

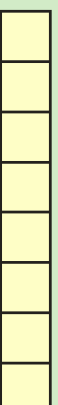


OPERACIONES DE UN PROCESO SOBRE UN FICHERO

Pedro de Miguel Anasagasti

En esta animación se muestra el resultado obtenido al realizar un proceso una serie de operaciones sobre un fichero.

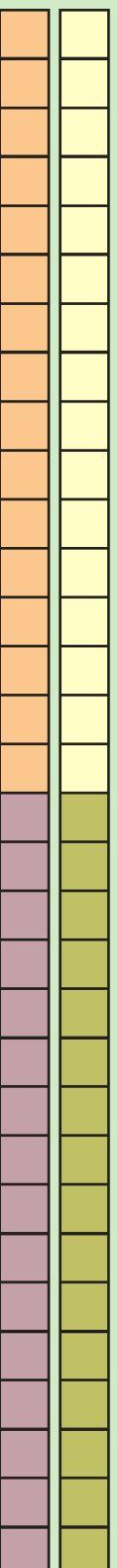
Para que los dibujos sean visibles vamos a considerar que el sector es de 8 bytes



el bloque de 2 sectores = 16 bytes



y la agrupación de 4 bloques = 64 bytes



(Sabemos que el sector real tiene un tamaño de entre 128 a 1024 bytes).

No se contempla el uso de cache de entrada/salida.

- La creación del fichero se puede realizar mediante el servicio 'open' o el servicio 'creat'. Supongamos el siguiente servicio:

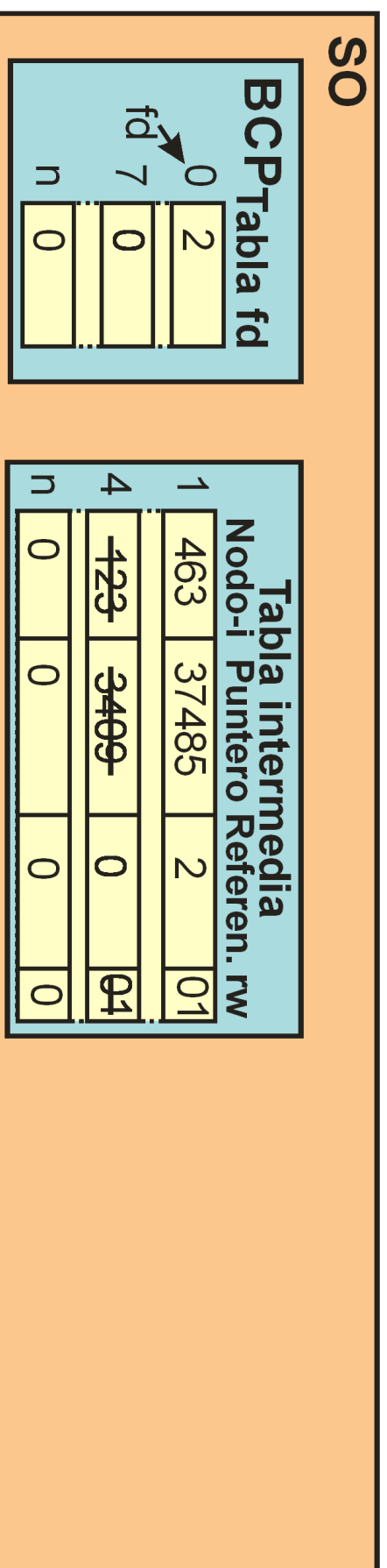
`d=open("f", O_CREAT|O_TRUNC|O_RDWR, 0666);`

Preparación

- Se selecciona la primera entrada libre en la tabla fd del proceso (7).
- Se selecciona la primera entrada libre de la tabla intermedia (4).
- Se selecciona un nodo-i libre (678).

Disco

Memoria



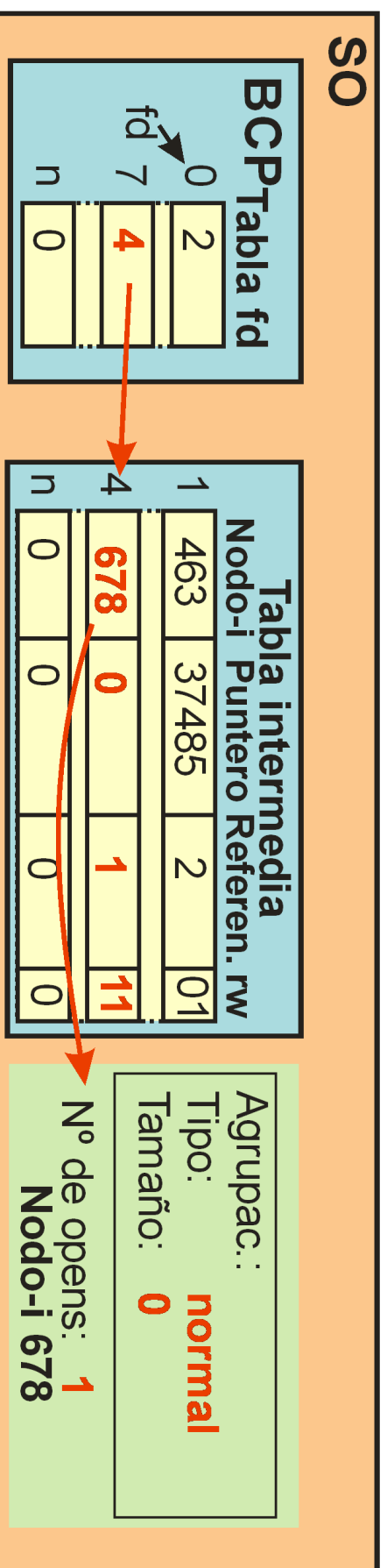
```
d=open("f", O_CREAT|O_TRUNC|O_RDWR, 0666);
```

Ejecución

- $d \leftarrow 7$.
- Se rellenan adecuadamente las entradas de la tabla fd, de la tabla intermedia y el nodo-i.
- El fichero de momento no tiene espacio, no tiene asignada ninguna agrupación y su tamaño es 0 bytes.

Disco

Memoria



escrito = write(d, "abcdefghijklmnpqrstuvwxyz", 26);

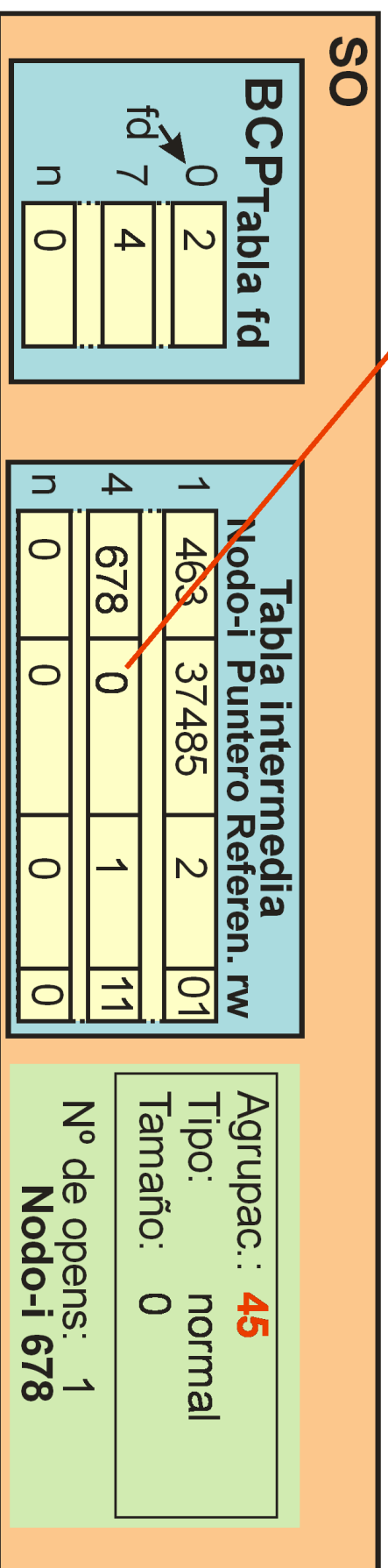
Preparación

- El 'write' se hace a través del descriptor d = 7.
- No hay espacio $\Rightarrow$ se asigna una agrupación (p.e. la 45).
- La agrupación contendrá los datos que se grabaron anteriormente.
- Marcaremos con $\downarrow$ el final del fichero.

Disco

J u a n Z u t a m e n d i R o m e r o I n g e n i e r o I
8 7 4 3 1 \ I g n a c i o P é r e z Z o p i c o P r o g r Agr. 45

Memoria



- La ejecución del 'write' introduce nuevos datos en el fichero, adelanta el puntero de posición y, en su caso, reajusta el tamaño del fichero.
- Puntero $\leftarrow$ 26, Tamaño $\leftarrow$ 26 y escrito $\leftarrow$ 26.
- En este caso el tamaño del fichero aumenta, pasando de 0 a 26 bytes.
- El resto de la agrupación queda disponible para futuro crecimiento.

SO

BCP

Tabla fd

02

4

0

n

fd

7

n

Tabla intermedia

Nodo-i Puntero Referen. rw

146337485201

467826111

n00000

Agrupac.: 45

Tipo: normal

Tamaño: 26

Nº de opens: 1

Nodo-i 678

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	0	0	0	0	0	0	0
8	7	4	3	1	\	I	g	n	a	c	i	o		P	é	r	e	n		Z	o	p	i	c	c	o		P	r	o	g	r
Agr.																																
45																																

SO

0	4	2
0	7	0

1	463	37485	2	01
4	678	26	1	11
n	0	0	0	0

Nodo-i 678

leido=read(d, buf, 26);

Ejecución

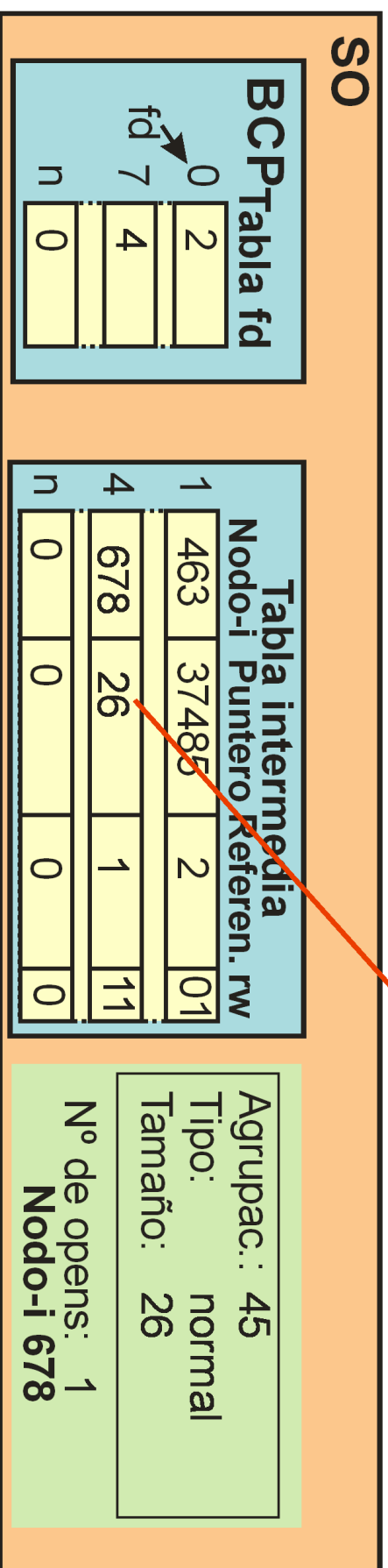
- El 'read' se hace a través del descriptor d = 7.
- No se lee nada, puesto que estamos al final del fichero.
- leido $\leftarrow$ 0.
- buf queda sin modificar.

Disco

a b c d e f g h i j k l m n o p q r s t u v w x y z 0 0 0 0 0 0 0 0 Agr. 45

8 7 4 3 1 \ I g n a c i o P é r e z Z o p i c o l P r o g r

Memoria





Preparación

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	0	0	0	0	0	0	0	0	0	0
8	7	4	3	1	\	I	g	n	a	c	i	o			P	e	r	e	z		Z	o	p	i	c	a		P	r	o	g	r			

Agr. 45

SO

BCP

Tabla fd

0	2
7	4
n	0

Tabla intermedia

Nodo-i Puntero Referen. rw

1	463	37485	2	01
4	678	26	1	11
n	0	0	0	0

Agrupac.: 45

Tipo: normal

Tamaño: 26

Nº de opens: 1

Nodo-i 678

lseek(d, 1, SEEK_SET);

Ejecución

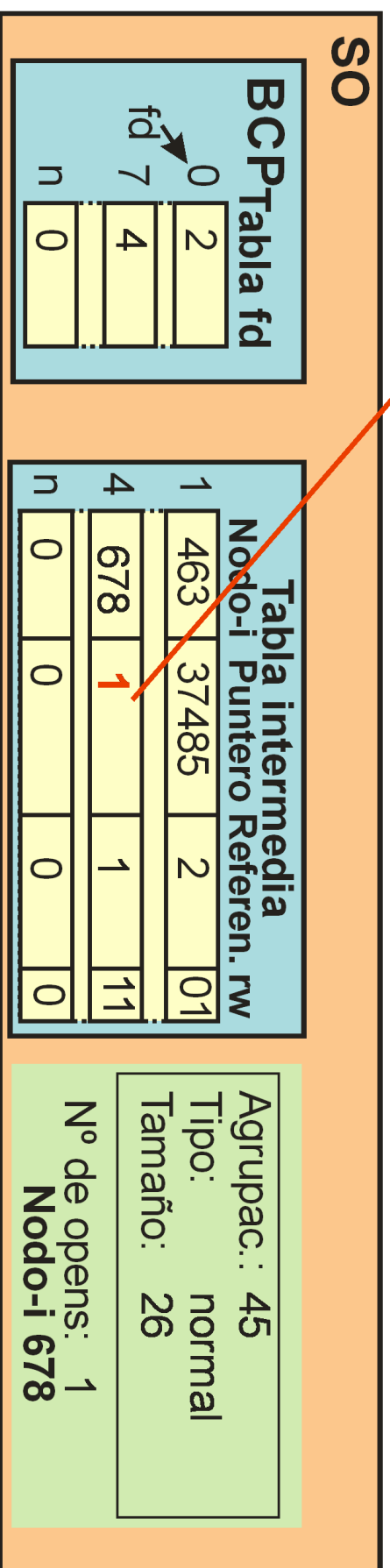
- Puntero $\leftarrow 1$.

Disco

a b c d e f g h i j k l m n o p q r s t u v w x y z 0 0 0 0 0 0 0 0 Agr. 45

8 7 4 3 1 \ I g n a c i o P é r e z Z o p i c o l P r o g r

Memoria



escrito = write(d,"BCDE", 4);

Preparación

- Hay espacio suficiente para realizar la escritura.

Disco

a b c d e f g h i j k l m n o p q r s t u v w x y z 0 0 0 0 0 0 Agr. 45

8 7 4 3 1 \ I g n a c i o P é r e z Z o p i c o l P r o g r

Memoria

SO
BCP Tabla fd

0
fd 7
n

2
4
0

Tabla intermedia				
Nodo-i	Puntero	Referen.	rw	
1	463	37485	2	01
4	678	1	1	11
n	0	0	0	0

Agrupac.: 45
Tipo: normal

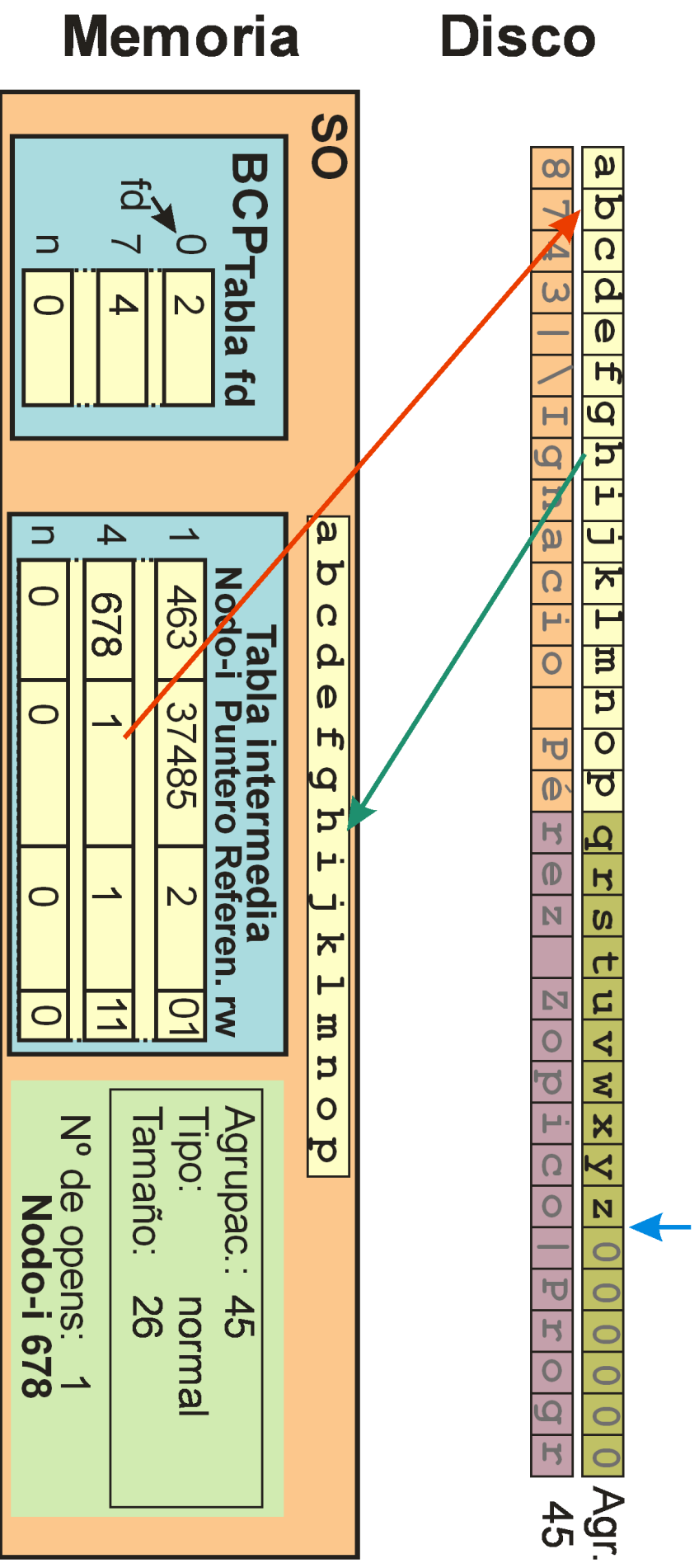
Tamaño: 26
Nº de opens: 1

Nodo-i 678

escrito = write(d,"BCDE", 4);

Preparación

- El sistema de ficheros **lee** el bloque en un buffer para no destruir su contenido.



escrito = write(d,"BCDE", 4);

Preparación

- El sistema de ficheros modifica el buffer.

Disco

a b c d e f g h i j k l m n o p q r s t u v w x y z 0 0 0 0 0 0 Agr. 45

8 7 4 3 1 \ I g n a c i o P é r e z Z o p i c o l P r o g r

Memoria

SO

BCPTabla fd

fd

0

2

4

0

n

Tabla intermedia

Nodo-i Puntero Referen. rw

1	463	37485	2	01
4	678	1	1	11
n	0	0	0	0

Agrupac.: 45

Tipo: normal

Tamaño: 26

Nº de opens: 1

Nodo-i 678

escrito = write(d,"BCDE", 4);

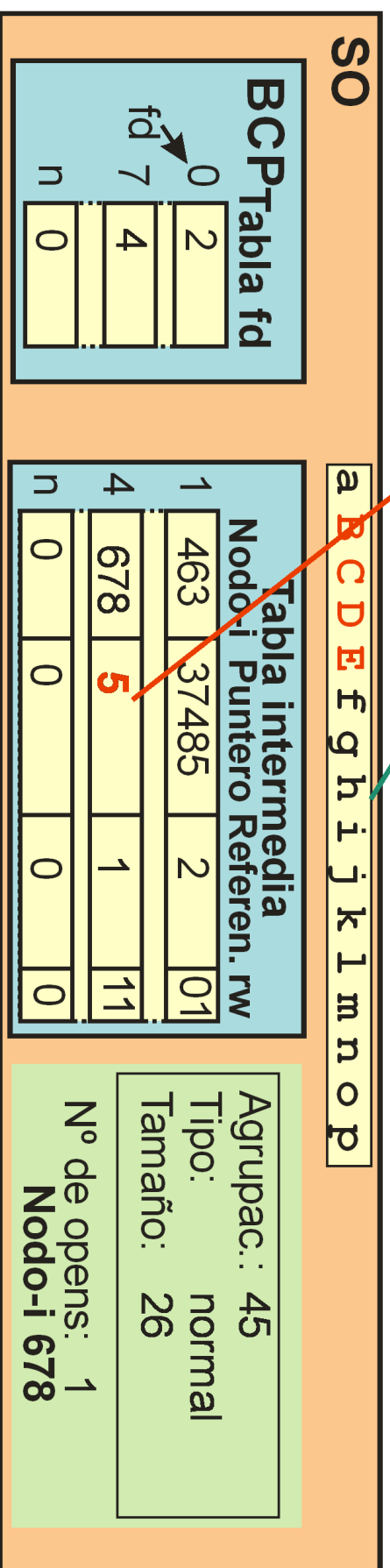
Ejecución: se escribe el bloque completo

- La escritura sobrescribe, NO inserta.
- escrito $\leftarrow$ 4.
- Puntero $\leftarrow$ 5.

Disco



Memoria





- El sistema de ficheros lee el bloque en un buffer.

Memoria



leido=read(d, buf, 5);

Ejecución: se usa la parte solicitada

- leido ← 5.
- Puntero ← 10.
- buf ← fghij.

Disco

a B C D E f g h i j k l m n o p q r s t u v w x y z 0 0 0 0 0 0 Agr. 45

8 7 4 3 1 \ I g n a c i o P é r e z Z o p i c o l P r o g r



Disco

Memoria

SO

BCPTabla fd

0

2

fd 7

4

n

0

Tabla intermedia

Nodo-i Puntero Referen. rw

1	463	37485	2	01
4	678	10	1	11
n	0	0	0	0

Agrupac.: 45

Tipo: normal

Tamaño: 26

Nº de opens: 1

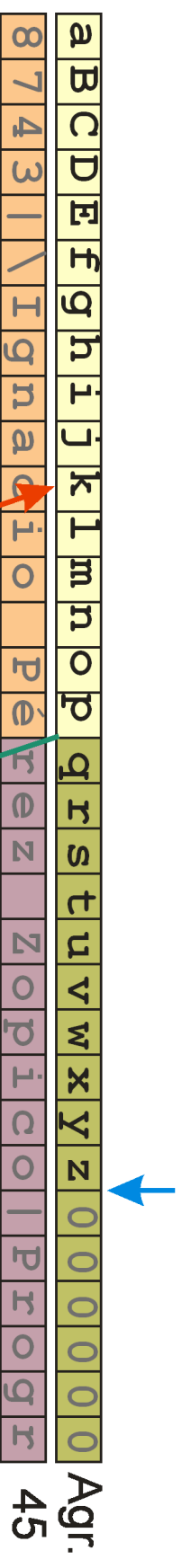
Nodo-i 678

leido=read(d, buf, 200);

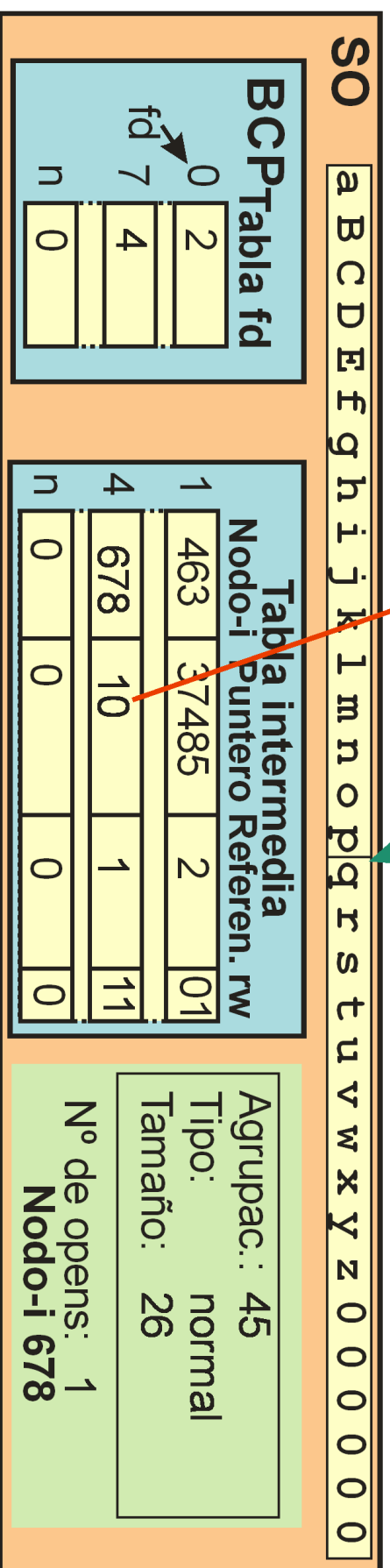
Preparación

- No hay suficientes datos. El SO suministra los que hay.
- Se leen los dos bloques afectados.

Disco



Memoria





- leido $\leftarrow$ 16.
- Puntero $\leftarrow$ 26.
- buf $\leftarrow$ klmnopqrstuvwxyz.

[illegible]

SO

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

BCP

0	2				
fd	4				
7					
n	0				

Tabla intermedia

Nodo-i	Puntero	Referen.	rw
1	463	37485	2
4	678	26	11
n	0	0	0

Agrupac.: 45

Tipo: normal

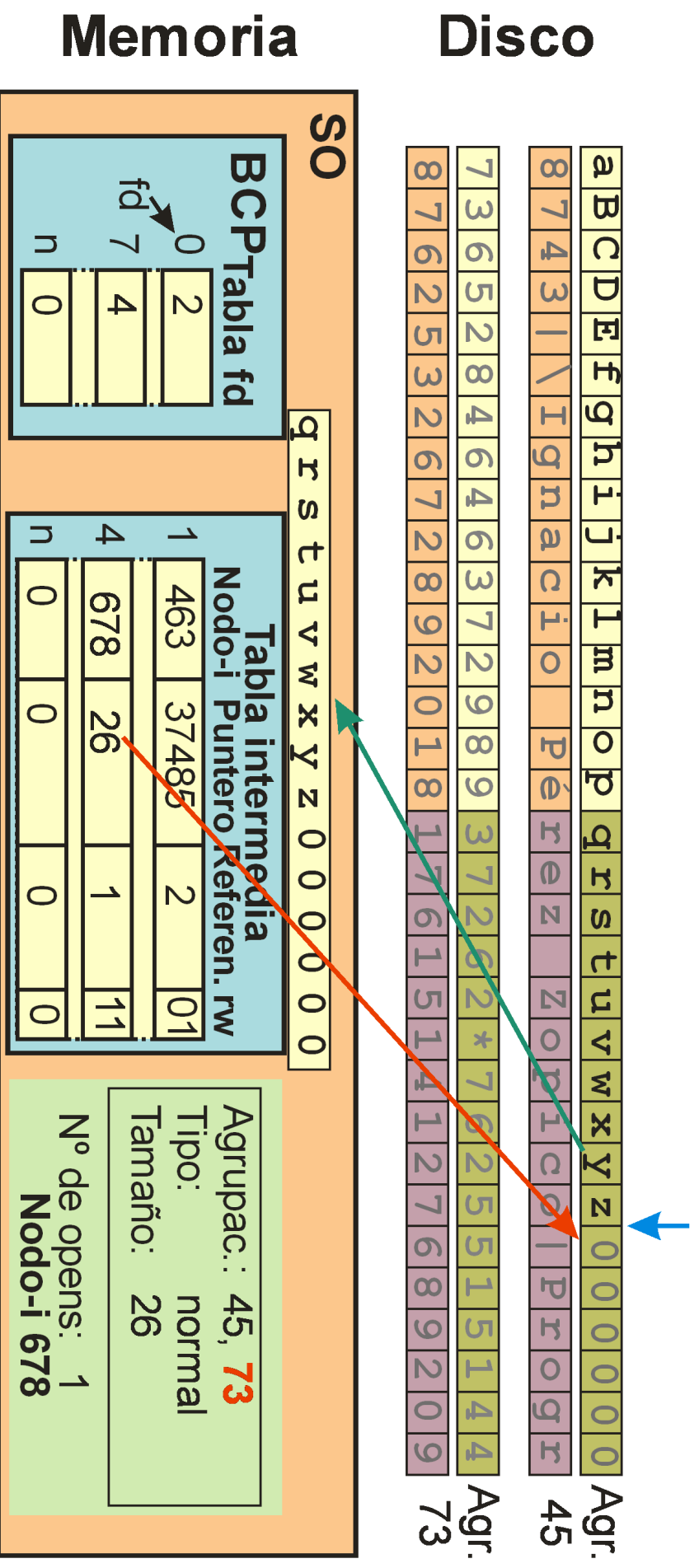
Tamaño: 26

Nº de opens: 1

Nodo-i 678

Preparación

- Se lee el bloque que hay que modificar parcialmente.
- No hay espacio suficiente, por lo que hay que asignar una nueva agrupación al fichero (p.e. la 73).
- La nueva agrupación conserva lo grabado anteriormente.



escrito = write(d, "12345678901234567890123456789012345678901234567890", 40);

Preparación

- Se sobrescribe el buffer.

Disco																									
a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	4	3	1	\	I	g	n	a	c	i	o		P	e	r	e	z	Z	o	p	i	c	o	l
45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45
7	3	6	5	2	8	4	6	4	6	3	7	2	9	8	9	3	7	2	6	2	*	7	6	2	5
8	7	6	2	5	3	2	6	7	2	8	9	2	0	1	8	1	7	6	1	5	1	4	1	2	7
73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73	73
Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.	Agr.

Memoria																									
SO																									
BCP																									
Tabla fd																									
0	2																								
fd 7	4																								
n	0																								
Tabla intermedia																									
Nodo-i Puntero Referen. rw																									
1	463	37	485	2																					
4	678	26		1																					
n	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Agrupac.: 45, 73																									
Tipo: normal																									
Tamaño: 26																									
Nº de opens: 1																									
Nodo-i 678																									

escrito = write(d, "12345678901234567890123456789012345678901234567890", 40);

Ejecución: se escriben los 4 bloques afectados.

- escrito ← 40.
- Puntero ← 66.
- Tamaño ← 66.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6		Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45	

9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	7	2	6	2	*	7	6	2	5	5	1	5	1	4	4	Agr.
8	7	6	2	5	3	2	6	7	2	8	9	2	0	1	8	1	7	6	1	5	1	4	1	2	7	6	8	9	2	0	9	73

Memoria

SO

BCP Tabla fd			
0	2		
fd 7	4		
n	0		

Tabla intermedia

Nodo-i Puntero Referen. rw			
1	463	37485	2
4	678	66	1
n	0	0	0

Agrupac.: 45, 73

Tipo: normal

Tamaño: **66**

Nº de opens: 1

Nodo-i 678



- Se hace un 'lseek' más allá del final del fichero.

- Si se escribe a partir de dicho punto se creará un agujero de 50 bytes.

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	7	2	6	2	*	7	6	2	5	5	1	5	1	4	4
8	7	6	2	5	3	2	6	7	2	8	9	2	0	1	8	1	7	6	1	5	1	4	1	2	7	6	8	9	2	0	9

Agr. 45

Agr. 73

SO

BCP

Tabla fd

0	2
7	4
n	0

Tabla intermedia

Nodo-i Puntero Referen. rw

1	463	37485	2	01
4	678	66	1	11
n	0	0	0	0

Agrupac.: 45, 73

Tipo: normal

Tamaño: 66

Nº de opens: 1

Nodo-i 678

lseek(d, 50, SEEK_CUR);

Ejecución

- Puntero $\leftarrow 116$.
- El tamaño NO aumenta
- El SO debe prohibir el acceso al hueco. Cuando éste se haga efectivo tendrá que borrar su contenido, p.e. rellenándolo con 'ceros'.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45

9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	7	2	6	2	*	7	6	2	5	5	1	5	1	4	4	Agr.
8	7	6	2	5	3	2	6	7	2	8	9	2	0	1	8	1	7	6	1	5	1	4	1	2	7	6	8	9	2	0	9	73

SO

BCPTabla fd

0	2
fd 7	4
n	0

Tabla intermedia

Nodo-i	Puntero	Referen.	rw
1	463	37485	2
4	678	116	1
n	0	0	0

Agrupac.: 45, 73

Tipo: normal

Tamaño: 66

Nº de opens: 1

Nodo-i 678

Memoria

texto = "Esto es la vida";
 escrito = write(d, texto, 6);



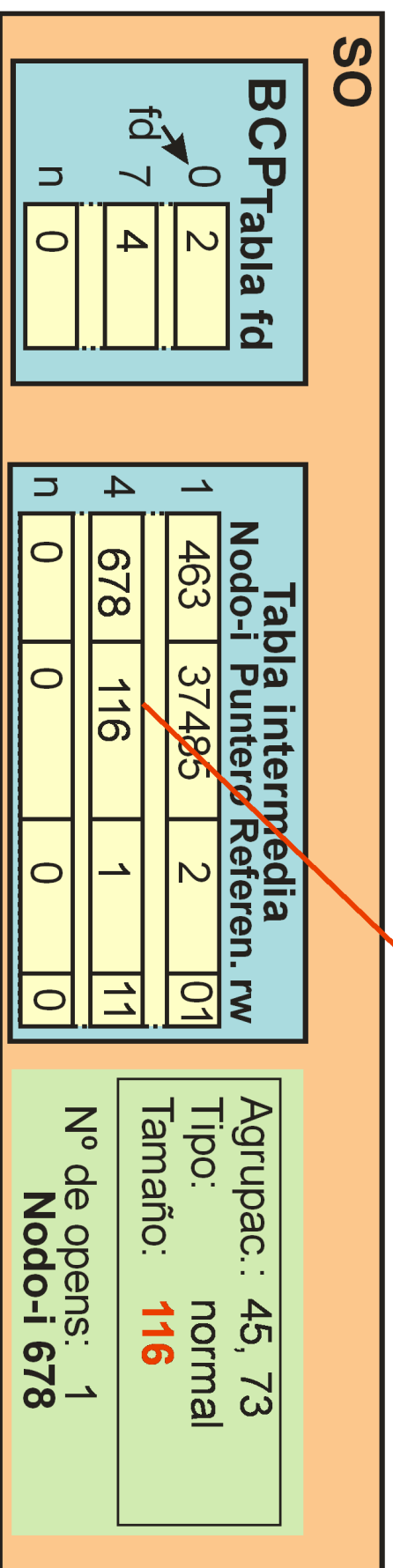
Preparación

- Se crea el agujero. El tamaño aumenta en el agujero + lo escrito
- No hace falta añadir agrupación
- El SO debe borrar el contenido del agujero existente, p.e. rellenándolo con 'ceros'.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	7	2	6	2	*	7	6	2	5	5	1	5	1	4	4	Agr.
8	7	6	2	5	3	2	6	7	2	8	9	2	0	1	8	1	7	6	1	5	1	4	1	2	7	6	8	9	2	0	9	73

Memoria



texto = "Esto es la vida"
escrito = write(d, texto, 6);

Ejecución: Se han tenido que escribir tres bloques

- escrito $\leftarrow 6$.
- Puntero $\leftarrow 126$.
- Tamaño $\leftarrow 126$.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Agr.
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73

↓ (at index 45) → (at index 73)

SO

BCP Tabla fd			
0	2		
fd 7	4		
n	0		

Tabla intermedia				
1	463	37485	2	01
4	678	122	1	11
n	0	0	0	0

Agrupac.: 45, 73	
Tipo: normal	
Tamaño: 122	
Nº de opens: 1	Nodo-i 678

Memoria

`lseek(d, 5000, SEEK_CUR);`

Preparación

- Se hace un 'lseek' más allá del final del fichero.
- Si se escribe a partir de dicho punto se creará un agujero de 5000 bytes.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Agr.
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73

Disco

SO

BCPTabla fd

0	2
7	4
n	0

Tabla intermedia
Nodo-i Puntero Referen. rw

1	463	37485	2	01
4	678	122	1	11
n	0	0	0	0

Agrupac.: 45, 73
Tipo: normal
Tamaño: 122
Nº de opens: 1
Nodo-i 678

Memoria

lseek(d, 5000, SEEK_CUR);

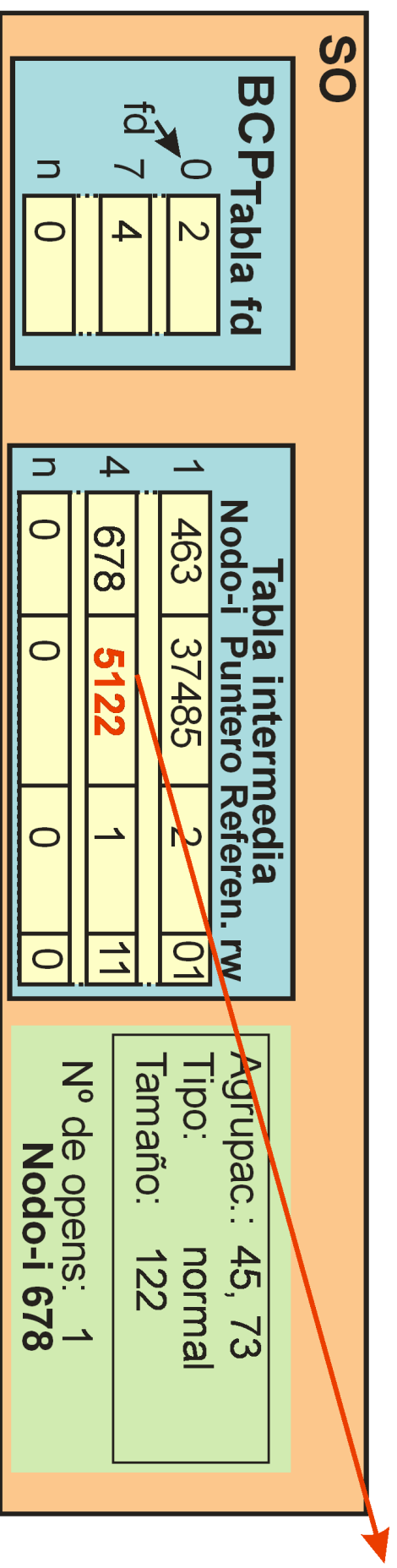
Ejecución

- Puntero $\leftarrow 5106$.
- El SO debe prohibir el acceso al hueco. Cuando éste se haga efectivo tendrá que borrar su contenido, p.e. rellenándolo con 'ceros'.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Agr.
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73

Memoria



a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Agr.
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	E	s	t	o	e	0	0	0	0	0	0	73	

SO

BCP

Tabla fd

02

4

0

fd

7

n

Tabla intermedia

Nodo-i

Puntero

Referen.

rw

146337485201

46785122111

n00000

Agrupac.: 45, 73

Tipo: normal

Tamaño: 122

Nº de opens: 1

Nodo-i 678

close(d);

Ejecución

- Se libera la entrada 7 de la tabla de descriptores.
- Se decrementa el número de duplicados de la entrada 4 de la tabla intermedia. Como se queda en cero se libera dicha entrada.
- Se decrementa el número de opens del nodo-i. Al quedarse en 0 indica que no hay usuarios, por tanto el nodo-i en memoria se libera.

Disco

a	B	C	D	E	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	1	2	3	4	5	6	Agr.
7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	45
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Agr.
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	73

Memoria

SO

BCP

Tabla fd

fd

0

2

7

0

0

n

Tabla intermedia

Nodo-i Puntero Referen. rw

1	463	37485	2	01
4	678	5422	0	14
n	0	0	0	0

Agrupac.: 45, 73

Tipo: normal

Tamaño: 122

Nº de opens: 0

Nodo-i 678

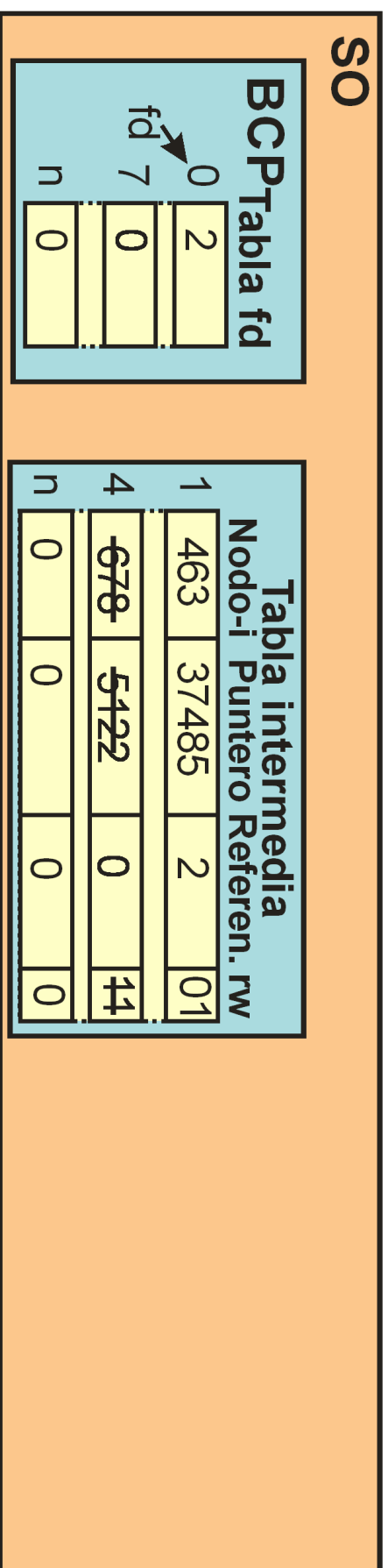
leído=read(d, buf, 26);

Preparación

- El descriptor d = 7 nos lleva a una entrada de la tabla fd no válida.

Disco

Memoria



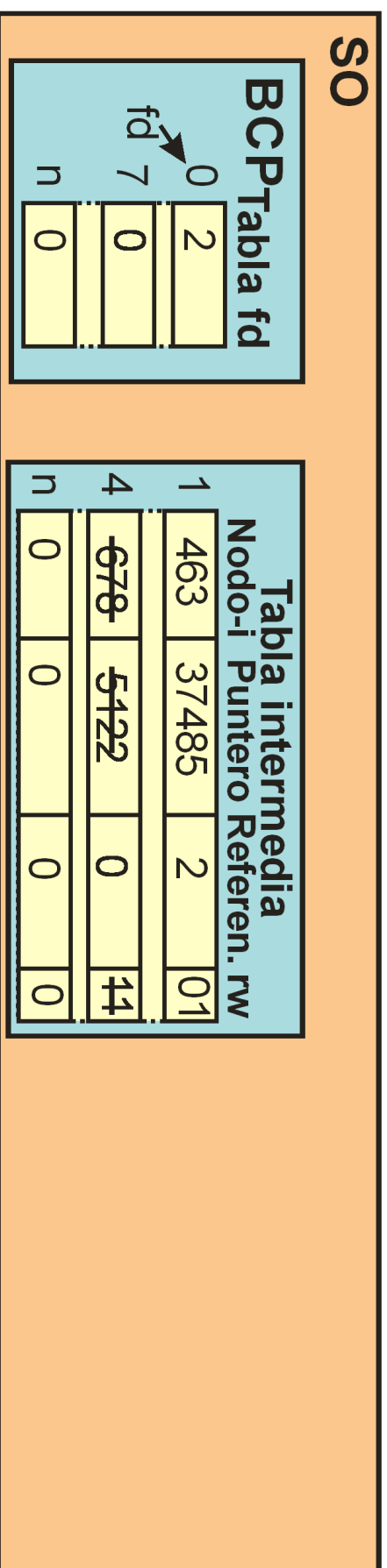
leido=read(d, buf, 26);

Ejecución

- El SO indica que la operación solicitada no es correcta.
- leido $\leftarrow$ -1
- errno $\leftarrow$ EBADF (descriptor inválido).

Disco

Memoria



FIN

