

Camí més llarg en un arbre binari

V22704_ca

Implementeu una funció *recursiva* que, donat un arbre binari d'enters, retorni la llista de valors que es troben des de l'arrel seguint el camí més llarg de l'arbre. En cas que hi hagi varis camins màxims, s'haurà d'escollir el camí que va el més a l'esquerra possible. Aquesta és la capçalera:

```
/**
 * @brief Retorna els valors del camí més llarg d'un arbre binari.
 *
 * Un camí va des de l'arrel de l'arbre fins a una fulla (un node sense fills).
 * Si hi ha més d'un camí màxim, cal retornar el que va per les branques de més
 * a l'esquerra possible.
 *
 * @param t L'arbre binari.
 * @returns Els valors dels nodes del camí més llarg de `t`.
 */
vector<int> longest_leftmost_path(BinTree<int> t);
```

Un exemple d'arbre i la seva sortida corresponent seria:

```
longest_leftmost_path( 3          ) => [3,3,2,1]
      |-- 1
      |   |-- #
      |   '--- 5
      '--- 3
           |-- 2
           |   |-- 1
           |   '--- 7
           '--- #
```

	bintree.hh	la classe BinTree
	bintree-io.hh	l'entrada/sortida de BinTree
Els fitxers públics (icona del gatet) contenen:	bintree-inline.hh	l'e/s en format "inline"
	vector-io.hh	la funció print_vector
	main.cc	el programa principal

També hi ha un Makefile i el directori .vscode que té la configuració per compilar i debuggar amb VSCode.

Cal implementar `longest_leftmost_path` en un **fitxer .cc nou**, compilar (està preparat per poder compilar i debuggar amb VSCode), i finalment **enviar només el fitxer amb la funció**.

Entrada

La primera línia de l'entrada descriu el format en el que es descriuen els arbres, "inline" o "visual". Després venen un nombre arbitrari de casos. Cada cas consisteix en la descripció d'un arbre binari d'enters.

Sortida

Per a cada cas, la sortida conté el corresponent camí més llarg i més a l'esquerra de l'arbre, en un format del que ja s'encarrega el programa principal.

Observació

La vostra funció i subfuncions que creeu han de treballar només amb arbres. Heu de trobar una solució *recursiva* del problema.

Us aconsellem fer aquest problema en dues fases: 1) fer una solució correcta malgrat no sigui eficient; i 2) buscar una solució més eficient utilitzant immersió.

Exemple d'entrada

visual

7

|-- 1

'-- 2

 |-- 3

 |-- 5

 '-- 4

 '-- 5

6

|-- 7

| |-- 8

| '-- 7

'-- 8

 |-- 4

 '-- 6

2

|-- 2

| |-- 7

| | |-- #

| | '-- 2

| | |-- #

| | '-- 7

| '-- 8

'-- 4

 |-- #

 '-- 7

 |-- 3

 '-- 5

3

|-- 3

| |-- 4

| '-- 5

'-- 7

 |-- 1

 '-- 5

7

|-- 4

'-- 3

6

|-- 5

| |-- 2

| '-- 7

'-- #

2

4

|-- #

'-- 6

 |-- 3

 '-- 1

4

|-- 8

| |-- 4

| | |-- #

| | '-- 7

| '-- 8

| |-- 5

| '-- 1

'-- #

4

Exemple de sortida

[7, 2, 3, 5]
[6, 7, 8]
[2, 2, 7, 2, 7]
[3, 3, 4]

[7, 4]
[6, 5, 2]
[2]
[4, 6, 3]
[4, 8, 4, 7]
[4]

Informació del problema

Autoria: Pau Fernández

Generació: 2026-04-02T17:11:15.584Z

© *Jutge.org*, 2006–2026.

<https://jutge.org>