CATEDRA DE INFORMATICĂ I

ИНДИВИДУАЛЬНАЯ РАБОТА №2

Дисциплина :

JavaScript

Студент : BZOVII VALENTIN

Группа: R-2222L

Профессор: Nichita Nartea

Кишинев , 2025

Изложенная задача :

Изучить основы работы с массивами и функциями в JavaScript, применяя их для обработки и анализа транзакций.
Создайте консольное приложение для анализа транзакций.

Шаг 1. Создание массива транзакций

1. Создайте файл main.js для размещения вашего кода.
2. Создайте массив объектов с транзакциями. Каждая транзакция должна содержать следующие свойства:
    - transaction_id - уникальный идентификатор транзакции.
    - transaction_date - дата транзакции.
    - transaction_amount - сумма транзакции.
    - transaction_type - тип транзакции (приход или расход).
    - transaction_description - описание транзакции.
    - merchant_name - название магазина или сервиса.
    - card_type - тип карты (кредитная или дебетовая).
3. Примеры транзакций: Link. Можете использовать их для тестирования.

Шаг 2. Реализация функций для анализа транзакций

Реализуйте следующие функции для анализа транзакций.

1. getUniqueTransactionTypes(transactions)
    - Возвращает массив уникальных типов транзакций.
    - Используйте Set() для выполнения задания.
2. calculateTotalAmount(transactions) – Вычисляет сумму всех транзакций.
3. calculateTotalAmountByDate(transactions, year, month, day) *[extra]*
    - Вычисляет общую сумму транзакций за указанный год, месяц и день.
    - Параметры year, month и day являются необязательными.
    - В случае отсутствия одного из параметров, метод производит расчет по остальным.
4. getTransactionByType(transactions, type) - Возвращает транзакции указанного типа (debit или credit).
5. getTransactionsInDateRange(transactions, startDate, endDate) – Возвращает массив транзакций, проведенных в указанном диапазоне дат от startDate до endDate.
6. getTransactionsByMerchant(transactions, merchantName) – Возвращает массив транзакций, совершенных с указанным merchantName.
7. calculateAverageTransactionAmount(transactions) – Возвращает среднее значение транзакций.
8. getTransactionsByAmountRange(transactions, minAmount, maxAmount) – Возвращает массив транзакций с суммой в заданном диапазоне от minAmount до maxAmount.
9. calculateTotalDebitAmount(transactions) – Вычисляет общую сумму дебетовых транзакций.
10. findMostTransactionsMonth(transactions) – Возвращает месяц, в котором было больше всего транзакций.
11. findMostDebitTransactionMonth(transactions) – Возвращает месяц, в котором было

больше дебетовых транзакций.

12. mostTransactionTypes(transactions)
    - Возвращает каких транзакций больше всего.
    - Возвращает debit, если дебетовых.
    - Возвращает credit, если кредитовых.
    - Возвращает equal, если количество равно.
13. getTransactionsBeforeDate(transactions, date) – Возвращает массив транзакций, совершенных до указанной даты.
14. findTransactionById(transactions, id) – Возвращает транзакцию по ее уникальному идентификатору (id).
15. mapTransactionDescriptions(transactions) – Возвращает новый массив, содержащий только описания транзакций.

Шаг 3. Тестирование функций

1. Создайте массив транзакций и протестируйте все функции.
2. Выведите результаты в консоль.
3. Проверьте работу функций на различных наборах данных.
4. Проверьте работу функций на пустом массиве транзакций *[extra]*.
5. Проверьте работу функций на массиве транзакций с одной транзакцией *[extra]*.

- Формулировка задачи
- Описание цели и основные этапы работы

Практическая часть

В функциях работа с датами :

- calculateTotalAmountByDate,
- getTransactionsInDateRange,
- findMostTransactionsMonth,
- findMostDebitTransactionMonth,
- getTransactionsBeforeDate

- используется объект Date для фильтрации и группировки по датам

Функции Группировки и подсчёт :

- findMostTransactionsMonth и
- findMostDebitTransactionMonth
- реализованы через объект-счётчик (counts), где ключами являются строки в формате YYYY-MM, а значениями — количество транзакций.
- Затем используется reduce для нахождения максимума — это простой и эффективный способ анализа распределения данных.

Работа с уникальными значениями

- getUniqueTransactionTypes использует Set для получения уникальных типов транзакций - способ избежать дубликатов.

Функции, такие как :

- getTransactionsByMerchant,
- getTransactionByType,
- getTransactionsByAmountRange,
- getTransactionsBeforeDate и
- getTransactionsInDateRange,

реализованы по единообразной логике: фильтрация массива с помощью filter.

Вывод:

В ходе выполнения работы была разработана и протестирована коллекция функций для анализа массива транзакций. Реализация охватывает широкий спектр задач, включая:

- фильтрацию транзакций по типу, дате, продавцу и диапазону сумм;
- агрегацию данных (общая и средняя сумма, количество по месяцам);
- извлечение уникальных значений и поиск по идентификатору.

👉 **GitHub: Valentin Bzovii (JavaScript)**

Ответы на контрольные вопросы

1. Методы массивов для обработки объектов в JavaScript

При работе с массивами объектов в JavaScript часто используют следующие методы:

.forEach(callback) — перебирает каждый элемент массива без возврата нового массива. Используется для побочных эффектов.

.map(callback) — возвращает новый массив, содержащий результат применения callback к каждому элементу.

.filter(callback) — возвращает новый массив, содержащий только те элементы, для которых callback вернул true

.reduce(callback, initialValue) — агрегирует значения в одно, например, сумму всех транзакций.

.find(callback) — возвращает первый элемент, удовлетворяющий условию, либо undefined.

.some(callback) / .every(callback) — проверяет, удовлетворяют ли некоторые / все элементы условию.

.sort(callback) — сортирует массив на месте, можно использовать для сортировки по дате или сумме.

2. Как сравнивать даты в строковом формате

Если даты представлены в формате ISO (например, "2024-05-20T10:00:00"), их можно сравнивать напрямую как строки или преобразовывать в объекты Date

3. Разница между .map(), .filter() и .reduce()

.map() — для преобразования каждого элемента.

.filter() — для выбора нужных элементов.

.reduce() — для подсчёта или объединения всех элементов в одно значение.