TSR

Examen de les sessions 1 i 2 de la Pràctica 2

1. Presentem seguidament el codi del fitxer origen1.js de la pràctica 2...

```
 1:  const {zmq, lineaOrdenes, error, adios, conecta} = require('../tsr')
 2:  lineaOrdenes("nombre hostSig portSig")
 3:  let salida = zmq.socket('push')
 4:  conecta(salida, hostSig, portSig)
 5:  salida.on('error', (msg) => {error(`${msg}`)})
 6:  process.on('SIGINT', adios([salida],"abortado con CTRL-C"))
 7:  for (let i=1; i<=4; i++) {
 8:          console.log(`enviando mensaje: [${nombre},${i}]`)
 9:          salida.send([nombre,i])
10:  }
```

I el codi del fitxer destino.js:

```
 1:  const {zmq, error, lineaOrdenes, traza, adios, creaPuntoConexion} = require('../tsr')
 2:  lineaOrdenes("nombre port")
 3:  let entrada = zmq.socket('pull')
 4:  creaPuntoConexion(entrada, port)
 5:  function procesaMensaje(filtro, nombre, iteracion) {
 6:          if (!iteracion) {
 7:                  iteracion = nombre
 8:                  nombre = filtro
 9:                  filtro = ""
10:          }
11:          traza('procesaMensaje','filtro nombre iteracion',[filtro, nombre, iteracion])
12:  }
13:  entrada.on('message', procesaMensaje)
14:  entrada.on('error', (msg) => {error(`${msg}`)})
15:  process.on('SIGINT', adios([entrada],"abortado con CTRL-C"))
```

Se sol·licita escriure el programa *filtre.js* , en què no caldrà gestionar els possibles esdeveniments 'error', i adaptar el programa *origen1.js* de manera que respecten totes aquestes condicions simultàniament:

- **Un procés que execute el programa *origen1.js* adaptat podrà interactuar amb tants processos que executen *filtre.js* com decidisca l'usuari. (15%)**
- **El programa *origen1.js* adaptat no ha d'enviar quatre missatges, sinó un missatge cada segon mentre dure la seua execució. El primer missatge inclourà un 1 com a valor del segon segment. Cada nou enviament incrementarà aquest valor en una unitat. (20%)**
- **El programa *filtre.js* rebrà des de la línia d'ordres el nom d'aquest filtre (l'usuari podrà iniciar-ne molts, cadascun amb un nom diferent), així com la informació necessària per intercomunicar-se amb *origen1.js* i *destino.js* . (15%)**
- **En filtrar cada missatge rebut, el programa *filtre.js* inclourà el seu nom com a primer segment addicional i duplicarà el valor rebut a l'últim segment. (15%)**
- **El programa *filtre.js* ha d'utilitzar els tipus de socket adequats i les operacions necessàries perquè la comunicació siga possible entre els tres tipus de procés. (15%)**
- **El programa *filtre.js* espera almenys el nombre de mil·lisegons especificat a l'últim segment del missatge rebut abans de reenviar aquest missatge filtrat a *destino.js* . (20%)**