**Ответы на вопросы**

1. Перечислите основные свойства глобальные объекты Node.js и поясните их предназначение.

**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

Существуют только в составе модулей:

**exports** - ссылка на module.exports

**module** - ссылка на текущий модуль

**require** - используется для импорта модулей

1. Поясните понятие «асинхронная функция».

**Асинхронная функция** – это функция, если ее выполнение осуществляется в 2 фазы: 1) заявка на исполнение; 2) получение результата; при этом участвуют два механизма: A-механизм, формирующий заявку и потом получающий результат; B-механизм, получающий заявку от A, исполняющий операцию и отправляющий результат A; продолжительность исполнения операции B-механизмом, как правило, непредсказуемо; в то время пока B-механизм исполняет операцию, А-механизм выполняет собственную работу.

1. Поясните понятие стандартные «системные потоки».

Эти потоки поставляются с node.js и могут быть использованы без подключения дополнительных библиотек.

**process.stdin** - поток на чтение содержит стандартный системный поток ввода.

**process.stdout** - поток на запись, содержащий стандартный системный вывод.

**process.stderr** - поток на запись, содержащий стандартный системный вывод ошибок.

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

Node.js является событийно-ориентированной системой. Все, что происходит в ноде, является реакцией на события и события вызывают каскад колбеков. Этот механизм работает на основе библиотеки libuv и называется циклом событий (event loop).

Цикл событий представляет собой процесс, состоящий из этапов (групп задач), которые выполняются по очереди.

**setImmediate()** предназначен для выполнения сценария после завершения текущей фазы опроса, отправляется в конец очереди.

**process.nextTick** отправляется в начало очереди событий, чтобы она выполнялась сразу после завершения текущей функции.

Основное различие между setImmediate и process.nextTick заключается в том, что обратные вызовы setImmediate помещаются в цикл обработки событий, а process.nextTick нет.

**process.nextTick()** технически не является частью цикла событий. Вместо этого.

**nextTickQueue** будет обрабатываться после завершения текущей операции, независимо от текущей фазы цикла событий.

**process.nextTick()** срабатывает сразу на той же фазе

**setImmediate()** срабатывает на следующей итерации или «тике» цикла событий.