**Лабораторная работа № 15-16**

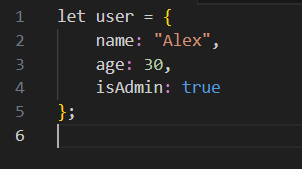
**ООП в JavaScript**

**Контрольные вопросы:**

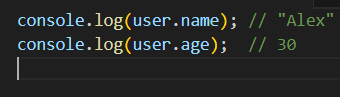
1. Что понимается под объектом в JavaScript? Приведите примеры известных вам объектов в JavaScript.

В JavaScript объект — это коллекция ключей (свойств), которые могут содержать значения любого типа: строки, числа, массивы, другие объекты и функции. Объекты используются для моделирования сущностей реального мира, таких как пользователь, машина, книга и т.д.

Примеры известных объектов:

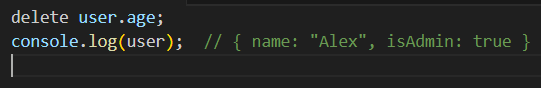
* document: объект, представляющий HTML-документ.
* console: объект для взаимодействия с консолью (например, вывод данных через console.log()).
* Math: объект, содержащий математические функции и константы.
* ****Array: объект, представляющий массив и его методы.

1. Приведите пример описания объекта в JS.
2. Приведите пример доступа к свойствам объекта в JS.

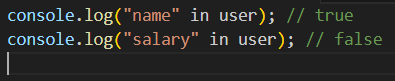
Через точечную нотацию

Через квадратные скобки

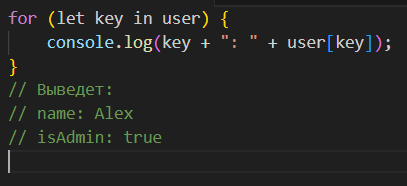
1. Какая команда в JS позволяет выполнить удаление свойства? Приведите пример.

Команда delete позволяет удалить свойство объекта.

1. Какая команда в JS позволяет выполнить проверку существования свойства? Приведите пример.

Команда in используется для проверки существования свойства в объекте.

1. Какая команда в JS позволяет выполнить перебор свойств объекта? Приведите пример.

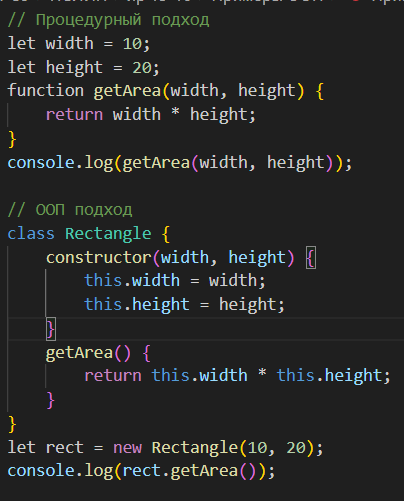
Команда for...in позволяет выполнить перебор свойств объекта.

**Тема "ООП в Java Script**

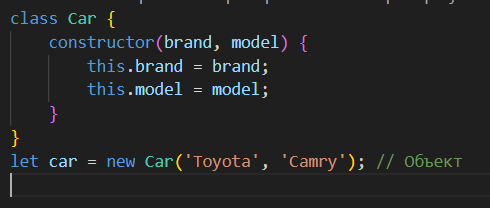
* 1. В чем заключается суть ООП программирования и его отличие от традиционного процедурного программирования?

ООП (Объектно-Ориентированное Программирование) — это парадигма программирования, где основной акцент делается на объект, который содержит данные и методы для работы с этими данными. Объекты взаимодействуют друг с другом через методы, что позволяет создавать более модульные и легко расширяемые системы.

Отличие от процедурного программирования:

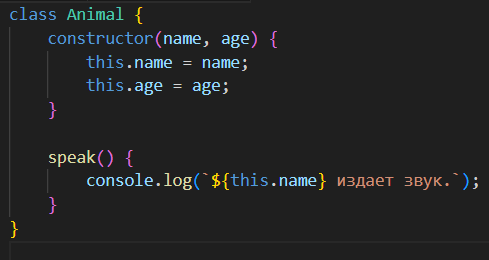
* В процедурном программировании акцент делается на функции и их последовательное выполнение. Данные и функции отделены друг от друга, что усложняет управление сложными системами.
* В ООП функции (методы) и данные (поля) объединяются в объекты, что способствует инкапсуляции и абстракции данных.
  1. Что понимается под Классом и Объектом в JS?

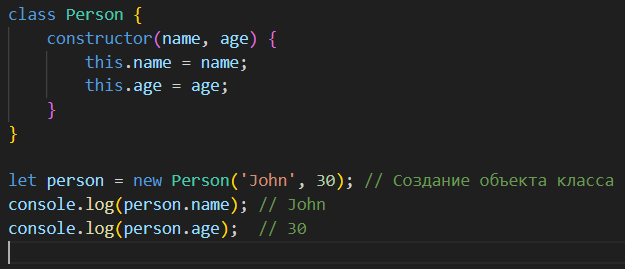
Класс — это шаблон для создания объектов, который определяет свойства и методы, общие для всех объектов этого типа.

Объект — это конкретный экземпляр класса, который содержит собственные данные и может выполнять методы, описанные в классе.

* 1. Приведите пример синтаксиса класса в JS.

Пример класса в JavaScript:

* 1. Как осуществляется создание объектов класса? Приведите пример.

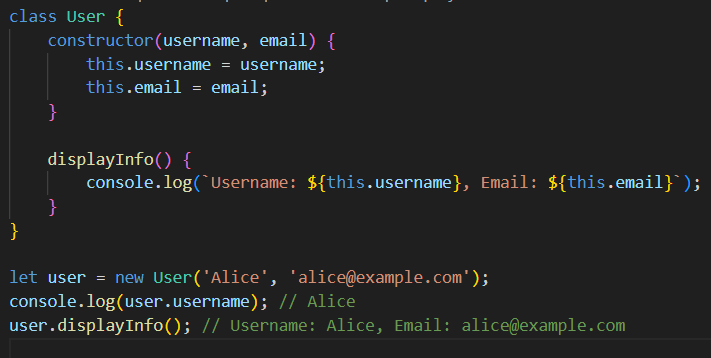
Объект создается с помощью ключевого слова new, которое создает новый экземпляр класса.

* 1. Как происходит обращение к полям и методам внутри класса?

Для обращения к полям и методам объекта используется точечная нотация (.).

### 6. Как обозначаются в JS публичные, приватные и защищённые поля и методы?

В JavaScript поля и методы класса можно сделать публичными, приватными или защищёнными.

* **Публичные поля и методы** доступны из любого места, как внутри класса, так и снаружи. В JS они создаются без каких-либо специальных обозначений.
* **Приватные поля и методы** обозначаются с помощью символа # перед именем переменной или метода и доступны только внутри класса.
* **Защищённые поля и методы** (формально нет прямого механизма в JS), но их можно условно реализовать через соглашения, добавляя символ \_ перед именем поля или метода. Это всего лишь соглашение, не обеспечивающее строгой приватности.
  1. Как обозначаются в JS публичные, приватные и защищённые поля и методы?

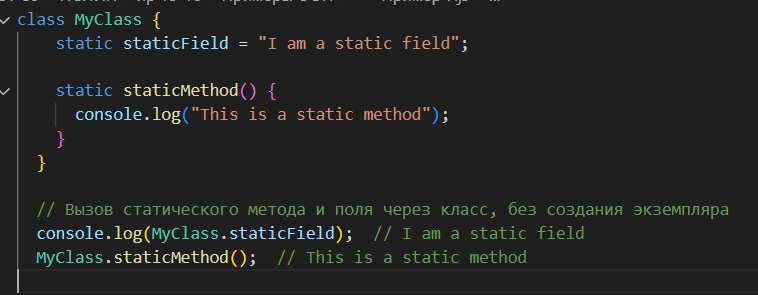
В JavaScript поля и методы класса можно сделать публичными, приватными или защищёнными.

Публичные поля и методы доступны из любого места, как внутри класса, так и снаружи. В JS они создаются без каких-либо специальных обозначений.

Приватные поля и методы обозначаются с помощью символа # перед именем переменной или метода и доступны только внутри класса.

Защищённые поля и методы (формально нет прямого механизма в JS), но их можно условно реализовать через соглашения, добавляя символ \_ перед именем поля или метода. Это всего лишь соглашение, не обеспечивающее строгой приватности.

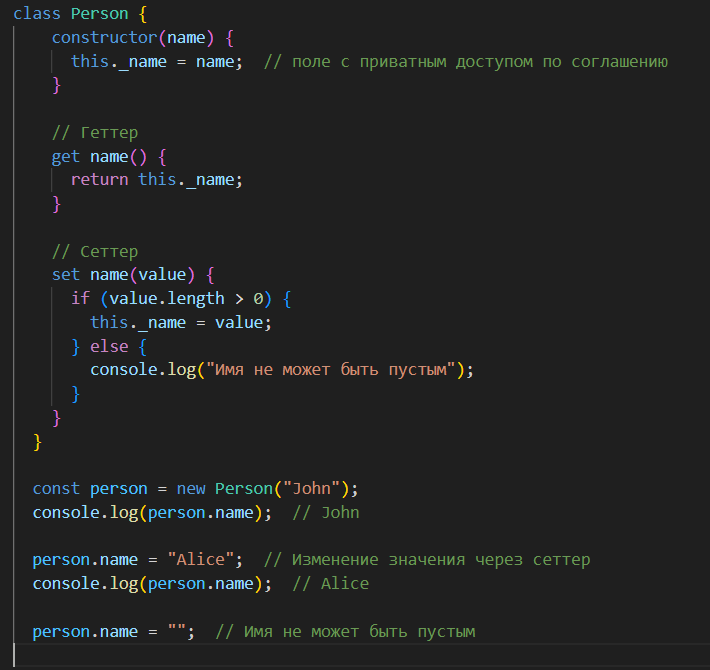
* 1. Для чего в ООП в JS используется служебное слово static?

Ключевое слово static используется для определения статических методов или полей в классе. Статические методы и поля принадлежат самому классу, а не экземплярам этого класса. Это значит, что они могут быть вызваны без создания экземпляра класса.

* 1. Что такое геттеры и сеттеры? Для чего и как они используются в ООП в JS?

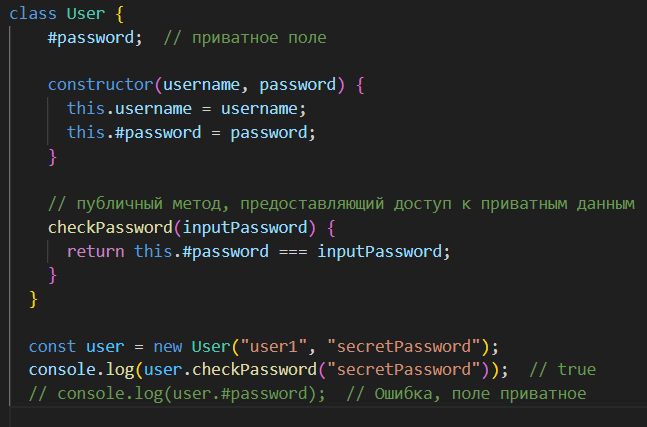
Геттеры и сеттеры — это специальные методы для получения (геттер) и установки (сеттер) значений свойств объекта. Они позволяют контролировать доступ к полям объекта и выполнять дополнительные действия при чтении или изменении значений, такие как валидация данных.

Геттер используется для получения значения свойства.

Сеттер используется для изменения значения свойства.

* 1. Что понимается под базовыми принципами ООП: инкапсуляция, наследование и полиморфизм?
* Инкапсуляция — это принцип скрытия деталей реализации и предоставление доступа только к необходимым данным и методам через публичный интерфейс. В JavaScript это реализуется с помощью приватных полей и методов.
* Наследование — это способность одного класса наследовать свойства и методы другого класса. Наследование позволяет переиспользовать код родительского класса в дочернем.
* Полиморфизм — это способность методов вести себя по-разному в зависимости от того, каким объектом они были вызваны. Это может включать переопределение методов в дочерних классах.
  1. Примеры реализации базовых принципов ООП в JavaScript

Инкапсуляция:

Наследование:

Полиморфизм:

