

Tipos de Datos Abstractos

Tipo de dato Cola.

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

cola
? ? ?
pri ult tamDisp
cola

?
pri
?
ult
cola

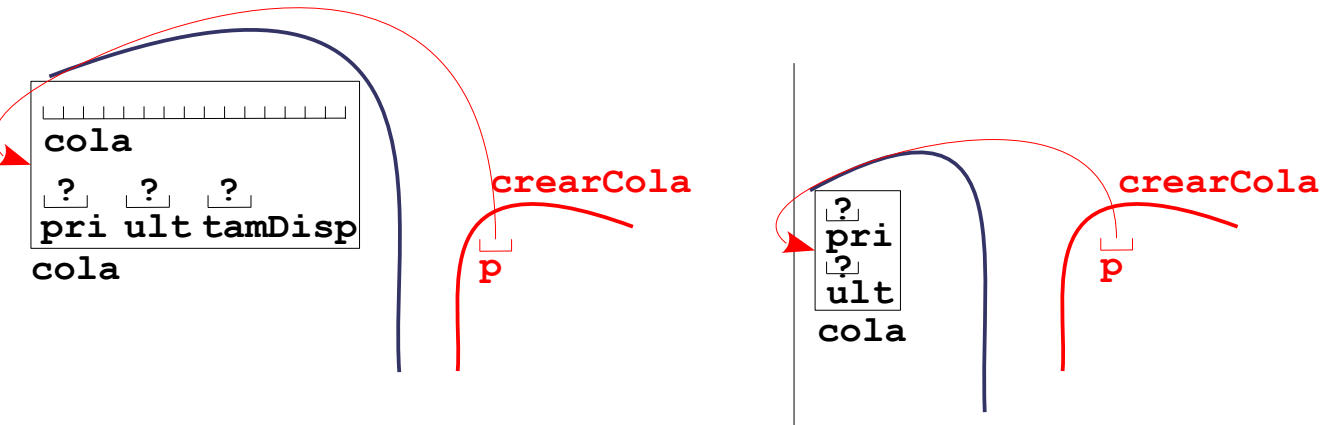
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
91
92
93
94
95
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



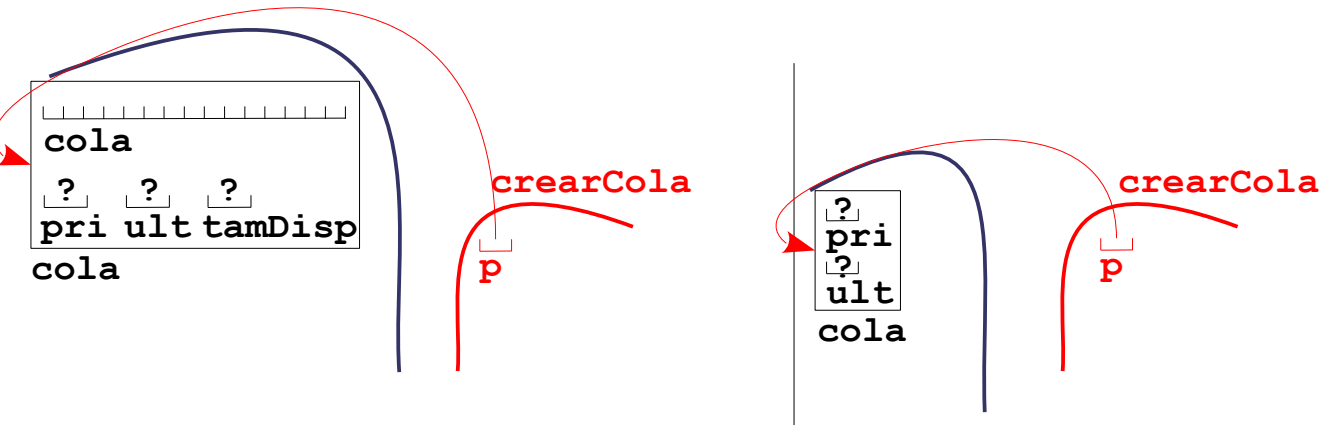
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
91
92     crearCola(&cola);
93
94
95
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



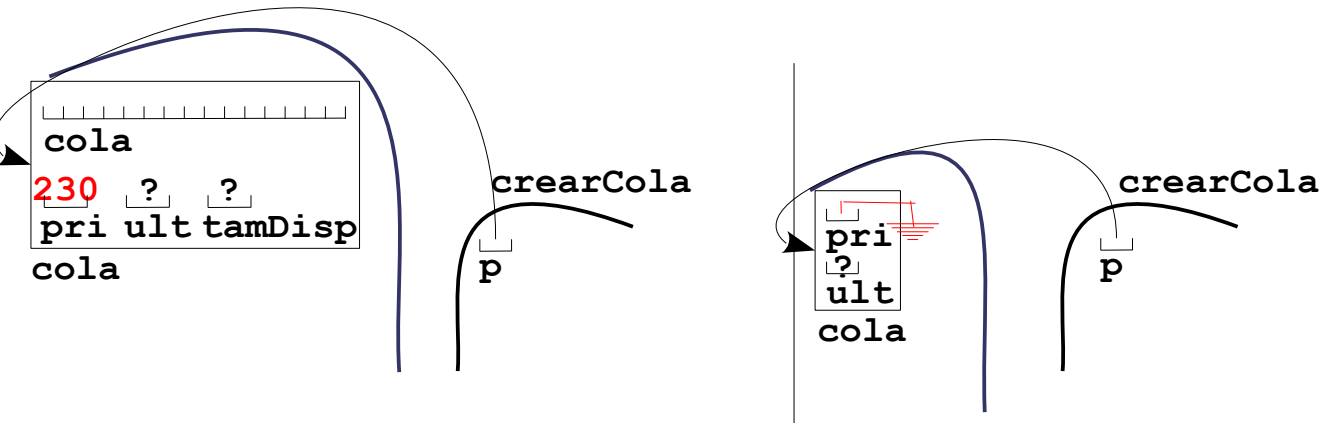
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
97 void probarOperarCola(void)
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
10 void crearCola(tCola *p)
11 {
12     p->pri = TAM_COLA - 70;
13     p->ult = TAM_COLA - 70;
14     p->tamDisp = TAM_COLA;
15 }
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
6 void crearCola(tCola *p)
7 {
8     p->pri = NULL;
9     p->ult = NULL;
10 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



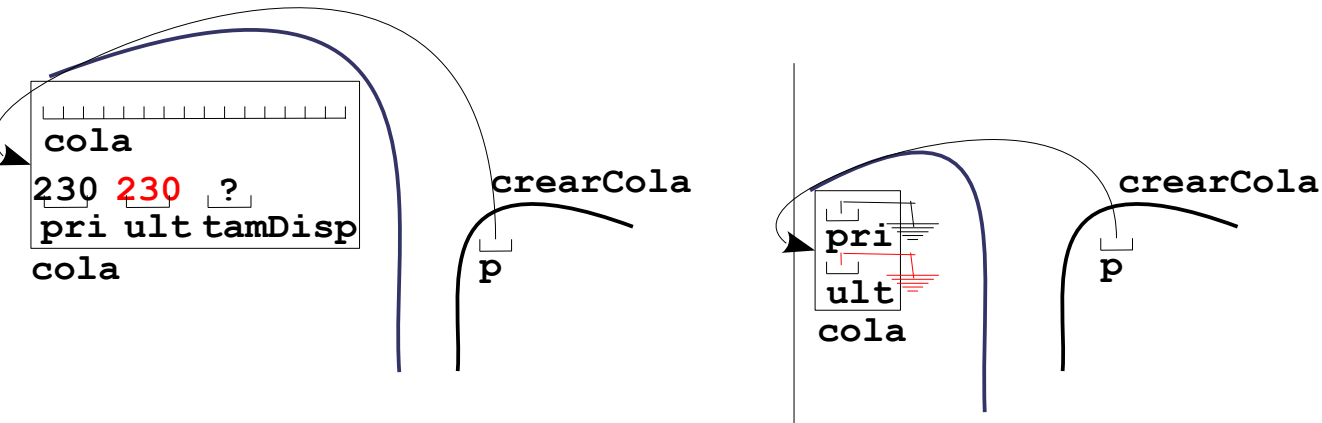
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
97 void probarDeclararYCrearDeCola(void)
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
10 void crearCola(tCola *p)
11 {
12     p->pri = TAM_COLA - 70;
13     p->ult = TAM_COLA - 70;
14     p->tamDisp = TAM_COLA;
15 }
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
6 void crearCola(tCola *p)
7 {
8     p->pri = NULL;
9     p->ult = NULL;
10 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



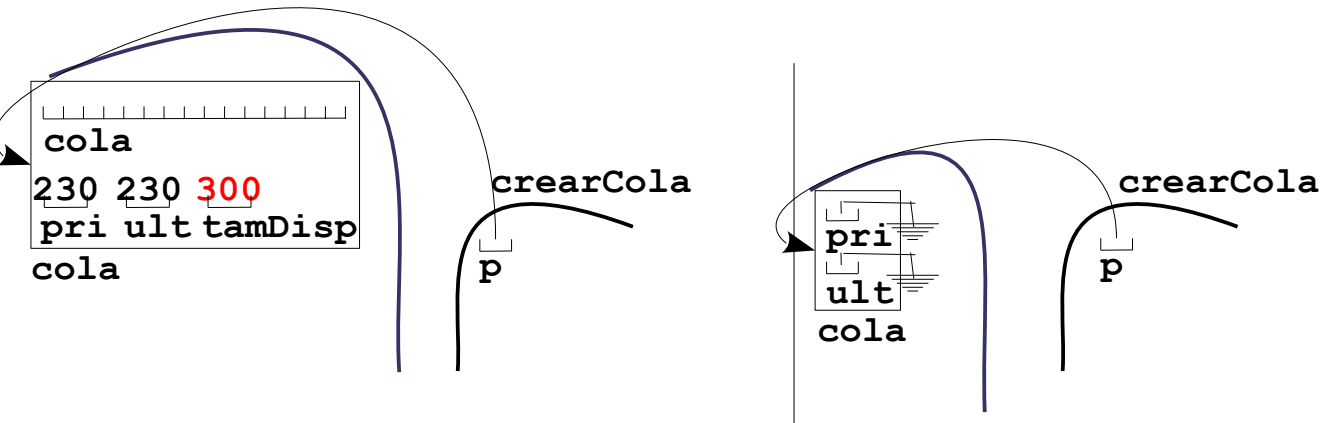
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
97 void probarDeclararYCrearDeCola(void)
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
10 void crearCola(tCola *p)
11 {
12     p->pri = TAM_COLA - 70;
13     p->ult = TAM_COLA - 70;
14     p->tamDisp = TAM_COLA;
15 }
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
6 void crearCola(tCola *p)
7 {
8     p->pri = NULL;
9     p->ult = NULL;
10 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



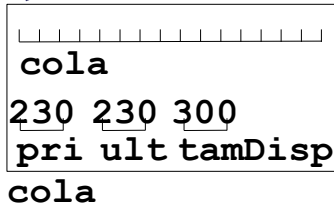
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
97 void probarOperarCola(void)
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
10 void crearCola(tCola *p)
11 {
12     p->pri = TAM_COLA - 70;
13     p->ult = TAM_COLA - 70;
14     p->tamDisp = TAM_COLA;
15 }
main.c x main.h x productos.c x productos.h x cola.c x cola.h x
6 void crearCola(tCola *p)
7 {
8     p->pri = NULL;
9     p->ult = NULL;
10 }
```

Tipos de Datos Abstractos

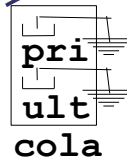
Tipo de dato Cola.

Implementación estática

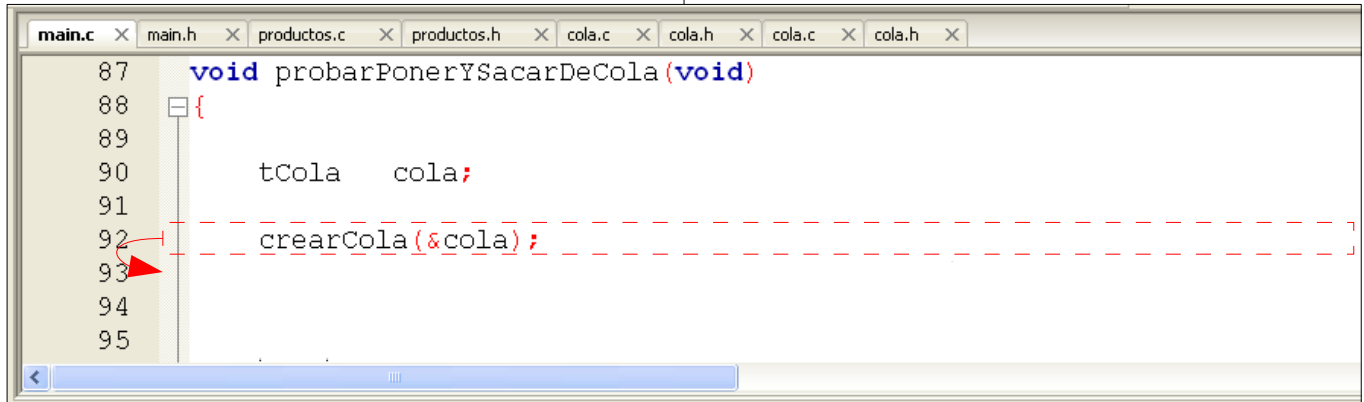
Implementación dinámica



A diagram illustrating a static queue implementation. It shows a horizontal array of 10 slots. Below the array, the variable `cola` is declared. Below that, the values `230`, `230`, and `300` are shown, corresponding to the first three slots of the array. Below these values, the variables `pri`, `ult`, and `tamDisp` are listed. Finally, the variable `cola` is declared again at the bottom.



A diagram illustrating a dynamic queue implementation. It shows a vertical stack of three boxes. The top box contains the variable `pri`, the middle box contains `ult`, and the bottom box contains `cola`. Arrows indicate the flow of data between these variables.



A screenshot of a code editor window showing the implementation of a queue. The editor has several tabs open: `main.c`, `main.h`, `productos.c`, `productos.h`, `cola.c`, `cola.h`, `cola.c`, and `cola.h`. The code in the `main.c` tab is as follows:

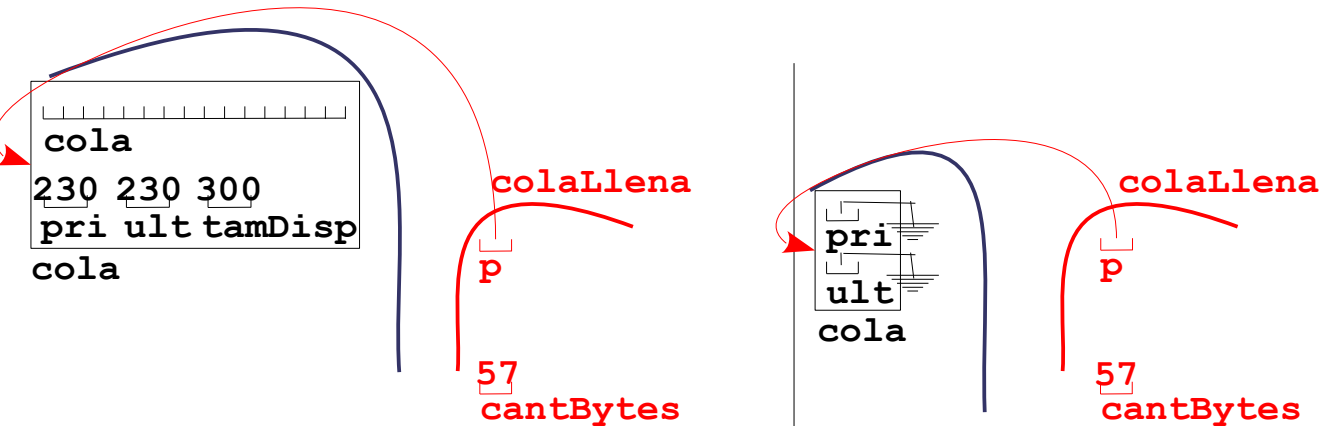
```
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
91
92     crearCola(&cola);
93
94
95
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



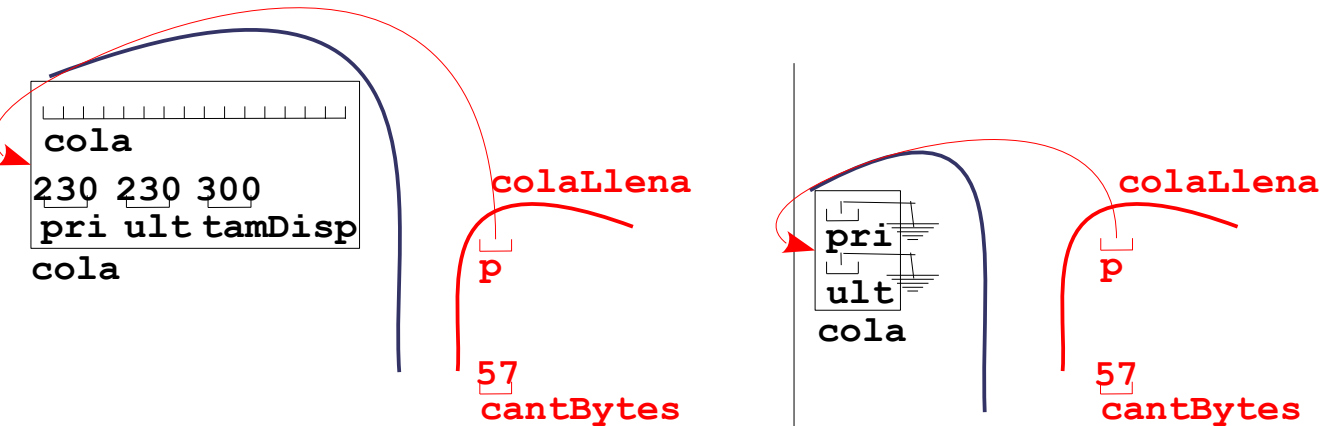
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) &&
94
95
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {

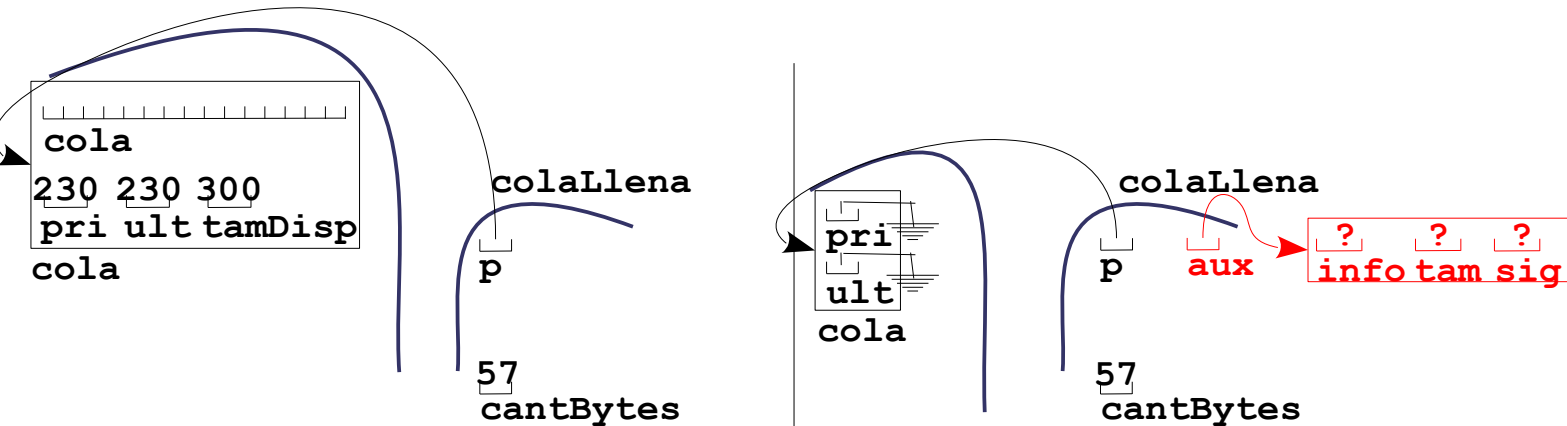
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp < cantBytes + sizeof(unsigned);
20 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
```

```
main.c x main.h x productos.c x productos.h
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp < cantBytes;
20 }
```

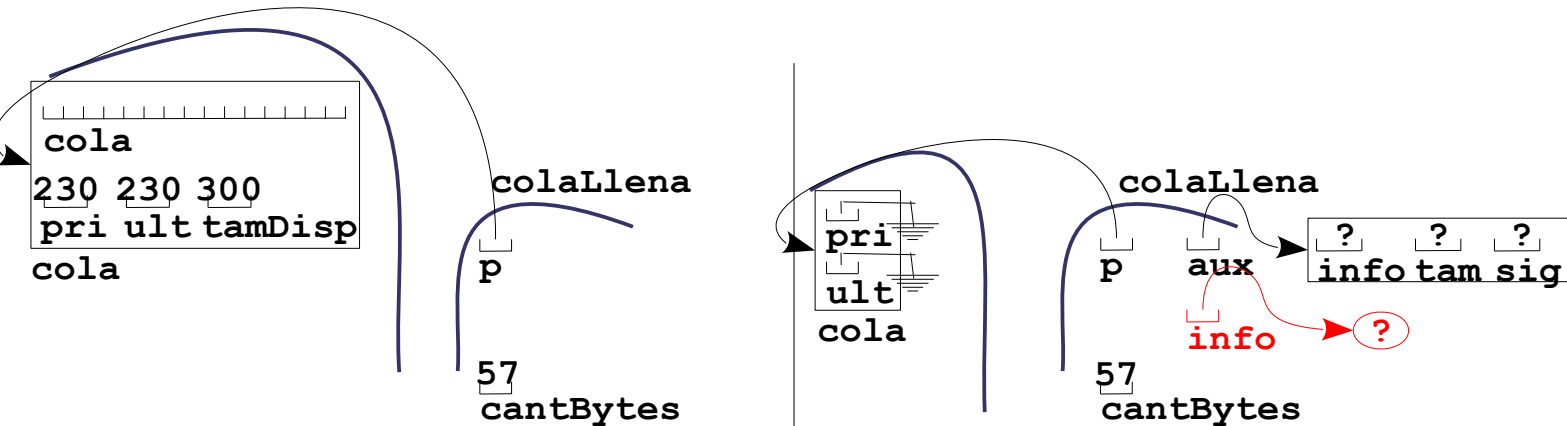
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
```

```
main.c x main.h x productos.c x productos.h
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp == cantBytes;
20 }
```

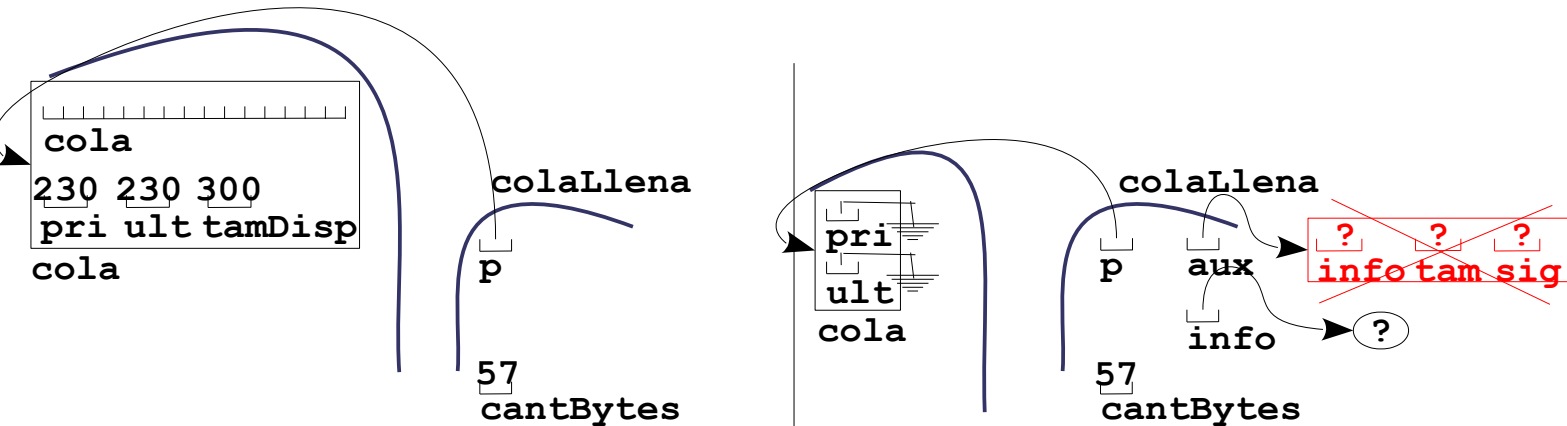
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
```

```
main.c x main.h x productos.c x productos.h
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp;
20 }
```

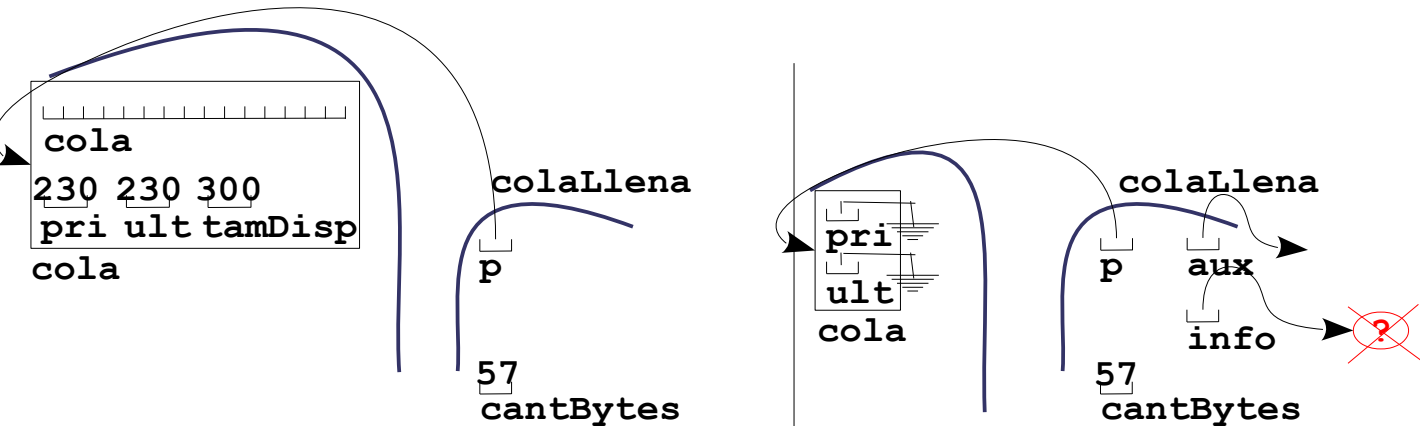
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
```

```
main.c x main.h x productos.c x productos.h
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp == cantBytes;
20 }
```

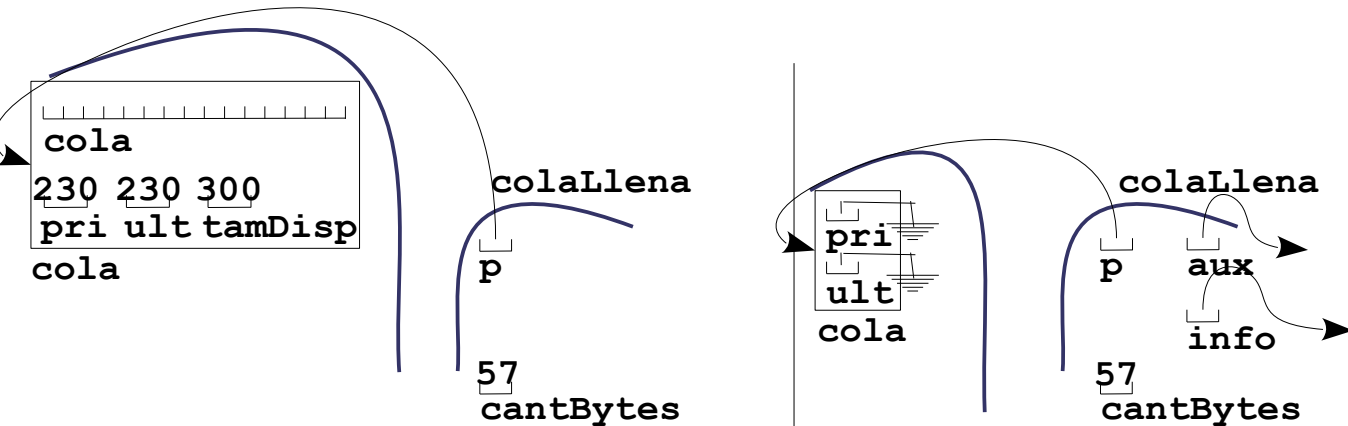
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;
```

```
main.c x main.h x productos.c x productos.h
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp == cantBytes;
20 }
```

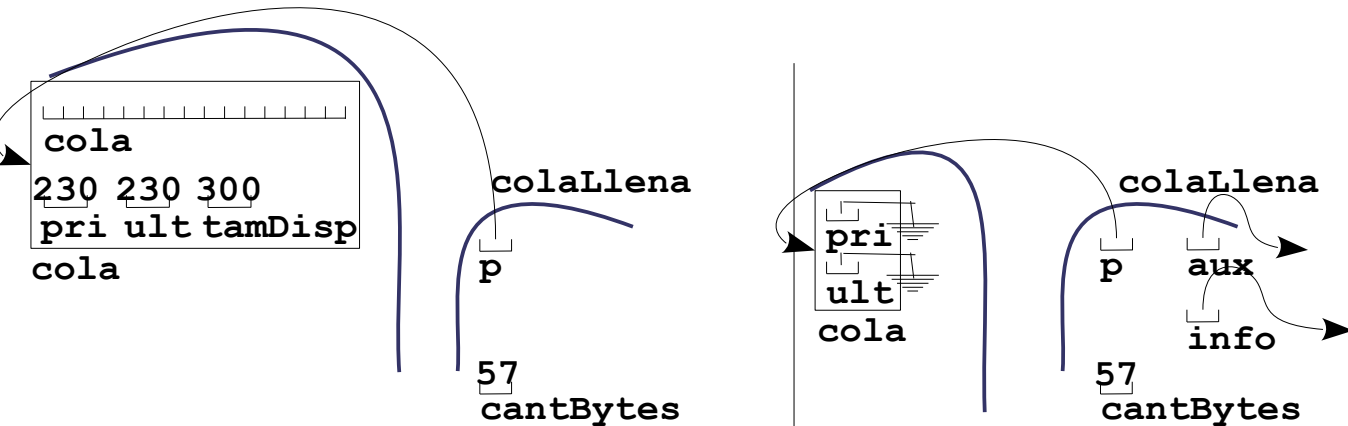
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89
90     tCola cola;

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {

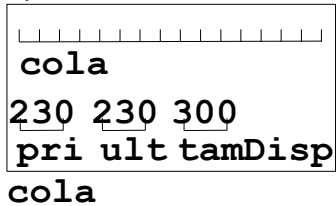
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp < cantBytes + sizeof(unsigned);
20 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

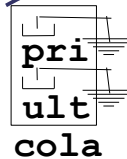
Implementación estática

Implementación dinámica



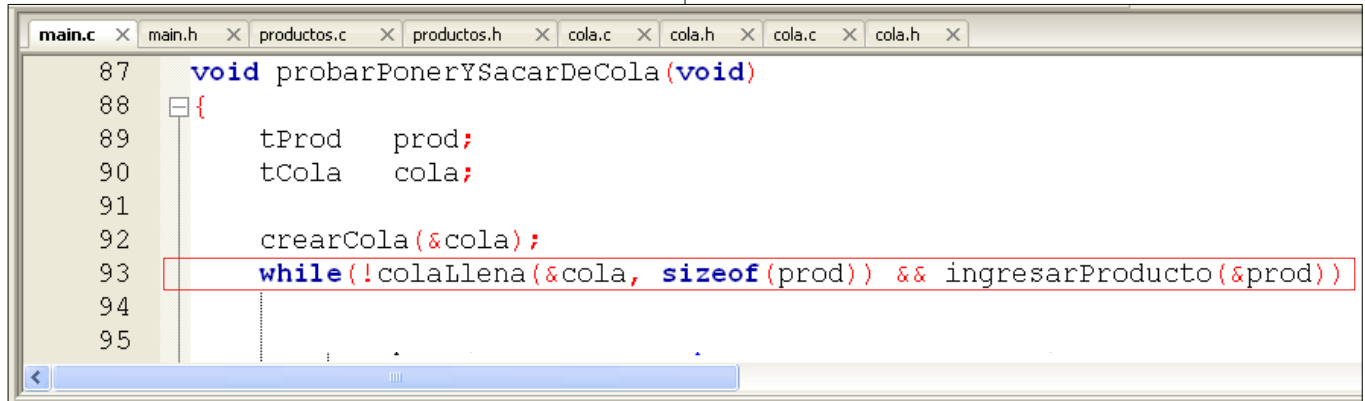
A diagram illustrating a static queue implementation. It shows a horizontal array of 10 slots. The first three slots are labeled 'cola', '230', and '230'. The next three slots are labeled '300', 'pri', and 'ult'. The last four slots are labeled 'tamDisp', 'cola', and two empty slots. A blue curved arrow points from the 'cola' label to the first slot.

```
cola  
230 230 300  
pri ult tamDisp  
cola
```



A diagram illustrating a dynamic queue implementation. It shows a vertical stack of three boxes labeled 'pri', 'ult', and 'cola'. A blue curved arrow points from the 'cola' label to the bottom box.

```
pri  
ult  
cola
```



A screenshot of a code editor window showing the implementation of a queue. The editor has several tabs open: main.c, main.h, productos.c, productos.h, cola.c, cola.h, cola.c, and cola.h. The code is in C and defines a function 'probarPonerYSacarDeCola' that takes a 'void' argument. The function body includes declarations for 'tProd prod;' and 'tCola cola;', a call to 'crearCola(&cola);', and a 'while' loop that continues as long as the queue is not full and a product is entered. The 'while' loop condition is highlighted with a red box.

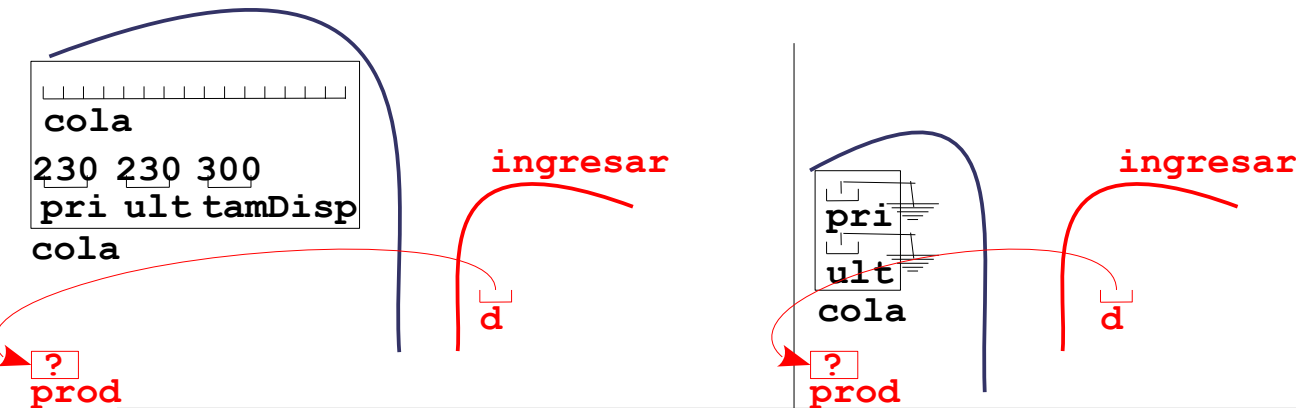
```
87 void probarPonerYSacarDeCola(void)  
88 {  
89     tProd  prod;  
90     tCola  cola;  
91  
92     crearCola(&cola);  
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))  
94  
95
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



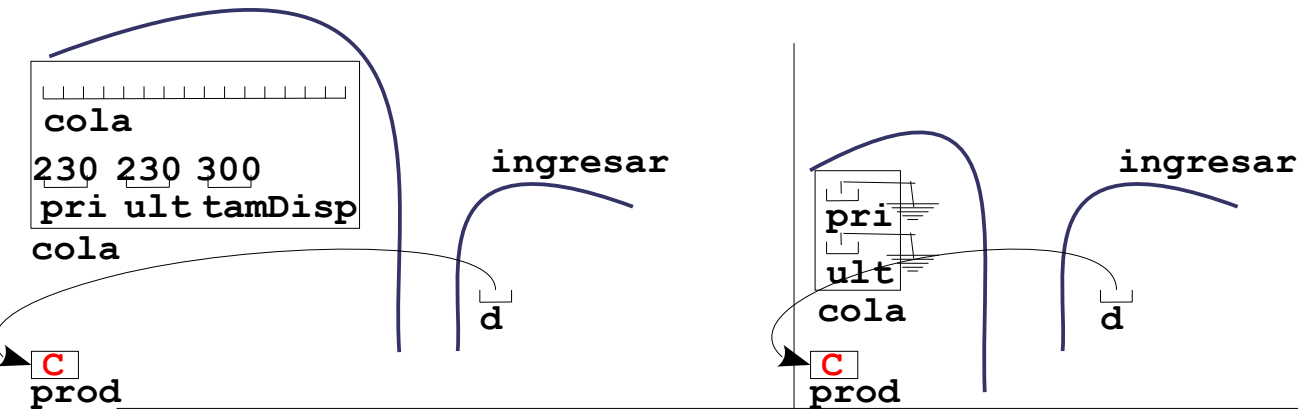
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94
95
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



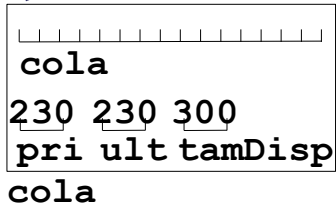
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94
95
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

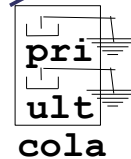
Implementación dinámica



A diagram showing a static queue implementation. It consists of an array of 10 slots, with the first three slots containing the values 230, 230, and 300. Below the array, the variables 'pri' and 'ult' are shown pointing to the first and third slots respectively. The variable 'tamDisp' is shown pointing to the third slot. The variable 'cola' is shown pointing to the first slot.

```
cola
230 230 300
pri ult tamDisp
cola
```

C
prod



A diagram showing a dynamic queue implementation. It consists of a linked list structure with three nodes. The first node contains the value 230, the second node contains 230, and the third node contains 300. The variable 'pri' points to the first node, 'ult' points to the third node, and 'cola' points to the first node.

```
pri
ult
cola
```

C
prod

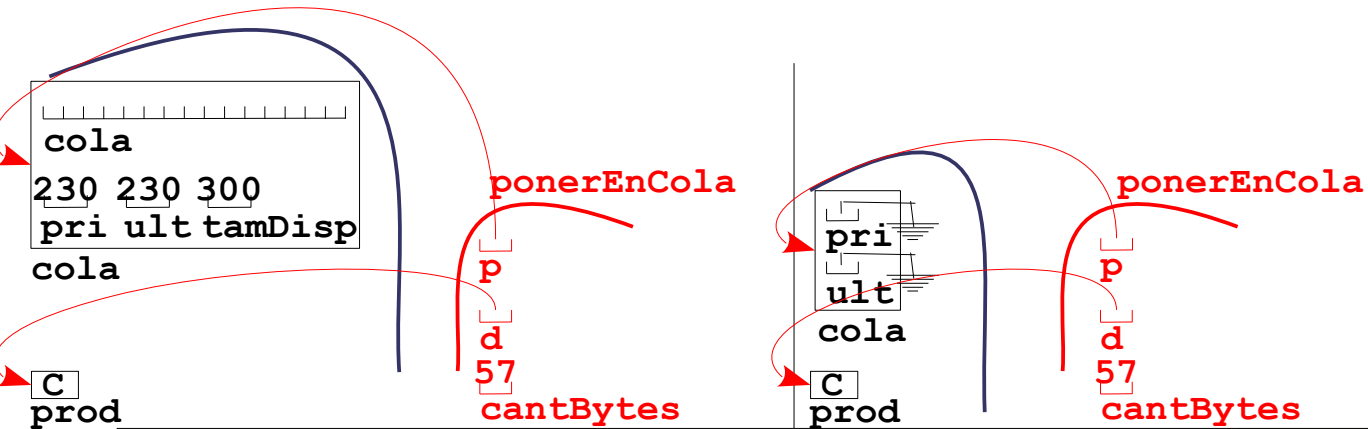
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



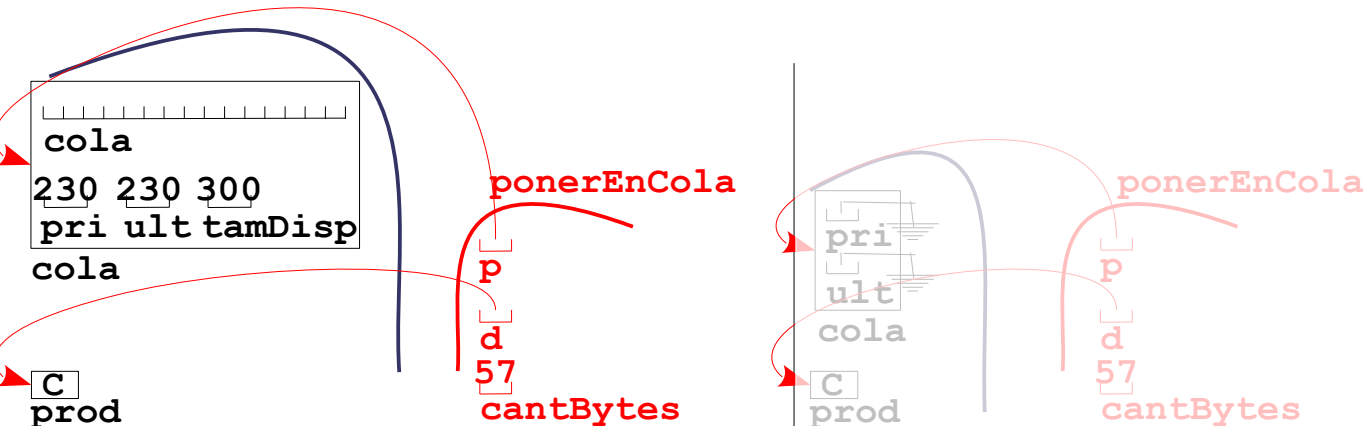
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



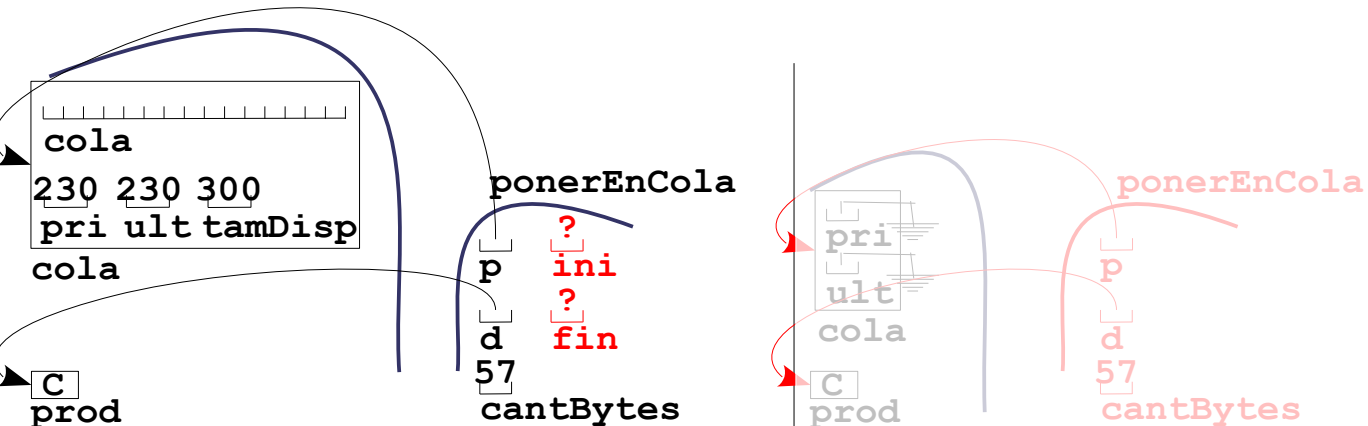
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



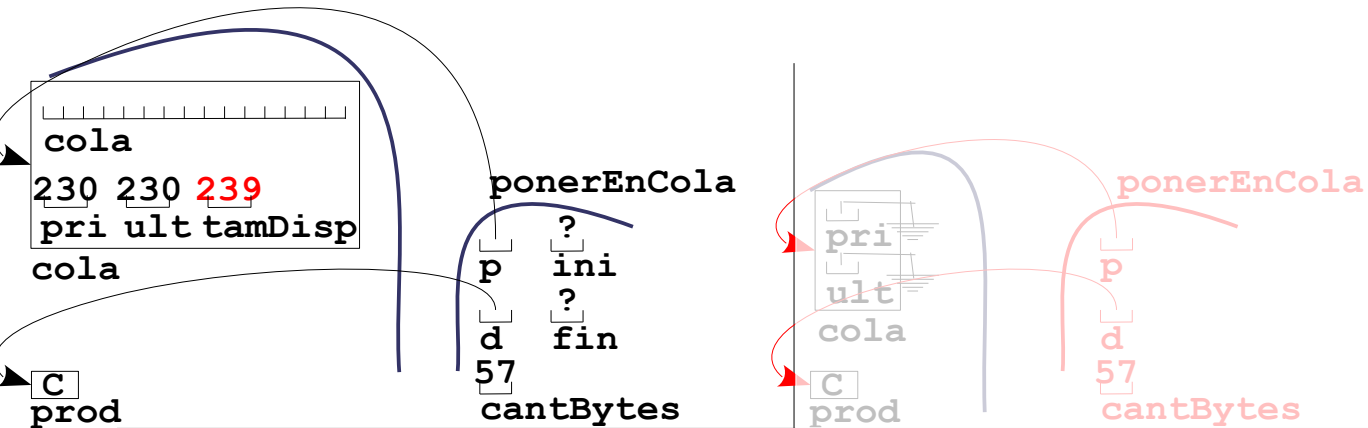
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



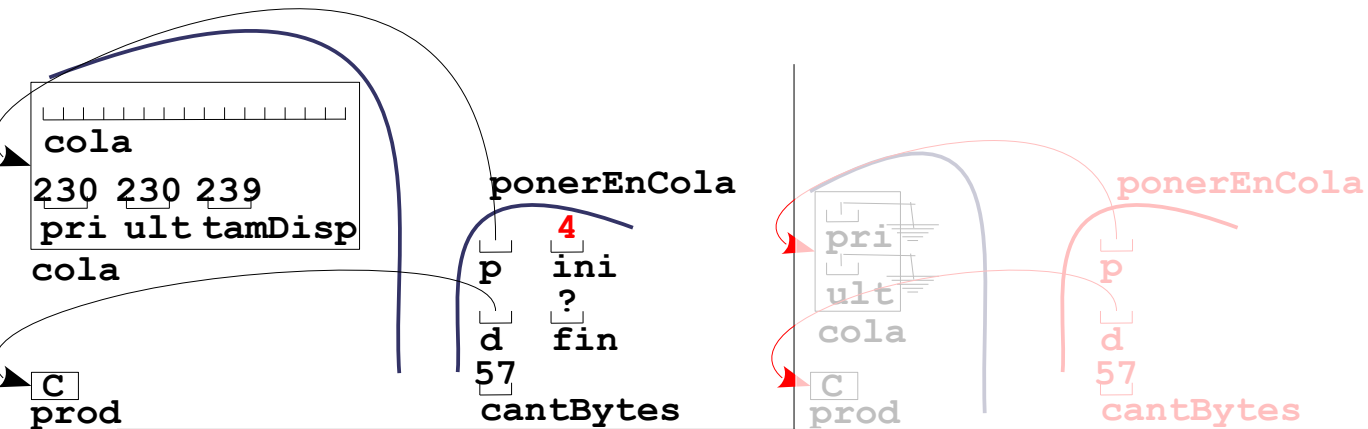
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



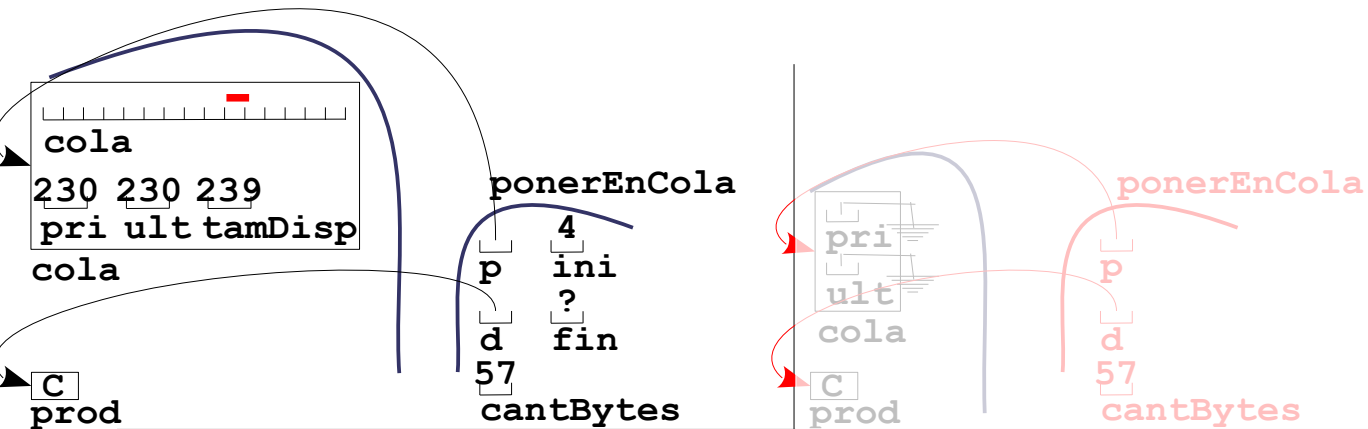
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



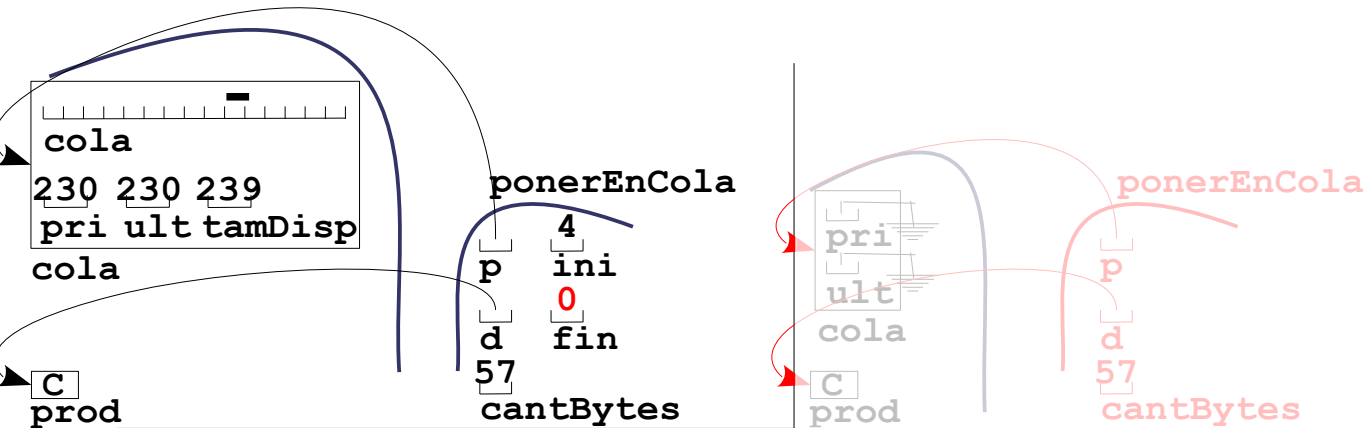
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                 fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

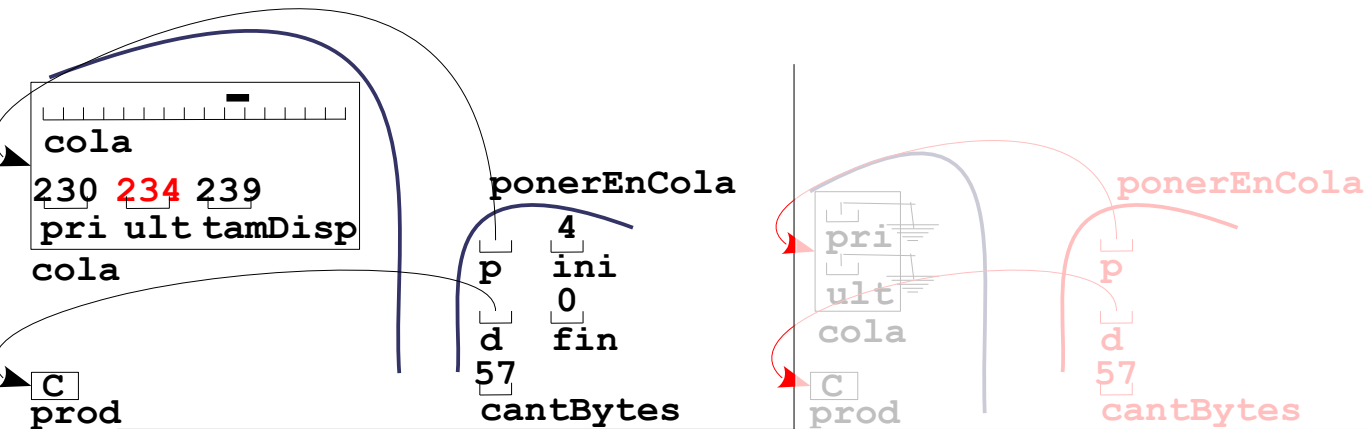
to(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



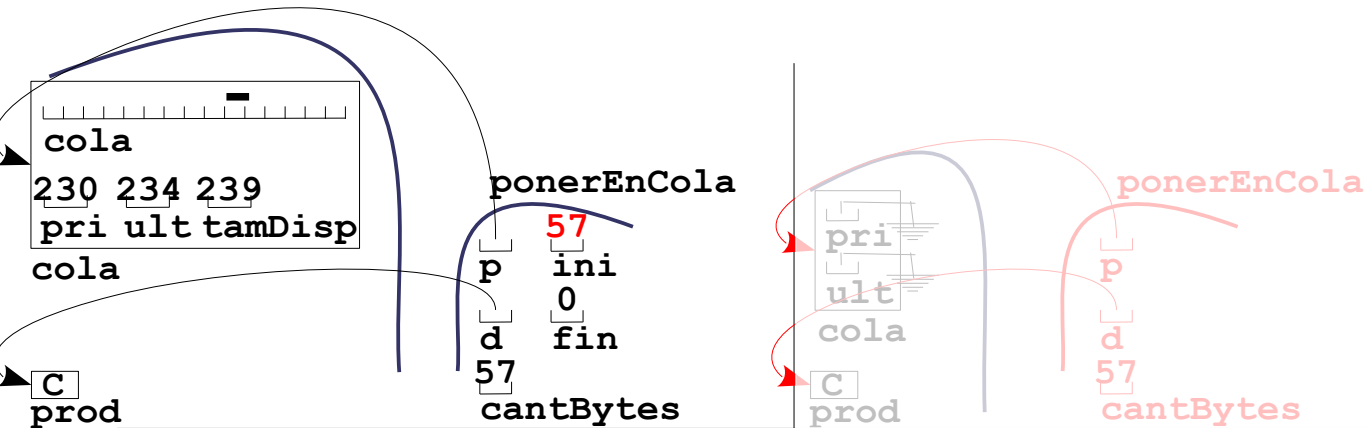
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



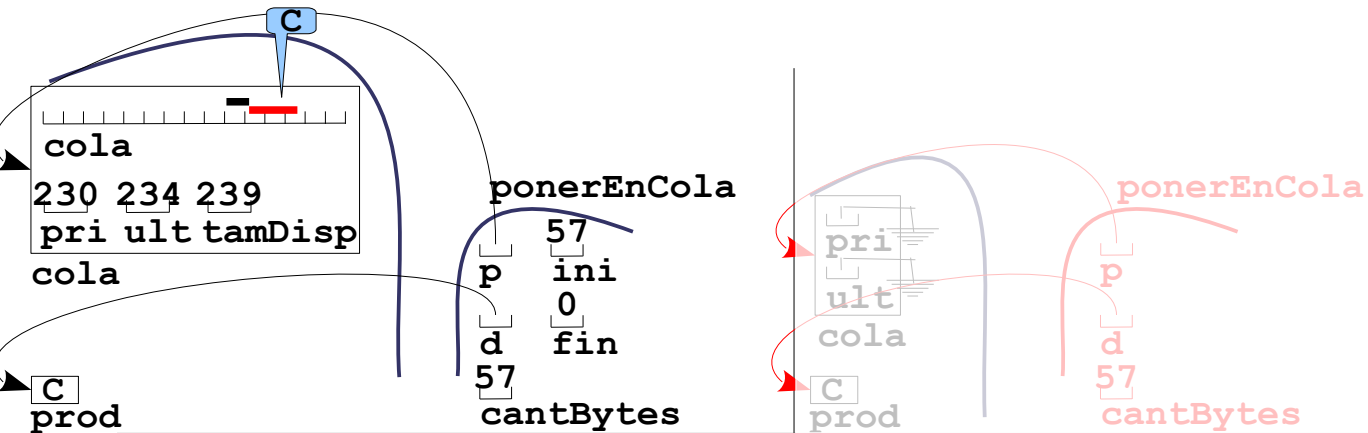
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

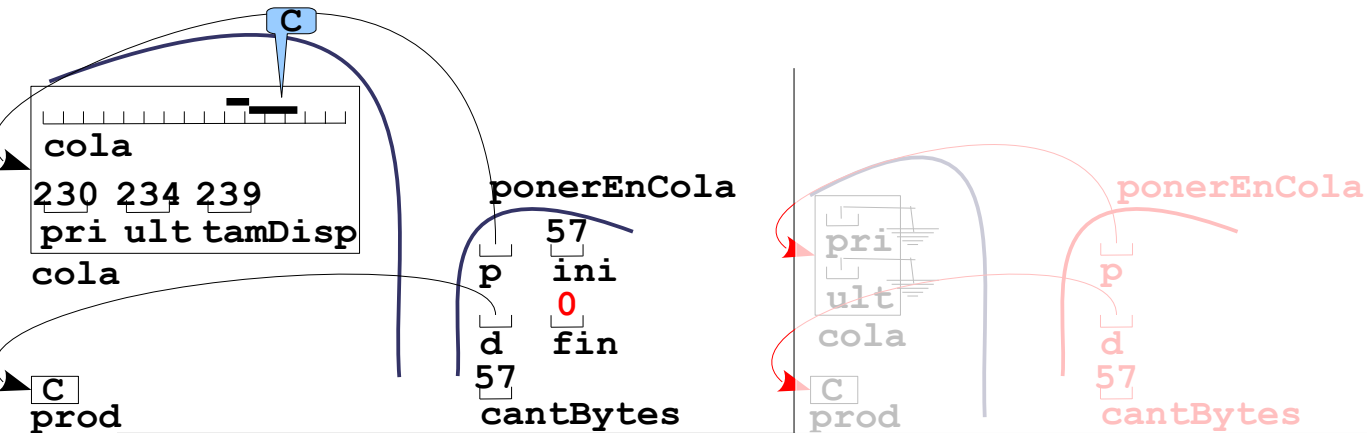
to(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



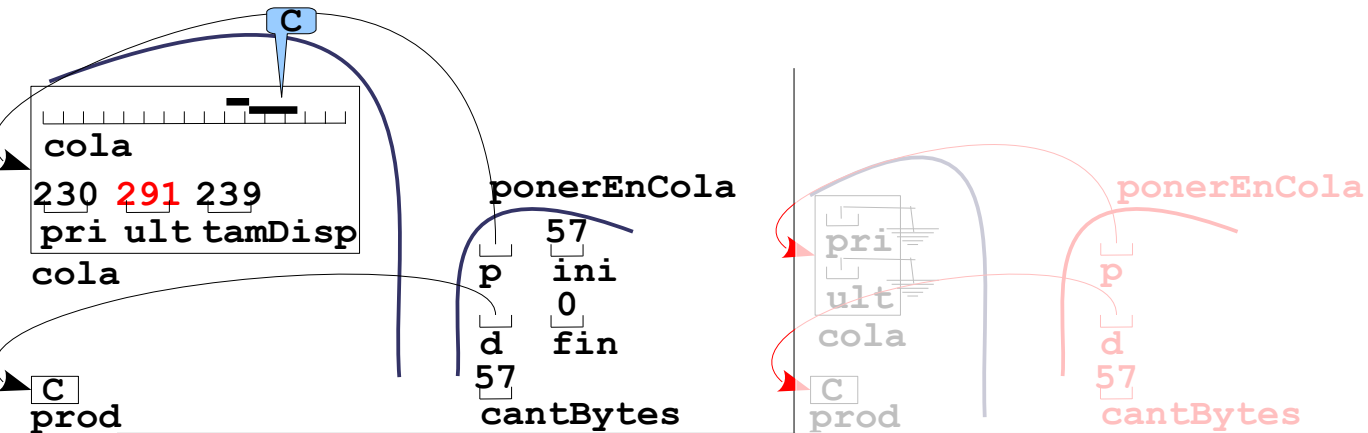
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
```

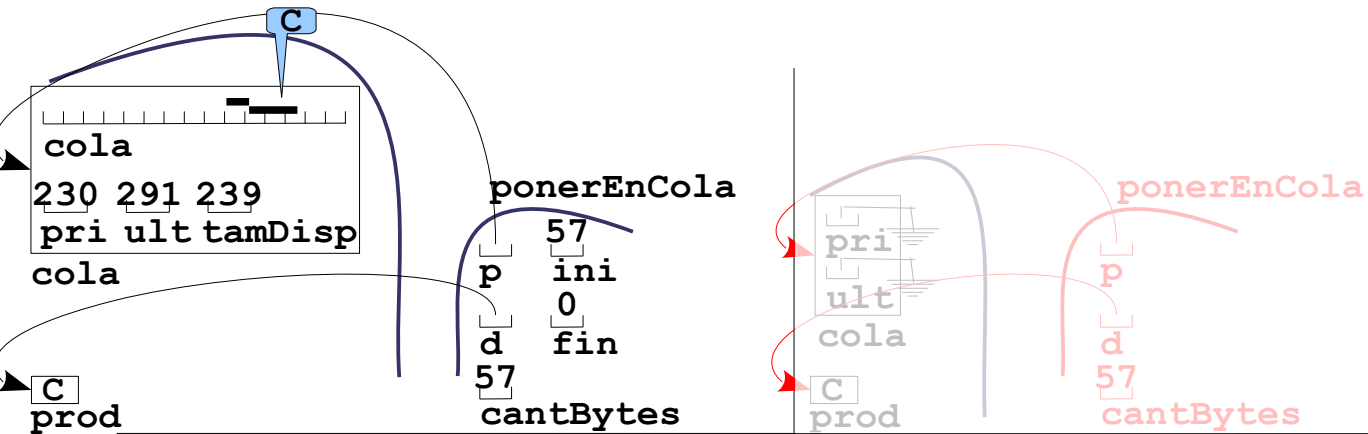
to(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



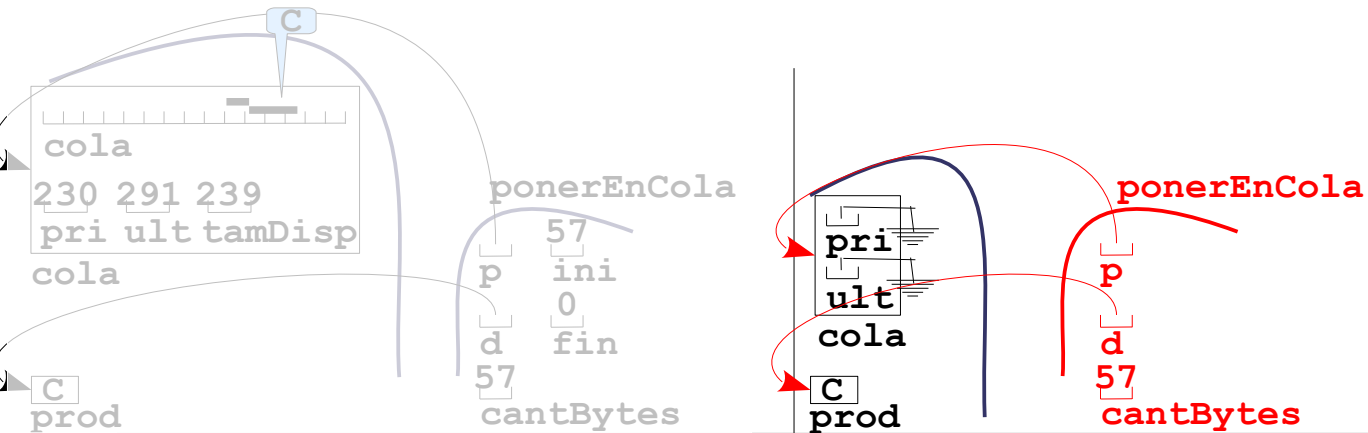
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



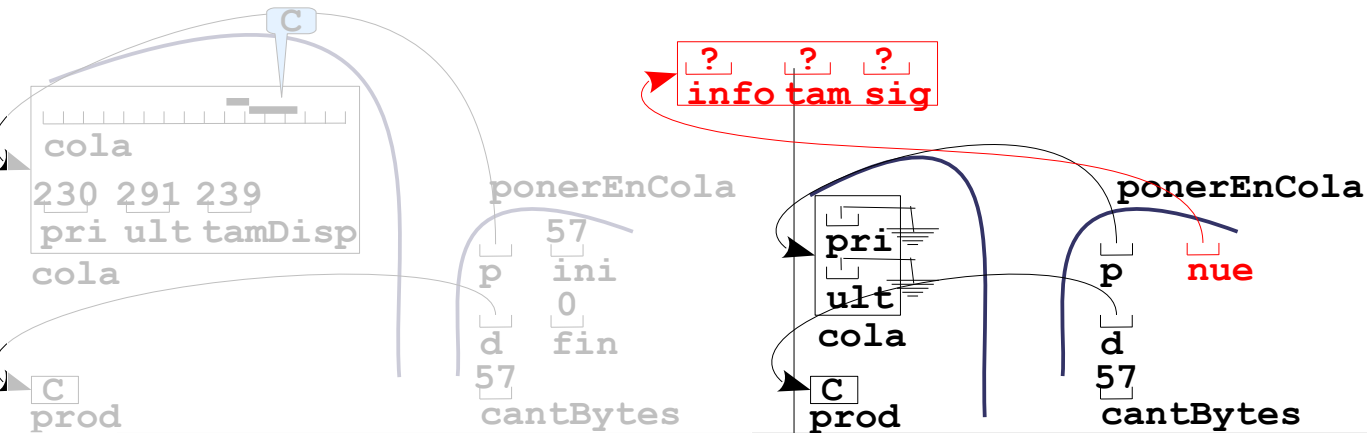
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void
88 {
89
90
91
92
93
94
95
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
31     return 1;
32 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



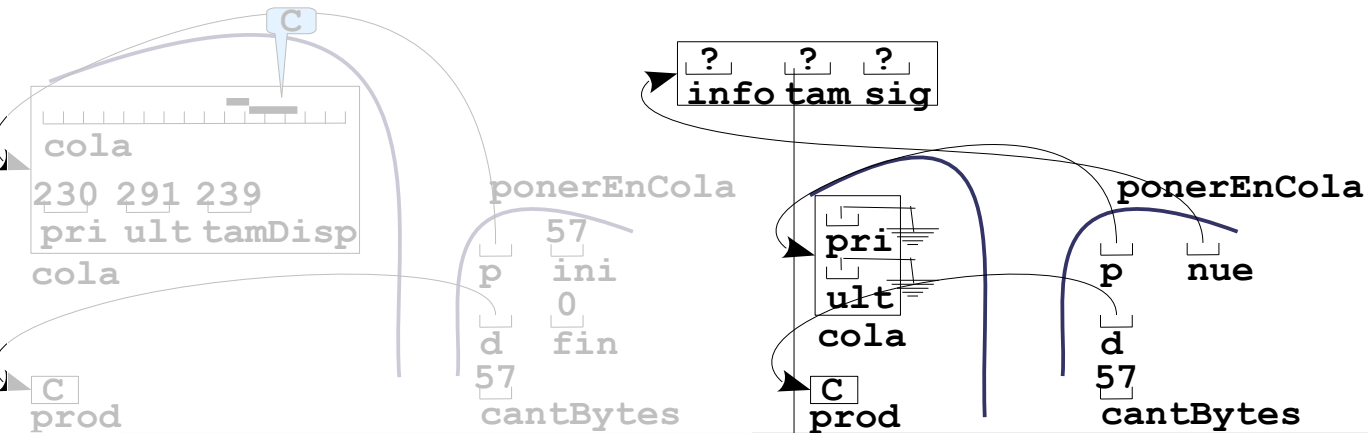
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87  void
88  {
89
90
91
92
93
94
95
21  int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22  {
23      tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25      if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26      {
27          free(nue);
28          return 0;
29      }
30      memcpy(nue->info, d, cantBytes);
31
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo
88 {
89
90
91
92
93
94
95

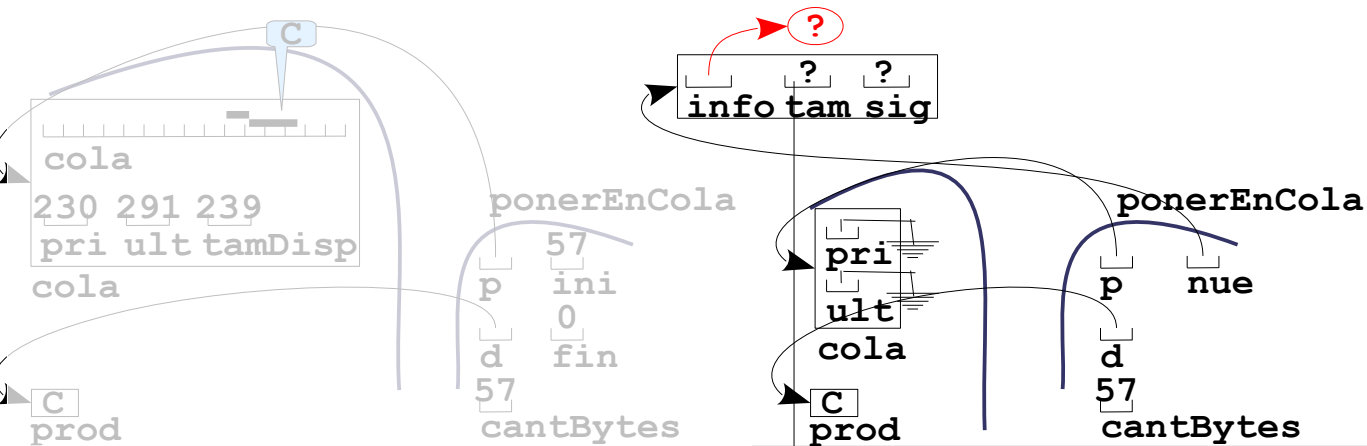
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
31     return 1;
32 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo
88 {
89
90
91
92
93
94
95

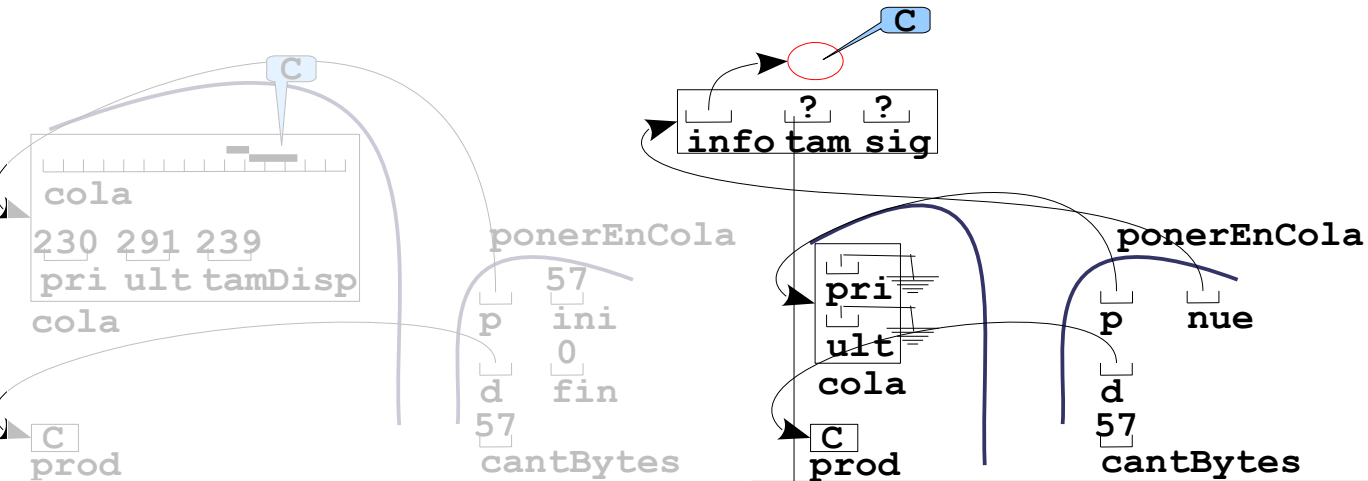
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87  vo:
88  {
89
90
91
92
93
94
95

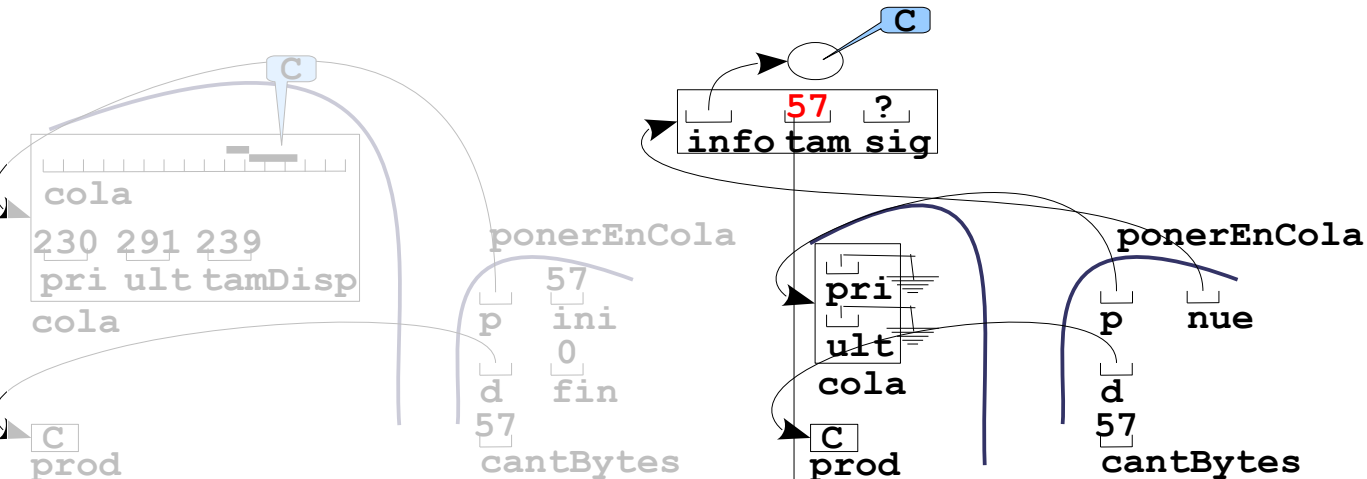
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21  int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22  {
23      tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25      if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26      {
27          free(nue);
28          return 0;
29      }
30      memcpy(nue->info, d, cantBytes);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



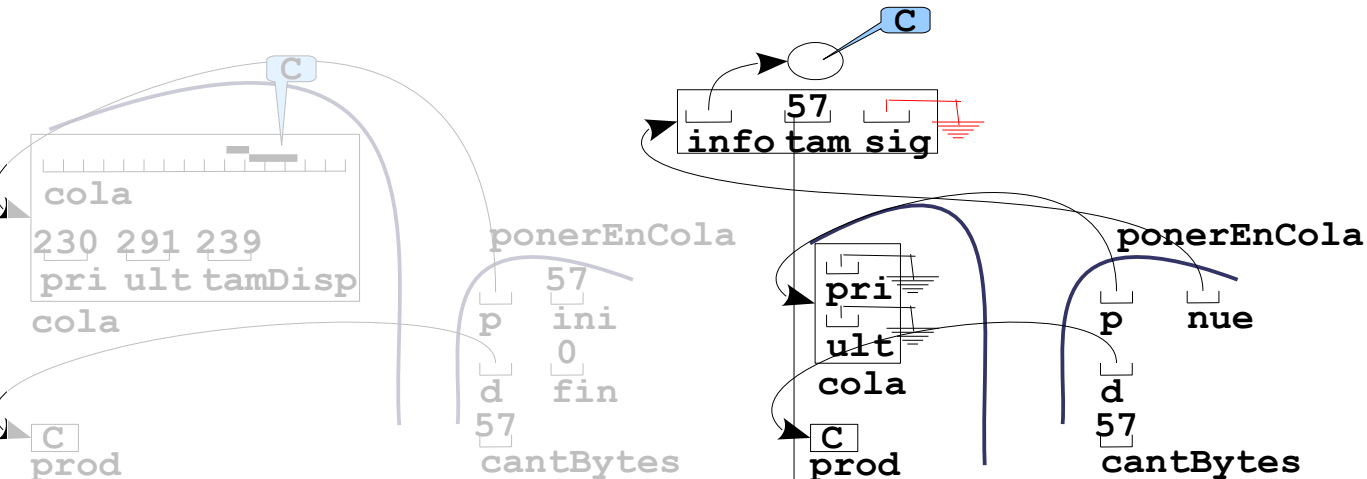
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40
98 41 return 1;
99 42 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



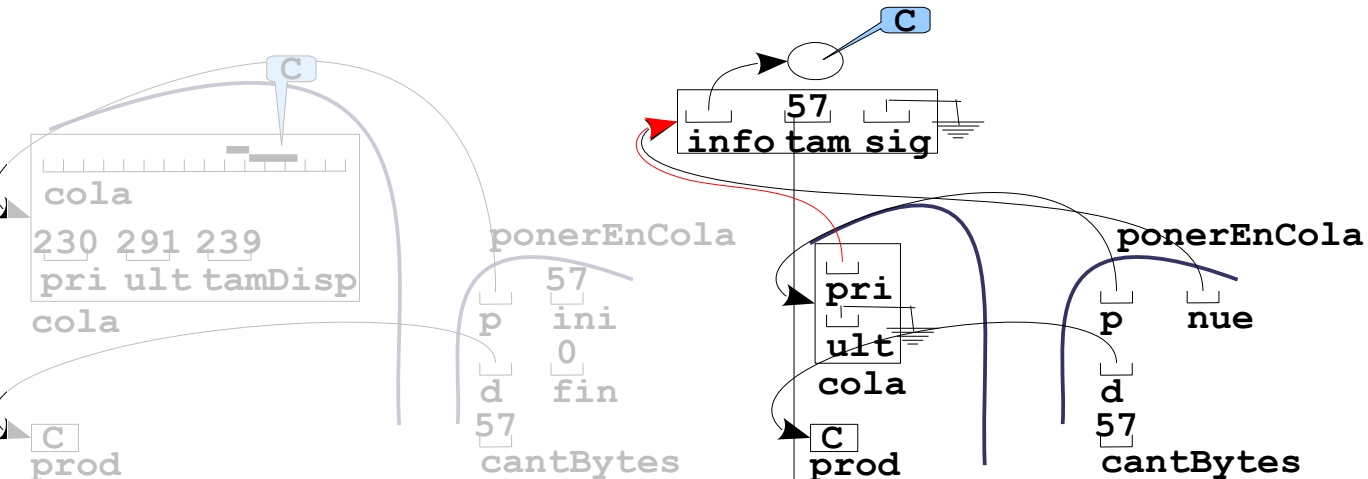
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void
88
89
90
91
92
93
94
95
31
32
33
34
35
36
37
38
39
nue->tamInfo = cantBytes;
nue->sig = NULL;
if(p->ult)
    p->ult->sig = nue;
else
    p->pri = nue;
p->ult = nue;
return 1;
}
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



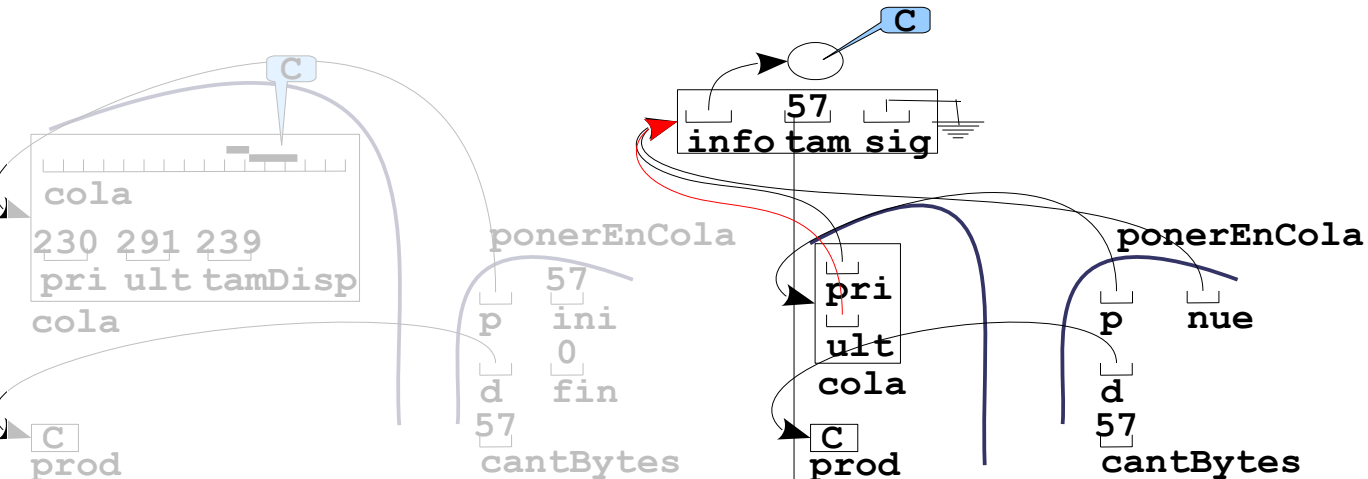
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36 else
94 37     p->pri = nue;
95 38
96 39 return 1;
97 40 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



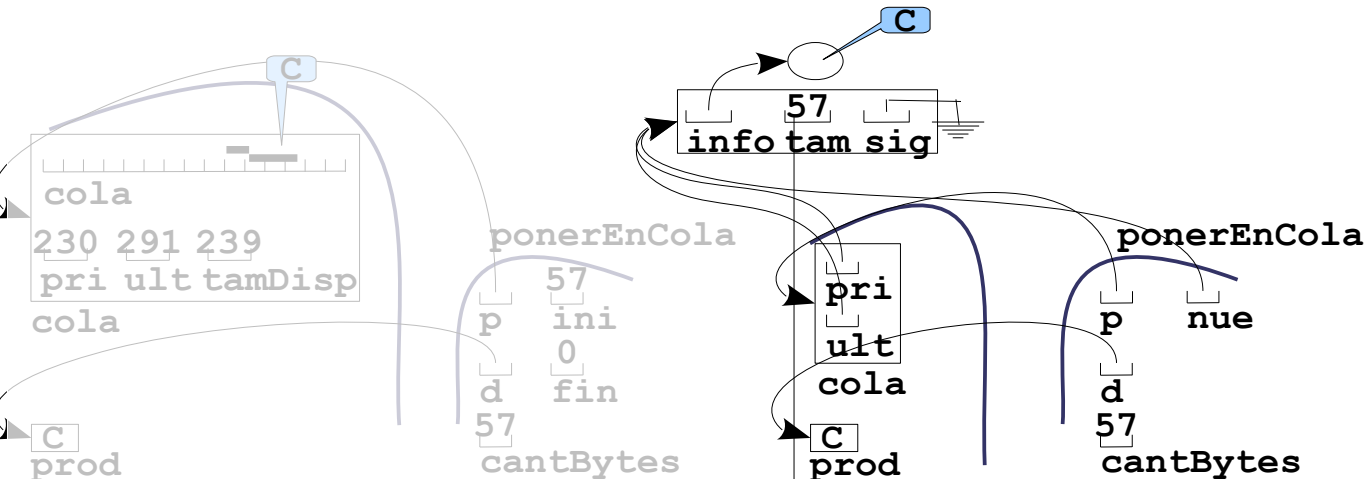
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40     return 1;
98 41 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

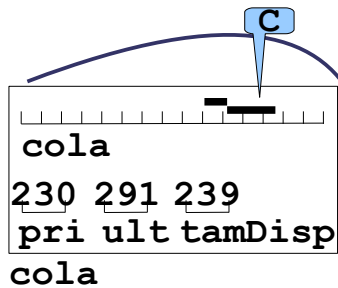


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40
98 41 return 1;
99 42 }
```

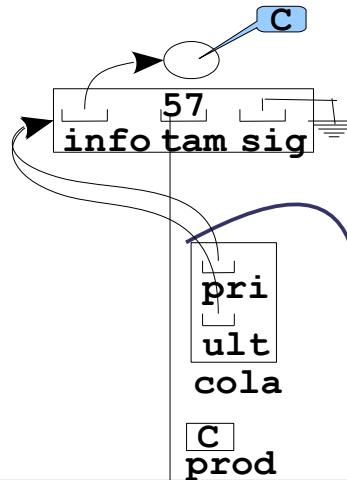
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



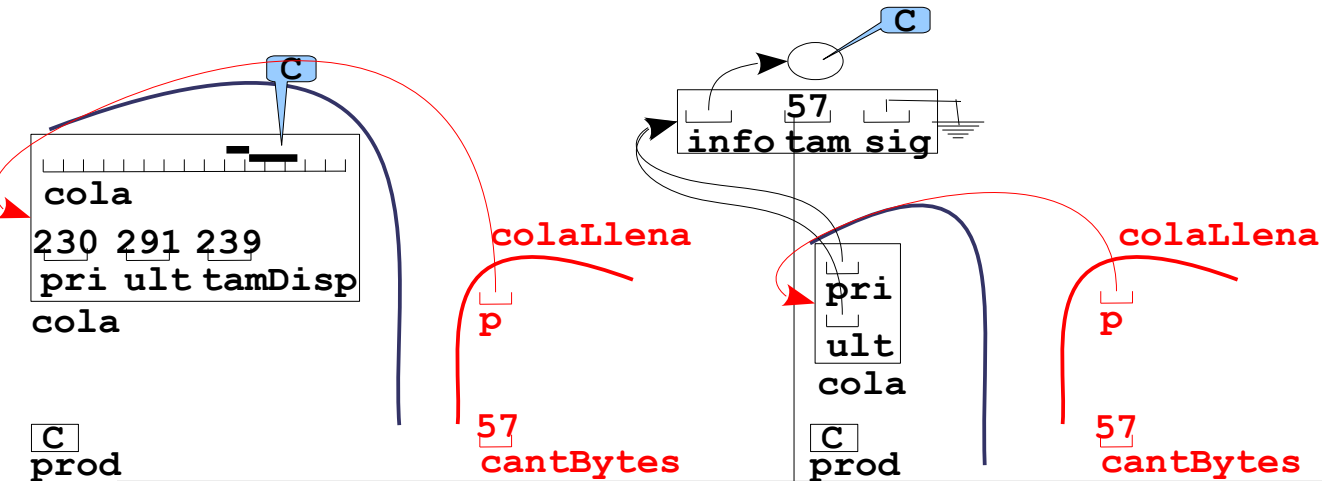
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



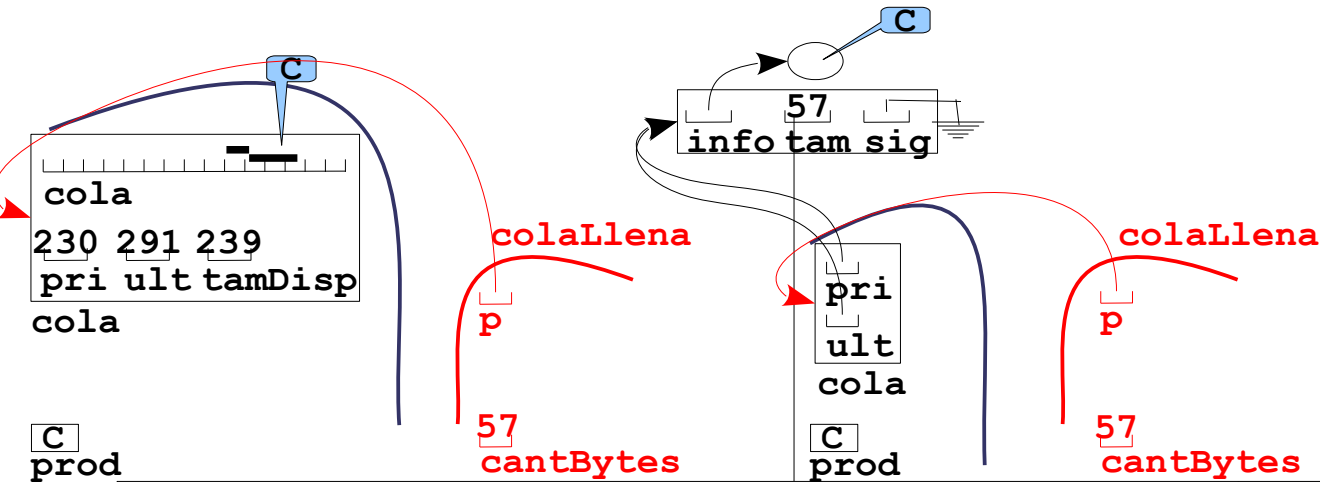
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while (!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if (!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd p;
90     tCola c;

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {

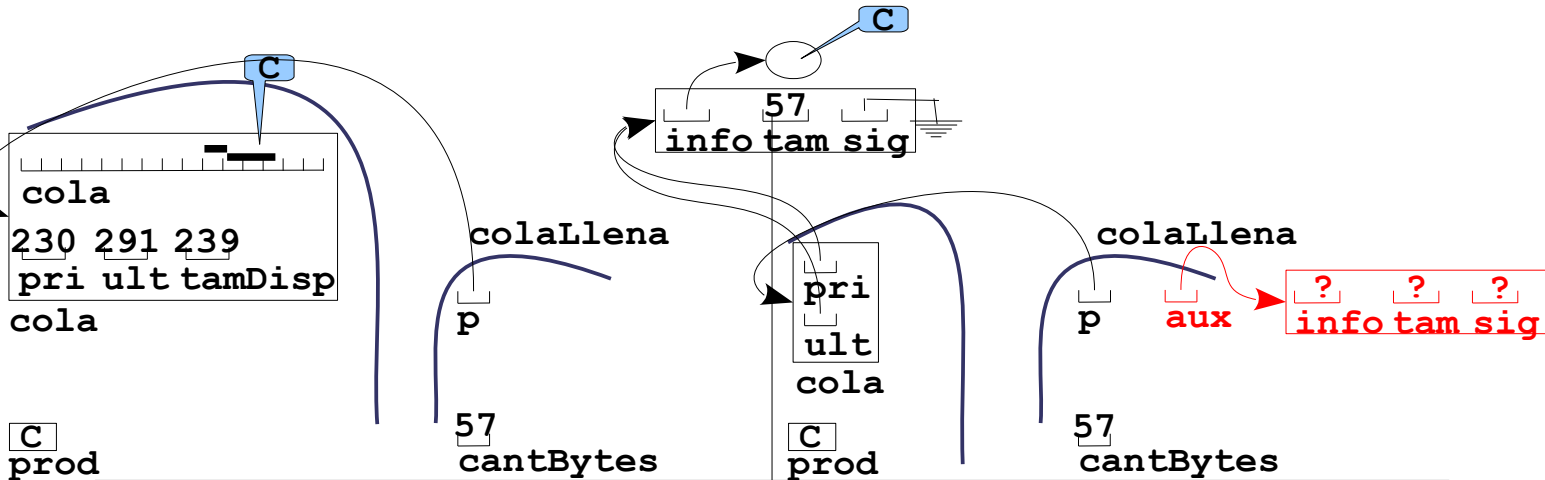
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp < cantBytes + sizeof(unsigned);
20 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
87 void probarPonerYSacarDeCola(void)
```

```
88 {
```

```
89     tProd p;
```

```
90     tCola c;
```

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
18 {
```

```
19     return p->tamDisp < cantBytes;
```

```
20 }
```

```
12 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
13 {
```

```
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
```

```
15     void *info = malloc(cantBytes);
```

```
16     free(aux);
```

```
17     free(info);
```

```
18     return aux == NULL || info == NULL;
```

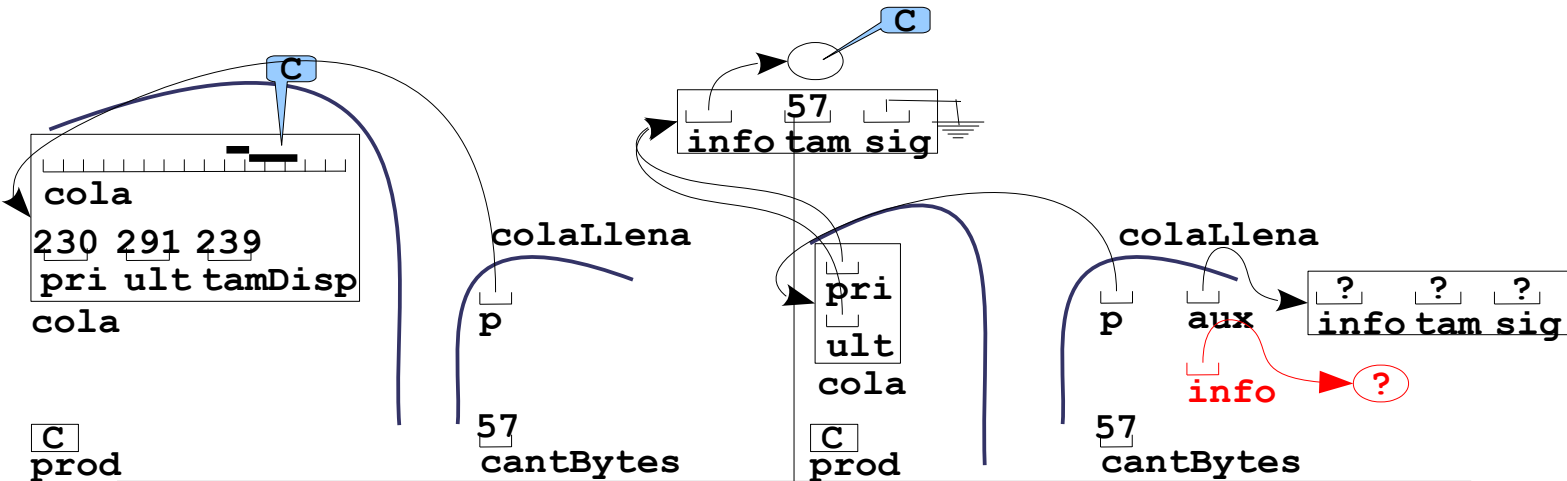
```
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
87 void probarPonerYSacarDeCola(void)
```

```
88 {
```

```
89     tProd p;
```

```
90     tCola c;
```

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
18 {
```

```
19     return p->tamDisponible < cantBytes;
```

```
20 }
```

```
12 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
13 {
```

```
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
```

```
15     void *info = malloc(cantBytes);
```

```
16     free(aux);
```

```
17     free(info);
```

```
18     return aux == NULL || info == NULL;
```

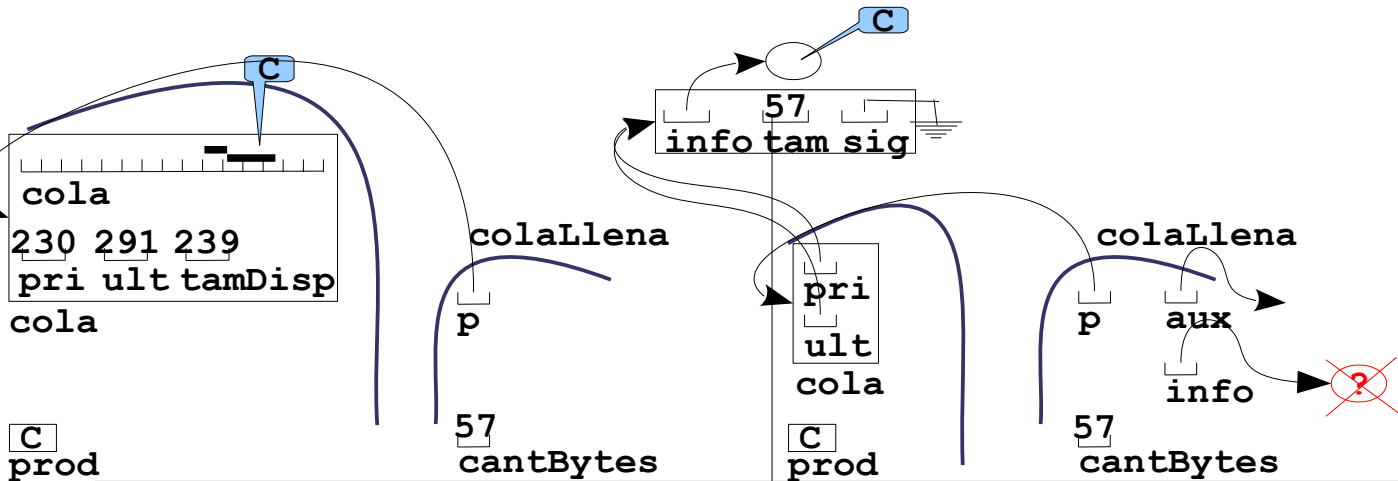
```
19 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x

```
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd p;
90     tCola c;
```

main.c x main.h x productos.c x productos.h

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp == cantBytes;
20 }
```

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x

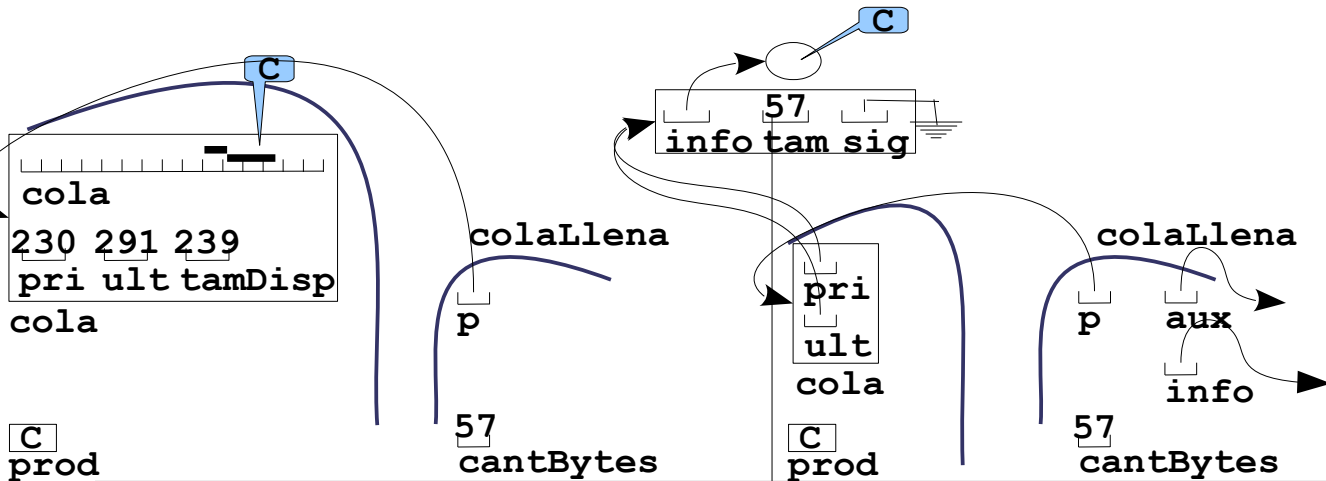
```
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
87 void probarPonerYSacarDeCola(void)
```

```
88 {
```

```
89     tProd p;
```

```
90     tCola c;
```

```
91 }
```

```
92 }
```

```
93 }
```

```
94 }
```

```
95 }
```

```
96 }
```

```
97 }
```

```
98 }
```

```
99 }
```

```
100 }
```

```
101 }
```

```
102 }
```

```
103 }
```

```
104 }
```

```
105 }
```

```
106 }
```

```
107 }
```

```
108 }
```

```
109 }
```

```
110 }
```

```
111 }
```

```
112 }
```

```
113 }
```

```
114 }
```

```
115 }
```

```
116 }
```

```
117 }
```

```
118 }
```

```
119 }
```

```
120 }
```

```
121 }
```

```
122 }
```

```
123 }
```

```
124 }
```

```
125 }
```

```
126 }
```

```
127 }
```

```
128 }
```

```
129 }
```

```
130 }
```

```
131 }
```

```
132 }
```

```
133 }
```

```
134 }
```

```
135 }
```

```
136 }
```

```
137 }
```

```
138 }
```

```
139 }
```

```
140 }
```

```
141 }
```

```
142 }
```

```
143 }
```

```
144 }
```

```
145 }
```

```
146 }
```

```
147 }
```

```
148 }
```

```
149 }
```

```
150 }
```

```
151 }
```

```
152 }
```

```
153 }
```

```
154 }
```

```
155 }
```

```
156 }
```

```
157 }
```

```
158 }
```

```
159 }
```

```
160 }
```

```
161 }
```

```
162 }
```

```
163 }
```

```
164 }
```

```
165 }
```

```
166 }
```

```
167 }
```

```
168 }
```

```
169 }
```

```
170 }
```

```
171 }
```

```
172 }
```

```
173 }
```

```
174 }
```

```
175 }
```

```
176 }
```

```
177 }
```

```
178 }
```

```
179 }
```

```
180 }
```

```
181 }
```

```
182 }
```

```
183 }
```

```
184 }
```

```
185 }
```

```
186 }
```

```
187 }
```

```
188 }
```

```
189 }
```

```
190 }
```

```
191 }
```

```
192 }
```

```
193 }
```

```
194 }
```

```
195 }
```

```
196 }
```

```
197 }
```

```
198 }
```

```
199 }
```

```
200 }
```

```
201 }
```

```
202 }
```

```
203 }
```

```
204 }
```

```
205 }
```

```
206 }
```

```
207 }
```

```
208 }
```

```
209 }
```

```
210 }
```

```
211 }
```

```
212 }
```

```
213 }
```

```
214 }
```

```
215 }
```

```
216 }
```

```
217 }
```

```
218 }
```

```
219 }
```

```
220 }
```

```
221 }
```

```
222 }
```

```
223 }
```

```
224 }
```

```
225 }
```

```
226 }
```

```
227 }
```

```
228 }
```

```
229 }
```

```
230 }
```

```
231 }
```

```
232 }
```

```
233 }
```

```
234 }
```

```
235 }
```

```
236 }
```

```
237 }
```

```
238 }
```

```
239 }
```

```
240 }
```

```
241 }
```

```
242 }
```

```
243 }
```

```
244 }
```

```
245 }
```

```
246 }
```

```
247 }
```

```
248 }
```

```
249 }
```

```
250 }
```

```
251 }
```

```
252 }
```

```
253 }
```

```
254 }
```

```
255 }
```

```
256 }
```

```
257 }
```

```
258 }
```

```
259 }
```

```
260 }
```

```
261 }
```

```
262 }
```

```
263 }
```

```
264 }
```

```
265 }
```

```
266 }
```

```
267 }
```

```
268 }
```

```
269 }
```

```
270 }
```

```
271 }
```

```
272 }
```

```
273 }
```

```
274 }
```

```
275 }
```

```
276 }
```

```
277 }
```

```
278 }
```

```
279 }
```

```
280 }
```

```
281 }
```

```
282 }
```

```
283 }
```

```
284 }
```

```
285 }
```

```
286 }
```

```
287 }
```

```
288 }
```

```
289 }
```

```
290 }
```

```
291 }
```

```
292 }
```

```
293 }
```

```
294 }
```

```
295 }
```

```
296 }
```

```
297 }
```

```
298 }
```

```
299 }
```

```
300 }
```

```
301 }
```

```
302 }
```

```
303 }
```

```
304 }
```

```
305 }
```

```
306 }
```

```
307 }
```

```
308 }
```

```
309 }
```

```
310 }
```

```
311 }
```

```
312 }
```

```
313 }
```

```
314 }
```

```
315 }
```

```
316 }
```

```
317 }
```

```
318 }
```

```
319 }
```

```
320 }
```

```
321 }
```

```
322 }
```

```
323 }
```

```
324 }
```

```
325 }
```

```
326 }
```

```
327 }
```

```
328 }
```

```
329 }
```

```
330 }
```

```
331 }
```

```
332 }
```

```
333 }
```

```
334 }
```

```
335 }
```

```
336 }
```

```
337 }
```

```
338 }
```

```
339 }
```

```
340 }
```

```
341 }
```

```
342 }
```

```
343 }
```

```
344 }
```

```
345 }
```

```
346 }
```

```
347 }
```

```
348 }
```

```
349 }
```

```
350 }
```

```
351 }
```

```
352 }
```

```
353 }
```

```
354 }
```

```
355 }
```

```
356 }
```

```
357 }
```

```
358 }
```

```
359 }
```

```
360 }
```

```
361 }
```

```
362 }
```

```
363 }
```

```
364 }
```

```
365 }
```

```
366 }
```

```
367 }
```

```
368 }
```

```
369 }
```

```
370 }
```

```
371 }
```

```
372 }
```

```
373 }
```

```
374 }
```

```
375 }
```

```
376 }
```

```
377 }
```

```
378 }
```

```
379 }
```

```
380 }
```

```
381 }
```

```
382 }
```

```
383 }
```

```
384 }
```

```
385 }
```

```
386 }
```

```
387 }
```

```
388 }
```

```
389 }
```

```
390 }
```

```
391 }
```

```
392 }
```

```
393 }
```

```
394 }
```

```
395 }
```

```
396 }
```

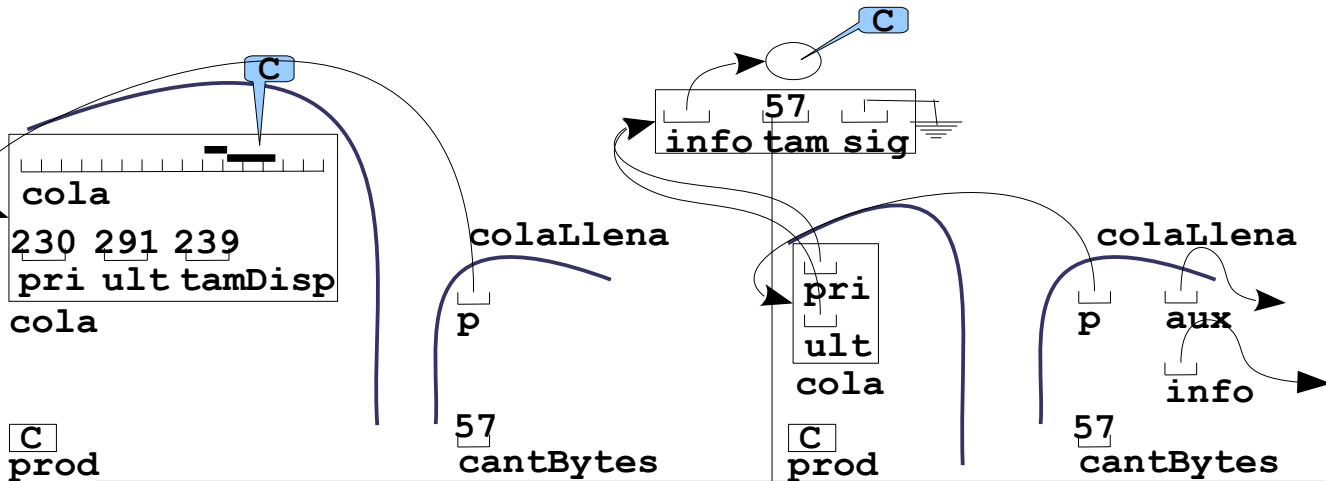
```
397 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
87 void probarPonerYSacarDeCola(void)
```

```
88 {
```

```
89     tProd p;
```

```
90     tCola c;
```

```
12 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
13 {
```

```
    *p)malloc(sizeof(tNodo));
```

```
    c(cantBytes);
```

```
    if (info == NULL;
```

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
18 {
```

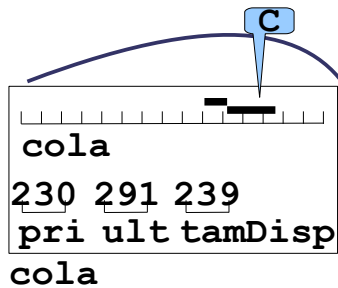
```
19     return p->tamDisp < cantBytes + sizeof(unsigned);
```

```
20 }
```


Tipos de Datos Abstractos

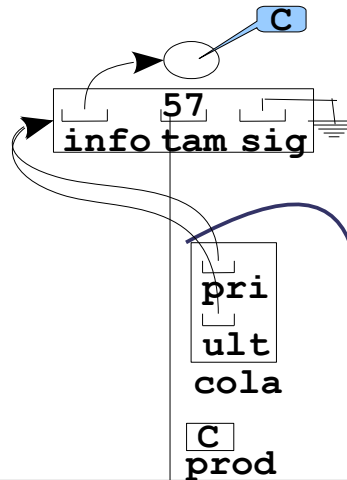
Tipo de dato Cola.

Implementación estática



C
prod

Implementación dinámica



C
prod

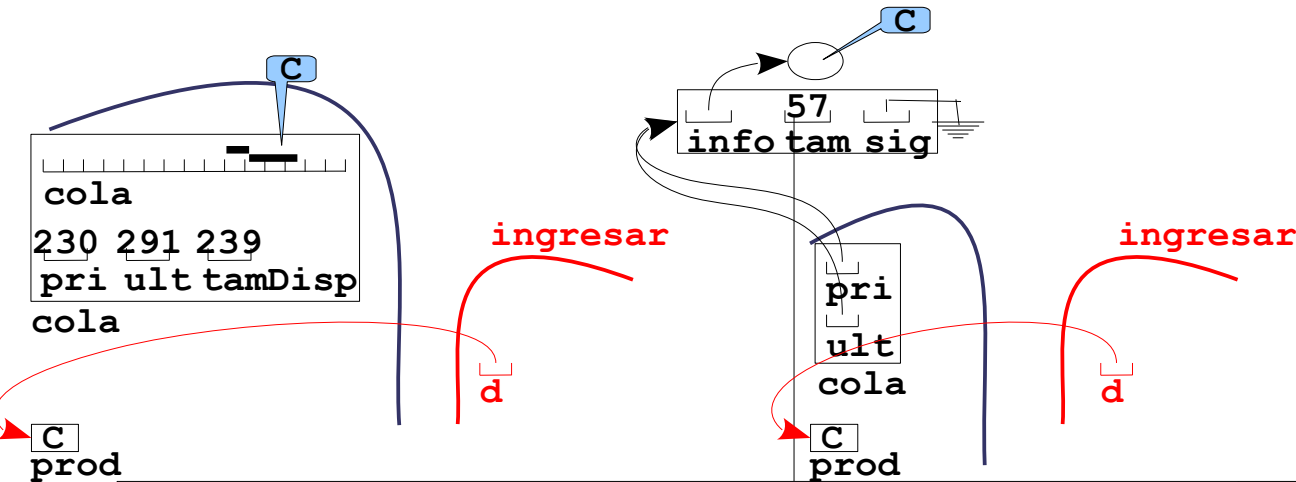
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while (!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if (!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



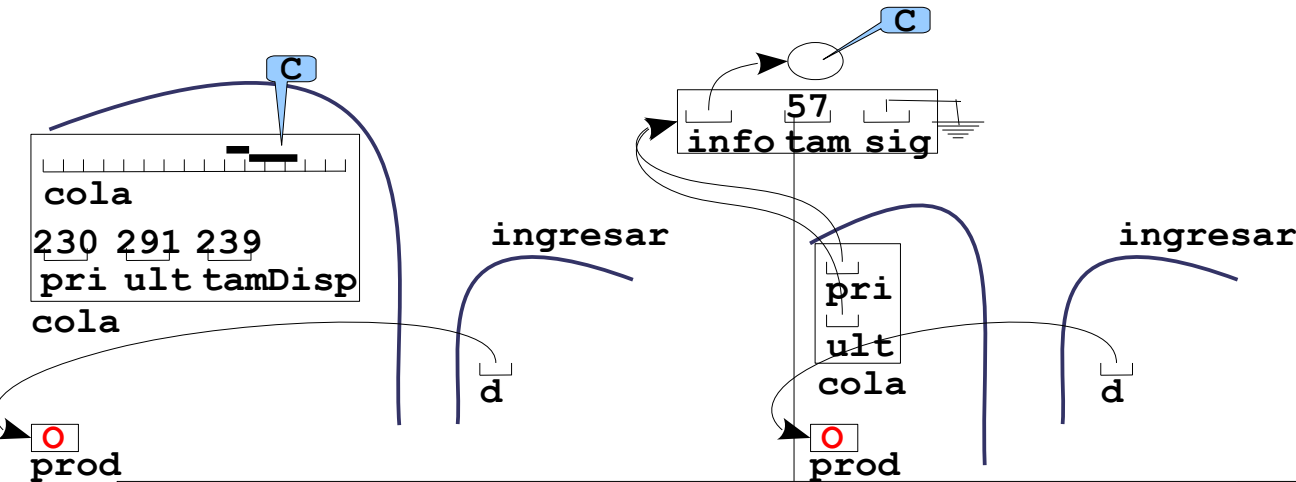
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

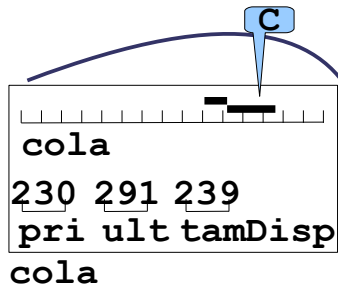


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

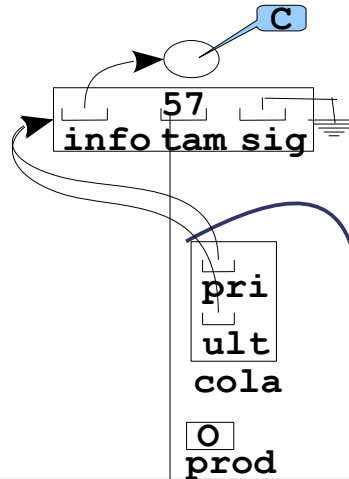
Tipo de dato Cola.

Implementación estática



O
prod

Implementación dinámica



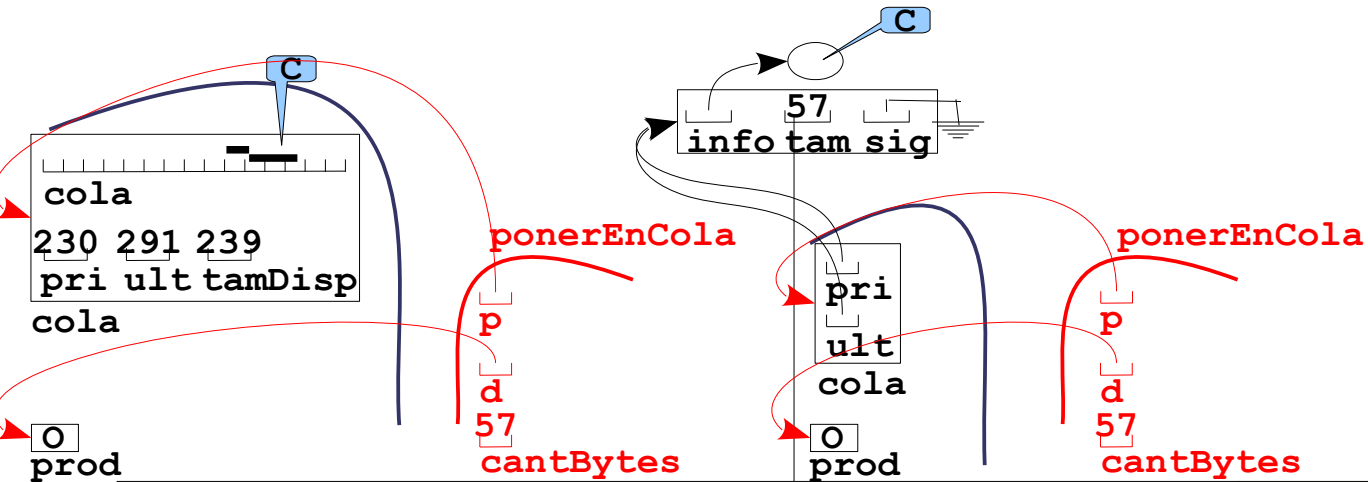
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



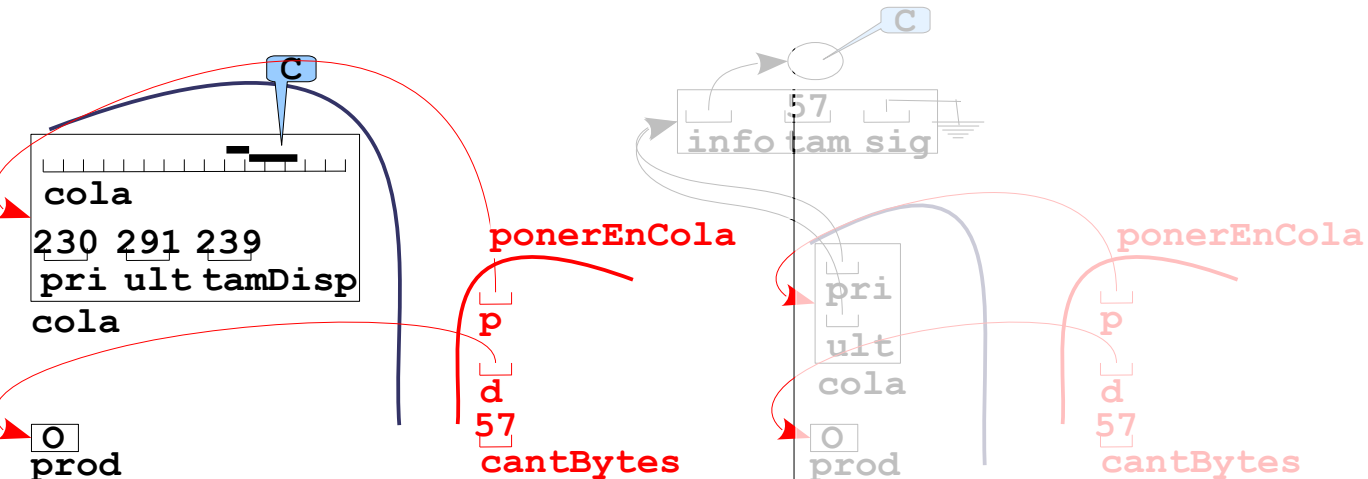
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



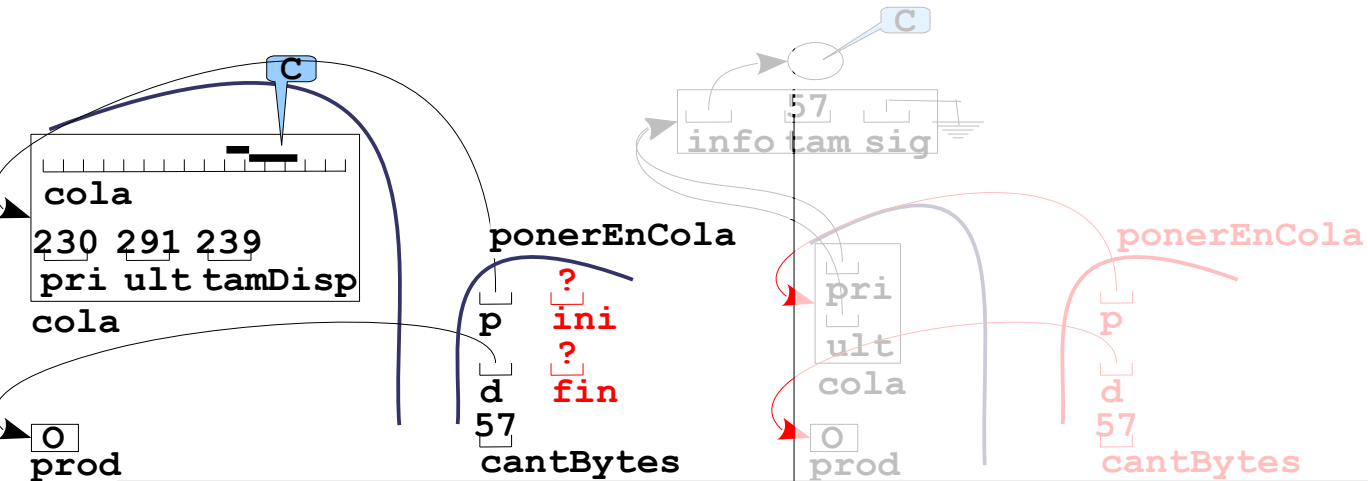
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



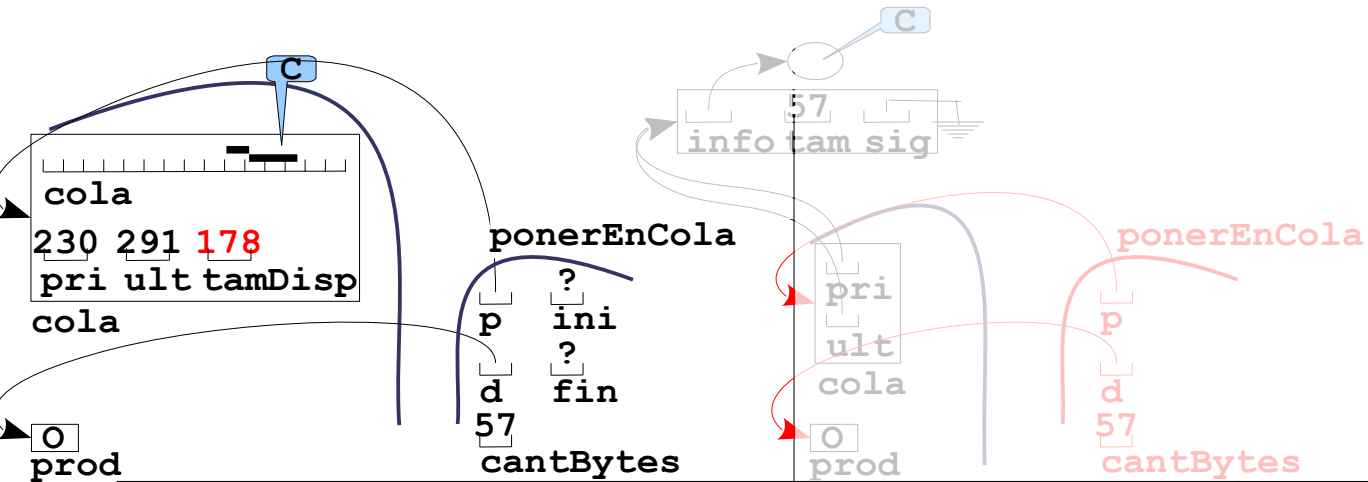
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



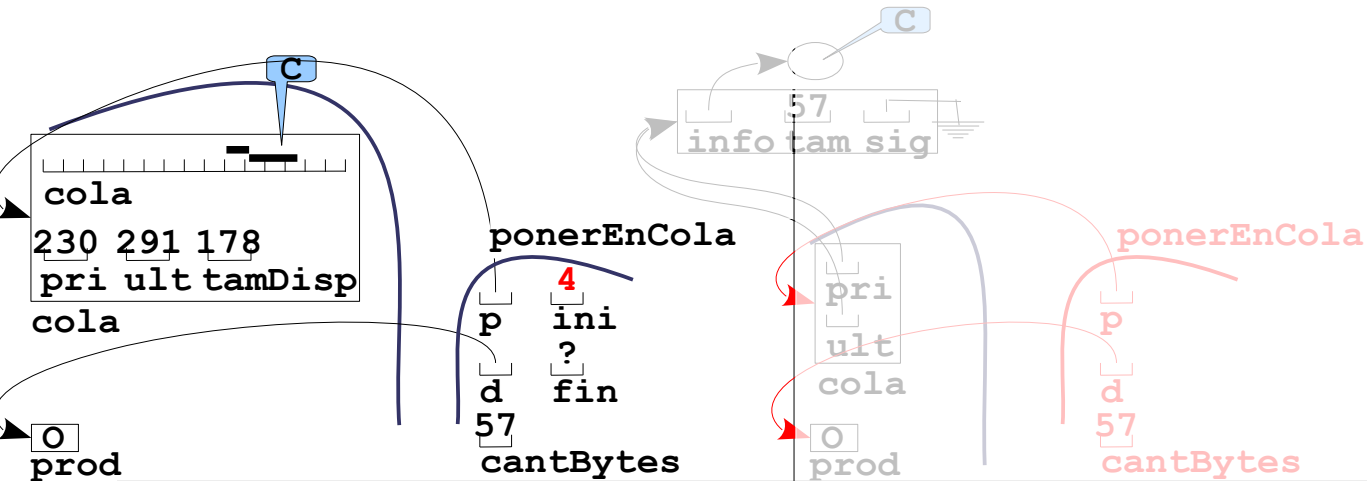
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



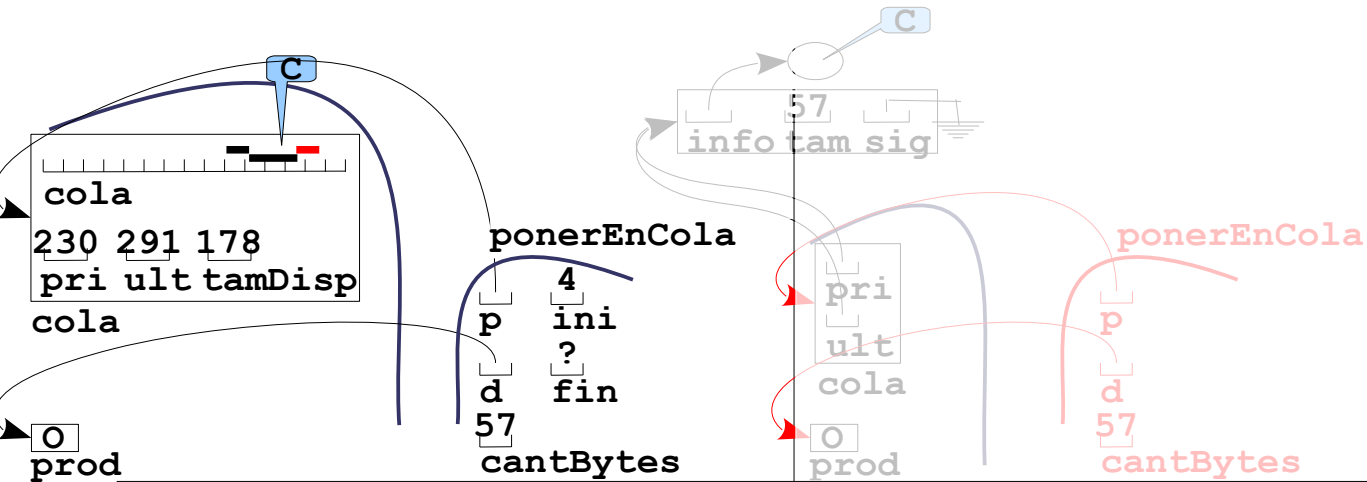
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                 fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



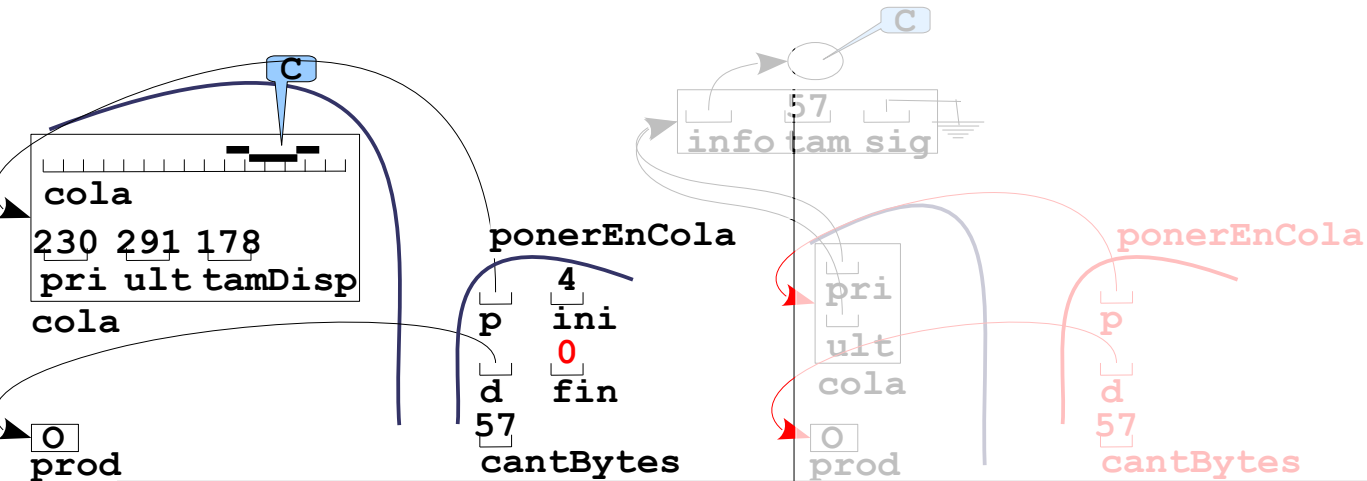
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

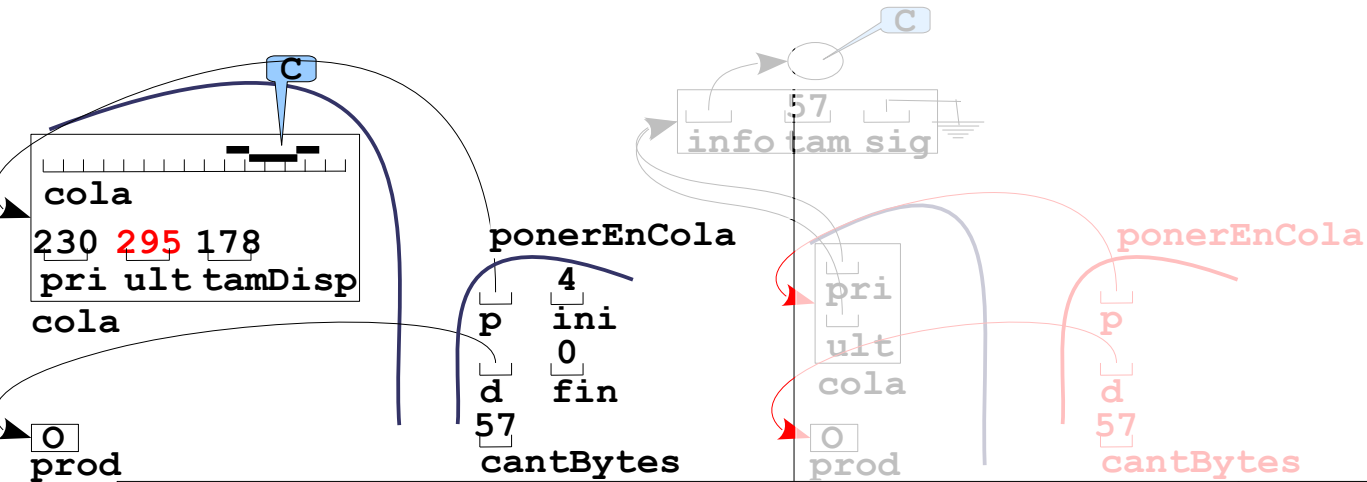
to(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



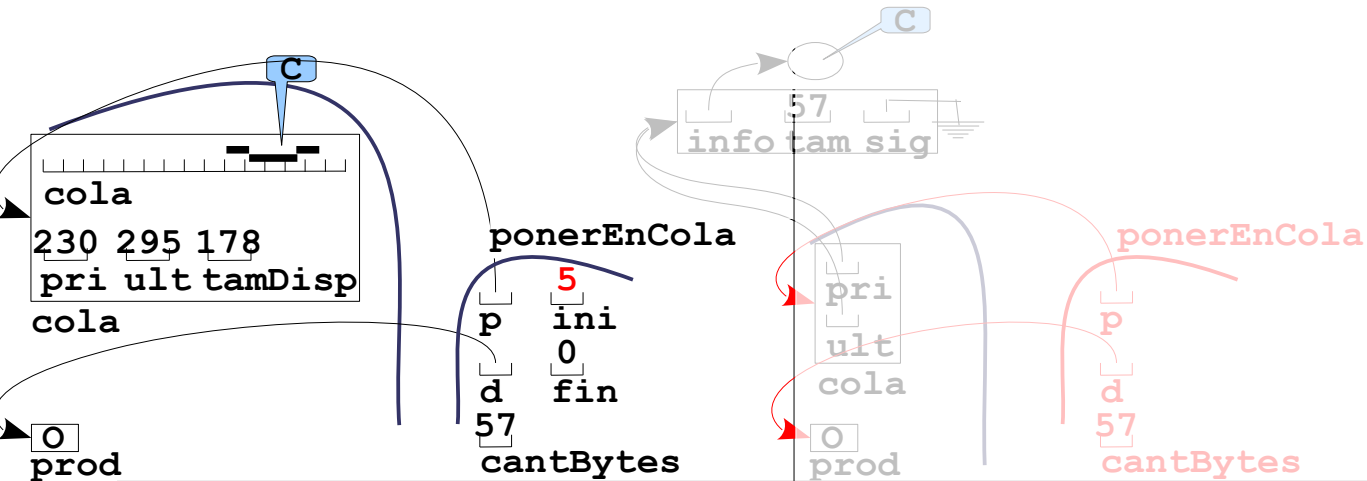
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



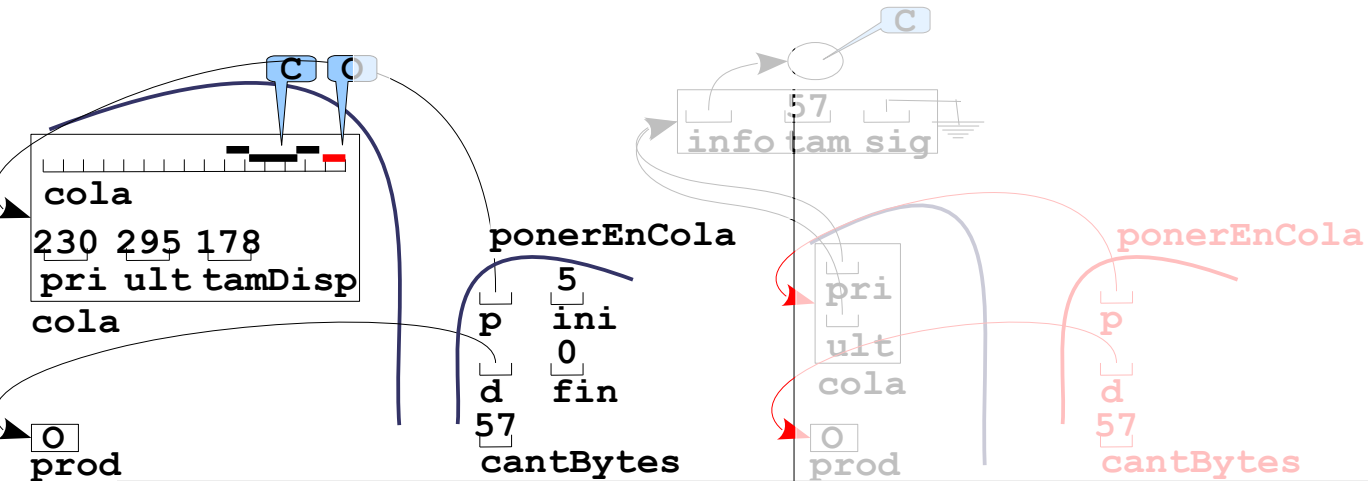
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



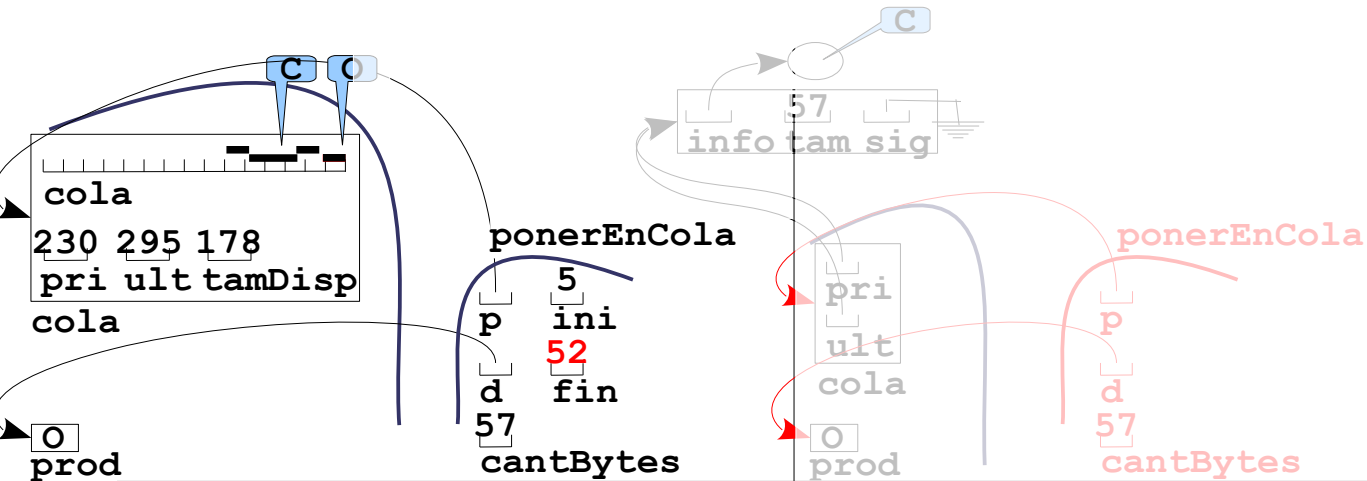
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32      memcpy(p->cola + p->ult, &cantBytes, ini);
23 33      if((fin = sizeof(cantBytes) - ini) != 0)
24 34      {
25 35          memcpy(p->cola + p->ult, &cantBytes, ini);
26 36          if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
27 37          {
28 38              memcpy(p->cola + p->ult, d, ini);
29 39              if((fin = cantBytes - ini) != 0)
30 40              {
31 41                  memcpy(p->cola, ((char *)d) + ini, fin);
32 42                  p->ult = fin ? fin : p->ult + ini;
33 43                  return 1;
34 44              }
35 45          }
36 46      }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



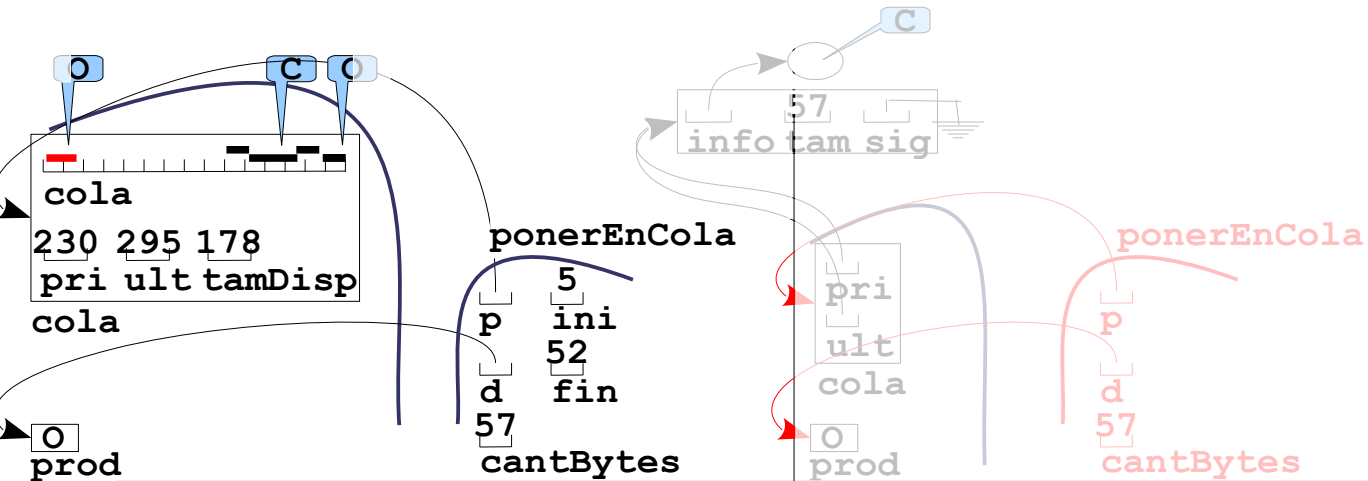
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32      memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
23 33
24 34      p->ult = fin ? fin : p->ult + ini;
25 35
26 36      if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
27 37          memcpy(p->cola + p->ult, d, ini);
28 38
29 39      if((fin = cantBytes - ini) != 0)
30 40          memcpy(p->cola, ((char *)d) + ini, fin);
31 41      p->ult = fin ? fin : p->ult + ini;
32 42      return 1;
33 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



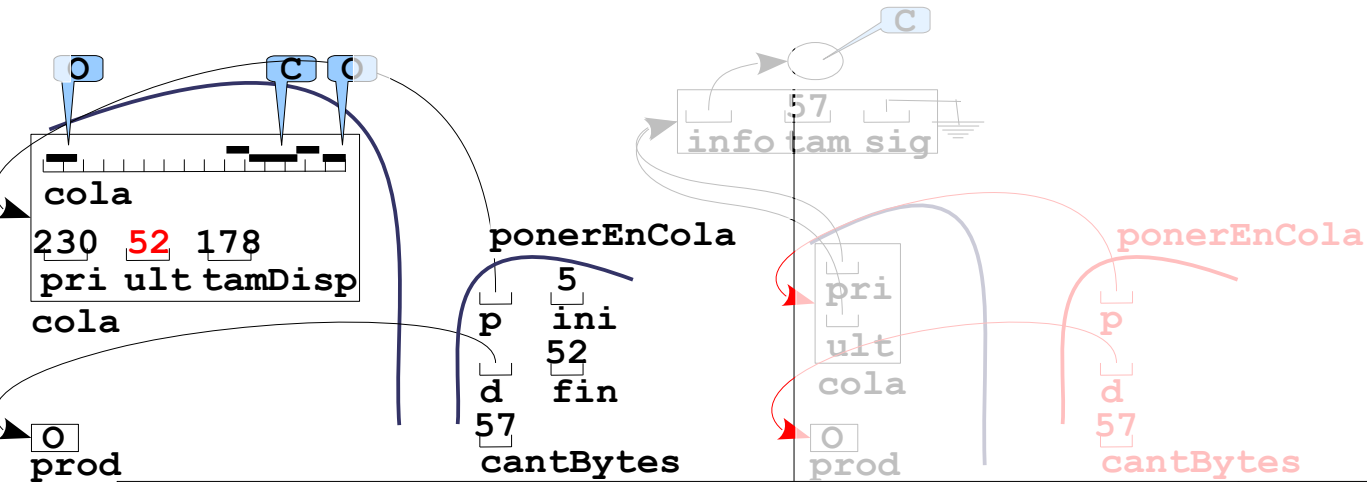
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 32      memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
23 33
24 34      p->ult = fin ? fin : p->ult + ini;
25 35
26 36      if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
27 37          memcpy(p->cola + p->ult, d, ini);
28 38
29 39      if((fin = cantBytes - ini) != 0)
30 40          memcpy(p->cola, ((char *)d) + ini, fin);
31 41      p->ult = fin ? fin : p->ult + ini;
32 42      return 1;
33 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

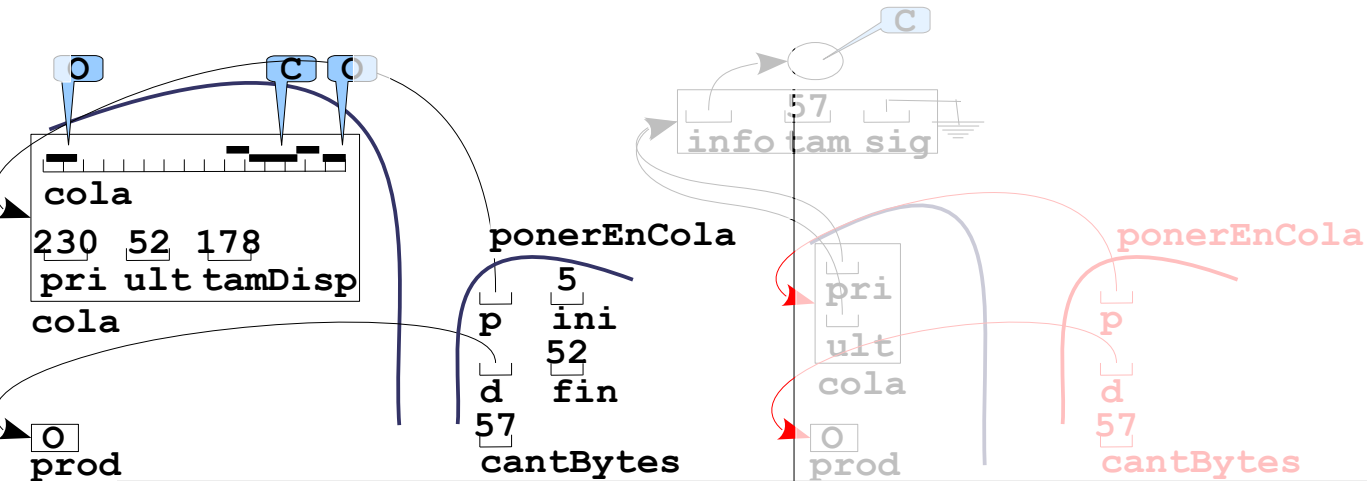
to(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



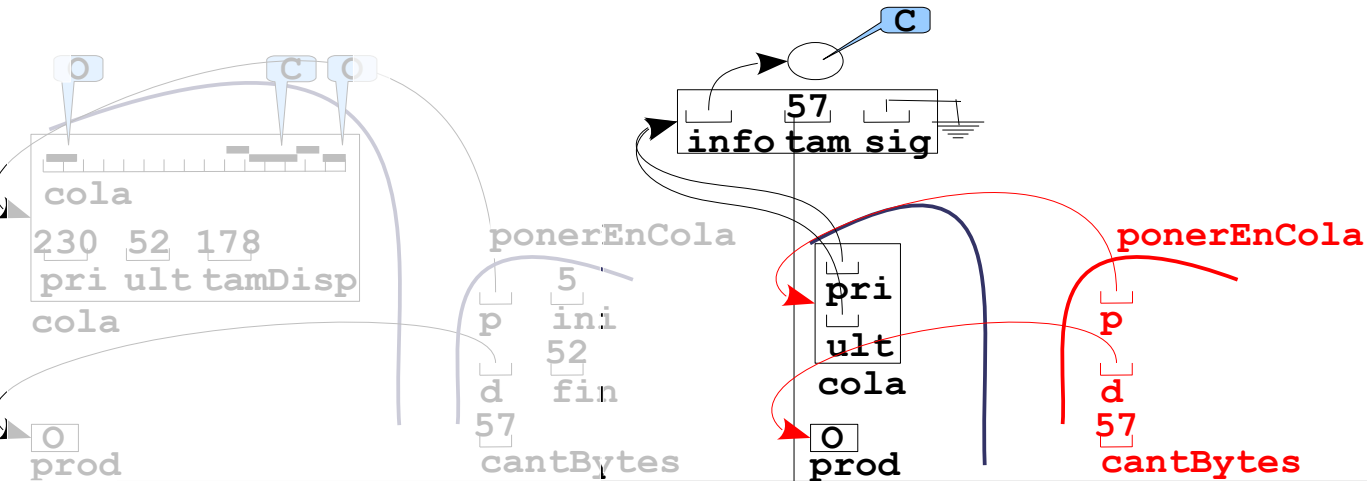
```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo
88 {
89
90
91
92
93
94
95

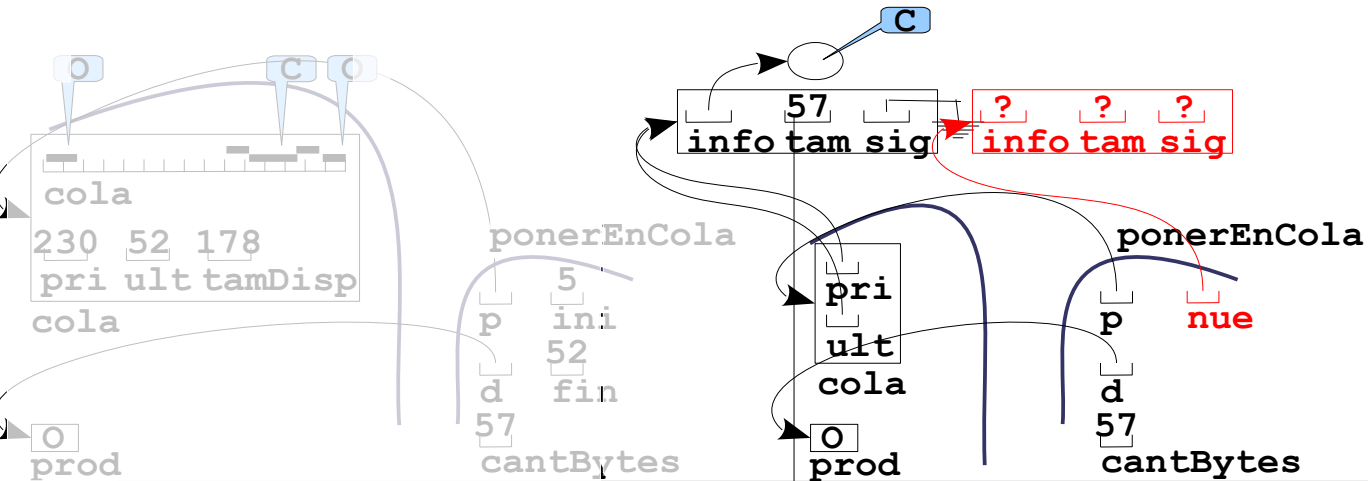
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
31     ...
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87  vo:
88  {
89
90
91
92
93
94
95

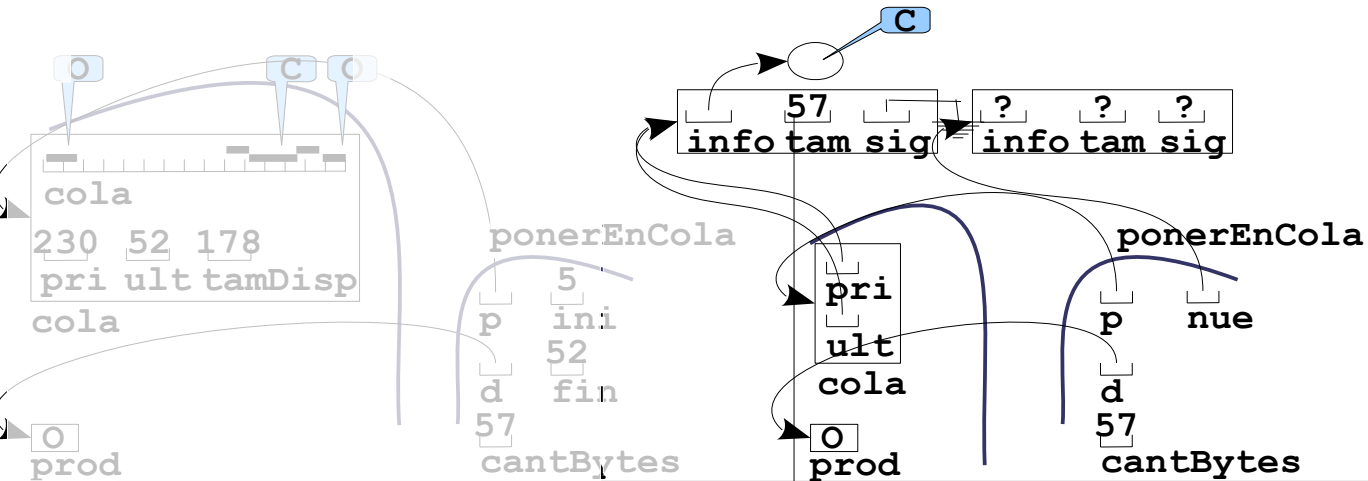
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21  int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22  {
23      tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25      if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26      {
27          free(nue);
28          return 0;
29      }
30      memcpy(nue->info, d, cantBytes);
31
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



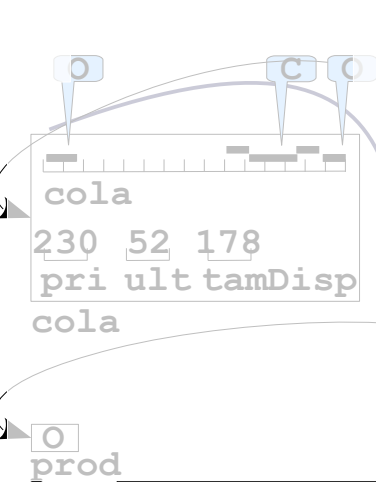
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87  vo:
88  {
89
90
91
92
93
94
95

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21  int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22  {
23      tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25      if (nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26      {
27          free(nue);
28          return 0;
29      }
30      memcpy(nue->info, d, cantBytes);
31
```


Tipos de Datos Abstractos

Tipo de dato Cola.

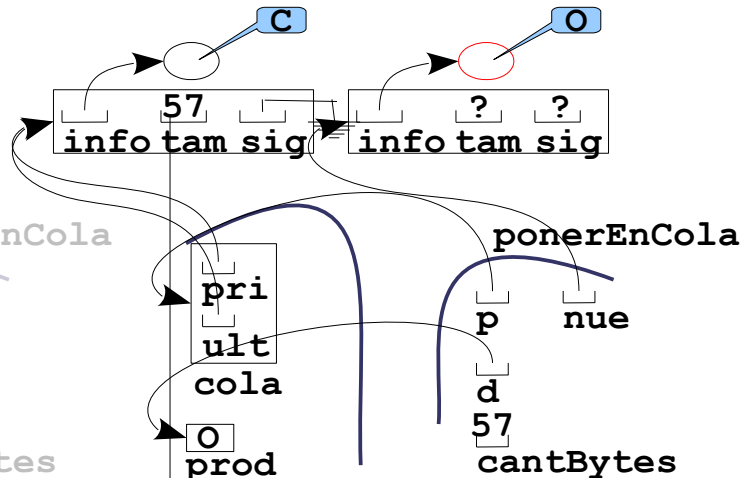
Implementación estática



ponerEnCola

p 5
ini
52
d fin
57
cantBytes

Implementación dinámica



ponerEnCola

p
nue
d 57
cantBytes

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo
88 {
89
90
91
92
93
94
95

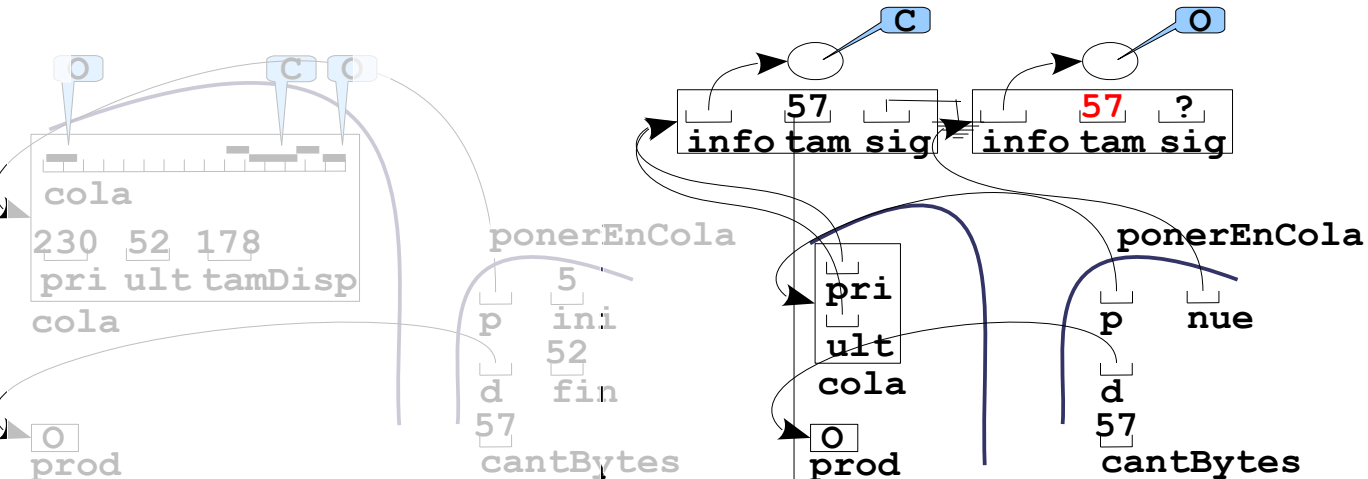
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



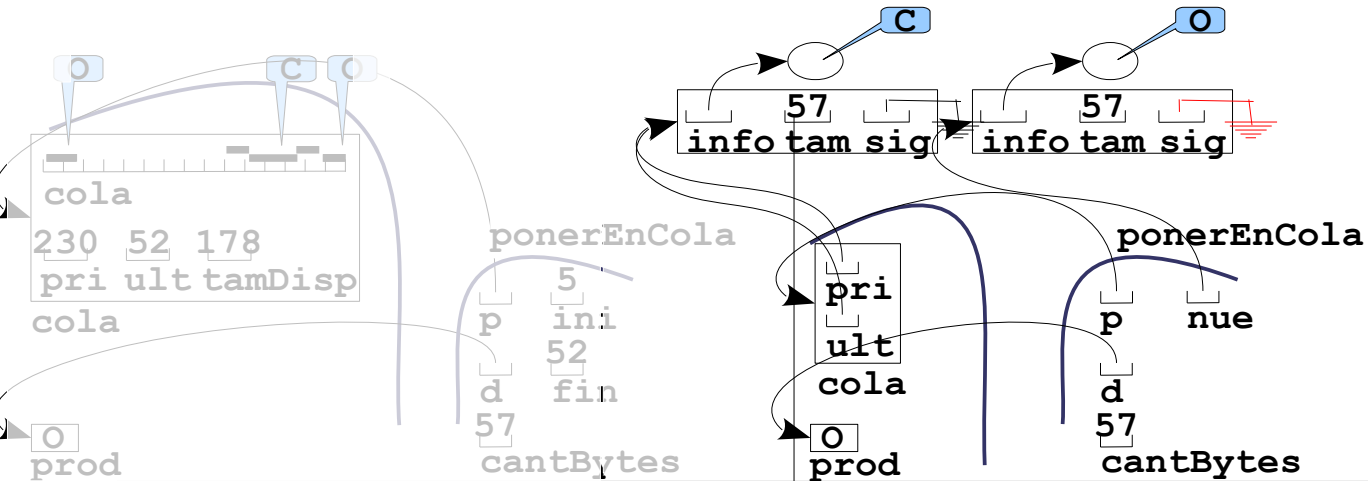
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40     return 1;
98 41
99 42 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



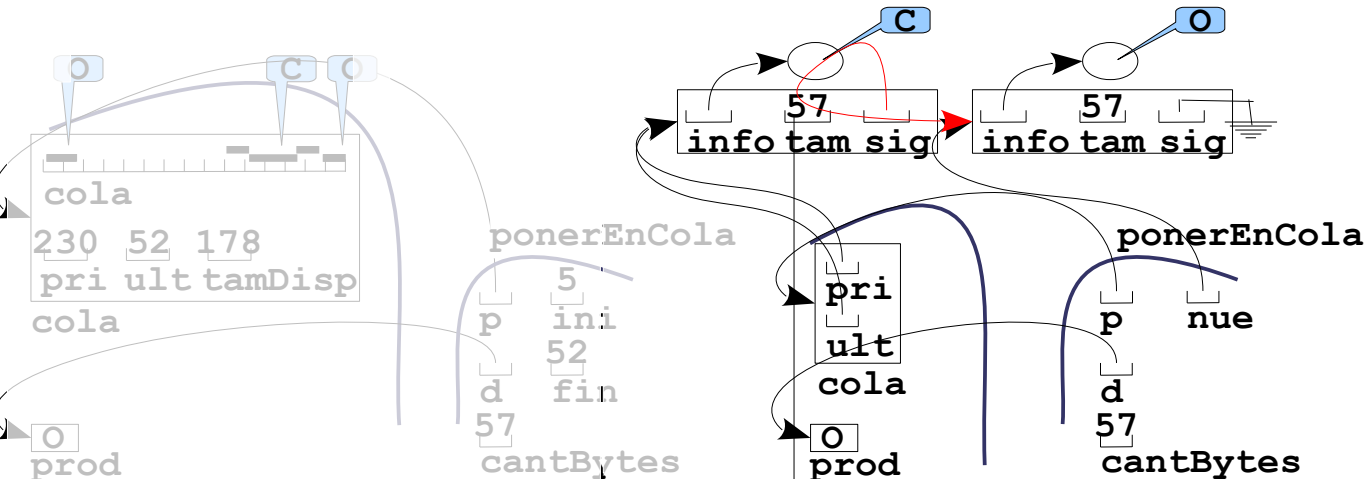
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36 else
94 37     p->pri = nue;
95 38     p->ult = nue;
96 39     return 1;
97 40 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



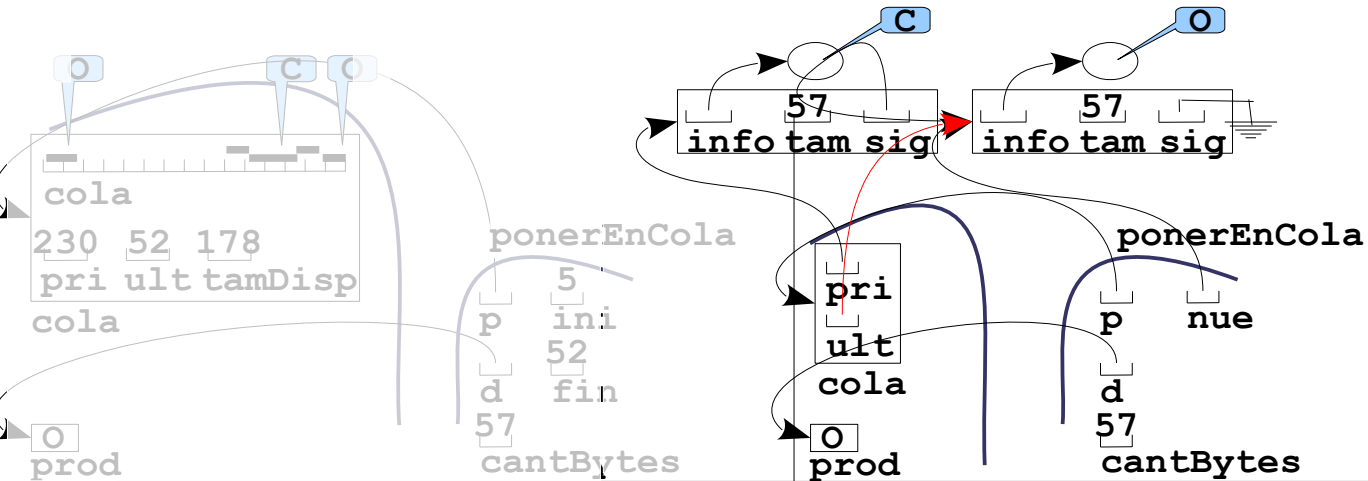
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->tamInfo = cantBytes;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36 else
94 37     p->pri = nue;
95 38 p->ult = nue;
    39 return 1;
    }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



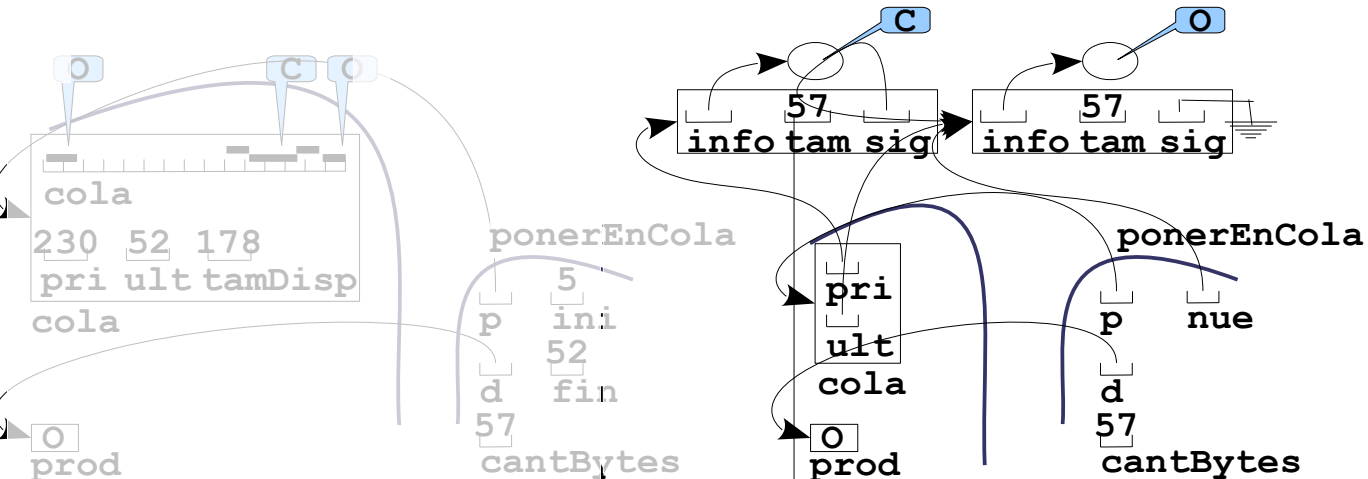
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36 else
94 37     p->pri = nue;
95 38 p->ult = nue;
39 39 return 1;
}
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

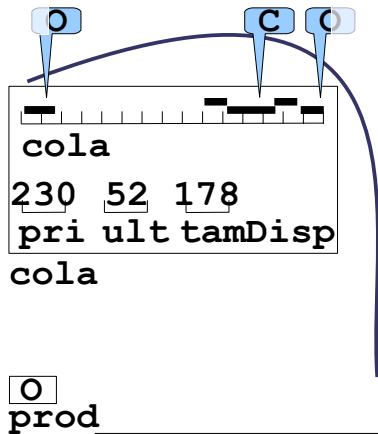


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->tamInfo = cantBytes;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40
98 41 return 1;
99 42 }
```

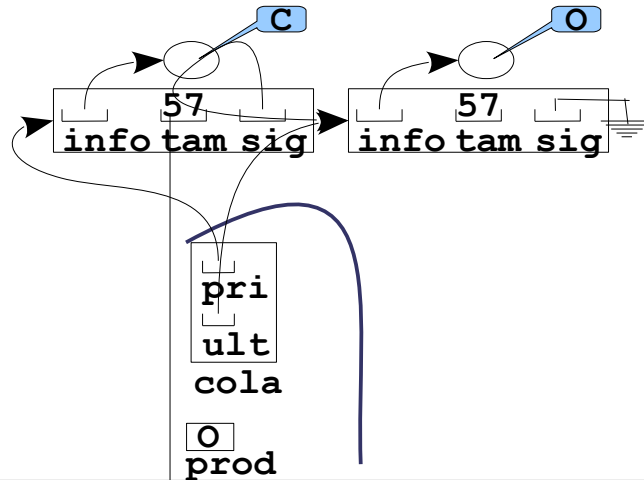
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



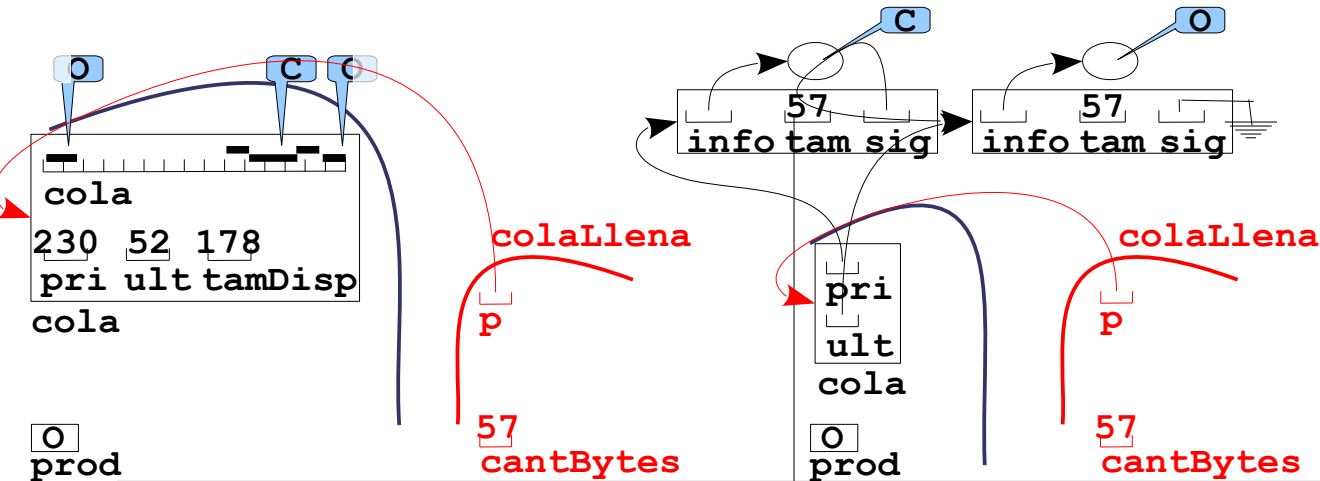
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



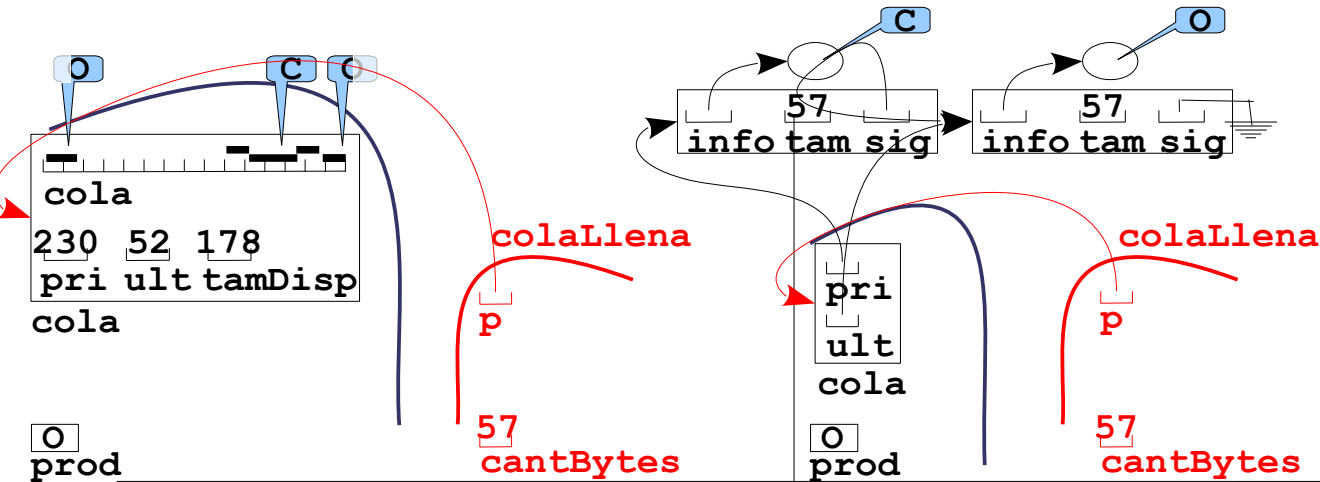
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while (!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if (!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd p;
90     tCola c;

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {

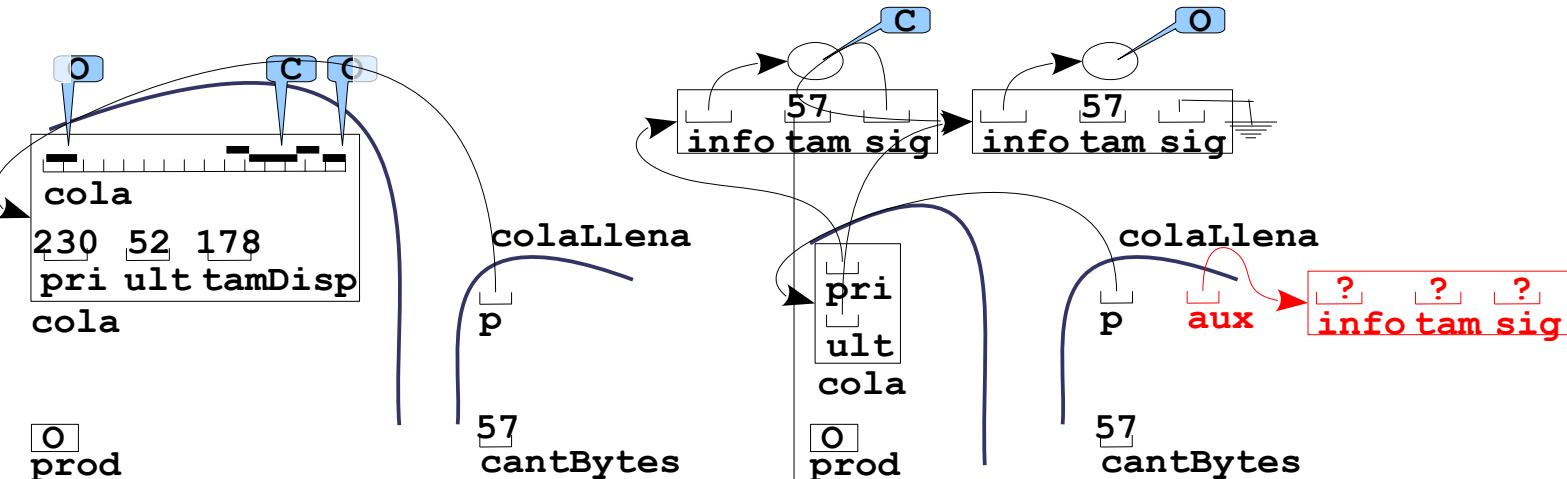
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp < cantBytes + sizeof(unsigned);
20 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
```

```
88 {
89     tProd p;
90     tCola c;
```

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisponible < cantBytes;
20 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
```

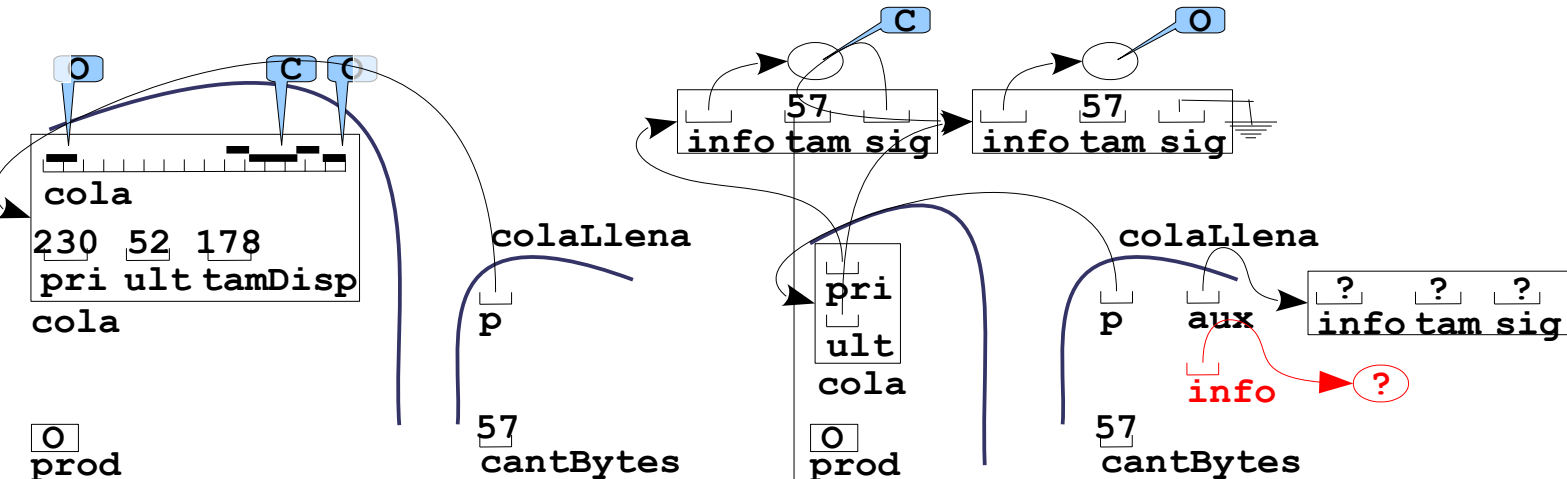
```
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
87 void probarPonerYSacarDeCola(void)
```

```
88 {
```

```
89     tProd p;
```

```
90     tCola c;
```

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
18 {
```

```
19     return p->tamDi
```

```
20 }
```

```
12 int colaLlena(const tCola *p, unsigned cantBytes)
```

```
13 {
```

```
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
```

```
15     void *info = malloc(cantBytes);
```

```
16     free(aux);
```

```
17     free(info);
```

```
18     return aux == NULL || info == NULL;
```

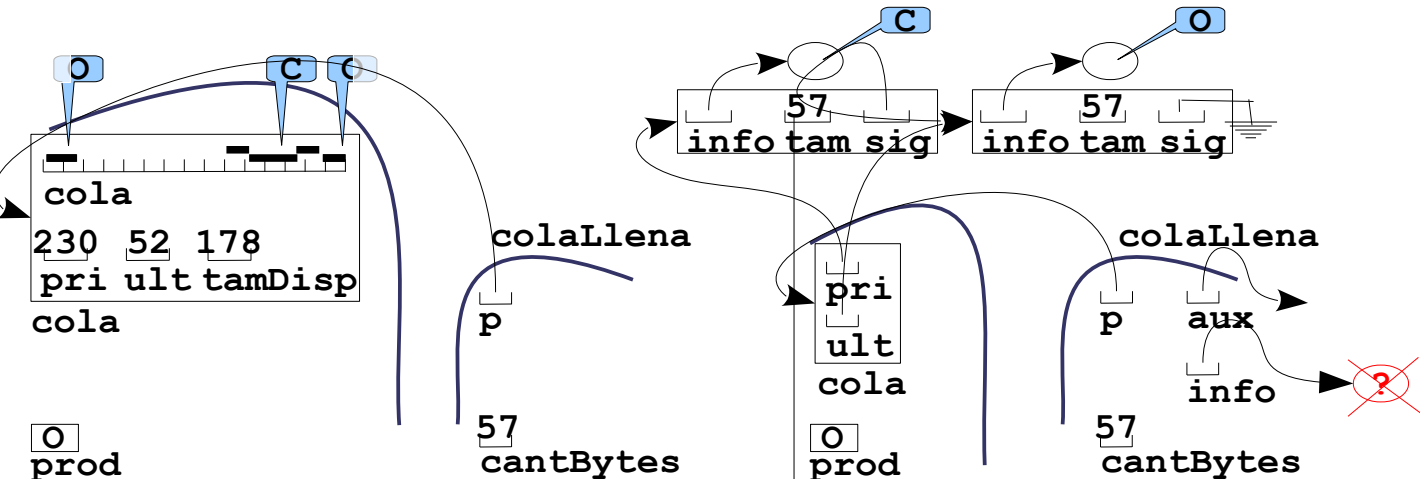
```
19 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x

```
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd p;
90     tCola c;
```

main.c x main.h x productos.c x productos.h

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisponible == cantBytes;
20 }
```

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x

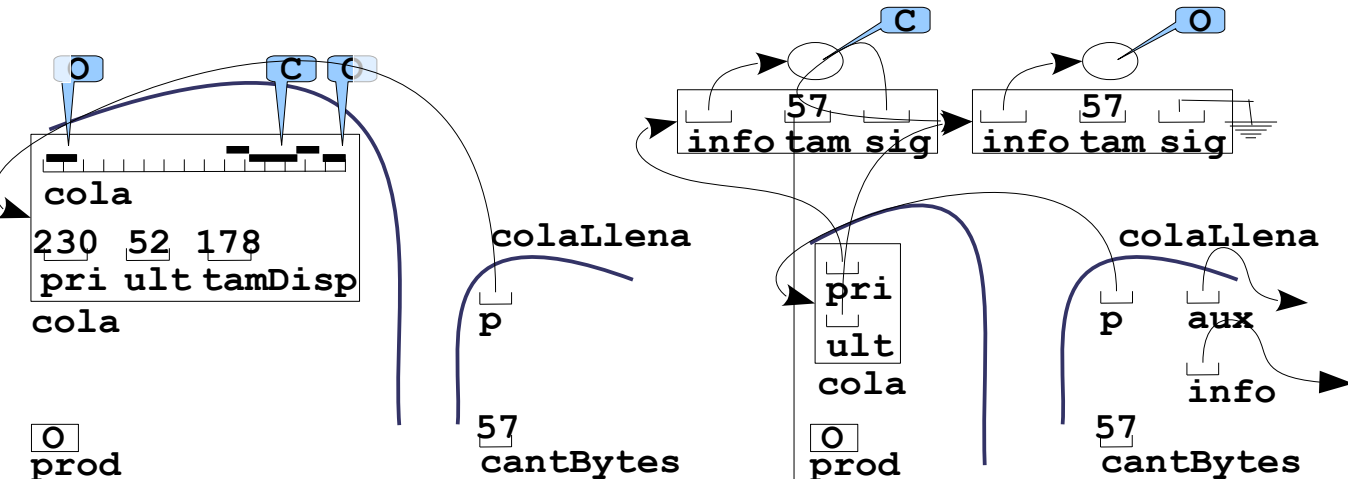
```
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x

```
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd p;
90     tCola c;
```

main.c x main.h x productos.c x productos.h

```
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisponible;
20 }
```

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x

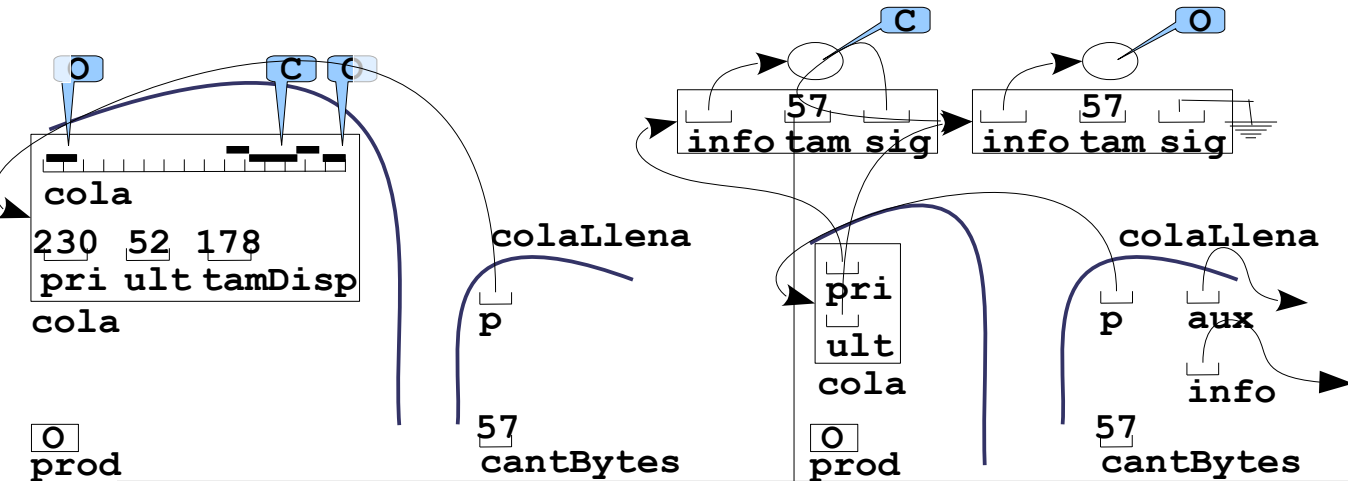
```
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {
14     tNodo *aux = (tNodo *)malloc(sizeof(tNodo));
15     void *info = malloc(cantBytes);
16     free(aux);
17     free(info);
18     return aux == NULL || info == NULL;
19 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd p;
90     tCola c;

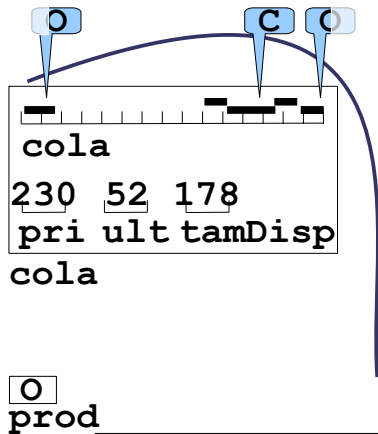
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
12 int colaLlena(const tCola *p, unsigned cantBytes)
13 {

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
17 int colaLlena(const tCola *p, unsigned cantBytes)
18 {
19     return p->tamDisp < cantBytes + sizeof(unsigned);
20 }
```

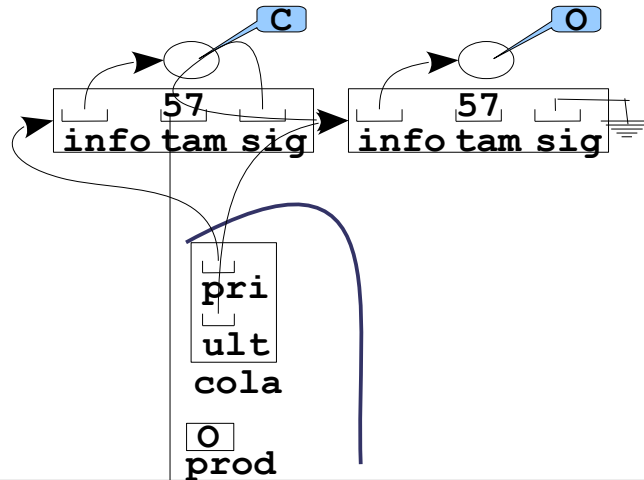
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



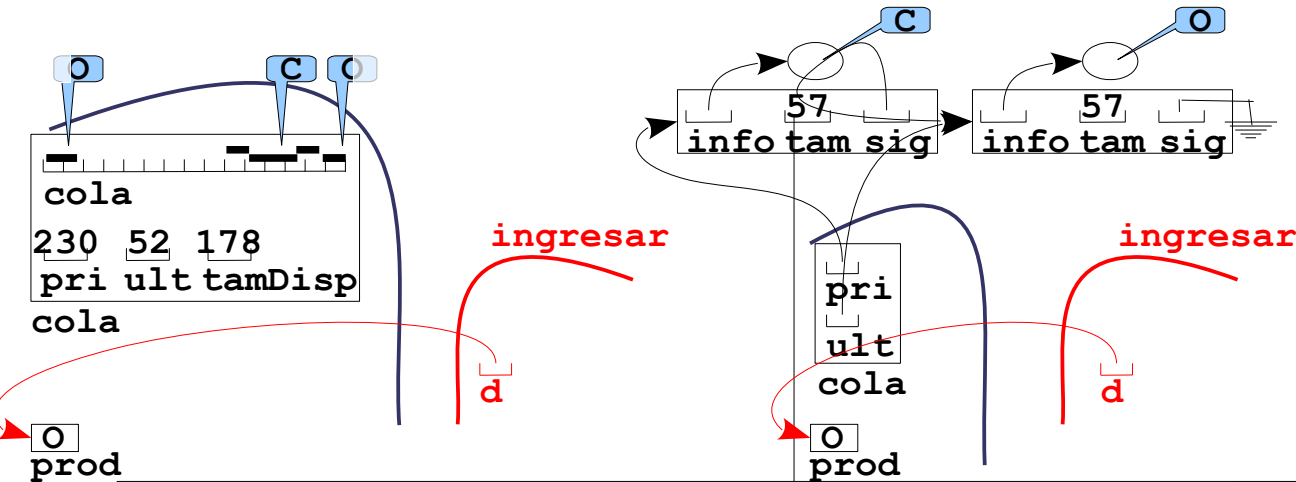
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while (!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if (!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



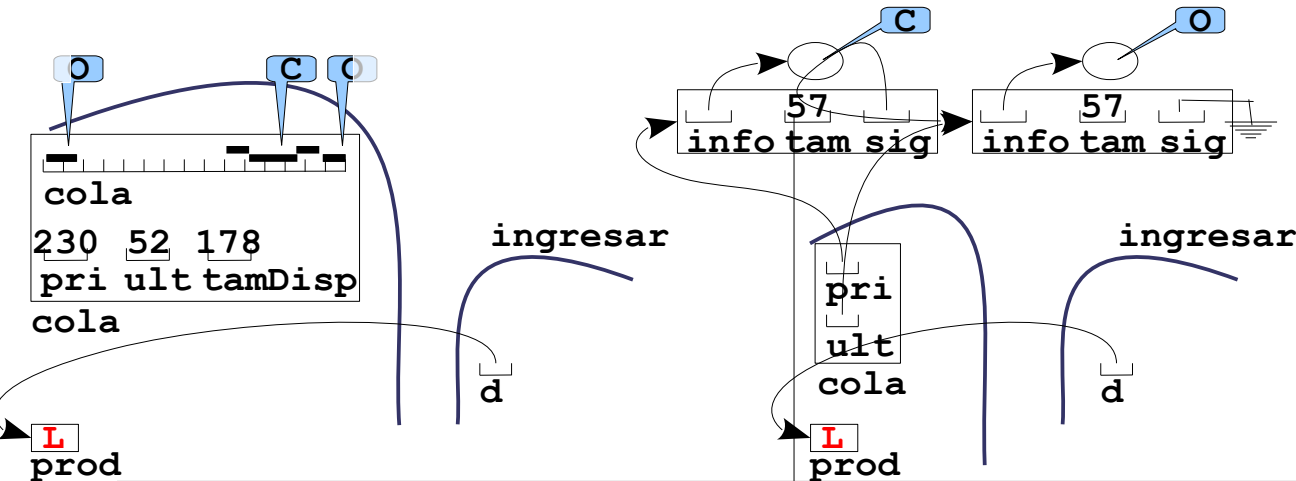
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

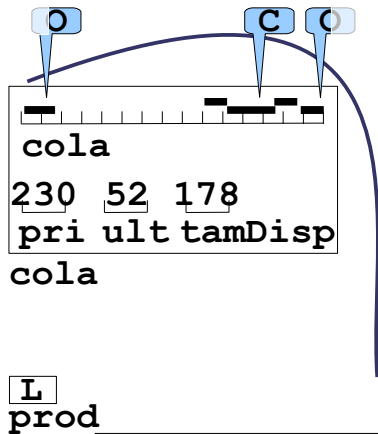


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

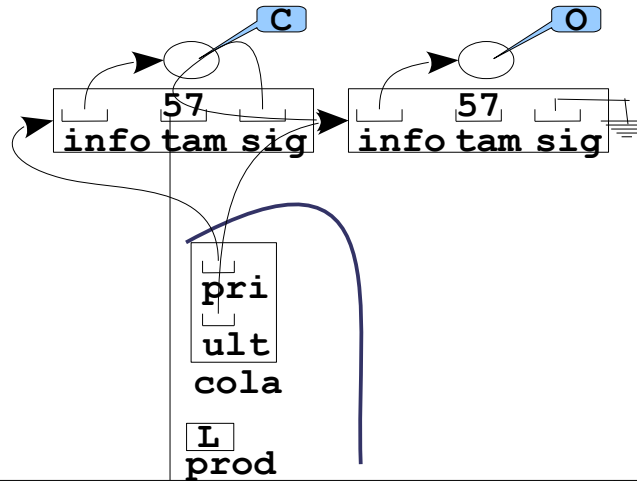
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



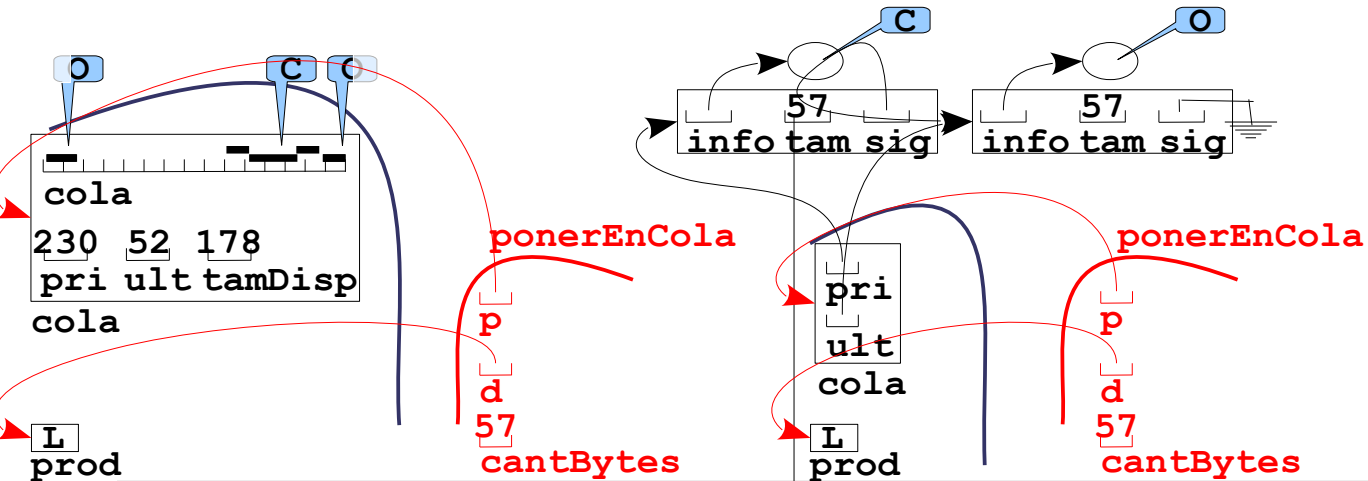
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



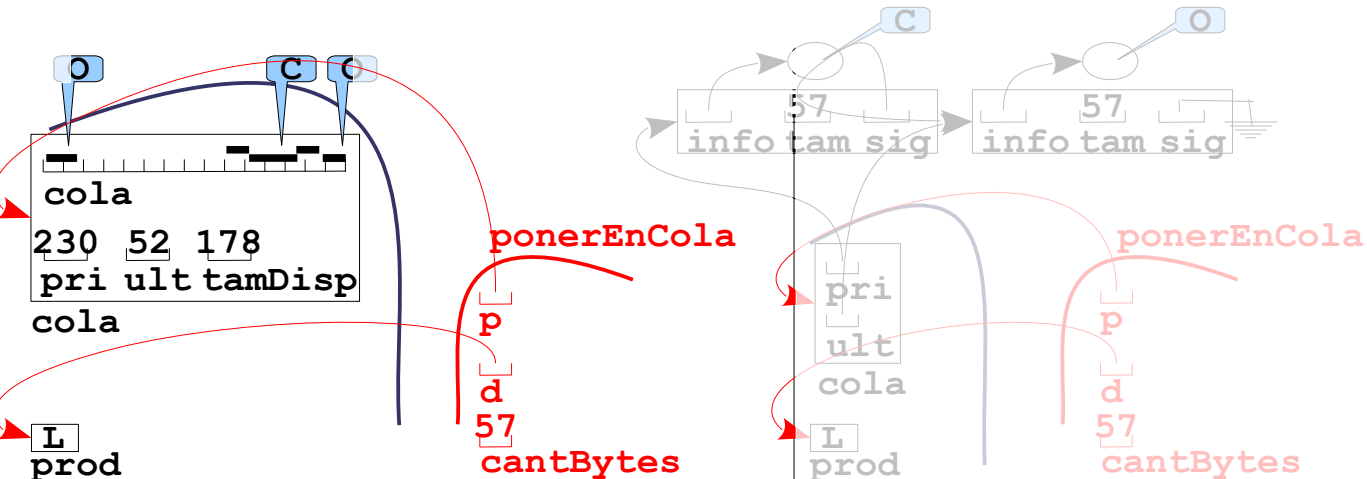
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



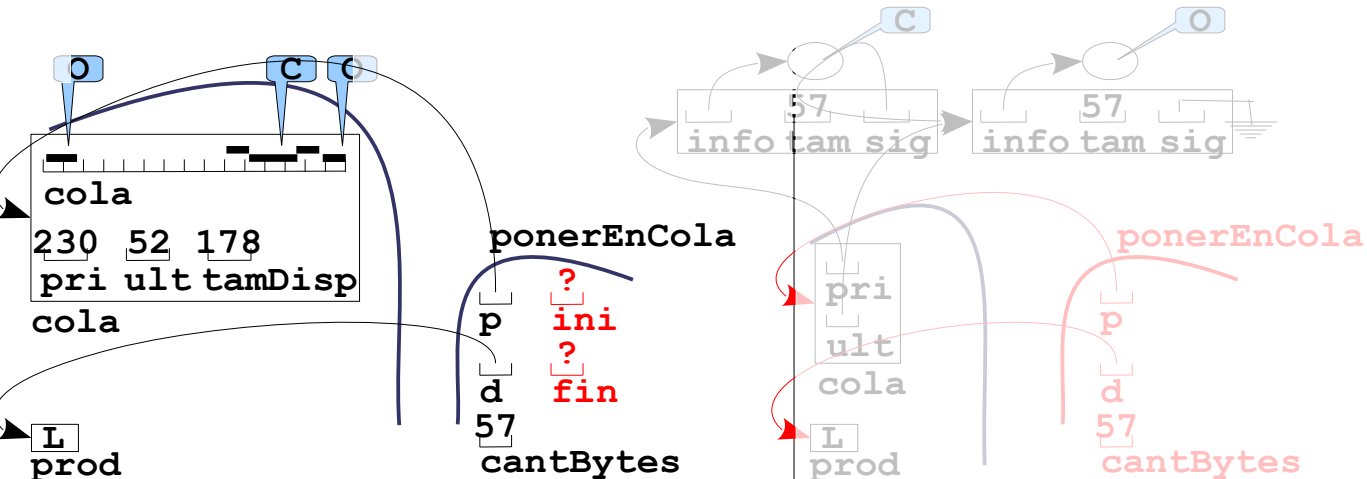
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                 fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

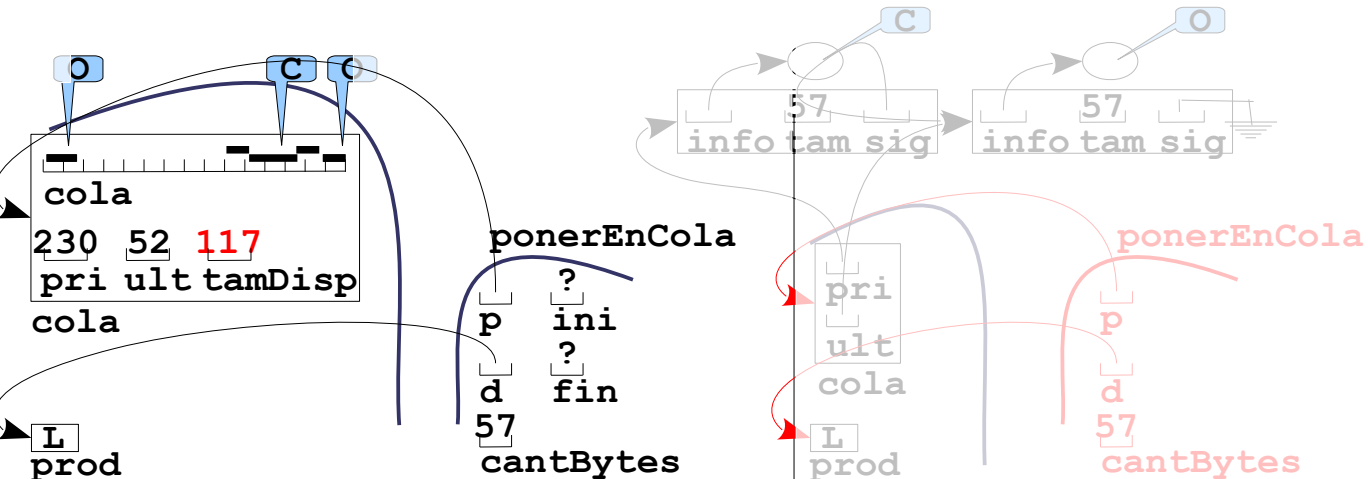
ducto(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

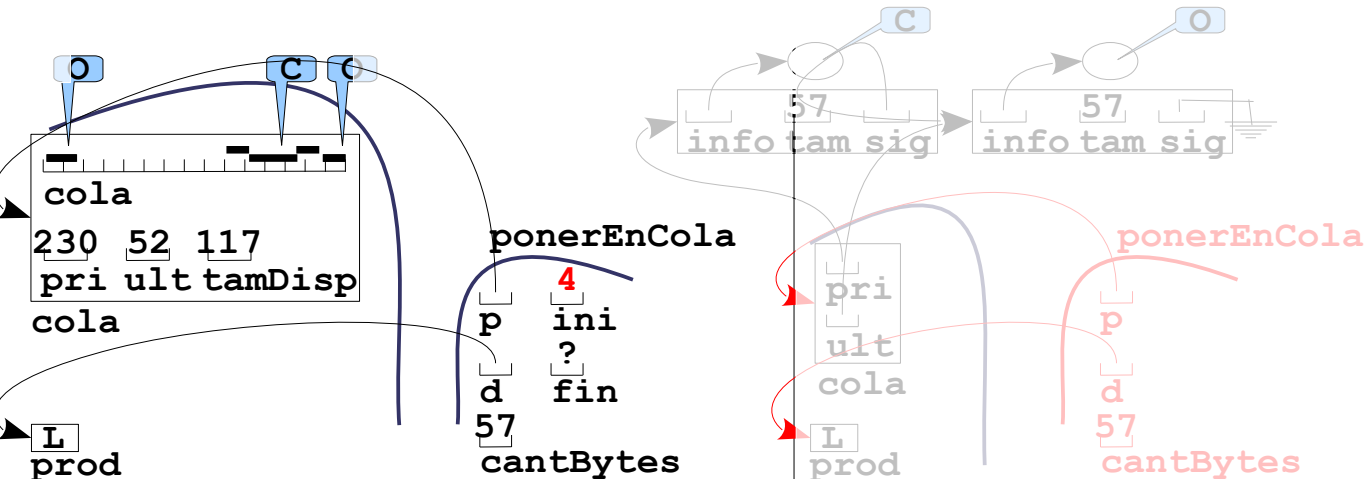
ducto(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



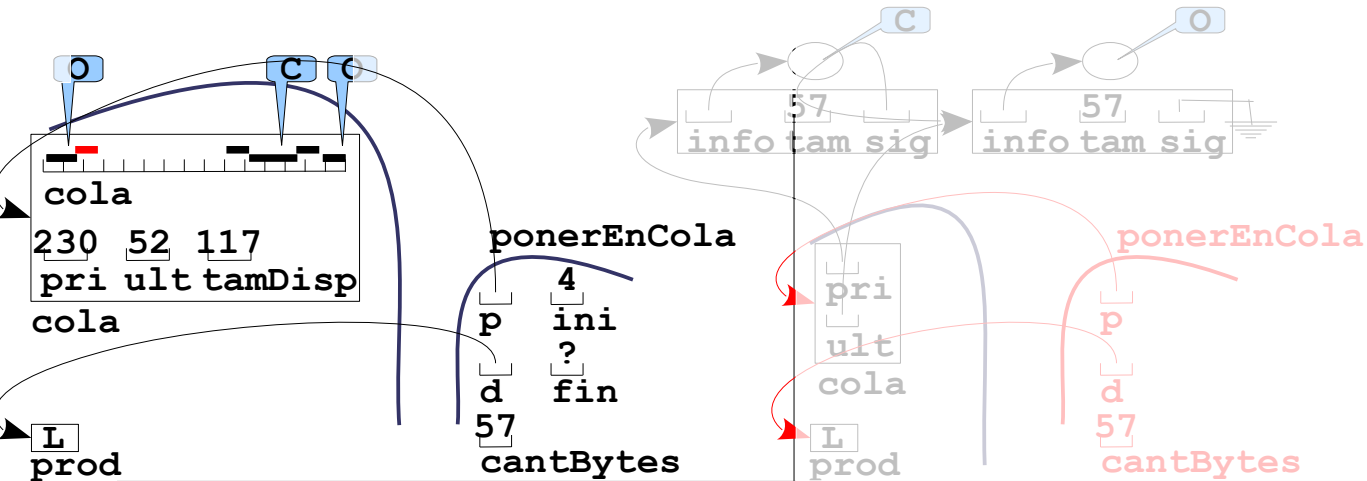
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
main.c x main.h x productos.c x productos.h x *cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



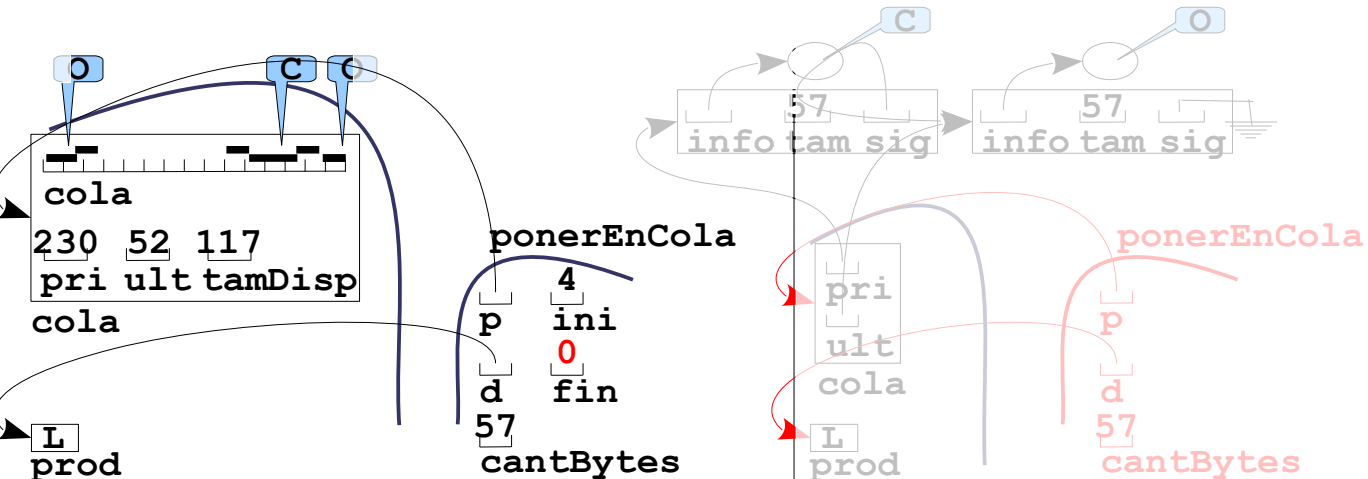
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
23 {
24     unsigned    ini,
25                fin;
26
27     if(p->tamDisp < sizeof(unsigned) + cantBytes)
28         return 0;
29     p->tamDisp -= sizeof(unsigned) + cantBytes;
30     if((ini = minimo(sizeof(cantBytes), TAM_COLA - p->ult)) != 0)
31         memcpy(p->cola + p->ult, &cantBytes, ini);
32     if((fin = sizeof(cantBytes) - ini) != 0)
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



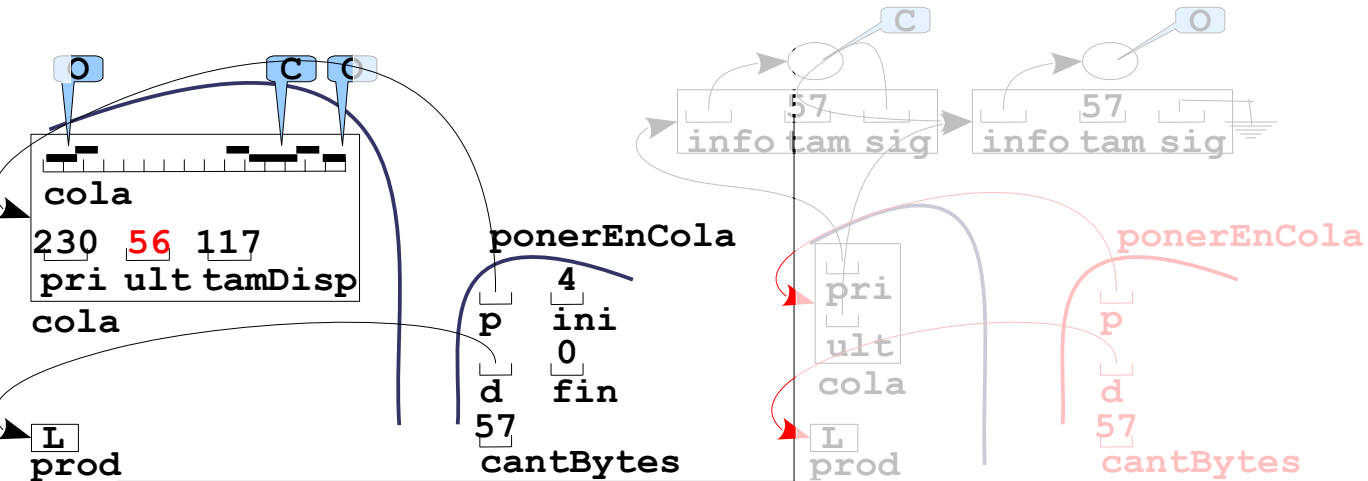
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



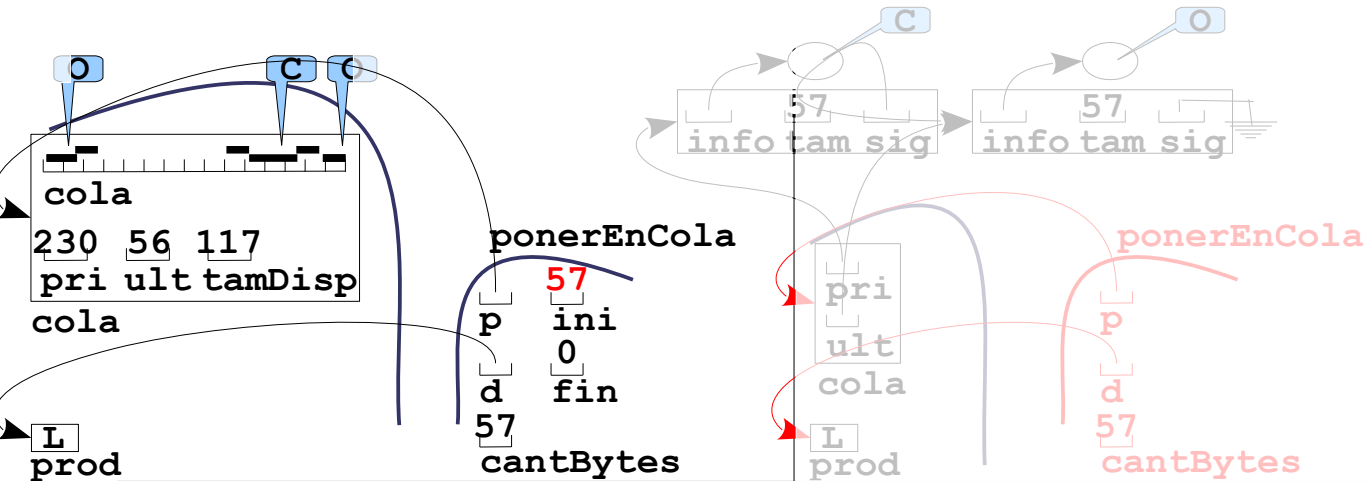
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



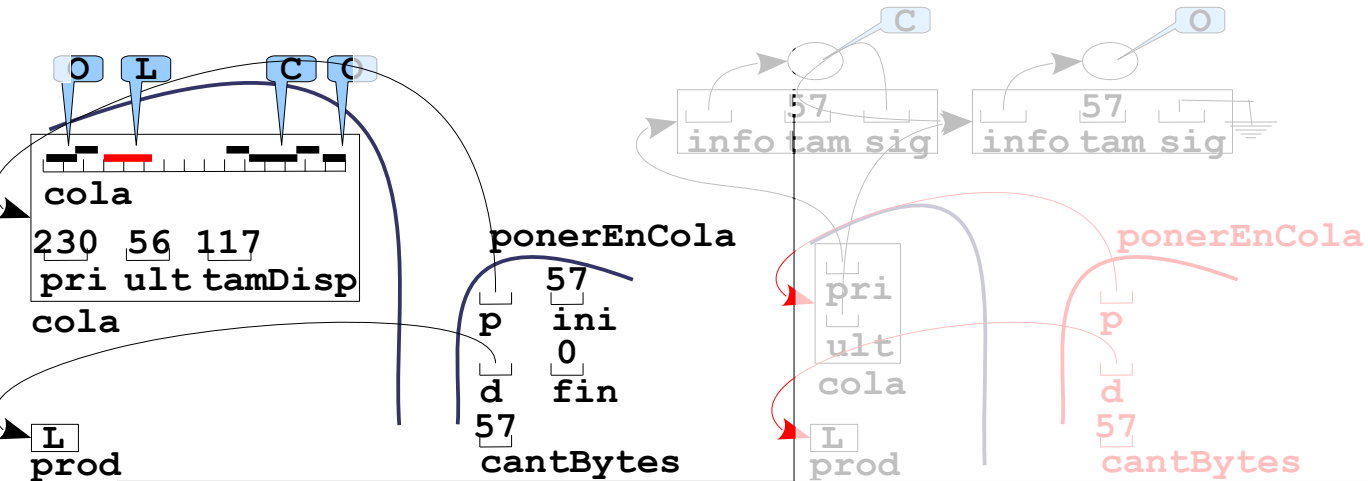
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



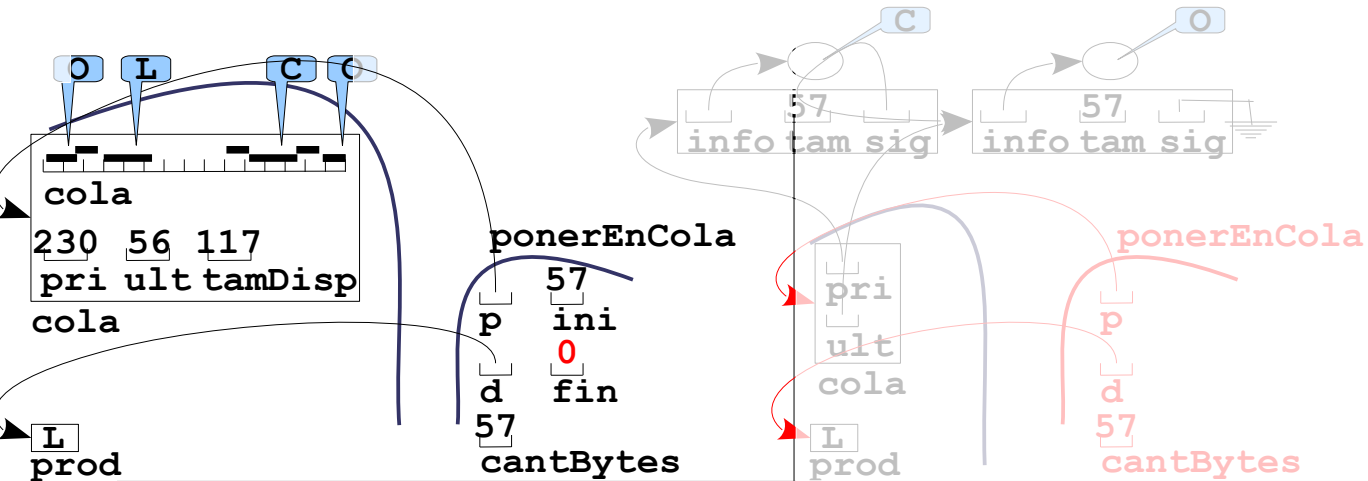
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



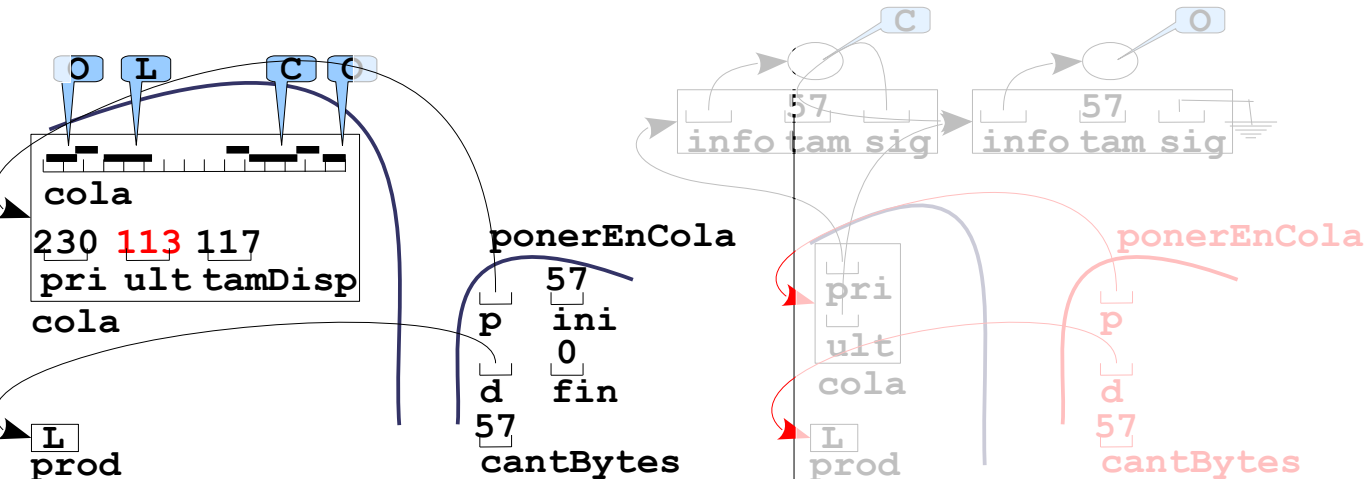
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
32
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola + p->ult, &cantBytes, ini);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
```

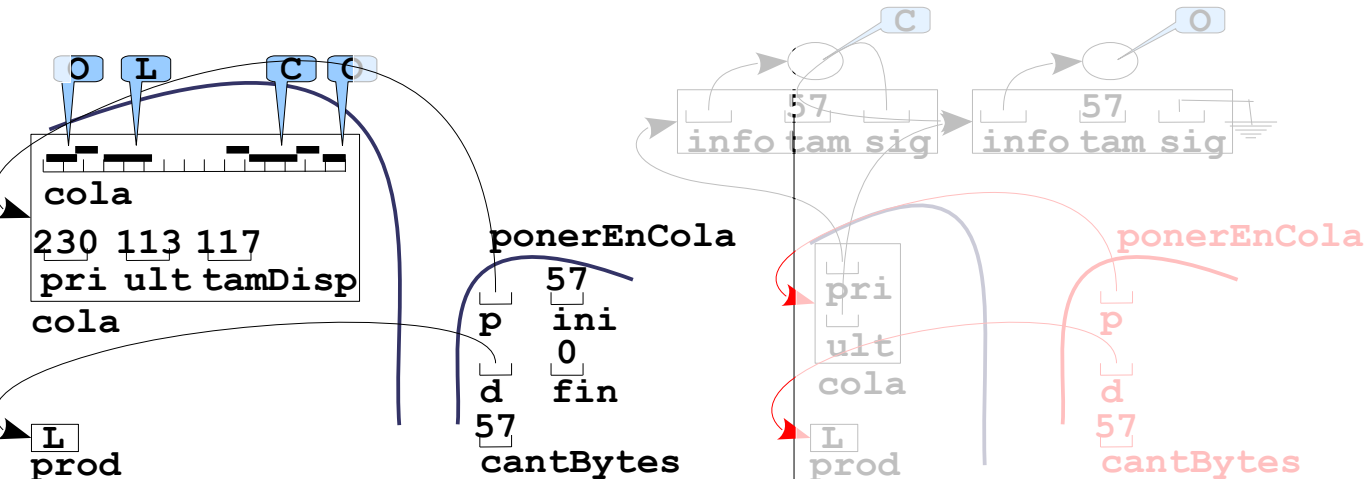
to(&prod))

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

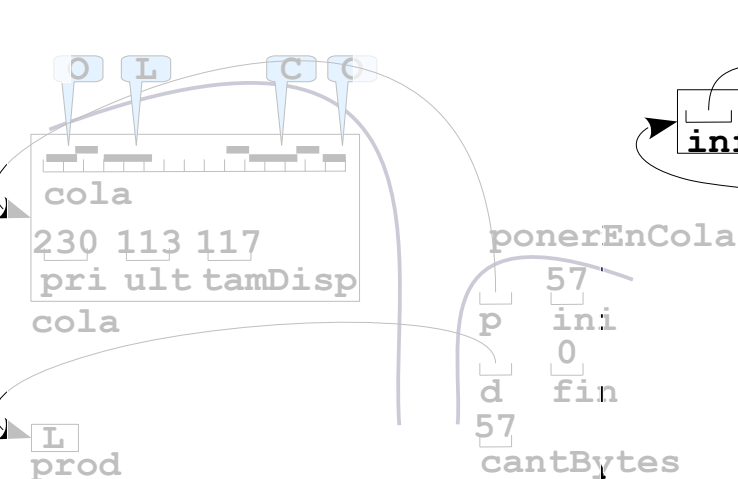


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
22 32 if((fin = sizeof(cantBytes) - ini) != 0)
23 33     memcpy(p->cola, ((char *)&cantBytes) + ini, fin);
24 34 p->ult = fin ? fin : p->ult + ini;
25 35 if((ini = minimo(cantBytes, TAM_COLA - p->ult)) != 0)
26 36     memcpy(p->cola + p->ult, d, ini);
27 37 if((fin = cantBytes - ini) != 0)
28 38     memcpy(p->cola, ((char *)d) + ini, fin);
29 39 p->ult = fin ? fin : p->ult + ini;
30 40 return 1;
31 41 }
```

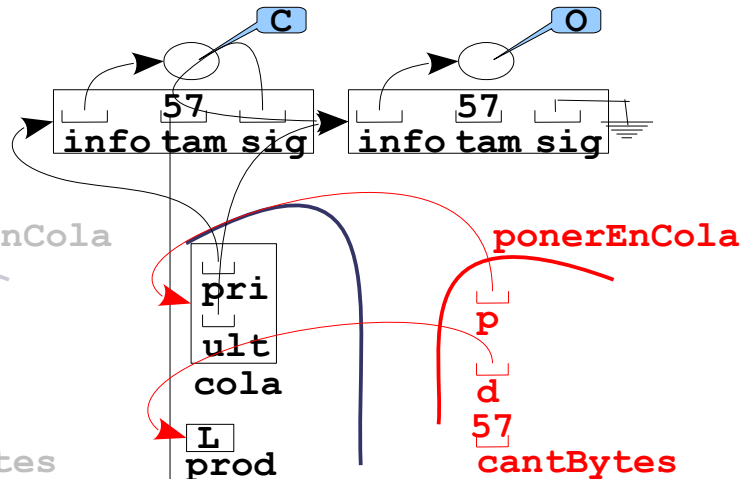
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



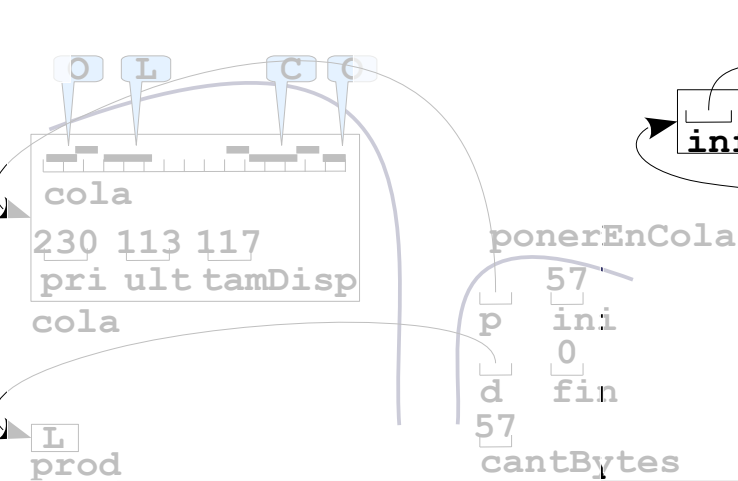
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87  vo:
88  {
89
90
91
92
93
94
95

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21  int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22  {
23      tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25      if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26      {
27          free(nue);
28          return 0;
29      }
30      memcpy(nue->info, d, cantBytes);
31
```

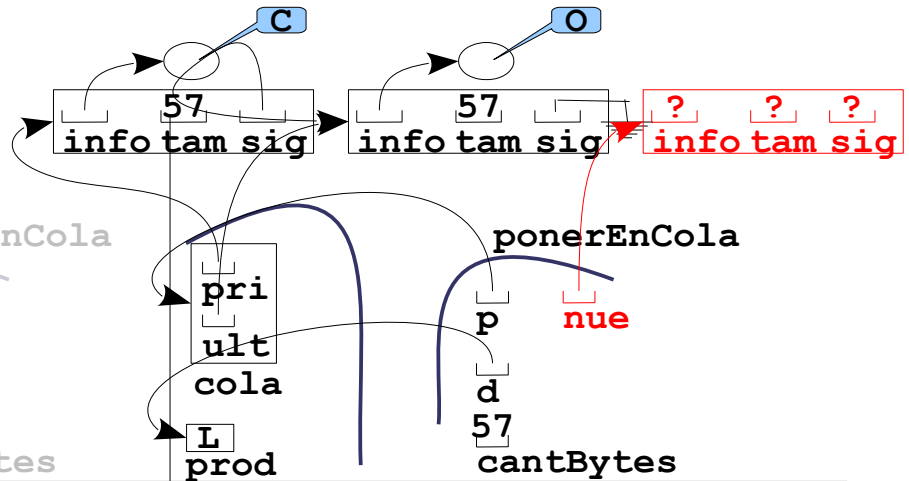

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



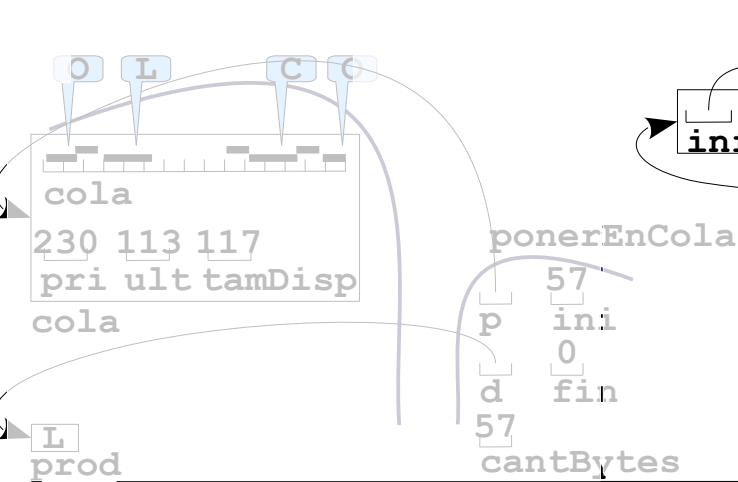
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo
88 {
89
90
91
92
93
94
95

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
31     return 1;
32 }
```

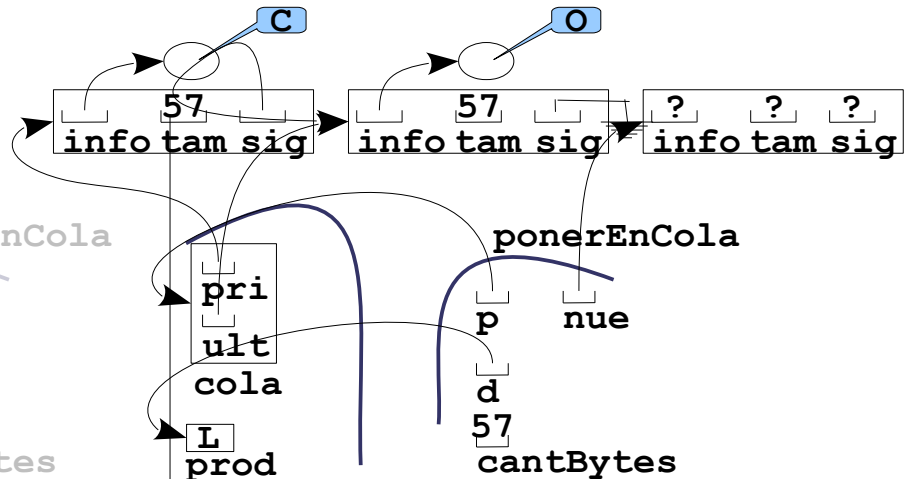
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



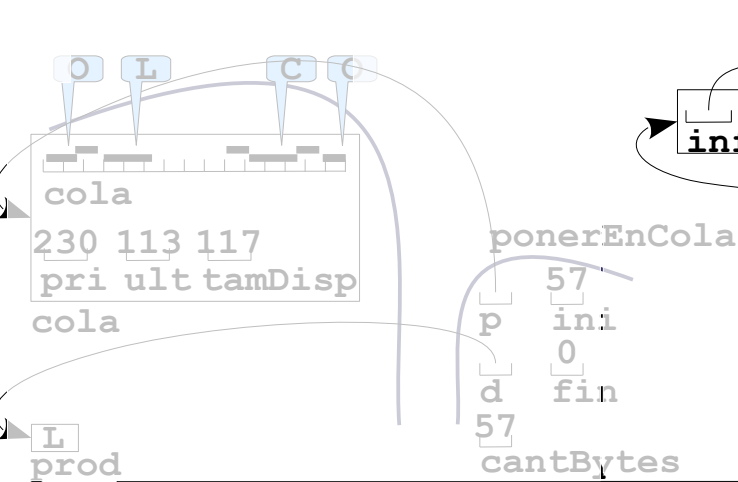
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void
88 {
89
90
91
92
93
94
95

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
31     return 1;
```

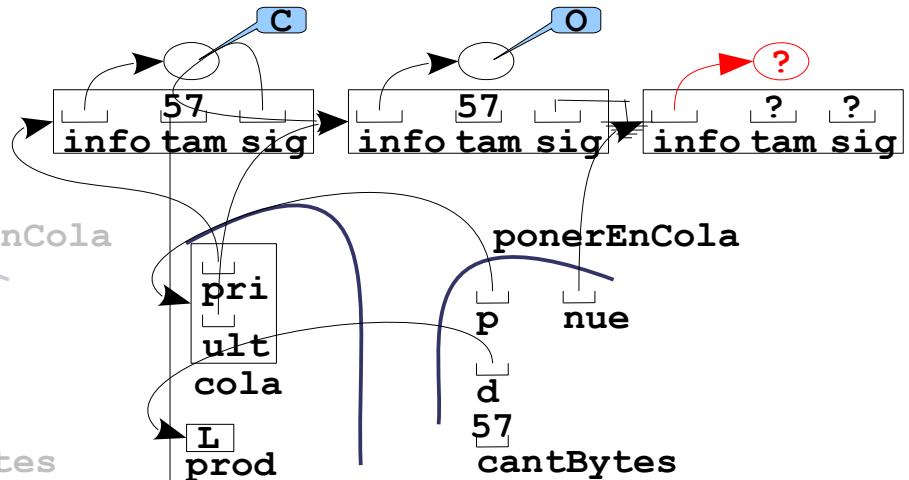
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



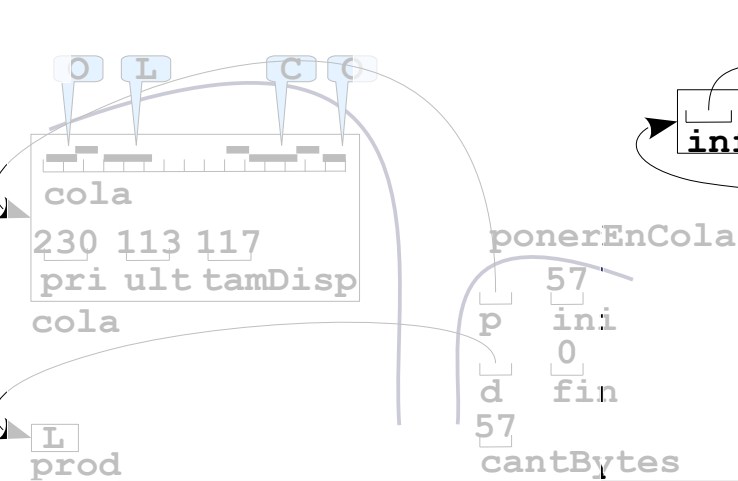
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87  vo:
88  {
89
90
91
92
93
94
95

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21  int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22  {
23      tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25      if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26      {
27          free(nue);
28          return 0;
29      }
30      memcpy(nue->info, d, cantBytes);
31
```

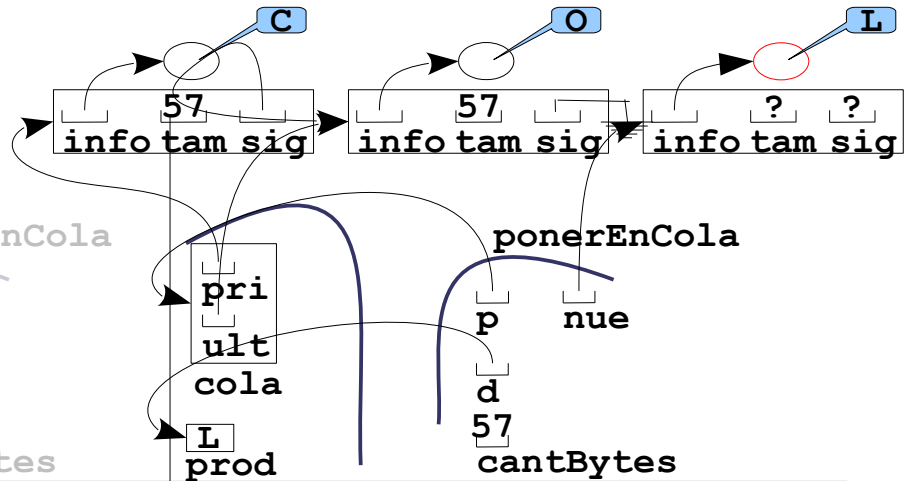
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



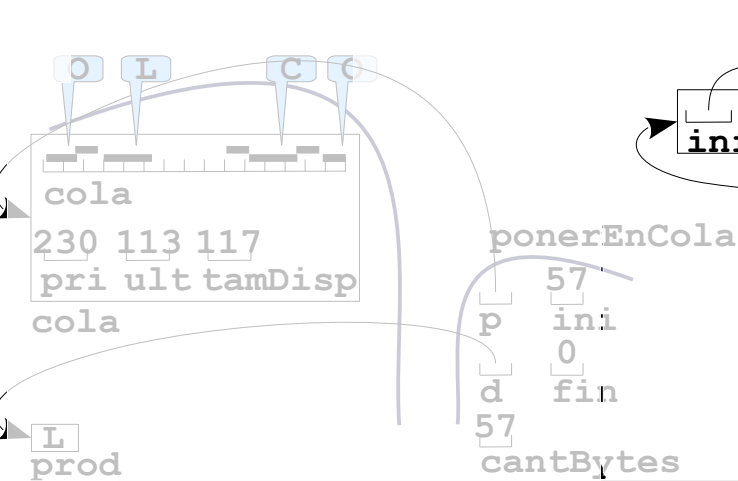
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo
88 {
89
90
91
92
93
94
95

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
21 int ponerEnCola(tCola *p, const void *d, unsigned cantBytes)
22 {
23     tNodo *nue = (tNodo *) malloc(sizeof(tNodo));
24
25     if(nue == NULL || (nue->info = malloc(cantBytes)) == NULL)
26     {
27         free(nue);
28         return 0;
29     }
30     memcpy(nue->info, d, cantBytes);
```

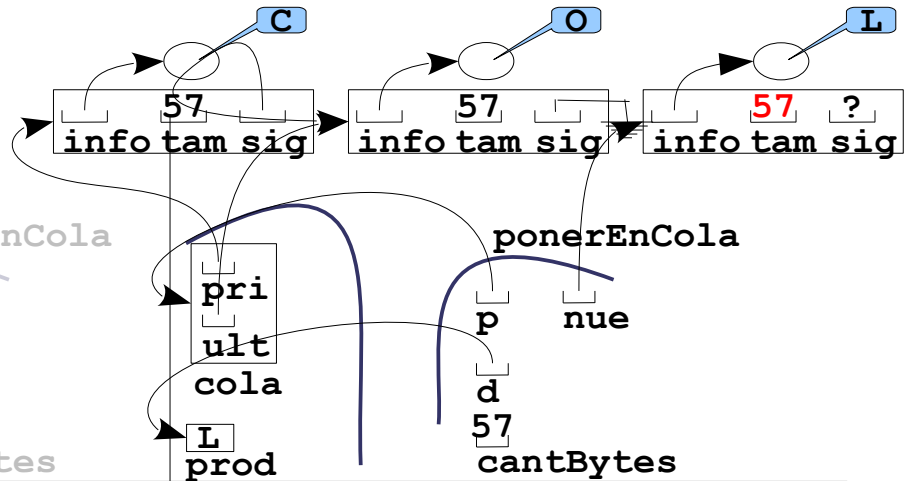
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

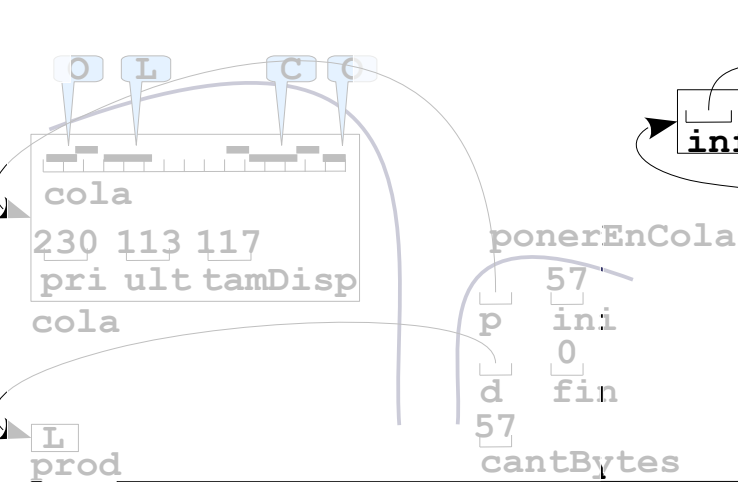


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40     return 1;
98 41
99 42 }
```

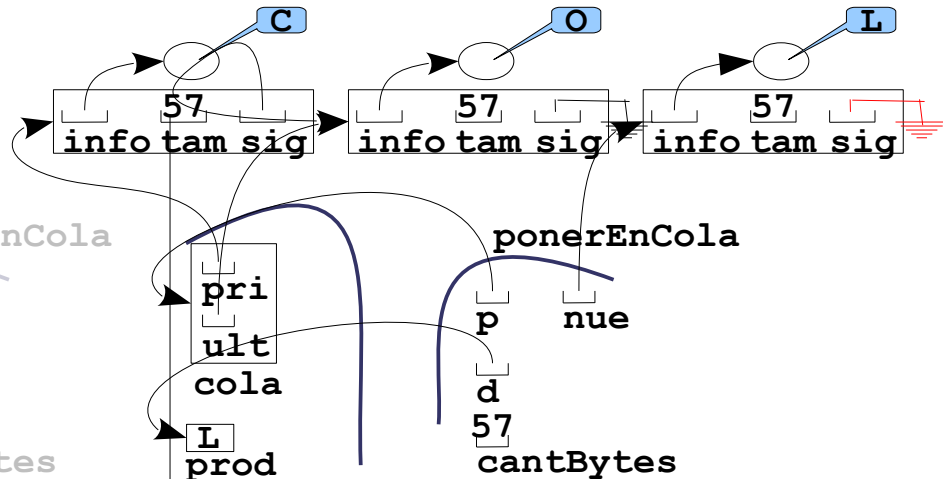
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

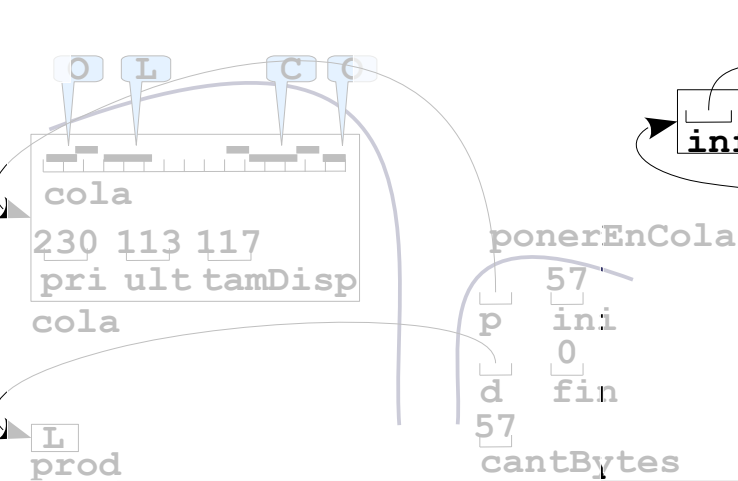


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36 else
94 37     p->pri = nue;
95 38 p->ult = nue;
    39 return 1;
    40 }
```

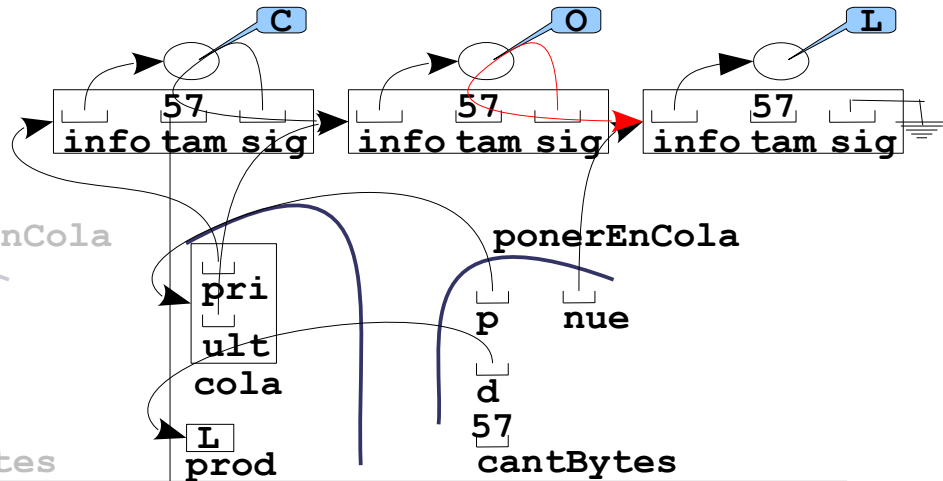
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

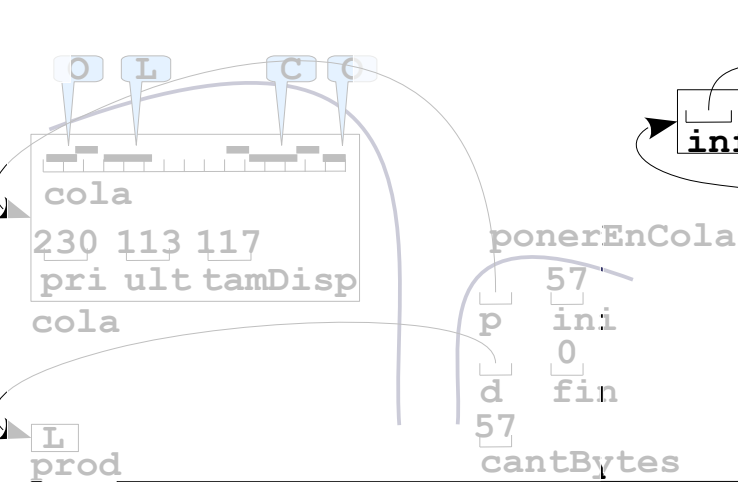


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 vo: main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
88 31 memcpy(nue->info, d, cantBytes);
89 32 nue->sig = NULL;
90 33 if(p->ult)
91 34     p->ult->sig = nue;
92 35 else
93 36     p->pri = nue;
94 37 p->ult = nue;
95 38 return 1;
39 }
```

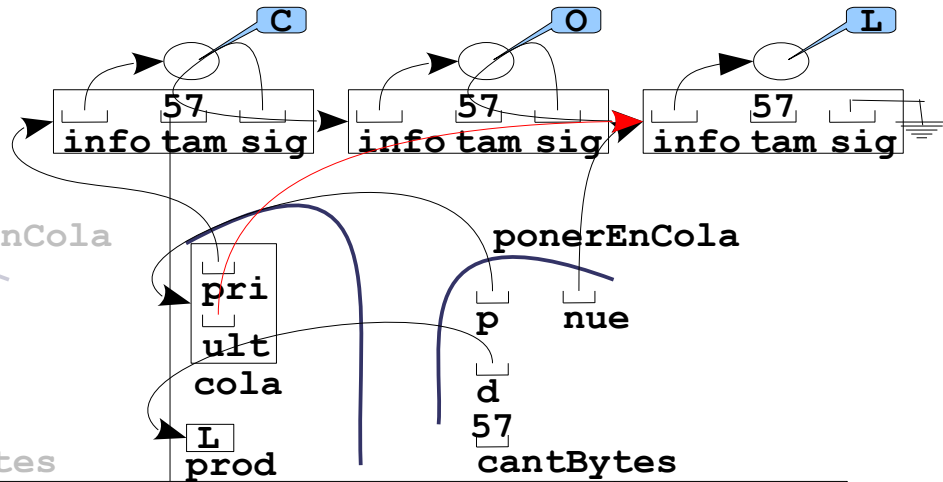
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

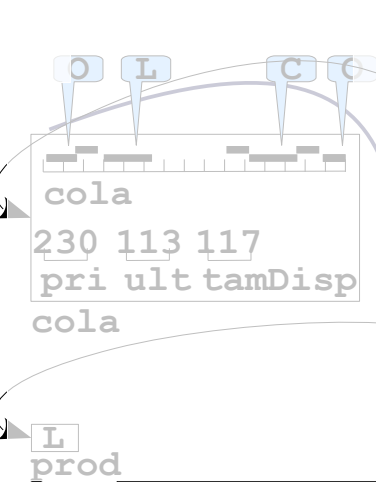


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40     return 1;
98 41 }
```

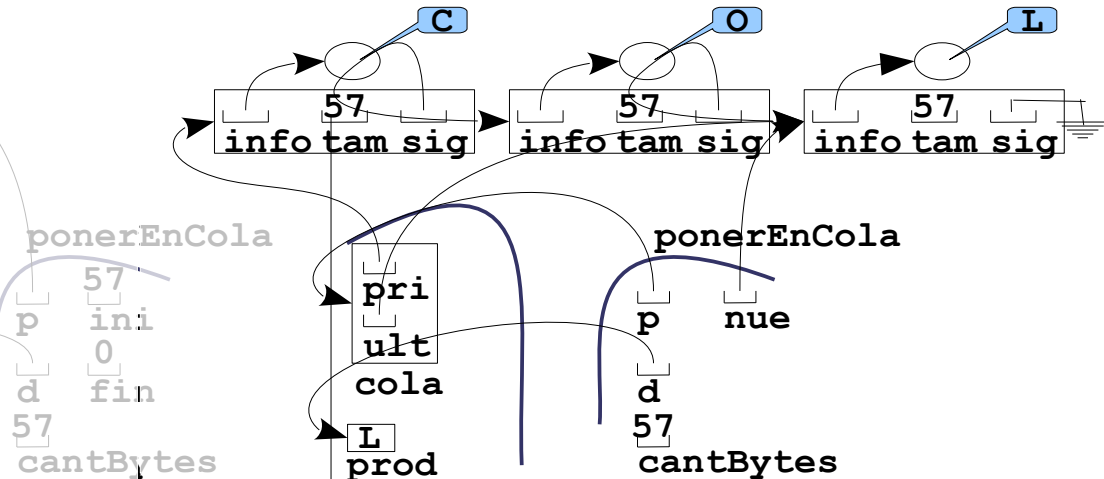

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

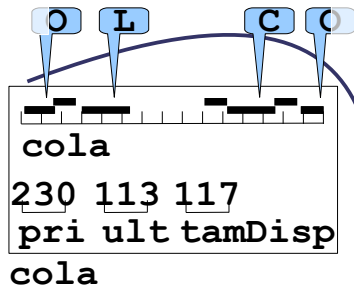


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 30 memcpy(nue->info, d, cantBytes);
88 31
89 32 nue->sig = NULL;
90 33
91 34 if(p->ult)
92 35     p->ult->sig = nue;
93 36
94 37 else
95 38     p->pri = nue;
96 39     p->ult = nue;
97 40
98 41 return 1;
99 42 }
```

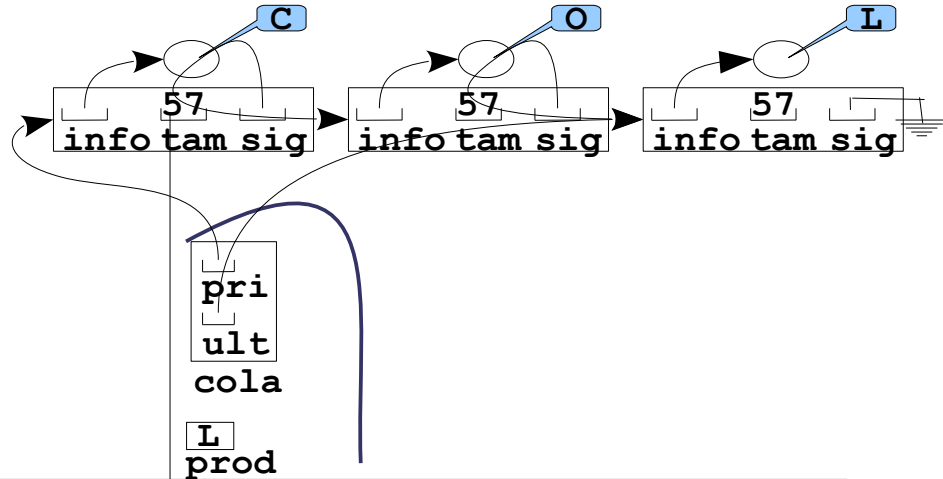
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

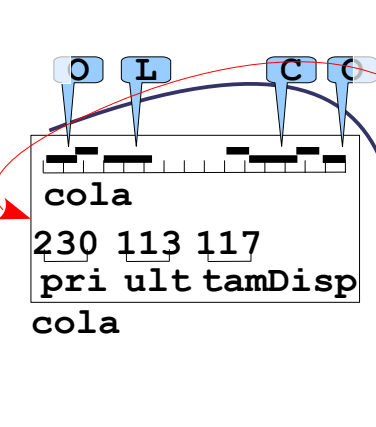


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
87 void probarPonerYSacarDeCola(void)
88 {
89     tProd prod;
90     tCola cola;
91
92     crearCola(&cola);
93     while(!colaLlena(&cola, sizeof(prod)) && ingresarProducto(&prod))
94         if(!ponerEnCola(&cola, &prod, sizeof(prod)))
95             puts("ERROR - inesperado de cola llena");
```

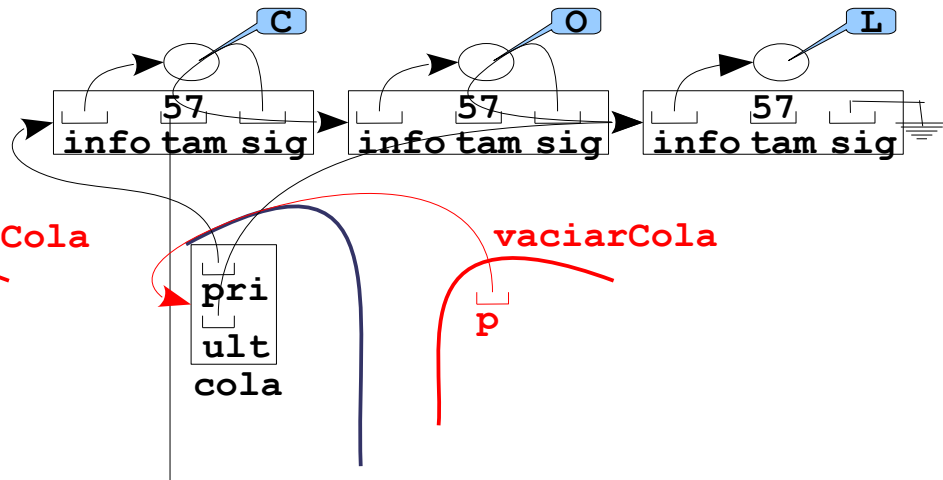

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



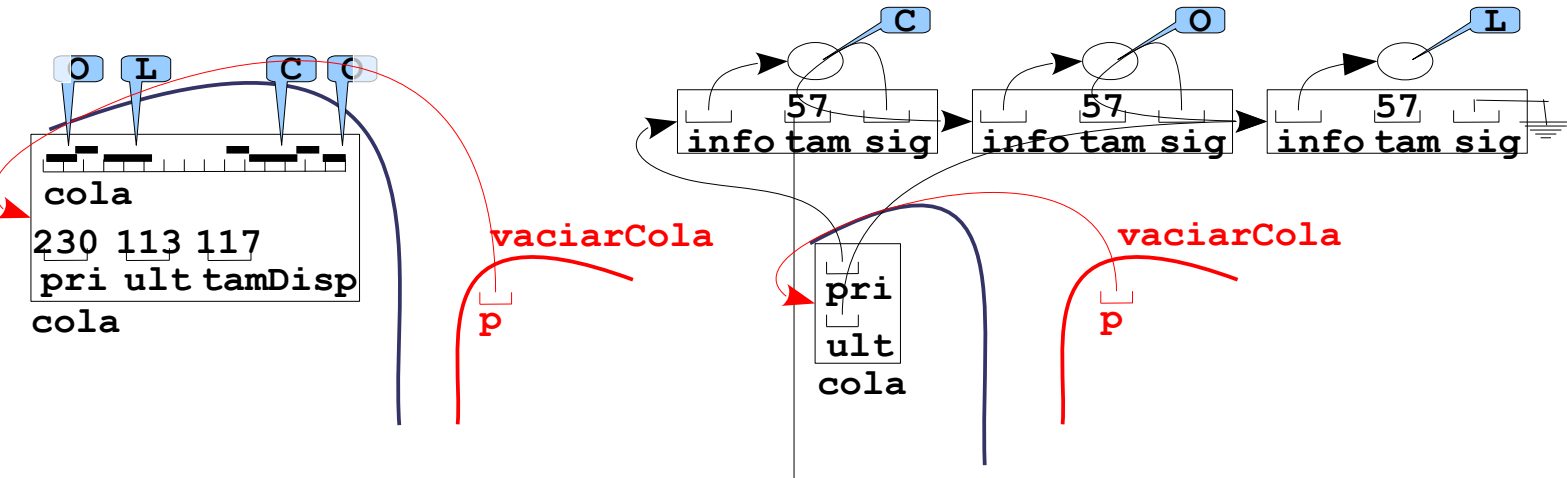
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c
97 vaciarCola(&cola);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



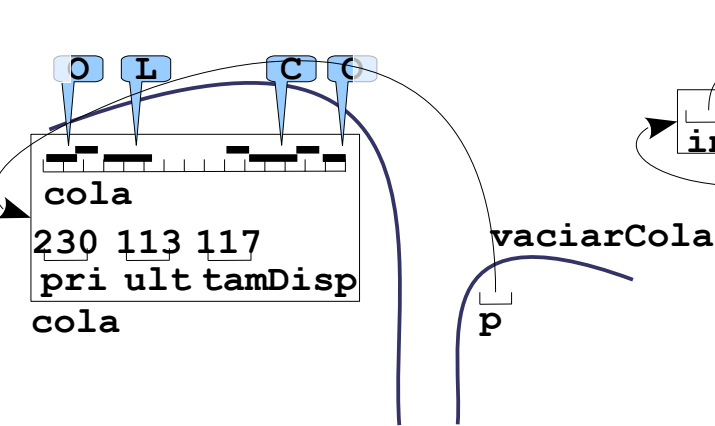
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

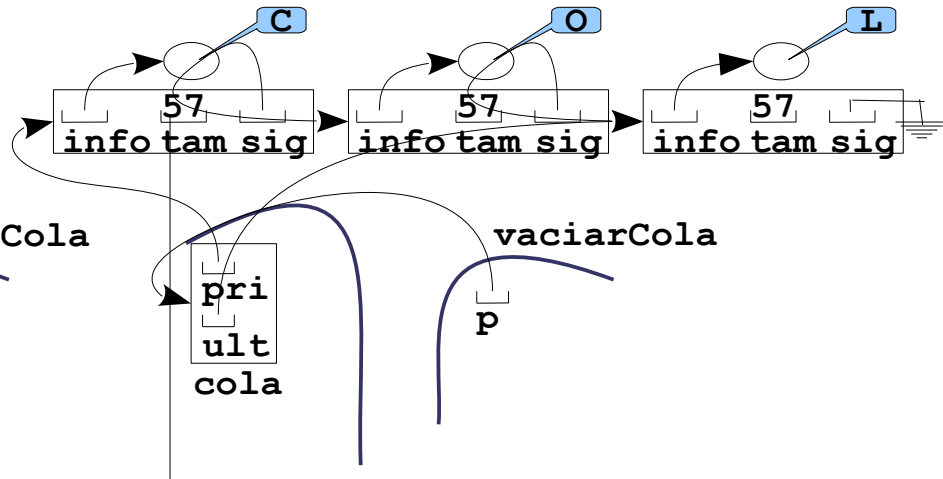
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



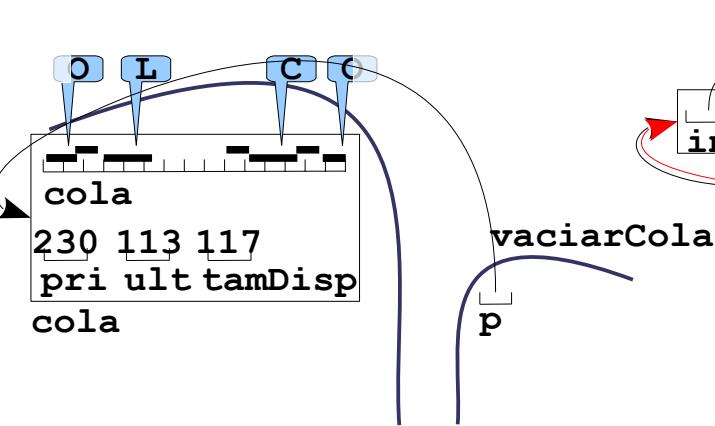
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

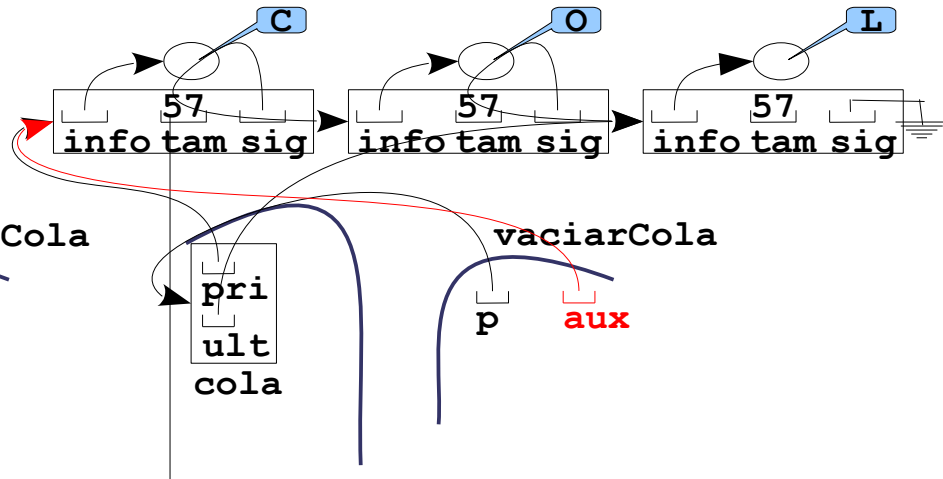
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



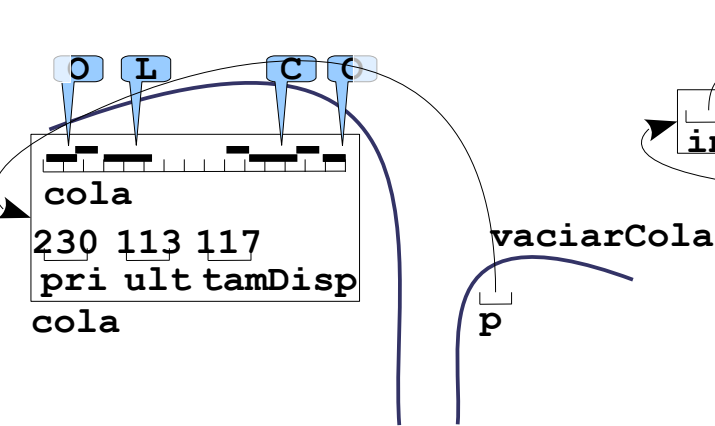
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

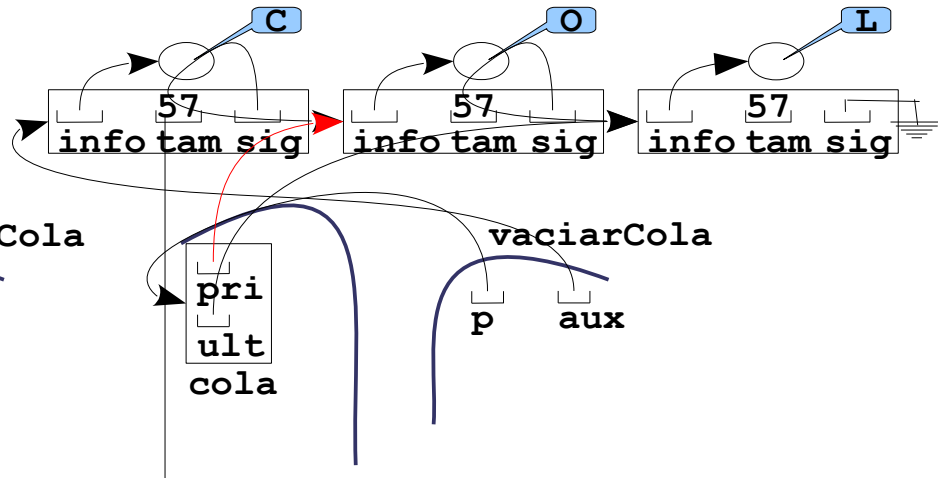
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



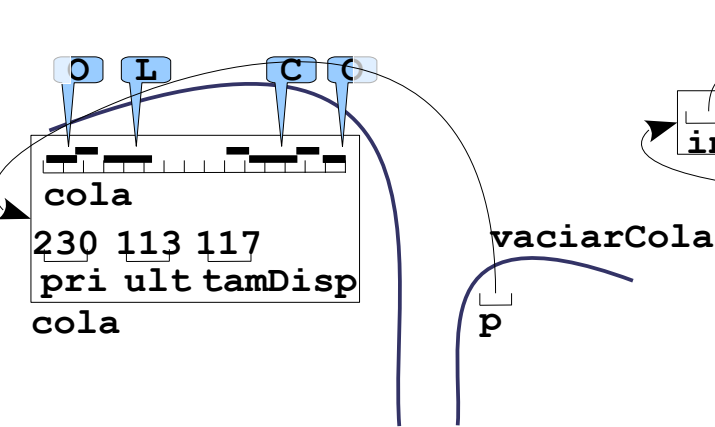
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

main.c x main.h x productos.c x productos.h x cola.c x
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

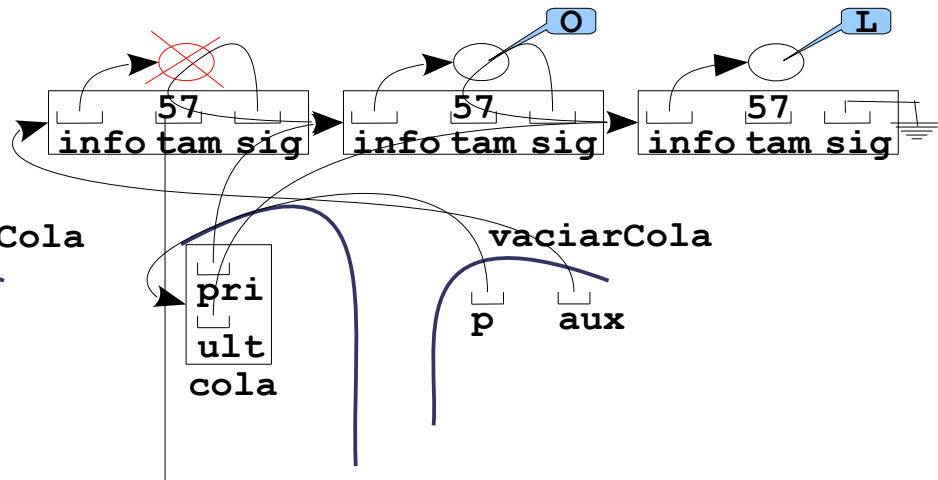

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



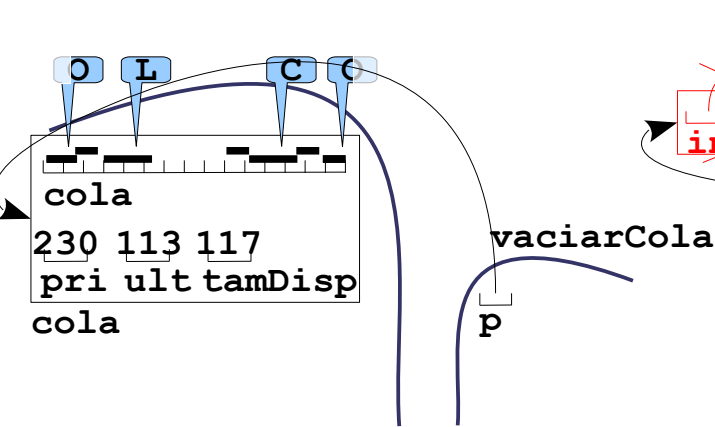
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

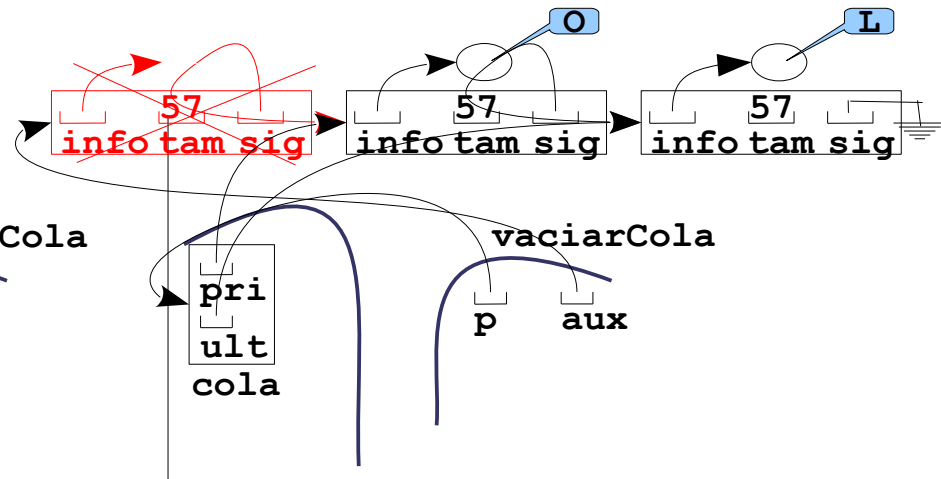
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



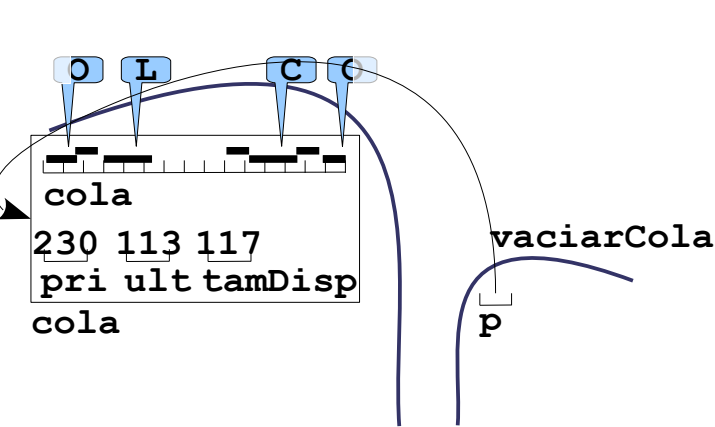
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

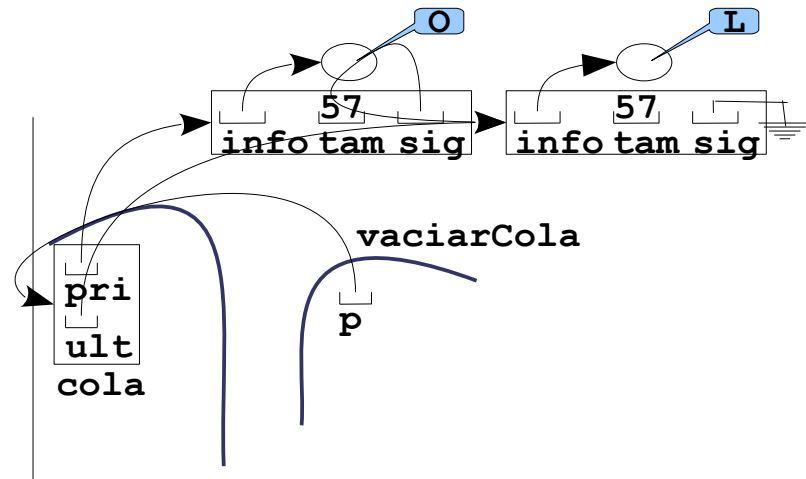
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



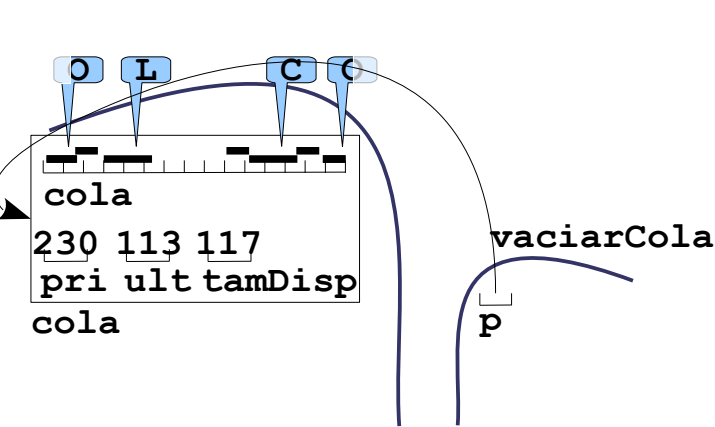
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

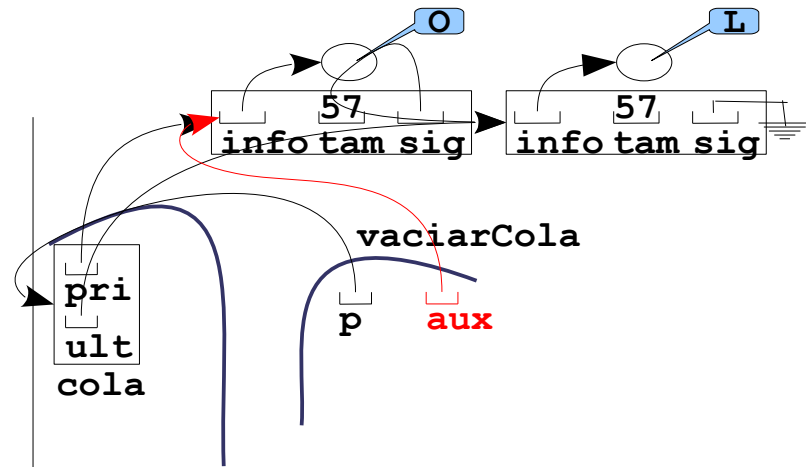
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



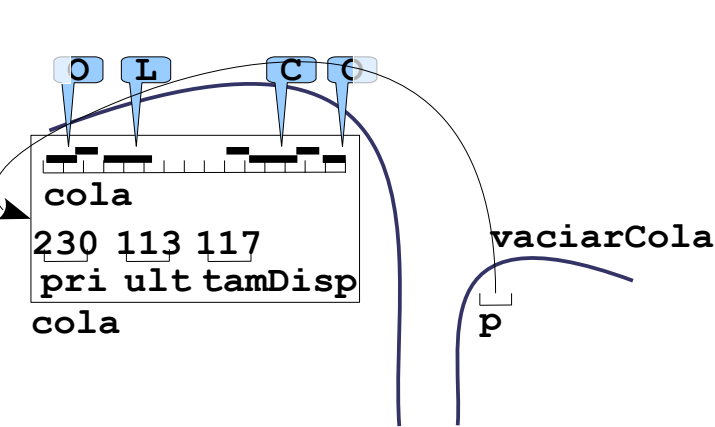
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

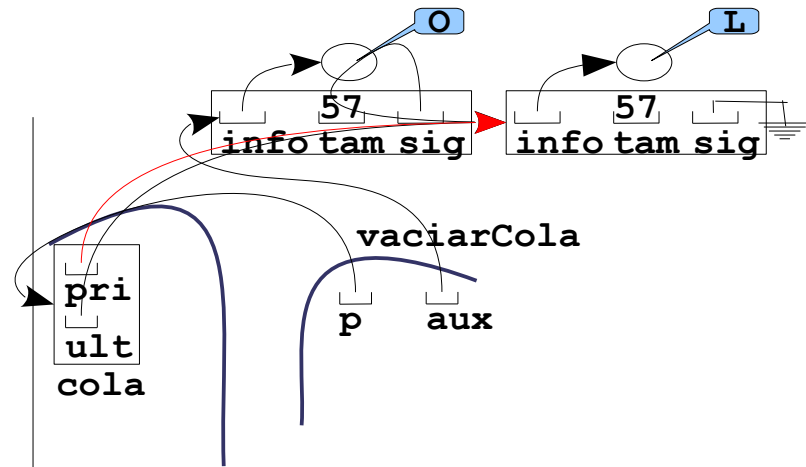
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



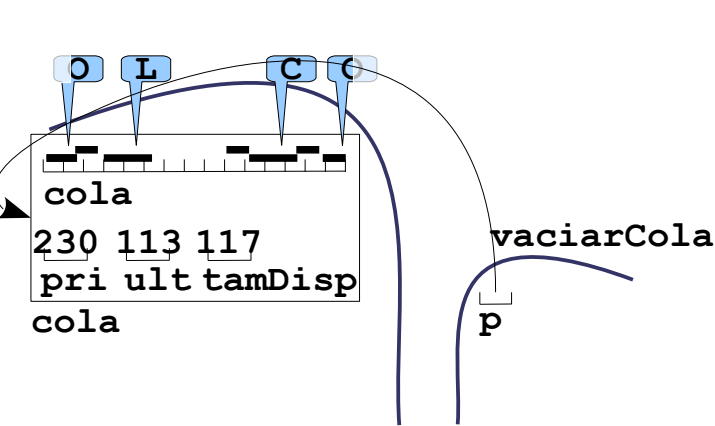
```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

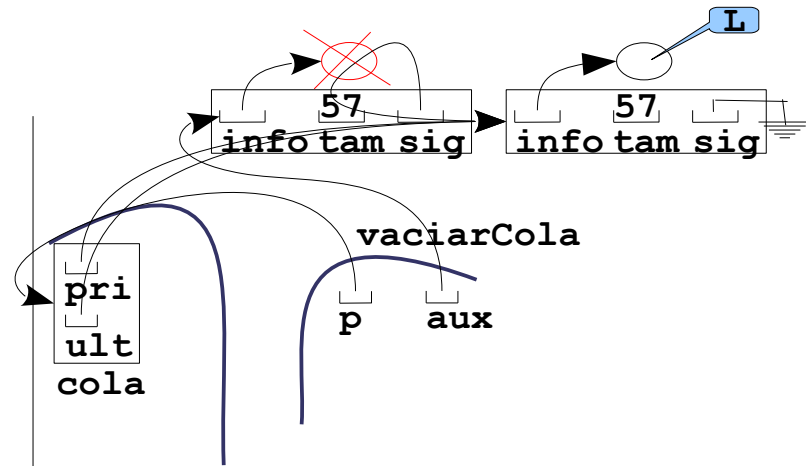
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

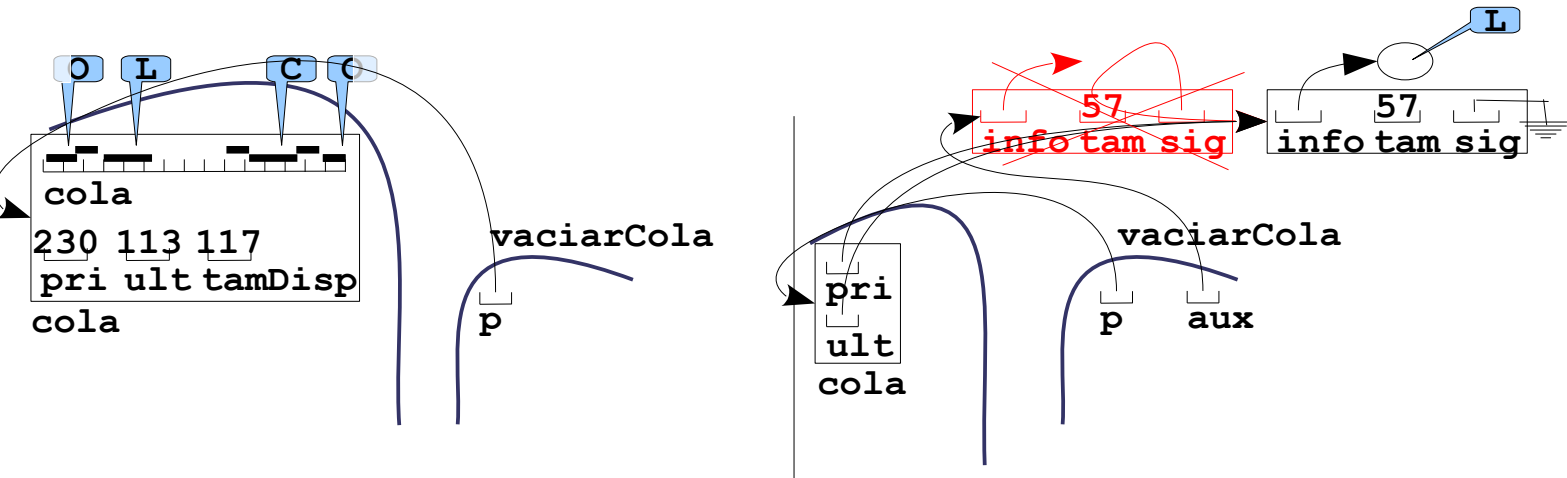
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

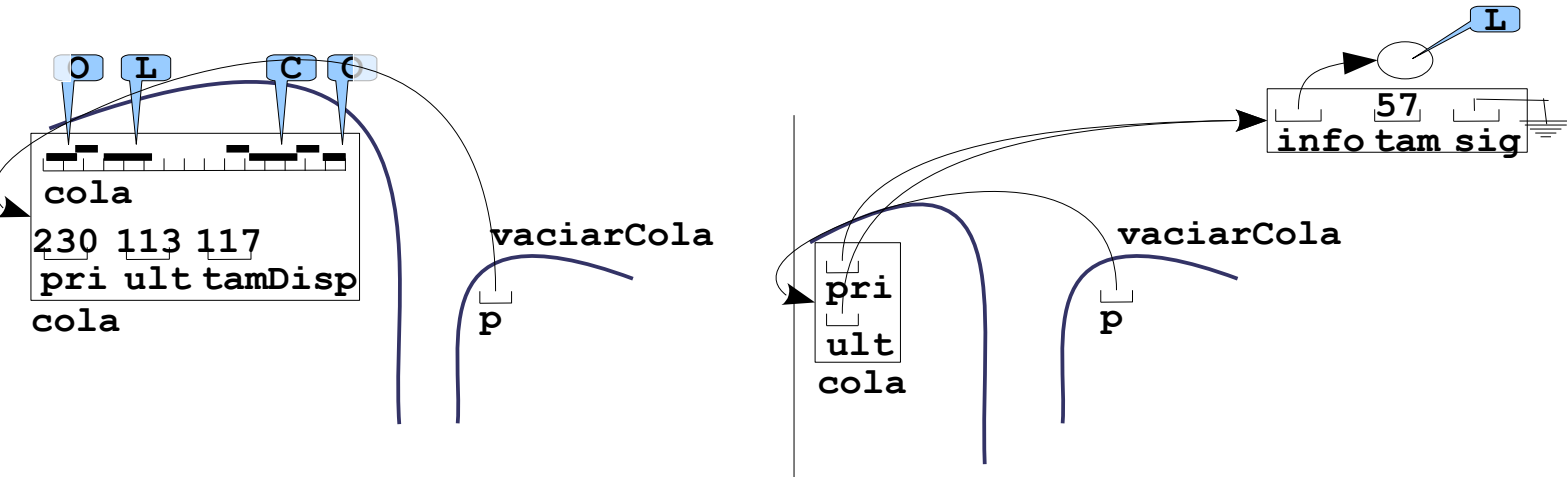
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

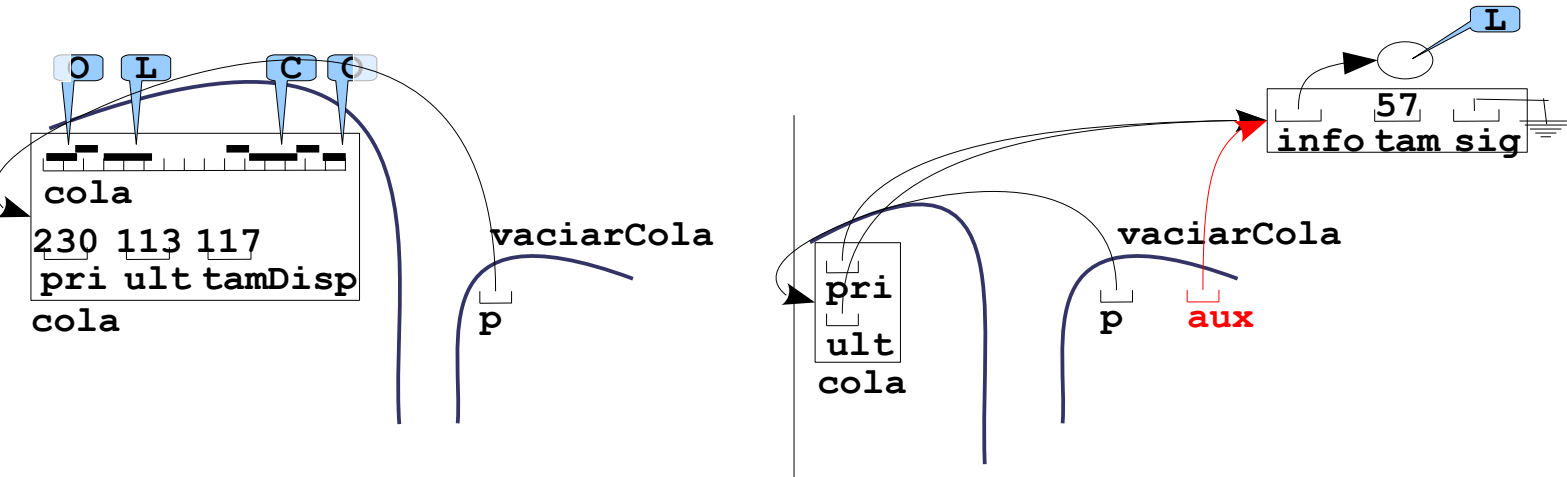
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

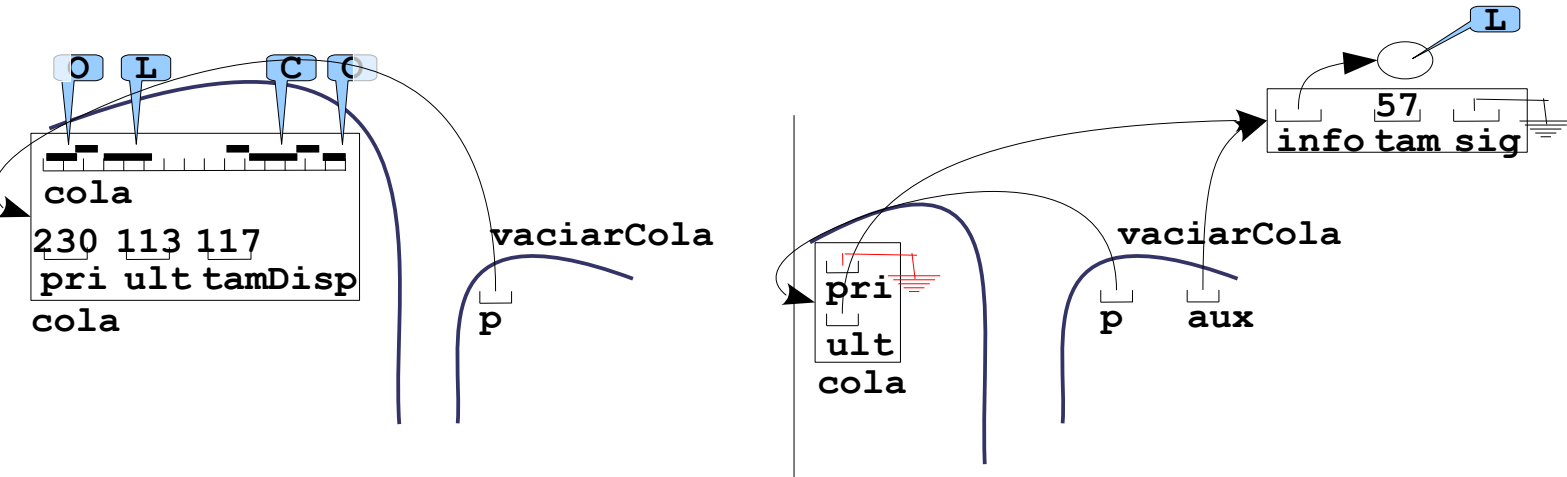
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

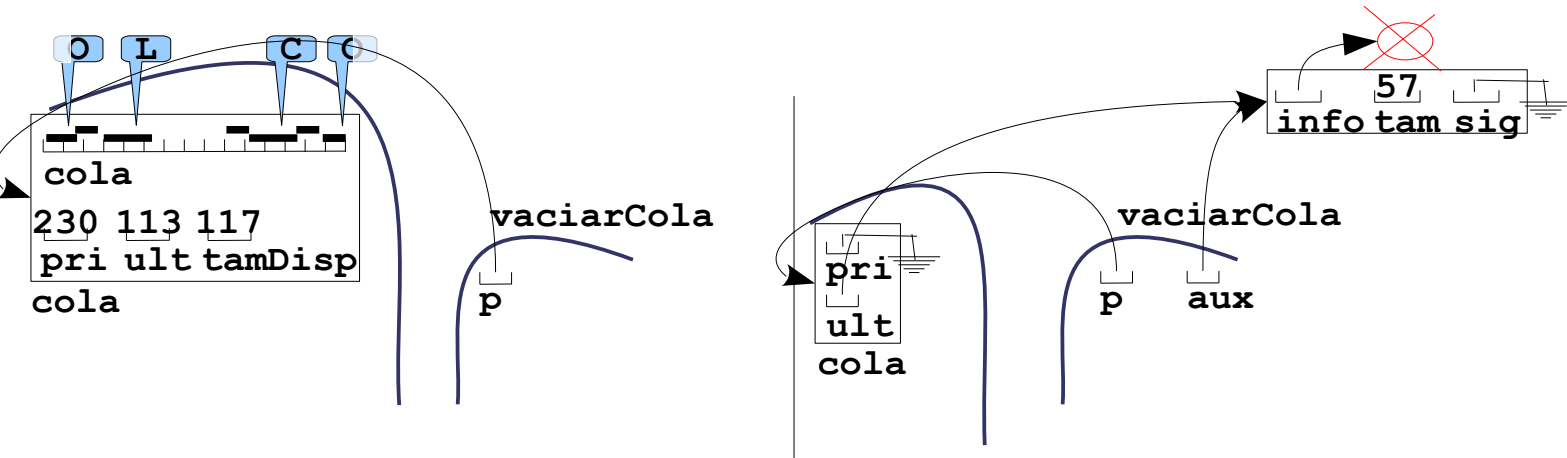
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

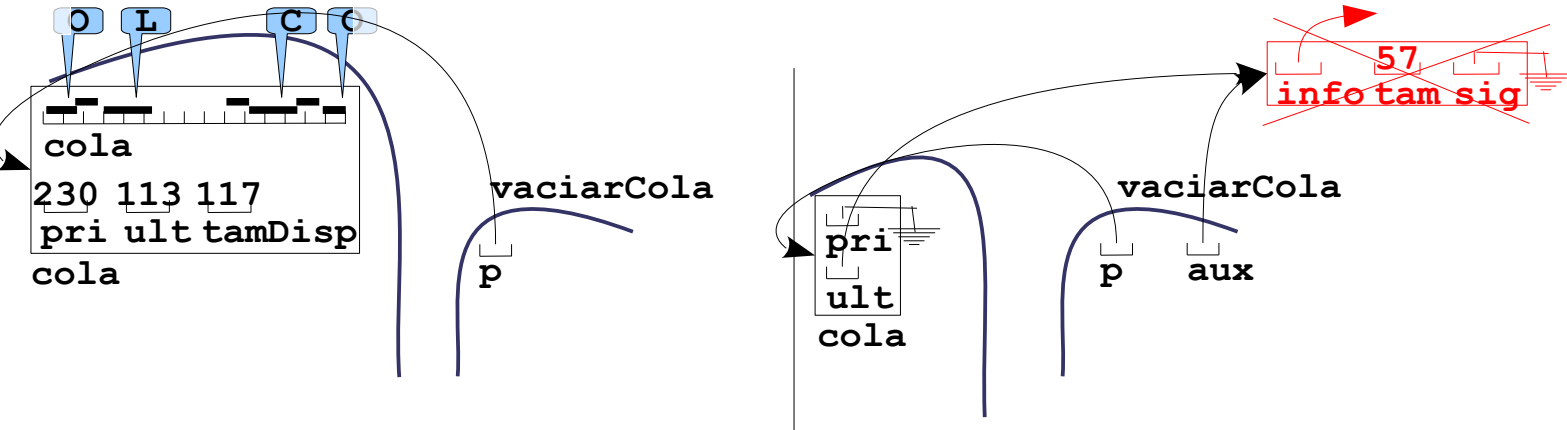
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

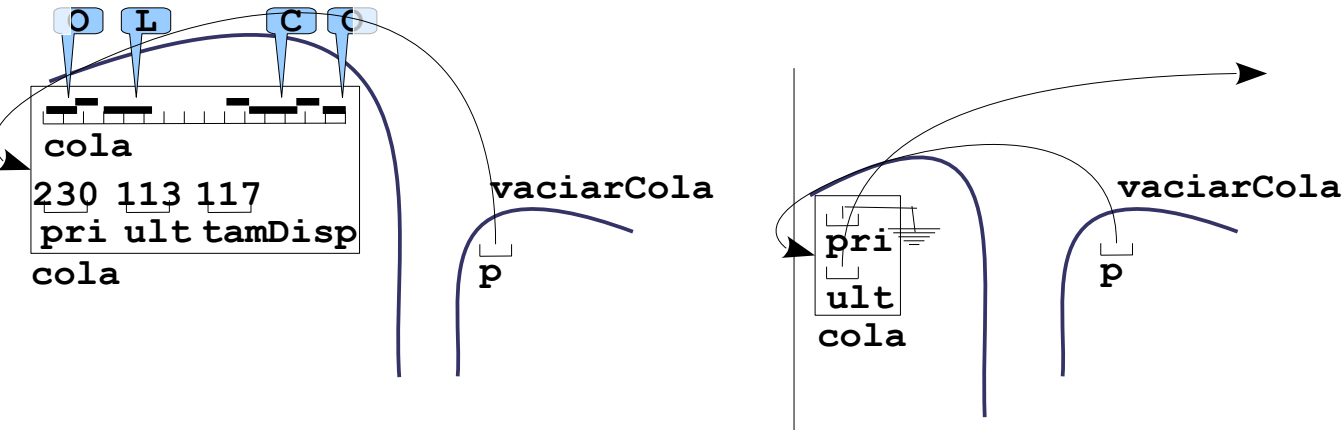
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

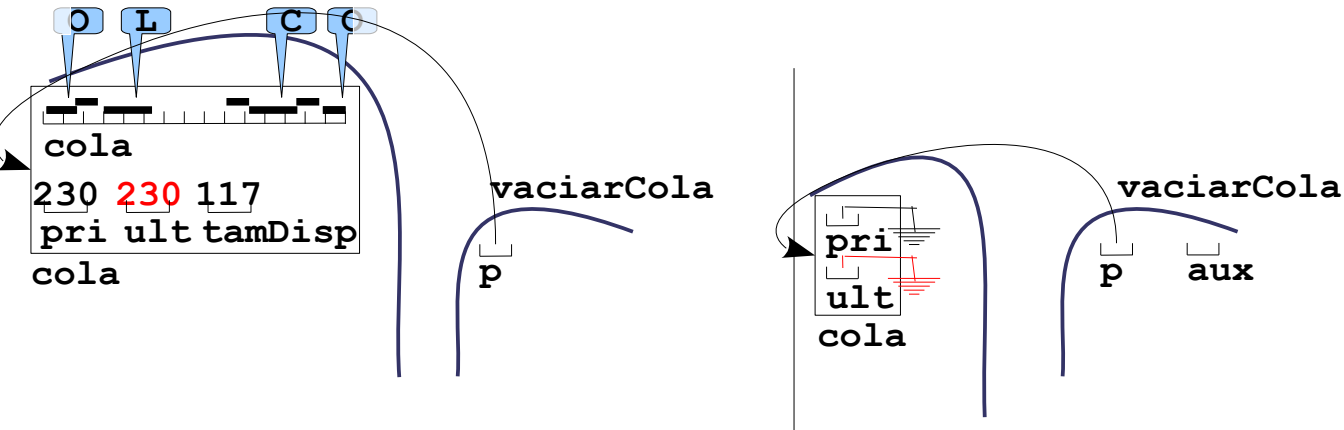
main.c x main.h x productos.c x productos.h x cola.c x col
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }

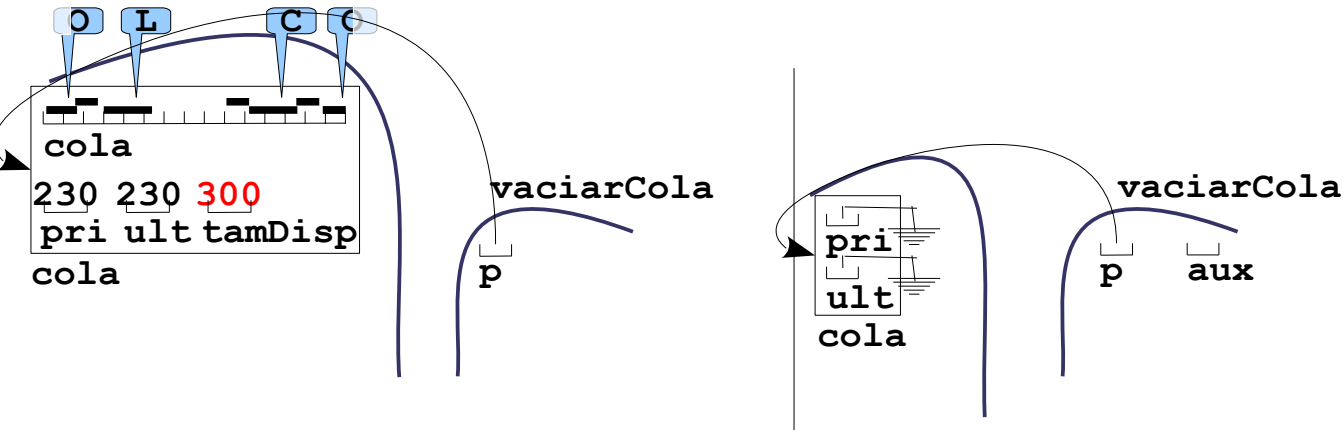
main.c x main.h x productos.c x productos.h x cola.c x
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
93 void vaciarCola(tCola *p)
94 {
95     p->ult = p->pri;
96     p->tamDisp = TAM_COLA;
97 }
```

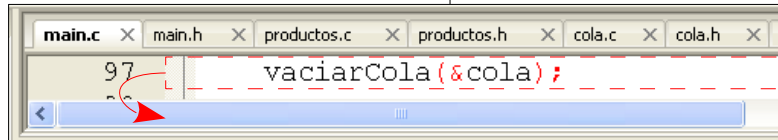
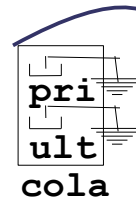
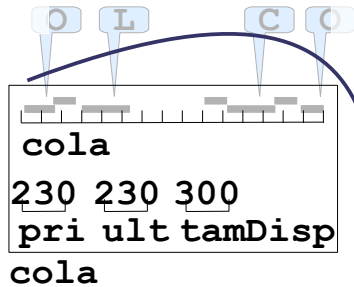
```
68 void vaciarCola(tCola *p)
69 {
70     while(p->pri)
71     {
72         tNodo *aux = p->pri;
73         p->pri = aux->sig;
74         free(aux->info);
75         free(aux);
76     }
77     p->ult = NULL;
78 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

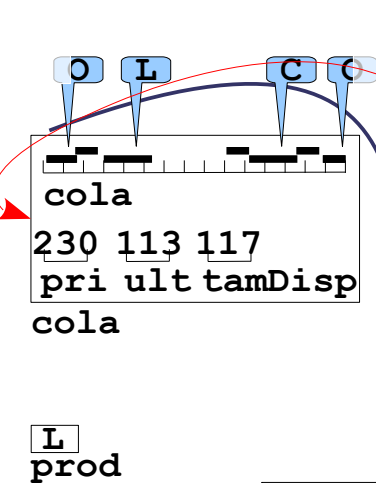
Implementación dinámica



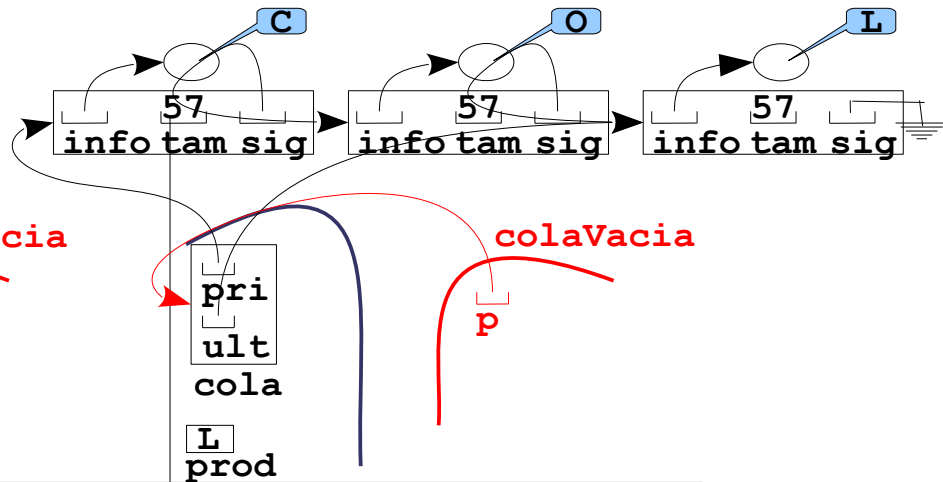
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



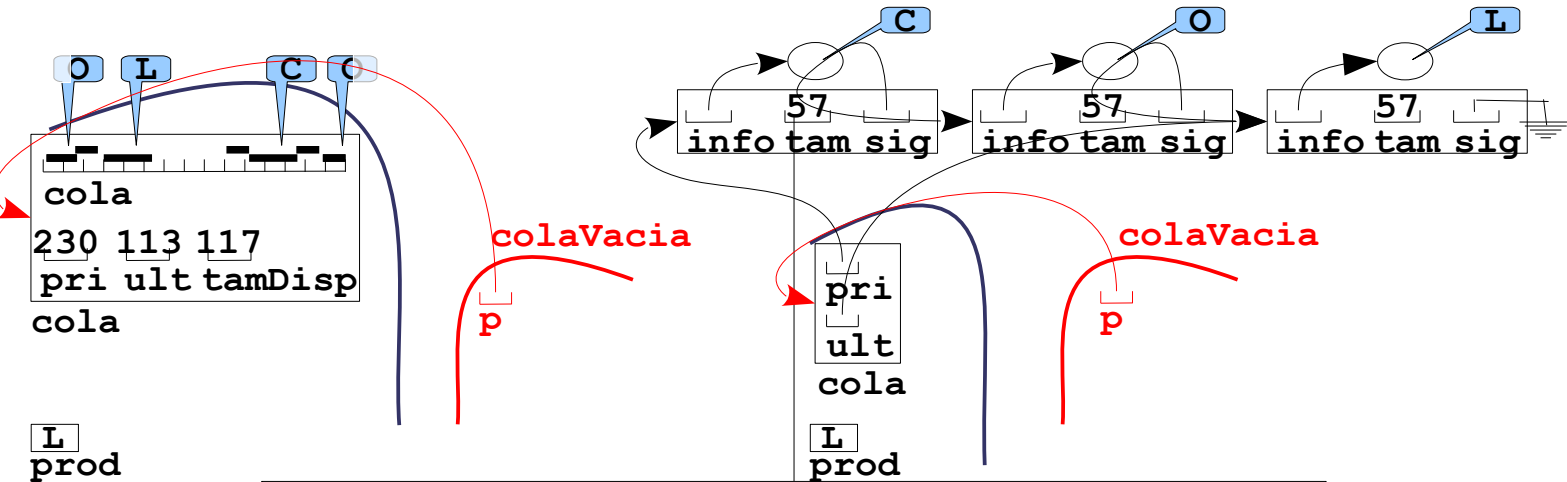
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102
```

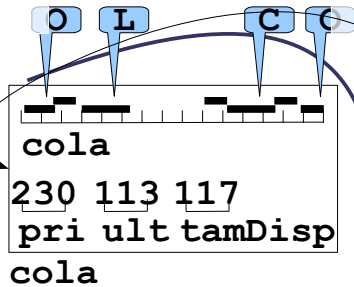
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
65 int colaVacia(const tCola *p)
66 {
67     return p->tamDisp == TAM_COLA;
68 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
49 int colaVacia(const tCola *p)
50 {
51     return p->pri == NULL;
52 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

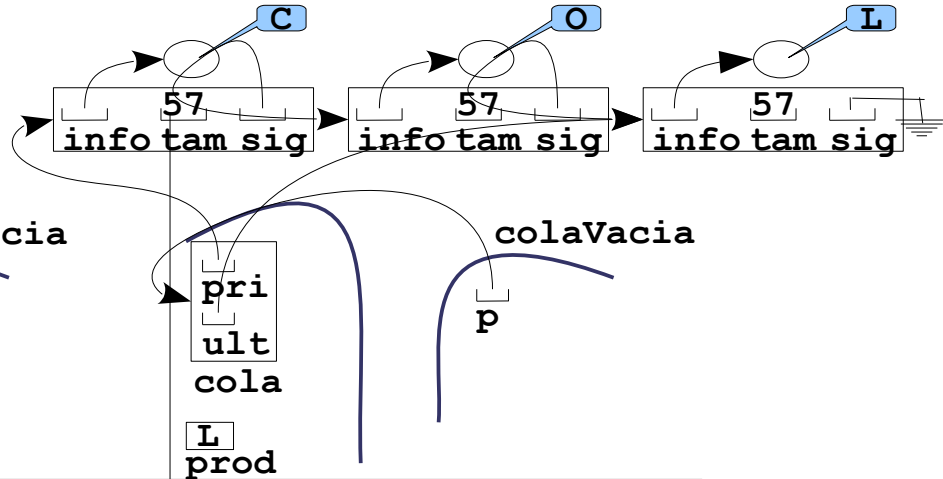


`L`
`prod`

`colaVacia`

`p`

Implementación dinámica



`colaVacia`

`p`

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102
```

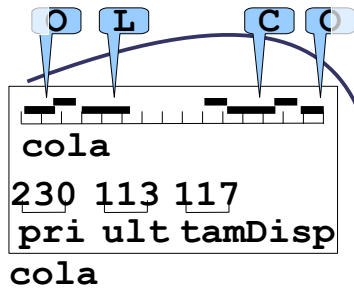
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
65 int colaVacia(const tCola *p)
66 {
67     return p->tamDisp == TAM_COLA;
68 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
49 int colaVacia(const tCola *p)
50 {
51     return p->pri == NULL;
52 }
```

Tipos de Datos Abstractos

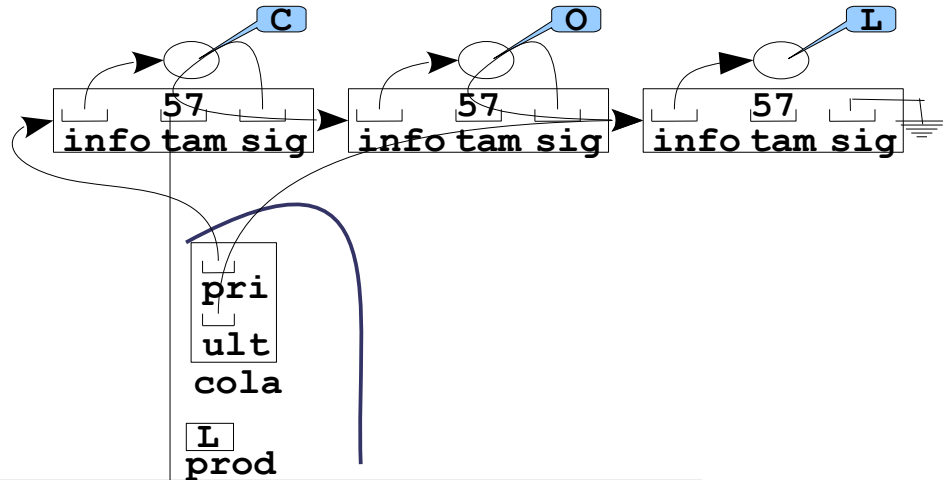
Tipo de dato Cola.

Implementación estática



L
prod

Implementación dinámica



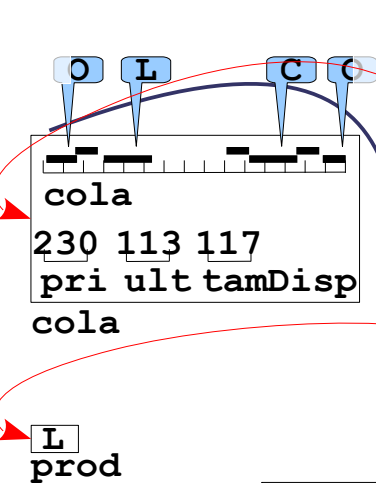
L
prod

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

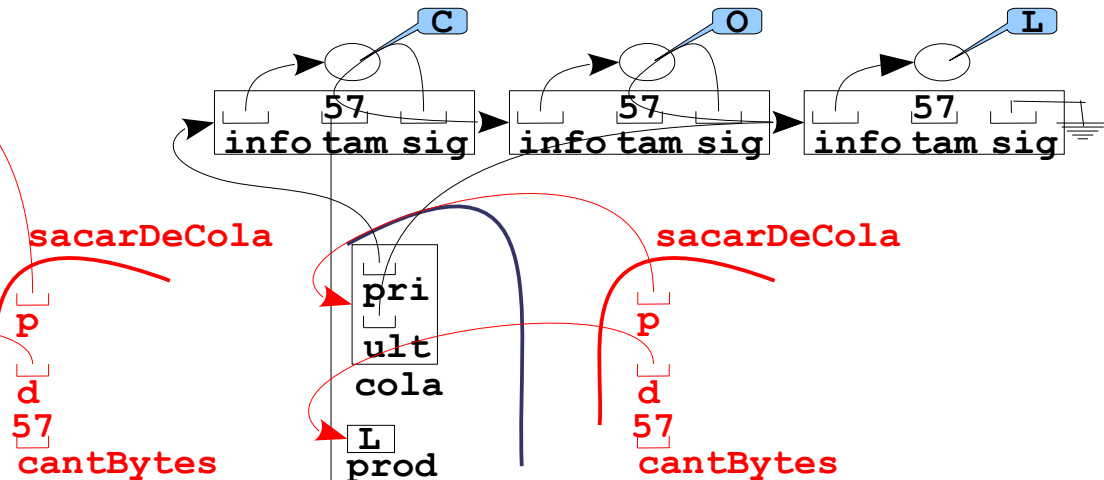
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



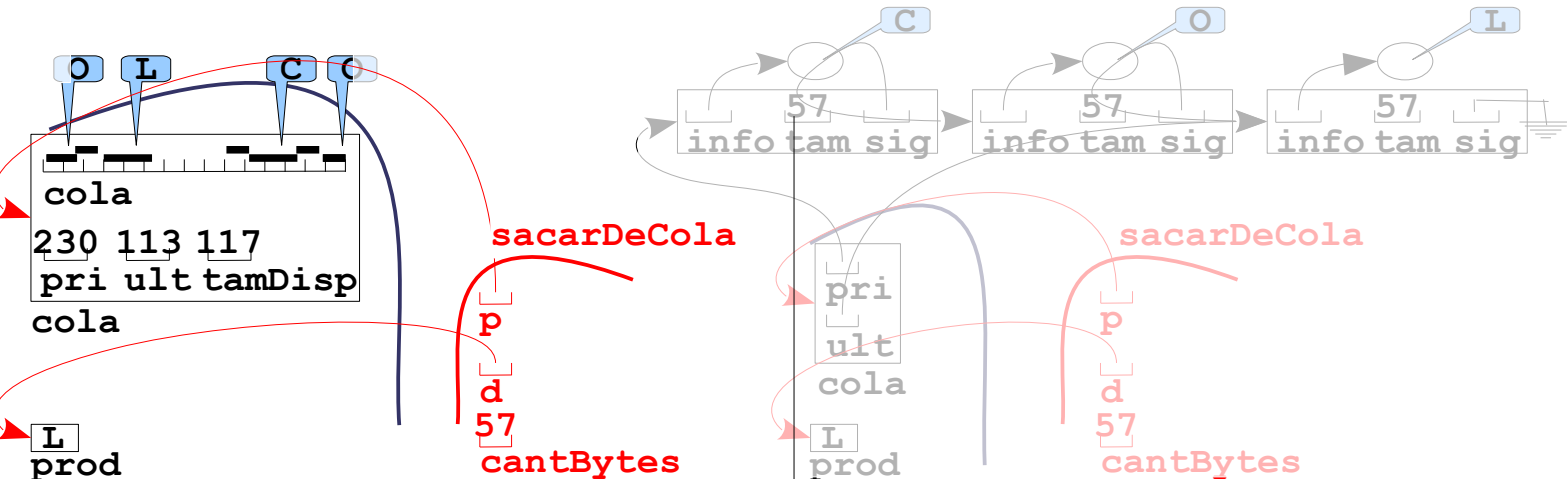
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



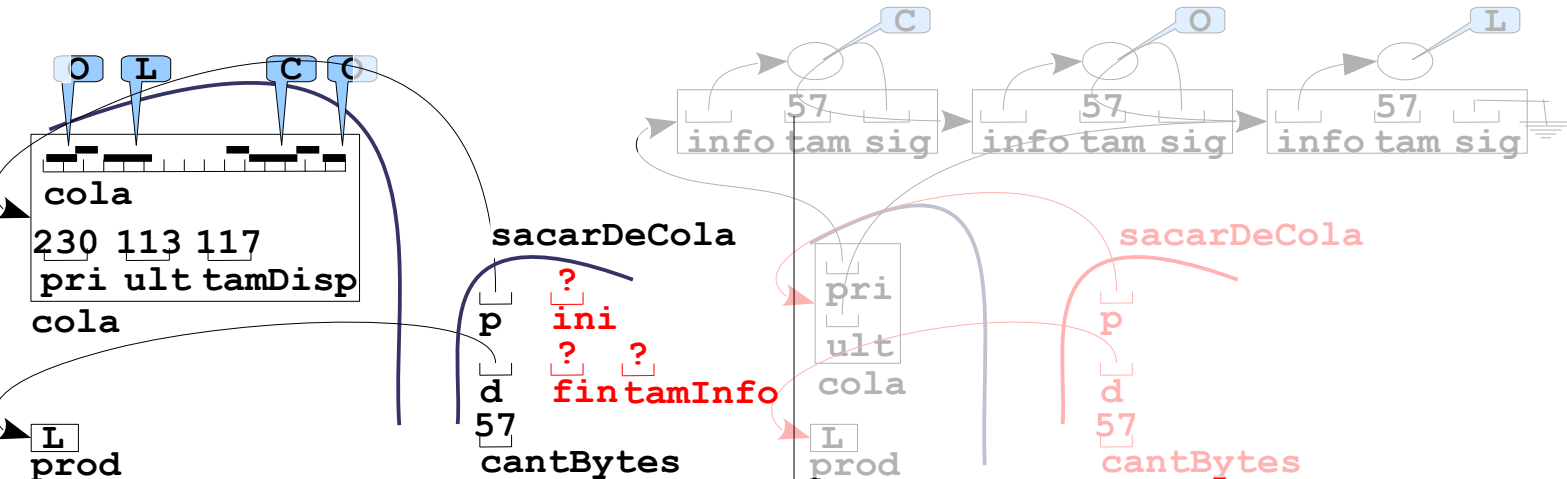
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



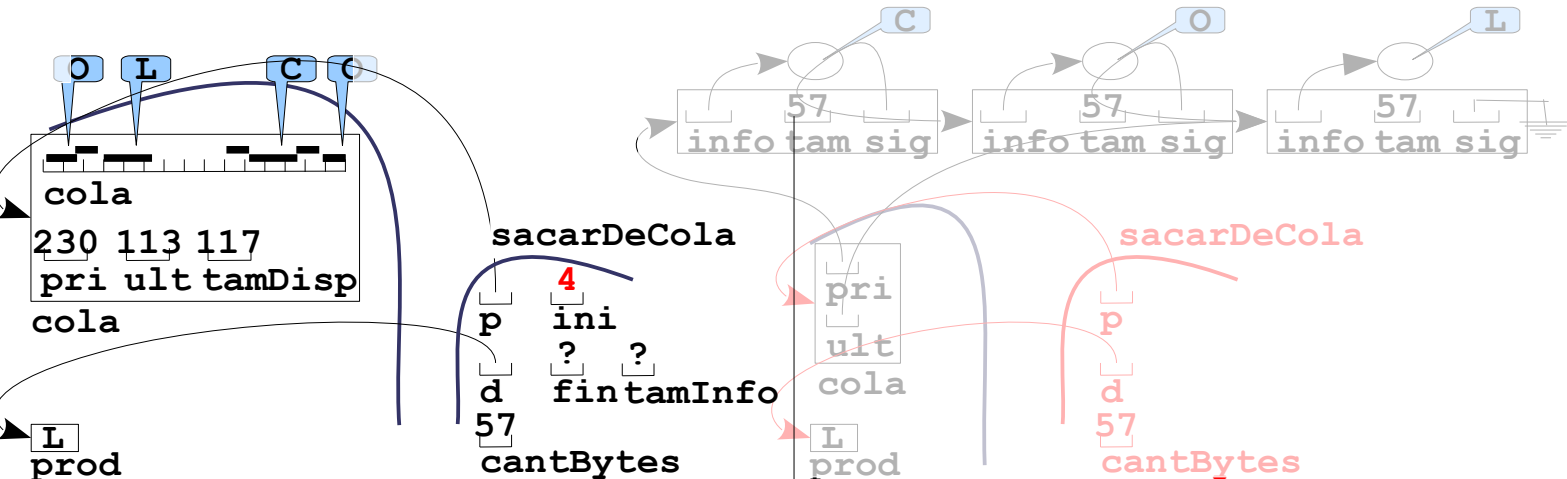
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



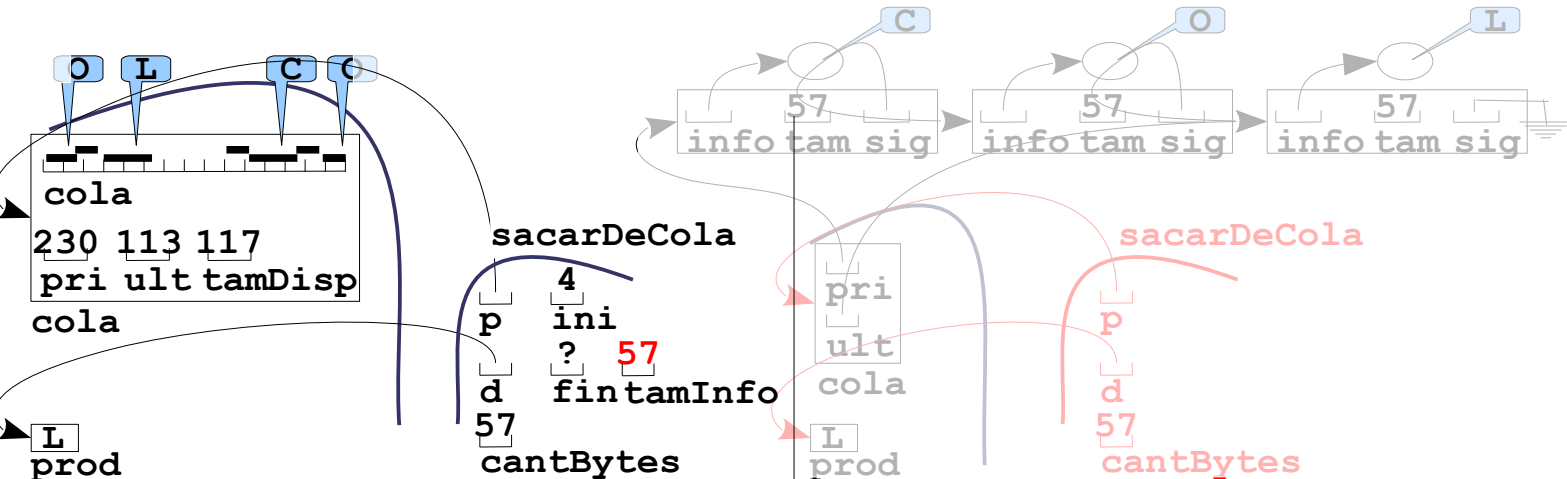
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



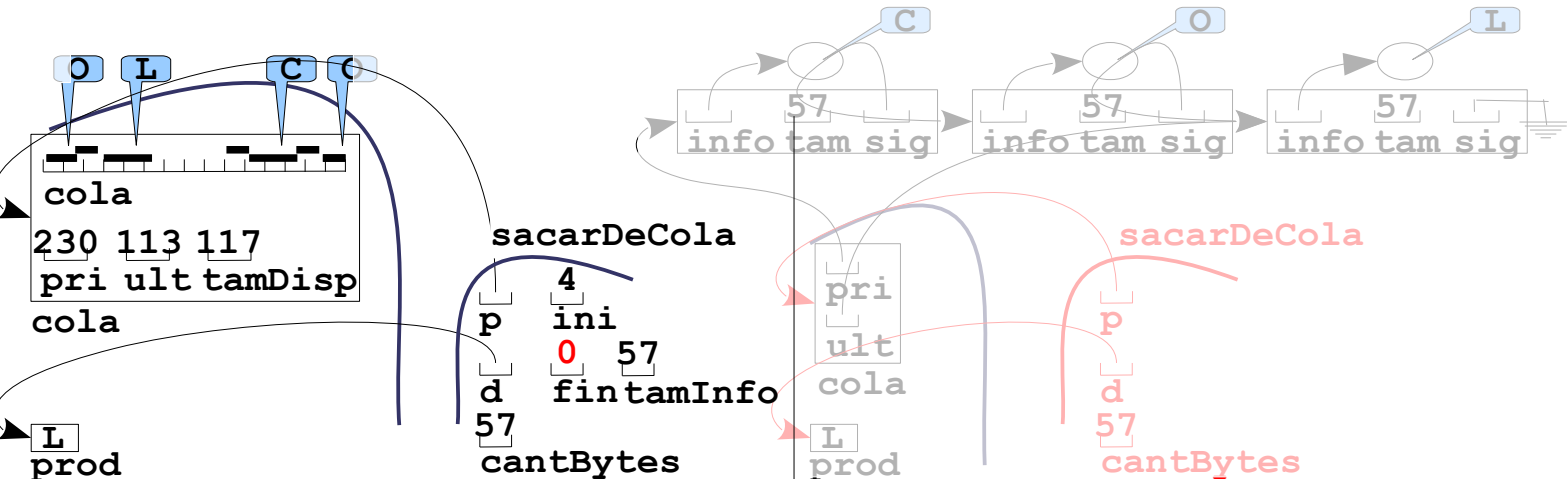
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



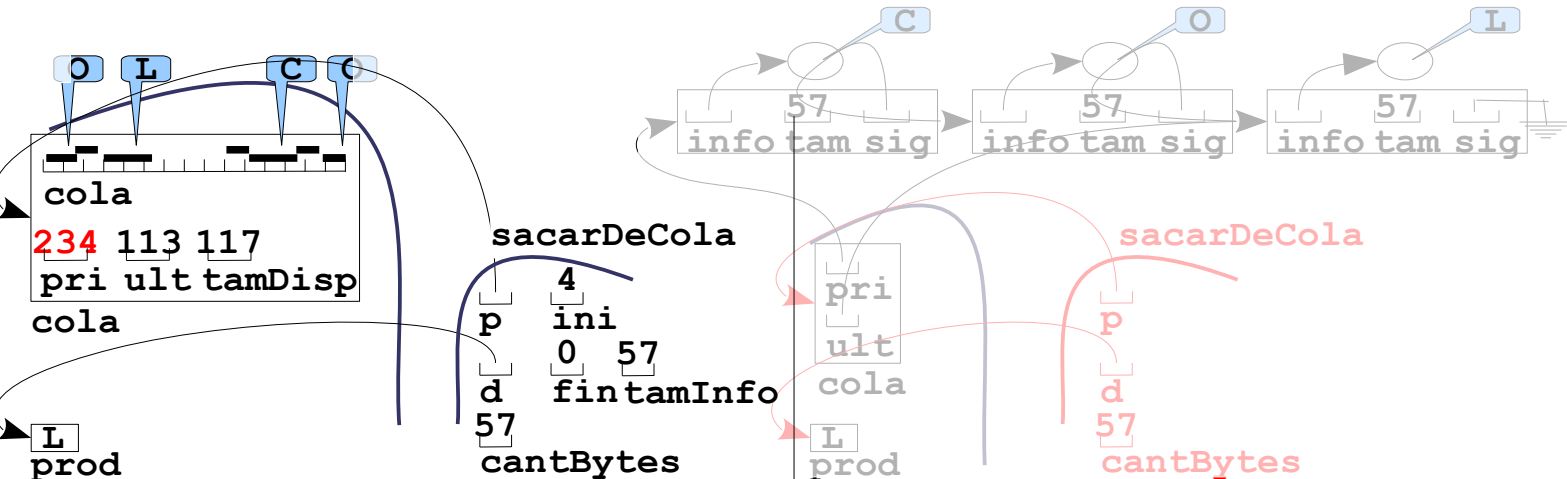
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



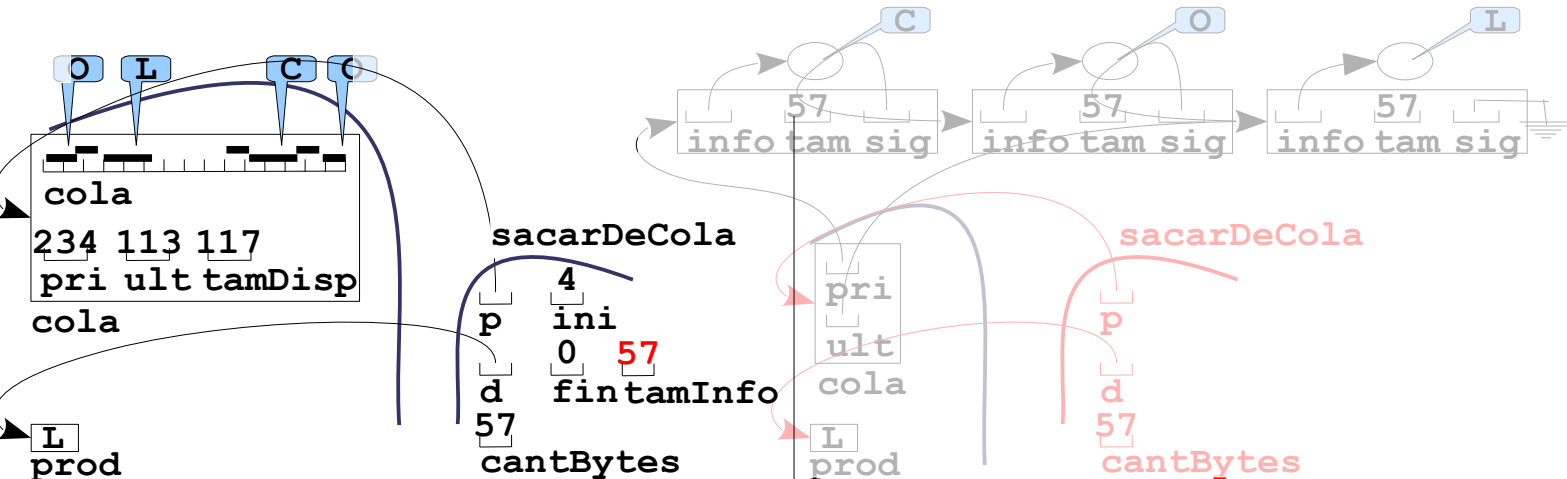
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



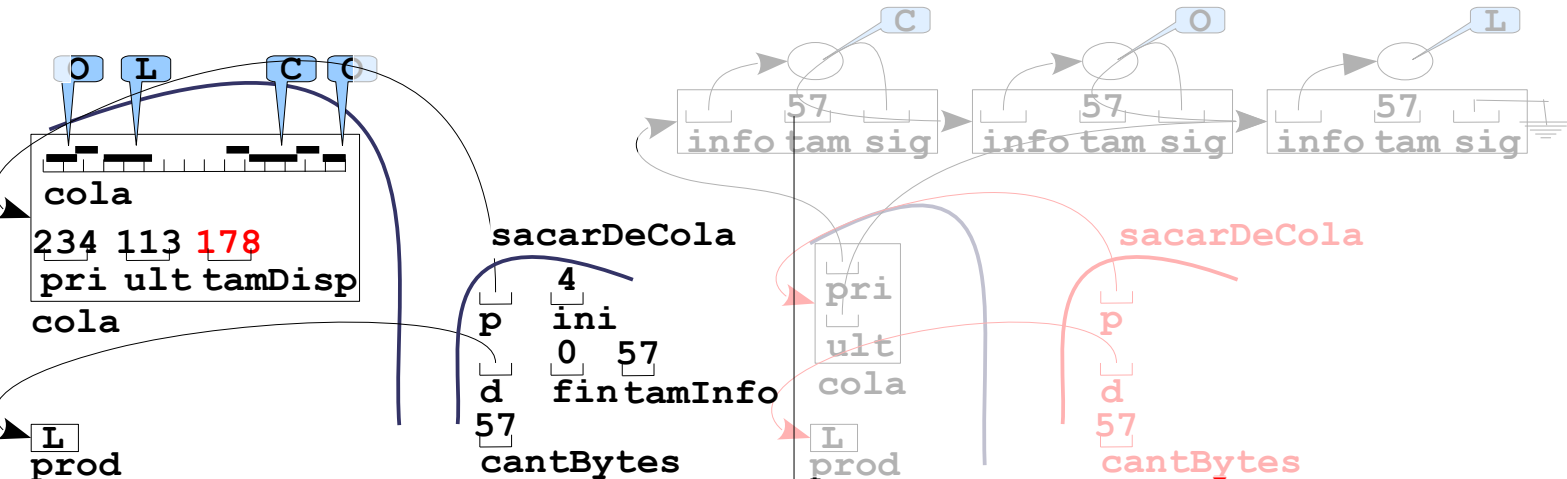
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



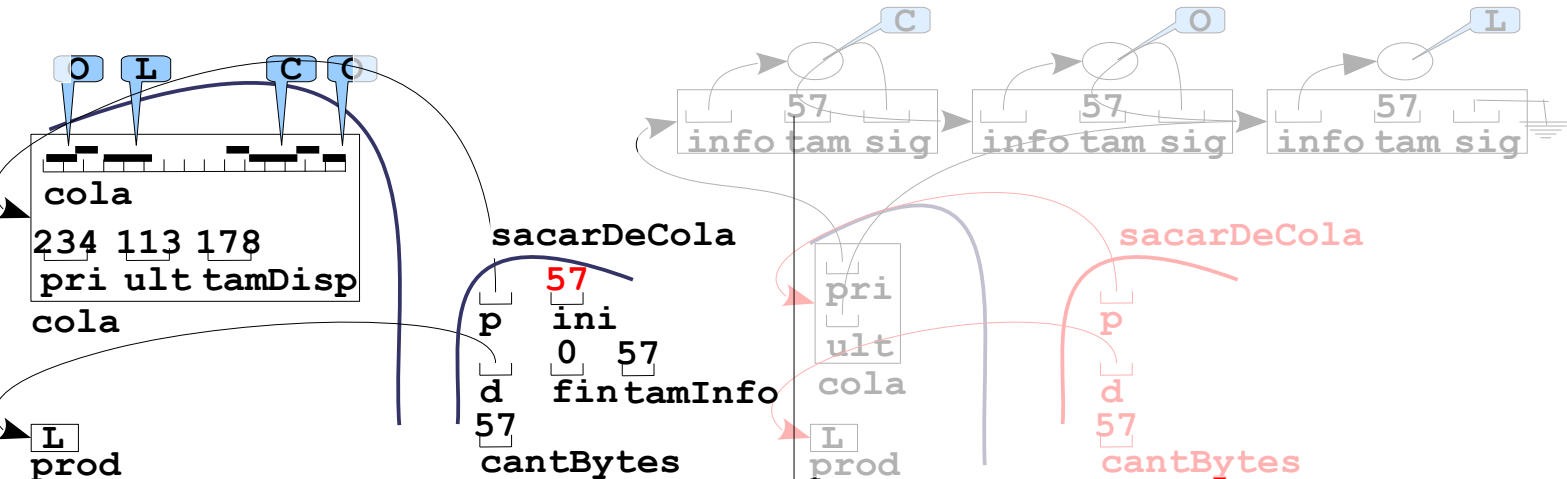
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



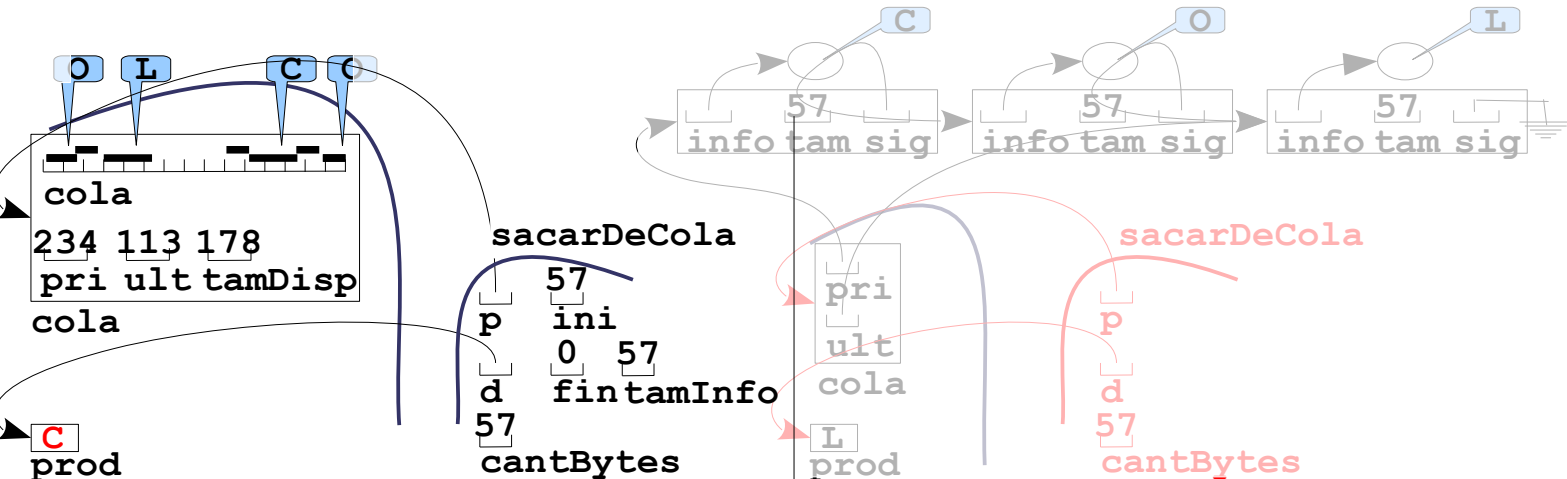
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



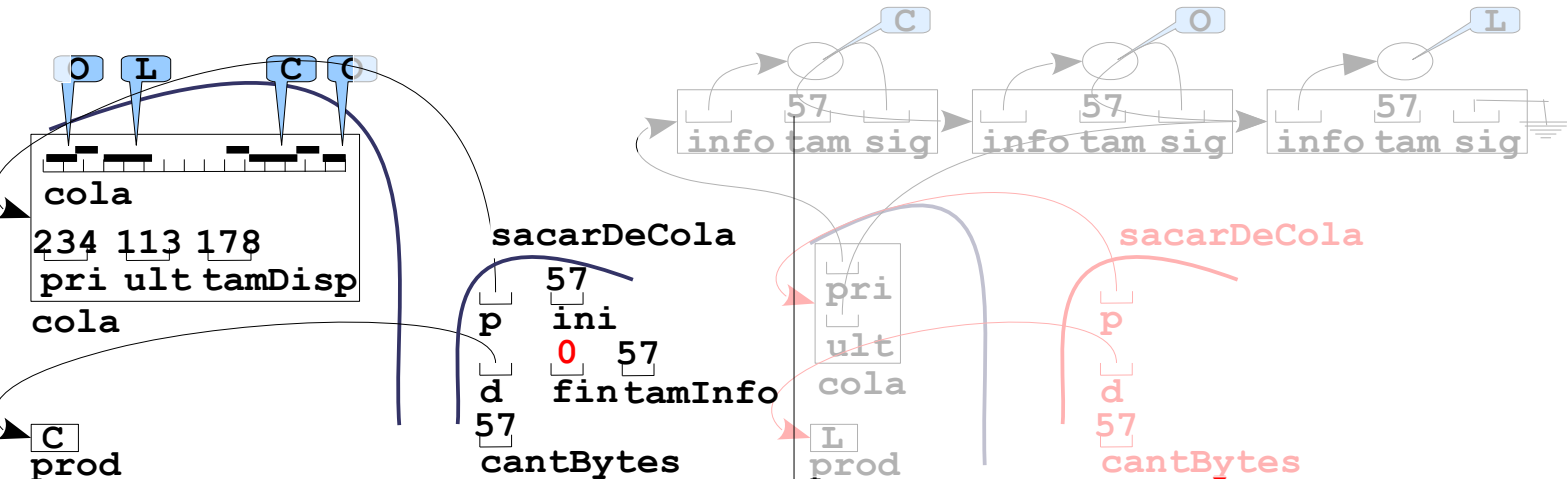
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



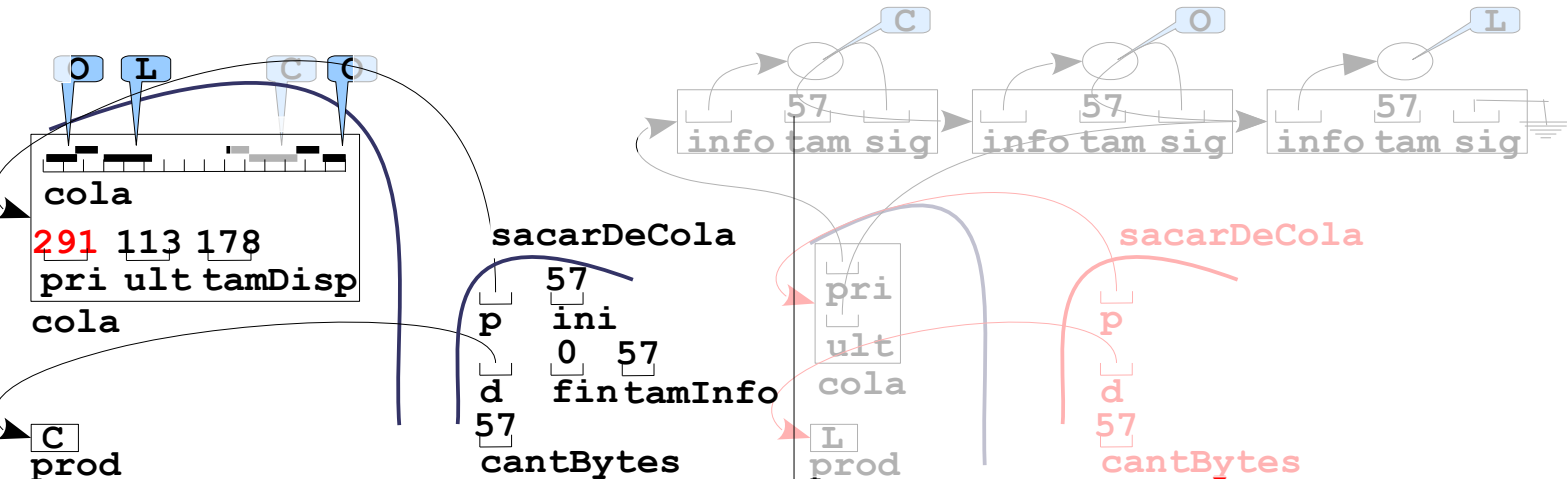
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

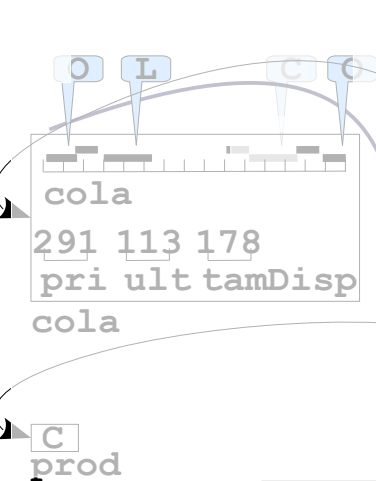


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81
82
```

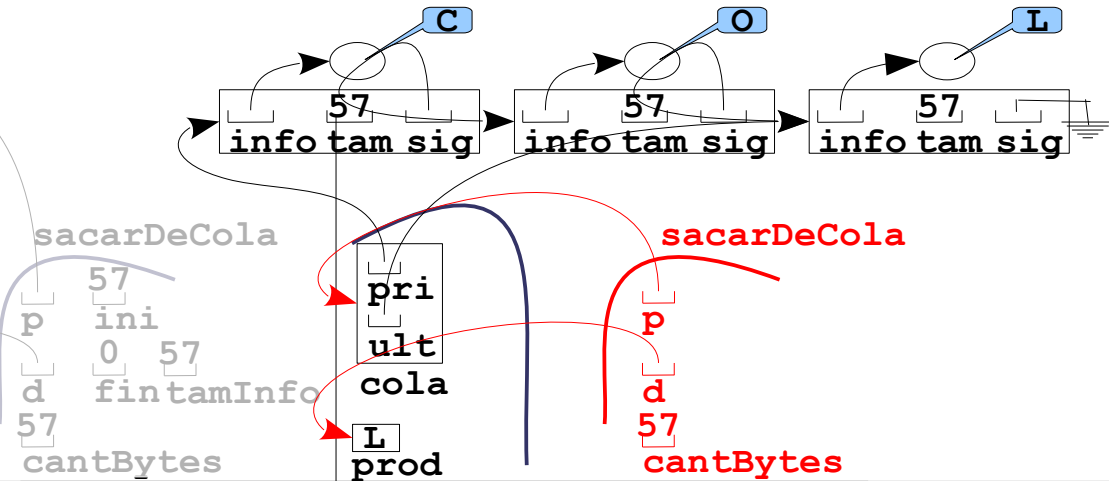
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

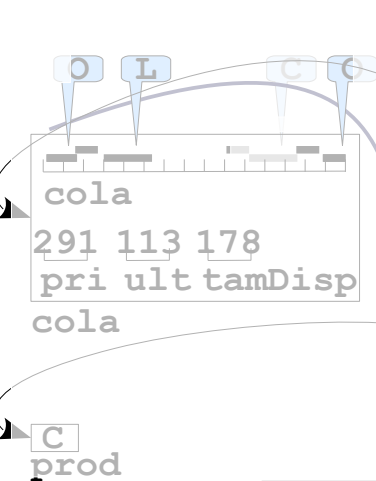


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

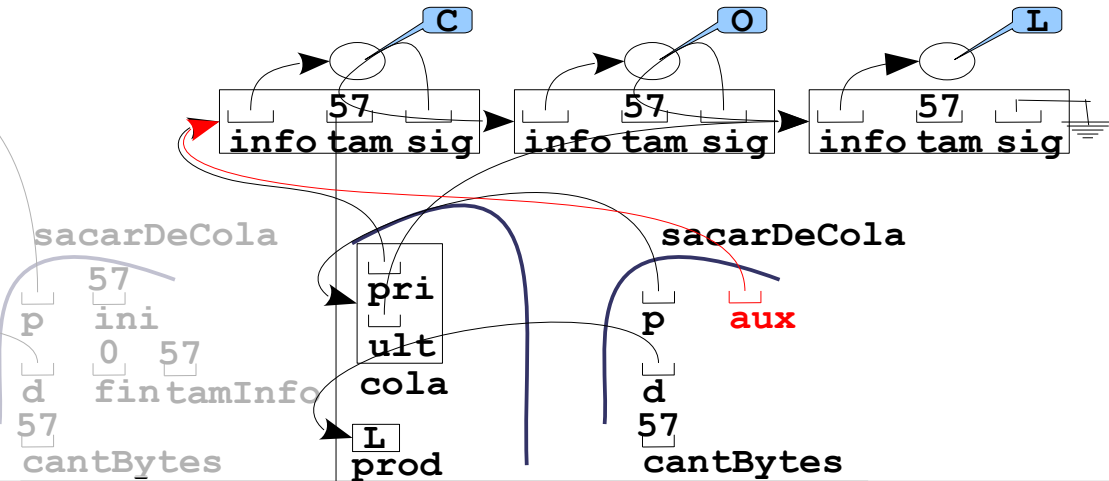
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

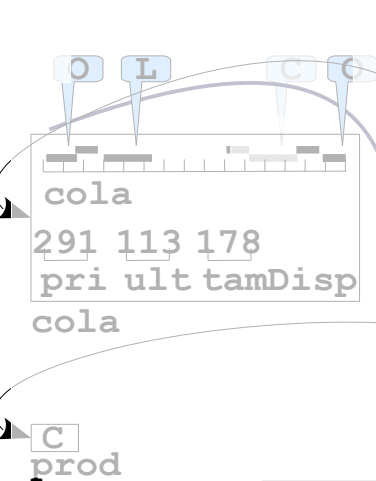


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

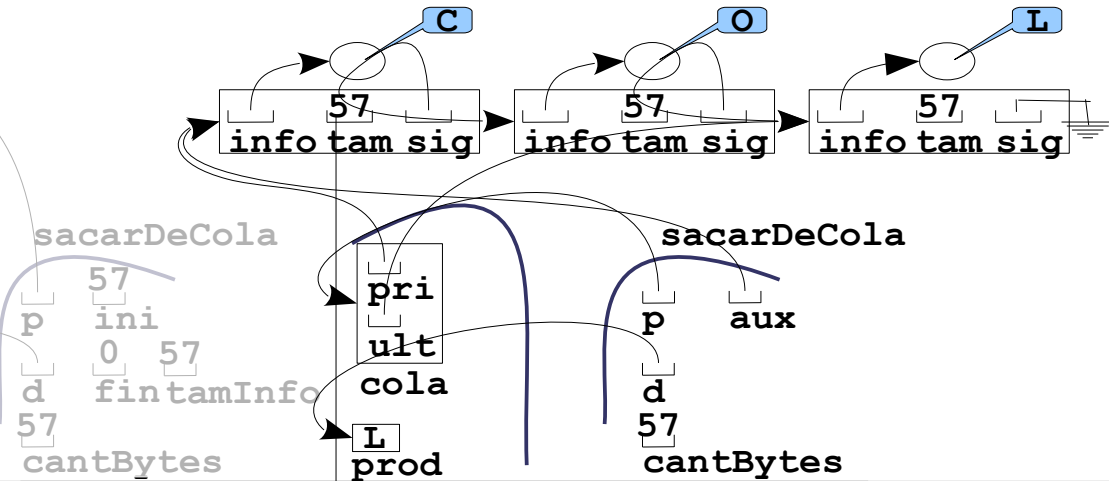
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

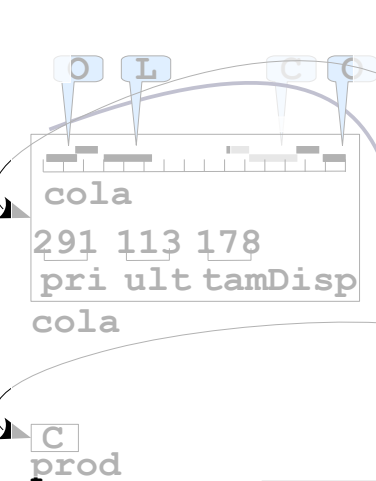


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

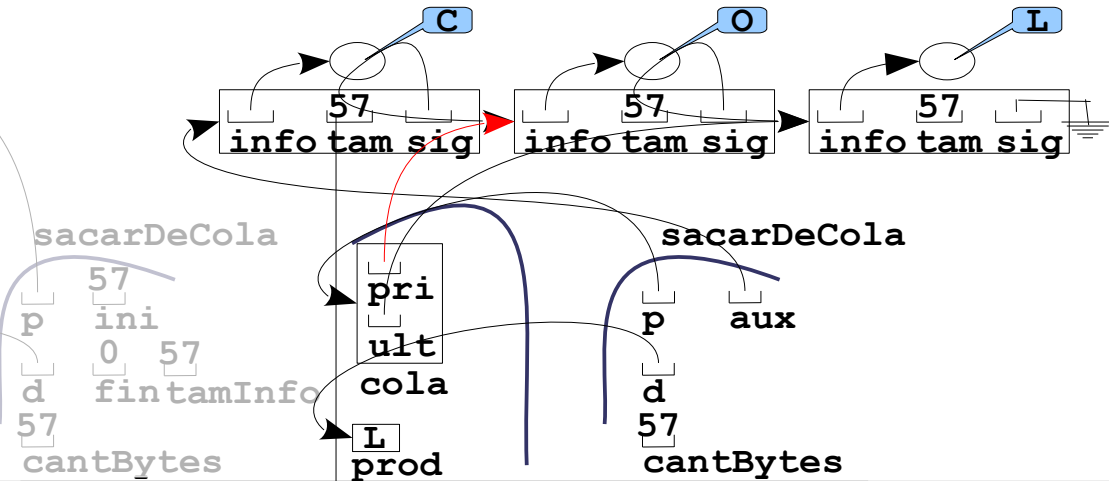
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

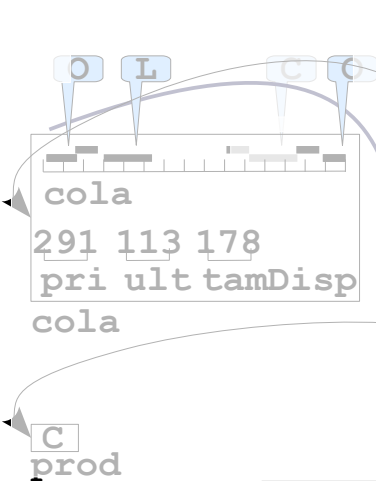


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

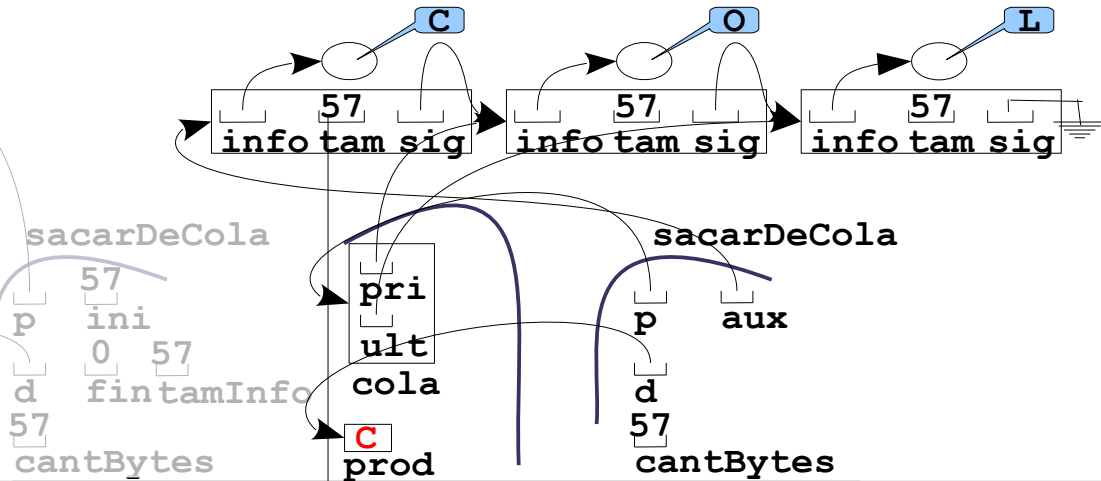
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

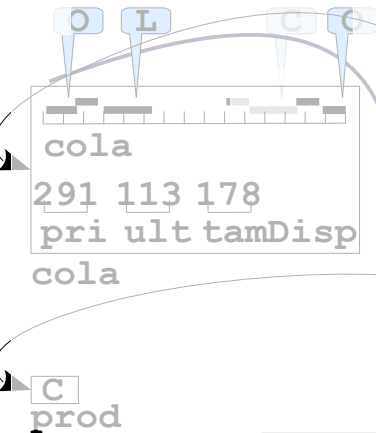


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

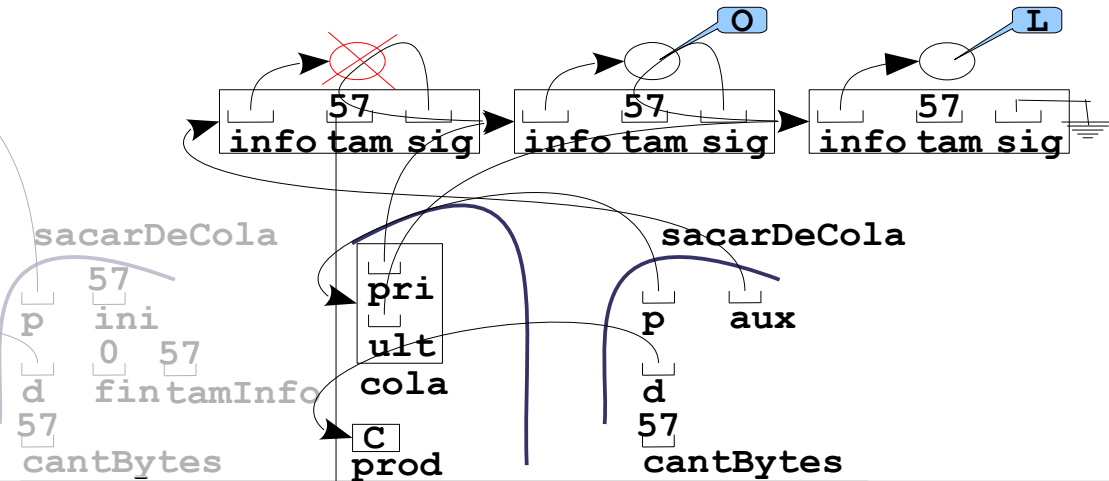
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

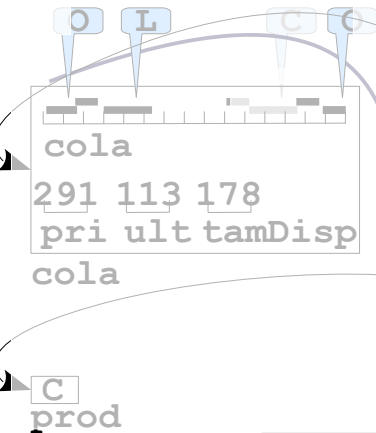


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

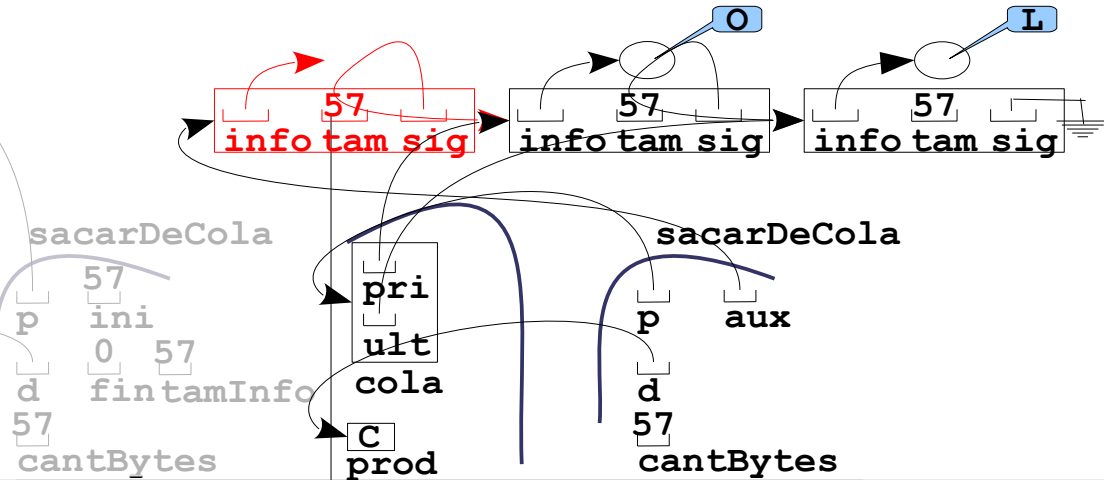

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

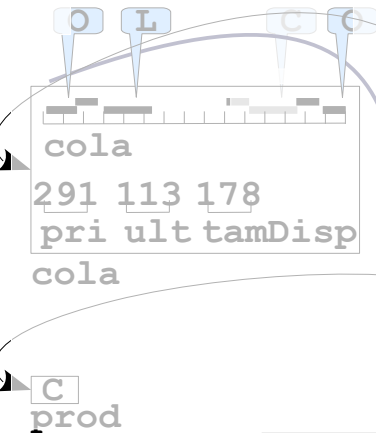


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

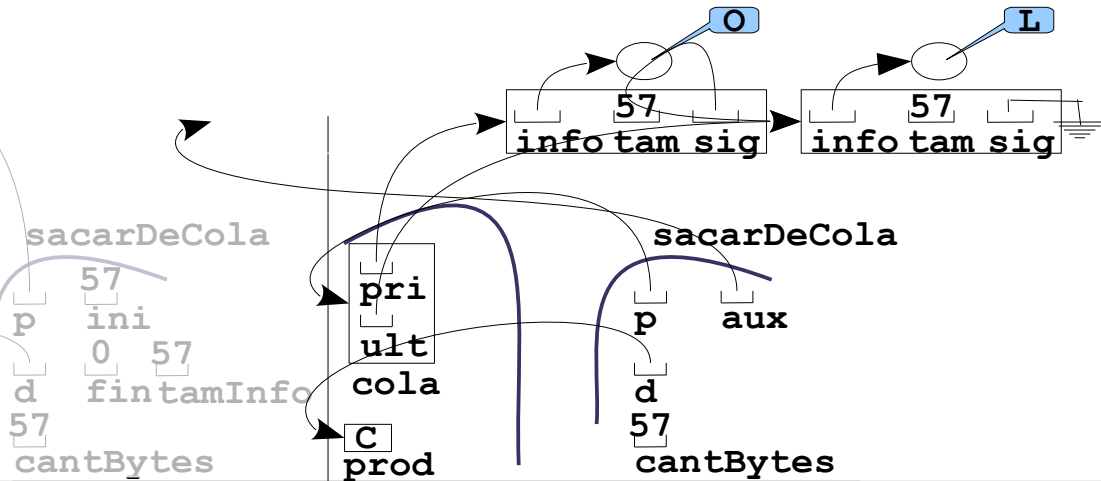
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



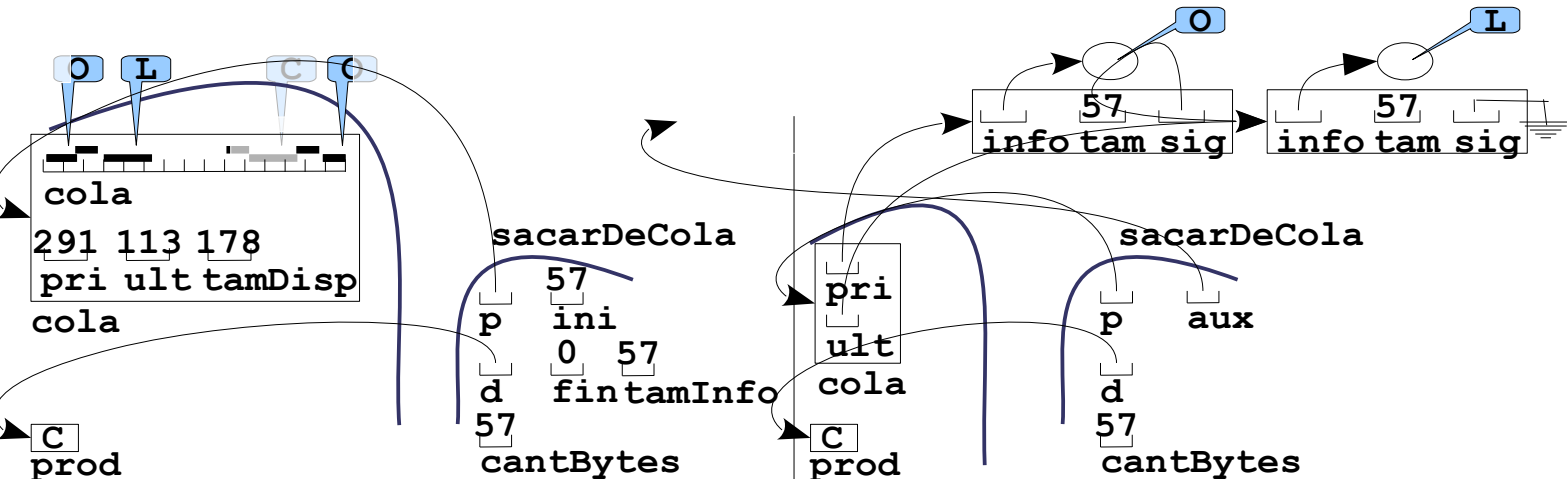
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



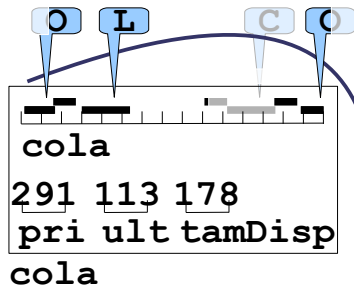
```
main.c x main.h x productos.c x product
70 int sacarDeCola(tCola *p)
71 {
72     82 p->pri = fin;
73     83 p->tamDisp = 0;
74     84 tamInfo = m;
75     85 if((ini = m) < 0)
76     86     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
77     87 if((fin = tamInfo) < 0)
78     88     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
79     89 p->pri = fin;
80     90 return 1;
81 }
82

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

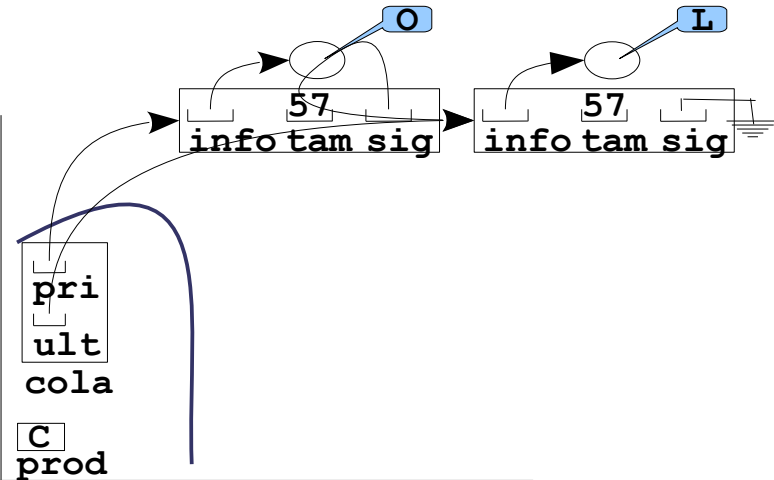
Tipo de dato Cola.

Implementación estática



C
prod

Implementación dinámica

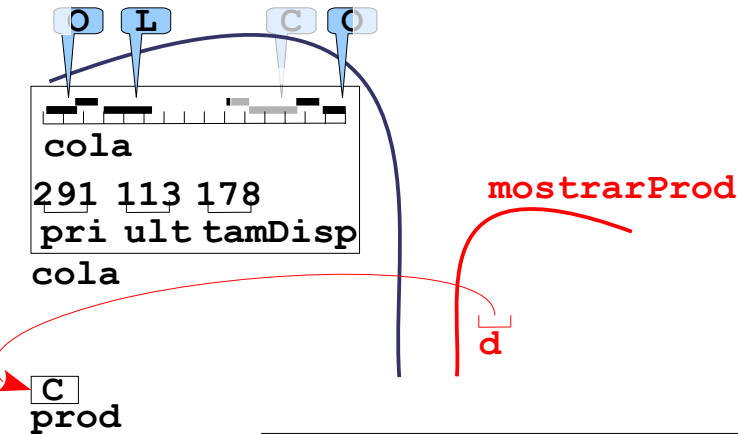


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100 if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101 puts("ERROR - cola vacia.");
102 else
103 mostrarProducto(&prod);
```

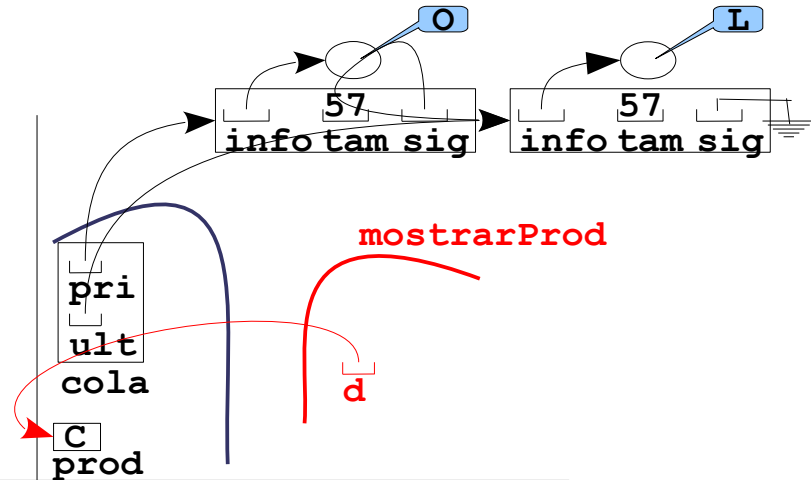
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

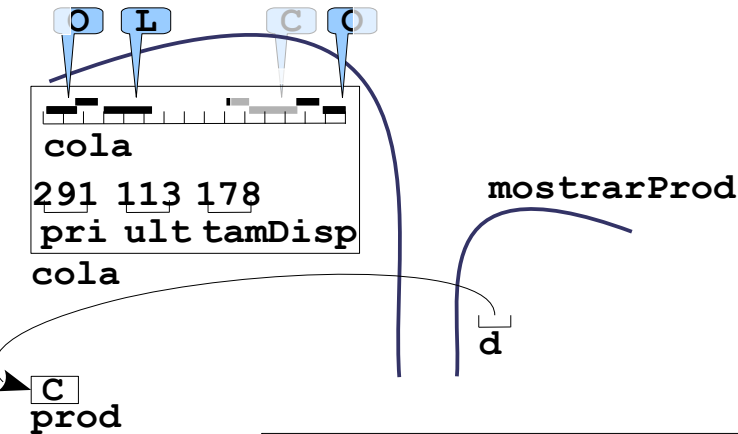


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

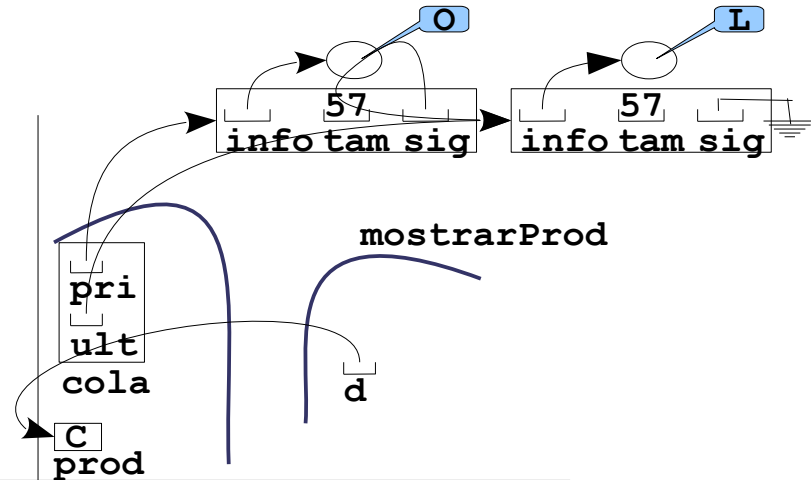
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

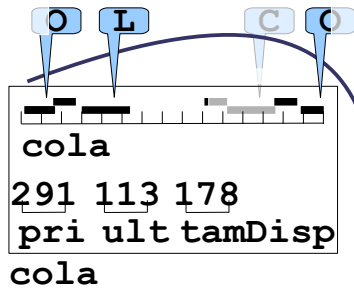


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

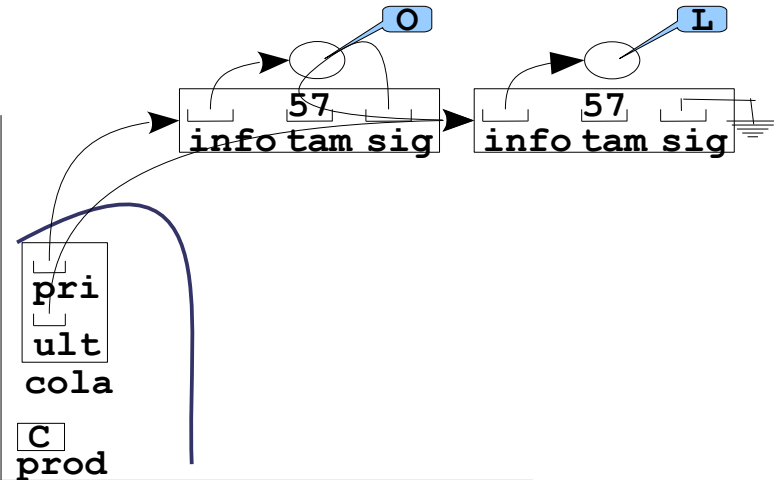
Tipo de dato Cola.

Implementación estática



C
prod

Implementación dinámica

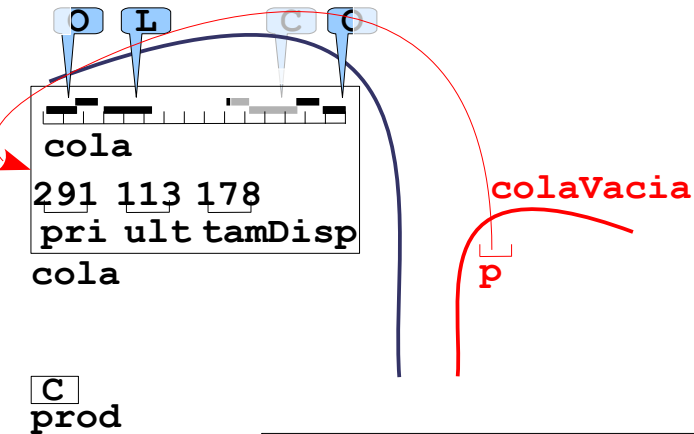


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

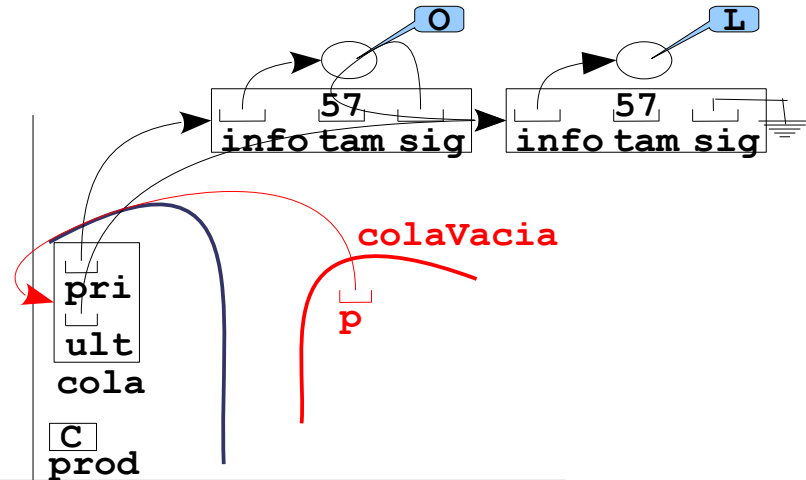

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica

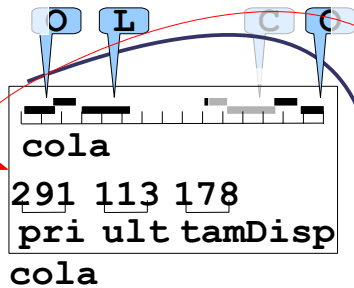


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

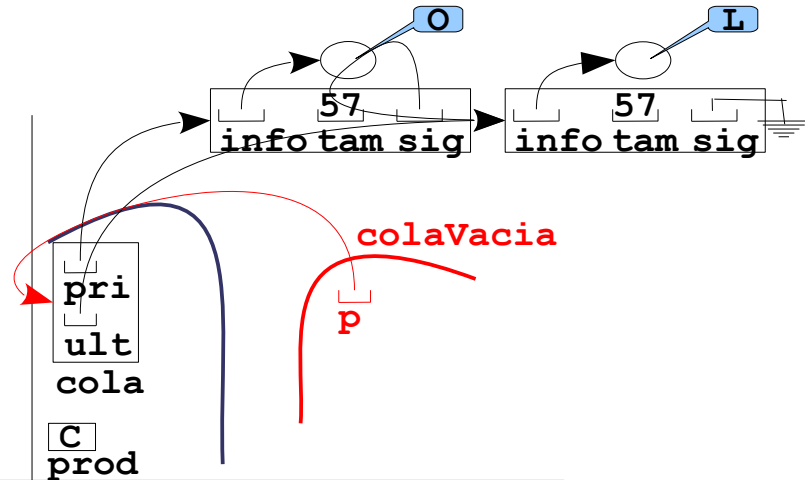
Tipo de dato Cola.

Implementación estática



C
prod

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102
```

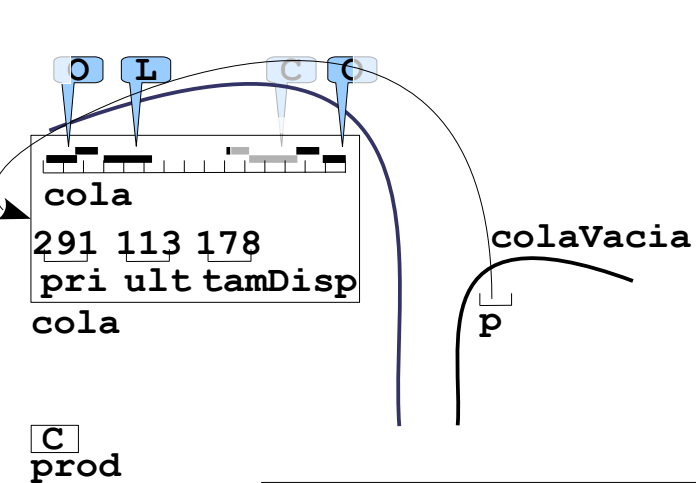
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
65 int colaVacia(const tCola *p)
66 {
67     return p->tamDisp == TAM_COLA;
68 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
49 int colaVacia(const tCola *p)
50 {
51     return p->pri == NULL;
52 }
```

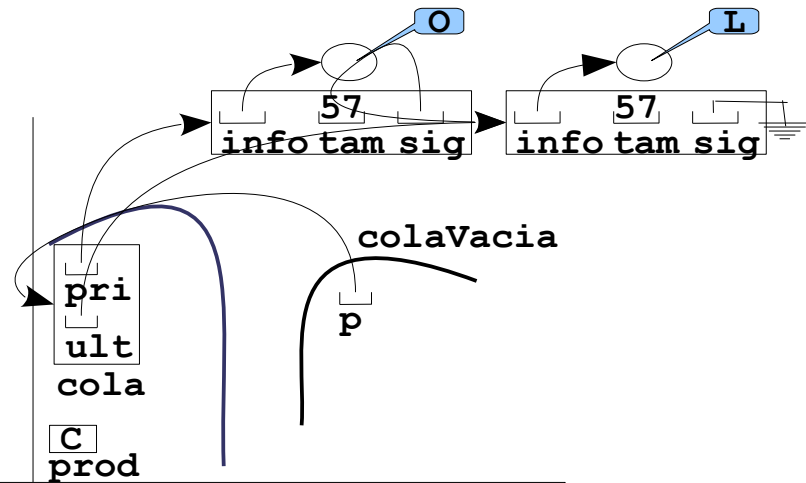
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102
```

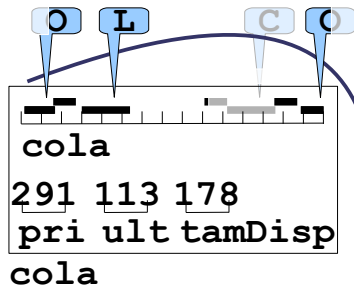
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
65 int colaVacia(const tCola *p)
66 {
67     return p->tamDisp == TAM_COLA;
68 }
```

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
49 int colaVacia(const tCola *p)
50 {
51     return p->pri == NULL;
52 }
```

Tipos de Datos Abstractos

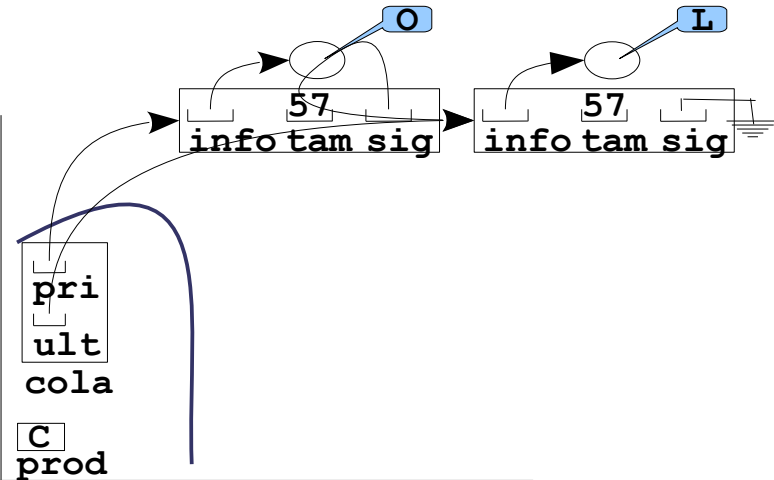
Tipo de dato Cola.

Implementación estática



C
prod

Implementación dinámica

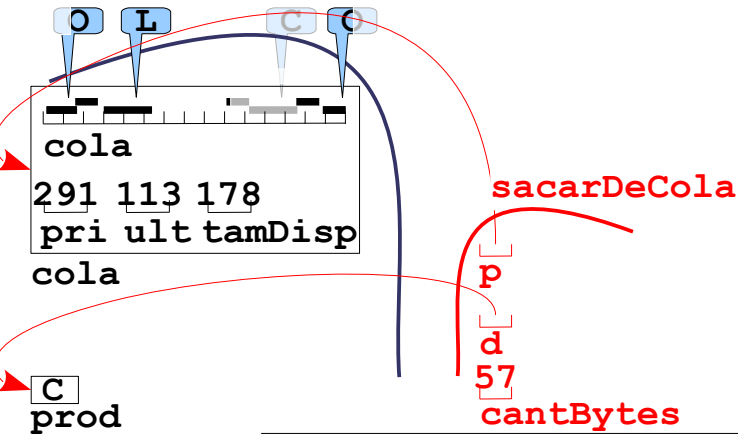


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

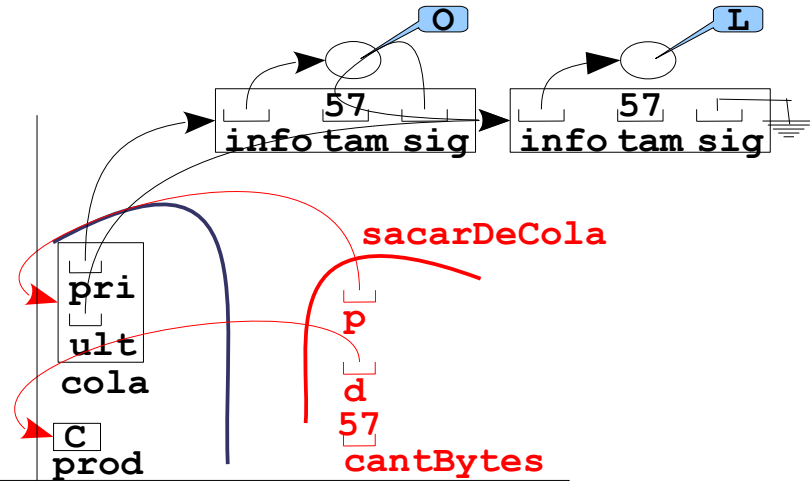
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



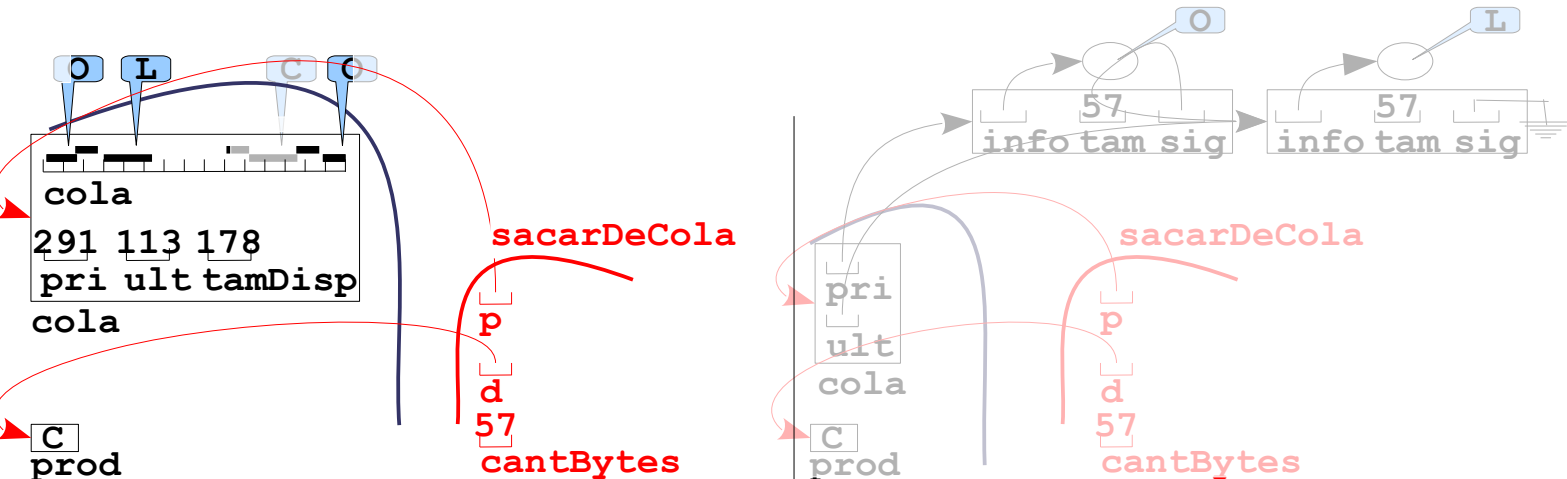
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



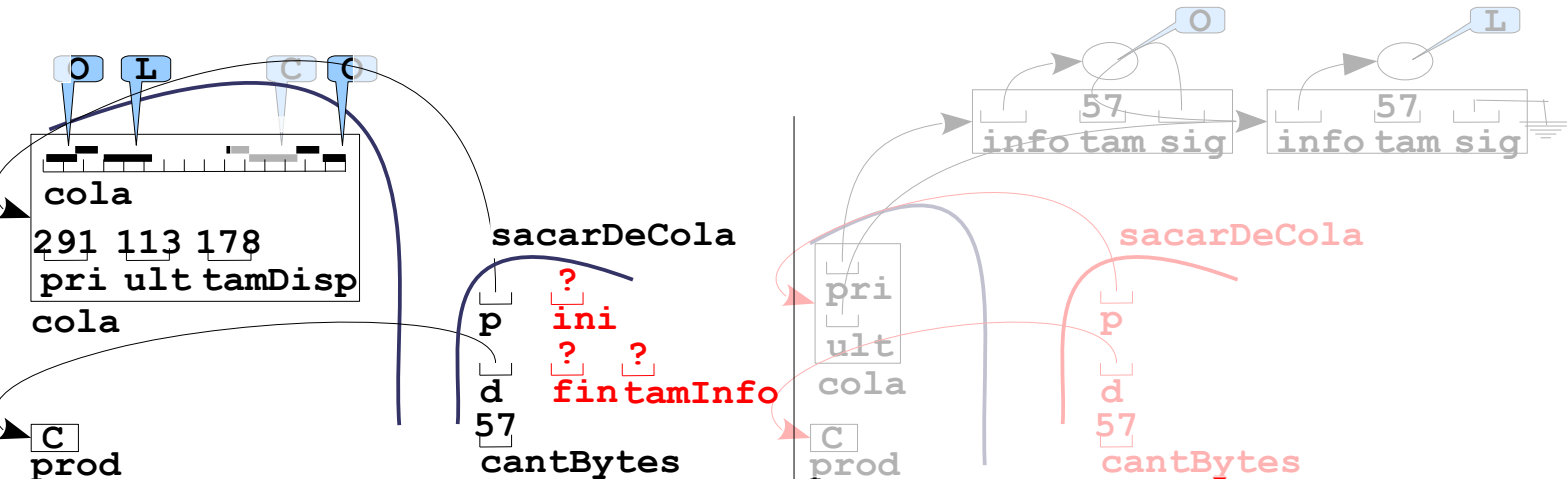
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



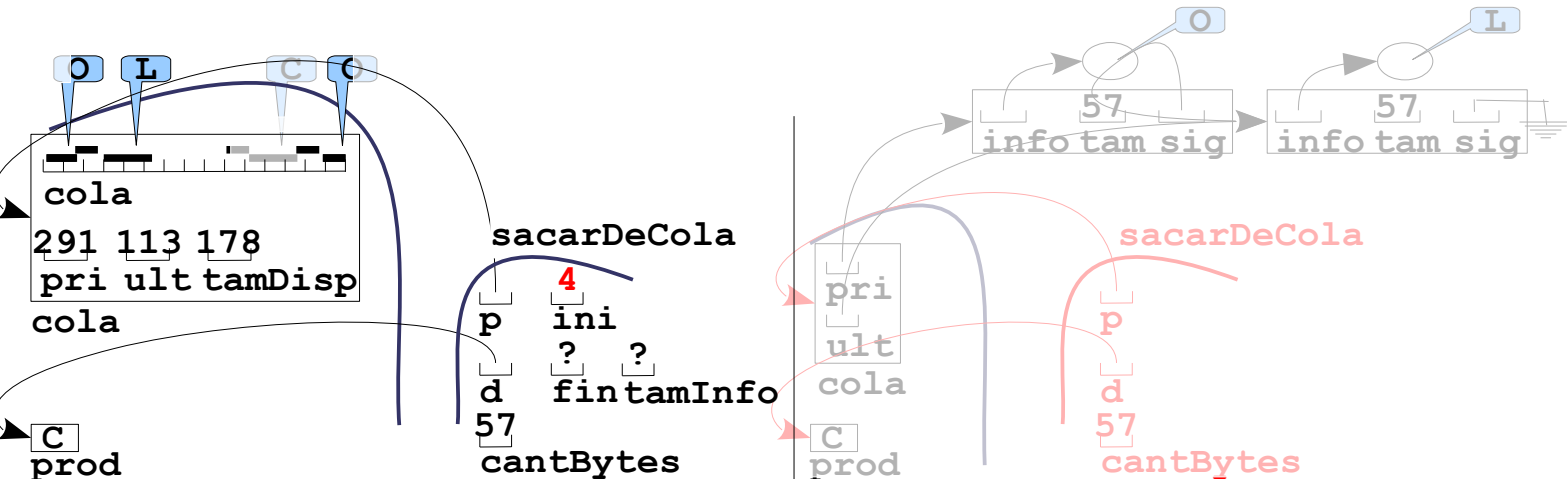
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



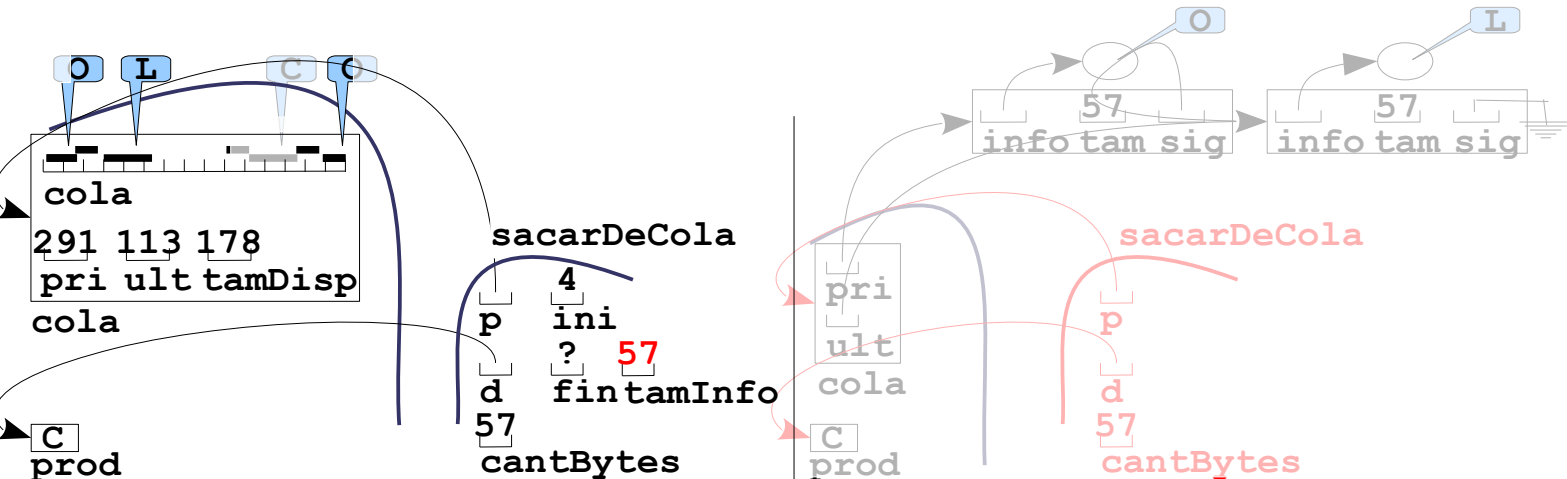
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



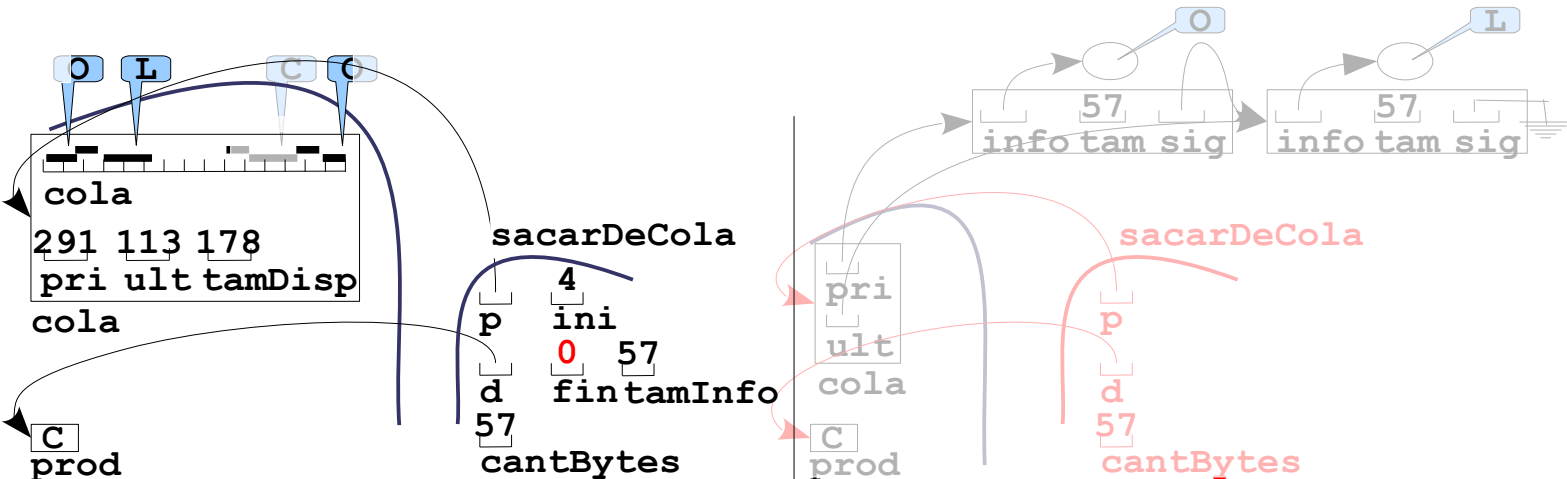
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



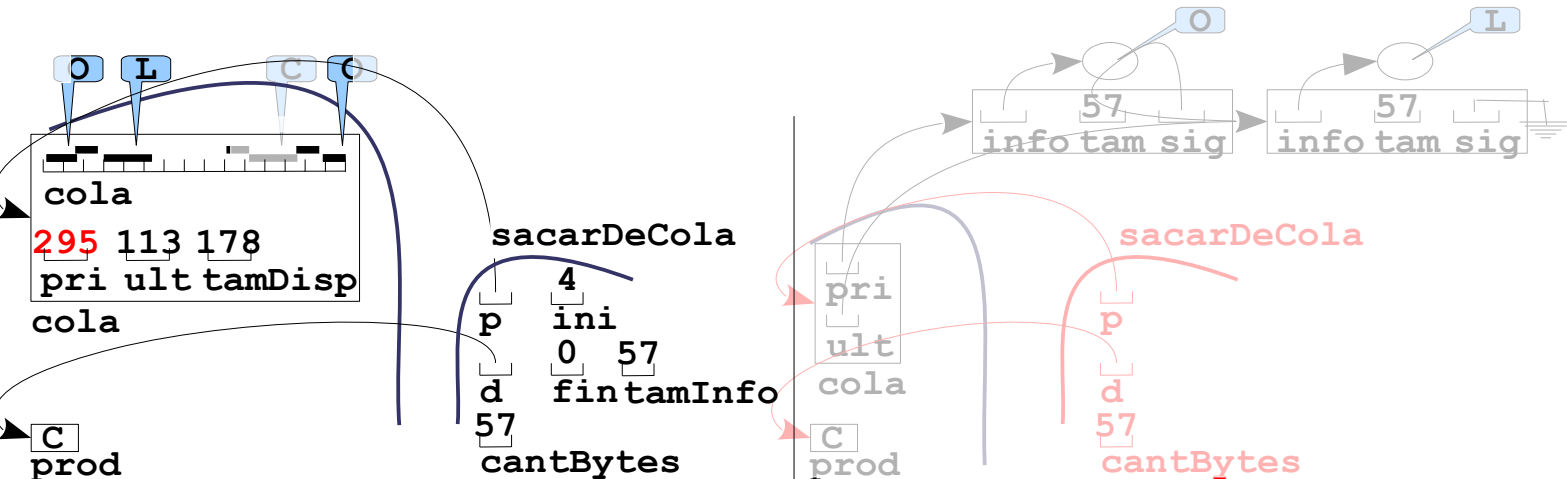
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     unsigned tamInfo,
73     ini,
74     fin;
75
76     if(p->tamDisp == TAM_COLA)
77         return 0;
78     if((ini = minimo(sizeof(unsigned), TAM_COLA - p->pri)) != 0)
79         memcpy(&tamInfo, p->cola + p->pri, ini);
80     if((fin = sizeof(unsigned) - ini) != 0)
81         memcpy(((char *)&tamInfo) + ini, p->cola, fin);
82     p->pri = fin ? fin : p->pri + ini;
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



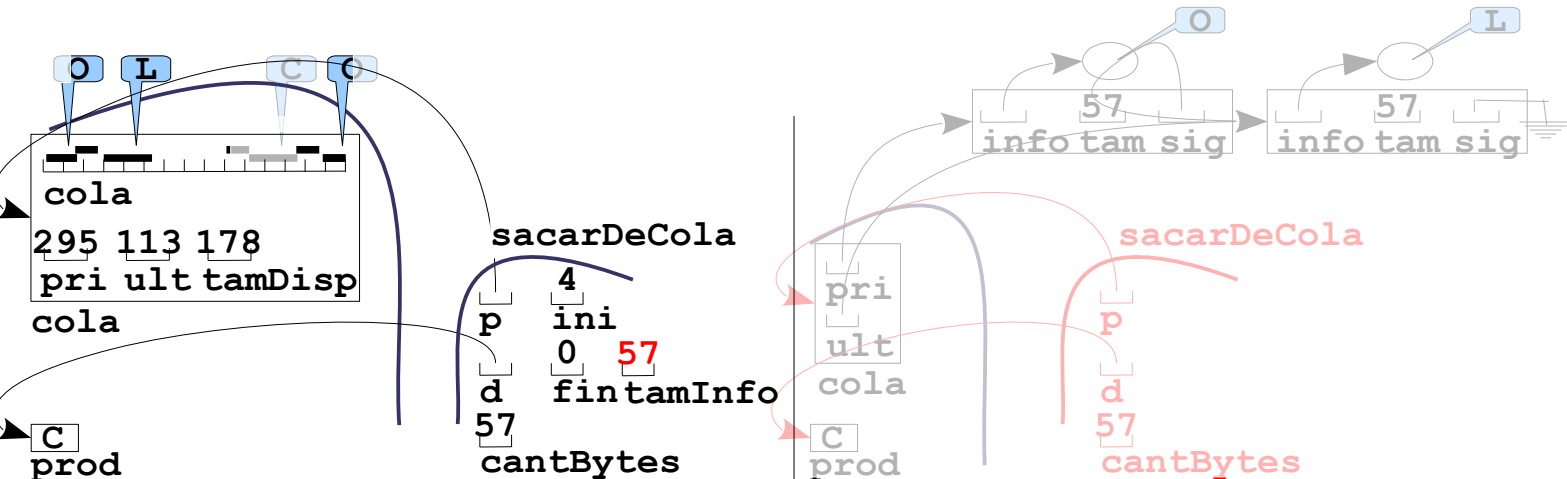
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



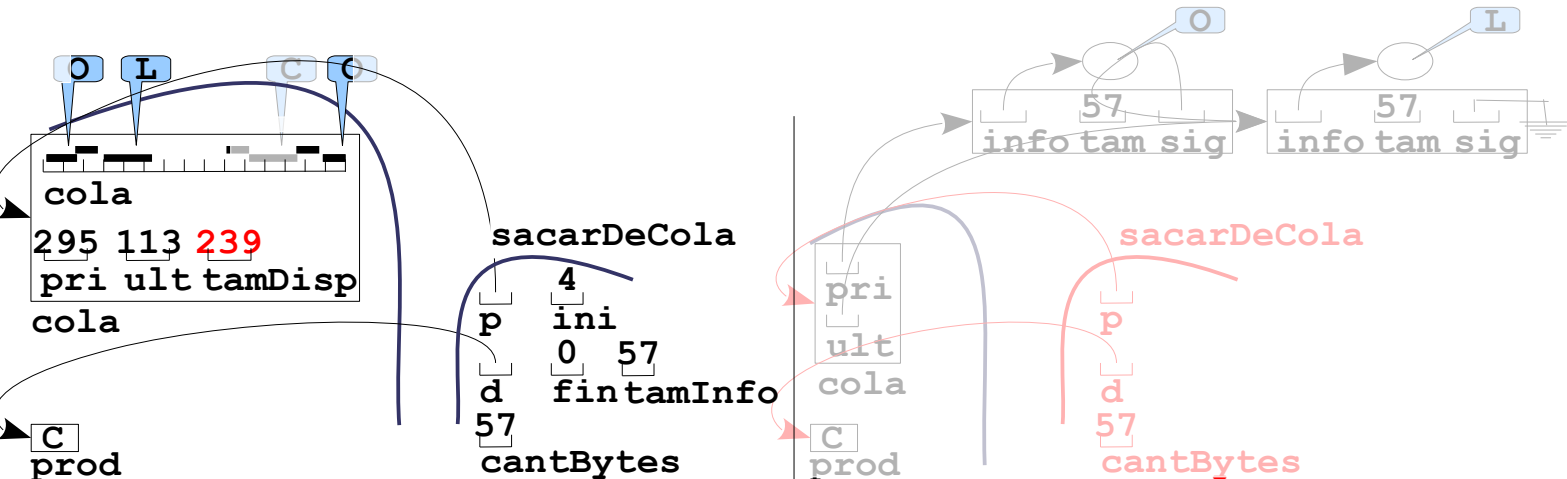
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     p->pri = fin ? fin : p->pri + ini;
73     p->tamDisp += sizeof(unsigned) + tamInfo;
74     tamInfo = minimo(tamInfo, cantBytes);
75     if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
76     {
77         memcpy(d, p->cola + p->pri, ini);
78     }
79     if((fin = tamInfo - ini) != 0)
80     {
81         memcpy(((char *)d) + ini, p->cola, fin);
82     }
83     p->pri = fin ? fin : p->pri + ini;
84     return 1;
85 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



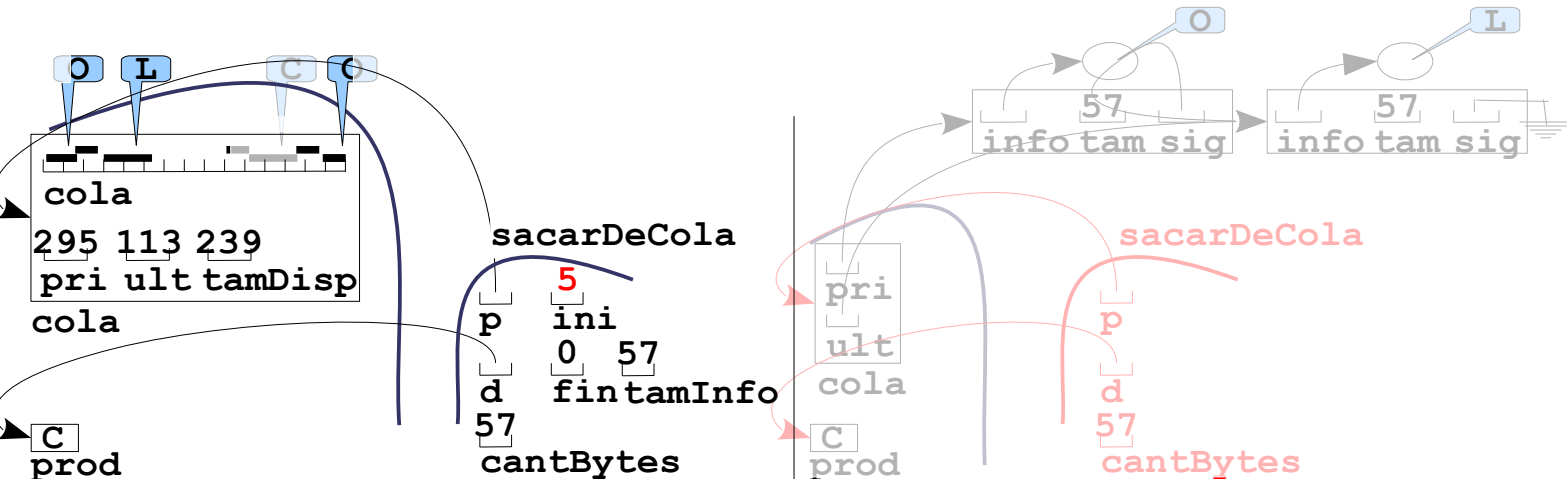
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



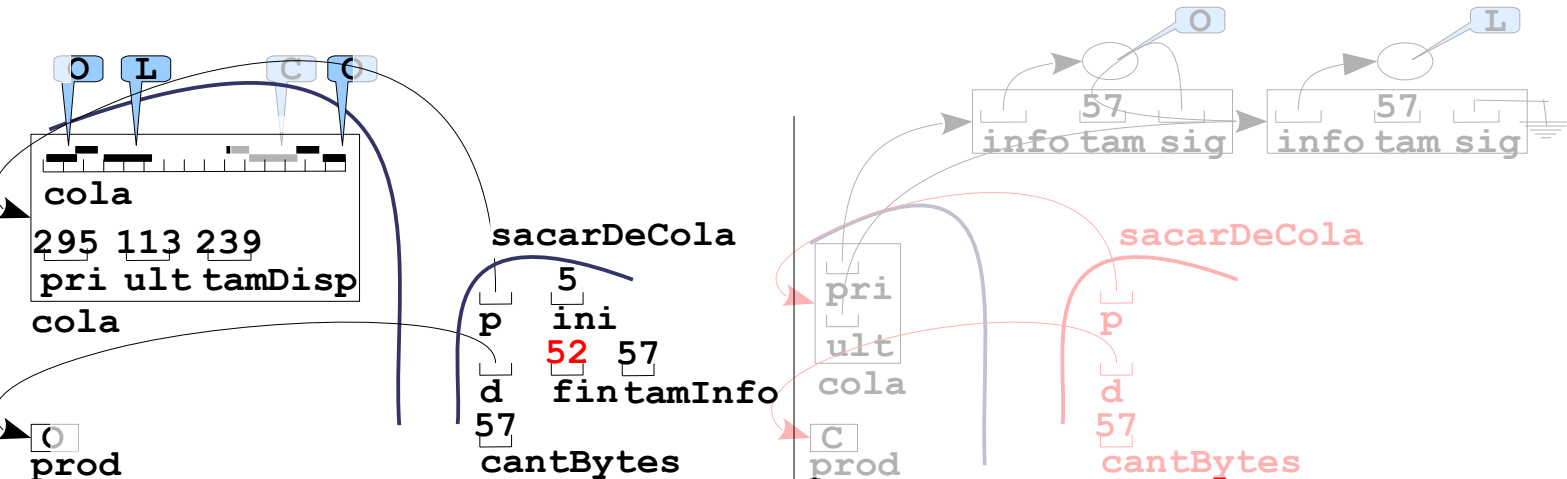
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



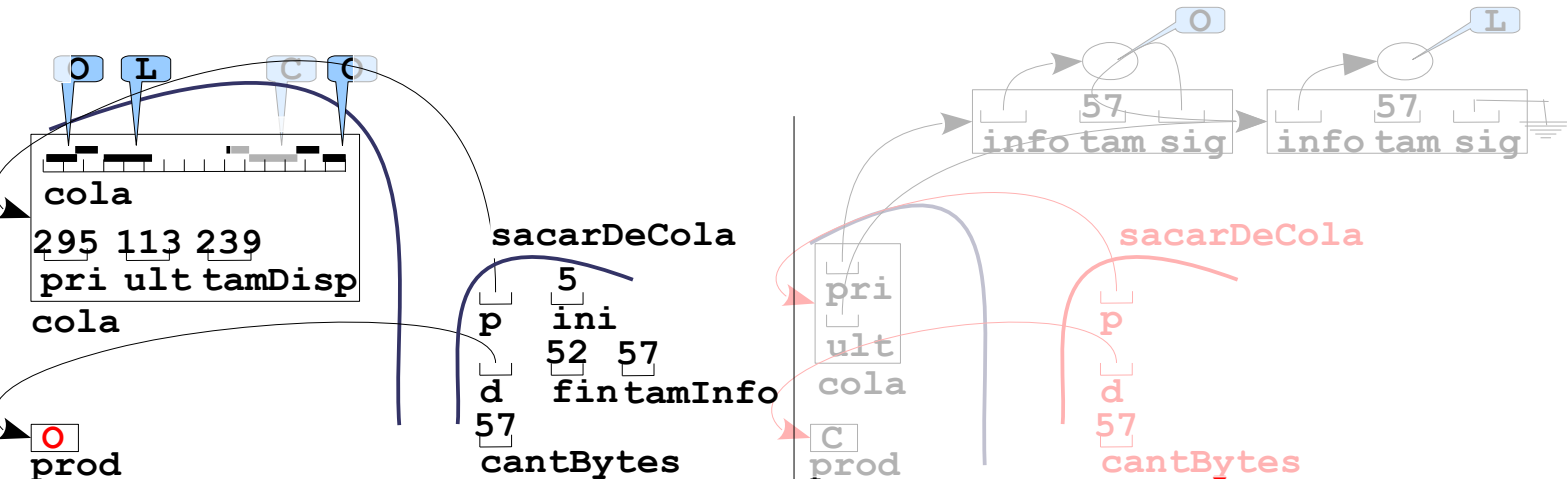
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



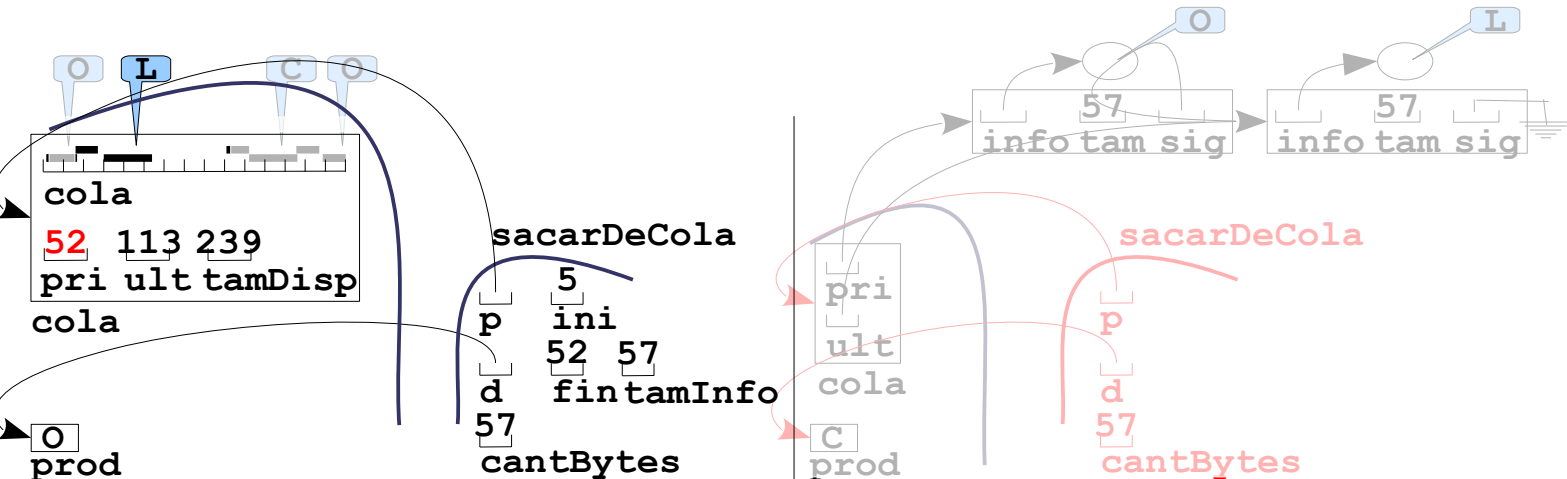
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81 92
82
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 82 p->pri = fin ? fin : p->pri + ini;
72 83 p->tamDisp += sizeof(unsigned) + tamInfo;
73 84 tamInfo = minimo(tamInfo, cantBytes);
74 85 if((ini = minimo(tamInfo, TAM_COLA - p->pri)) != 0)
75 86     memcpy(d, p->cola + p->pri, ini);
76 87 if((fin = tamInfo - ini) != 0)
77 88     memcpy(((char *)d) + ini, p->cola, fin);
78 89 p->pri = fin ? fin : p->pri + ini;
79 90 return 1;
80 91 }
81
82
```

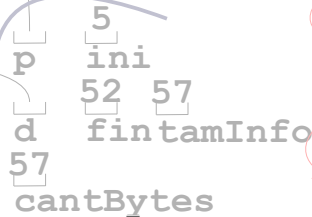
Tipos de Datos Abstractos

Tipo de dato Cola.

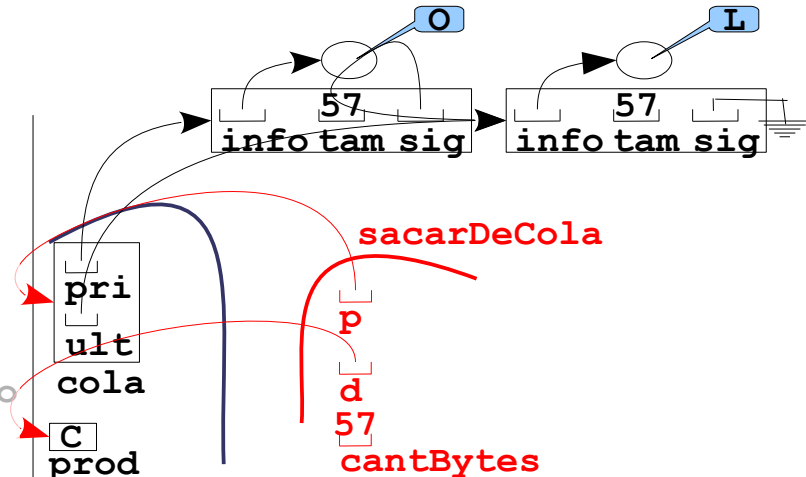
Implementación estática



sacarDeCola



Implementación dinámica

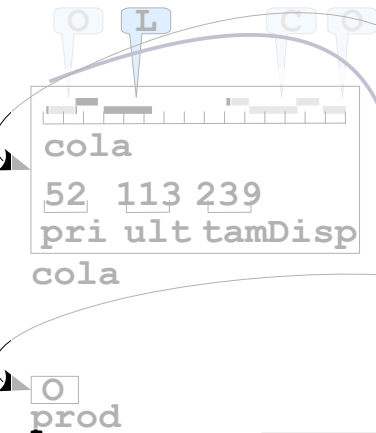


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

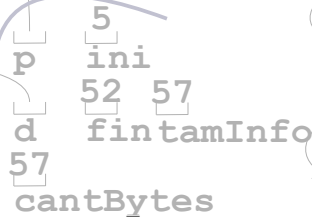
Tipos de Datos Abstractos

Tipo de dato Cola.

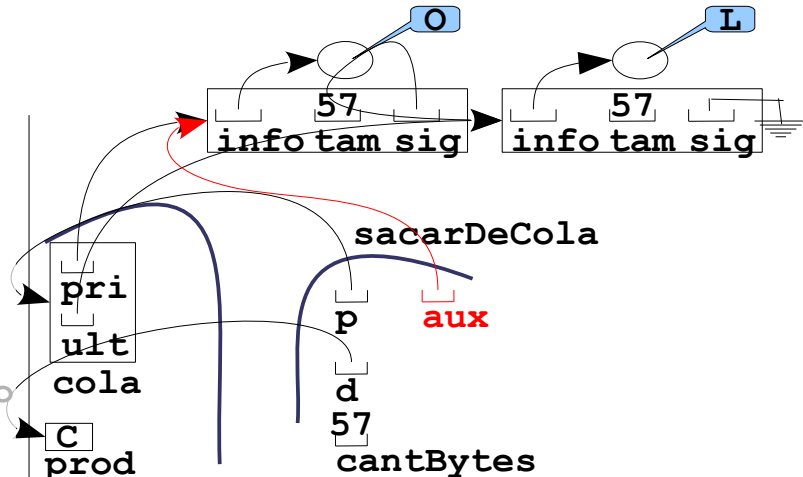
Implementación estática



sacarDeCola



Implementación dinámica



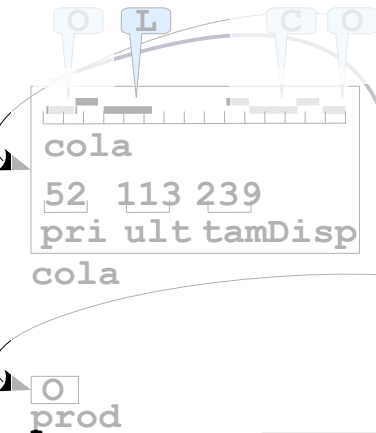
sacarDeCola

```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

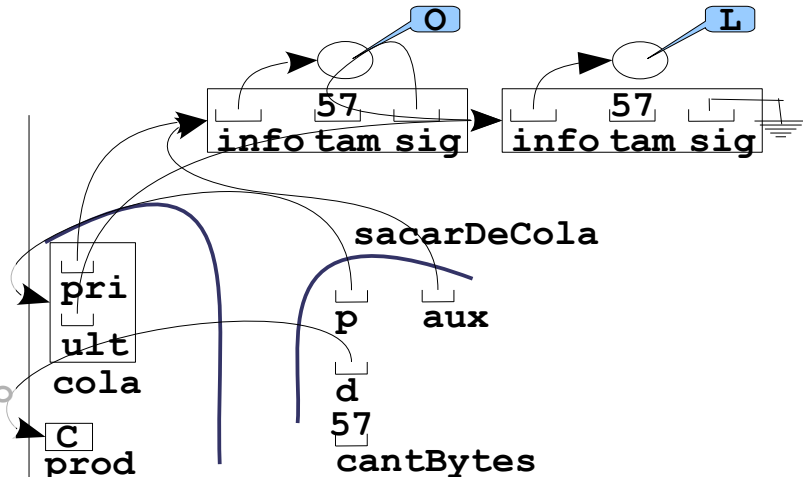
Implementación estática



sacarDeCola



Implementación dinámica

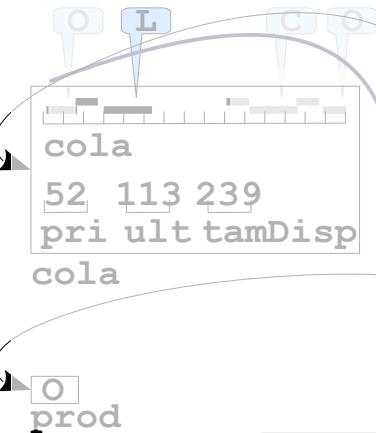


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

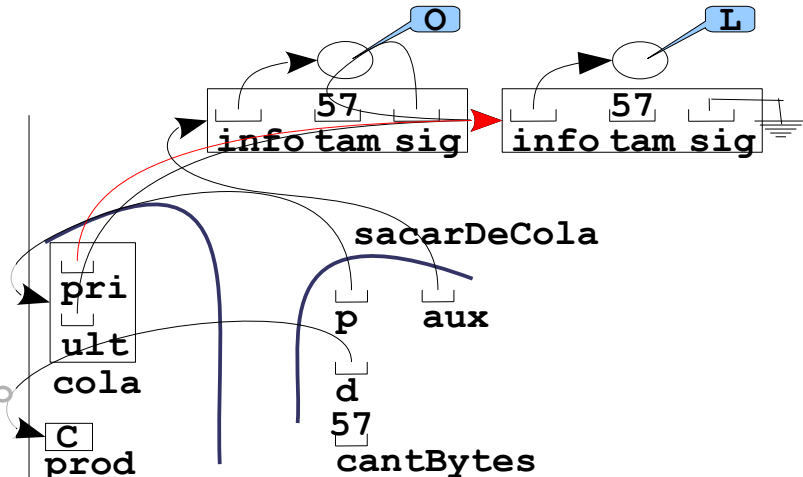
Implementación estática



sacarDeCola



Implementación dinámica

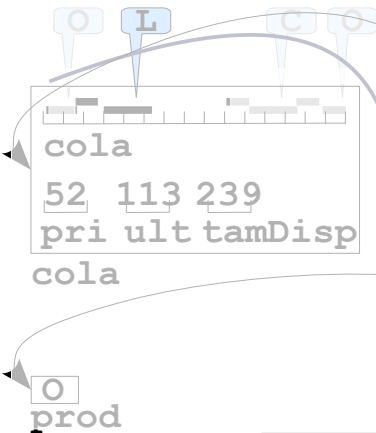


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

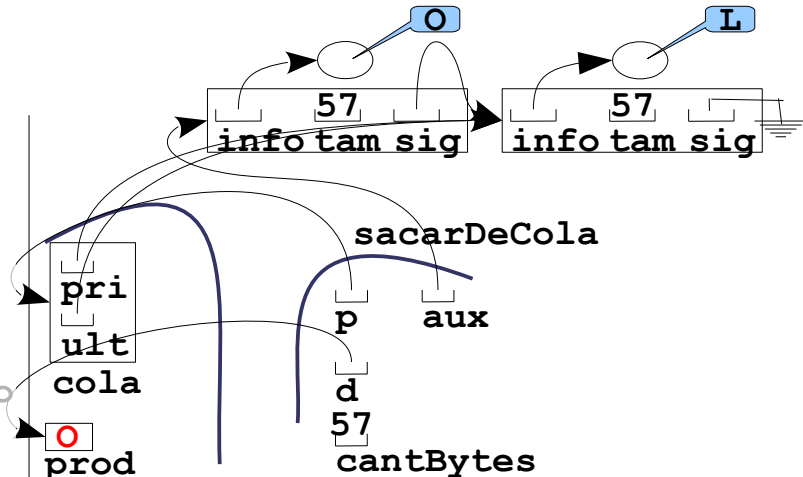
Implementación estática



sacarDeCola



Implementación dinámica



sacarDeCola

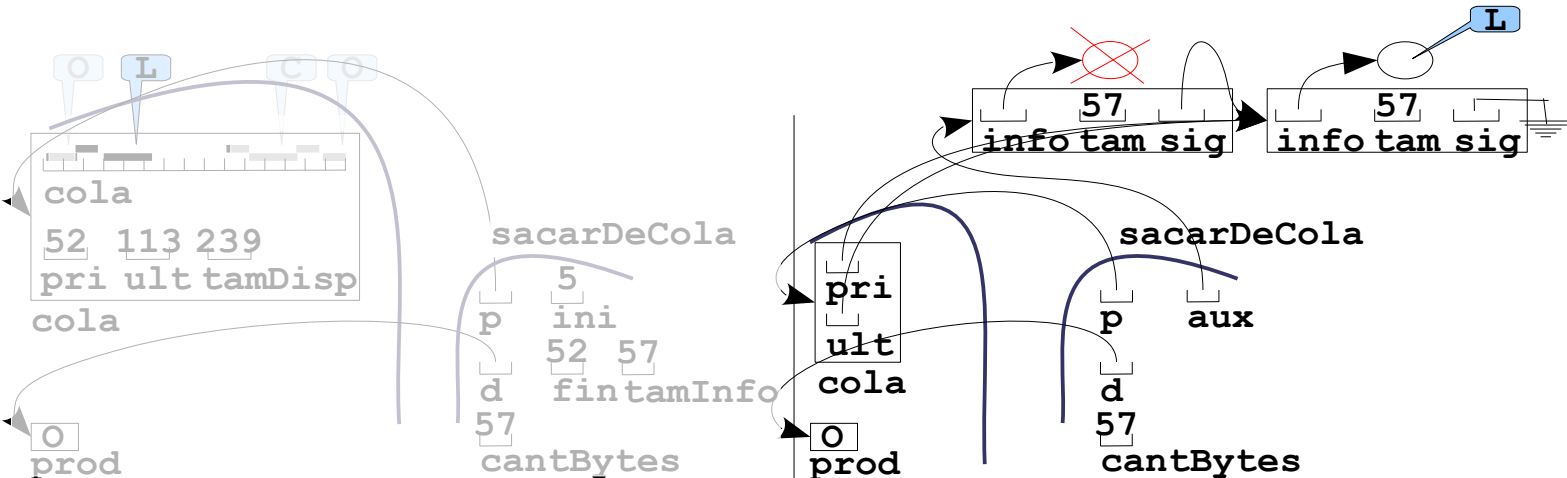
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



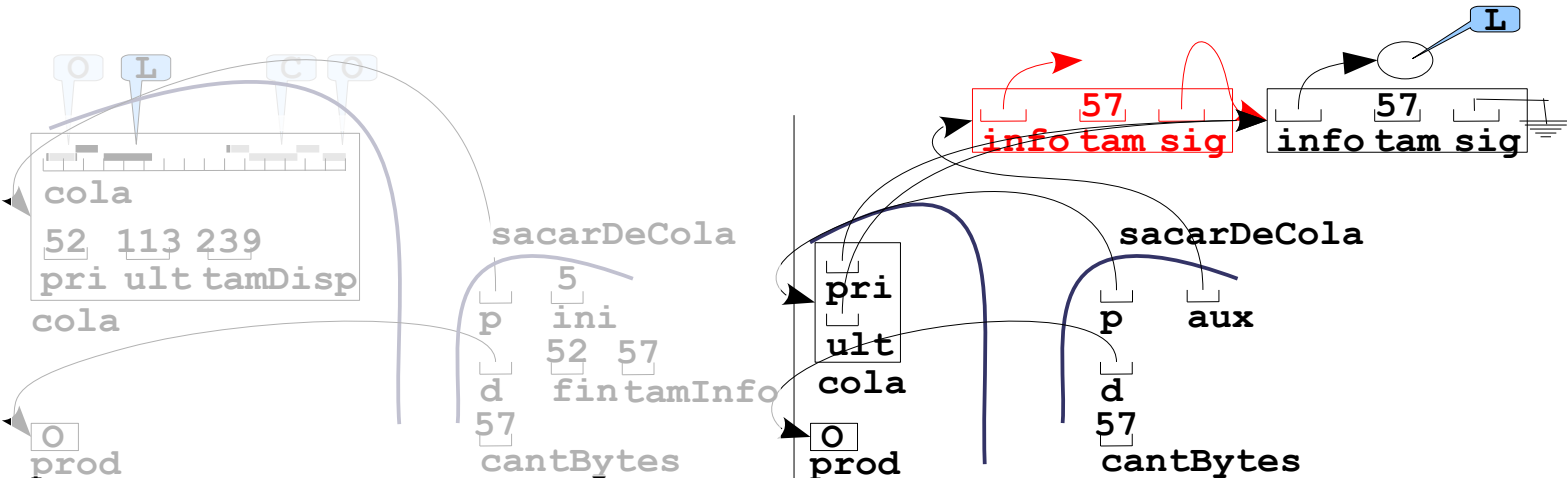
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```


Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



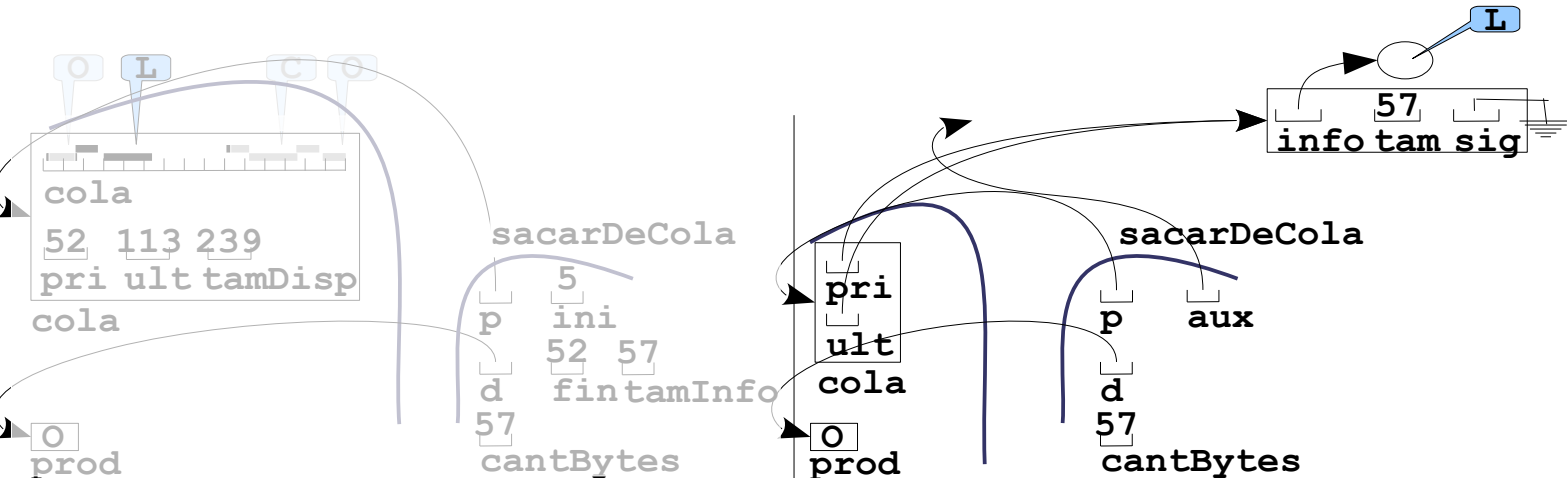
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



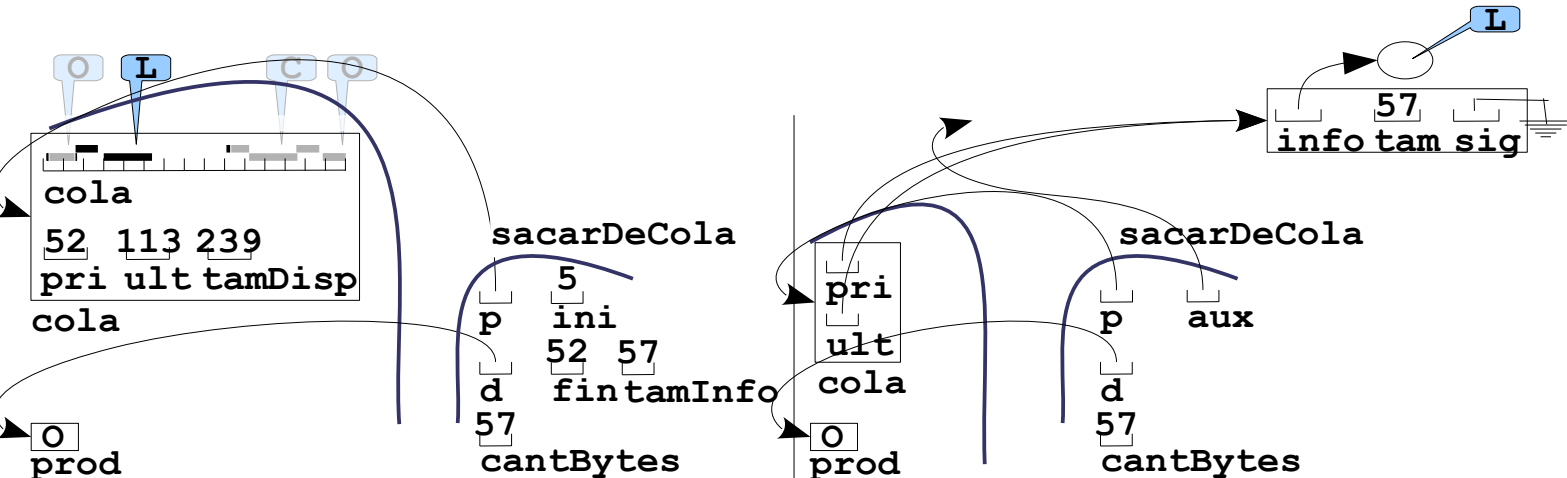
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99
100
101
102
103
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



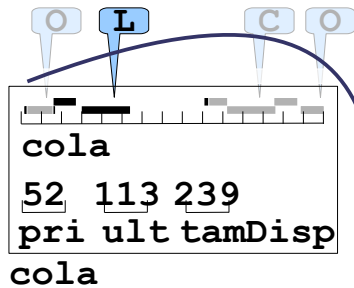
```
main.c x main.h x productos.c x product
70 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
71 {
72     82 p->pri = fin;
73     83 p->tamDisp = m;
74     84 tamInfo = m;
75     85 if((ini = m)
76     86     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
77     87 if((fin = ta
78     88     memcpy(
79     89 p->pri = fin;
80     90 return 1;
81     91 }
82 }

main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
54 int sacarDeCola(tCola *p, void *d, unsigned cantBytes)
55 {
56     tNodo *aux = p->pri;
57     if(aux == NULL)
58         return 0;
59     p->pri = aux->sig;
60     memcpy(d, aux->info, minimo(aux->tamInfo, cantBytes));
61     free(aux->info);
62     free(aux);
63     if(p->pri == NULL)
64         p->ult = NULL;
65     return 1;
66 }
```

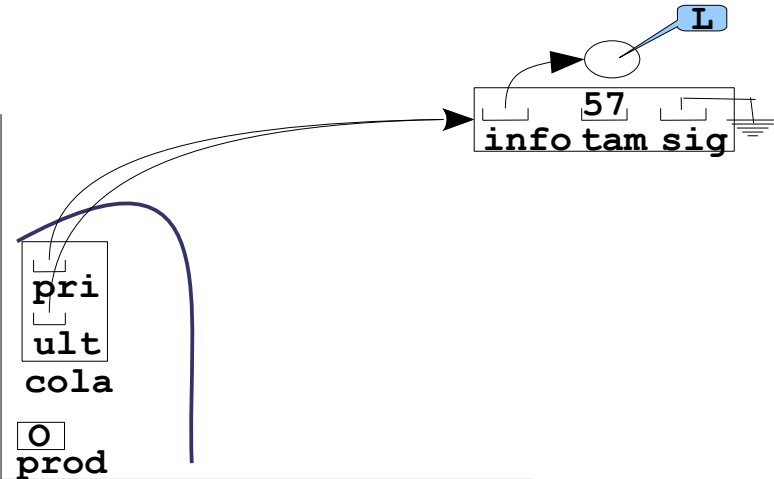
Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática



Implementación dinámica



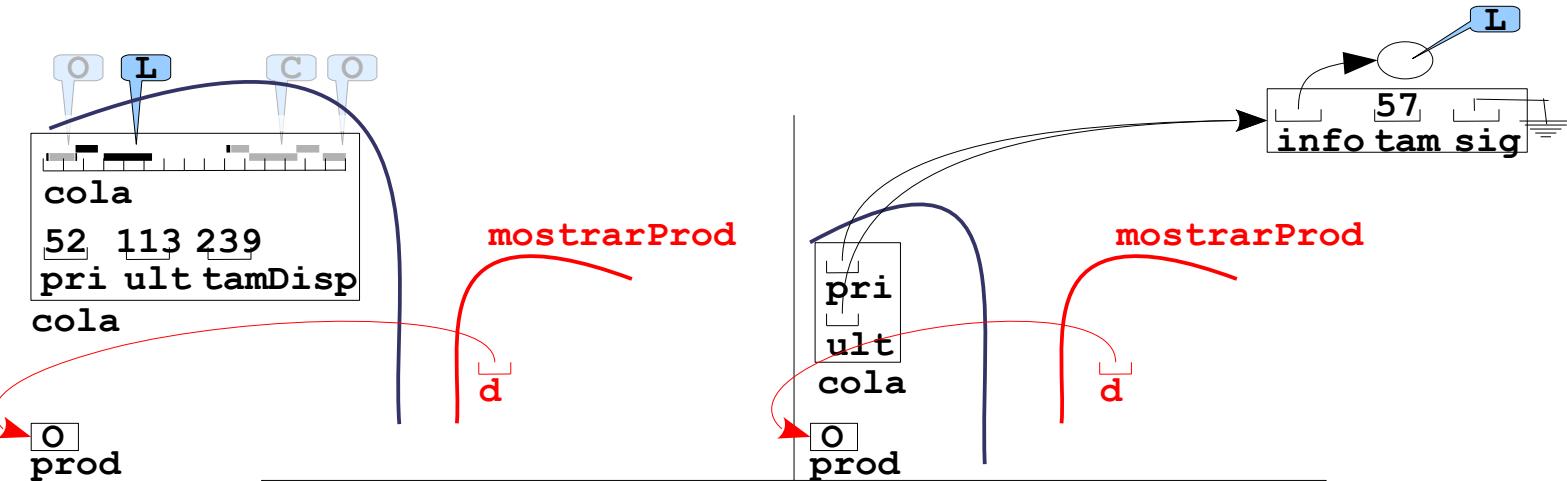
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100 if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101 puts("ERROR - cola vacia.");
102 else
103 mostrarProducto(&prod);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica



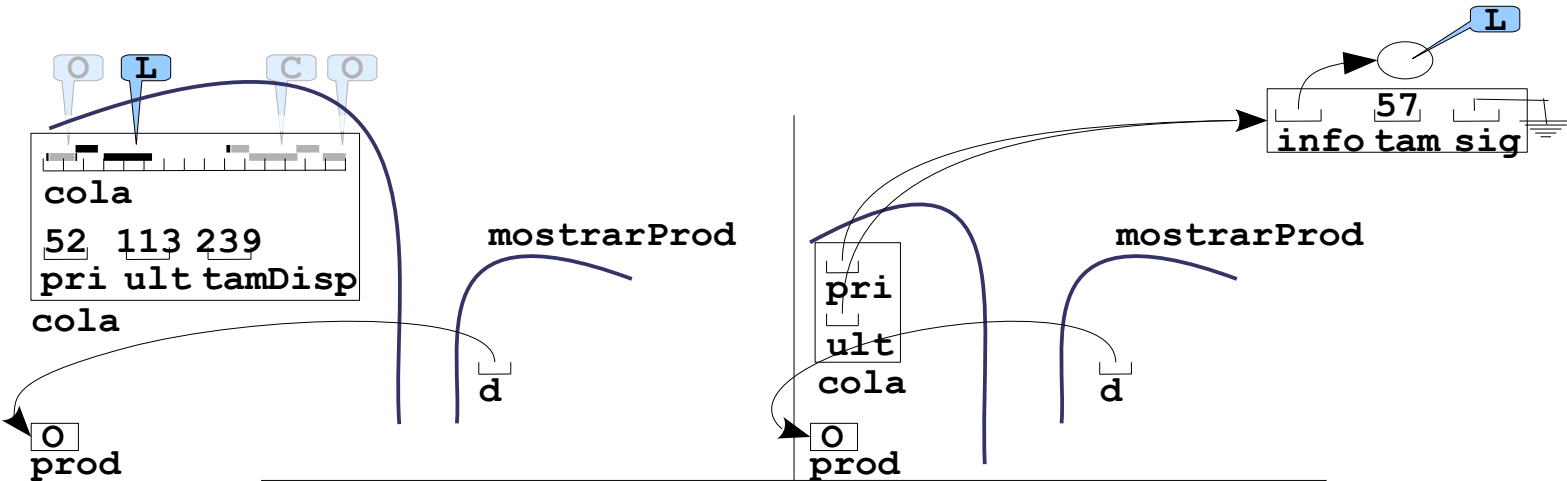
```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

Tipo de dato Cola.

Implementación estática

Implementación dinámica

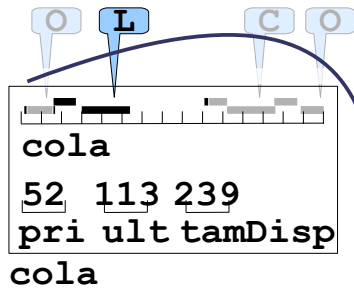


```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         mostrarProducto(&prod);
```

Tipos de Datos Abstractos

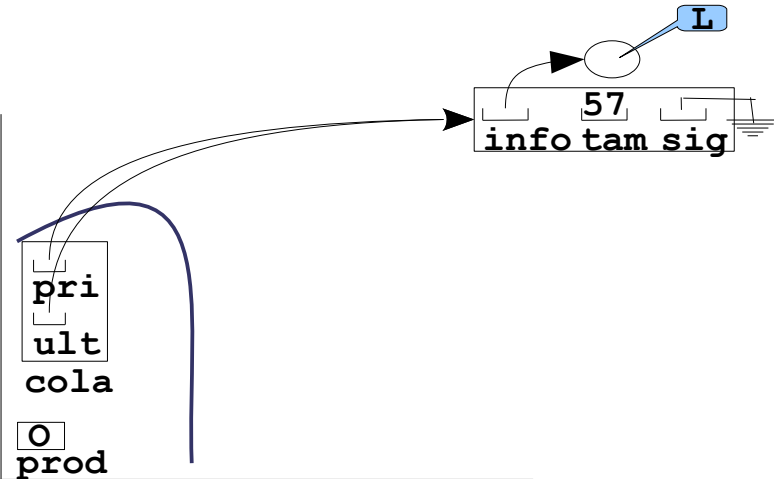
Tipo de dato Cola.

Implementación estática



O
prod

Implementación dinámica



```
main.c x main.h x productos.c x productos.h x cola.c x cola.h x cola.c x cola.h x
99 while(!colaVacia(&cola))
100     if(!sacarDeCola(&cola, &prod, sizeof(prod)))
101         puts("ERROR - cola vacia.");
102     else
103         [mostrarProducto(&prod);
```

