {

    greeting = newGreeting;

  };

  return {

    greet: greet,

    salutation: salutation,

  };

}());

Look at the code above.

a. What is the idea of the code? What is the extra value it produces to JavaScript?

b. What is IEFE?

c. Use object greeter. Call its functions. Try to read and set the greeting attribute without calling a method. What do you notice?

d. Look at the object the function returns. Use property value shorthands to make it a bit less verbose. Do you lose anything when using the shorthands?