Домашнее задание к лекции 2.1 «Дата и время, математические функции и JSON»

Перед началом работы

1.Активируйте строгий режим соответствия.

2.Скопируйте код ниже и вставьте его в начало своей работы:

```
let positions = [
  {
    title: 'Телепорт бытовой VZHIH-101',
    producer: {
      name: 'Рязанский телепортостроительный завод',
      deferPeriod: 10,
      lot: 3
    },
    price: 10000
  },
  {
    title: 'Ховерборд Mattel 2016',
    producer: {
      name: 'Волжский Ховерборд Завод',
      deferPeriod: 24,
      lot: 14
    },
    price: 9200
  },
  {
    title: 'Меч световой FORCE (синий луч)',
    producer: {
      name: 'Тульский оружейный комбинат',
      deferPeriod: 5,
      lot: 1
    },
    price: 57000
  }
];
```

Задача № 1. Учет по партиям.

Многие поставщики поставляют нам товар фиксированными партиями. Например, если клиент хочет заказать один Ховерборд Mattel 2016, то мы должны закупить у Волжского Ховерборд Завода партию из 14 штук. А если нам нужно поставить 15 таких товаров, то заказать придется 2 партии, или 28 штук. Поэтому нам нужна функция, чтобы не считать это вручную.

Описание функции

Реализовать функцию `lotCalculator`, которая будет рассчитывать количество партий и общую стоимость заказа у поставщика. Функция должна принимать:

1.Товар из глобального массива `positions`, объект:
`title` — наименование товара, строка;
`producer` — поставщик, объект:
`name` — наименование поставщика, строка;
`lot` — размер партии, число;
`price` — цена одной штуки у поставщика, число.
2.Количество товаров, которое нужно поставить, число.

Функция должна вернуть объект вида:
`lots` — рассчитанное количество партий, число;
`total` — общая стоимость заказа у поставщика, число.
Функция должна рассчитать минимально необходимое количество партий, исходя из количества заказанного товараи размера партии (из свойства `lot` поставщика `producer` в объекте товара).

Пример использования функции

```
let result1 = lotCalculator(positions[1], 15);
console.log(result1); // { lots: 2, total: 257600 }

let result2 = lotCalculator(positions[2], 1);
console.log(result2); // { lots: 1, total: 57000 }
```

Процесс выполнения

1.Создайте функцию `lotCalculator`, принимающую нужное количество аргументов.
2.Рассчитайте минимальное количество партий, необходимых для заказа, на основании количества товаров, которое нужно поставить и размера партии из свойств товара.

3.Рассчитайте общую стоимость товаров с учетом рассчитанного количество партий и стоимости товара.

4.Создайте объект со свойствами `lots` и `total`, поместив туда количество партий и общую стоимость товаров.
5.Вызовите функцию `lotCalculator`, используя данные из массива positions и придуманные самостоятельно.

6.Каждый раз выводите полученный результат в консоль в виде:

```
Ховерборд Mattel 2016 15 штук: заказать партий 2, стоимость 257600 Q
```

Задача № 2. Отсрочка платежа.

Поставщики нашего интернет-магазина начали давать нам отсрочку платежа. И мы уже забыли оплатить несколько поставок. Поэтому решено вести учет этих платежей.

Описание функции

Реализовать функцию `deferPay`, которая будет вести учет отсроченных платежей, помещая их в глобальный массив `deferedPayments`. Функция должна принимать следующие аргументы:

1.Поставщик, объект со свойствами:
`name` — название поставщика (строка),
`deferPeriod` — срок отсрочки в земных сутках (число);
2.Сумма отгрузки, число;

3.Дата отгрузки, от которой считать отсрочку, объект `Date`. Функция ничего не должна возвращать. Она должна рассчитать дату оплаты, используя дату отгрузки и свойство поставщика `deferPeriod`. И поместить в глобальный массив `deferedPayments` объект со свойствами:
`producer` — поставщик (объект), переданный в функцию;
`paymentDate` — дата платежа (объект `Date`), рассчитанная внутри функции;
`amount` — сумма платежа (число), переданная в функцию.

Пример использования функции

```
const deferedPayments = [];
const producer = {
  name: 'Рязанский телепортостроительный завод',
  deferPeriod: 10
};

deferPay(producer, 7200, new Date(2030, 4 - 1, 10));

console.log(deferedPayments.length); // 1
console.log(deferedPayments[0].producer.name); // Рязанский
телепортостроительный завод
console.log(deferedPayments[0].amount); // 7200
console.log(deferedPayments[0].paymentDate); // Sat Apr 20 2030 00:00:00 GMT
```

Процесс выполнения

1.Объявите константу `deferedPayments`, присвойте в неё пустой массив.
2.Создайте функций `deferPay`, принимающую нужное количество аргументов.

3. Вычислите дату платежа, используя дату отгрузки и свойство поставщика `deferPeriod`.
4. Создайте новый объект со свойствами producer, `amount` и `paymentDate`, поместив туда поставщика, сумму платежа и рассчитанную дату платежа.
5. Поместите объект в массив `deferedPayments`.
6. Вызовите созданную функцию несколько раз, используя данные из массива `positions` и придуманные самостоятельно.
7. Выведите содержимое массива `deferedPayments` в формате:

```
20.04.2030: Рязанский телепортостроительный завод, сумма 7200 Q
18.05.2030: Волжский Ховерборд Завод, сумма 14600 Q
```

Используйте метод даты `.toLocaleDateString('ru-Ru')`, чтобы вывести её в таком виде.

Задача № 3. Пересчет по курсу валют.

Перед началом работы добавьте в редактор следующий код:

```
function loadCurrencyJSON() {
  return
'{"AUD":44.95,"AZN":33.73,"GBP":73.42,"AMD":0.12,"BYN":30.96,"BGN":32.01,

"BRL":18.8,"HUF":0.2,"DKK":8.42,"USD":58.85,"EUR":62.68,"INR":0.88,"KZT":0.18,

"CAD":44.74,"KGS":0.85,"CNY":8.55,"MDL":2.94,"NOK":7.02,"PLN":14.55,"RON":13.92
,

"ZZZ":79.91,"SGD":41.36,"TJS":7.43,"TRY":15.97,"TMT":16.84,"UZS":0.02,"UAH":2.1
6,
    "CZK":2.32,"SEK":6.6,"CHF":58.69,"ZAR":4.4,"KRW":0.05,"JPY":0.52}';
}
```

Часто нам приходится производить расчеты в разных валютах. И для этого нам бы помогла удобная функция конвертации. У нас уже реализован механизм получения актуальных курсов валют. Он идентичен работе функции `loadCurrencyJSON`. Наш тим-лид говорит, что вернется некая строка, представляющая объект в формате JSON.

Описание функции

Создать функцию `convertCurrency`, которая принимает сумму, код исходной валюты и код валюты, в которую нужно сумму перевести, пересчитывает сумму в новой валюте и возвращает его. Принимает аргументы:

1. `amount` — исходная сумма, число;
2. `from` — буквенный код исходной валюты, строка;
3. `to` — буквенный код валюты, в которую необходимо перевести, строка. Функция должна вернуть число, округленное до сотых.

Для получения актуальных курсов валют воспользуйтесь функцией `loadCurrencyJSON`, которая возвращает строку в формате JSON с курсами валют. Имена свойств — это буквенные коды валют, например USD, а значения свойств — курс этой валюты в условных единицах.

Пример использования функции

```
let price1 = convertCurrency(7000, 'ZZZ', 'USD');
console.log(`Сумма ${price1} USD`);
// Сумма 9505.01 USD

let price2 = convertCurrency(790, 'EUR', 'ZZZ');
console.log(`Сумма ${price2} ZZZ`);
// Сумма 619.66 ZZZ
```