

---

## Fusió i suma dels elements de dos llistes doblement encadenades, circulars i amb fantasma

---

X67932\_ca

Donada la classe *Llista* que permet guardar seqüències d'enters amb una llista doblement encadenada, amb fantasma i circular, cal implementar el mètode

```
void fusiona_suma(Llista &l2, nat n);
```

que fusiona els elements del paràmetre implícit i de *l2* agafant *n* elements del paràmetre implícit i *n* elements de *l2* alternativament (o els que quedin si n'hi ha menys de *n*). Al principi del paràmetre implícit s'afegeix un nou element que conté la suma de tots els elements del paràmetre implícit i *l2*. La llista *l2* queda buida.

Per exemple, si inicialment tenim aquestes dues llistes:

```
p.i. [2 5 3 8 4]
l2   [1 6 9]
```

després de cridar *fusiona\_suma* amb *n* = 2, les dues llistes quedaran així:

```
p.i. [38 2 5 1 6 3 8 9 4]
l2   []
```

Pots veure més exemples en els jocs de prova públics. Cal enviar a jutge.org només la implementació del mètode *fusiona\_suma*. Indica dins d'un comentari a la capçalera del mètode el seu cost en funció del nombre d'elements *n1* de la llista del p.i. i nombre d'elements *n2* de la llista *l2*.

Per testejar la solució, jutge.org ja té implementats la resta de mètodes de la classe *Llista* i un programa principal que processa línies d'enters amb els que crea dues llistes i després crida diverses vegades el mètode *fusiona\_suma* amb diferents valors de *n*.

### Entrada

L'entrada conté dues línies formades per seqüències d'enters, són els elements que tindran les dues llistes inicials. A continuació segueix una seqüència d'enters que representen diferents valors de *n*.

### Sortida

Per a cada valor *n* d'entrada es fusionen i es sumen els elements de les dues llistes inicials agafant alternativament *n* elements de cadascuna i s'escriu tres línies: El valor *n*, el contingut de la primera llista i el contingut de la segona llista. Per cada llista s'escriu el nombre d'elements de la llista seguit d'un espai, els elements de la llista entre claudàtors i separats per espais, i finalment aquests mateixos elements però amb ordre invers, també entre claudàtors i separats per espais.

### Observació

Cal enviar la solució (el fitxer *solution.cpp*) comprimida en un fitxer *.tar*:

```
tar cvf solution.tar solution.cpp
```

Només cal enviar la implementació del mètode *fusiona\_suma* i el seu cost en funció del nombre d'elements  $n1$  i  $n2$  de les dues llistes inicials. Segueix estrictament la definició de la classe de l'enunciat.

### Exemple d'entrada 1

```
2 5 3 8 4
1 6 9
1
2
3
4
5
6
```

### Exemple de sortida 1

```
1
9 [38 2 1 5 6 3 9 8 4] [4 8 9 3 6 5 1 2 38]
0 [] []
2
9 [38 2 5 1 6 3 8 9 4] [4 9 8 3 6 1 5 2 38]
0 [] []
3
9 [38 2 5 3 1 6 9 8 4] [4 8 9 6 1 3 5 2 38]
0 [] []
4
9 [38 2 5 3 8 1 6 9 4] [4 9 6 1 8 3 5 2 38]
0 [] []
5
9 [38 2 5 3 8 4 1 6 9] [9 6 1 4 8 3 5 2 38]
0 [] []
6
9 [38 2 5 3 8 4 1 6 9] [9 6 1 4 8 3 5 2 38]
0 [] []
```

### Exemple d'entrada 2

```
3 -6 8 0 4 -2
-5
1
2
3
4
5
6
7
```

### Exemple de sortida 2

```
1
8 [2 3 -5 -6 8 0 4 -2] [-2 4 0 8 -6 -5 3 2]
0 [] []
2
8 [2 3 -6 -5 8 0 4 -2] [-2 4 0 8 -5 -6 3 2]
0 [] []
3
8 [2 3 -6 8 -5 0 4 -2] [-2 4 0 -5 8 -6 3 2]
0 [] []
4
8 [2 3 -6 8 0 -5 4 -2] [-2 4 -5 0 8 -6 3 2]
0 [] []
5
8 [2 3 -6 8 0 4 -5 -2] [-2 -5 4 0 8 -6 3 2]
0 [] []
6
8 [2 3 -6 8 0 4 -2 -5] [-5 -2 4 0 8 -6 3 2]
0 [] []
7
8 [2 3 -6 8 0 4 -2 -5] [-5 -2 4 0 8 -6 3 2]
0 [] []
```

### Exemple d'entrada 3

```
-5
3 -6 8 0 4 -2
1
2
```

### Exemple de sortida 3

```
1
8 [2 -5 3 -6 8 0 4 -2] [-2 4 0 8 -6 3 -5 2]
0 [] []
2
8 [2 -5 3 -6 8 0 4 -2] [-2 4 0 8 -6 3 -5 2]
0 [] []
```

### Exemple d'entrada 4

3 -6 8 0 4 -2

1

2

### Exemple d'entrada 5

3 -6 8 0 4 -2

1

2

### Exemple d'entrada 6

1

2

### Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T21:21:43.215Z

© *Jutge.org*, 2006–2026.

<https://jutge.org>

### Exemple de sortida 4

1

7 [7 3 -6 8 0 4 -2] [-2 4 0 8 -6 3 7]

0 [] []

2

7 [7 3 -6 8 0 4 -2] [-2 4 0 8 -6 3 7]

0 [] []

### Exemple de sortida 5

1

7 [7 3 -6 8 0 4 -2] [-2 4 0 8 -6 3 7]

0 [] []

2

7 [7 3 -6 8 0 4 -2] [-2 4 0 8 -6 3 7]

0 [] []

### Exemple de sortida 6

1

1 [0] [0]

0 [] []

2

1 [0] [0]

0 [] []