

---

## Dibuixar imatges en una pissarra (en mode caràcter) X45762\_ca

---

En aquest exercici us donem un programa a mig fer que haureu de completar. El programa treballa amb imatges que voldrem dibuixar sobre una pissarra.

Més concretament, tindrem un tipus de dada `Image`, que té un nom, una profunditat, una ubicació representada per dos naturals  $i, j$  i un vector de strings  $v$ . Usualment anomenarem `image` a les variables de tipus `Image`.

Una pissarra serà un vector de strings que usualment anomenarem `board`.

Diem que una `image` és vàlida si el seu  $v$  és una matriu rectangular d'unes certes dimensions  $n \times m$  no nul·les.

Diem que un `board` és vàlid si és una matriu rectangular d'unes certes dimensions  $N \times M$  no nul·les.

A més a més, diem que aquesta `image` encaixa dins d'aquest `board` si  $i + n \leq N$  i  $j + m \leq M$ , on  $i, j$  és la ubicació d'aquesta `image`.

El resultat de dibuixar aquesta `image` sobre aquest `board` consisteix en modificar `board` de manera que, per a cada posició  $i', j'$  de `image` amb un caràcter diferent de ' . ' , es compleixi `board[i + i'][j + j'] == image[i'][j']`. Cap altre caràcter de `board` haurà canviat.

La funció `main`, que ja us donem feta, llegeix una llista d'imatges, les ordena de major a menor profunditat, i les dibuixa sobre una pissarra en aquest ordre.

Haureu d'implementar una funció per a llegir una imatge de l'entrada, una per a calcular les dimensions mínimes d'una pissarra que fan que totes les imatges d'una llista hi encaixin, i una funció per a dibuixar una imatge sobre una pissarra. A més a més, haureu d'implementar una funció que compara imatges i es fa servir per a ordenar-les. Una imatge és "menor que" una altra si té major profunditat que l'altra, o té la mateixa profunditat que l'altra però el seu nom és menor que el nom de l'altra en ordre lexicogràfic.

Completeu el següent codi a mig per per tal de solucionar l'exercici:

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>

// Here you can add more includes if you wish.
// ...

using namespace std;

struct Image {
    string name;
    int depth;
    int i, j;
    vector<string> v;
};

typedef vector<Image> ListImages;

// Auxiliary functions (you can add more functions if you wish)
```

```

// Pre: The input has a description of an image with this format:
//      - First line: name depth i j n m
//      - n lines with m characters each (the contents of v)
//      These characters are different from whitespace, as we use '.' to represent
Image readImage()
{
// Implement this function.
//...
}

// Pre: listimages contains a non-empty list of valid images.
// Post: N,M are the dimensions of the minimum board such that
//      all of those images fit in it.
//      In other words, N,M must be the minimum naturals satisfying that,
//      for each image in listimages,
//      if i,j are its location and n,m are the dimensions of its v,
//      then  $i+n \leq N$  and  $j+m \leq M$  must be satisfied,
void computeMinimumBoardDimensions(const ListImages &listimages, int &N, int &M)
// Implement this function.
//...
}

// Pre: image is valid and board is valid and image fits in board.
// Post: image has been drawn on board. Nothing else has changed.
//      Recall that occurrences of character '.' in image are not printed on board
void drawImage(const Image &image, vector<string> &board)
{
// Implement this function.
//...
}

// Pre: image1, image2 represent valid images.
// Post: Returns true iff one of the following conditions holds:
//      - depth of image1 is strictly bigger than depth of image2.
//      - image1 and image2 have same depth, but image1 has smaller name than image2
bool compareImages(Image image1, Image image2)
{
// Implement this function.
//...
}

// Pre: listimages has a list of valid images.
// Post: prints on the output the result of drawing all of those
//      images on the minimum board such that all of them fit in,
//      and sorted by depth and name.
void drawListImages(const ListImages &listimages)
{
sort(listimages.begin(), listimages.end(), compareImages);
}

```





.  
n  
n  
name18 22 16 3 4 5  
www.  
.www.  
.www  
w.www  
name19 19 10 1 3 4  
h.hh  
hhh  
h.hh  
name20 17 3 4 5 5  
.nn.n  
n.nnn  
nn.nn  
nn...  
nn.nn  
name21 13 14 10 4 1  
t  
t  
t  
t  
name22 3 11 1 5 5  
l...l  
lll.l  
lllll  
lllll  
lllll  
name23 20 4 6 5 3  
mmm  
mmm  
mm.  
mmm  
mm.  
name24 6 1 10 4 3  
ppp  
ppp  
.pp  
.pp  
name25 5 10 1 2 1  
u  
u  
name26 25 16 2 1 4  
oooo  
name27 4 14 12 4 4  
hhh.  
hhh.  
hhhh  
h.hh  
name28 9 16 11 3 1  
p  
p  
p  
name29 20 4 7 1 4  
u.uu  
name30 8 0 9 3 2  
bb  
bb  
..  
name31 20 6 6 3 3

YYY  
Y.Y  
YY.  
name32 16 0 8 2 5  
YY.Y.  
Y.YYY  
name33 9 6 17 3 2  
.e  
e.  
ee  
name34 18 0 11 1 2  
tt  
name35 18 16 1 1 5  
.kk.k  
name36 11 12 13 4 1  
q  
q  
q  
q  
name37 19 2 9 4 3  
.bb  
b.b  
bbb  
bbb  
name38 16 15 5 5 3  
bbb  
bbb  
b.b  
bb.  
bbb  
name39 9 9 16 3 2  
.t  
.t  
.t  
name40 6 5 9 2 3  
ww.  
ww.  
name41 4 15 13 3 5  
YYYYY  
YYYYY  
.Y.Y.  
name42 3 7 13 5 5  
u..uu  
uuu.u  
uuu.u  
uuuuu  
uuuuu  
name43 1 2 19 3 1  
m  
m  
m  
name44 6 11 1 1 3  
tt.  
name45 13 17 6 1 4  
..p.  
name46 20 0 16 2 2  
zz  
z.  
name47 10 2 8 4 4  
nn.n  
nnnn

n..n  
..nn  
name48 4 7 1 3 2  
ww  
ww  
.w  
name49 4 0 7 4 1  
.  
o  
o  
o  
name50 22 12 11 3 5  
mmmmmm  
.mmmm  
mmmmmm  
name51 24 3 13 3 5  
.hhhh  
hhhhh  
hhhhh  
name52 4 6 8 5 4  
ff.f  
ffff  
.fff  
f.ff  
ffff  
name53 3 14 7 2 3  
vvv  
v..  
name54 13 10 11 1 5  
bbbbbb  
name55 6 11 9 4 3  
f.f  
fff  
fff  
ff.  
name56 17 9 2 4 1  
n  
n  
n  
n  
name57 19 8 15 3 3  
g.g  
g.g  
...  
name58 5 4 8 5 2  
.u  
u.  
.u  
.u  
uu  
name59 2 6 12 3 2  
..  
.s  
ss  
name60 12 13 17 4 1  
h  
h  
h  
h  
name61 9 19 5 1 4  
.YY.

name62 0 3 11 5 3  
ppp  
ppp  
.pp  
p.p  
.pp  
name63 10 14 5 4 4  
.jj.  
..jj  
jjjj  
jjjj  
name64 13 13 14 5 1  
j  
j  
j  
j  
j  
name65 14 17 15 3 3  
Y.Y  
YY.  
YYY  
name66 23 16 7 3 5  
cc.c.  
c.ccc  
cccc.  
name67 2 15 3 4 3  
ee.  
eee  
eee  
.ee  
name68 22 1 10 5 5  
uuuu.  
uuu.u  
uuuu.  
u.uuu  
uu..u  
name69 2 2 3 2 1  
a  
a  
name70 15 1 1 1 1  
g  
name71 8 15 12 1 5  
mmmmmm  
name72 15 15 16 4 2  
vv  
vv  
v.  
vv  
name73 7 17 0 3 3  
qqq  
qqq  
qqq  
name74 12 7 3 4 3  
bbb  
bbb  
b.b  
.b.  
name75 14 2 0 5 5  
qq.qq  
..qqq  
.qqqq

qq...  
.qqq.  
name76 23 5 4 3 5  
.pppp  
pppp.  
p.ppp  
name77 7 11 15 5 5  
ll.l.  
l.lll  
lllll  
lllll  
lllll  
name78 8 14 11 4 3  
iii  
i..  
iii  
.ii  
name79 9 12 5 3 5  
qqqqq  
qqqq.  
qqqqq  
name80 15 2 7 1 1  
v  
name81 7 5 2 2 4  
eeee  
eee.  
name82 6 14 11 2 5  
rrrrr  
rr.rr  
name83 15 2 11 1 1  
p  
name84 25 3 0 2 2  
.s  
ss  
name85 9 6 4 3 2  
mm  
mm  
mm  
name86 22 11 1 5 4  
oooo  
.ooo  
oooo  
oooo  
...o  
name87 0 8 13 2 5  
c.ccc  
cccc.  
name88 6 9 0 2 5  
.cccc  
c..cc  
name89 2 1 15 5 1  
v  
v  
v  
v  
.  
name90 0 1 3 1 5  
ggg.g  
name91 25 13 14 1 5  
mmmm.  
name92 13 2 2 5 1

l  
l  
l  
l  
l  
name93 7 18 13 2 5  
uuuuu  
uuuuu  
name94 10 11 8 4 5  
yyyyy  
...yy  
yy.yy  
yyyyy  
name95 16 8 10 2 2  
.x  
xx  
name96 21 7 14 1 2  
ll  
name97 10 13 15 5 4  
mmm.  
.mm.  
m.mm  
mmmm  
mmm.  
name98 17 10 12 1 2  
kk  
name99 11 10 11 3 5  
fffff  
f.fff  
f.fff

## Exemple de sortida 2

```
..mm..m.ybbyt...zz..
.gmggmggybpppppvz..i
qqmammonnppppuv...m
.slambmonnnppppvhh.m
sqlwwbnubppppvhh.m
qqewewwuwwpppwjh.i
.qeeemyyffwkpkk..je.
.wwbmynfffkppwluu..
.wwbmyyuffkkcucce.
.cwcbbbf.fkkcccu..
```

```
cuncqqbbfffkkuu..
.lthqqbbyfyfyuuuul.
.111qqqqfffyfqq.111
.111qlqqqfffyqq1111
.111qqqvvrhqq1111
.lleelbvj.trhyqqyyl
..keeejjltihyyym.
qqqeeejjctphiyhm..
qqq.eebwcccp.uuuu..
qqqw.by.....uuuu..
```

## Informació del problema

Autor : PRO1

Generació : 2024-01-02 20:19:05

© *Jutge.org*, 2006–2024.

<https://jutge.org>