*Индивидуальная работа*

*Предмет: Java Script*

*Тема: Транзакции*

Подготовила: Доцен Анна

Группа: IA2303

Проверил: преподаватель Nartea Nichita

Кишинев, 2024

Оглавление

***Теоретическая часть 3***

*Формулировка задачи 3*

*Описание целей и основных этапов работы3*

***Практическая часть4***

*Краткое описание реализации4*

***Вывод и ссылка на репозитории Git5***

***Ответы на контрольные вопросы5***

***Список использованных источников6***

# **Теоретическая часть**

# **Формулировка задачи**

Ознакомить студентов с основными функциями и с синтаксисом JavaScript на основе консольного приложения для анализа транзакций.

# **Описание целей и основных этапов работы**

1. Необходимо создать файл transactions.json
2. Необходимо создать файл index.js
3. Необходимо создать класс TransactionAnalyzer
4. Необходимо добавить методы для добавления новой транзакции и получения списка всех транзакций:

а) addTransaction()

б) getAllTransaction()

1. Каждая транзакция должна быть представлена отдельным объектом, содержащим все необходимые данные.
2. В каждую транзакцию необходимо добавить метод string(), который будет возвращать строковое представление транзакции в формате JSON.
3. Необходимо провести анализ транзакций, реализовав следующие методы:
   1. Метод getUniqueTransactionType().
      1. Возвращает массив всевозможных типов транзакций (например, ['debit', 'credit']).
      2. Используйте Set() для выполнения задания.
   2. Метод calculateTotalAmount().
      1. Рассчитывает общую сумму всех транзакций.
   3. Метод calculateTotalAmountByDate(year, month, day).
      1. Вычисляет общую сумму транзакций за указанный год, месяц и день.
      2. Параметры year, month и day являются необязательными.
      3. В случае отсутствия одного из параметров, метод производит расчет по остальным.
   4. Метод getTransactionByType(type).
      1. Возвращает транзакции указанного типа (debit или credit).
   5. Метод getTransactionsInDateRange(startDate, endDate)
      1. Возвращает транзакции, проведенные в указанном диапазоне дат от startDate до endDate.
   6. Метод getTransactionsByMerchant(merchantName).
      1. Возвращает транзакции, совершенные с указанным торговым местом или компанией.
   7. Метод calculateAverageTransactionAmount().
      1. Возвращает среднее значение транзакций.
   8. Метод getTransactionsByAmountRange(minAmount, maxAmount).
      1. Возвращает транзакции с суммой в заданном диапазоне от minAmount до maxAmount.
   9. Метод calculateTotalDebitAmount() (у которых тип debit).
      1. Вычисляет общую сумму дебетовых транзакций.
   10. Метод findMostTransactionsMonth().
       1. Возвращает месяц, в котором было больше всего транзакций.
   11. Метод findMostDebitTransactionMonth()
       1. Возвращает месяц, в котором было больше дебетовых транзакций.
   12. Метод mostTransactionTypes()
       1. Возвращает каких транзакций больше всего.
       2. Возвращает debit, если дебетовых.
       3. Возвращает credit, если кредитовых.
       4. Возвращает equal, если количество равно.
   13. Метод getTransactionsBeforeDate(date)
       1. Возвращает транзакции, совершенные до указанной даты.
   14. Метод findTransactionById(id).
       1. Возвращает транзакцию по ее уникальному идентификатору.
   15. Метод mapTransactionDescriptions()
       1. Возвращает новый массив, содержащий только описания транзакций.

# Практическая часть

# Краткое описание реализации

Класс *TransactionAnalyzer* представляет собой инструмент для анализа транзакций. Он содержит методы для работы с массивом транзакций, такие как добавление новой транзакции, получение всех транзакций, анализ типов транзакций, расчет общей суммы транзакций, а также другие методы для анализа данных.

Основные части кода включают в себя конструктор класса, методы для добавления и получения транзакций, расчета общей суммы и других метрик. Каждая транзакция представлена объектом, к которому добавлен метод string() для получения строкового представления транзакции в формате JSON.

Методы класса реализуют различные функции, такие как вычисление общей суммы транзакций, нахождение наиболее частого типа транзакций, определение месяца с наибольшим количеством транзакций, а также другие задачи анализа данных, предусмотренные требованиями.

# Вывод и ссылка на репозитории Git

В ходе создания консольного приложения для анализа транзакций в JavaScript, мы ознакомились с основными функциями и синтаксисом этого языка. В процессе разработки мы использовали различные возможности JavaScript, такие как работа с массивами, использование объектов, создание и использование классов.

Получаем что:

- JavaScript обладает мощным функционалом для обработки данных и создания приложений, даже в консольной среде.

- Массивы и методы работы с ними, такие как map, filter и reduce, позволяют эффективно обрабатывать и анализировать данные.

- Классы в JavaScript предоставляют удобный способ организации кода, позволяя создавать экземпляры объектов с общими свойствами и методами.

- Разработка консольного приложения для анализа транзакций позволила нам понять, как использовать язык программирования JavaScript для решения практических задач.

Ссылка: <https://github.com/Any-utka/-/tree/main/js>

# Ответы на контрольные вопросы

1. Какие примитивные типы данных существуют в JavaScript?

- Число (number)

- Строка (string)

- Булево значение (boolean)

- null

- undefined

- Символ (symbol)

1. Какие методы массивов вы использовали для обработки и анализа данных в вашем приложении, и как они помогли в выполнении задачи?

* map () – служит для преобразования массива объектов в массив описаний транзакций.
* filter () – служит для фильтрации транзакций по различным критериям, таким как тип транзакции или диапазон суммы.
* reduce () – служит для вычисления общей суммы транзакций и других агрегированных значений.

1. В чем состоит роль конструктора класса?

Роль конструктора класса заключается в том, чтобы инициализировать новые классы.

1. Каким образом вы можете создать новый экземпляр класса в JavaScript?

Для создания нового экземпляра класса можно использовать ключевое слово ***«new».***

# Источники:

1. <https://github.com/MSU-Courses/javascript_typescript/blob/main/lab/LI1/JS01.md>
2. <https://github.com/MSU-Courses/javascript_typescript/blob/main/lab/LI1/files/transaction.json>