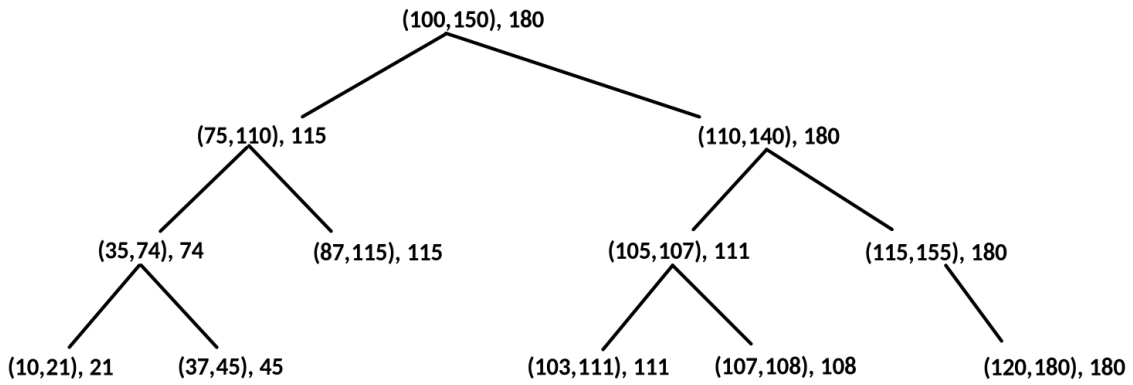
Solapament d'intervals X23211_ca

Definim un interval **(inici, fi)** com una tupla de dos elements on **inici** $\leq$ **fi**. Volem ser capaços de detectar si un interval donat es solapa amb algun interval d'una determinada col·lecció d'intervals que tenim guardats d'alguna manera (que, naturalment, es poden solapar entre ells).
Farem servir BSTs (*Binary Search Trees*) lleugerament modificats. Cada node de l'arbre és un interval **(inici, fi)** (i més informació que descrivim més avall). És important entendre que el BST organitza els intervals pel seu temps d'inici, es a dir, per mantenir la propietat BST compara els intervals directament, on **(x0,y0) < (x1,y1)** si i només si **x0 < x1 or (x0 == x1 and y0 < y1)** (és com ho fa Python per defecte). Així, els intervals guardats en els nodes es poden solapar.
Cada node, a més de guardar un interval **(inici, fi)**, també guarda el valor màxim de **fi** per a cada interval en el subarbre del que el node és arrel. Aquest valor màxim a cada node el guardem ja que ens facilita força la implementació del mètode que demanem més endavant.
Farem servir una subclasse **BST_intervals** de la classe **BST**, amb les modificacions que cal per gestionar el que s'ha descrit més amunt (vegeu el fitxer **code.py**).
Es demana: Afegir un mètode a la classe **BST_intervals**, anomenat **hi_ha_solapament(interval)**, tal que, donada una instància de **BST_intervals** (anomenem-la **a**), i un **interval**, retorni **True** si l'**interval** es solapa amb algún dels intervals emmagatzemats en **a**, i **False** en altre cas.
Per exemple:
Suposem que tenim una instància de **BST_intervals** anomenada **a** i que representem gràficament així:



Recordem que a cada node tenim: un interval **(fita inferior, fita superior)** i el màxim de les fites superiors de cada interval del subarbre que té aquest node com a arrel.
Aleshores, **a.hi_ha_solapament((22,30))** hauria de retornar **False**, i **a.hi_ha_solapament((109,** hauria de retornar **True**.

Paràmetres i retorn del mètode demanat

El paràmetre del mètode **hi_ha_solapament(interval)** és un interval. Retorna un booleà.
Entrada
L'entrada al programa és un nombre n, que indica quants intervals defineixen la instància de **BST_intervals** amb la que treballarem. Tot seguit trobem els intervals, en el format

$fita\_inf, fita\_sup, fita\_inf, fita\_sup, \ldots$ (per tant tenim $2n$ nombres). Després se'ns proporciona un nombre m, que indica quants intervals volem provar amb la instància de **`BST_intervals`** definida anteriorment. Tenim, doncs, $2m$ nombres descrivint els intervals en el mateix format que hem vist abans.
Per exemple, per dur a terme el cas que hem vist més amunt, el fitxer d'entrada seria:

```
12
100 150 75 110 110 140 35 74 87 115 105 107 115 155
10 21 37 45 103 111 107 108 120 180
2
22 30 109 110
```

Vegeu els exemples del joc de proves públic.
Sortida
El programa ha d'escriure **`True`** o **`False`**, per a cada interval provat.
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, el mètode 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 **`BST`** i la subclasse **`BST_intervals`** amb tot el descrit a l'enunciat, menys el mètode demanat. No cal que la vostra solució faci cap *import* ni res. Tot el codi que us cal el teniu dins de **`code.py`**.

Exemple d'entrada 1

```
12
100 150 75 110 110 140 35 74 87 115 105
107 115 155 10 21 37 45 103 111 107 108
120 180
5
5 9 22 30 75 80 90 105 109 110
```

Exemple de sortida 1

```
False
False
True
True
True
```

Exemple d'entrada 2

```
1
100 200
7
10 20 30 100 50 110 110 150 160 210 200 215 224 300
```

Exemple de sortida 2

```
False
False
True
True
True
False
False
```

Exemple d'entrada 3

```
8
100 150 75 110 110 140 35 74 87 115 115 155
10 21 107 108
3
22 30 75 80 90 105
```

Exemple de sortida 3

```
False
True
True
```

Informació del problema

Autoria: Jordi Delgado

Generació: 2026-05-12T18:25:46.271Z