Push negations P68644_en

We wish to trasform a Boolean expression by pushing all the negations down to the variables. For instance, $\neg(a \wedge b)$ should be trasformed to $(\neg a \vee \neg b)$.

Expressions may contain variables (lowercase letters), conjunctions (`'*'`), disjunctions (`'+'`) and negations (`'!'`). For simplicity, expressions are written in their fully parenthised form. See the examples.

You are given an almost complete program (see attached code `code.cc`). The main function is already written, as well as a a simple *Formula* class that represents Boolean expressions using tree nodes. Here is the *Node* structure of these trees:

```
struct Node {
    char op;        // operand ('a'-'z') or operator ('+' or '*')
    bool neg;       // tells if this node is negated
    Node* left ;    // left subformula
    Node* right ;   // right subformula
};
```

Each leaf node contains a variable in its *op* field. Each non-leaf node contains `'+'` or `'*'` in its *op* field and pointers to its left and right children. In addition, all nodes have a *neg* field that indicates whether or not this node is negated.

The only attribute in the *Formula* class is the *root* of the tree od nodes.

Your task is just to implement the **void** *push_negations* () method of the *Formula* class.

Do do so, you can add private methods to the class, but you cannot alter the *Node* structure nor the existing methods, constructors and destructors. The *main*() function is already written and you do not have to modify it.

Sample input

```
a
!a
!(a*b)
!(a+b)
(!a*!b)
!(a+(b*a))
!(a+(!b*a))
!!!(a+(!!b*a))
(a+!(b+!(c*!(d+!e))))
!(!!c+(!(g+!((!!p+b)+!(!r*t)))*i))
```

Sample output

```
a
!a
(!a+!b)
(!a*!b)
(!a*!b)
(!a*(!b+!a))
(!a*(b+!a))
(!a*(!b+!a))
(a+(!b*(c*(!d*e))))
(!c*((g+((!p*!b)*(!r*t)))+!i))
```

Problem information

Author : Jordi Petit
Generation : 2023-07-15 12:39:27