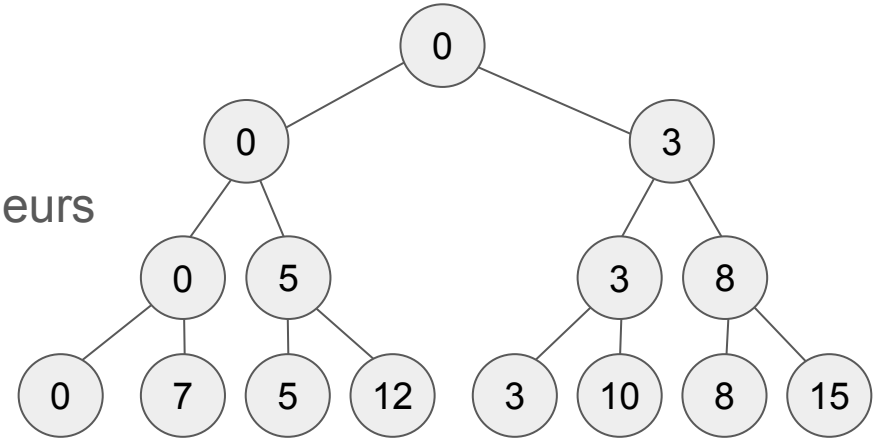
Recherche en profondeur | **Exemple pas à pas**

Voici l'arbre complet, si on devait le construire.

Gardons en tête que, précisément, nous ne possédons pas les valeurs des noeuds !

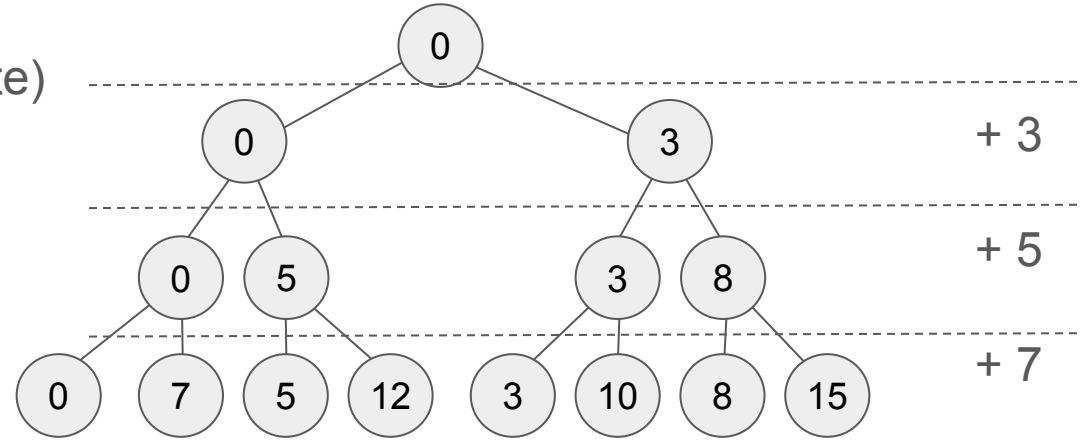
Dans cet exemple, E = [3, 5, 7]



Recherche en profondeur | **Exemple pas à pas**

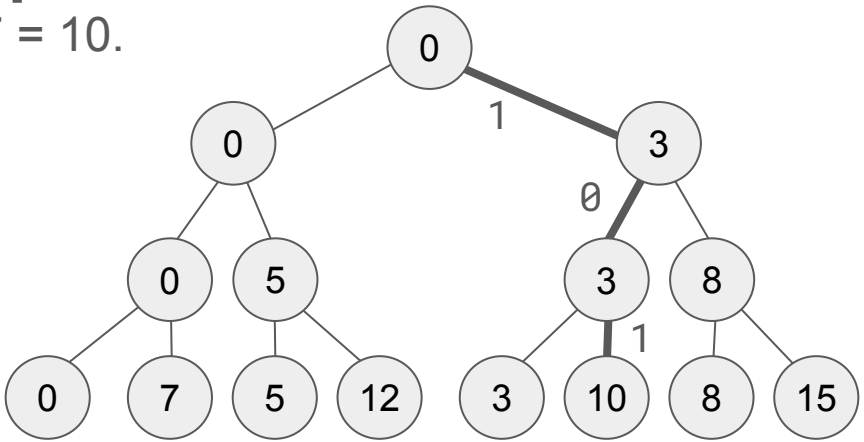
Chaque étage de l'arbre est associé avec un élément donné de E, que l'on prend (à droite) ou pas (à gauche).

Dans cet exemple, E = [3, 5, 7]



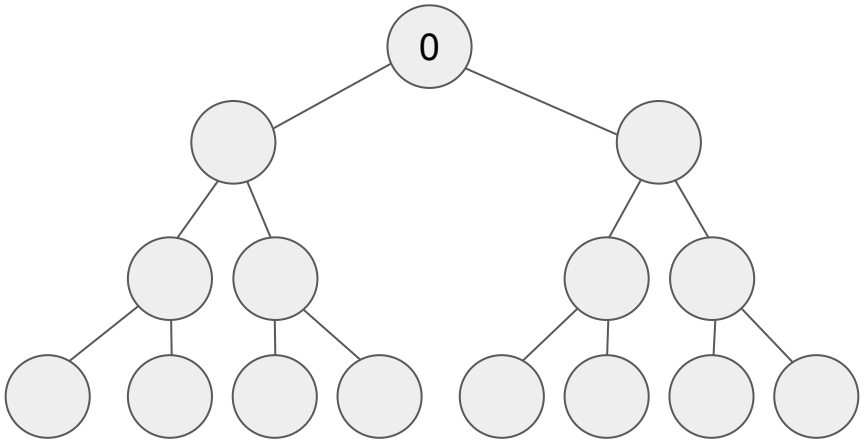
Recherche en profondeur | **Exemple pas à pas**

Par exemple, ce chemin
correspond à u = [1,0,1]
et donc à S = 3 + 0 + 7 = 10.



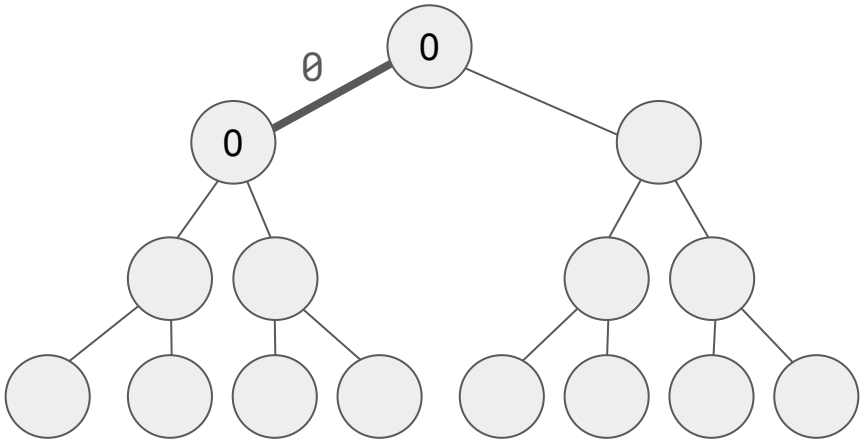
Recherche en profondeur | **Exemple pas à pas**

Nous voulons trouver un sous-ensemble qui génère S = 12.



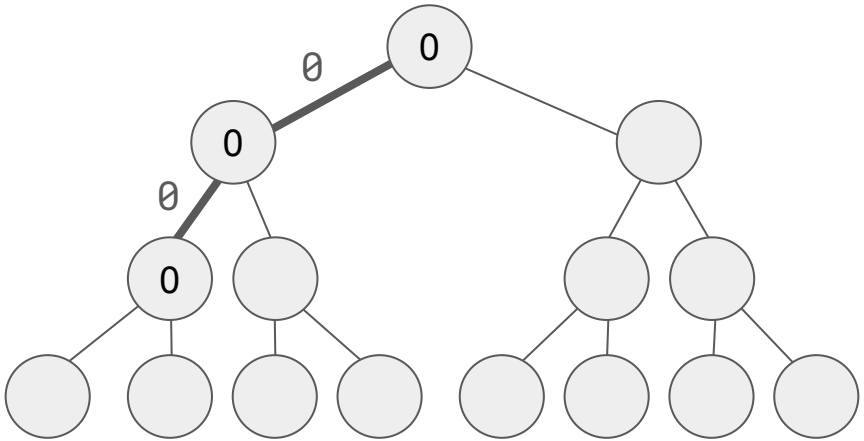
On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**



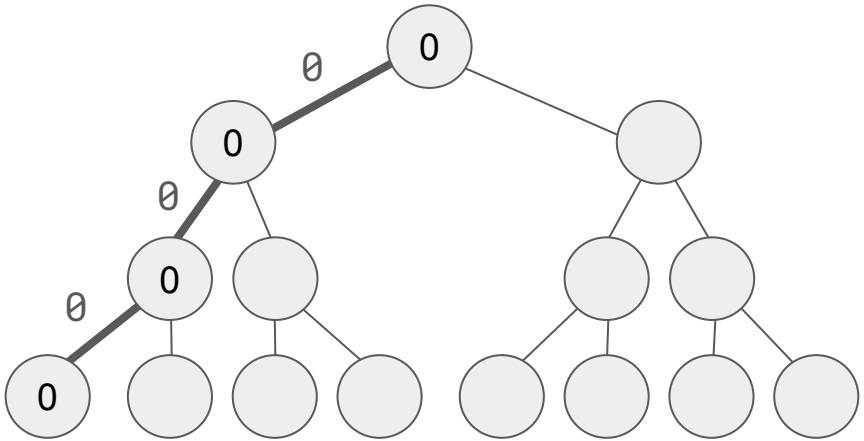
On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

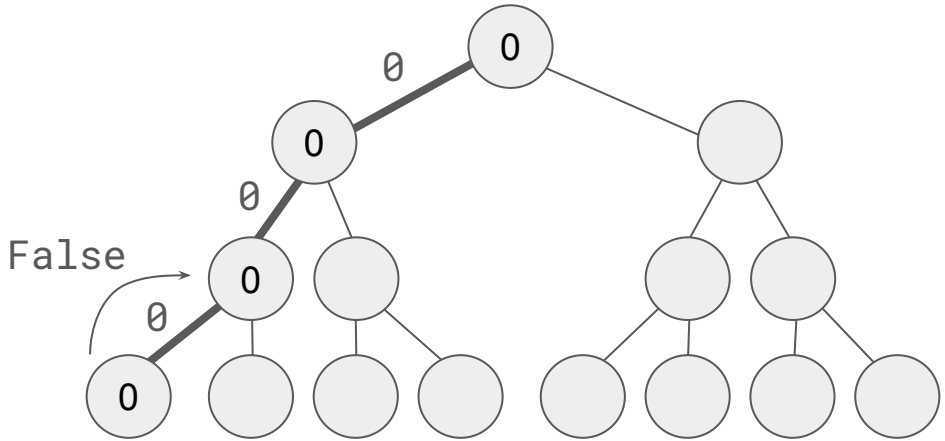
Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

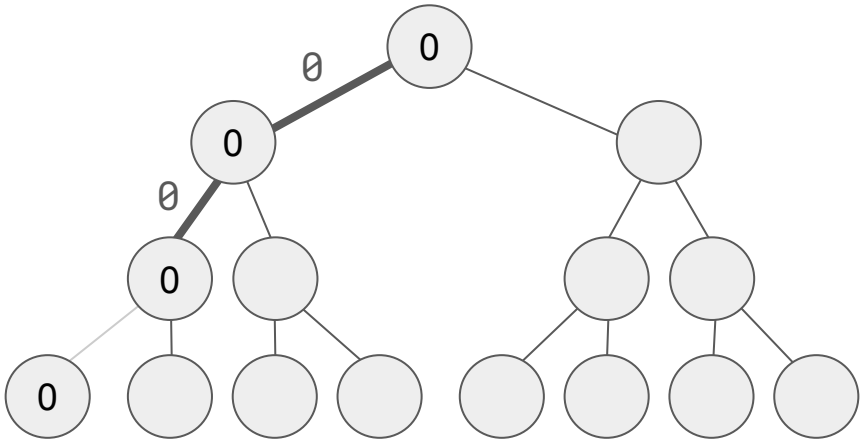
Recherche en profondeur | **Exemple pas à pas**

Retourne car la profondeur max est atteinte.



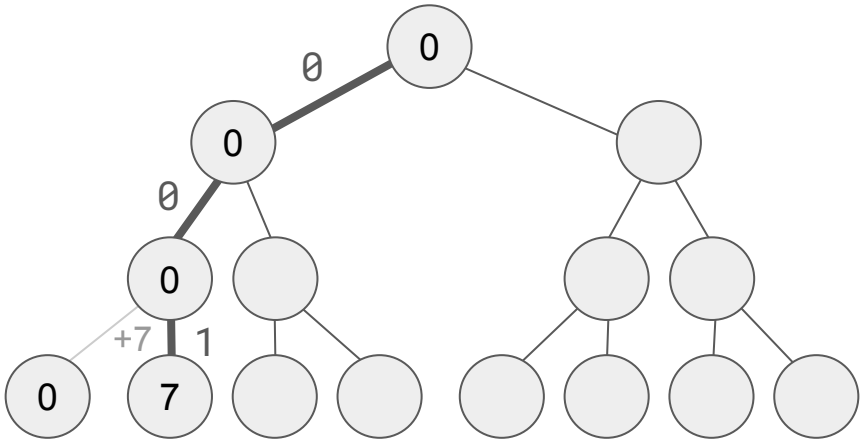
On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

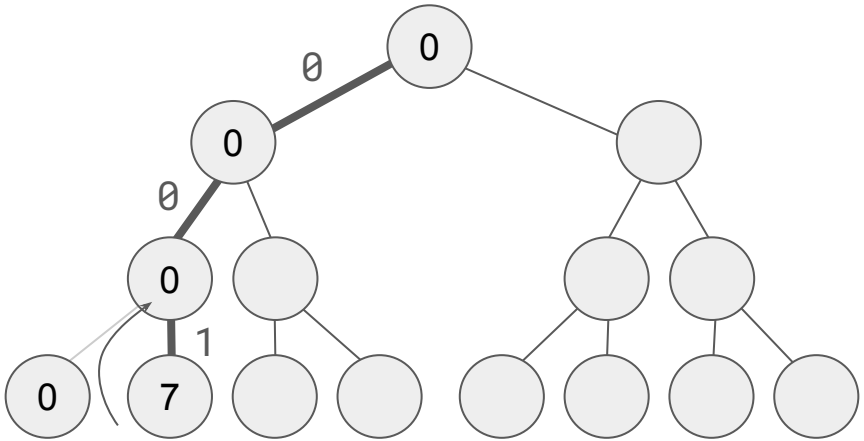
Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**

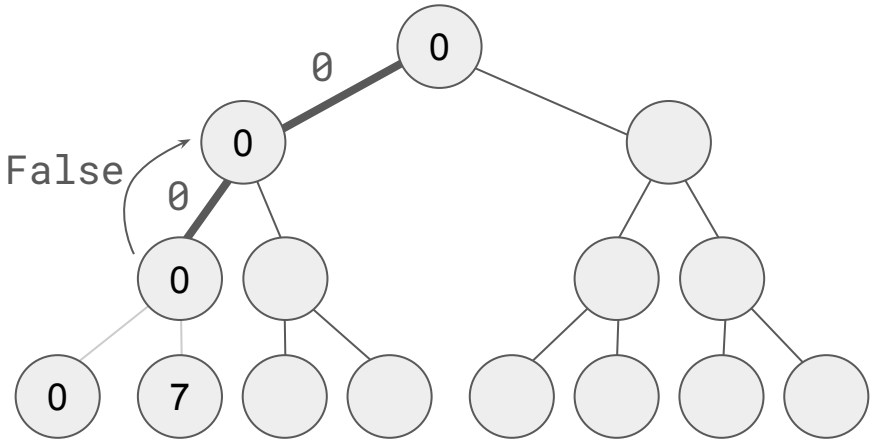
Retourne car la profondeur max est atteinte.



On cherche le chemin donnant S = 12.

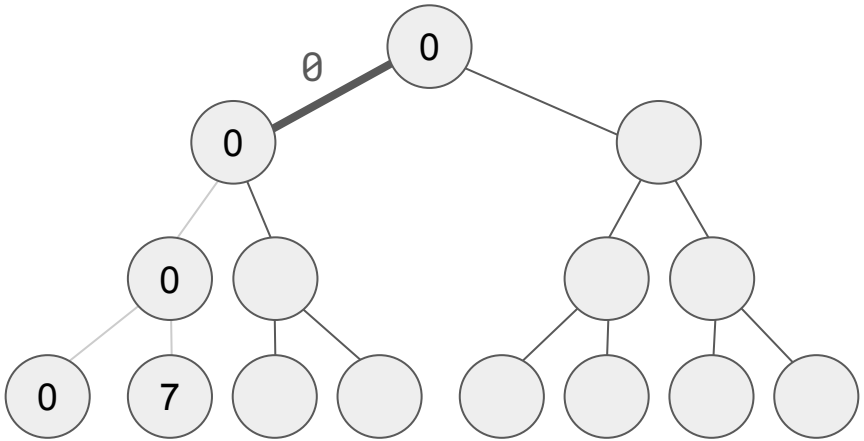
Recherche en profondeur | **Exemple pas à pas**

Retourne car tous les enfants ont été générés.



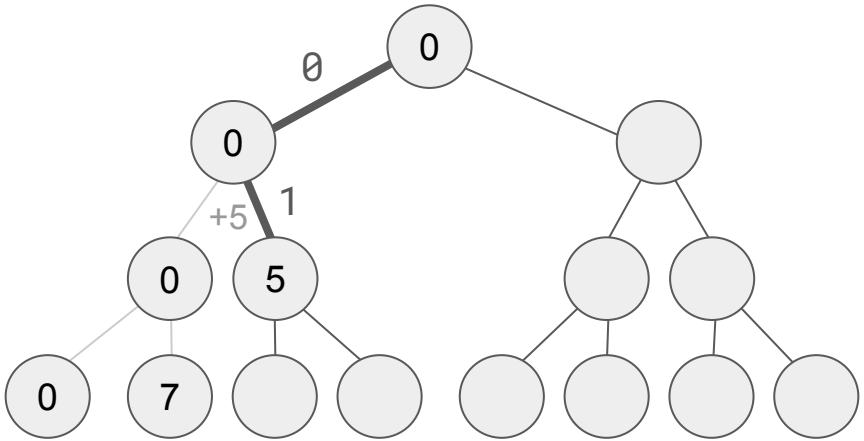
On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**



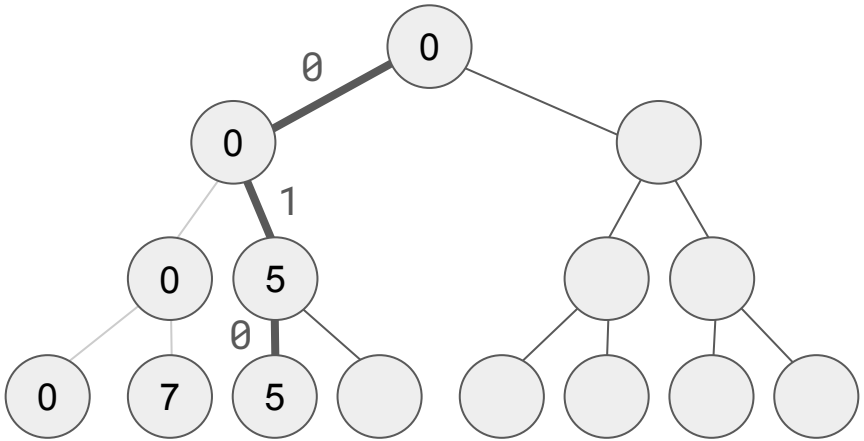
On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

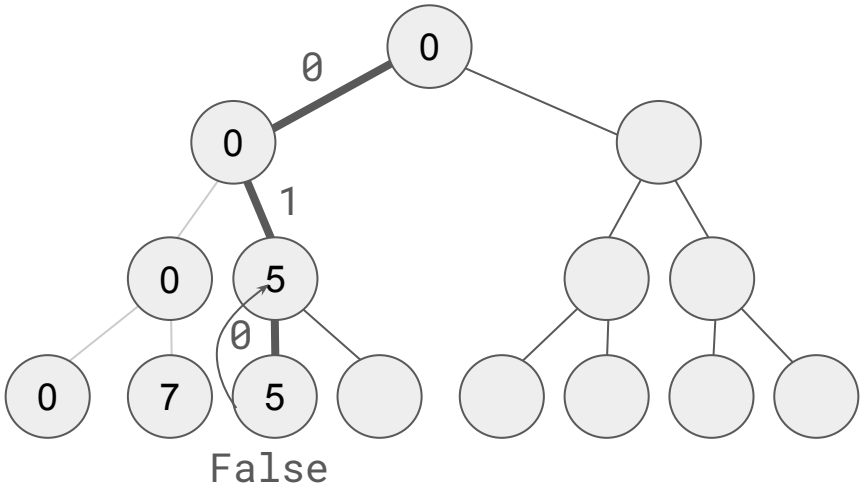
Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

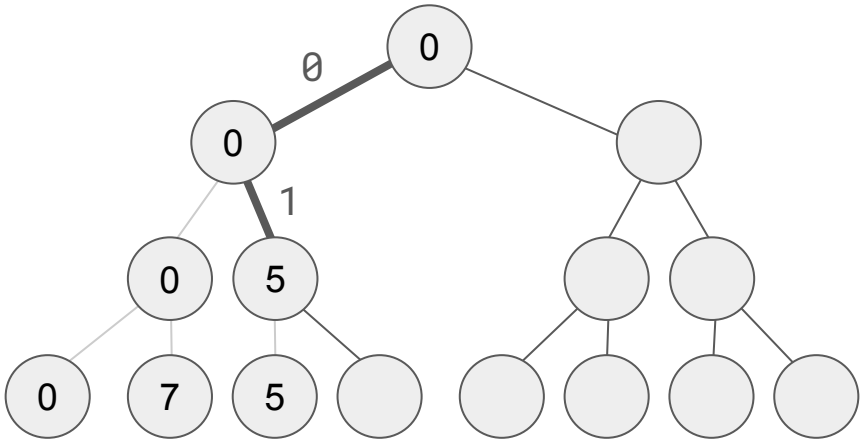
Recherche en profondeur | **Exemple pas à pas**

Retourne car la profondeur max est atteinte.



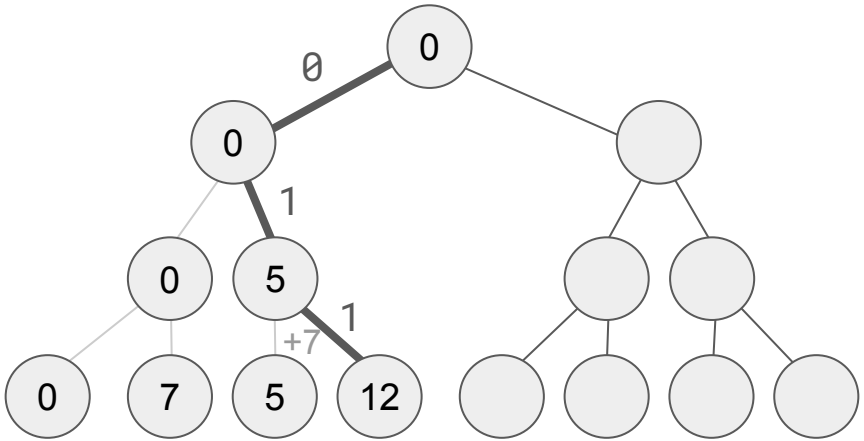
On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

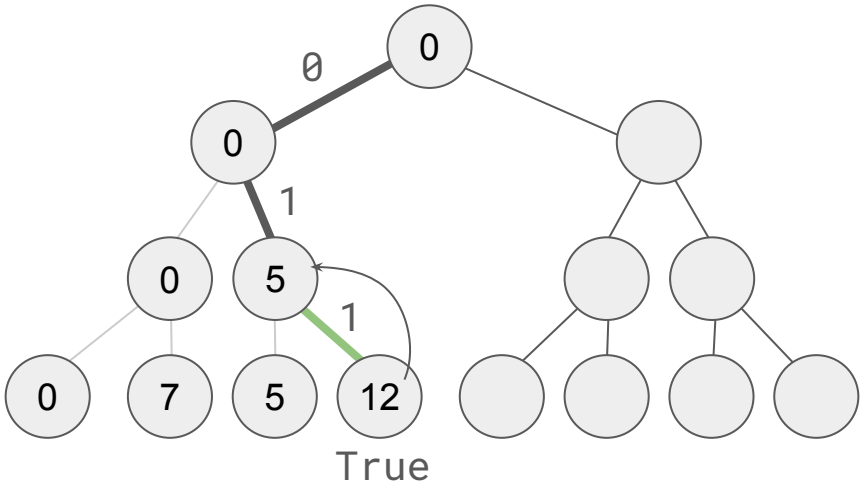
Recherche en profondeur | **Exemple pas à pas**



On cherche le chemin donnant S = 12.

Recherche en profondeur | **Exemple pas à pas**

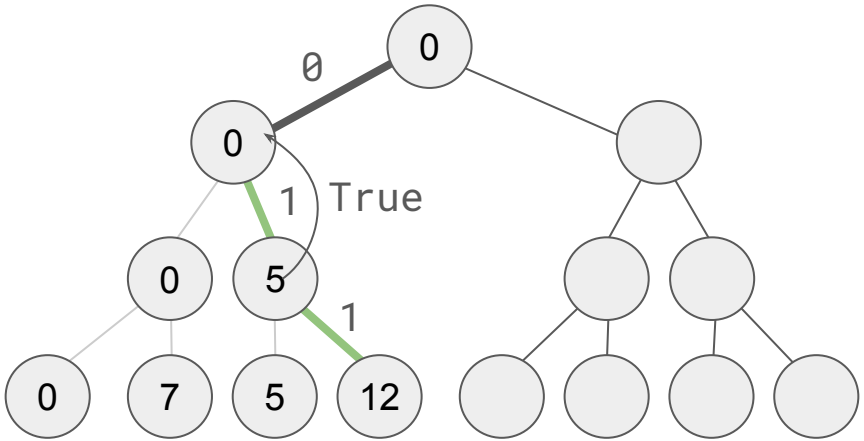
Retourne car S = 12.



On cherche le chemin donnant S = 12.

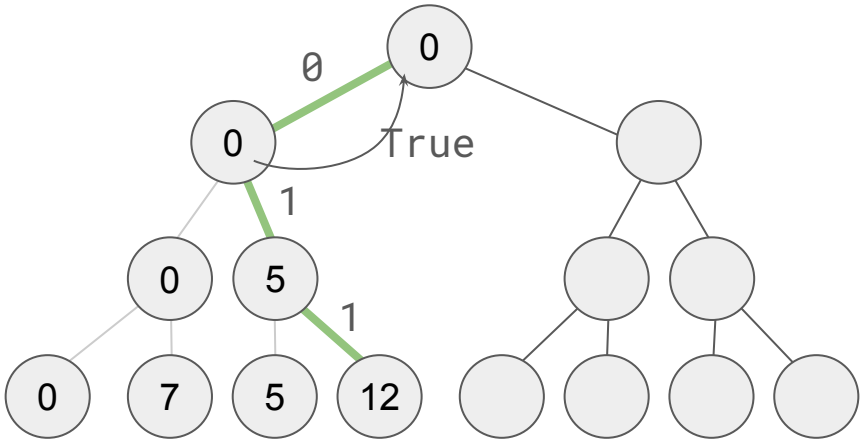
Recherche en profondeur | **Exemple pas à pas**

Retourne car l'un des enfants a retourné `True`.



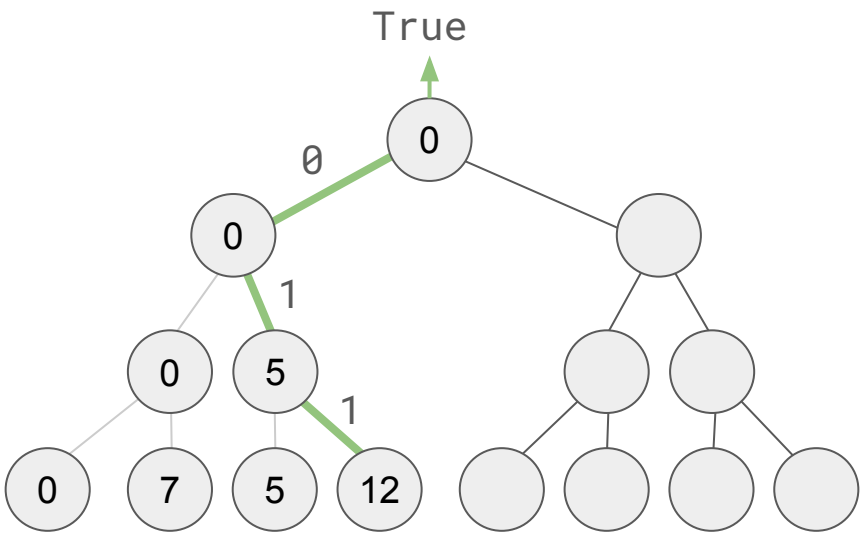
Recherche en profondeur | **Exemple pas à pas**

Retourne car l'un des enfants a retourné `True`.



Recherche en profondeur | **Exemple pas à pas**

Fin.



Recherche en profondeur | **Exemple pas à pas**

- La méthode qui vient d'être vue constitue une **recherche en profondeur**, ou DFS (*Depth-First Search*), car la priorité est systématique donnée aux nouveaux enfants avant l'exploration des noeuds frères.

- De nombreux raffinements sont possibles selon les détails du problème.

- Par exemple, si E ne contient que des nombres positifs, un noeud retourne `False` dès que la somme provisoire excède S. C'est ce que l'on nomme un **backtracking**.

- De la même façon, on peut explorer l'ensemble des mots de passe de longueur donnée possibles, des parties d'échecs possibles, etc. Mais, en général, ces ensembles sont bien trop grands pour les ordinateurs, même les plus puissants !

Recherche en profondeur | **Idée de l'algorithme**

À chaque fois qu'un noeud est généré, on effectue dans l'ordre :

1. Si la somme associée au chemin qui mène jusque là vaut S, retourner `True`.
2. Si la profondeur maximale est atteinte, retourner `False`.
3. Générer l'enfant de gauche. S'il retourne `True`, retourner `True`.
4. Générer l'enfant de droite. Retourner ce qu'il retourne.

Recherche en profondeur | **Idée de l'algorithme**

Exemple de code Python récursif :

```
E = [3,5,7,8]
n = len(E)
S = 13
def parcours(i, valeur):
    if valeur == S: # somme exacte atteinte
        return True
    if i == n: # profondeur max atteinte
        return False
    if parcours(i+1, valeur): # l'un des descendants de gauche a atteint la somme exacte
        return True
    return parcours(i+1, valeur + E[i]) # retourner ce que la branche de droite retourne
```

(Hors champ)