1. What is the Node.js Event Loop?

- Node.js is a run time environment that makes JavaScript runs outside Browser, thanks to it we can use JavaScript to make backend applications.

- Event Loop is the method that Node.js uses to manage which Async code in js will run first, it's inside libUV library wich is made with c++, because Node.js is a single thread blocking I/O runtime is needs a way to handle the Async code and the way is Event Loop.

---

2. What is Libuv and What Role Does It Play in Node.js?

- Libuv is an open source library which is made with c++, Node.js uses it to convert JS code into c++ code to deal with I/O and network tasks because JS can't deal with them.

---

3. How Does Node.js Handle Asynchronous Operations Under the Hood?

- It passes it to the event loop then event loop with each task in its queue to run it with its priority(according to event loop priorities) and also after it finishes.

---

4. What is the Difference Between the Call Stack, Event Queue, and Event Loop in Node.js?

- Call Stack is where Sync tasks run in Node.js like console.log or normal function.
- Event Queue is a queue that holds callback functions ready to be executed and Node.js has a lot of queues with different types and priorities like (Timer queue, I/O queue, check queue, close callbacks queue).
- Event loop that controls these queues and runs them with order and then after it is finished when any tasks that are not handled in any queue it returns to the top and makes another loop.

---

5. What is the Node.js Thread Pool and How to Set the Thread Pool Size?

- Node.js thread pool is a collection of worker threads separated from Node.js main single thread, because Node.js is a single third when these is a big task that need a lot of resources and much time like I/O, DNS lookups, network tasks it pass it to thread pool to make it. with this method we ensure that the main thread is non-blocking.

We can change thread pool side with this code:

```
process.env.UV_THREADPOOL_SIZE = "number of threads";
```

---

6. How Does Node.js Handle Blocking and Non-Blocking Code Execution?

- Node.js runs Non-blocking code in call stack and it runs in the top-level which means that it will appear in the top results, and it passes Blocking code that it takes time to loop

event to handle it and put it in a queue to run it after it finished, event loop run with this sequence:

1. check if there are any timers and put them in the timer queue (set Timeout , set Interval) and it checks if there is an expired timer to execute it.
2. run pending callbacks (TCP errors , DNS lookups)
3. idle/prepare
4. pool phase (check if there are any I/O callbacks and run them).
5. check queue (runs set Immediate)
6. close callback phase (runs close events).
7. after it ends if there is any task.

- Between every phase and another it runs process.nextTick() if it is there and runs promises queue also