TSR	Recuperación (Ejercicio 1er Parcial)	Unidades 1,2,3	2019-01-25	A

Esta parte consta de tres preguntas tipo test y dos modificaciones de un programa. Las modificaciones aportan 0.4 puntos. Cada pregunta del test tiene 4 opciones y solo una de ellas es correcta. Cada respuesta correcta aporta 0.2 puntos y cada error descuenta 0.067 puntos. El test debe responderse en la hoja final. Las modificaciones deben realizarse en una hoja aparte.

El programa (ejer01.js) que se muestra a continuación recibe como argumento el nombre de dos ficheros de texto, ordena el contenido de cada uno, y genera nuevos ficheros con el sufijo 2 (ej. node ejer01.js a b genera ficheros a2 y b2). Tras escribir cada fichero ordenado muestra un mensaje informativo, y al completar todas las operaciones un mensaje final.

Asume que no hay errores en las operaciones con los archivos, ni en el número de argumentos. Debes tener un cuidado especial en los momentos en que se producen las operaciones de escritura (console.log). Modifica este programa para estos casos independientes:

1 Puede comenzar a procesar el segundo fichero sin necesidad de esperar a que el primero finalice

2 Puede procesar un número arbitrario de ficheros (argumentos), sin obligar a una ordenación secuencial, pero manteniendo el significado y corrección de las operaciones (especialmente escrituras en fichero y en pantalla)

```
1   var fs = require('fs')
2   var f1 = process.argv[2]
3   var f2 = process.argv[3]
4
5   function sl(s) {
6     return s.toString().split("\n").sort().join("\n")
7   }
8   fs.readFile(f1, 'utf-8', function (err, data) {
9     fs.writeFile(f1 + "2", sl(data), 'utf-8', () => {
10      console.log('ok '+ f1)
11      fs.readFile(f2, 'utf-8', function (err, data) {
12        fs.writeFile(f2+"2",sl(data),'utf-8',() => {
13          console.log('ok '+ f2)
14          console.log('terminated')
15        }) // writeFile(f2)
16      }) // readFile(f2)
17    }) // writeFile(f1)
18  }) // readFile(f1)
```

1 *La función de callback (tipo function (err, data) {...}) empleada en las operaciones readFile:*

a Tiene una cabecera diferente de las empleadas en writeFile (tipo ()=>{...}) para poder pasar parámetros.

b Podría ser sustituida por una cabecera tipo (data, err)=>{...} para poder recibir resultados de readFile.

c Podría ser sustituida por una cabecera tipo (err, data)=>{...}, similar a las empleadas en writeFile.

d Puede ser abreviada como (data)=>{...} porque err no se usa en el resto del código.

2 *Si movemos la línea 14, que muestra el mensaje 'terminated' después de la 18 podría ocurrir que …*

a El mensaje se muestre entre los dos mensajes 'ok'.

b El mensaje se muestre antes que el mensaje 'ok' de la línea 10.

c El mensaje se muestre después del mensaje 'ok' de la línea 13.

d Todas las opciones son posibles.

3 *Se puede seguir mostrando el mensaje 'terminated' tras finalizar todas las operaciones realizando el siguiente cambio:*

a Desplazar la línea 14 entre la 15 y la 16

b Desplazar la línea 14 entre la 16 y la 17

c Desplazar la línea 14 tras la 18

d Ninguna de las restantes afirmaciones es cierta