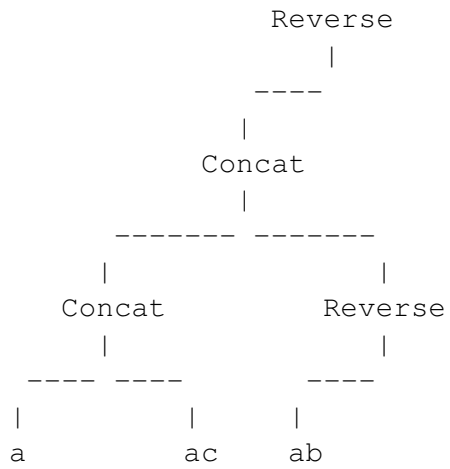
Escriu les primeres lletres de l'avaluació d'una expressió sobre strings i revessar X16311_ca

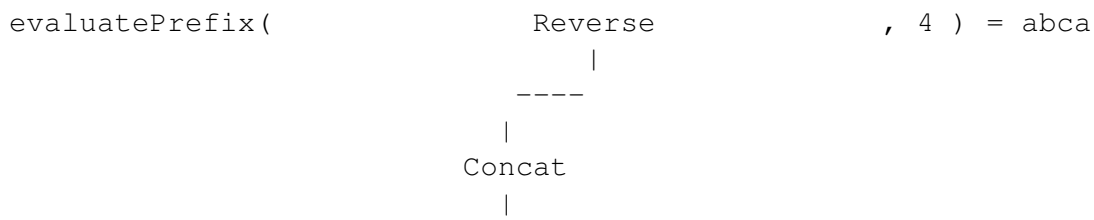
INTRODUCCIÓ:
En aquest exercici considerarem arbres que representen expressions sobre valors de tipus string (de lletres minúscules) i els operadors de concatenació de dos strings i revessat de un string **Concat, Reverse**. En el cas de **Reverse**, que és un operador amb un sol operand, considerarem que aquest operand és sempre el fill esquerra. Per exemple, el següent arbre s'avalua a **abcaa**.

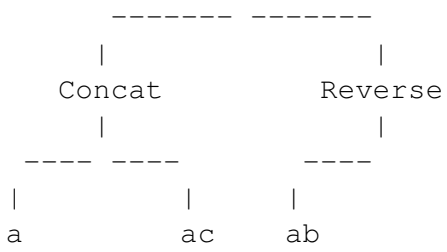


EXERCICI:
Implementeu una funció que, donat un arbre binari `t` d'strings que representa una expressió correcta sobre strings de lletres minúscules i operadors **Concat, Reverse**, i donat un natural `n`, retorna un string amb les `n` primeres lletres de l'avaluació de `t`. En cas que `n` sigui major que la mida de l'avaluació de `t`, llavors simplement retorna l'avaluació de `t`, sense cap caràcter més. Aquesta és la capcelera:

```
// Pre:  t és un arbre no buit que representa una expressió correcta
//       sobre strings de lletres minúscules i els operadors Concat, Reverse.
//       n>=0
// Post: Retorna el prefix de mida n de l'avaluació de l'expressió representada
//       En cas que n sigui més gran que la mida d'aquesta avaluació,
//       llavors retorna només l'avaluació, cap caràcter més.
string evaluatePrefix(BinTree<string> t, int n);
```

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





Fixeu-vos que l'enunciat d'aquest exercici ja ofereix uns fitxers que haureu d'utilitzar per a compilar: `main.cc, BinTree.hh, evaluatePrefix.hh`. Us falta crear el fitxer `evaluatePrefix.cc` amb els corresponents `includes` i implementar-hi la funció anterior. Només cal que pugeu `evaluatePrefix.cc` al jutge.

Observació: Us recomanem fortament que comenceu mirant d'obtenir una solució lenta consistent en avaluar l'arbre i extraient-ne després el corresponent prefix, per tal d'obtenir una part de la nota, i que mireu d'optimitzar aquesta solució després.

Observacio: Els jocs de proves privats tenen strings petits als nodes (mida menys que 10), però els àrbres poden ser grans i els strings resultants d'avaluar aquests arbres poden ser grans. Hi ha alguns jocs de proves privats amb àrbres molt grans i n's molt petites.

Entrada

La primera línia de l'entrada descriu el format en el que es descriuen els arbres, o bé INLINEFORMAT o bé VISUALFORMAT. Després venen un nombre arbitrari de casos. Cada cas consisteix en una descripció d'un arbre binari que representa una expressió correcta, seguit d'un enter n en una nova linia. Fixeu-vos en que el programa que us oferim ja s'encarrega de llegir aquestes entrades. Només cal que implementeu la funció abans esmentada.

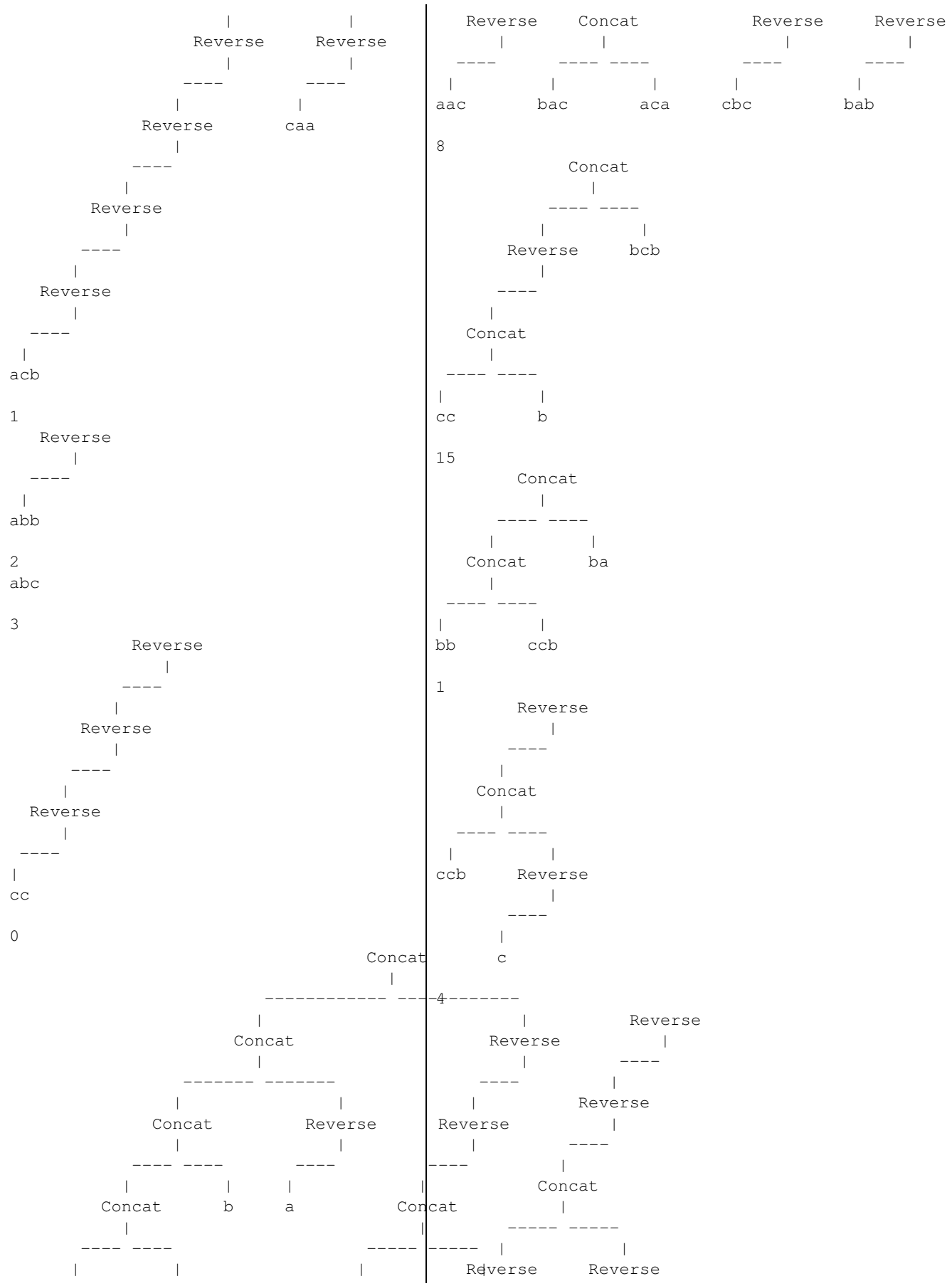
Sortida

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

Exemple d'entrada 1

```
VISUALFORMAT
    Concat
      |
  ---- ----
 |         |
bba        b

2
        Concat
          |
       ---- ----
      |         |
  Reverse    Concat
     |         |
 ----      ---- ----
|         |         |
bc       ccb       bc

6
```

```
abb

2
          Concat
            |
         ---- ----
        |         |
    Concat        c
      |
  ---- ----
 |         |
cac       Concat
            |
         ---- ----
        |         |
      acc         ba

6
                          Concat
                            |
                       ----- -----
```


Reverse
Reverse
Reverse
caa
Reverse
Reverse
acb
1
Reverse
abb
2
abc
3
Reverse
Reverse
Reverse
cc
0
Concat
Concat
Concat
Reverse
Concat
b
a
Reverse
Concat
Reverse
Reverse
aac
bac
aca
cbc
bab
8
Concat
Reverse
bcb
Concat
cc
b
15
Concat
Concat
ba
bb
ccb
1
Reverse
Concat
ccb
Reverse
c
4
Reverse
Reverse
Reverse
Concat
Reverse
Reverse

```
      |           |
  ----         ----
 |             |
cac           bbc

10
  Reverse
     |
 ----
|
ca

1
c

0
caa

0
cab

3
                  Concat
                    |
             ------- -------
            |               |
         Reverse          Reverse
            |               |
        ----             ----
       |                 |
    Concat           Reverse
       |                 |
  ---- ----         ----
 |          |      |
cba         b     cab

0
b

2
   Concat
     |
 ---- ----
|         |
ba        c

10
```

Exemple de sortida 1

```
bb
cbccbb
ab
cacacc
a
bb
abc

caabacac
bccbcb
b
cbcc
caccbb
a


cab

b
bac
```

Exemple d'entrada 2

```
INLINEFORMAT
Concat(bba,b)
2
Concat(Reverse(bc,),Concat(ccb,bc))
6
abb
2
Concat(Concat(cac,Concat(acc,ba)),c)
6
Concat(Reverse(Reverse(Reverse(Reverse(acb,),),),),Reverse(caa,))
1
Reverse(abb,)
2
abc
3
Reverse(Reverse(Reverse(cc,),),)
0
Concat(Concat(Concat(Concat(Reverse(aac,),Concat(bac,a
8
Concat(Reverse(Concat(cc,b),),bcb)
15
Concat(Concat(bb,ccb),ba)
1
```

```
Reverse(Concat(ccb,Reverse(c,)),)
4
Reverse(Reverse(Concat(Reverse(cac,),Reverse(bbc,)),),)
10
Reverse(ca,)
1
c
0
caa
0
cab
3
Concat(Reverse(Concat(cba,b),),Reverse(Reverse(cab,),))
0
b
2
Concat(ba,c)
10
```

Exemple de sortida 2

```
bb
cbccbb
ab
cacacc
a
bb
abc

caabacac
bccbcb
b
cbcc
caccbb
a


cab

b
bac
```

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres. Heu de trobar una solució **RECURSIVA** del problema, però podeu revessar strings iterativament si ho preferiu. Aquesta és la casuïstica de l'avaluació sobre 10 punts:

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

Entenem com a solució ràpida una que és correcta, de cost proporcional al mínim nombre de nodes que cal visitar per a calcular el corresponent prefix, 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

Autor : PRO2
Generació : 2023-12-19 19:52:27