

---

**Imprimir expressions****Z14339\_ca**

---

En aquest exercici considerarem arbres que representen expressions sobre els operadors +, -, \*, i sobre operands que són nombres naturals. Per exemple, el següent arbre

```
*
|-- -
|   |-- 7
|   |-- 1
|   |-- 5
|   '-- 2
'-- +
    |-- 7
    |-- 3
    '-- 8
```

representa l'expressió  $((7 - 1 - 5 - 2) * (7 + 3 + 8))$ .

Tots els nodes de l'arbre que són fulles contenen un valor natural, i els nodes amb fills sempre en tenen almenys 2 i el seu valor és sempre un operador. No hi ha subarbres buits. Implementeu, doncs, la funció següent:

```
/**
 * @brief Transforma una expressió en la seva representació com a 'string'.
 *
 * Una expressió està formada per operadors ('+', '*', i '-'), i
 * operands (naturals), que són nodes d'un arbre. Els operands tenen dos
 * fills o més, i els operands són fulles (no tenen fills).
 *
 * L'expressió representada com a 'string' és de la següent manera. Per a
 * operands: cal retornar l'operand mateix. Per a operadors, cal retornar
 * els fills de l'operador, separats per l'operador, amb un espai entre
 * operand i operadors, i tot el conjunt sempre entre parèntesis.
 *
 * @pre L'arbre representa una expressió ben formada
 *
 * @param t L'arbre que representa l'expressió.
 * @returns La representació com a 'string' de 't'
 */
string expression_to_string(Tree<string> t);
```

**Observació**

Els fitxers públics (icona del gatet) contenen:

tree.hh	la classe Tree
tree-io.hh	l'entrada/sortida de Tree
main.cc	el programa principal

També hi ha un `Makefile` i el directori `.vscode` que té la configuració per compilar i depurar amb VSCode.

Cal implementar `expression_to_string` en un **fitxer** `.cc` **nou**, compilar, i finalment **enviar només el fitxer amb la funció**.

## Entrada

Cada cas consisteix en una representació textual d'una expressió del tipus definit. (Aquesta lectura ja la fa el programa principal.)

## Sortida

Per a cada cas, la sortida és el resultat de transformar l'expressió a la seva representació textual, amb cada resultat en una línia separada. (Això també ho fa el programa principal.)

### Exemple d'entrada

```
*
|-- 1
'-- 5

*
|-- -
|   |-- 7
|   |-- 1
|   |-- 5
|   '--- 2
'--- +
      |-- 7
      |-- 3
      '--- 8

*
|-- +
|   |-- 6
|   '--- 5
'--- 7

3

*
|-- 4
|-- 7
|-- +
|   |-- 1
|   |-- 6
|   |-- 7
|   |-- 6
|   '--- 9
'--- 6
```

### Exemple de sortida

```
(1 * 5)
((7 - 1 - 5 - 2) * (7 + 3 + 8))
((6 + 5) * 7)
3
(4 * 7 * (1 + 6 + 7 + 6 + 9) * 6)
```

## Informació del problema

Autor : Pau Fernández

Generació : 2025-03-20 19:55:04

© *Jutge.org*, 2006–2025.  
<https://jutge.org>