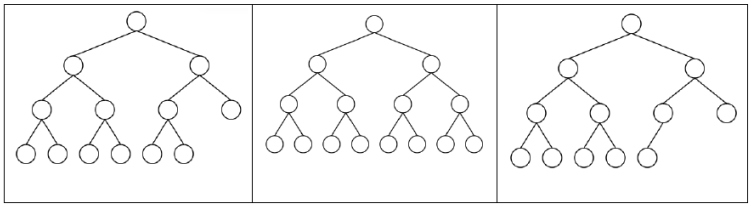
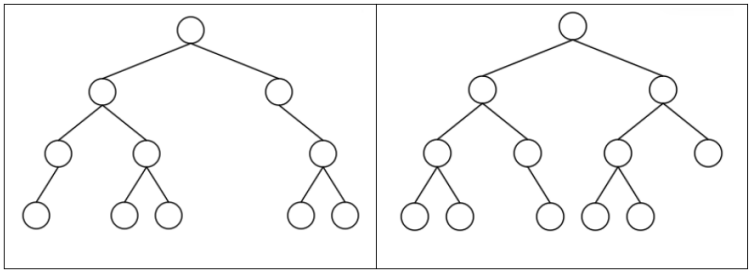
L'Arbre Complet U88568_ca

Un arbre binari *complet* és un arbre binari en què tots els nivells, excepte potser l'últim, estan completament omplerts i tots els nodes estan tan a l'esquerra com sigui possible (mireu el *pdf* d'aquest enunciat si no us surt la imatge amb els exemples).

Exemples d'arbres complets:



Exemples d'arbres incomplets:



Us demanem que feu una funció **`complet(a)`**, que, donada una instància de la classe `Arbre Binari` *a*, retorni `True` si *a* és complet, i `False` si *a* no és complet, i l'afegiu al fitxer **`code.py`** tal com s'especifica a les Observacions.

Paràmetres i retorn

El paràmetre de la funció **`complet(a)`** és una instància de la classe `ArbreBinari` (que ja coneixem).
La funció retorna un *booleà*: `True` si *a* és complet, `False` si no ho és.

Entrada

L'entrada al programa serà el preordre de l'arbre binari, amb marca "-1" (atenció, és una *string*) per indicar els arbres buits (ja coneixem aquest format dels exercicis a classe de laboratori, aquí, però, treballem amb *strings* perquè el contingut dels nodes de l'arbre no ens importa).
Vegeu els exemples del joc de proves públic.

Sortida

El programa ha d'escriure un *booleà*: `True` si *a* és complet, `False` si no ho és.
Vegeu els exemples del joc de proves públic.

Observacions

Heu de baixar-vos el fitxer **`code.py`** (icona de la serp). Aquest fitxer és un programa amb **tot** el que cal per executar els jocs de prova públics. Només falta, clar, la funció que us demana l'enunciat. Aquest fitxer l'heu de completar amb el codi que falta, i això, **tot**, és el que heu d'enviar al Jutge com a solució.

Dins el fitxer **`code.py`** teniu la classe **`ArbreBinari`** que hem treballat a les classes de teoria. No caldrà que la vostra solució faci cap *import* ni res. Tot el codi que us cal el teniu dins de **`code.py`**.

Pista: Aquest problema es pot resoldre d'una manera similar al recorregut per nivells.

L'eficiència i la qualitat de la solució es tindran en compte a la correcció manual.

Exemple d'entrada 1

```
0 -1 0 -1 -1
```

Exemple de sortida 1

```
False
```

Exemple d'entrada 2

```
0 0 -1 -1 0 -1 -1
```

Exemple de sortida 2

```
True
```

Exemple d'entrada 3

```
0 0 0 -1 -1 0 -1 -1 0 0 -1 -1 0 -1 -1
```

Exemple de sortida 3

```
True
```

Exemple d'entrada 4

```
0 0 0 -1 -1 0 -1 -1 0 -1 -1
```

Exemple de sortida 4

```
True
```

Exemple d'entrada 5

```
0 0 -1 -1 -1
```

Exemple de sortida 5

```
True
```

Exemple d'entrada 6

```
0 -1 0 -1 0 0 -1 -1 0 -1 -1
```

Exemple de sortida 6

```
False
```

Exemple d'entrada 7

```
0 0 -1 -1 0 0 -1 -1 -1
```

Exemple de sortida 7

```
False
```

Exemple d'entrada 8

```
-1
```

Exemple de sortida 8

```
True
```

Exemple d'entrada 9

```
0 -1 -1
```

Exemple de sortida 9

```
True
```

Informació del problema

Autor : Jordi Delgado
Generació : 2025-06-15 09:15:24