Suma d'una pila — X25739_ca

Preliminars
En aquest exercici extendrem la classe `Stack` suposant que el tipus `T` dels elements de la pila té definida la operació de suma `+`, és a dir, que dues variables de tipus `T` es poden sumar. Aquesta suma satisfarà la propietat d'associativitat usual ($x + (y + z) = (x + y) + z$), i tindrà element neutre (per exemple, 0 és el neutre per a la suma d'enters, i l'string buit és el neutre per a la suma d'strings).
També suposem que una variable `x` de tipus `T` permet una assignació `x = 0`, de manera que, internament, a `x` se li assigna el neutre de la suma.
Testejarem l'exercici amb el tipus `int` i amb un tipus contenidor d'`string` que permetrà fer assignacions `x = 0` (internament s'assignarà l'string buit). Fixeu-vos que en el cas d'`strings` la suma és de fet la concatenació d'`strings`, i que no és commutativa. Per exemple, "a"+"b" = "ab" ≠ "ba" = "b"+"a".
Els jocs de proves grans d'aquest exercici es fan amb `int` i no pas amb `string` per tal d'evitar el cost adicional de concatenar strings grans.
Exercici
Implementeu un nou mètode de la classe Stack que retorni la suma dels elements continguts a la pila des del fons fins al top. És a dir, si $[a_1, \ldots, a_n]$ és el contingut de la pila des del fons fins al top, el mètode ha de retornar $a_1 + \cdots + a_n$.
D'entre els fitxers que s'adjunten en aquest exercici, trobareu `stack.hh`, a on hi ha una implementació de la classe genèrica `Stack`. Haureu de buscar dins `stack.hh` les següents línies:

```
// Pre:  Sigui [a1,...,an] el contingut actual de la pila des del fons fins al t
// Post: Retorna a1+...+an.
// Descomenteu les següents dues linies i implementeu el mètode:
// T sum() {
// }
```

Descomenteu les dues linies que s'indiquen i implementeu el mètode. Potser necessitareu modificar més coses de la classe depenent de quin enfoc seguiu. Aquí us en recomanem dos:

- **Enfoc senzill i ineficient:** una implementació senzilla del mètode `sum`, inicialitzant una variable de tipus T a 0 (que es transformarà automàticament en el neutre de la suma), i recorrent i sumant-hi els elements de la pila, hauria de ser suficient per a superar els jocs de proves públics, però no els privats.
- **Enfoc eficient:** la idea és afegir un nou camp de tipus T a l'`struct Item` que guardi, per a cada item, la suma de tot el que hi ha per sota a la pila més el `value` d'aquest mateix `Item`. Per exemple, podem anomenar `sumbelow` a aquest nou camp. Cada cop que fem un push, haurem d'inicialitzar el camp `sumbelow` del nou `Item` com la suma del `sumbelow` del que era fins ara el top de la pila més el `value` que rebem com a paràmetre.

D'entre els fitxers que s'adjunten a l'exercici també hi ha `main.cc` (programa principal), i el podeu compilar directament, doncs inclou `stack.hh`. Només cal que pugeu `stack.hh` al jutge.

Entrada

L'entrada del programa té una primera línia amb o bé `int` o bé `string`, que indica el tipus T dels elements de la pila `s` amb la que treballarà el programa, que se suposa inicialment buida. Després, hi ha una seqüència d'instruccions del següent tipus que s'aniran aplicant sobre la pila:

```
push x (x és de tipus T)
pop
top
print
sum
```

Se suposa que la seqüència d'entrada serà correcta (sense pop ni top sobre pila buida).
El programa principal que us oferim ja s'encarrega de llegir aquestes entrades i fer les crides als corresponents mètodes de la classe pila. Només cal que implementeu el mètode abans esmentat.

Sortida

Per a cada instrucció `top`, s'escriurà el top actual de la pila, per a cada instrucció `print`, s'escriurà el contingut de la pila, i per a cada instrucció `sum`, s'escriurà la suma dels elements de la pila des del fons fins al top. El programa que us oferim ja fa això. Només cal que implementeu el mètode abans esmentat.

Exemple d'entrada 1

```
int
sum
push 10
top
print
sum
push 20
top
print
sum
push 30
top
print
sum
pop
top
print
sum
push 31
top
print
sum
push -40
top
print
sum
pop
pop
sum
```

```
pop
pop
sum
```

Exemple de sortida 1

```
0
10
10
10
20
10 20
30
30
10 20 30
60
20
10 20
30
31
10 20 31
61
-40
10 20 31 -40
21
30
0
```

Exemple d'entrada 2

```
string
sum
push a
top
print
sum
push bb
top
print
sum
push ccc
top
print
sum
pop
top
print
sum
push dd
top
print
sum
push eeee
top
print
sum
pop
pop
sum
pop
pop
sum
```

Exemple de sortida 2

```
a
a
a
bb
a bb
abb
ccc
a bb ccc
abbccc
bb
a bb
abb
dd
a bb dd
abbdd
eeee
a bb dd eeee
abbddeeee
abb
```

Exemple d'entrada 3

```
int
push 3
push -10
push 15
pop
push 17
push 20
sum
top
top
sum
top
push 18
top
pop
pop
sum
push 12
push -7
pop
top
pop
push 19
sum
```

```
sum
push -16
sum
push 0
sum
push -4
push 12
push -19
pop
pop
top
push -4
sum
top
pop
push 10
push 5
top
```

Exemple de sortida 3

```
30
20
20
30
20
18
10
12
29
29
13
13
-4
5
-4
5
```

Exemple d'entrada 4

```
string
push a
top
push b
push acc
top
top
sum
top
push a
pop
pop
sum
push cc
pop
top
pop
push d
sum
push b
push ccc
push ccb
pop
top
push cab
push d
pop
sum
push c
sum
sum
pop
push ddc
push bc
push bc
top
sum
pop
push aa
sum
```

```
top
top
```

Exemple de sortida 4

```
a
acc
acc
abacc
acc
ab
b
ad
```

```
ccc
adbccccab
adbccccabc
adbccccabc
bc
adbccccabddcbcbc
adbccccabddcbcaa
aa
aa
```

Observació

Avaluació sobre 10 punts: (Afegiu comentaris si el vostre codi no és prou clar)

- Solució lenta: 5 punts.
- solució ràpida: 10 punts.

Entenem com a solució ràpida una que és correcta, on totes les operacions tenen cost constant (excepte `PRINT`, i suposant que operar amb elements de tipus T té cost constant), i capaç de superar els jocs de proves públics i privats. Entenem com a solució lenta una que no és ràpida, però és correcta i capaç de superar els jocs de proves públics.

Informació del problema

Autoria: PRO2

Generació: 2026-01-27T18:52:06.606Z