

a) void eliminar (Lista L, int x)

```

{
    int aux, p;
    for (p=primero(L); p!=fin(L);)
    {
        aux=elemento (p,L);
        if (aux==x)
            borrar (p,L);
        else p++;
    }
}

```

Complexity analysis for code a):

- int aux, p; —  $O(1)$
- for (p=primero(L); p!=fin(L);) —  $O(1)$
- aux=elemento (p,L); —  $O(n)$
- if (aux==x) —  $O(1)$
- borrar (p,L); —  $O(n)$
- else p++; —  $O(1)$
- Total complexity:  $O(n^2)$

b) void eliminar (Lista L, int x)

```

{
    int aux, p;
    for (p=primero(L); p!=fin(L);)
    {
        aux=elemento (p,L);
        if (aux==x)
            borrar (p,L);
        else p++;
    }
}

```

Complexity analysis for code b):

- int aux, p; —  $O(1)$
- for (p=primero(L); p!=fin(L);) —  $O(1)$
- aux=elemento (p,L); —  $O(1)$
- if (aux==x) —  $O(1)$
- borrar (p,L); —  $O(1)$
- else p++; —  $O(1)$
- Total complexity:  $O(n^2)$

¿Cómo mejorarías esa eficiencia con un ligero cambio en el código?

```

void eliminar(Lista L, int x) {
    int aux, p;
    int final = fin(L);
    for (p=primero(L); p!=final;){
        aux=elemento(p, L);
        if (aux == x)
            borrar(p, L);
        else p++;
    }
}

```

Complexity analysis for the improved code:

- int aux, p; —  $O(1)$
- int final = fin(L); —  $O(1)$
- for (p=primero(L); p!=final;){ —  $O(1)$
- aux=elemento(p, L); —  $O(1)$
- if (aux == x) —  $O(1)$
- borrar(p, L); —  $O(1)$
- else p++; —  $O(1)$
- Total complexity:  $O(n)$

c) void eliminar (Lista L, int x)

```

{
    int aux, p;
    for (p=primero(L); p!=fin(L);)
    {
        aux=elemento (p,L);
        if (aux==x)
            borrar (p,L);
        else p++;
    }
}

```

Complexity analysis for code c):

- int aux, p; —  $O(1)$
- for (p=primero(L); p!=fin(L);) —  $O(1)$
- aux=elemento (p,L); —  $O(1)$
- if (aux==x) —  $O(1)$
- borrar (p,L); —  $O(1)$
- else p++; —  $O(1)$
- Total complexity:  $O(n)$

¿Puede mejorarse la eficiencia con un ligero cambio en el código?

No se puede, porque puede pasar que un elemento de los que se quiere borrar sea el último de la lista, y hay que recorrer todos sus elementos. Es decir, es lo más eficiente que se puede.