

Cientes <nombrefifo> <num clientes>

```
mimanejador(int señal)
```

```
{printf  
}
```

```
...
```

```
signal (SIGPIPE, mimanejador);
```

abre <nombrefifo>e solo para escr.

abre <nombrefifo>s para lect y escr.

crea <num clientes> hijos, cada uno de ellos:

```
escribe algo en <nombrefifo>e
```

```
lee <pidproxy> de <nombrefifo>s
```

```
abre fifo.<pidproxy> para escritura
```

MIENTRAS haya caracteres que escribir {

```
escribe en fifo.<pidproxy> //cliente generando datos
```

```
}
```

```
// si se escribe en un fifo aún no abierto para lectura se
```

```
//genera la señal SIGPIPE que por omisión termina el proceso
```

```
while (wait (&estado)!=-1);
```

proxy

MIENTRAS NO FIN DE LECTURA {

```
lee de 0 y escribe en temporal //temporal es solo de este proceso
```

```
}
```

```
dbloqueo= open( "bloqueo", O_RWR|O_CREAT)
```

```
bloquear (dbloqueo, F_WRLCK)
```

```
// ya tenemos la pantalla en ex. mutua
```

```
lee de temporal y escribe en 1 // escribimos en pantalla
```

```
bloquear (dbloqueo, F_UNLCK)
```

```
exit (0)
```

Se ejecuta en primer lugar el servidor en segundo plano:

Servidor <nombrefifo> &

```
mimanejador (int señal)
```

```
{pid=wait(&estado);}
```

```
.....
```

```
signal (SIGCHLD, mimanejador);
```

```
.....
```

```
crea <nombrefifo>e y <nombrefifo>s
```

abre <nombrefifo>e para lect.

abre <nombrefifo>s para lect. y escr.

```
crear archivo "bloqueo" que usan los proxys
```

MIENTRAS NO FIN DE LECTURA {

```
lee algo de <nombrefifo>e // ha llegado un cliente nuevo
```

```
lanza hijo que será el proxy para este cliente
```

```
}
```

```
crea fifo.<pidproxy>
```

```
escribe <pidproxy> en <nombrefifo>s
```

```
abre fifo.<pidproxy> para lectura
```

```
ducplca fifo.<pidproxy> en 0
```

```
execlp("./proxy", "proxy", NULL)
```

```
//no sabe cuántos clientes hay
```

```
//concebido para esperar periodos de tiempo que
```

```
// podrian ser largos pues un cliente podría demorarse
```

```
//un tiempo largo en generar datos
```