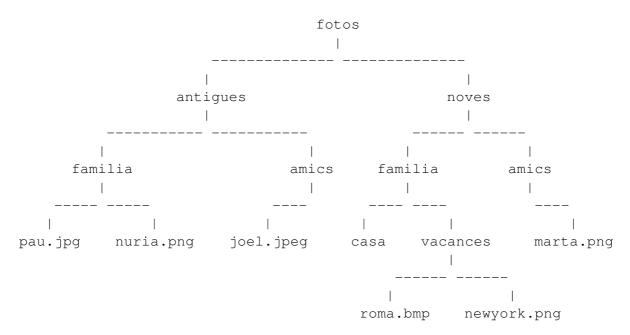
Esborra fitxers amb una extensió

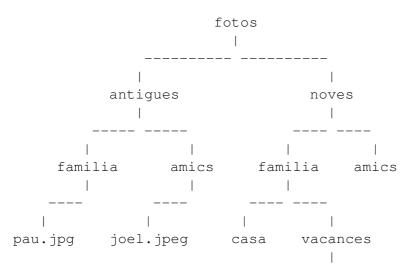
X24232_ca

En aquest exercici, considerarem arbres binaris d'strings que representen arbres de directoris i fitxers. Observeu el següent exemple:



En un arbre de directoris i fitxers hi ha dos tipus de nodes: directoris i fitxers. L'string d'un directori és no buit i està format per lletres minúscules. L'string d'un fitxer està format per una primera seqüència no buida de lletres minúscules, seguida del caràcter '.', seguit d'una segona seqüència no buida de lletres minúscules. Aquesta segona seqüència s'anomena la extensió del fitxer. Els fitxers han de ser necessàriament fulles de l'arbre. Els directoris poden ser nodes interns i també fulles.

Volem executar una comanda per a esborrar tots els fitxers que pengen del directori principal i que tenen una certa extensió. Per exemple, si volguessim esborrar tots els fitxers amb extensió png (La típica comanda rm -r *.png), l'arbre quedaria transformat així:





Heu d'implementar una funció RECURSIVA que rep un arbre de directoris i fitxers, i també rep un string que representa una extensió, i retorna un nou arbre de directoris i fitxers que és el resultat d'esborrar tots els fitxers amb aquella extensió de l'arbre rebut. Aquesta és la capcelera:

```
// Pre: Els nodes de 't' o bé son strings no buits de lletres minuscules, o bé
// son de la forma "s.e", on s i e son strings no buits de lletres minúsc
// En l'últim cas, el node ha de ser una fulla, i e s'anomena la extensió
// Post: Retorna l'arbre resultant d'esborrar de t tots els nodes amb strings d
// la forma "s.e" a on 'e' és igual a 'extension'.
BinTree<string> removeAll(const string &extension, const BinTree<string> &t);
```

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

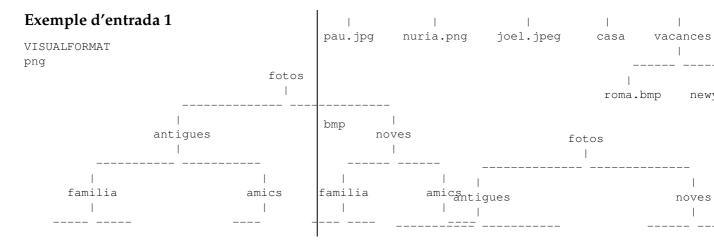
Observació: Donat un string s i un índex i, podeu utilitzar s.substr(i) per a obtenir el substring que és el sufix de s a partir de posició i.

Entrada

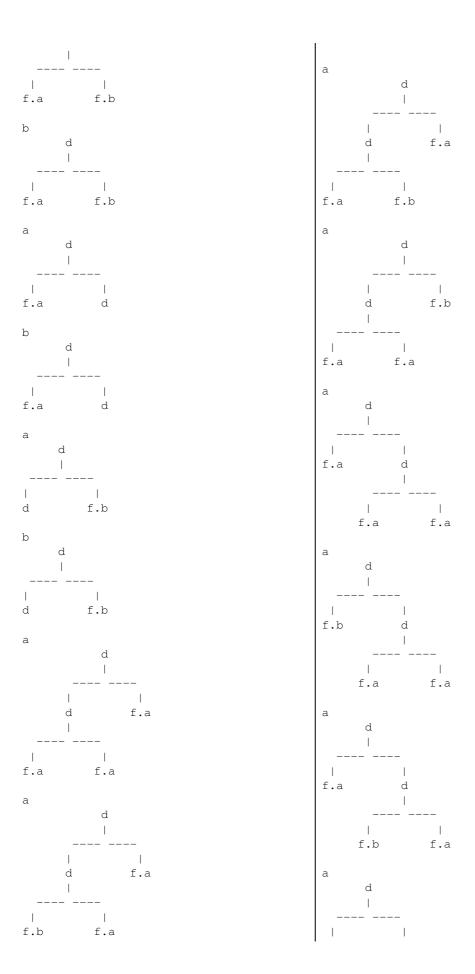
La primera linia de l'entrada descriu el format en el que es descriuen els arbres, o bé IN-LINEFORMAT o bé VISUALFORMAT. Després venen un nombre arbitrari de casos. Cada cas té una primera linia amb la extensió, un string no buit de lletre minúscules, i després una descripció d'un arbre binari que representa 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.

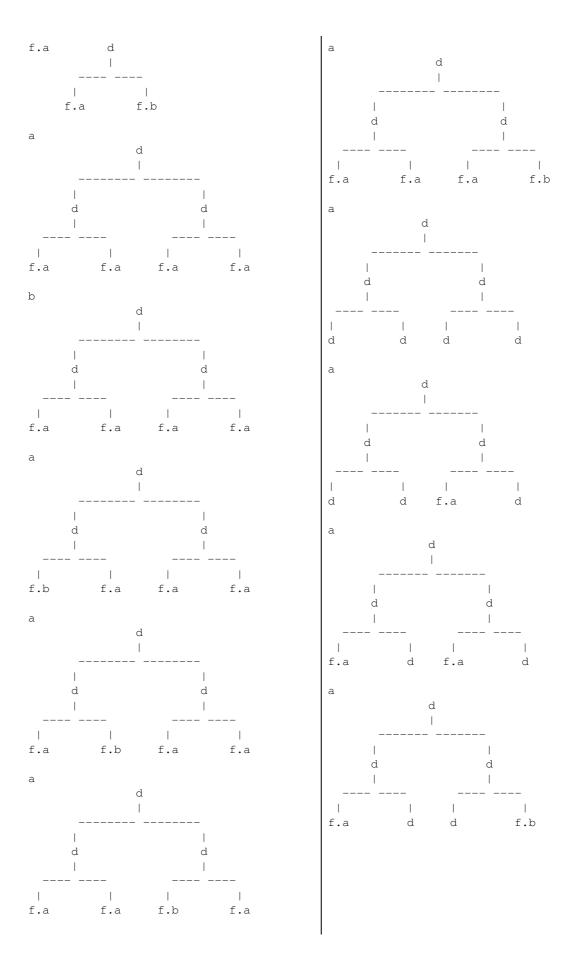
Sortida

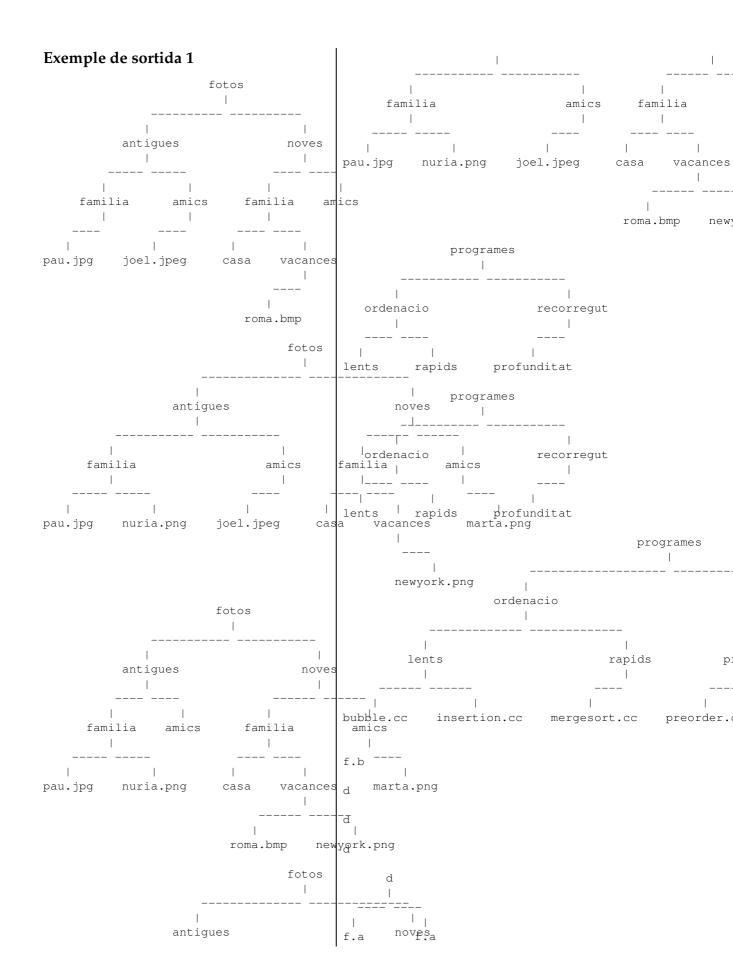
Per a cada cas, la sortida conté el corresponent arbre resultant. Fixeu-vos en que el programa que us oferim ja s'encarrega d'escriure aquesta sortida. Només cal que implementeu la funció abans esmentada.



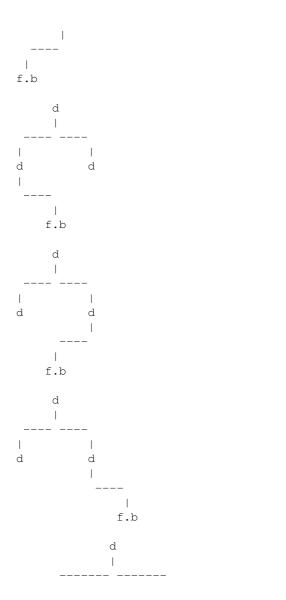
```
familia amics famibilia.cc iasmicsion.cc mergesort.cc preorder.
                                                                                                                                            | ordenacio
 jpeg
                                                                                                                                         lents
|
                                                                                                                                                                                                                                                        rapids
                                                                                                             fotos
                                                                                                               ·----
                                                                                                                                     bubble.coves insertion.cc mergesort.cc preorder.
                                                           antigues
programes
                                                                                                                               rapids p
                                                                                                         fotos bubble.cc insertion.cc mergesort.cc preorder.
 gif
                                                                                                                                  f.a | noves a | f.b-----
| d |
                                                                                                                                       1
 CC
                                                                                                                                      pfrogrames f.a
                                                                                                               d | recorregut | r
                                                                                ordenacio
                                                                     lents
```

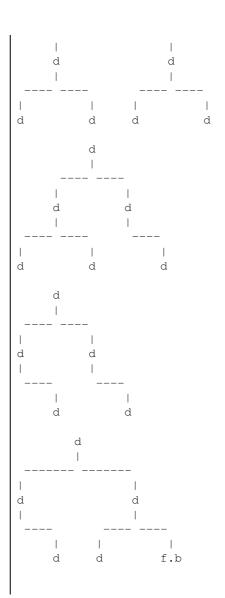






	, c 1
d I	f.b d
 f.b	
d I	d f.b
 I	I
f.a d	d d
 	d
d	f.b d
d 	d
f.a d	 d
d 	
d f.b	f.b
d I	d
 d	d
d I	 f.b
' 	d
d d	
 I	d d
d 	
 f.b	d d
d I	
 d	d I
•	





Exemple d'entrada 2

f.a

а f.b

```
INLINEFORMAT
png
```

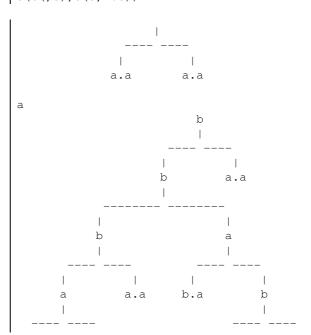
```
fotos (antigues (familia (pau. jpg, nuria.png), amics (foel. jpeg)), noves (familia (casa, vacances (roma.bmp
fotos (antigues (familia (pau.jpg, nuria.png), amics (fice).jpeg)), noves (familia (casa, vacances (roma.bmp
fotos (antigues (familia (pau.jpg, nuria.png), amics (foel.jpeg)), noves (familia (casa, vacances (roma.bmp
fotos (antigues (familia (pau. jpg, nuria.png), amics (fool. jpeg)), noves (familia (casa, vacances (roma.bmp
programes (ordenacio (lents (bubble.cc, insertion.cc), rapids (mergesort.cc)), recorregut (profunditat (p
programes (ordenacio (lents (bubble.cc, insertiph.cc), rapids (mergesort.cc)), recorregut (profunditat (p
programes (ordenacio (lents (bubble.cc, insertion.ca), rapids (mergesort.cc)), recorregut (profunditat (p
                                              d(d, f.b)
                                              d(d(f.a,f.a),f.a)
```

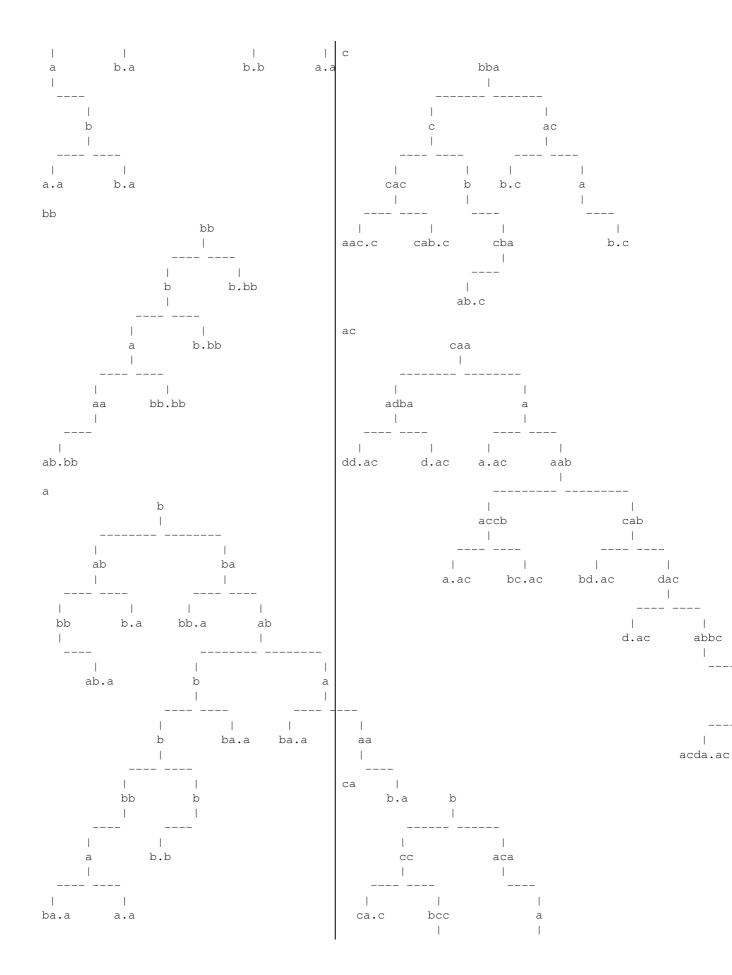
```
d(d(f.b,f.a),f.a)
d(d(f.a,f.b),f.a)
d(d(f.a,f.a),f.b)
d(f.a,d(f.a,f.a))
d(f.b,d(f.a,f.a))
d(f.a,d(f.b,f.a))
d(f.a,d(f.a,f.b))
d(d(f.a,f.a),d(f.a,f.a))
d(d(f.a,f.a),d(f.a,f.a))
d(d(f.b,f.a),d(f.a,f.a))
d(d(f.a,f.b),d(f.a,f.a))
d(d(f.a, f.a), d(f.b, f.a))
d(d(f.a, f.a), d(f.a, f.b))
d(d(d,d),d(d,d))
d(d(d,d),d(f.a,d))
d(d(f.a,d),d(f.a,d))
d(d(f.a,d),d(d,f.b))
```

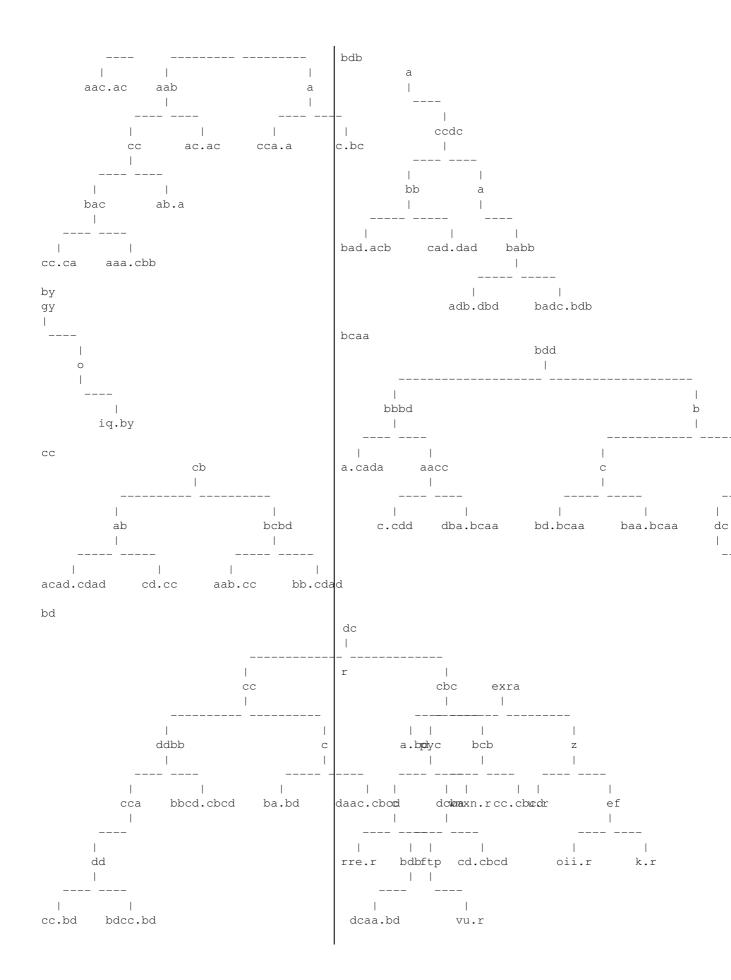
Exemple d'entrada 3

Exemple de sortida 2

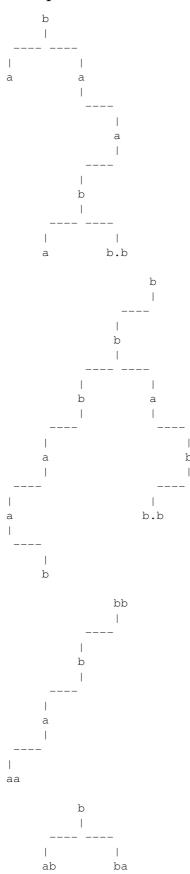
```
fotos (antigues (familia (pau.jpg,), amics (joel.jpeg,)), no
fotos (antigues (familia (pau. jpg, nuria.png), amics (joel. j
fotos (antigues (familia (pau. jpg, nuria.png), amics), noves
fotos (antigues (familia (pau. jpg, nuria.png), amics (joel. j
programes (ordenacio(lents, rapids), recorregut (profundit
programes (ordenacio(lents, rapids), recorregut (profundit
programes (ordenacio (lents (bubble.cc, insertion.cc), rapi
()
f.b
d
d
d
d(f.a, f.a)
d(,f.b)
d(f.a,)
d(,d)
d(f.a,d)
d(d, f.b)
d(d,)
d(d,)
d(d(f.b,),)
d(d(,f.b),)
d(d, f.b)
d(,d)
d(f.b,d)
d(,d(f.b,))
d(,d(,f.b))
d(d,d)
d(d(f.a,f.a),d(f.a,f.a))
d(d(f.b,),d)
d(d(,f.b),d)
d(d,d(f.b,))
d(d,d(,f.b))
d(d(d,d),d(d,d))
d(d(d,d),d(,d))
d(d(,d),d(,d))
d(d(,d),d(d,f.b))
```

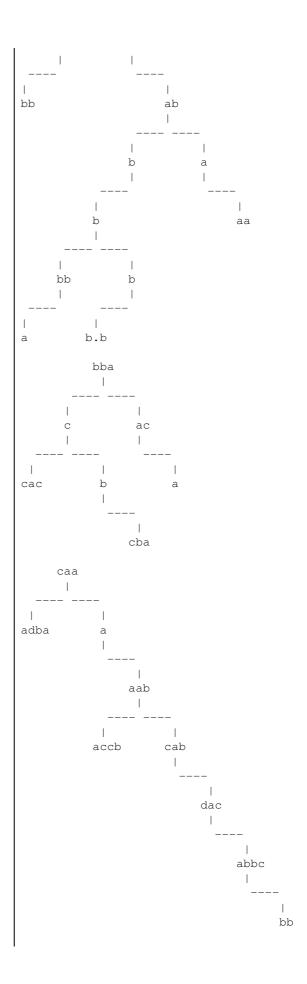


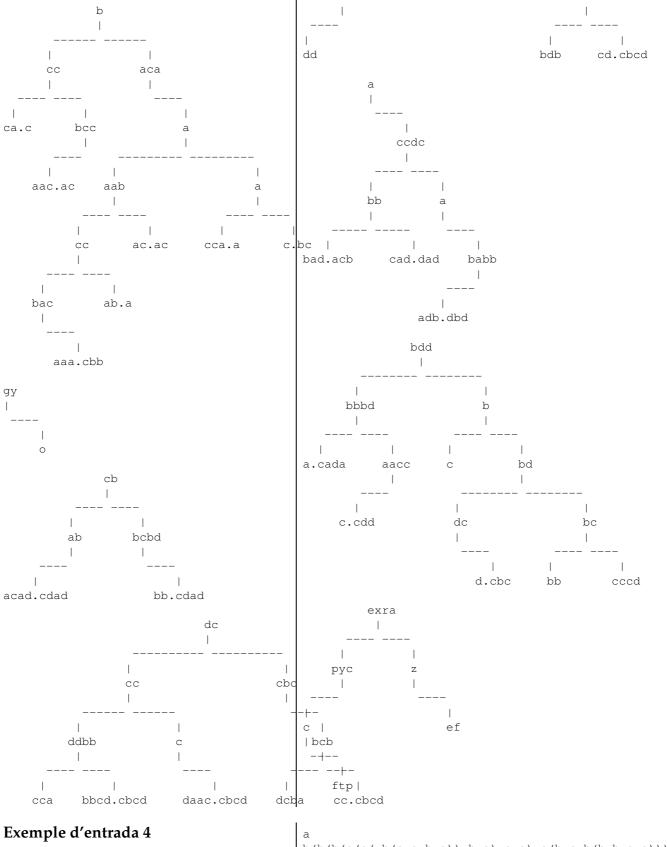




Exemple de sortida 3







```
b(b(b(a(a(,b(a.a,b.a)),b.a),a.a),a(b.a,b(b.b,a.a))),a.
INLINEFORMAT
bb(b(a(aa(ab.bb,),bbb),b.bb),b.bb)
b(a(b.a,a.a),a(b.a,a(b(a(a.a,a.a),b.b),a.a)))
```

```
Exemple de sortida 4
b(ab(bb(,ab.a),b.a),ba(bb.a,ab(b(b(bb(a(ba.a,a.a),),b(b.b,)),ba.a),a(ba.a,aa(,b.a)))))
b(ab(bb(,ab.a),b.a),ba(bb.a,ab(b(b(bb(a(ba.a,a.a),),b(b.b,)),ba.a),a(ba.a,aa(,b.a)))))
bba(c(cac(aac.c,cab.c),b(,cba(ab.c,))),ad (b(b(b(a(a.c,b))),a(,b(b.b,))),)
                                        bb(b(a(aa,),),)
bba(c(cac,b(,cba)),ac(,a))
b(cc(ca.c,bcc(aac.ac,)),aca(,a(aab(cc(bac(ca.c,bcc(aac.ac,)),aca(,a(aab(cc(bac(ca.c,bcc(aac.ac,)),aca(,a(aab(cc(bac(ca.c,bcc(aac.ac,))),aca(,a(aab(cc(bac(,aaa.cbb),ab
                                        gy(,o)
gy(,o(,iq.by))
                                        cb(ab(acad.cdad,),bcbd(,bb.cdad))
cb(ab(acad.cdad,cd.cc),bcbd(aab.cc,bb.cdad)(cc(ddbb(cca(dd,),bbcd.cbcd),c(,daac.cbcd)),cbc(,bcb
                                        a(,ccdc(bb(bad.acb,cad.dad),a(,babb(adb.dbd,))))
exra(pyc(c(,ftp),),z(,ef))
a(,ccdc(bb(bad.acb,cad.dad),a(,babb(adb.dbd,badc.bdb))))
bdd(bbbd(a.cada,aacc(c.cdd,dba.bcaa)),b(d(bd.bcaa,baa.bcaa),bd(dc(,d.cbc),bc(bb(c.bcaa,),cccd(dd
exra(pyc(c(rre.r,ftp(,vu.r)),wmxn.r),z(u.r,ef(oii.r,k.r)))
```

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres. Heu de trobar una solució **RECURSIVA** del problema. Avaluació sobre 10 punts:

• Solució lenta: 5 punts.

• solució ràpida: 10 punts.

Informació del problema

Autor: PRO2

Generació: 2023-10-21 13:54:01

© *Jutge.org*, 2006–2023. https://jutge.org