PSEUDOCODE

```
// node myFile.js

const pendingTimers = [];
const pendingOSTasks = [];
const pendingOperations = [];

// New timers, tasks, operations are recorded from myFile running
myFile.runContents();

function shouldContinue() {
  // Check one: Any pending setTimeout, setInterval, setImmediate?
  // Check two: Any pending OS tasks? (Like server listening to port)
  // Check three: Any pending long running operations? (Like fs module)
  return (
    pendingTimers.length || pendingOSTasks.length ||
pendingOperations.length
  );
}

// Entire body executes in one 'tick'
while (shouldContinue()) {
  // 1) Node looks at pendingTimers and sees if any functions
  // are ready to be called.  setTimeout, setInterval
  // 2) Node looks at pendingOSTasks and pendingOperations
  // and calls relevant callbacks
  // 3) Pause execution. Continue when...
  //  - a new pendingOSTask is done
  //  - a new pendingOperation is done
  //  - a timer is about to complete
  // 4) Look at pendingTimers. Call any setImmediate
  // 5) Handle any 'close' events
}

// exit back to terminal
```

CALLBACK QUEUE

TASK QUEUE

setimeout()

setinterval()

http.get()

MICROTASK QUEUE

promises()

Process.nexTick()