Mètode de la classe pila que divideix una pila en dues pilesX58096_ca

Implementa un nou mètode de la classe `Stack` que divideix els elements de la pila entre el paràmetre implícit i la pila que passen com a paràmetre. Els elements en posició senar començant des del fons estaran en el paràmetre implícit i els elements en posició parell començant des del fons en la pila que ens passen com a paràmetre.

D'entre els fitxers que s'adjunten en aquest exercici, trobaràs `stack.old.hpp`, a on hi ha una implementació de la classe genèrica `Stack`. En primer lloc, hauràs de fer:

```
cp stack.old.hpp stack.hpp
```

A continuació si obres el fitxer `stack.hpp` al final del mateix trobaràs el mètode que has d'implementar:

```
// Pre: Sigui [a1,...,an] el contingut actual de la pila des
// del fons fins al top, s és una pila buida.
// Post: El contingut de pi és [a1, a3, a5, ...] (manté els elements
// en posicions senars començant des del fons) i el contingut de s
// és [a2, a4, ...] (conté els elements en posicions parells començant
// des del fons de la pila).
void split(Stack &s);
```

IMPORTANT: No toquis la resta de la implementació de la classe, excepte si per algun motiu, consideres que necessites afegir algun mètode auxiliar o atribut a la part privada.

D'entre els fitxers que s'adjunten a l'exercici també hi ha `program.cpp` (programa principal) i `Makefile` per a compilar i generar l'executable. El programa principal que t'oferim ja s'encarrega de llegir les piles i fer les crides al mètode indicat. **Només cal que implementis el mètode `split`.**

Per a pujar la teva solució, has de crear el fitxer `solution.tar` així:

```
tar cf solution.tar stack.hpp
```

Observació 1

Hauries d'aconseguir implementar el mètode demanat a base d'intercanviar els punters de l'objecte. De fet, una solució a base d'usar `push` i `pop` potser et permetrà passar els jocs de proves, però atès que la solució ha de ser eficient en temps i espai, aquest tipus de solució serà fortament penalitzat.

Observació 2

Recorda que si crees funcions auxiliars, has d'afegir-hi les corresponents **Precondició** (Pre) i **Postcondició** (Post). En els bucles inclou l'**invariant del bucle** (Inv) i la **funció de fita** (FF). En les crides recursives inclou la **hipòtesi d'inducció** (HI) i la **funció de fita** (FF).

Entrada

L'entrada del programa és una seqüència de piles. Per llegir les piles, s'utilitza l'operador `>>` que es troba definit en el fitxer `stack.hpp`.

Sortida

Per cada pila s'escriurà el resultat del mètode `split`, és a dir, les dues piles (paràmetre implícit i paràmetre). Per escriure les piles, s'ha utilitzat l'operador `<<` que es troba definit en el fitxer `stack.hpp`.

Exemple d'entrada 1

```
10  1 2 3 4 5 6 7 8 9 10
10  2 4 6 8 10 12 14 16 18 20
9   -1 -2 -3 -4 -5 -6 -7 -8 -9
3   100 1000 10000
```

Exemple de sortida 1

```
s: 5 1 3 5 7 9
s2: 5 2 4 6 8 10
---
s: 5 2 6 10 14 18
s2: 5 4 8 12 16 20
---
s: 5 -1 -3 -5 -7 -9
s2: 4 -2 -4 -6 -8
---
s: 2 100 10000
s2: 1 1000
---
```

Exemple d'entrada 2

```
2  -9 -7
1  3
17  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
16  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Exemple de sortida 2

```
s: 1 -9
s2: 1 -7
---
s: 1 3
s2: 0
---
s: 9 0 0 0 0 0 0 0 0 0
s2: 8 0 0 0 0 0 0 0 0
---
s: 8 0 0 0 0 0 0 0 0
s2: 8 0 0 0 0 0 0 0 0
---
```

Informació del problema

Autoria: Bernardino Casas

Generació: 2026-01-25T21:17:28.359Z