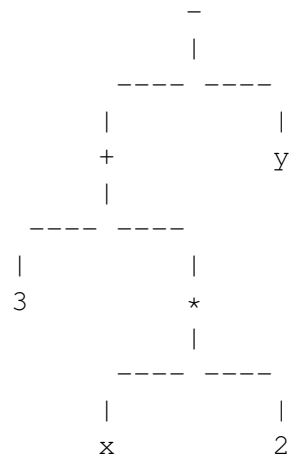


Avaluar expressions amb variables

X59547_ca

INTRODUCCIÓ:

En aquest exercici considerarem arbres que representen expressions sobre els operadors +, -, *, i sobre operands naturals i variables (una variable serà una seqüència de lletres minúscules). Per exemple, el següent arbre representa l'expressió $3+x*2-y$.



EXERCICI:

Implementeu una funció que, donat un arbre binari d'strings que representa una expressió correcta sobre naturals, variables i operadors +, -, *, i també donat un `map<string, int>` que conté els valors de les variables, retorna la seva avaluació de l'expressió. Aquesta és la capçalera:

```

// Pre: t és un arbre no buit que representa una expressió correcta
//       sobre naturals i variables enteres, i els operadors +,-,*.
//       Totes les variables que apareixen a t estan definides a variable2value
//       Les operacions no produeixen errors d'overflow.
// Post: Retorna l'avaluació de l'expressió representada per t.
int evaluate(map<string,int> &variable2value, BinTree<string> t);
  
```

Aquí tenim un exemple de paràmetre d'entrada de la funció i la corresponent sortida:

```

variable2value = {x:2, y:3}
t=
      *
      |
      -----
     |         |
     +         -
     |         |
     -----
    |         |         |         |
    1         x         5         y
  
```

=>

Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `main.cc`, `BinTree.hh`, `evaluate.hh`, `utils.hh`, `utils.cc`. Us falta crear el fitxer `evaluate.cc` amb els corresponents `includes` i implementar-hi la funció anterior. Valdrà la pena que utilitzeu algunes de les funcions oferides a `utils.hpp`. Només cal que pugeu `evaluate.cc` al jutge.

Entrada

La primera línia de l'entrada descriu el format en el que es descriuen els arbres, o bé `IN-LINEFORMAT` o bé `VISUALFORMAT`. Després venen un nombre arbitrari de casos. Cada cas té una primera línia amb una descripció dels valors de les variables (una seqüència de parelles `<variable,valor>`), i després una descripció d'un arbre binari que representa una expressió. Fixeu-vos en que el programa que us oferim ja s'encarrega de llegir aquesta entrada. Només cal que implementeu la funció abans esmentada.

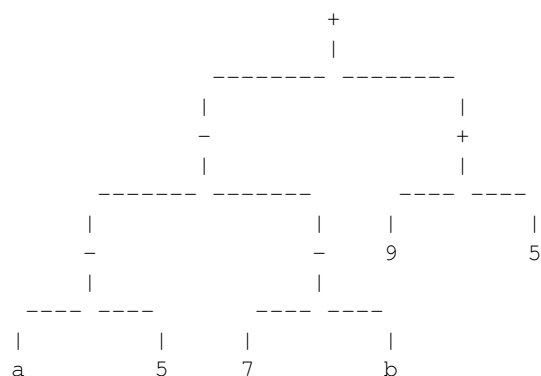
Sortida

Per a cada cas, la sortida conté la corresponent avaluació de l'arbre. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquesta sortida. Només cal que implementeu la funció abans esmentada.

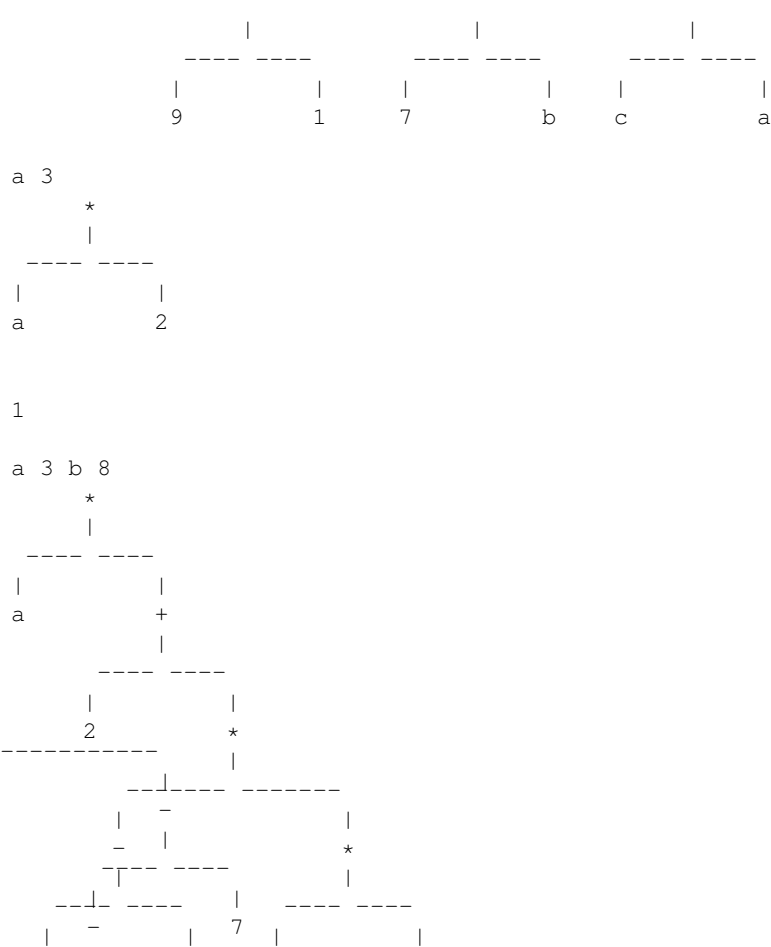
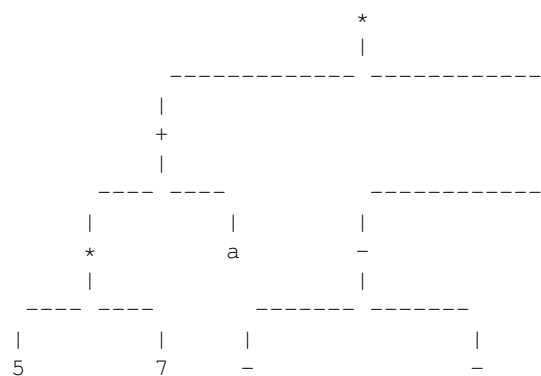
Exemple d'entrada 1

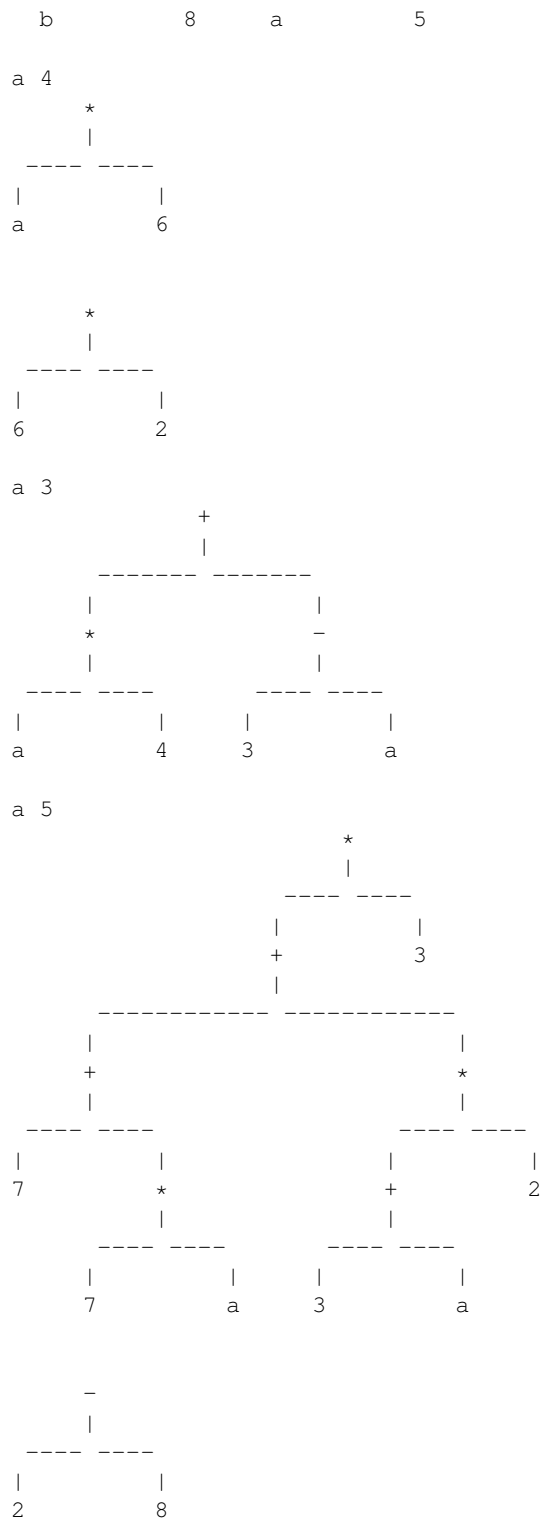
VISUALFORMAT

a 2 b 2



a 6 b 7 c 1





Exemple de sortida 1

```

6
-164
6
1
6
24
12
12
174
-6

```

Exemple d'entrada 2

```

INLINEFORMAT
a 2 b 2
+(-(-(a,5),-(7,b)),+(9,5))
a 6 b 7 c 1
*(+(* (5,7),a),+(-(-9,1),-(7,b)),-(c,a),7))

```

```

a 3
*(a,2)

1
a 3 b 8
*(a,+(2,*(-(b,8),*(a,5))))

```

```
a 4
*(a, 6)

*(6, 2)
a 3
+(*(a, 4), -(3, a))
a 5
*(*(+(7, *(7, a)), *(+(3, a), 2)), 3)

-(2, 8)
```

Exemple de sortida 2

```
6
-164
6
1
6
24
12
12
174
-6
```

Informació del problema

Autor : PRO2

Generació : 2023-10-21 13:43:35

© *Jutge.org*, 2006–2023.

<https://jutge.org>