Imprimir expresiones Z14339_es

En este ejercicio consideraremos árboles que representan expresiones sobre los operadores `+`, `-`, `*`, y sobre operandos que son números naturales. Por ejemplo, el siguiente árbol

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

representa la expresión `((7 - 1 - 5 - 2) * (7 + 3 + 8))`.

Todos los nodos del árbol que son hojas contienen un valor natural, y los nodos con hijos siempre tienen al menos 2 y su valor es siempre un operador. No hay subárboles vacíos. Implementad, pues, la función siguiente:

```
/**
 * @brief Transforma una expresión en su representación como `string`.
 *
 * Una expresión está formada por operadores (`+`, `*` y `-`), y
 * operandos (naturales), que son nodos de un árbol. Los operadores tienen dos
 * hijos o más, y los operandos son hojas (no tienen hijos).
 *
 * La expresión representada como `string` es de la siguiente manera. Para
 * operandos: hay que devolver el operando mismo. Para operadores, hay que devo
 * los hijos del operador, separados por el operador, con un espacio entre
 * operando y operadores, y todo el conjunto siempre entre paréntesis.
 *
 * @pre El árbol representa una expresión bien formada
 *
 * @param t El árbol que representa la expresión.
 * @returns La representación como `string` de `t`
 */
string expression_to_string(Tree<string> t);
```

Observación

Los ficheros públicos (icono del gatito) contienen:

`tree.hh`	la clase `Tree`
`tree-io.hh`	la entrada/salida de `Tree`
`main.cc`	el programa principal

También hay un `Makefile` y el directorio `.vscode` que tiene la configuración para compilar y depurar con VSCode.

Hay que implementar `expression_to_string` en un **fichero `.cc` nuevo**, compilar, y finalmente **enviar solo el fichero con la función**.

Entrada

Cada caso consiste en una representación textual de una expresión del tipo definido. (Esta lectura ya la hace el programa principal.)

Salida

Para cada caso, la salida es el resultado de transformar la expresión a su representación textual, con cada resultado en una línea separada. (Esto también lo hace el programa principal.)

Ejemplo de entrada

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

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

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

3

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

Ejemplo de salida

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

Información del problema

Autoría: Pau Fernández
Traducción: Pau Fernández