
Longitud de segments amb elements d'igual valor als extrems d'una llista

X42679_ca

Escriviu un programa que simuli el manteniment d'una llista d'enters, a base de llegir instruccions que la van actualitzant, i instruccions que la van consultant. La llista d'enters se suposa inicialment buida, i les instruccions son dels següents tipus: afegir elements al principi o final de la llista, treure elements del principi o final de la llista, o consultar quin és el nombre d'elements iguals i consecutius que es troben al principi de la llista, o al final de la llista.

És obligatori utilitzar només el constructor de tipus `list` com a mecanisme d'emmagatzemament massiu de dades. En particular, no es pot usar ni `vector`, ni `stack`, ni `queue`. Podeu declarar tants `list` com vulgueu i del tipus que vulgueu (que no sigui cap altre mecanisme d'emmagatzemament massiu), i podeu usar iteradors sobre llistes.

Entrada

La entrada consisteix en un nombre arbitrari de línies, cadascuna amb una instrucció. Les instruccions poden ser de la següent forma:

```
push_front x
push_back x
pop_front
pop_back
max_equal_left
max_equal_right
```

Sortida

Les instruccions `pop_front` i `pop_back` poden escriure `error` a la sortida, i les instruccions `max_equal_left` i `max_equal_right` escriuran un valor a la sortida. Se sobreentén que la llista que simulem està inicialment buida, i que l'efecte de cada instrucció és el següent:

- `push_front x` afegeix l'enter `x` al principi de la llista.
- `push_back x` afegeix l'enter `x` al final de la llista.
- `pop_front` elimina l'element que es troba al principi de la llista. Si la llista és buida ha d'escriure `error` i salt de línia.
- `pop_back` elimina l'element que es troba al final de la llista. Si la llista és buida ha d'escriure `error` i salt de línia.
- `max_equal_left` escriu quin és el nombre d'elements iguals i consecutius que es troben començant des de la part inicial de la llista.
- `max_equal_right` escriu quin és el nombre d'elements iguals i consecutius que es troben començant des de la part final de la llista.

Exemple d'entrada 1

```
max_equal_left
max_equal_right
pop_front
pop_back
push_front 0
max_equal_left
max_equal_right
push_back 3
push_back 0
max_equal_left
max_equal_right
pop_back
push_back 3
push_front 0
max_equal_left
max_equal_right
push_front 3
push_back 3
max_equal_left
max_equal_right
push_back 3
max_equal_left
max_equal_right
push_back 0
max_equal_left
max_equal_right
push_front 5
push_back 6
push_back 6
push_back 0
max_equal_left
max_equal_right
```

Exemple d'entrada 2

```
push_front 0
pop_back
push_back 0
max_equal_left
push_front 0
push_front 3
max_equal_left
max_equal_right
push_front 3
pop_front
push_front 3
push_front 3
push_back -1
max_equal_right
max_equal_right
max_equal_right
push_front 2
push_front 2
max_equal_right
push_front -2
push_front -1
push_front 2
push_back 2
push_front 2
```

Exemple de sortida 1

```
0
0
error
error
1
1
1
1
1
2
2
1
3
1
4
1
1
1
1
1
```

```
push_front -2
push_front -2
push_back -2
pop_front
pop_back
push_back -2
pop_front
push_front 3
push_back 0
push_back 0
max_equal_left
push_front 2
push_front 2
push_front 2
pop_back
push_back 2
pop_back
push_front 0
max_equal_right
push_front 0
push_back -1
pop_front
push_front -1
push_back -1
push_front -2
push_back -2
```

Exemple de sortida 2

1
1
2
1

1
1
1
1
1

Observació

Els jocs de proves privats són grans. Per tal d'aconseguir superar-los tots i obtenir així la màxima nota, convindrà trobar un enfoc astut que faci que totes les instruccions tinguin cost constant. Això sí, només es pot utilitzar el tipus de dades `list`, com ja hem mencionat anteriorment.

Informació del problema

Autoria: PRO1

Generació: 2026-01-25T16:04:56.919Z

© *Jutge.org*, 2006–2026.

<https://jutge.org>