Задачи на закрепление ссылочного типа данных и spread оператора:

1. Без запуска кода проговорите что тут происходит и что будет в консоли

const a = {};

const b = {};

const c = a;

const d = c;

console.log(a === b); //false

console.log(a === c); //true

console.log(c === d); //true

console.log(a === c); //true

1. Что будет в консоль логах, ответьте без запуска кода и ответьте на вопрос

const a = {}; //#12345

const b = a; //#12345

b['test'] = 'test value';

console.log(b === a); //true

console.log(a); // {test: 'test value'}

// смутирует ли объект a? Да, так как ссылается на одну ячейку в памяти

1. У вас есть объект пользователя с информацией о его имени и адресе. Обновите значение города в адресе пользователя(сделайте это иммутабельно, используя spread оператор)

Исходный объект и ожидаемый результат:

const user = {

name: 'John',

address: {

city: 'New York',

country: 'USA',

},

};

console.log(updatedUser);

// Ожидаемый вывод: { name: 'John', address: { city: 'Toronto', country: 'USA' } }

1. У вас есть массив и новый элемент, который вы хотите добавить в конец этого массива. Напишите функцию addElementToArray, которая принимает исходный массив и новый элемент, и возвращает новый массив, в котором новый элемент добавлен в конец(используйте spread опретаор для решения)

Ожидаемый результат:

const originalArray = [1, 2, 3, 4, 5];

const newArray = addElementToArray(originalArray, 6);

console.log(newArray);

// Ожидаемый вывод: [1, 2, 3, 4, 5, 6]

Задачи для отработки деструктуризации:  
  
1) Извлечь значения свойств объекта в переменные name, age, city

const user = { name: 'John', age: 25, city: 'New York' };

и вывести в консоль эти переменные

console.log(name); // 'John'

console.log(age); // 25

console.log(city); // 'New York'

2) Извлечь значения из массива в переменные secondFruit, thirdFruit, а первую переменную пропустить

const fruits = ['apple', 'banana', 'orange'];

и вывести в консоль эти переменные

console.log(secondFruit); // 'banana'

console.log(thirdFruit); // 'orange'

3) Использовать деструктуризацию в параметрах функции

const person = { name: 'Alice', age: 30 };

// Деструктуризация в параметрах функции

function printPersonDetails(тут применить деструктуризацию надо) {

console.log(`Name: ${name}, Age: ${age}`);

}

printPersonDetails(person); // Вывод: Name: Alice, Age: 30

4) Установите значение по умолчанию для свойства year которого нет в объекте car

const car = { brand: 'Toyota', model: 'Camry' };

выведите в консоль его

console.log(brand); // 'Toyota'

console.log(model); // 'Camry'

console.log(year); // 2022 (значение по умолчанию)

5) Извлечь значения из массива объектов

const students = [

{ name: 'Alex', grade: 'A' },

{ name: 'Emma', grade: 'B' },

{ name: 'Chris', grade: 'C' },

];

и вывести их в консоль

console.log(student1); // 'Alex'

console.log(student3); // 'Chris'

6) Установите значение по умолчанию для свойства age которого нет в объекте person

и переименовать переменные, чтобы достать значение с свойства firstName

const person = { firstName: 'Max', lastName: 'Johnson' };

const firstName = 'first Name: Nina';

const lastName = 'last Name: Ivanov;

и вывести результат в консоль

console.log(тут название вашей новой перемееной вместо firstName); // 'Max'

console.log(тоже самое с lastName сделать); // 'Johnson'

console.log(age); // 25 (значение по умолчанию)

**Решение задач на spread оператор и ссылочный тип данных:**

1)

const a = {};

const b = {};

const c = a;

const d = c;

console.log(a === b);//false

console.log(a === c);//true

console.log(c === d);//true

console.log(a === c);//true

2)

const a = {};

const b = a;

b['test'] = 'test value';

console.log(b === a);//true

console.log(a);//{ test: 'test value' }

// смутриурет ли объект a? - да, смутирует

3)

const user = {

name: 'John',

address: {

city: 'New York',

country: 'USA',

},

};

const updatedUser = {

...user,

address: {

...user.address,

city: 'Toronto',

},

};

console.log(updatedUser);

// Ожидаемый вывод: { name: 'John', address: { city: 'Toronto', country: 'USA' } }

4)

function addElementToArray(array, element) {

return [...array, element];

}

const originalArray = [1, 2, 3, 4, 5];

const newArray = addElementToArray(originalArray, 6);

console.log(newArray);

// Ожидаемый вывод: [1, 2, 3, 4, 5, 6]

**Решение задачек на деструктуризацию**

1)

const user = { name: 'John', age: 25, city: 'New York' };

const { name, age, city } = user;

console.log(name); // 'John'

console.log(age); // 25

console.log(city); // 'New York'

2)

const fruits = ['apple', 'banana', 'orange'];

const [, secondFruit, thirdFruit] = fruits;

console.log(secondFruit); // 'banana'

console.log(thirdFruit); // 'orange'

3)

const person = { name: 'Alice', age: 30 };

// Деструктуризация в параметрах функции

function printPersonDetails({ name, age }) {

console.log(`Name: ${name}, Age: ${age}`);

}

printPersonDetails(person); // Вывод: Name: Alice, Age: 30

4)

const car = { brand: 'Toyota', model: 'Camry' };

const { brand, model, year = '2022' } = car;

console.log(brand); // 'Toyota'

console.log(model); // 'Camry'

console.log(year); // 2022 (значение по умолчанию)

5)

const students = [

{ name: 'Alex', grade: 'A' },

{ name: 'Emma', grade: 'B' },

{ name: 'Chris', grade: 'C' },

];

const [{ name: student1 }, , { name: student3 }] = students;

console.log(student1); // 'Alex'

console.log(student3); // 'Chris'

6)

const person = { firstName: 'Max', lastName: 'Johnson' };

const firstName = 'first Name: Nina';

const lastName = 'last Name: Ivanov';

const { firstName: nameVariable, lastName: lastNameVariable, age = 25 } = person;

console.log(nameVariable); // 'Max'

console.log(lastNameVariable); // 'Johnson'

console.log(age); // 25 (значение по умолчанию)