1. Перечислите основные глобальные объекты Node.js и поясните их назначение.

global - хранит var-данные на уровне модуля

process - информация о среде выполнения, о текущем процессе

buffer - класс Buffer – предназначен для работы с двоичными данными.

console - используется для печати в stdout и stderr

1. Поясните понятие «асинхронная функция».

Асинхронная функция - это функция, которая выполняется в фоновом режиме, не блокируя основной поток выполнения программы. Она позволяет выполнять операции ввода-вывода, сетевые запросы и другие задачи, которые могут занимать время, без ожидания их завершения. Асинхронные функции обычно принимают обратные вызовы (callback) или возвращают объект Promise для обработки результата выполнения.

1. Поясните понятие «стандартные системные потоки».

Стандартные системные потоки - это три основных потока ввода и вывода, доступные в большинстве операционных систем:

stdin (стандартный вход): Поток для ввода данных, обычно с клавиатуры. В Node.js это представлено как process.stdin.

stdout (стандартный вывод): Поток для вывода данных, обычно на экран. В Node.js это представлено как process.stdout.

stderr (стандартный вывод ошибок): Поток для вывода сообщений об ошибках, обычно на экран. В Node.js это представлено как process.stderr.

1. Поясните назначение функций **process.nextTick**, **setImmediate**. Поясните в чем их разница.

**Event Loop** — это цикл, с помощью которого Node.js может выполнять неблокирующие операции I/O — input/output.



┌─>│ таймеры │  
│ └──────────┬────────────┘  
│ ┌──────────┴────────────┐  
│ │ I/O коллбэки │  
│ └──────────┬────────────┘  
│ ┌──────────┴────────────┐  
│ │ ожидание, подготовка │  
│ └──────────┬────────────┘ ┌───────────────┐  
│ ┌──────────┴────────────┐ │ входящие: │  
│ │ опрос │<─────┤ соединения, │  
│ └──────────┬────────────┘ │ данные, итд. │  
│ ┌──────────┴────────────┐ └───────────────┘  
│ │ проверка │  
│ └──────────┬────────────┘  
│ ┌──────────┴────────────┐  
└──┤ коллбэк `close` │  
└───────────────────────┘

1. **Event Loop: макрозадачи** - выполняются по одной за один проход цикла; **микрозадачи** - на каждом проходе цикл выполняет все накопившееся.

process.nextTick:

process.nextTick помещает функцию в начало текущей фазы событий (event loop).

Это означает, что функция будет выполнена сразу после завершения текущей операции I/O, но до любых других событий в очереди.

Это позволяет создавать "микрозадачи" (microtasks), которые выполняются максимально быстро и приоритетно перед другими событиями.

Это означает, что колбэк будет выполнен как можно быстрее, после завершения текущей операции Event Loop, до начала выполнения следующей фазы Event Loop.

setImmediate:

Функция setImmediate планирует выполнение колбэка в очереди таймеров Event Loop, после текущей фазы Event Loop.

Это означает, что, если есть другие задачи в очереди событий (например, таймеры), они будут выполнены перед setImmediate.