**1. Что такое callback-функция и как она используется в JavaScript?**

**Callback-функция** — это функция, переданная в другую функцию в качестве аргумента, которая выполняется после завершения какой-либо операции (например, асинхронной).

**Примеры использования:**

Асинхронные операции (setTimeout, fs.readFile в Node.js).

Обработка событий (addEventListener).

Методы массивов (forEach, map).

**2. Как работает setTimeout, и в чем ее отличие от setInterval?**

**setTimeout** выполняет функцию **один раз** через указанное время (в миллисекундах).

**setInterval** выполняет функцию **периодически** с заданным интервалом.

#### ****3.**** Как можно отменить выполнение функции, запланированной с помощью setTimeout?

Используйте clearTimeout с ID, который возвращает setTimeout

#### ****4.**** Почему использование callback-функций может привести к "адскому коллбэку" (callback hell), и как этого избежать?

**Callback Hell** — это вложенность колбэков, которая усложняет чтение кода

**Способы избежать:**

**Промисы (Promises)**

**Async/Await**

**Модули типа async.js**

#### ****1.**** В чем основные отличия между объектами (Object) и коллекциями (Map, Set) в JavaScript?

| **Критерий** | **Object** | **Map** | **Set** |
| --- | --- | --- | --- |
| Ключи | Только строки/symbols | Любые типы | Нет ключей (только значения) |
| Порядок элементов | Не гарантирован | Сохраняет порядок вставки | Сохраняет порядок |
| Размер | Нет свойства size | map.size | set.size |
| Производительность | Медленнее при частых изменениях | Оптимизирован для добавления/удаления | Быстрый поиск уникальных значений |

#### ****2.**** Каковы преимущества использования Map для хранения пар "ключ-значение" по сравнению с обычными объектами?

**Гибкие ключи**: можно использовать объекты, числа и т.д.

**Методы управления**: set, get, has, delete.

**Автоматическое определение размера** (size).

**Лучшая производительность** для частых обновлений.

#### ****3.**** Какие методы предоставляет класс Set, и как они могут быть полезны для работы с уникальными значениями?

add() — добавить значение.

has() — проверить наличие.

delete() — удалить значение.

size — количество элементов.

clear() — очистить коллекцию.

**4. Как использовать методы forEach, map и filter с коллекциями?**

**Map:**

const map = new Map([[1, 'a'], [2, 'b']]);

map.forEach((value, key) => console.log(key, value));

**Set:**

const set = new Set([1, 2, 3]);

set.forEach((value) => console.log(value));