Question 1:
What is Libuv?
According to the official libuv documentation, libuv is described as a multi-platform support library with a focus on asynchronous I/O. It was primarily developed for use by Node.js, but it's also used by Luvit, Julia, uvloop, and others.

```
const fs = require('fs');
console.log('start');
// Asynchronous file read (uses libuv)
fs.readFile('example.txt', 'utf8', (err, data) => {
 if (err) throw err;
 console.log('File content:');
});
console.log('end');
```

Libuv allows Node.js to run tasks concurrently despite being single-threaded. So in the code shared above. The output would be:

```
start
end
File content
```

Execution starts - because fs is an I/O operation that could take a long time, it is sent to the libuv thread pool.
Libuv handles this in the background and, once done, pushes the callback to the event loop, which executes it.

Question 2:
Explain the difference between setImmediate(f) and setTimeout(f, Time)
setImmediate
schedules a callback to run at the check phase of the event loop after IO events' callbacks.
while
setTimeout schedules a callback to run after a specific time. The callback function is registered in the timers phase of the event loop.

Question 3:

Explain the difference between process.nextTick(f) and setImmediate(f)?

process.nextTick() is not part of the event loop, it adds the callback into the nextTick queue. Node processes all the callbacks in the nextTick queue after the current operation completes and before the event loop continues.

process.nextTick() callbacks run before any additional I/O events or timers fire in subsequent ticks of the event loop.

While

setImmediate is part of the event loop and schedules a callback to run at the check phase of the event loop after IO events' callbacks.

Question 4

Pls write down the output without executing the following code snippets and check it with result.

```
const fs = require('fs');
//you may assume input.txt is in the same folder
const rd = fs.createReadStream("input.txt");
rd.close();
rd.on("close", () => console.log('readablStream close event'))
fs.readFile('input.txt', "utf-8", (error, data) => {
    if (error) console.log(error);
    else console.log(data)
});
setTimeout(() => console.log("this is setTimeout"), 5000);
setTimeout(() => console.log("this is setTimeout"), 0);

setImmediate(() => console.log("this is setImmediate 1"));
setImmediate(() => {
    console.log("this is setImmediate 2")
    Promise.resolve().then(() => console.log('Promise.resolve inside
setImmediate'));
});
Promise.resolve().then(() => console.log('Promise.resolve 1'));
Promise.resolve().then(() => {
    console.log('Promise.resolve 2')
    process.nextTick(() => console.log('nextTick inside Promise'));
});
process.nextTick(() => console.log('nextTick 1'));
```

Output:

```
nextTick 1
Promise.resolve 1
Promise.resolve 2
nextTick inside Promise
this is setTimeout
this is setImmediate 1
this is setImmediate 2
Promise.resolve inside setImmediate
readablStream close event
[input.txt file contents]
this is setTimeout
```