Algorithmique et structure de données

Structures de données - suite

Rohan Fossé

18 Octobre 2021

Rappels

Rappel du cours précédent

Les piles

Une pile est une liste linéaire d'objets où les consultations, les insertions et les suppressions se font du même côté. On dit que c'est le dernier arrivé, premier servi.

Rappel du cours précédent

Les piles

Une pile est une liste linéaire d'objets où les consultations, les insertions et les suppressions se font du même côté. On dit que c'est le dernier arrivé, premier servi.

Les listes

Une liste chaînée désigne en informatique une structure de données représentant une collection ordonnée et de taille arbitraire d'éléments de même type.

Rappel du cours précédent

Les piles

Une pile est une liste linéaire d'objets où les consultations, les insertions et les suppressions se font du même côté. On dit que c'est le dernier arrivé, premier servi.

Les listes

Une liste chaînée désigne en informatique une structure de données représentant une collection ordonnée et de taille arbitraire d'éléments de même type.

Les files

La file est une structure de données abstraite, un peu similaire aux piles. Contrairement aux piles, une file d'attente est ouverte à ses deux extrémités. On dit que c'est le premier arrivé, premier servi.

Rappel du cours précédent

Les piles

Une pile est une liste linéaire d'objets où les consultations, les insertions et les suppressions se font du même côté. On dit que c'est le dernier arrivé, premier servi.

Les listes

Une liste chaînée désigne en informatique une structure de données représentant une collection ordonnée et de taille arbitraire d'éléments de même type.

Les files

La file est une structure de données abstraite, un peu similaire aux piles. Contrairement aux piles, une file d'attente est ouverte à ses deux extrémités. On dit que c'est le premier arrivé, premier servi.

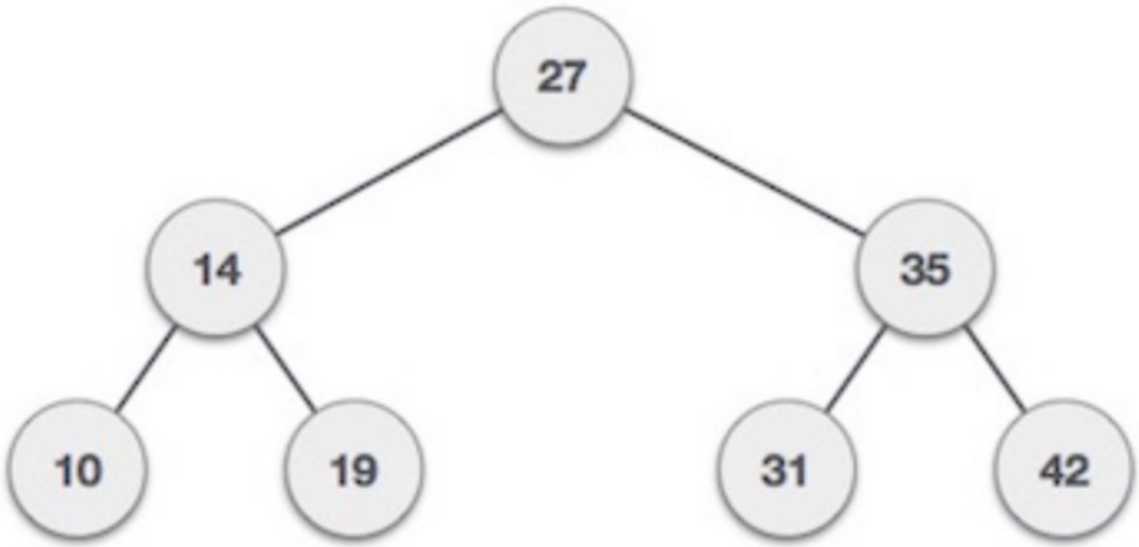
Les arbres binaires

L'arbre binaire a une condition spéciale : chaque nœud peut avoir un maximum de deux enfants.

Rappel sur les arbres binaire de recherche

Rappel

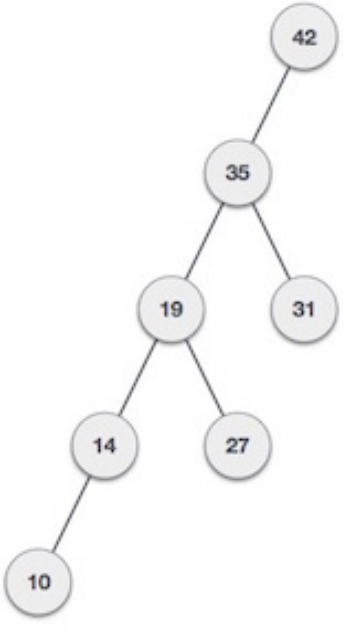
L'arbre de binaire de recherche (ou ABR) présente un comportement particulier. L'enfant gauche d'un nœud doit avoir une valeur inférieure à celle de son parent et l'enfant droit du nœud doit avoir une valeur supérieure à celle de son parent.



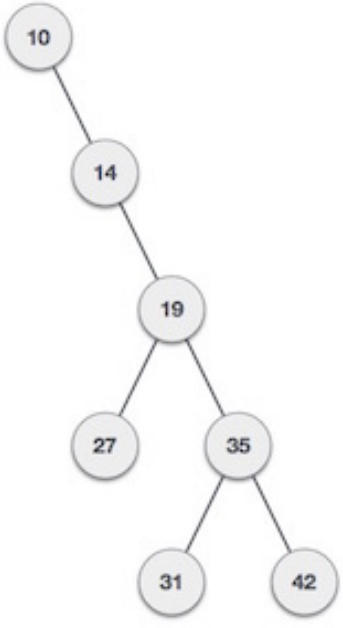
AVL

Les AVL

Que se passe-t-il si l'entrée de l'arbre de recherche binaire est triée (ascendante ou descendante) ? Il ressemblera alors à ceci :

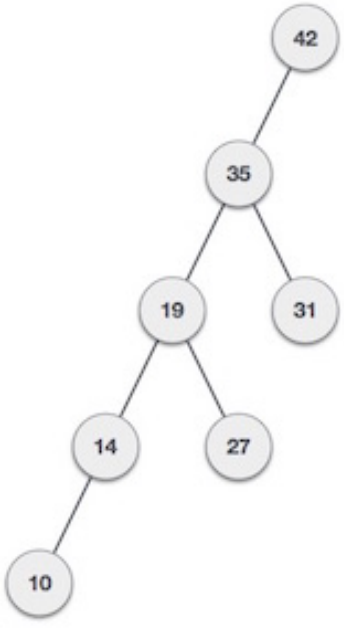


If input 'appears' non-increasing manner

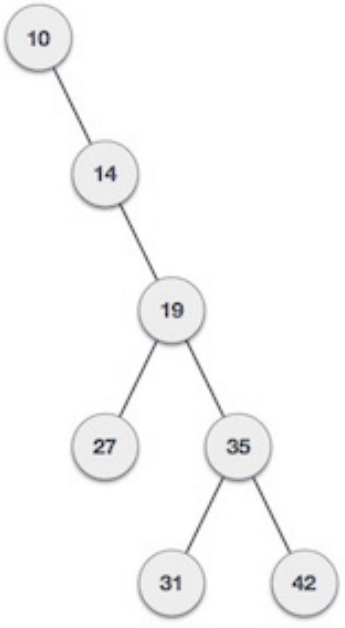


If input 'appears' in non-decreasing manner

Les AVL



If input 'appears' non-increasing manner



If input 'appears' in non-decreasing manner

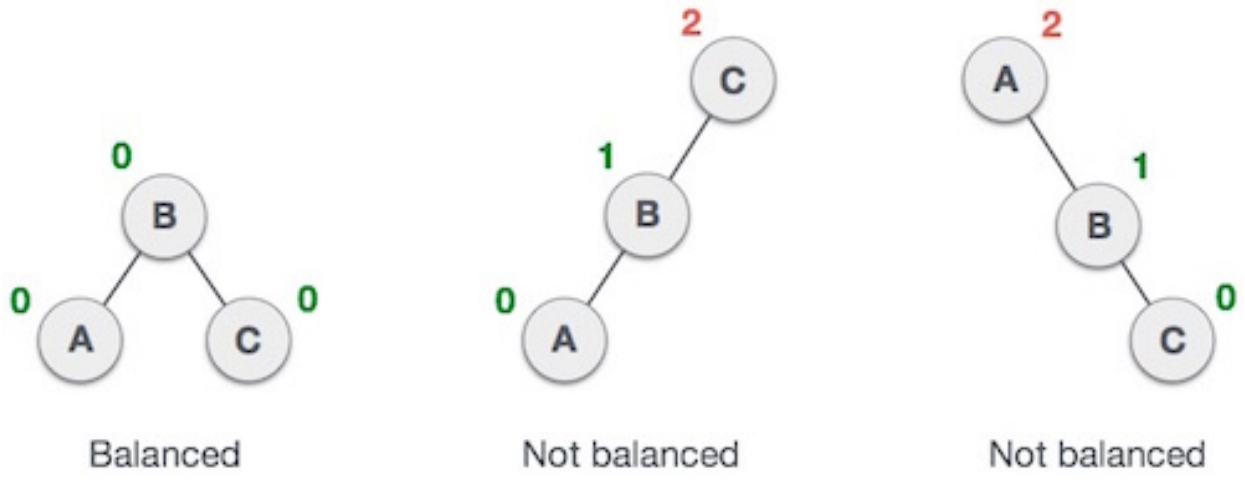
On observe que la performance de recherche dans les ABR dans le pire des cas est la plus proche des algorithmes de recherche linéaire, soit $\mathcal{O}(n)$. . Ainsi, on a besoin d'utiliser une méthode pour équilibrer les ABR.

Les AVL

Définition

Nommés d'après leur inventeur *Adelson, Velski et Landis*, les arbres AVL sont des arbres de recherche binaire à équilibrage de hauteur. L'arbre AVL vérifie la hauteur des sous-arbres de gauche et de droite et s'assure que la différence n'est pas supérieure à 1. Cette différence est appelée **le facteur d'équilibre**.

Les AVL



- Deuxième arbre: le sous-arbre gauche de **C** a une hauteur de **2** et le sous-arbre droit a une hauteur de **0**, la différence est donc de **2**;
- Troisième arbre: le sous-arbre droit de **A** a une hauteur de **2** et le sous-arbre gauche a une hauteur de **0**, la différence est donc de **2**.

Les AVL

Le facteur d'équilibre

Facteur-équilibre = taille(sous-arbre-gauche) - taille(sous-arbre-droit)

Si la différence de hauteur entre les sous-arbres gauche et droit est supérieure à $\mathbf{1}$, l'arbre est équilibré à l'aide de certaines techniques de rotation.

Rotations des AVL

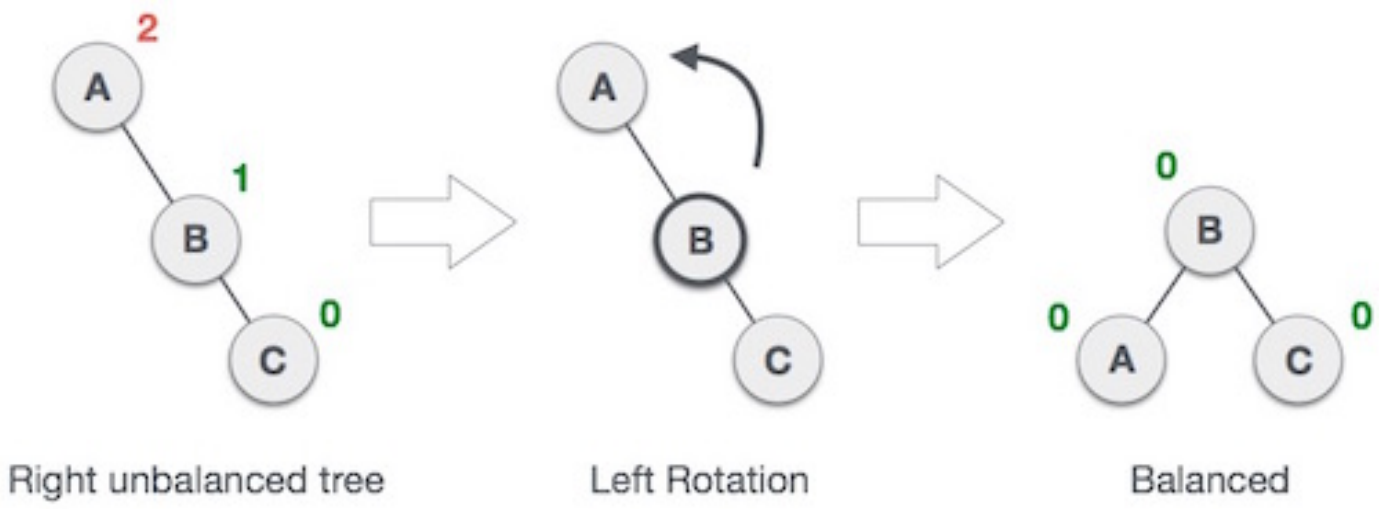
Pour s'équilibrer, un arbre **AVL** peut effectuer les quatre types de rotations suivants:

- Rotation vers la gauche;
- Rotation vers la droite;
- Rotation gauche-droite;
- Rotation droite-gauche.

Les deux premières rotations sont des rotations simples et les deux suivantes sont des rotations doubles. Pour avoir un arbre déséquilibré, il nous faut au moins un arbre de hauteur **2**.

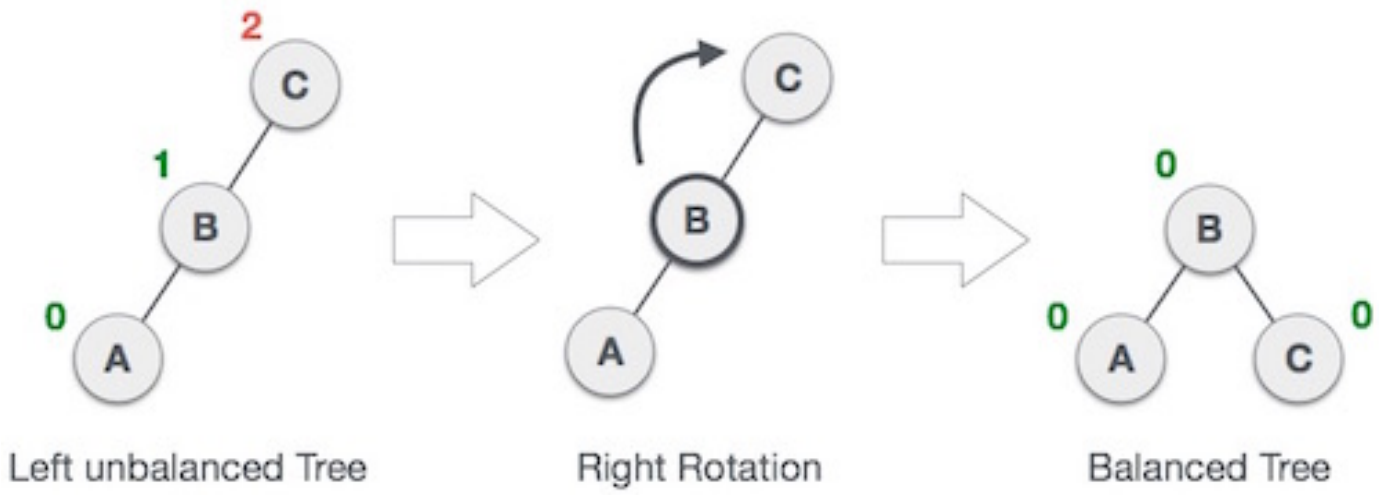
Rotation vers la gauche

Si un arbre devient déséquilibré, lorsqu'un nœud est inséré dans le sous-arbre droit du sous-arbre droit, nous effectuons alors une seule rotation vers la gauche:



Rotation vers la droite

Si un arbre devient déséquilibré, lorsqu'un nœud est inséré dans le sous-arbre gauche du sous-arbre droite, nous effectuons alors une seule rotation vers la gauche:

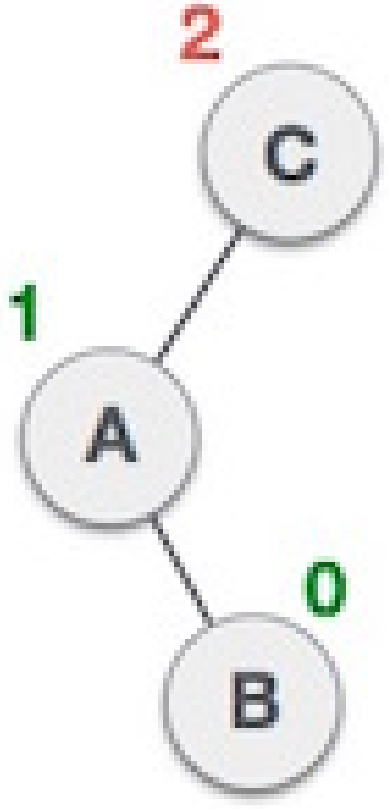


Rotation gauche-droite

Les **doubles rotations** sont une version légèrement plus complexe des versions déjà expliquées des rotations. Voyons d'abord comment effectuer une rotation gauche-droite. Une rotation gauche-droite est une combinaison d'une rotation gauche suivie d'une rotation droite.

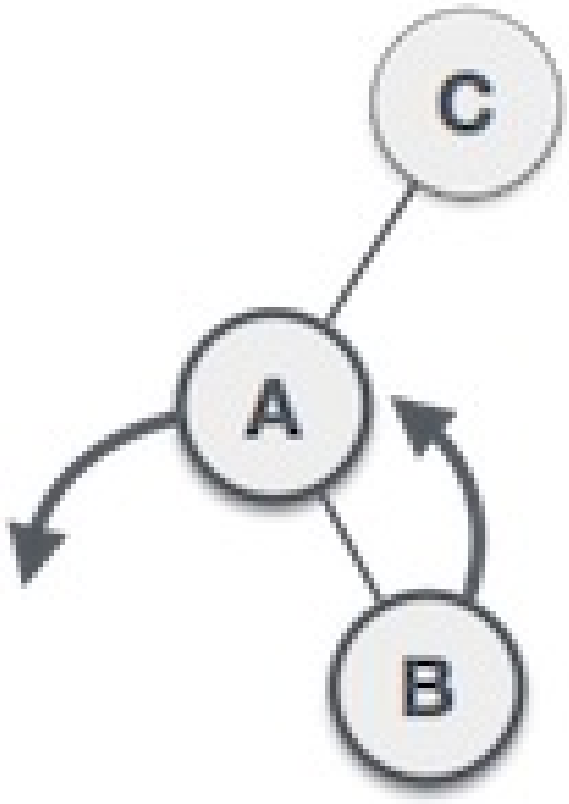
Rotation gauche-droite

Un nœud a été inséré dans le sous-arbre de droite du sous-arbre de gauche. Cela fait de C un nœud déséquilibré. Ces scénarios font que l'arbre AVL effectue une rotation gauche-droite.



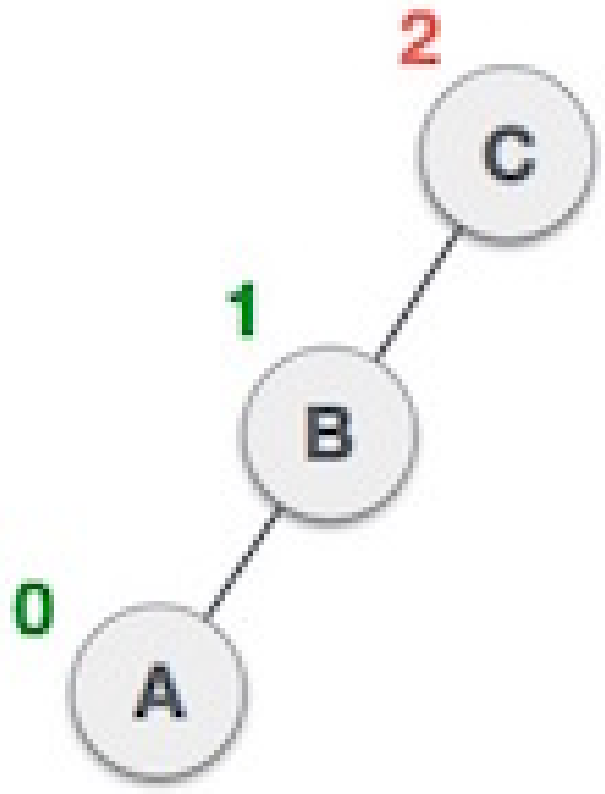
Rotation gauche-droite

Nous effectuons d'abord la rotation gauche sur le sous-arbre gauche de C. Cela fait de A, le sous-arbre gauche de B.



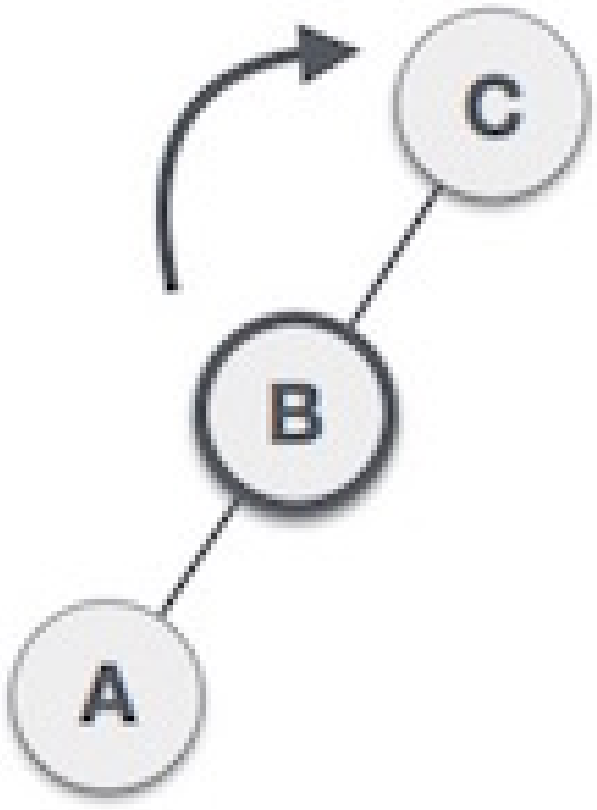
Rotation gauche-droite

Le nœud C est toujours déséquilibré, mais maintenant, c'est à cause du sous-arbre gauche du sous-arbre gauche.



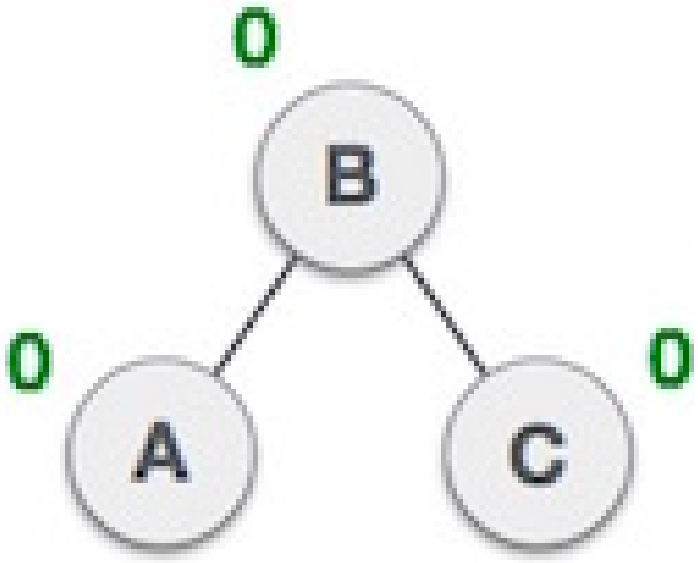
Rotation gauche-droite

Nous allons maintenant effectuer une rotation à droite de l'arbre, faisant de B le nouveau nœud racine de ce sous-arbre. C devient maintenant le sous-arbre droit de son propre sous-arbre gauche.



Rotation gauche-droite

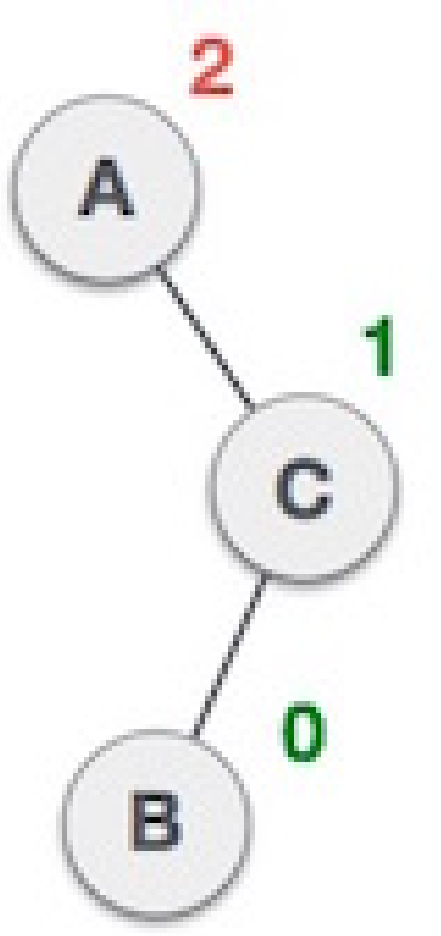
L'arbre est maintenant équilibré.



Rotation droite-gauche

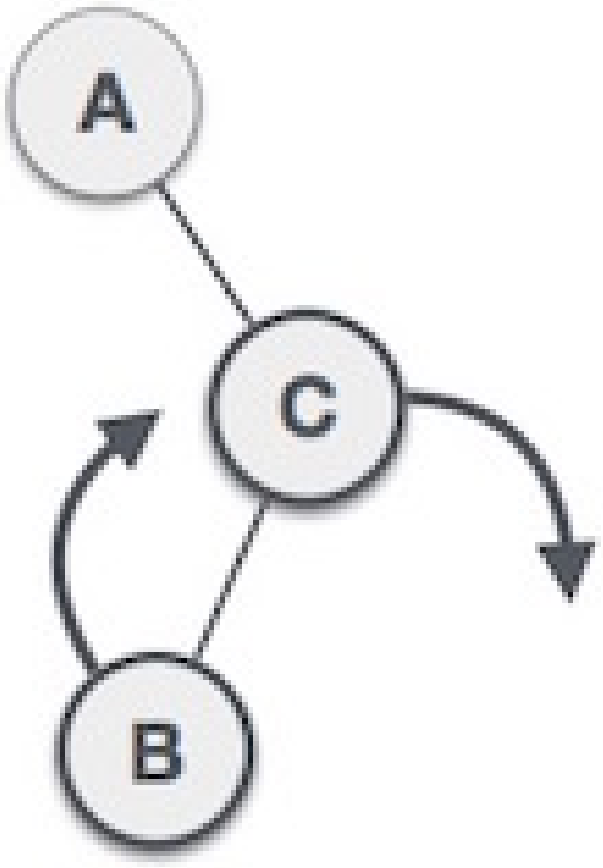
Le deuxième type de double rotation est la rotation droite-gauche. Il s'agit d'une combinaison de la rotation droite suivie de la rotation gauche.

Un nœud a été inséré dans le sous-arbre gauche du sous-arbre droit. Cela fait de A, un nœud déséquilibré avec un facteur d'équilibre de 2.



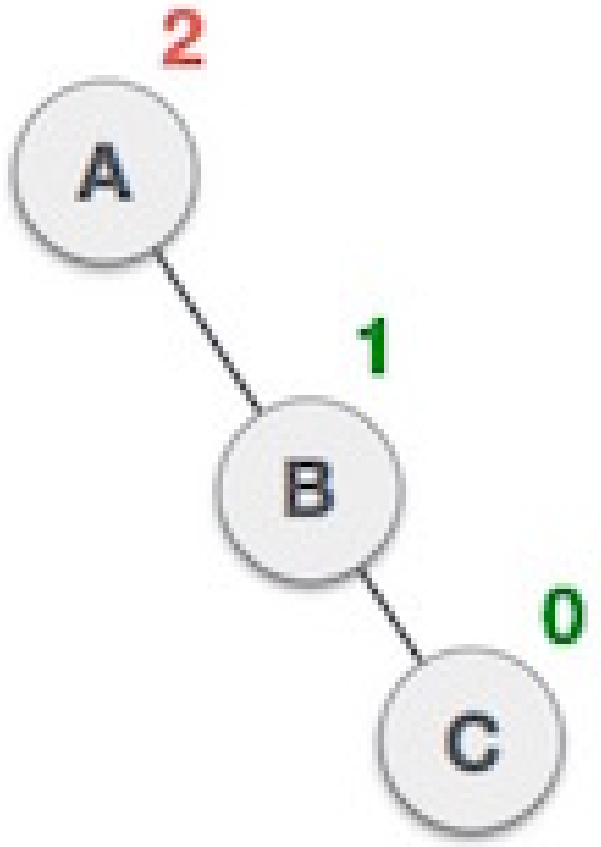
Rotation droite-gauche

Tout d'abord, nous effectuons une rotation vers la droite le long du nœud C, faisant de C le sous-arbre droit de son propre sous-arbre gauche B. Maintenant, B devient le sous-arbre droit de A.



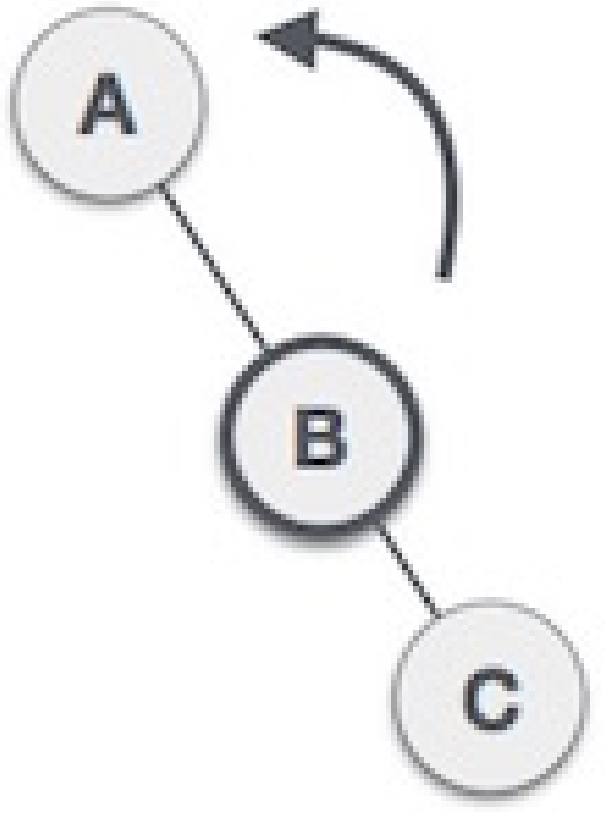
Rotation droite-gauche

Le nœud A est toujours déséquilibré à cause du sous-arbre droit de son sous-arbre droit et nécessite une rotation vers la gauche.



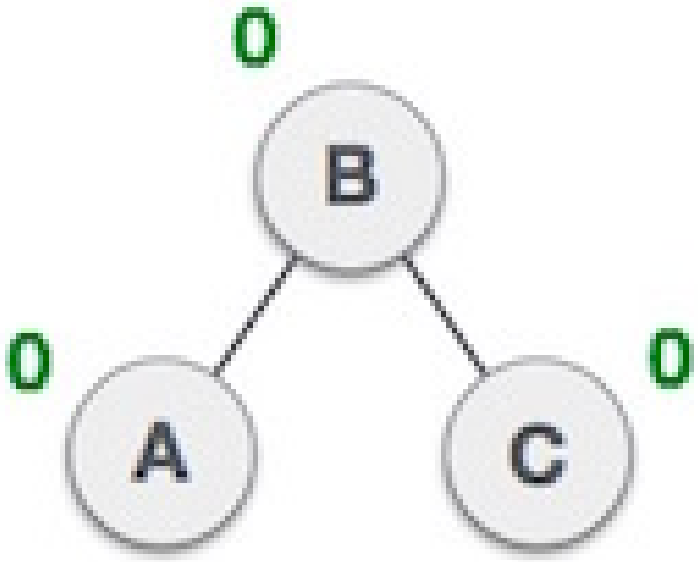
Rotation droite-gauche

Une rotation vers la gauche est effectuée en faisant de B le nouveau nœud racine du sous-arbre. A devient le sous-arbre gauche de son sous-arbre droit B.



Rotation droite-gauche

L'arbre est maintenant équilibré.



Les tas

Les tas (Heap)

Le tas est un cas particulier de structure de données d'arbre binaire équilibré où la clé du nœud racine est comparée à ses enfants et arrangée en conséquence. Par exemple, si α a un noeud enfant β alors - $cl(\alpha) \geq cl(\beta)$.

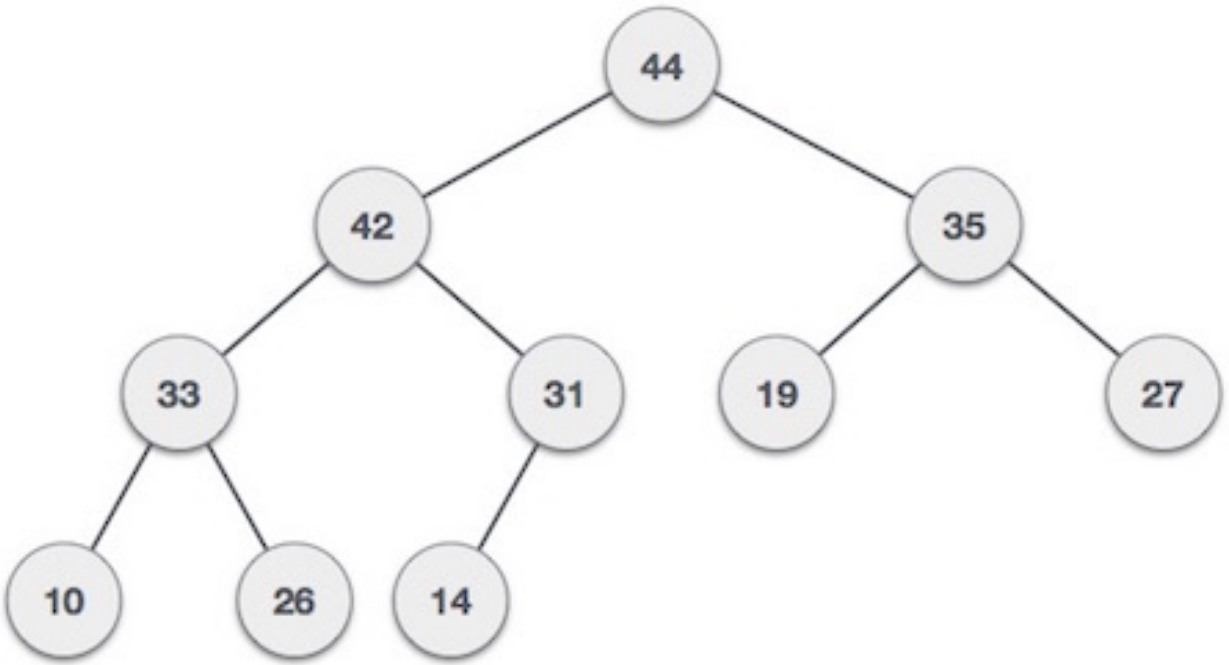
Comme la valeur du parent est supérieure à celle de l'enfant, cette propriété génère un Max Heap. Sur la base de ce critère, un tas peut être de deux types :

Max Heap

Lorsque la valeur du nœud racine est supérieure ou égale à celle de l'un de ses enfants.

Entrée

35 33 42 10 14 19 27 44 26 31

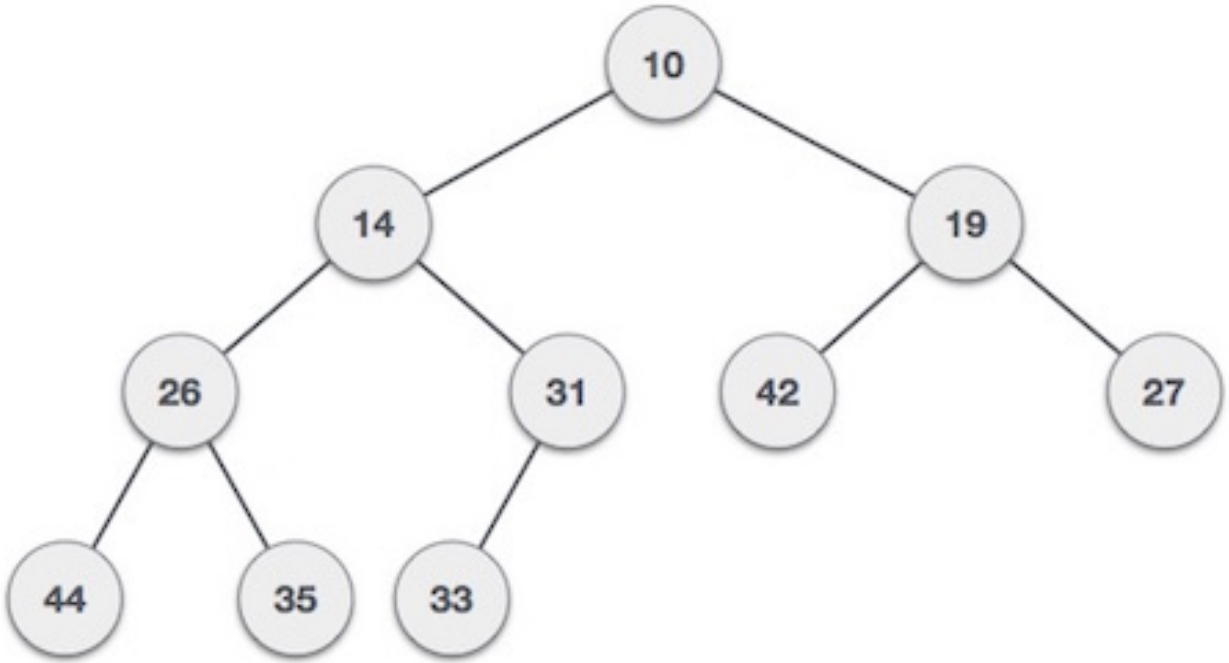


Min Heap

Lorsque la valeur du nœud racine est inférieure ou égale à celle de l'un de ses enfants.

Entrée

35 33 42 10 14 19 27 44 26 31



Algorithme de construction d'un Max Heap

Nous allons utiliser le même exemple pour montrer comment créer un Max Heap. La procédure de création du tas min est similaire, mais nous choisissons des valeurs min au lieu de valeurs max.

Nous allons dériver un algorithme pour le Max Heap en insérant un élément à la fois. À tout moment, le tas doit conserver sa propriété.

Algorithme

1. Créer un nouveau nœud à la fin du tas.
2. Attribuer une nouvelle valeur au noeud.
3. Comparer la valeur de ce noeud enfant avec son parent.
4. Si la valeur du parent est inférieure à celle de l'enfant, alors échangez-les.
5. Répétez les étapes 3 et 4 jusqu'à ce que la propriété du tas soit respectée.

Exemple de construction de Max Heap

Input **35** 33 42 10 14 19 27 44 26 31

Exemple de construction de Max Heap

Input **35** 33 42 10 14 19 27 44 26 31

Exemple de construction de Max Heap

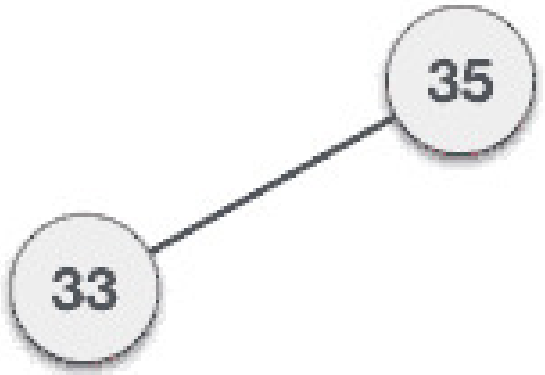
Input **35** 33 42 10 14 19 27 44 26 31

Exemple de construction de Max Heap

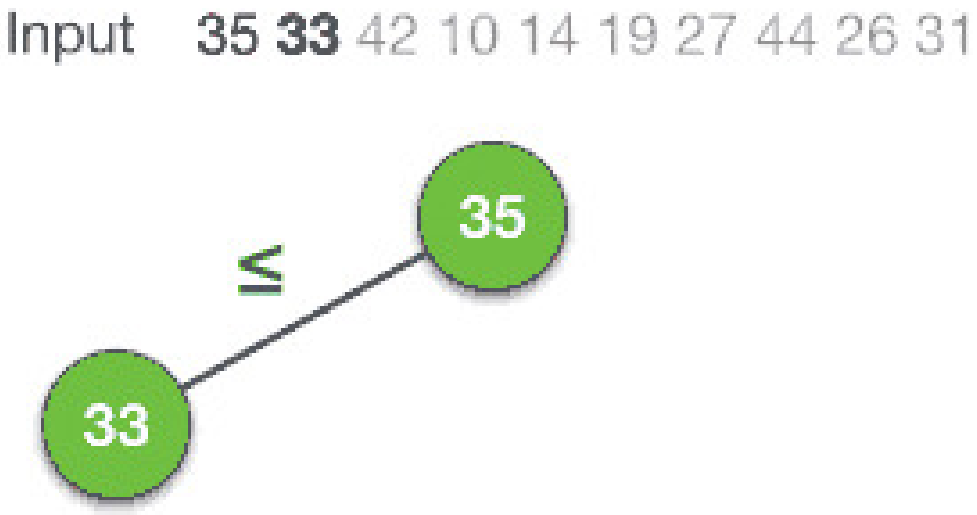
Input **35 33** 42 10 14 19 27 44 26 31

Exemple de construction de Max Heap

Input **35 33** 42 10 14 19 27 44 26 31

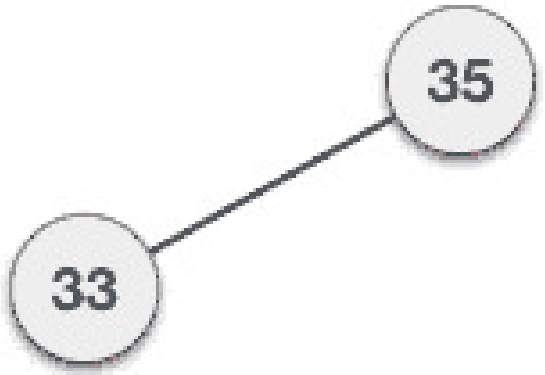


Exemple de construction de Max Heap



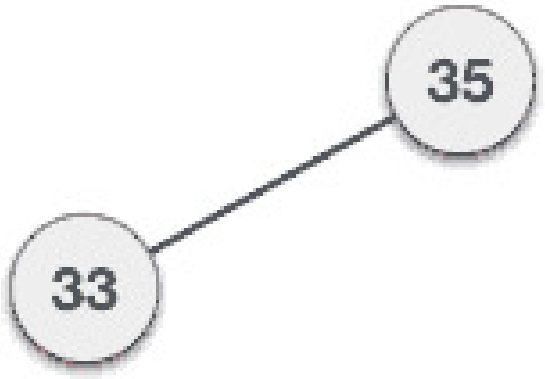
Exemple de construction de Max Heap

Input **35 33** 42 10 14 19 27 44 26 31



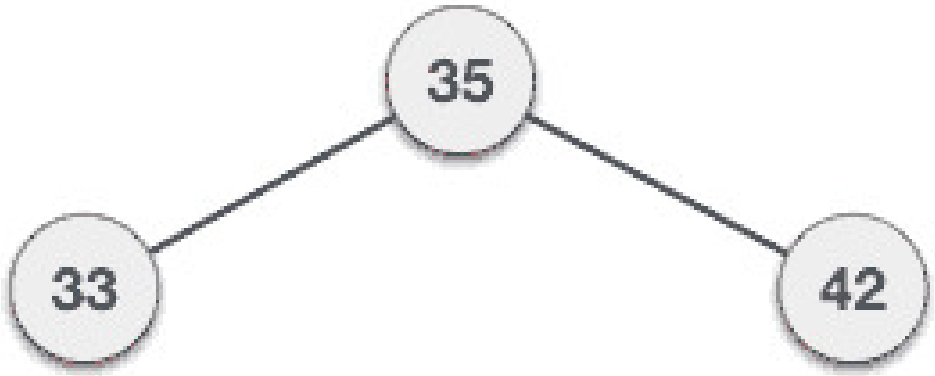
Exemple de construction de Max Heap

Input 35 33 **42** 10 14 19 27 44 26 31

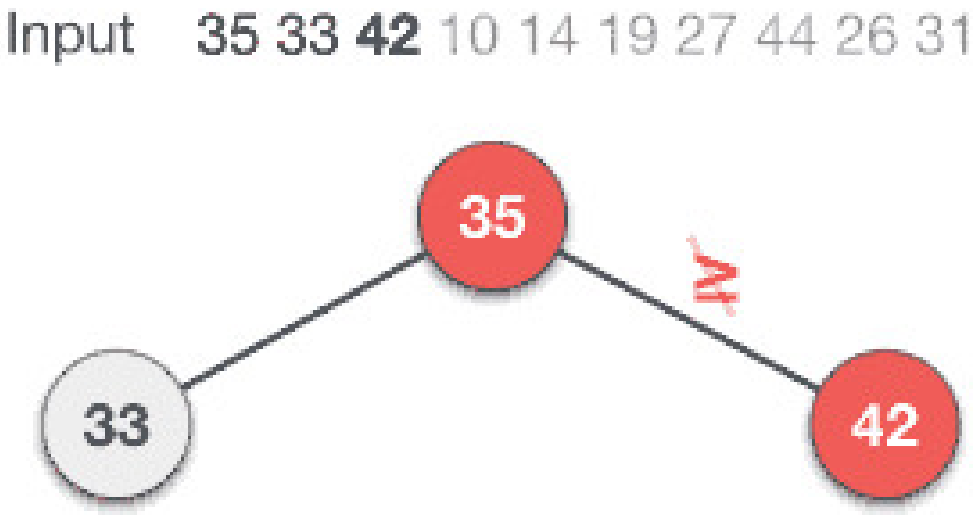


Exemple de construction de Max Heap

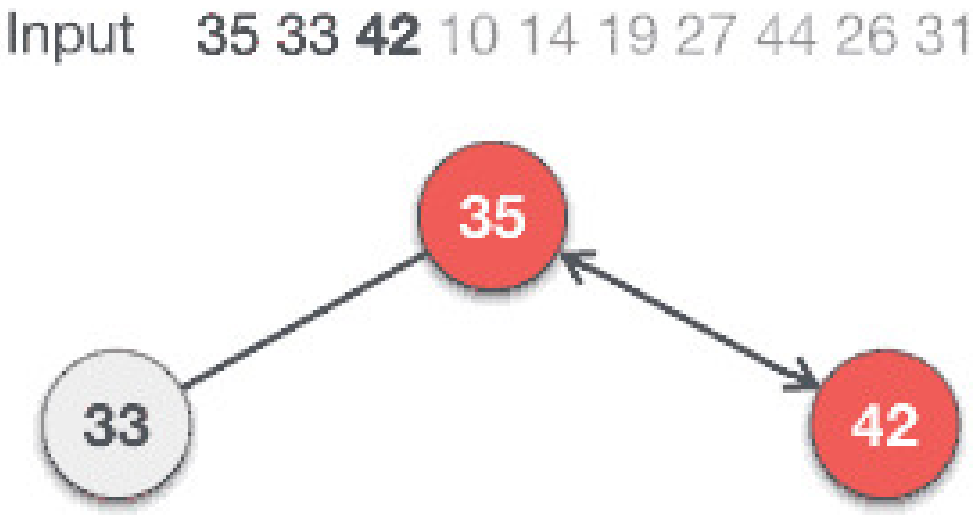
Input **35 33 42** 10 14 19 27 44 26 31



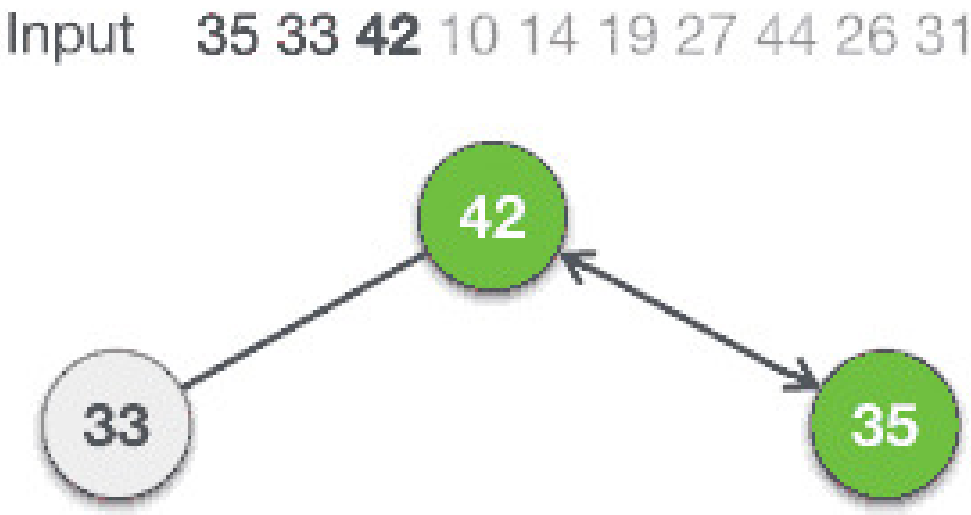
Exemple de construction de Max Heap



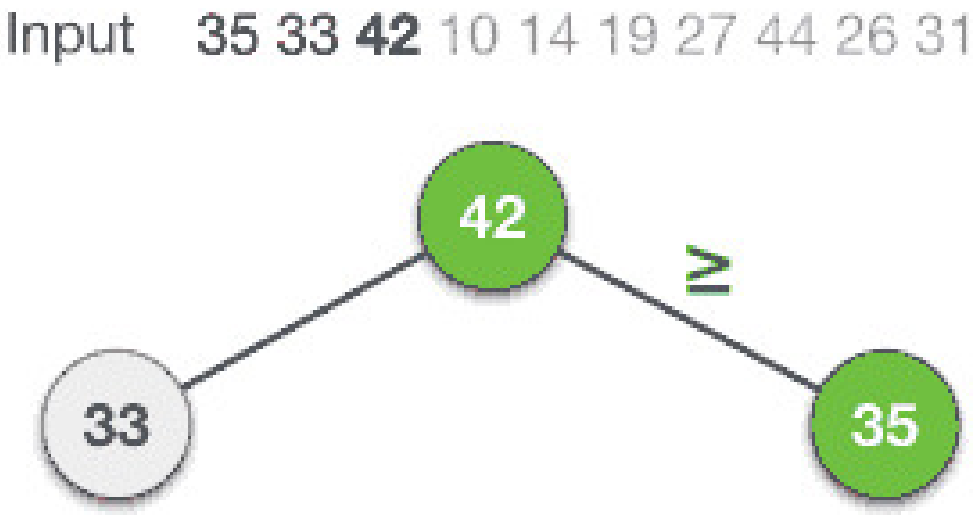
Exemple de construction de Max Heap



Exemple de construction de Max Heap



Exemple de construction de Max Heap



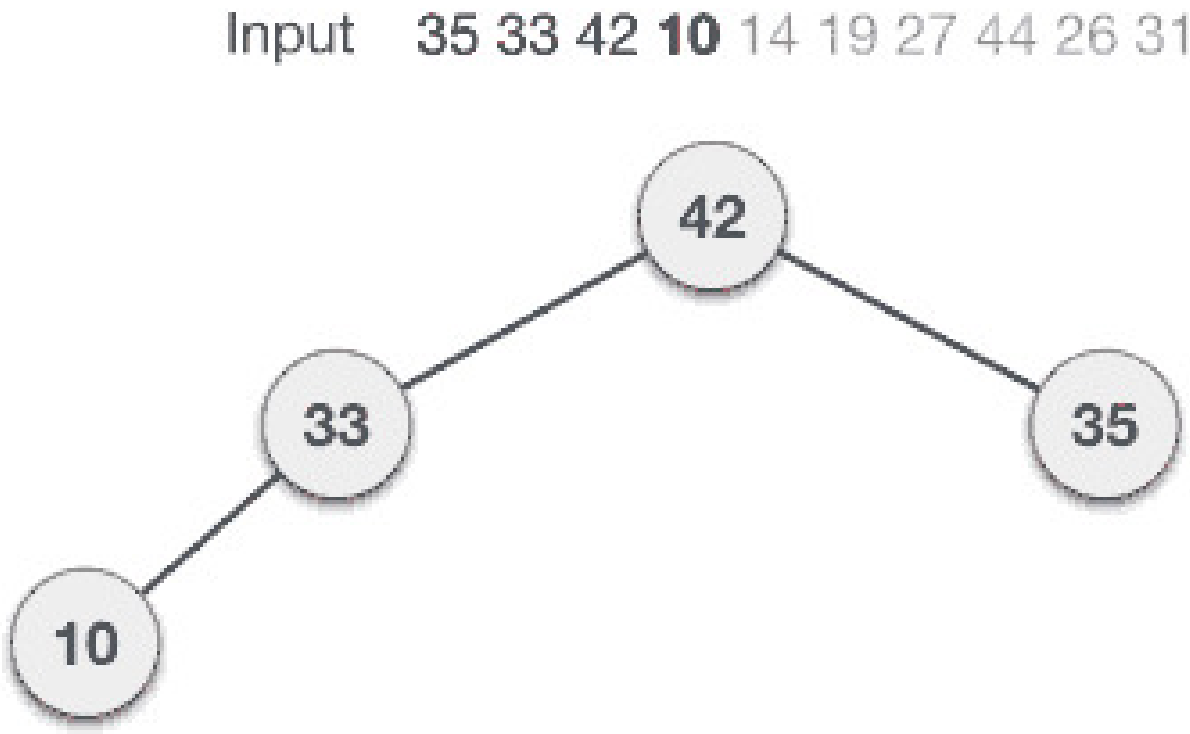
Exemple de construction de Max Heap

Input 35 33 **42** 10 14 19 27 44 26 31

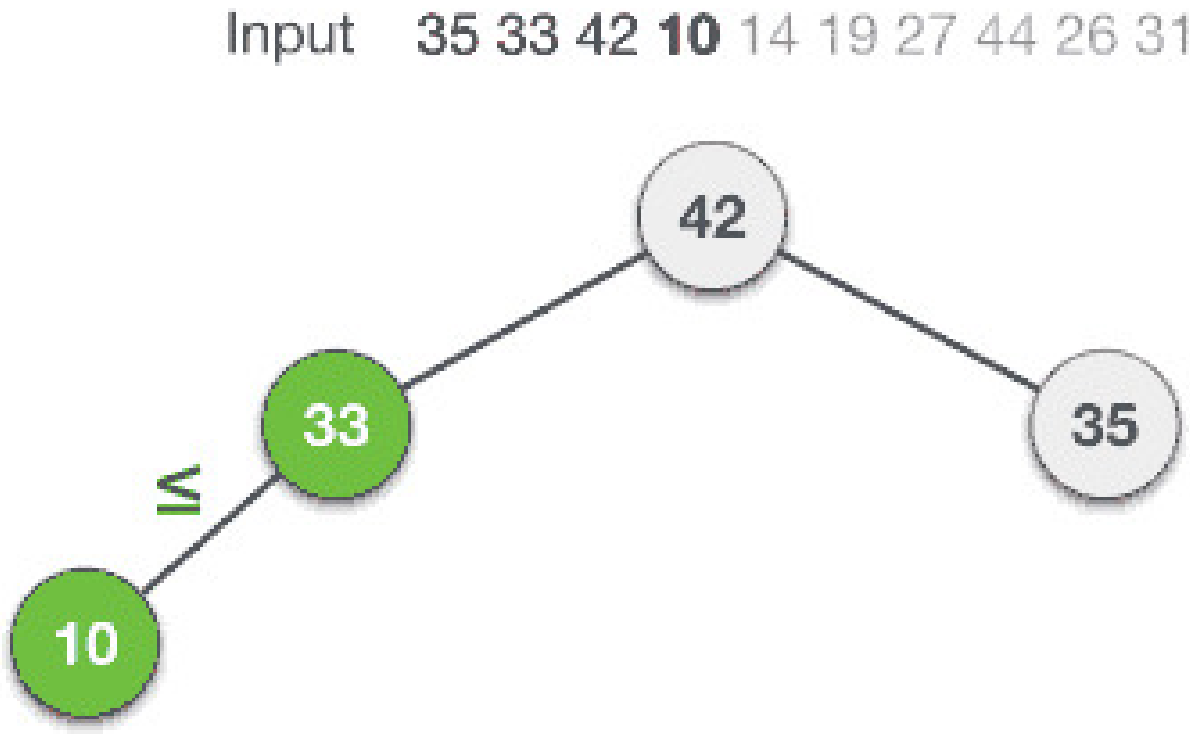
Exemple de construction de Max Heap

Input 35 33 42 **10** 14 19 27 44 26 31

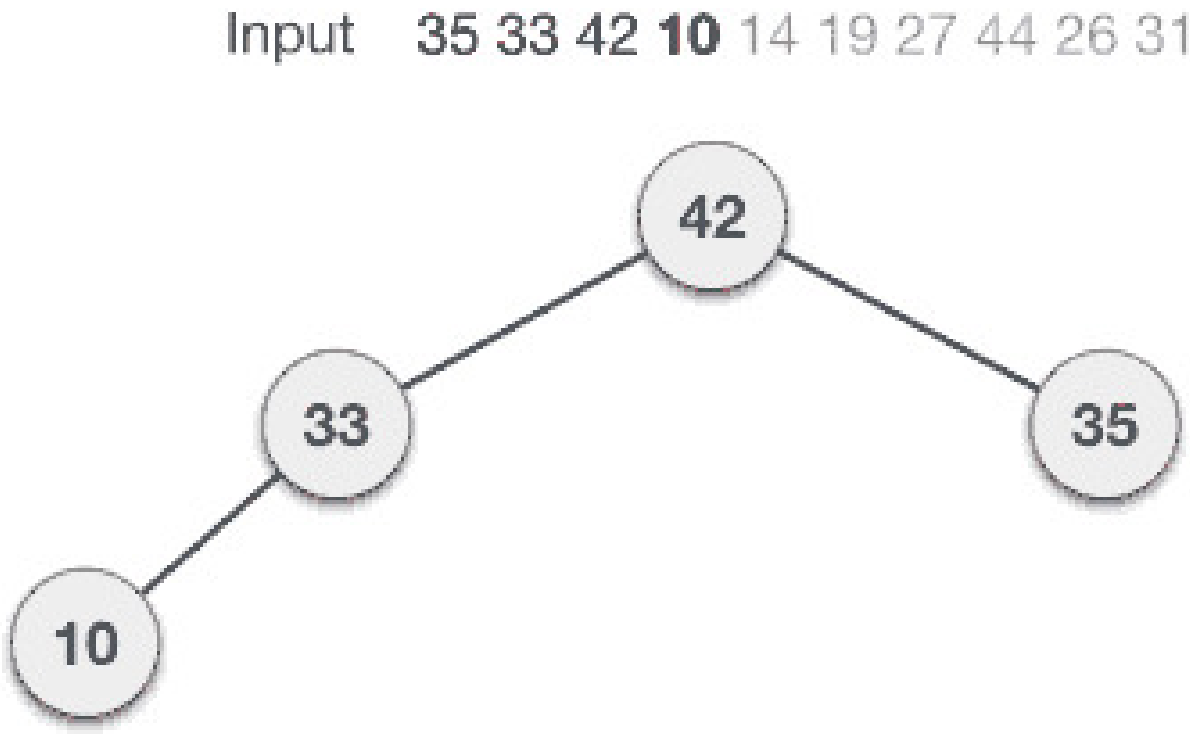
Exemple de construction de Max Heap



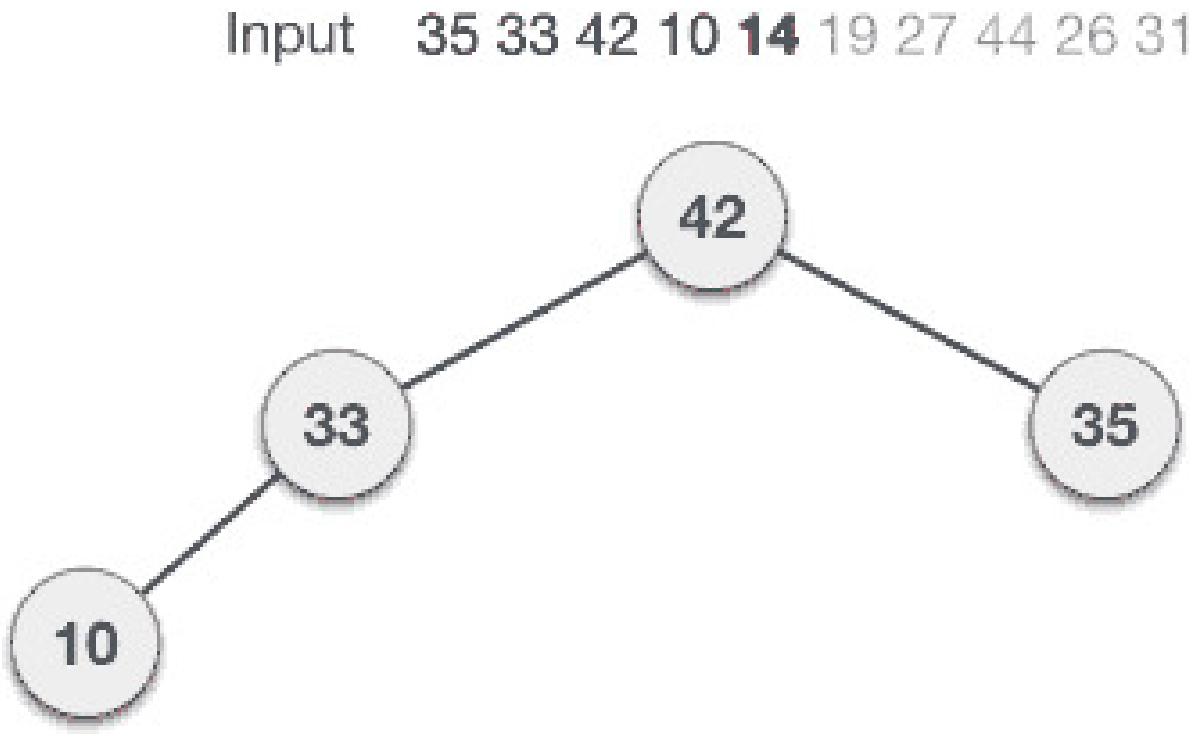
Exemple de construction de Max Heap



Exemple de construction de Max Heap

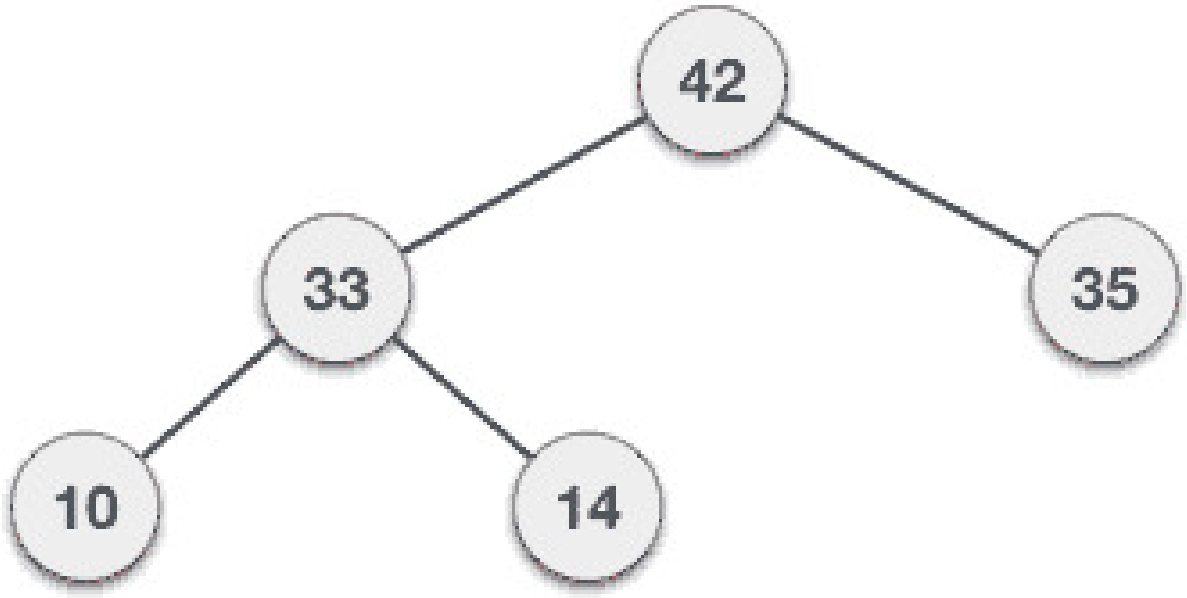


Exemple de construction de Max Heap

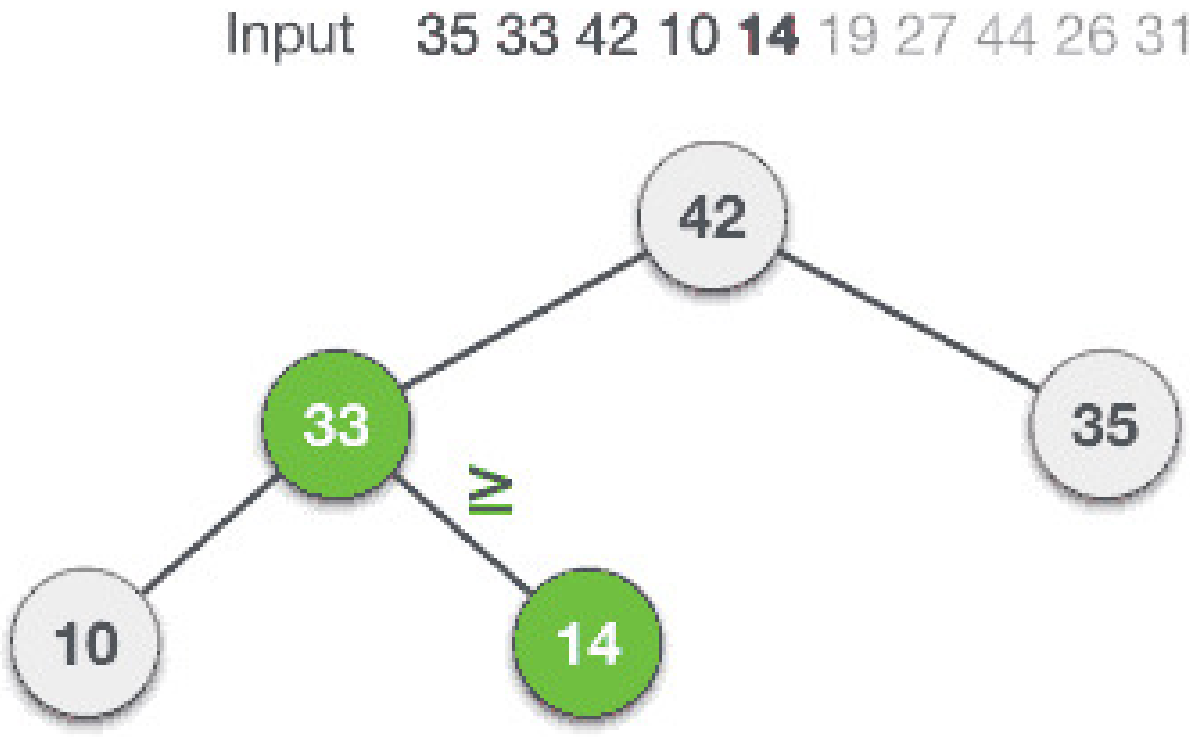


Exemple de construction de Max Heap

Input 35 33 42 10 **14** 19 27 44 26 31

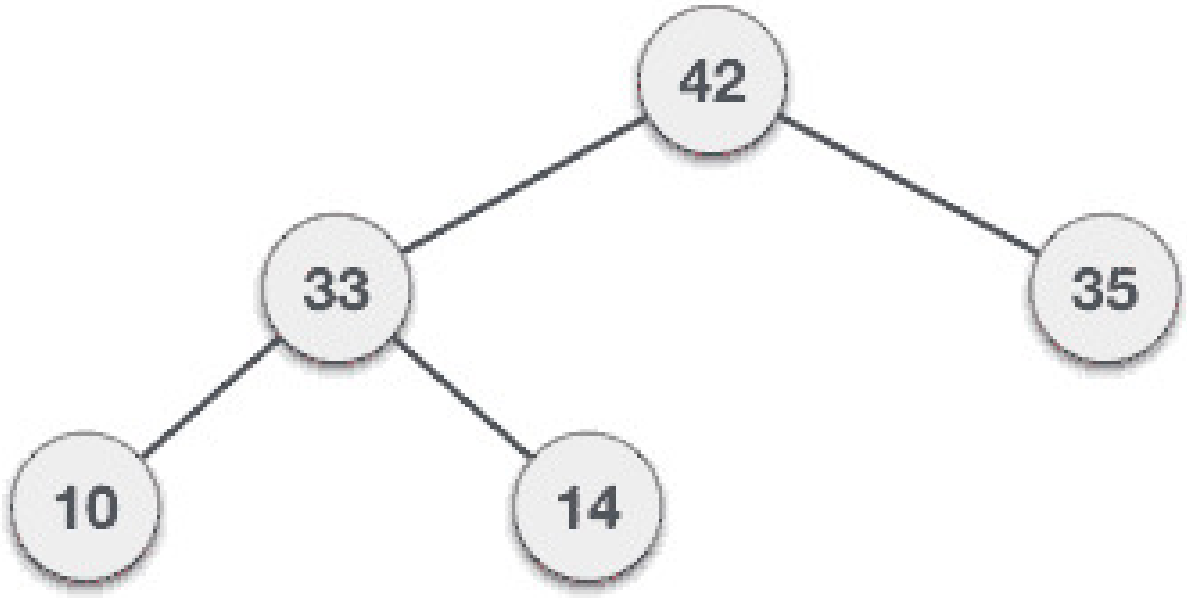


Exemple de construction de Max Heap



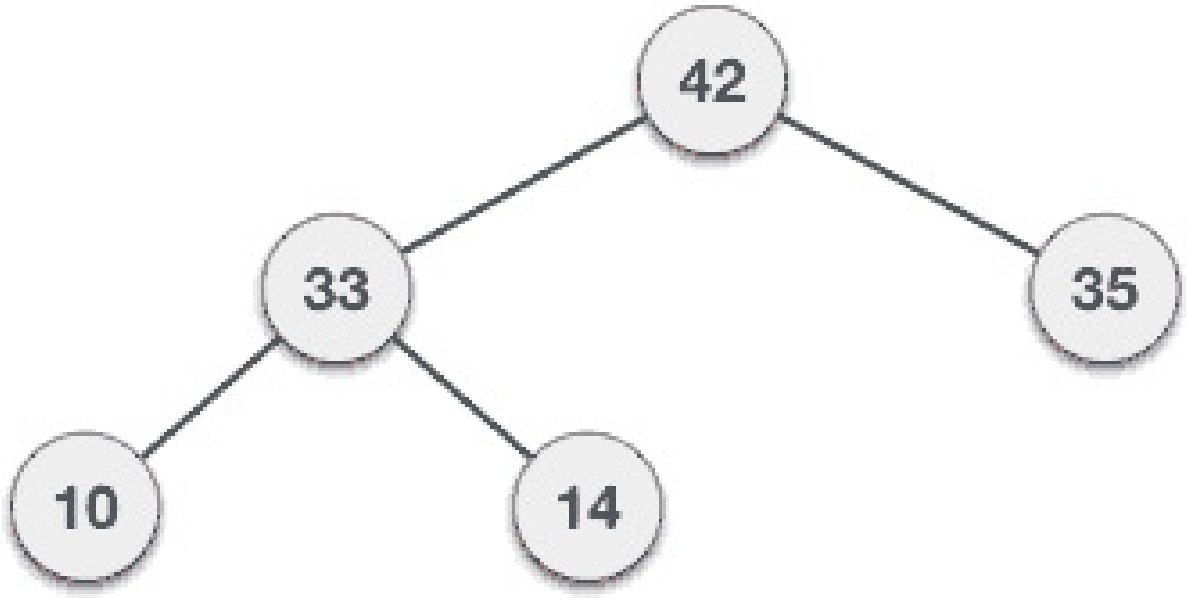
Exemple de construction de Max Heap

Input 35 33 42 10 **14** 19 27 44 26 31

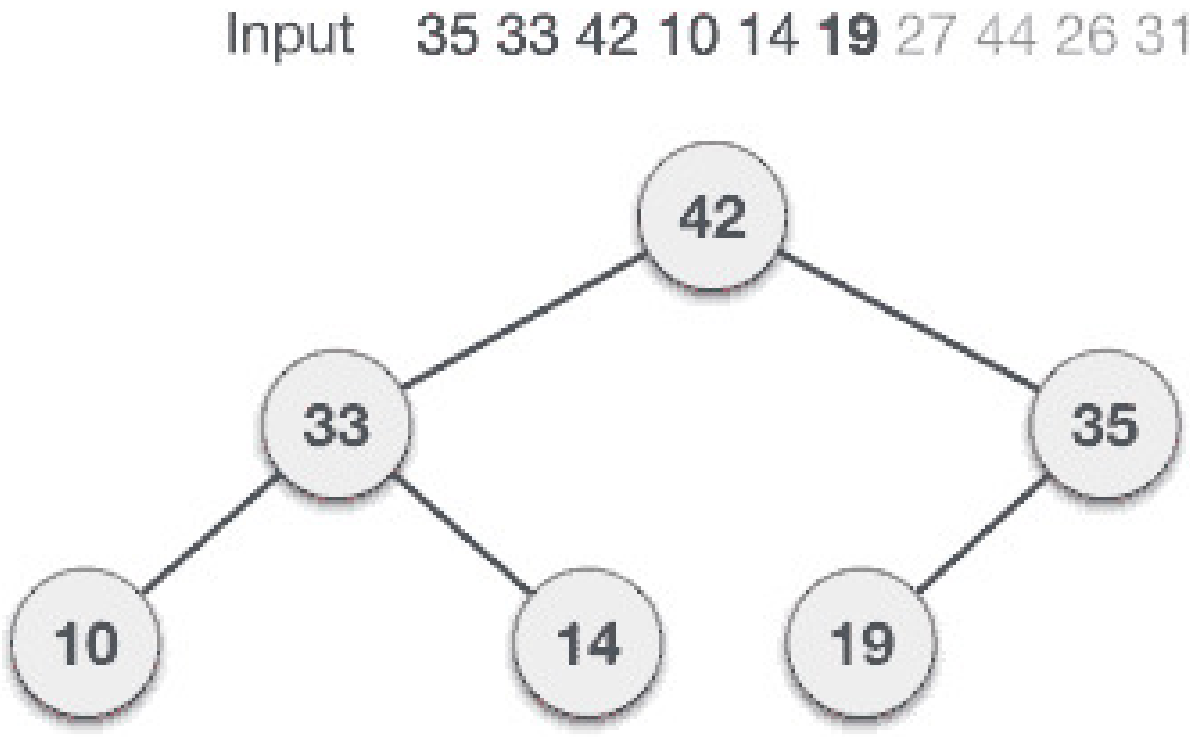


Exemple de construction de Max Heap

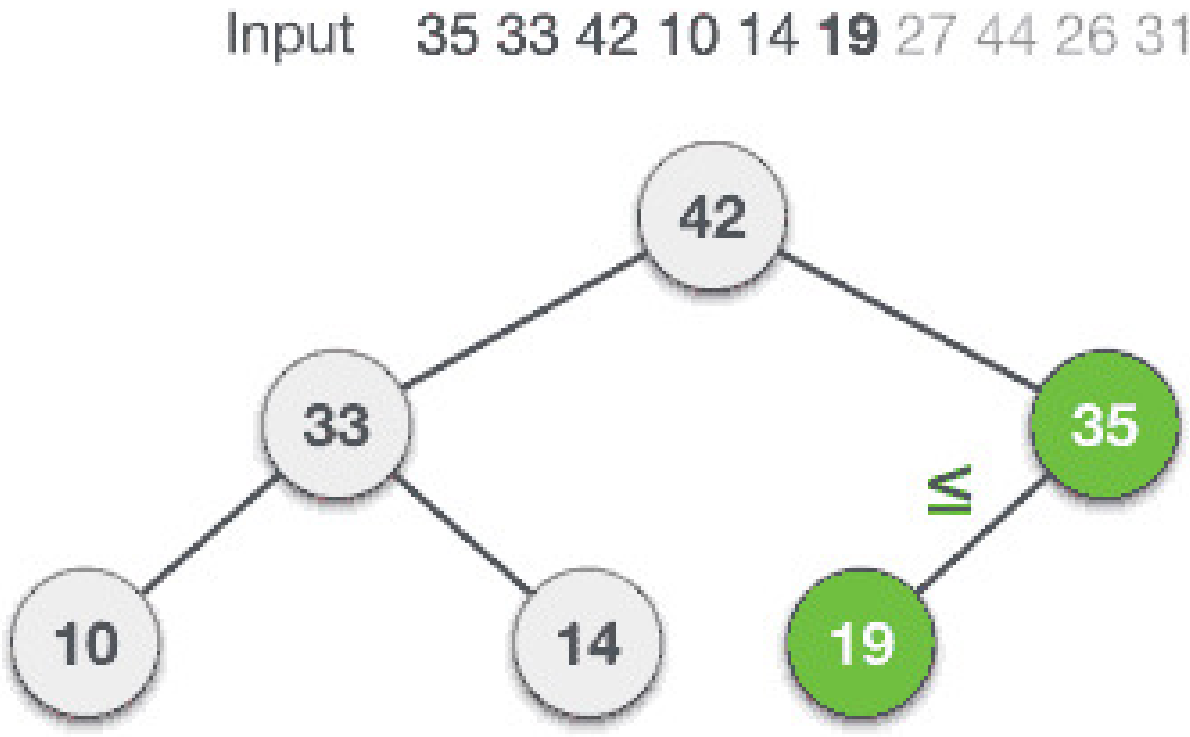
Input 35 33 42 10 14 **19** 27 44 26 31



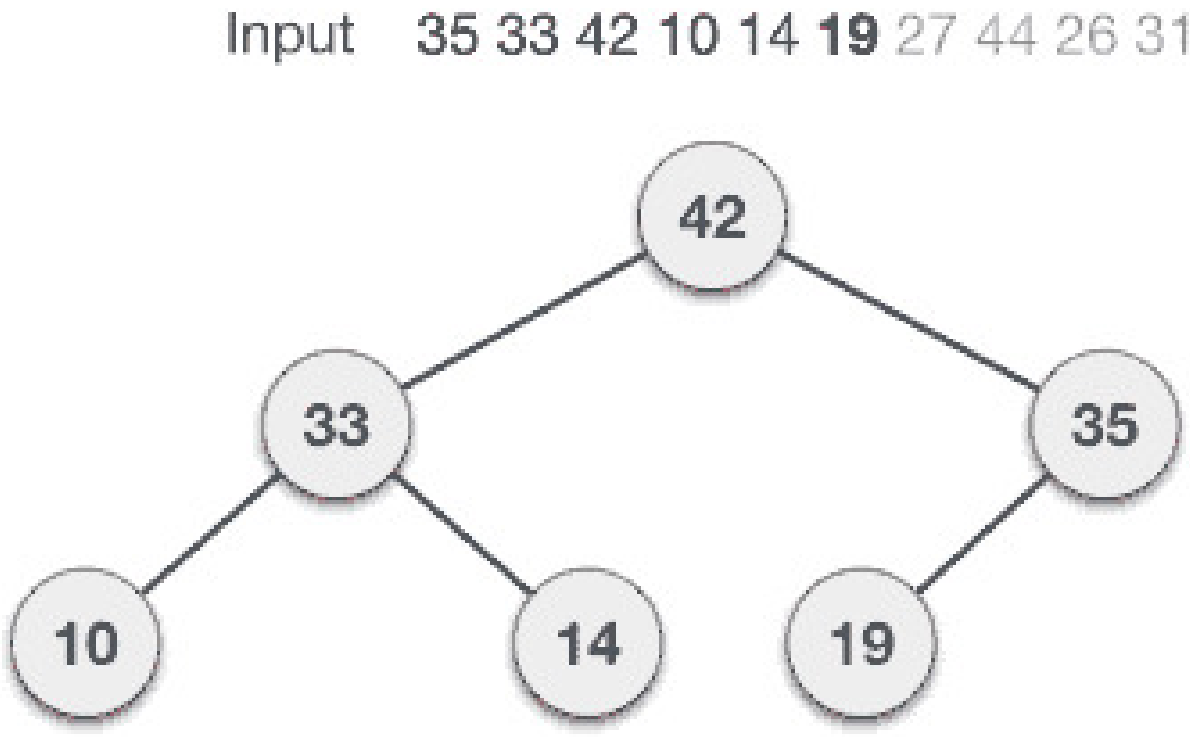
Exemple de construction de Max Heap



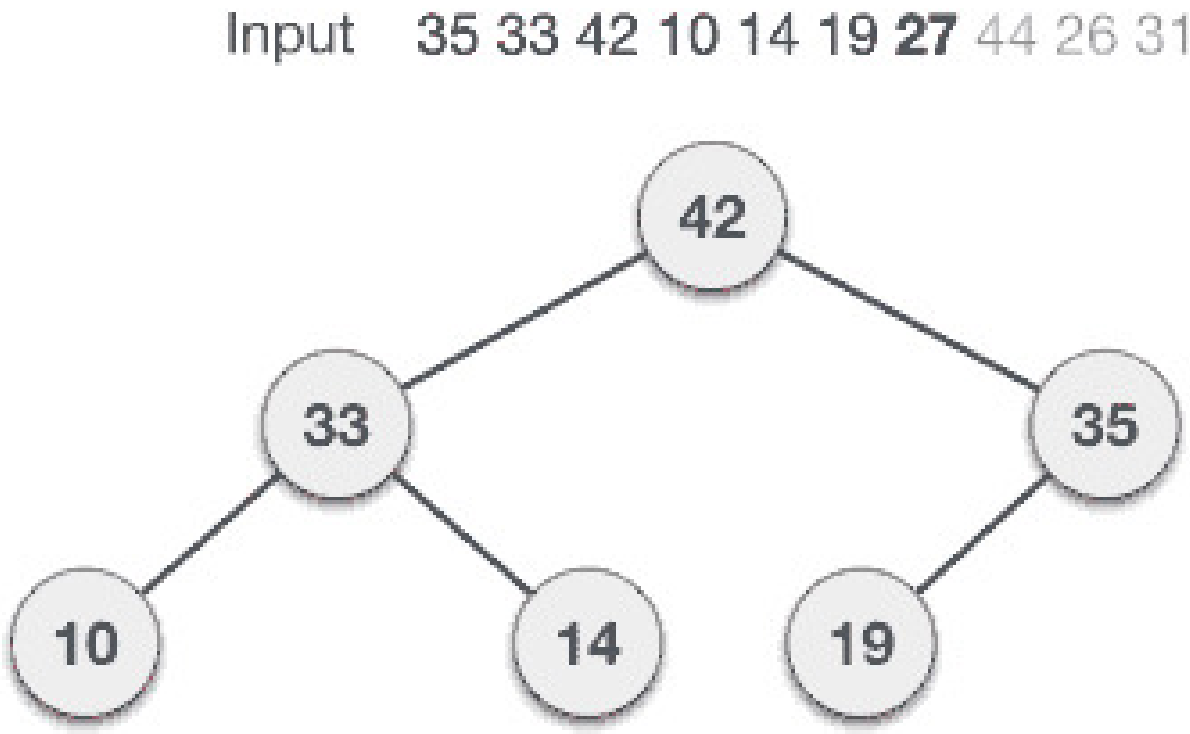
Exemple de construction de Max Heap



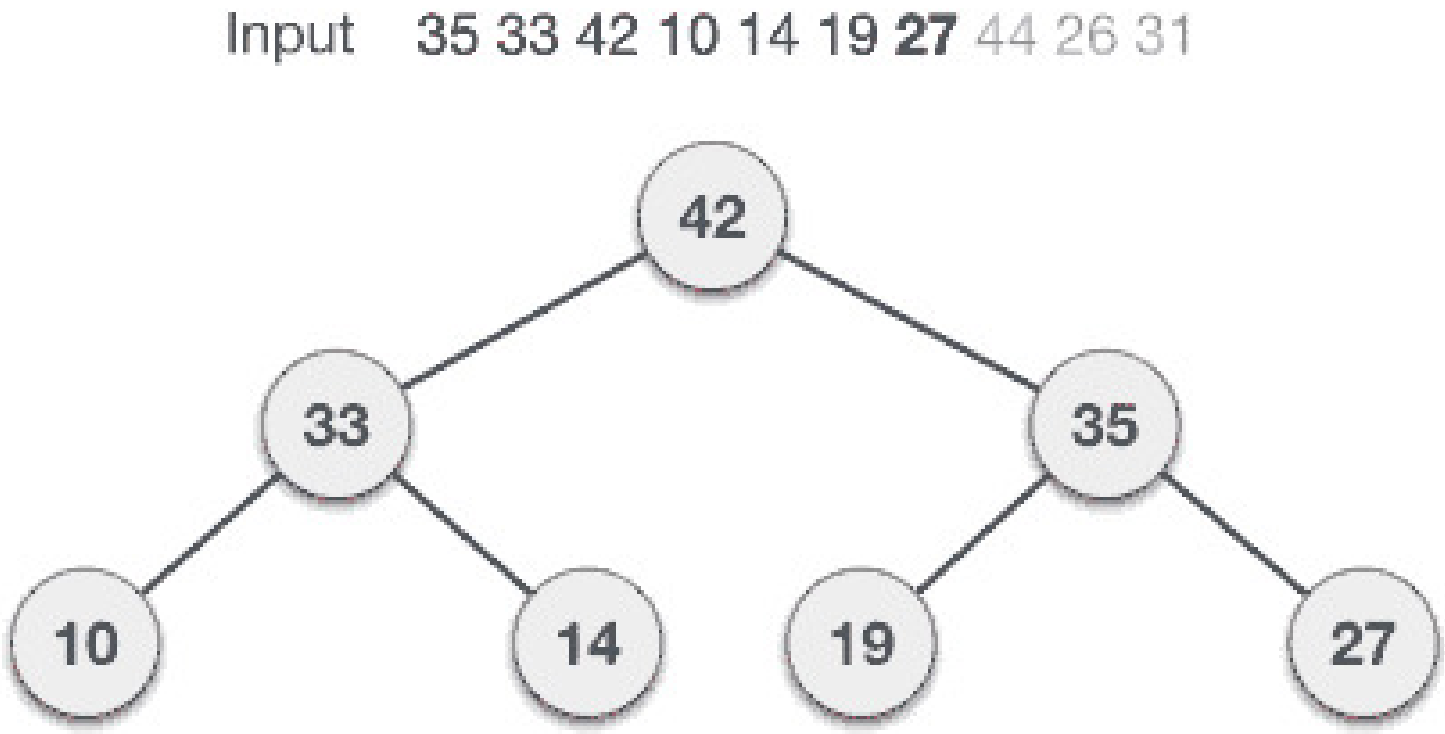
Exemple de construction de Max Heap



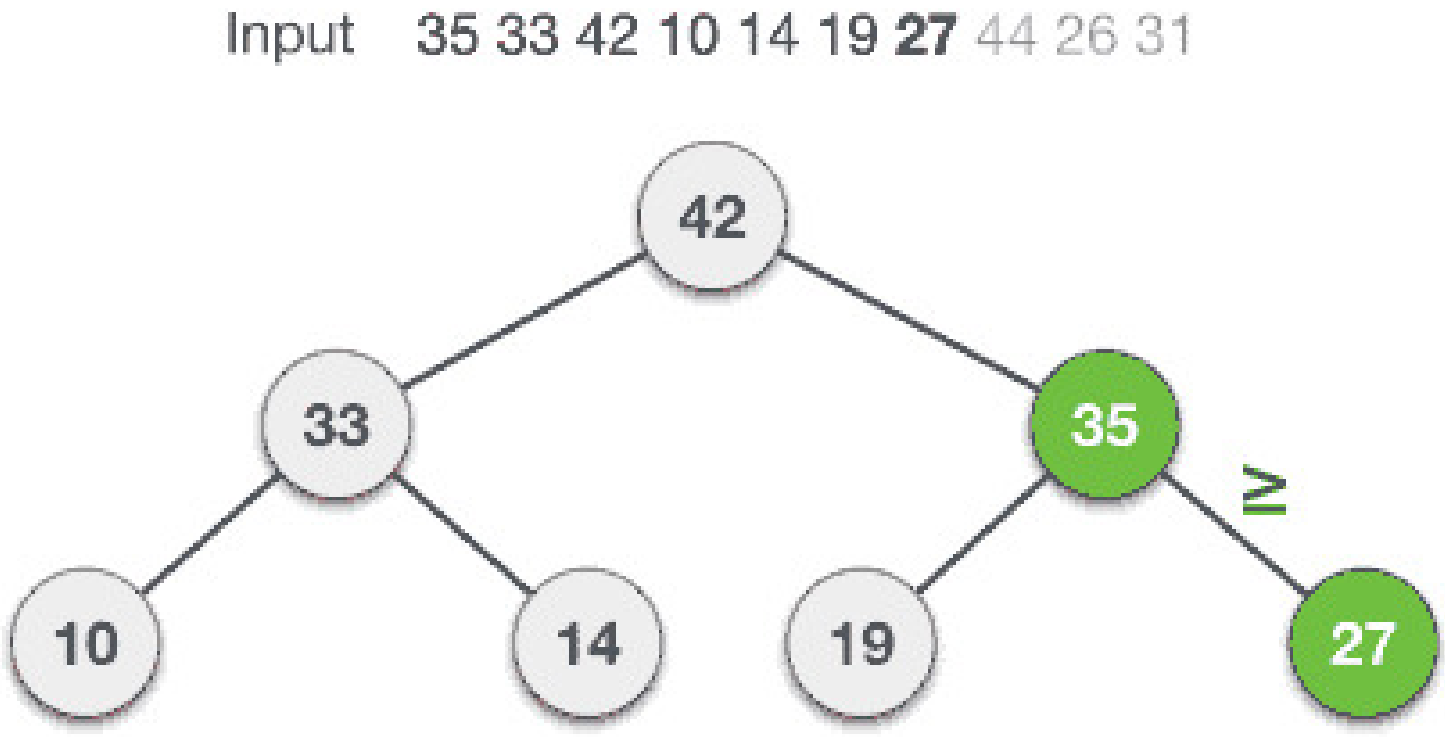
Exemple de construction de Max Heap



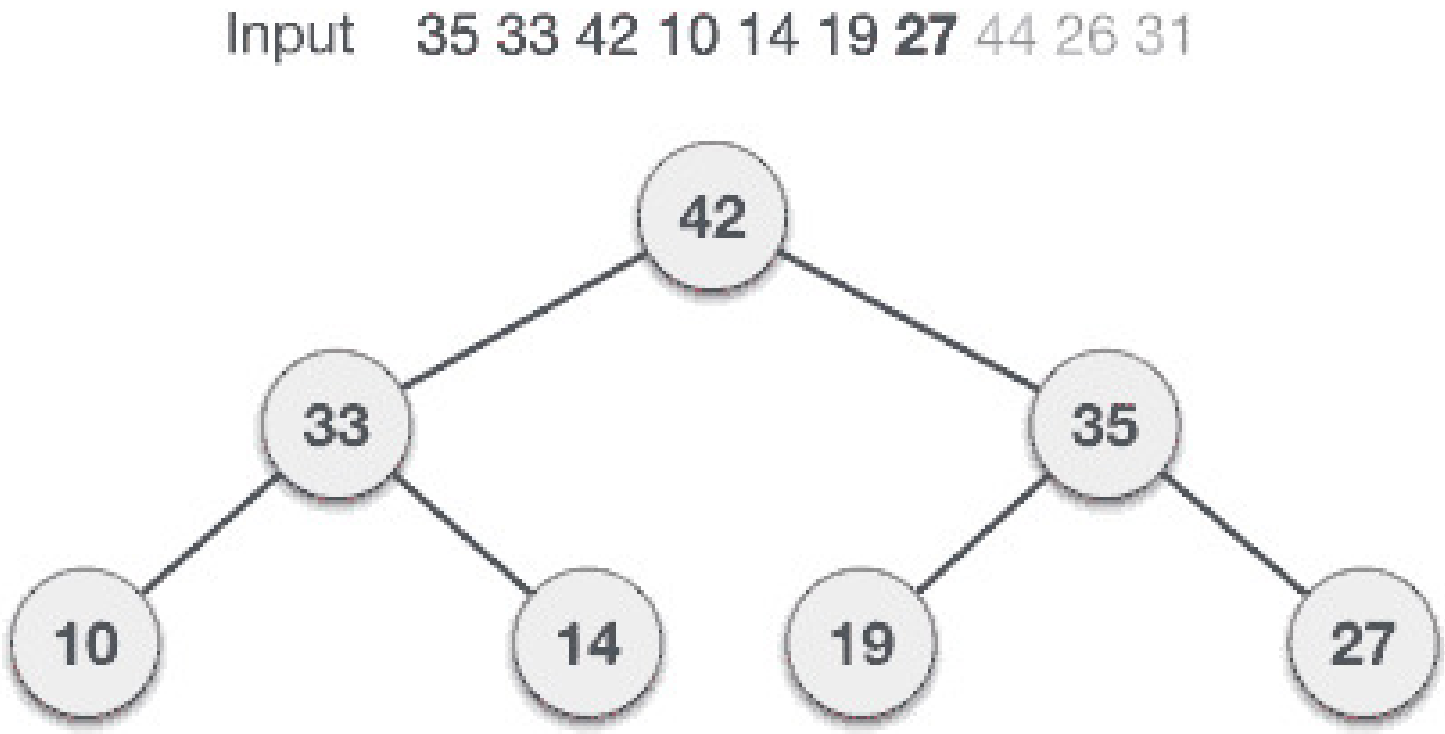
Exemple de construction de Max Heap



Exemple de construction de Max Heap

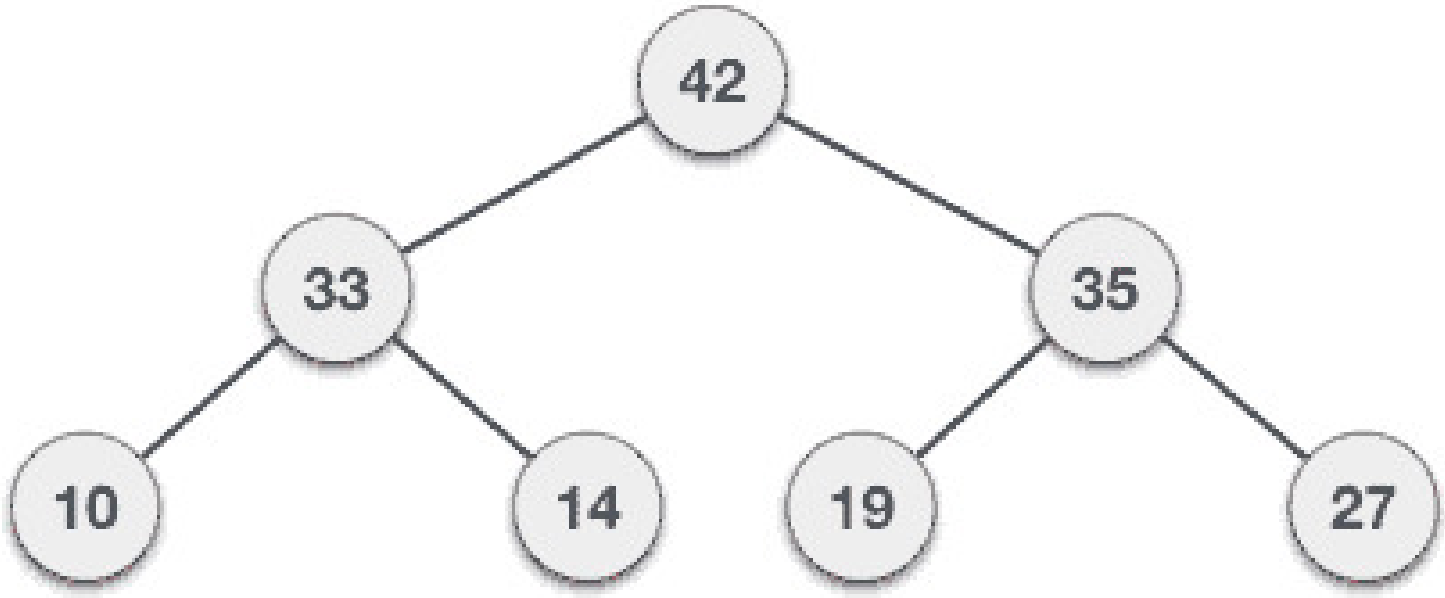


Exemple de construction de Max Heap

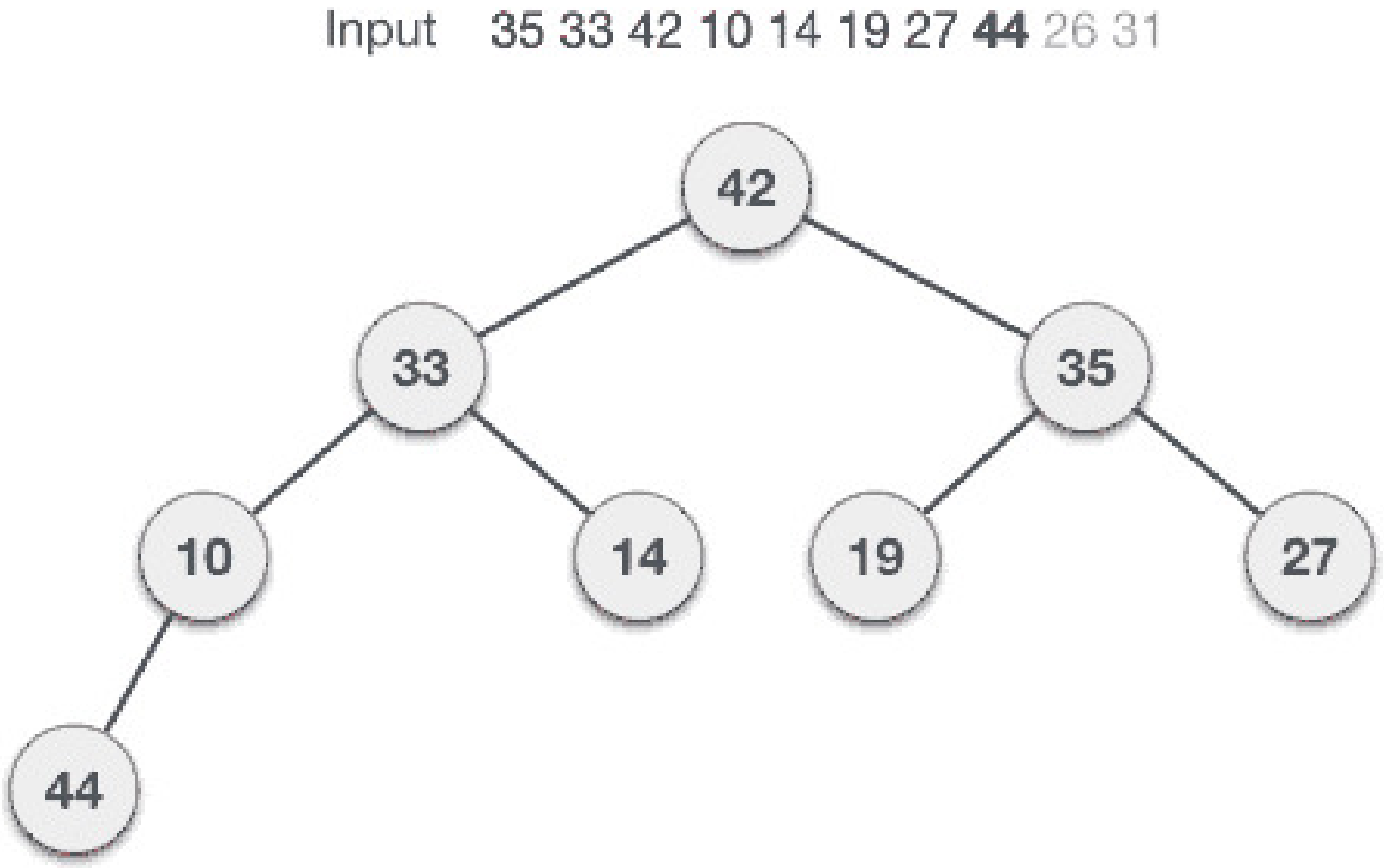


Exemple de construction de Max Heap

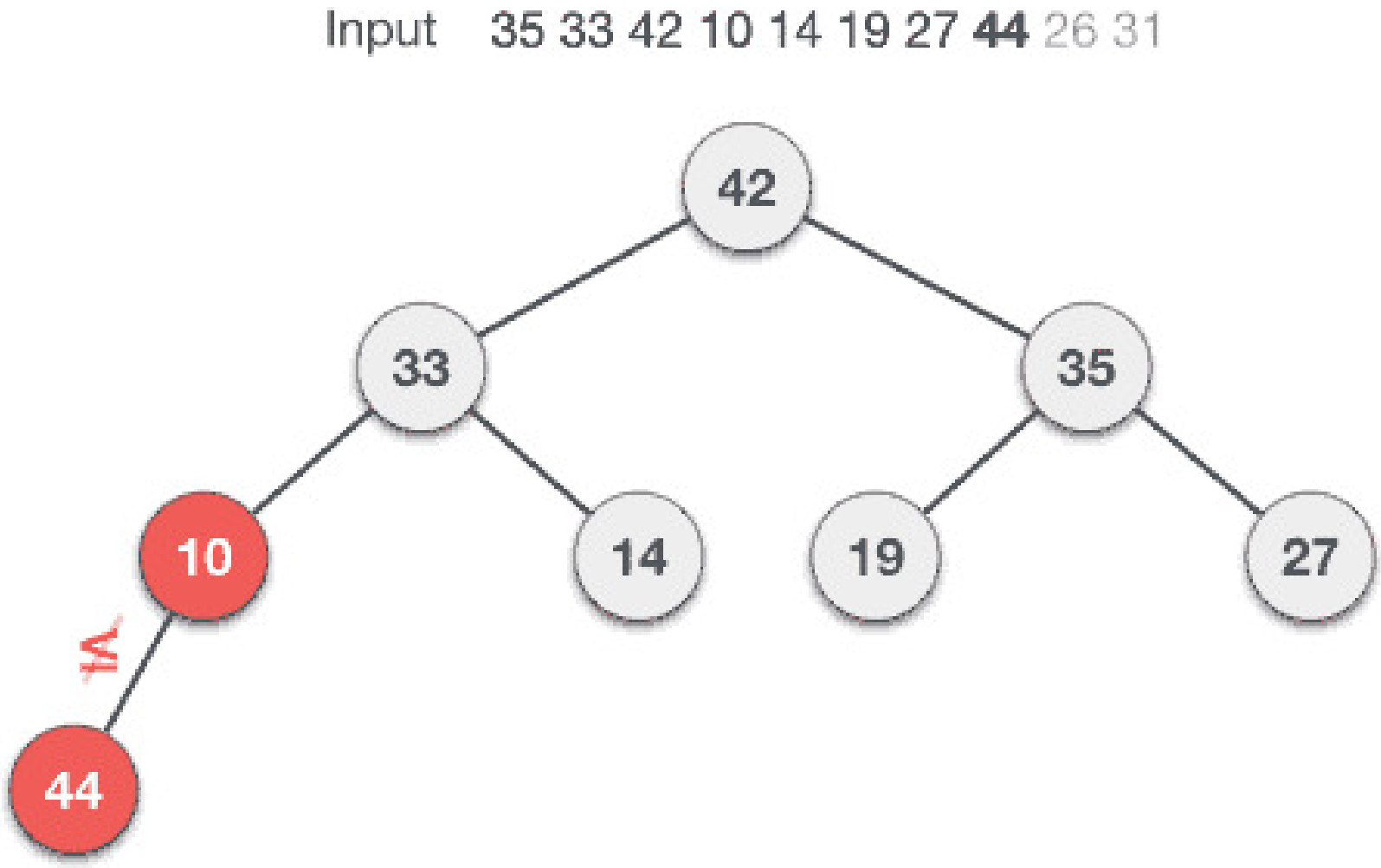
Input 35 33 42 10 14 19 27 **44** 26 31



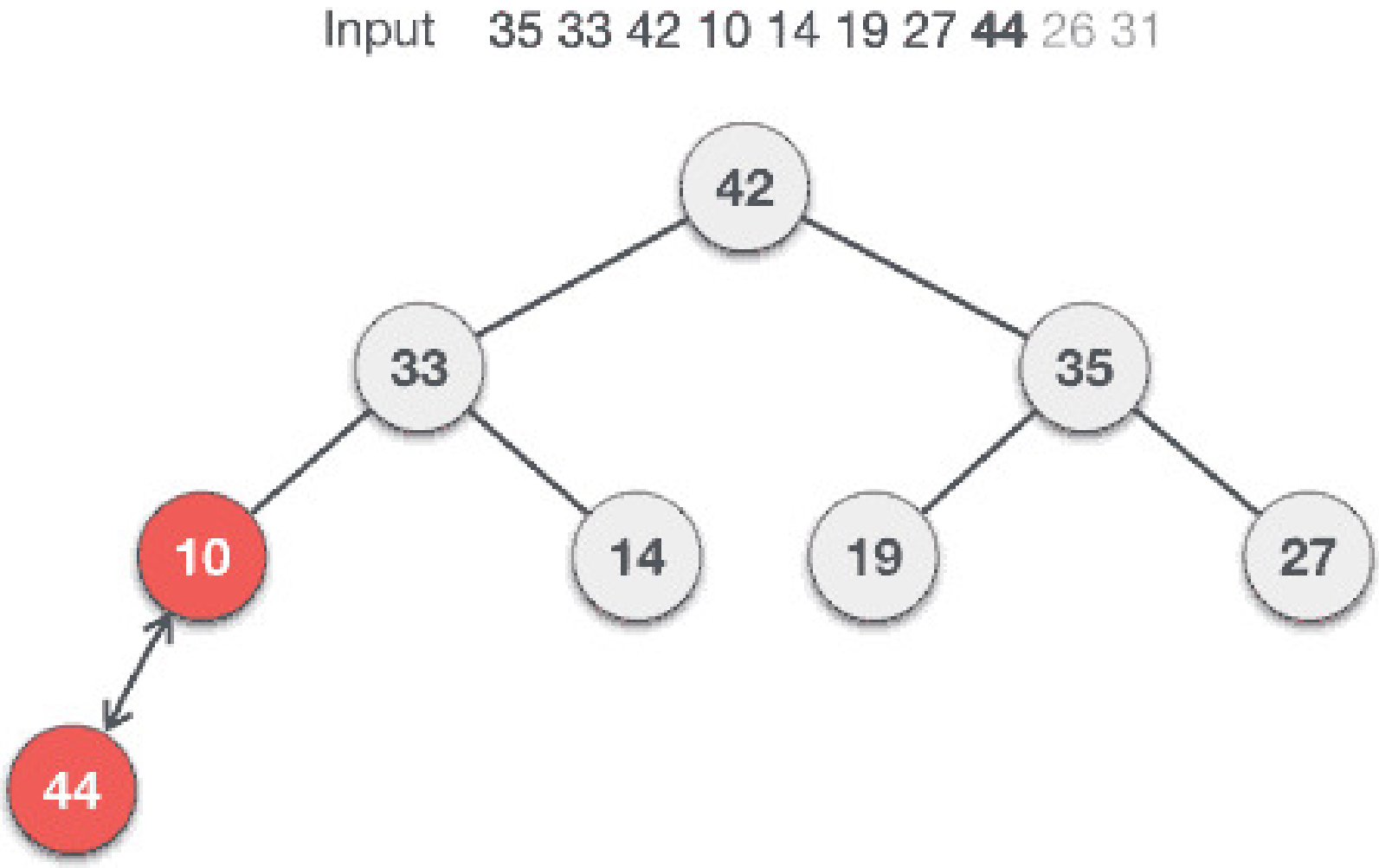
Exemple de construction de Max Heap



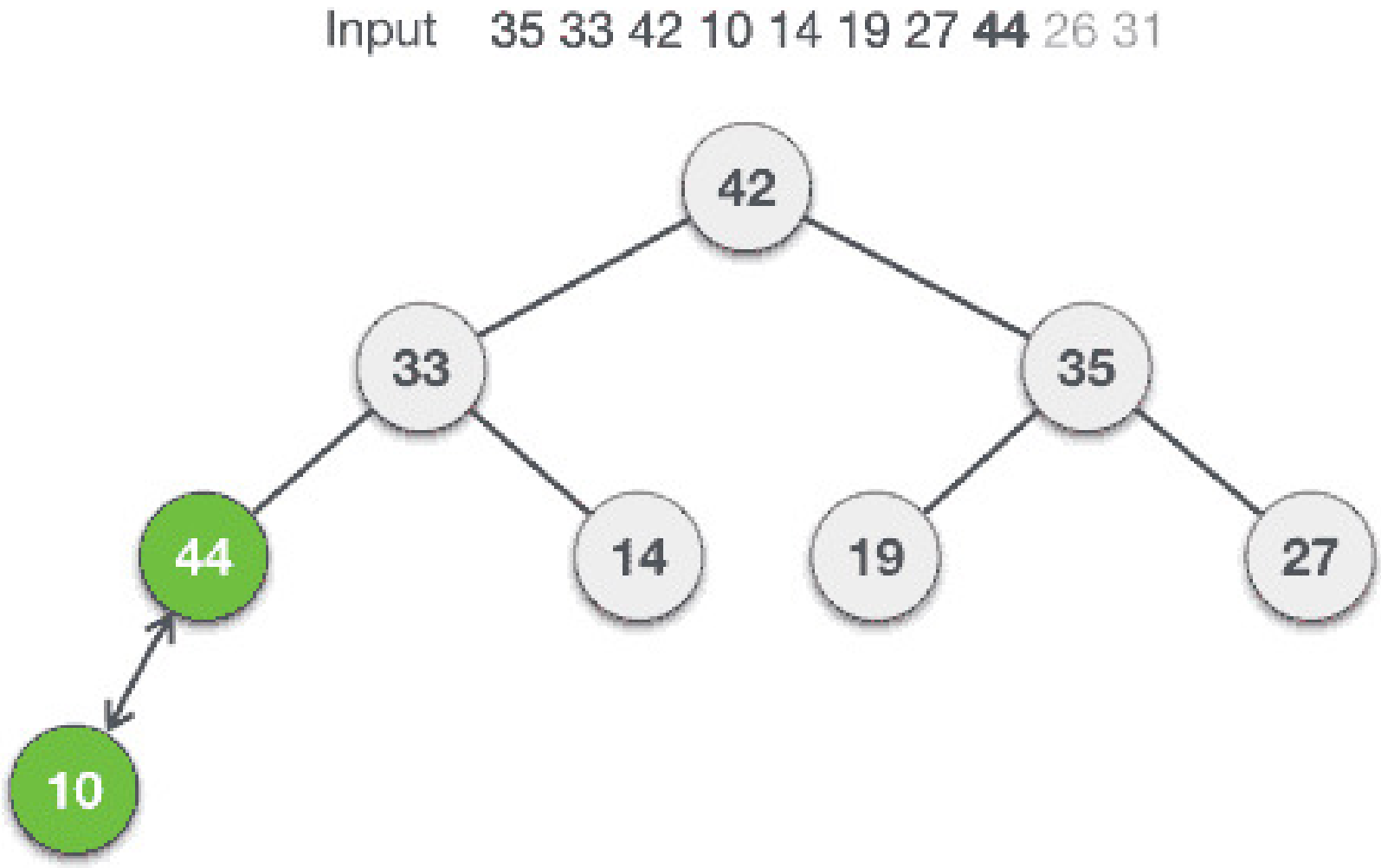
Exemple de construction de Max Heap



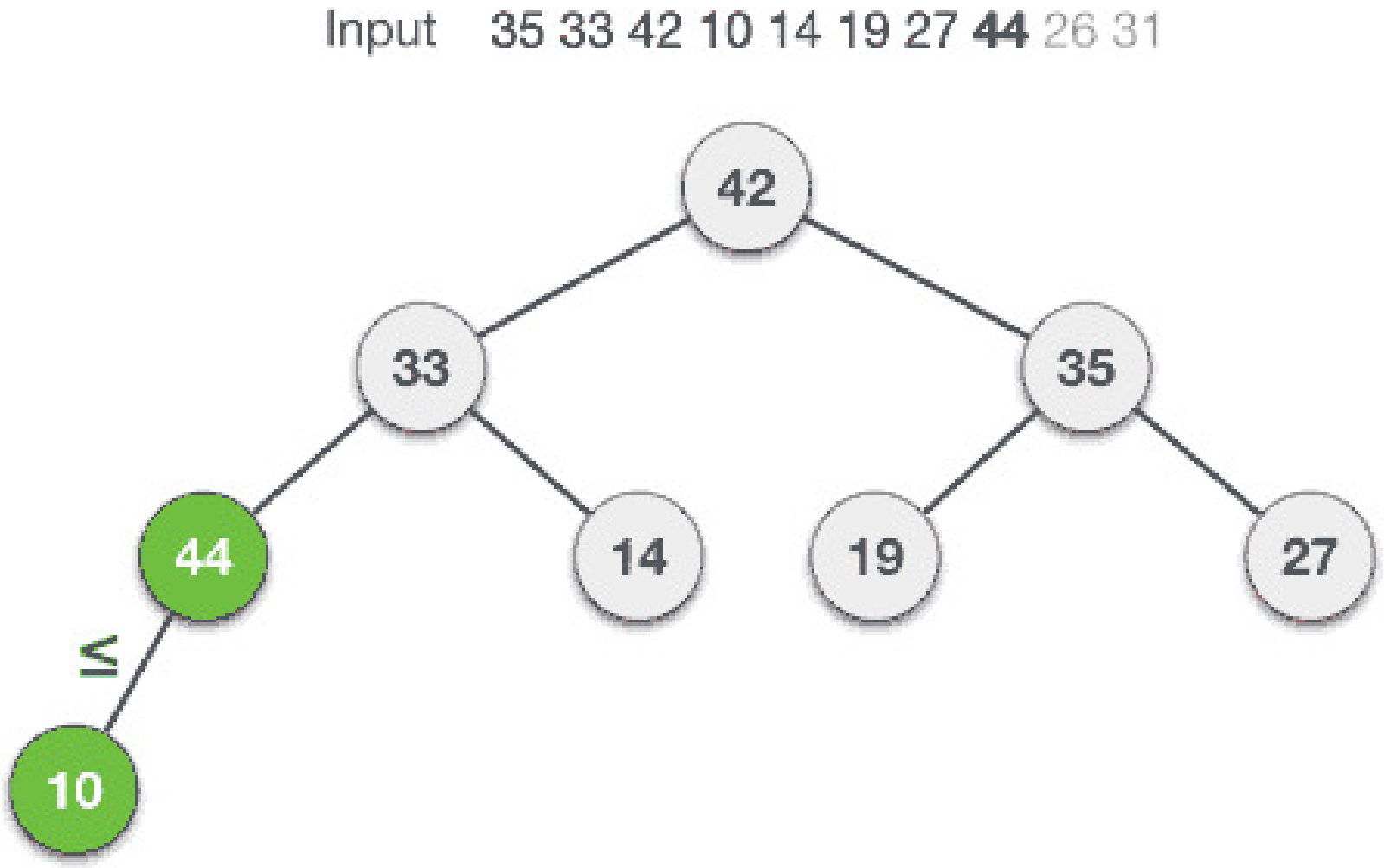
Exemple de construction de Max Heap



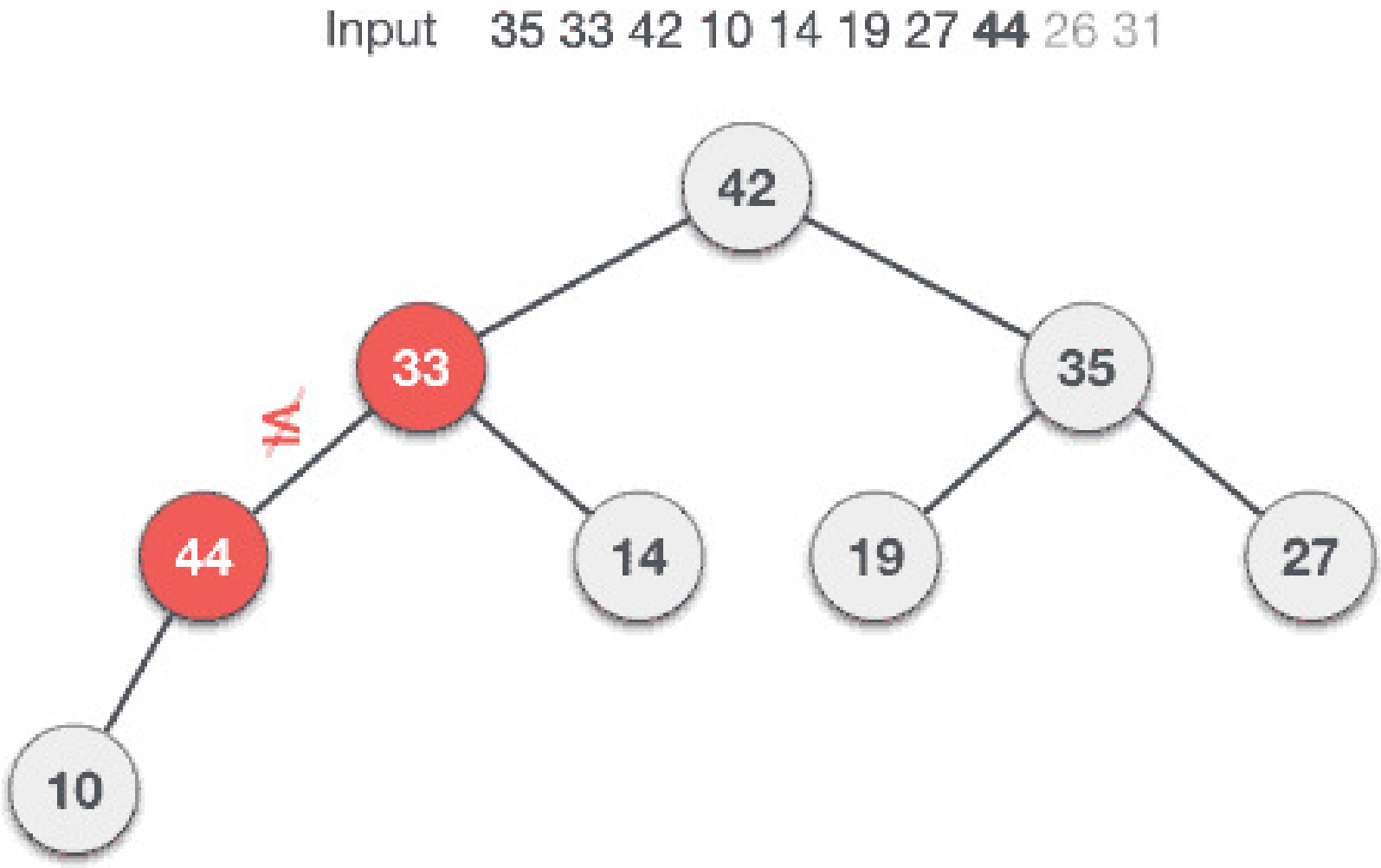
Exemple de construction de Max Heap



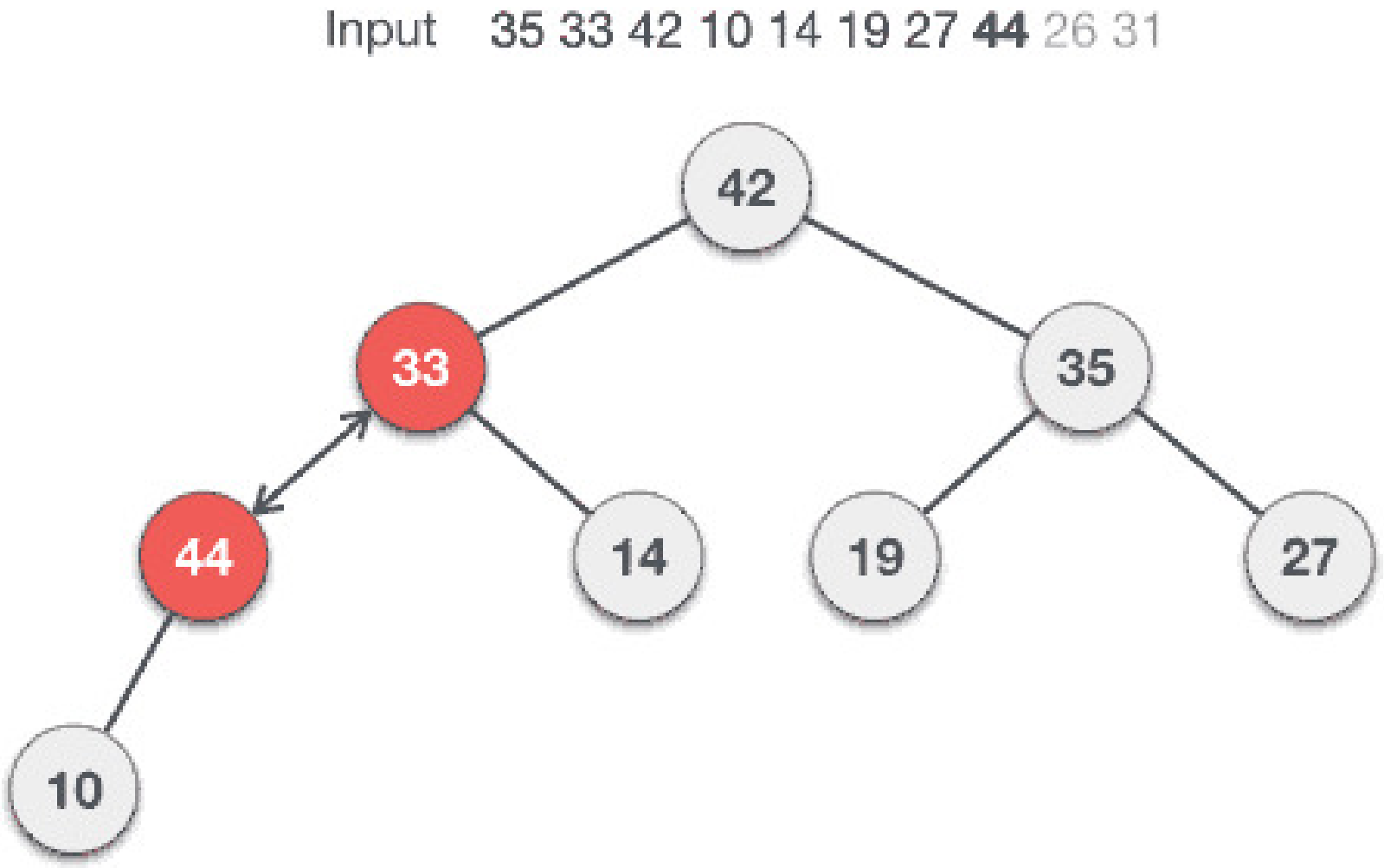
Exemple de construction de Max Heap



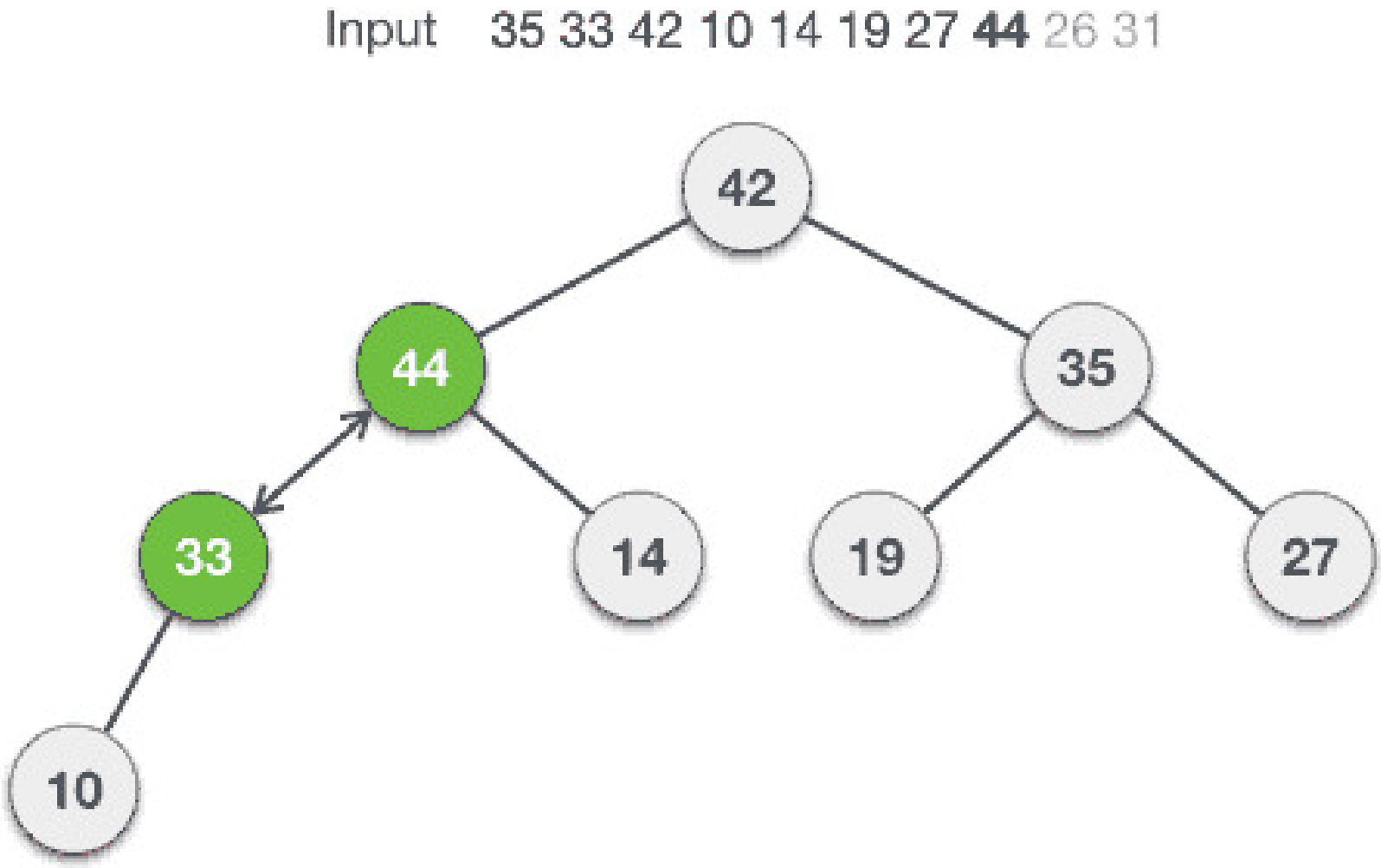
Exemple de construction de Max Heap



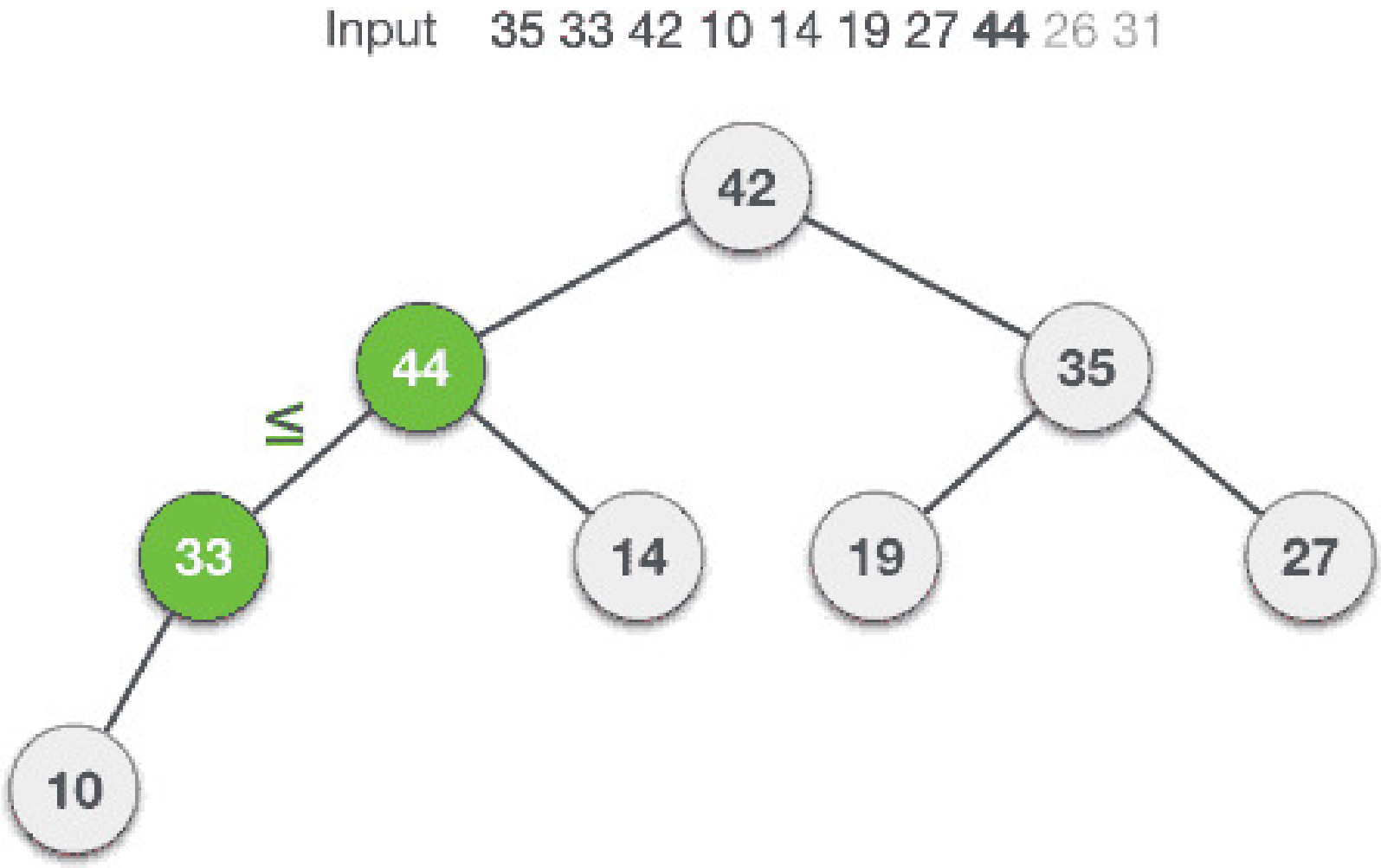
Exemple de construction de Max Heap



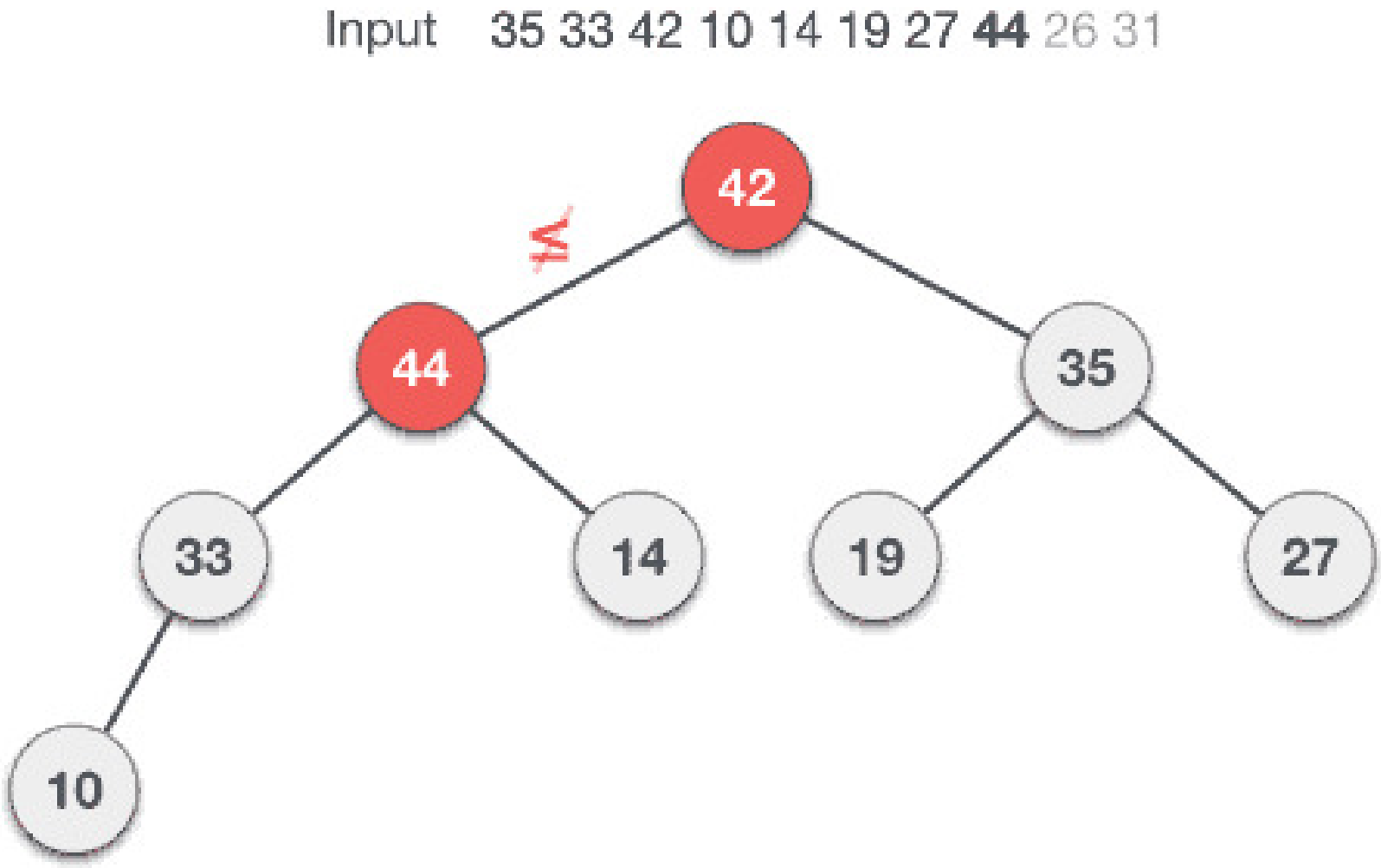
Exemple de construction de Max Heap



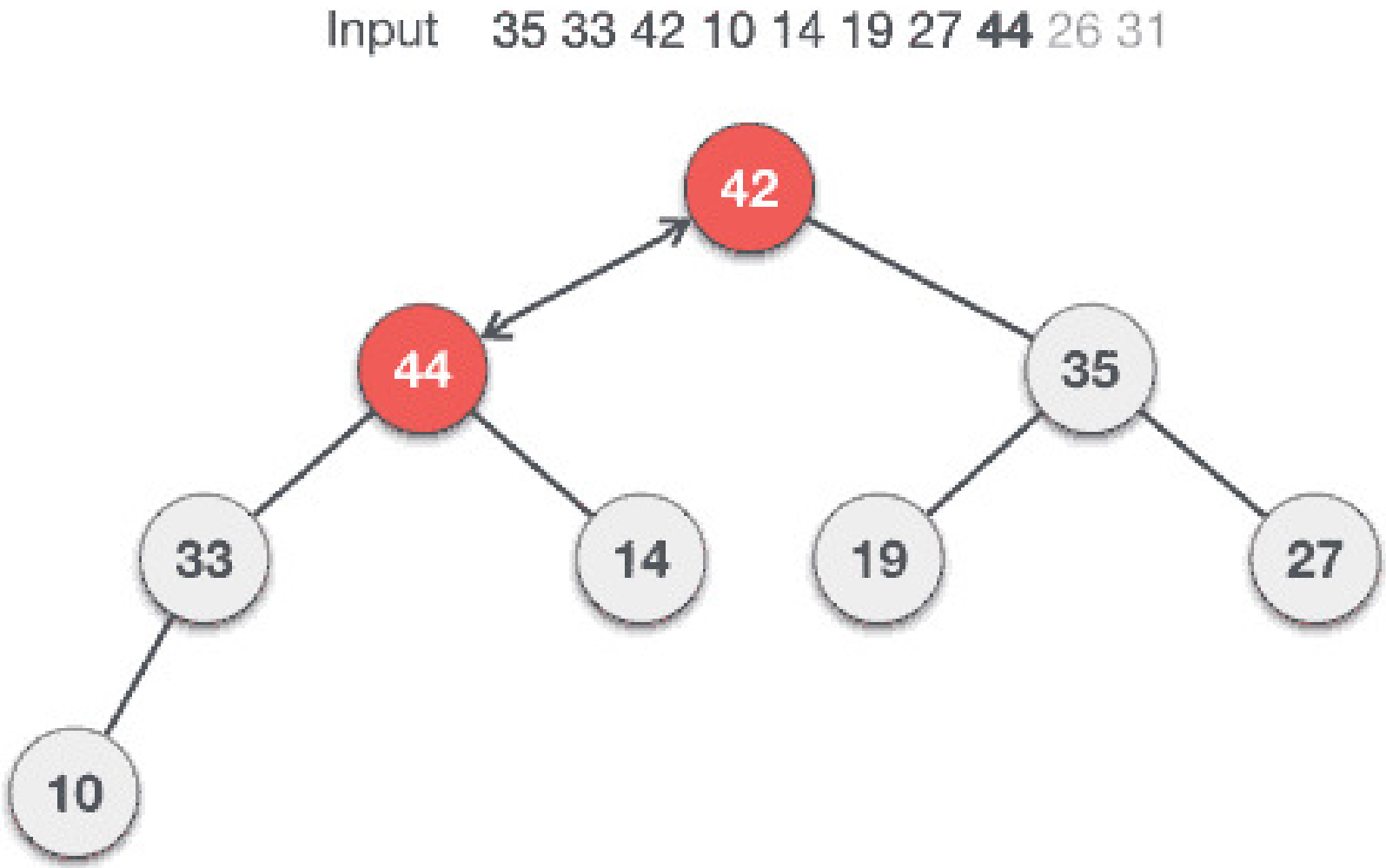
Exemple de construction de Max Heap



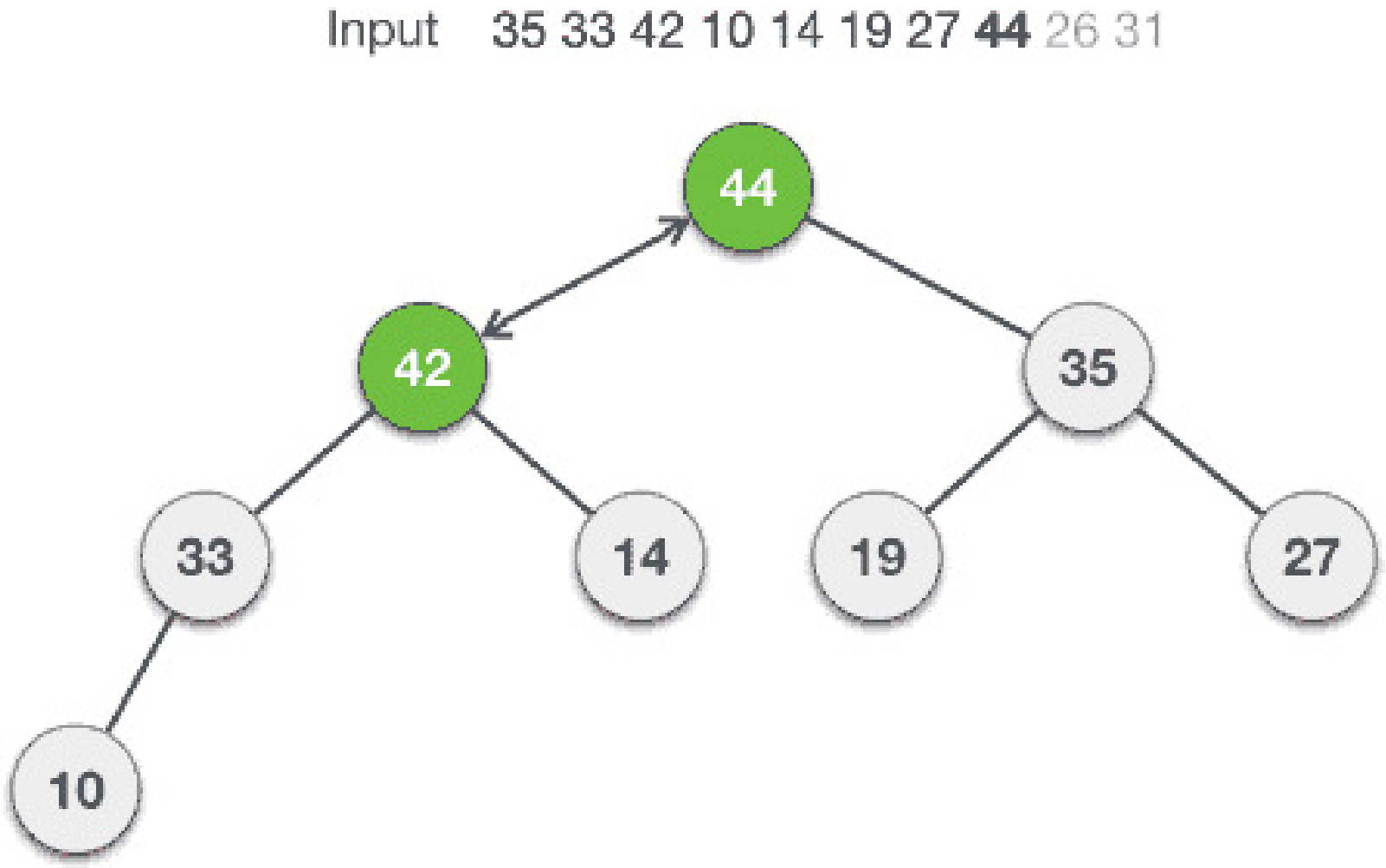
Exemple de construction de Max Heap



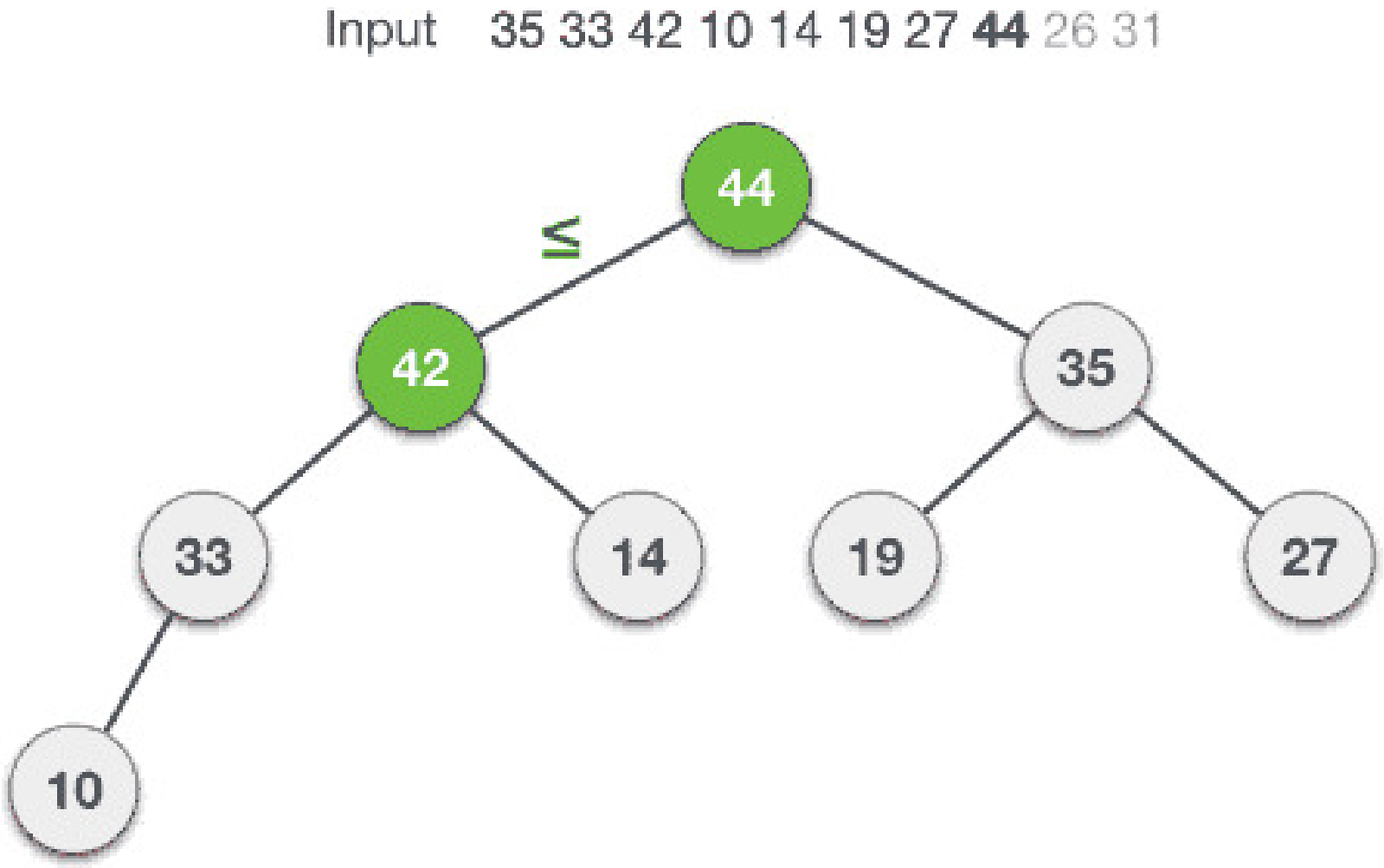
Exemple de construction de Max Heap



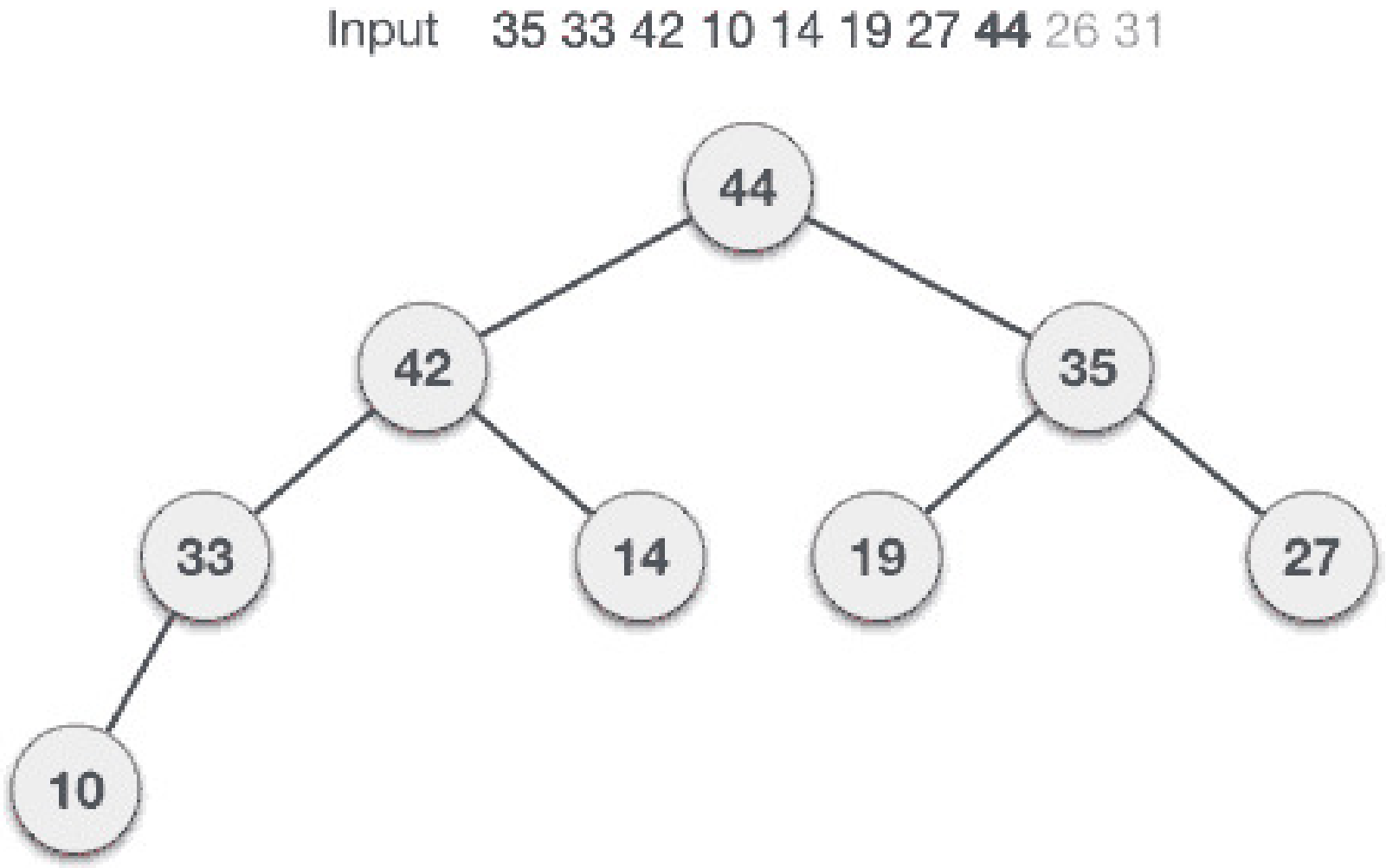
Exemple de construction de Max Heap



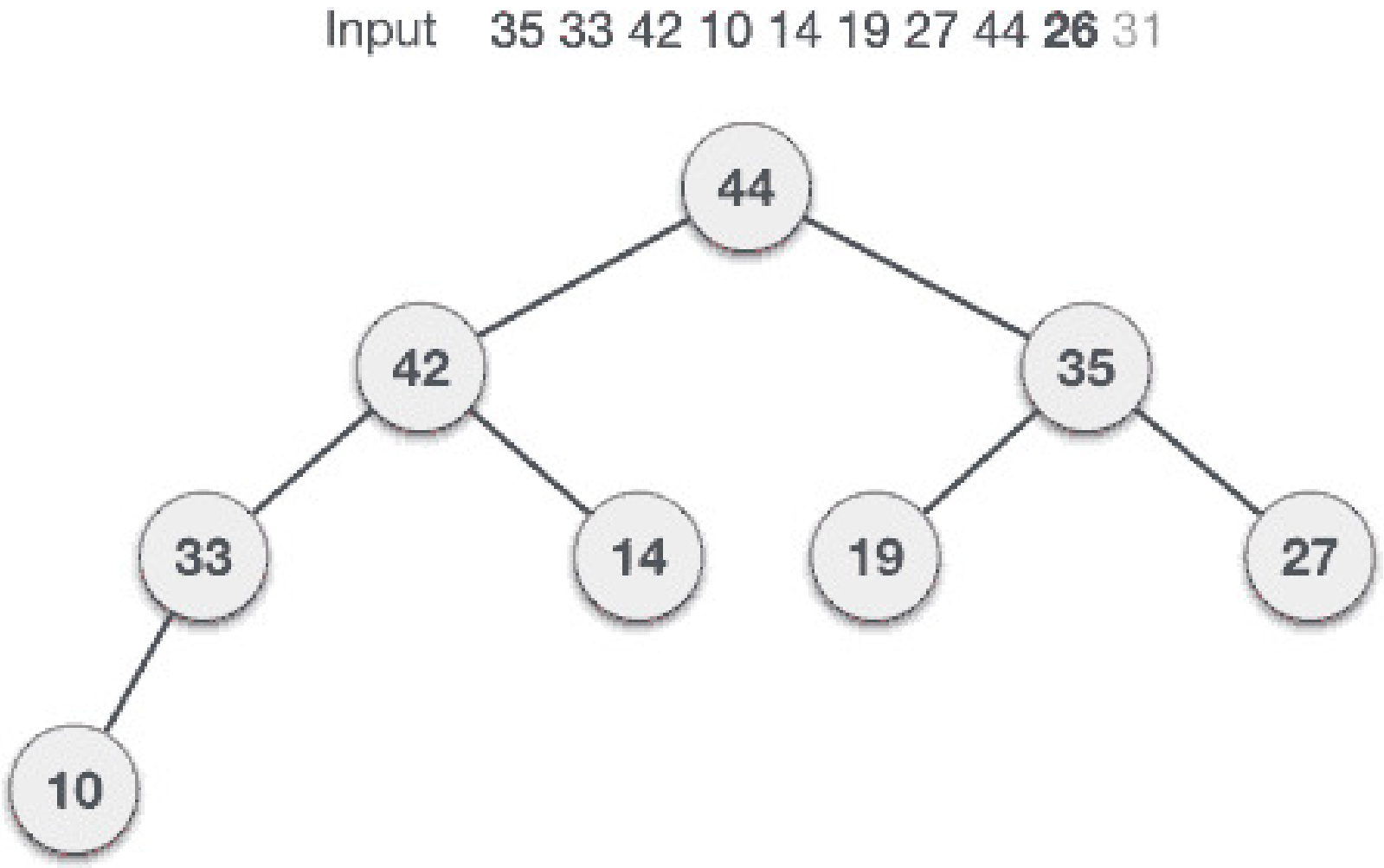
Exemple de construction de Max Heap



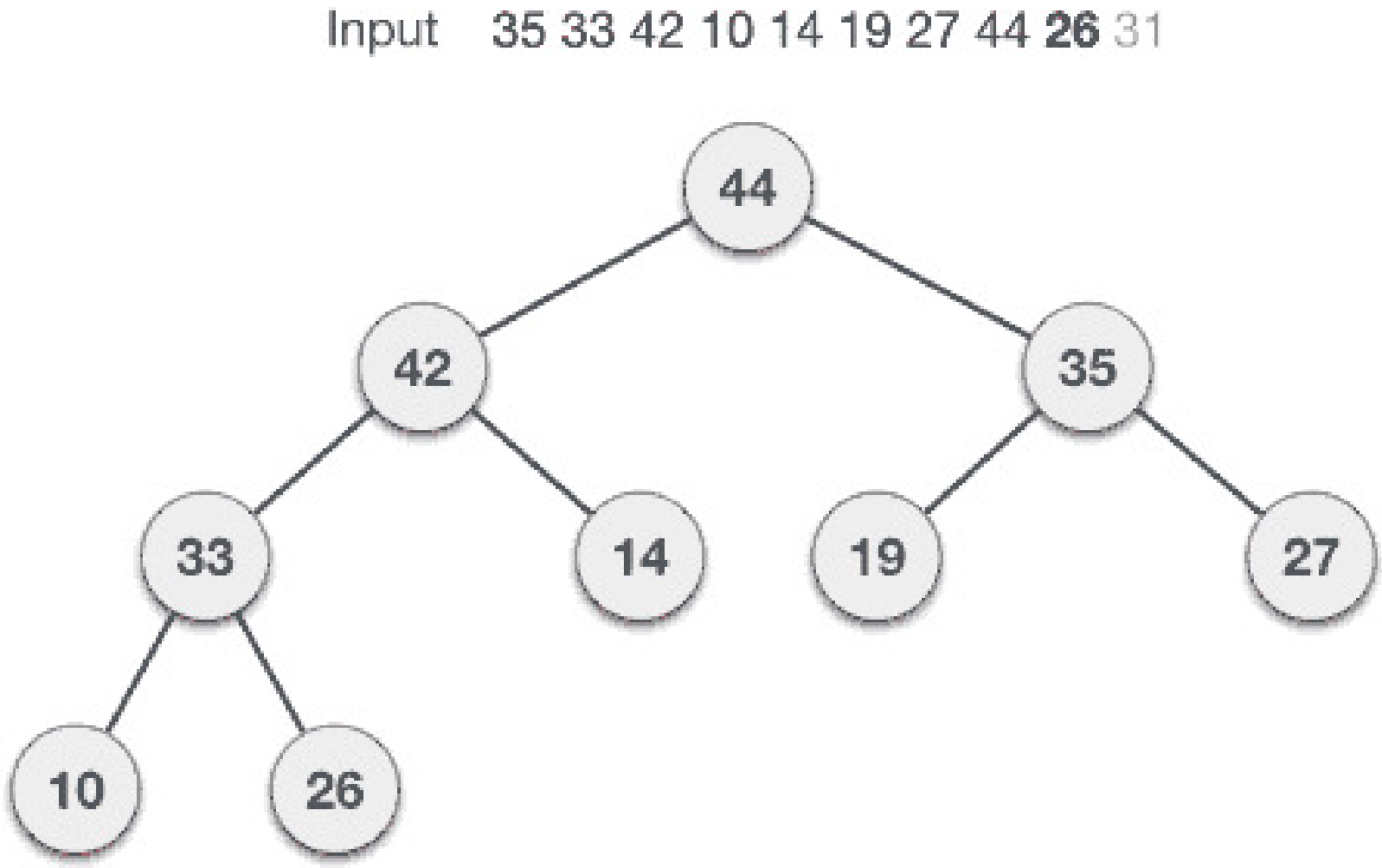
Exemple de construction de Max Heap



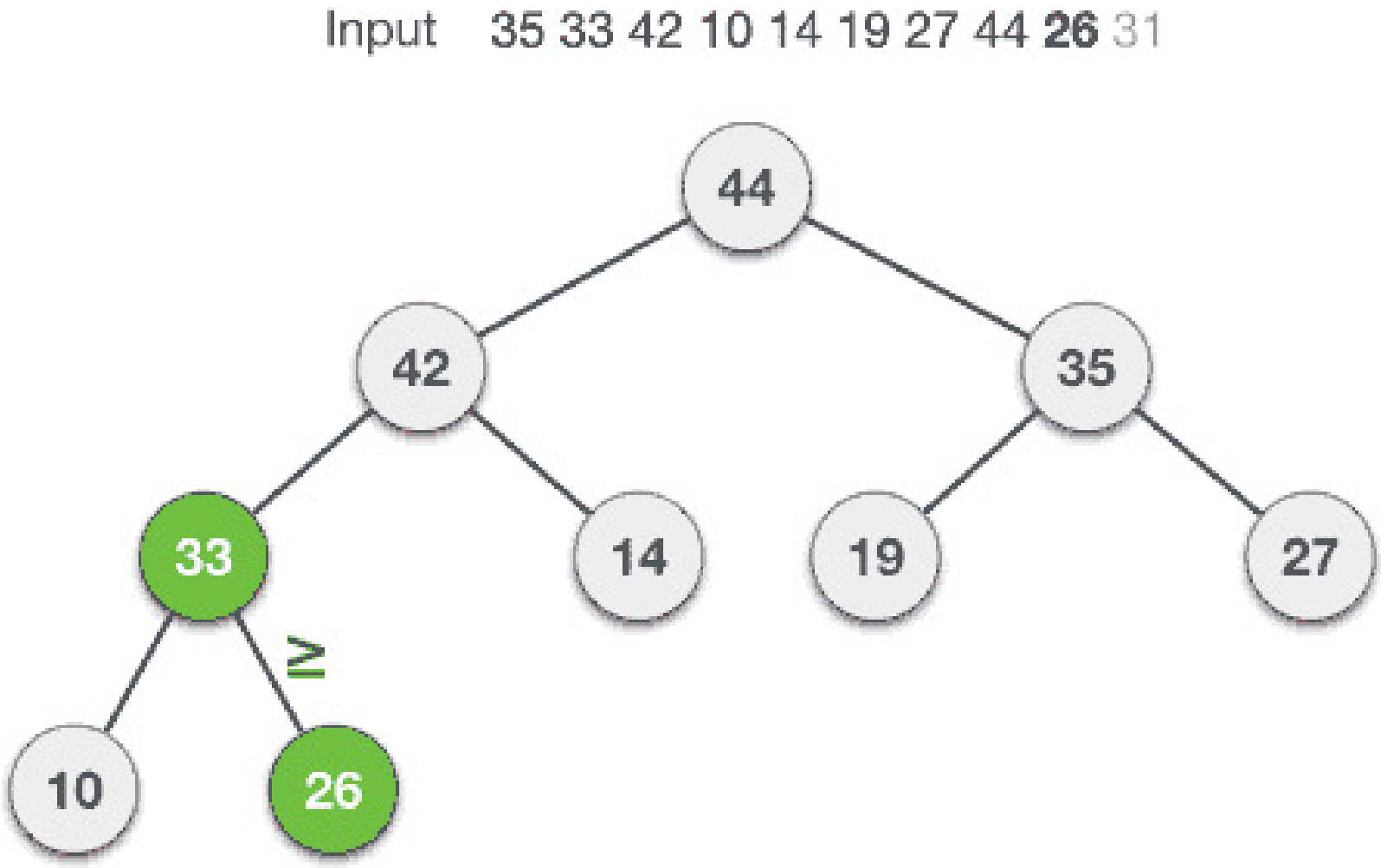
Exemple de construction de Max Heap



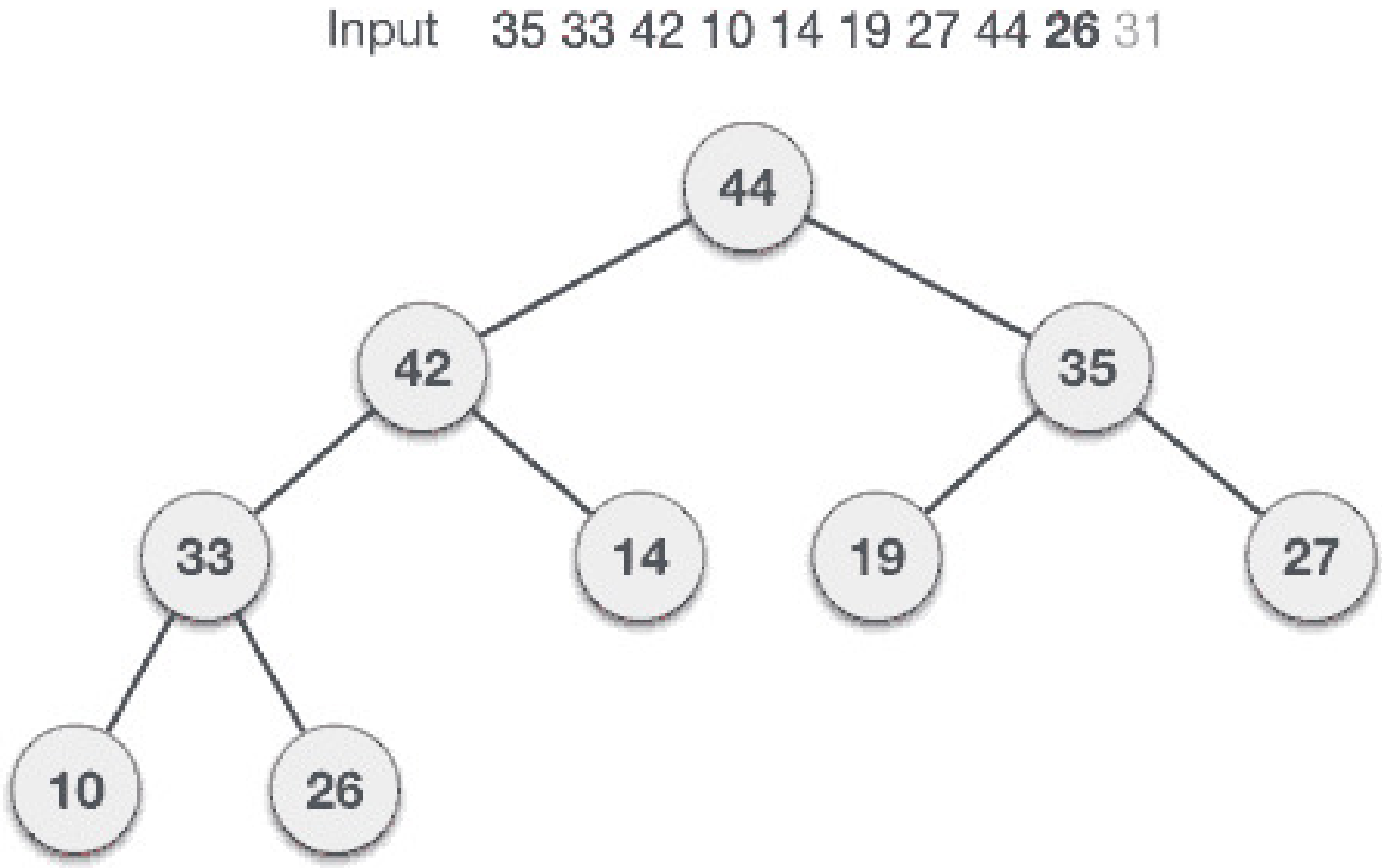
Exemple de construction de Max Heap



Exemple de construction de Max Heap

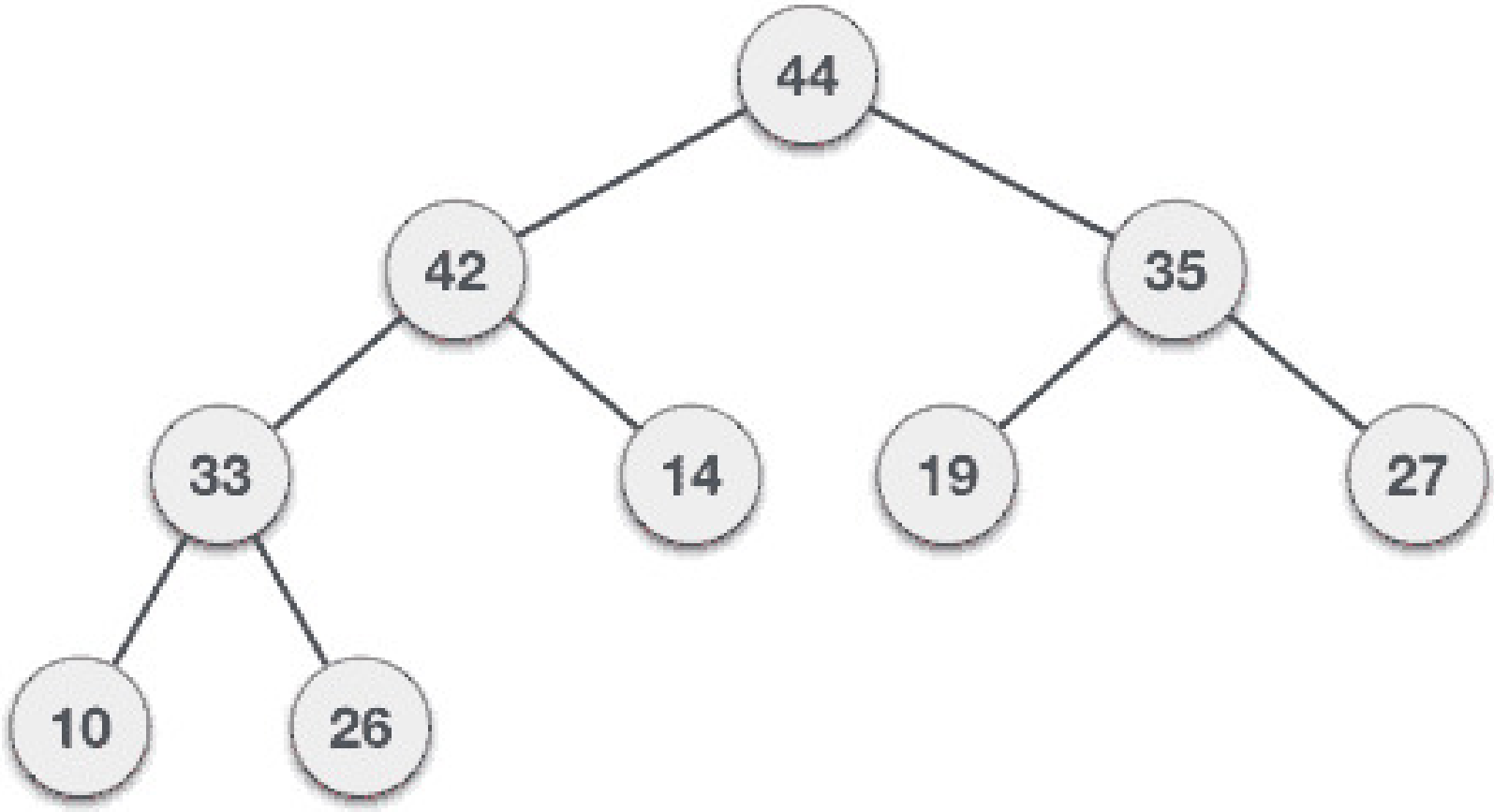


Exemple de construction de Max Heap



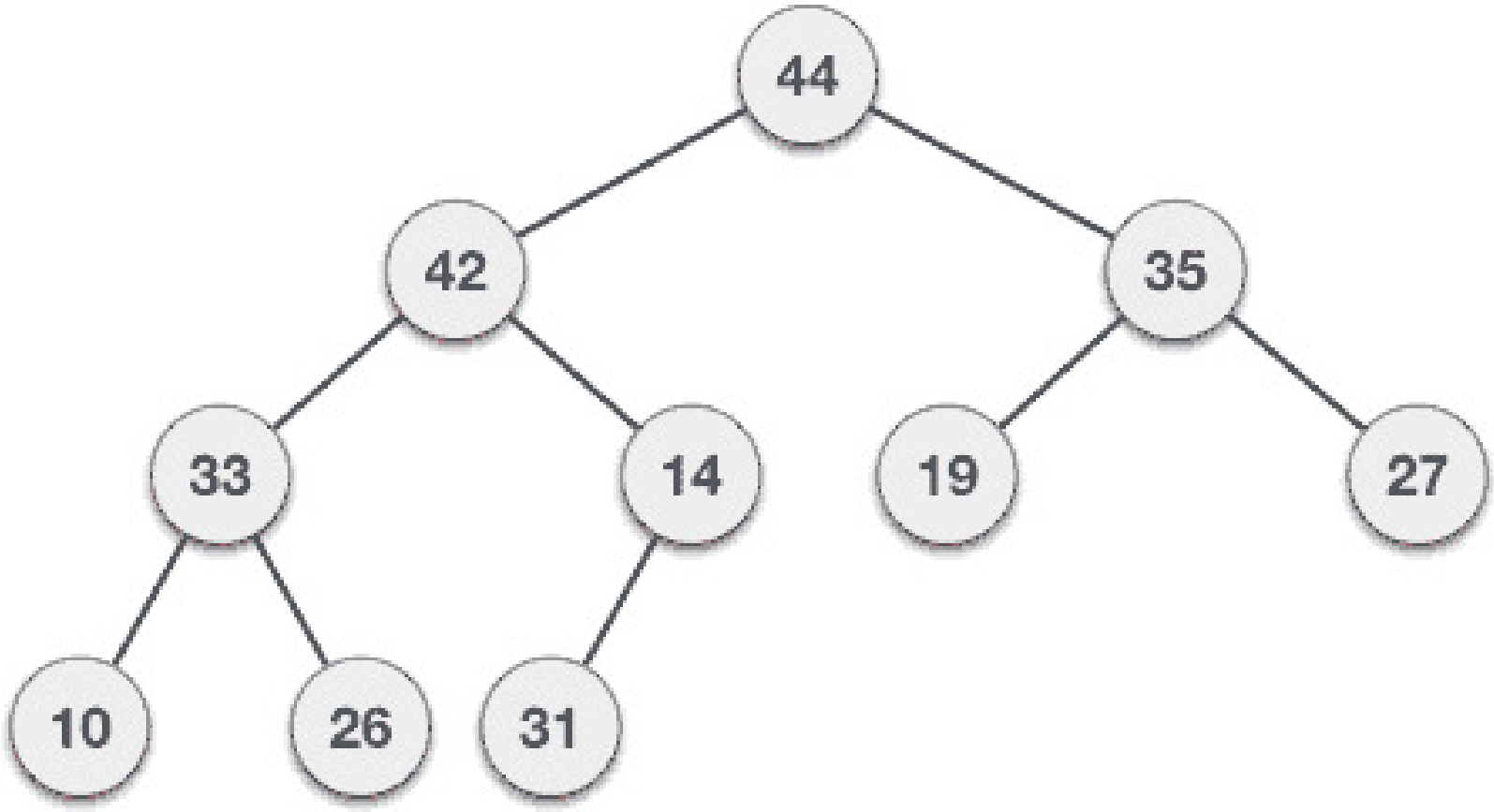
Exemple de construction de Max Heap

Input 35 33 42 10 14 19 27 44 26 **31**

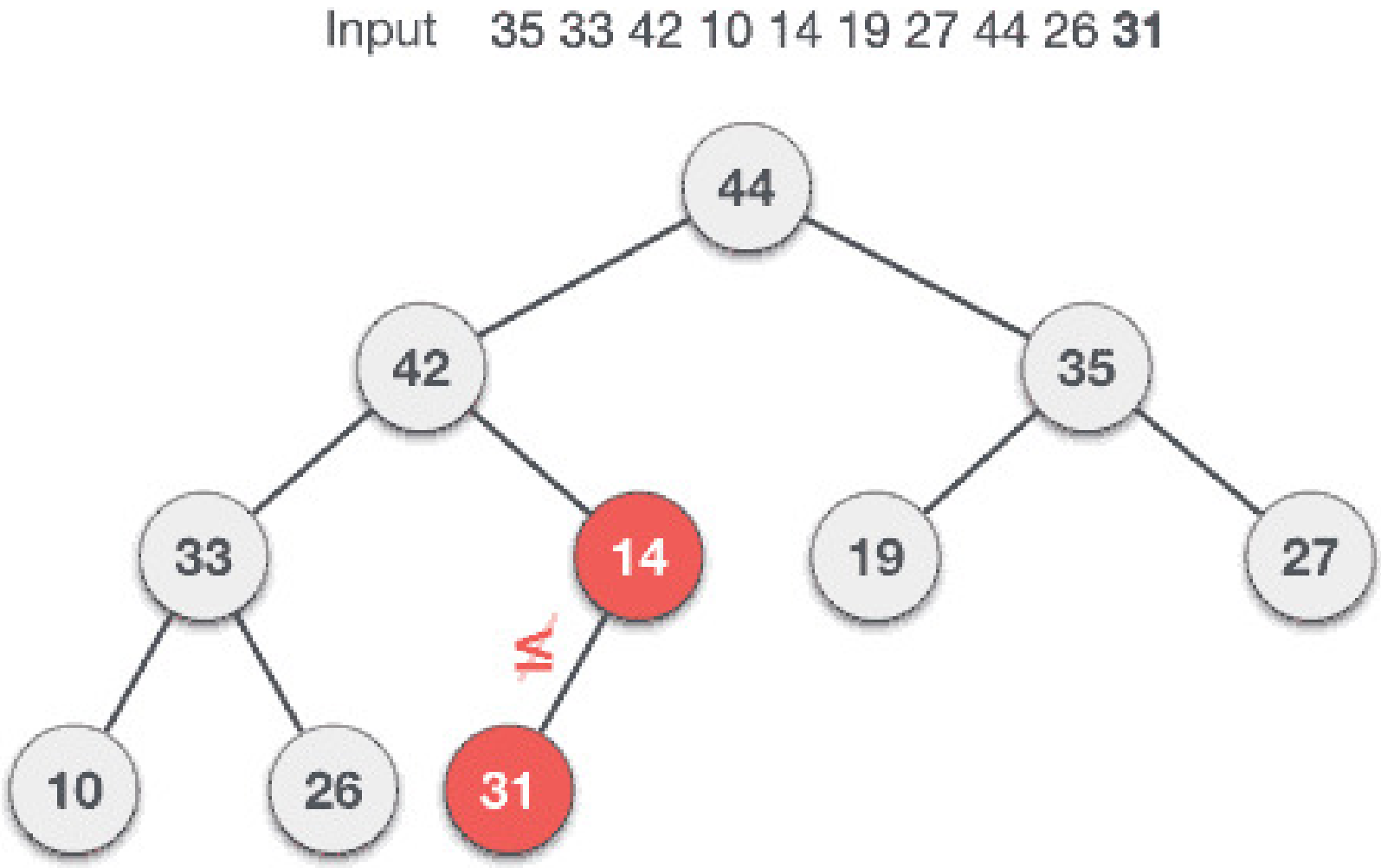


Exemple de construction de Max Heap

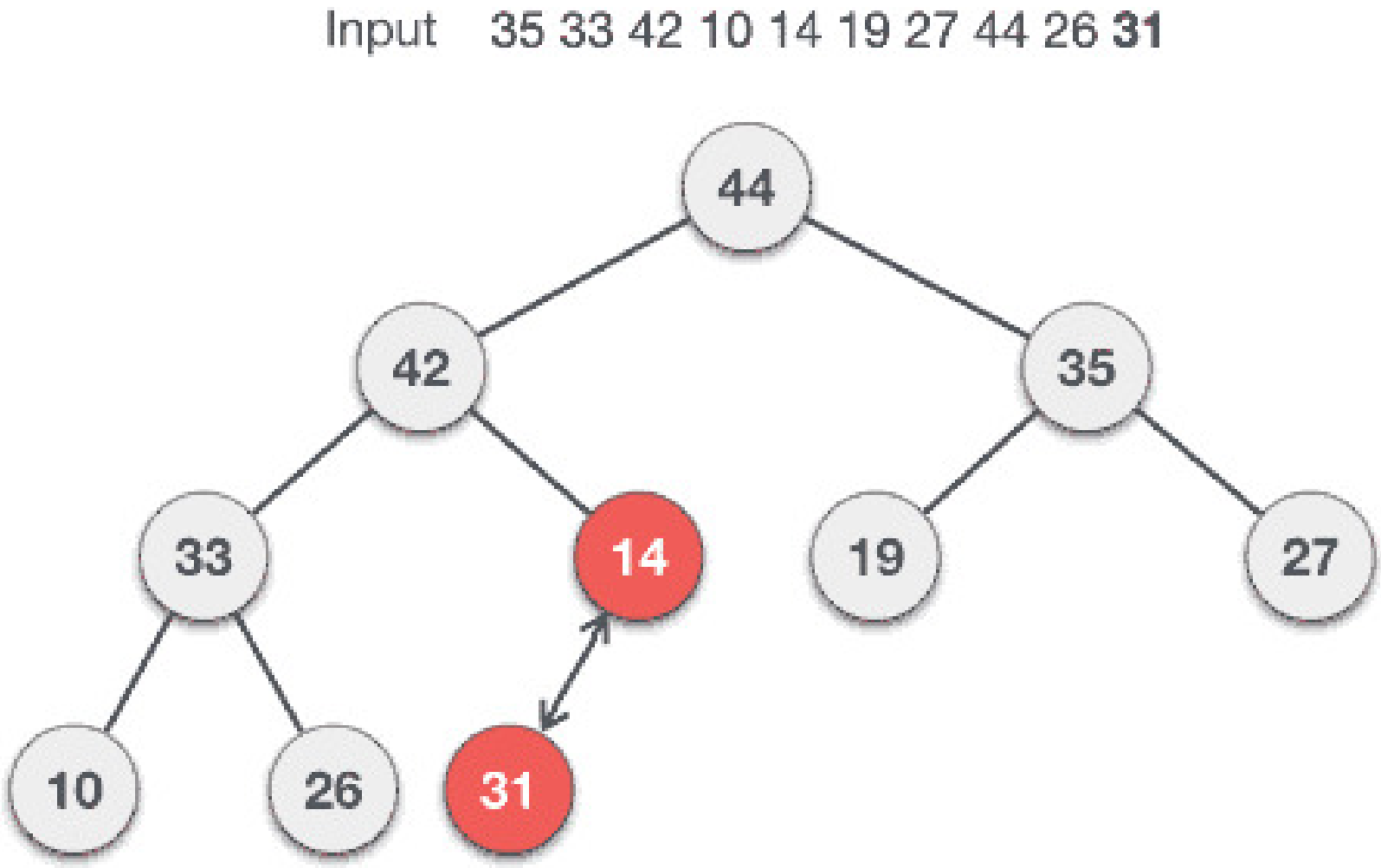
Input 35 33 42 10 14 19 27 44 26 **31**



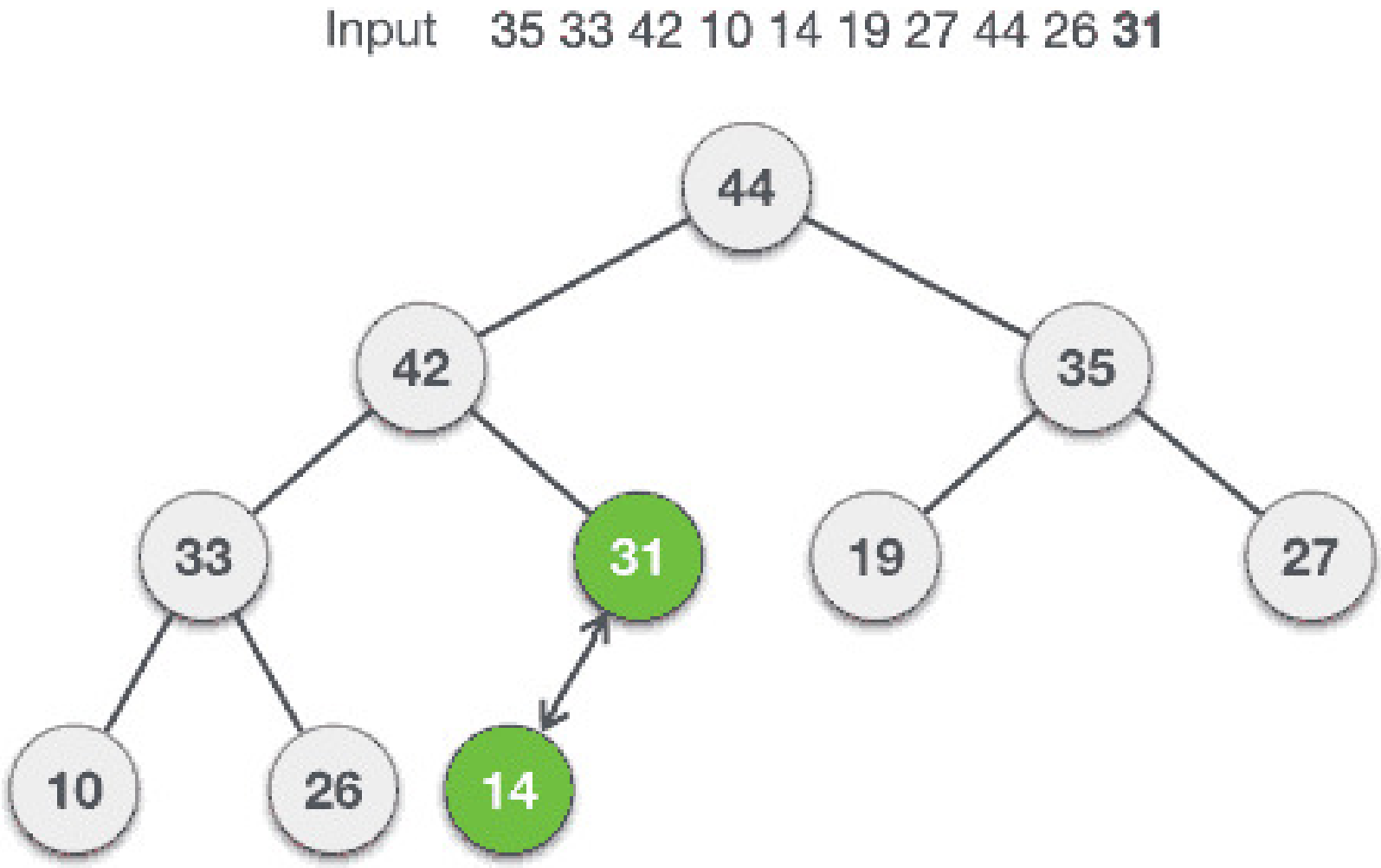
Exemple de construction de Max Heap



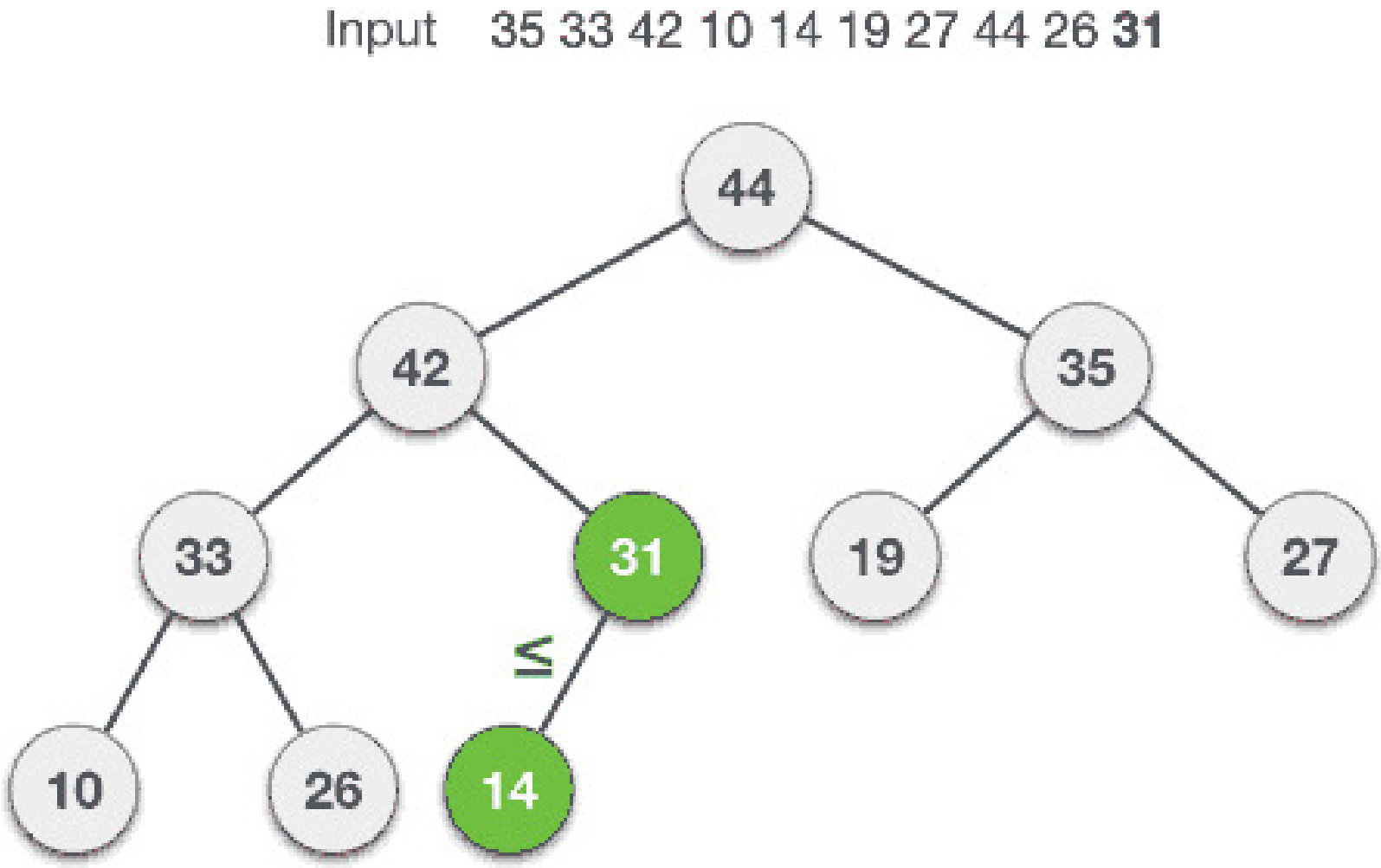
Exemple de construction de Max Heap



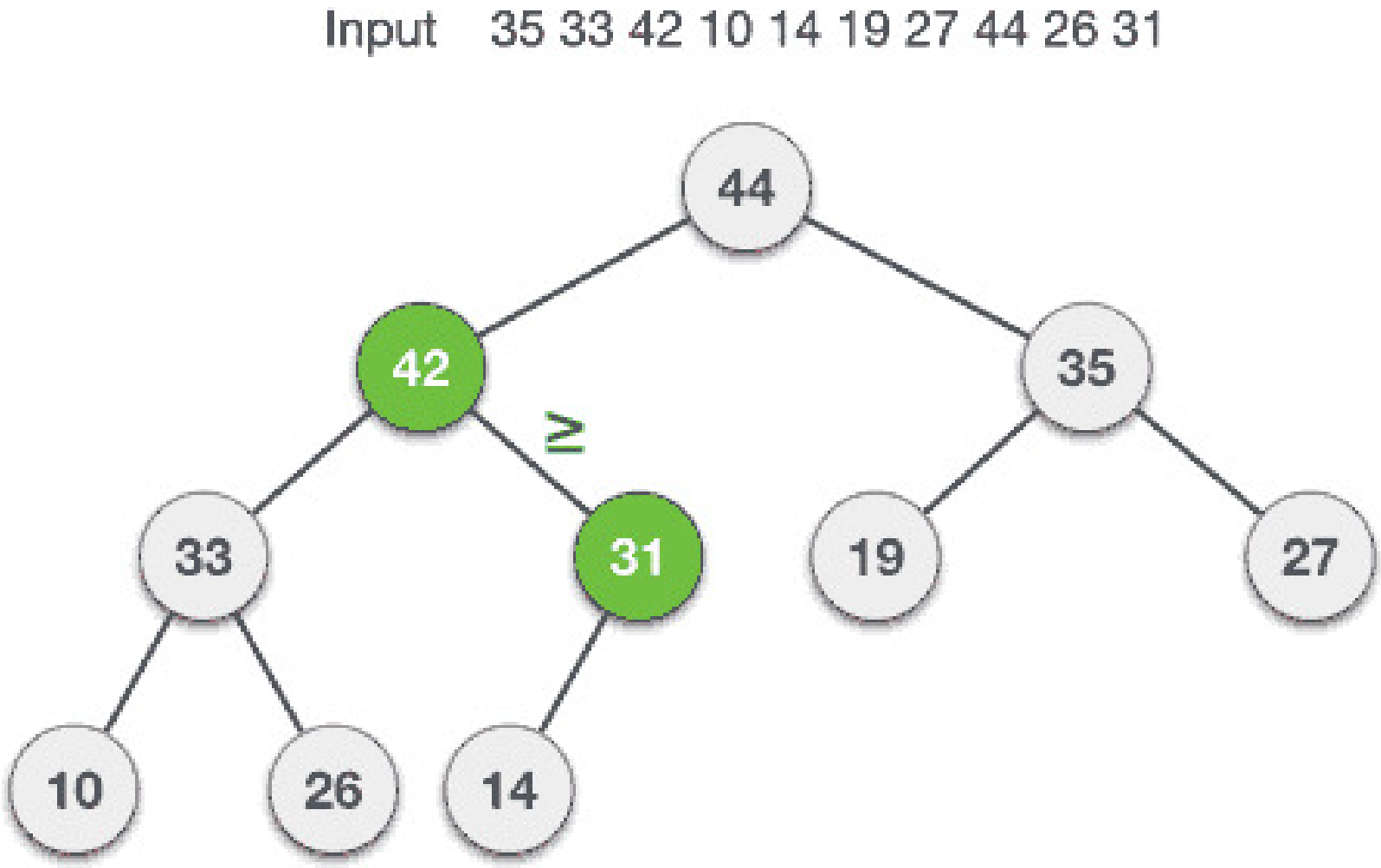
Exemple de construction de Max Heap



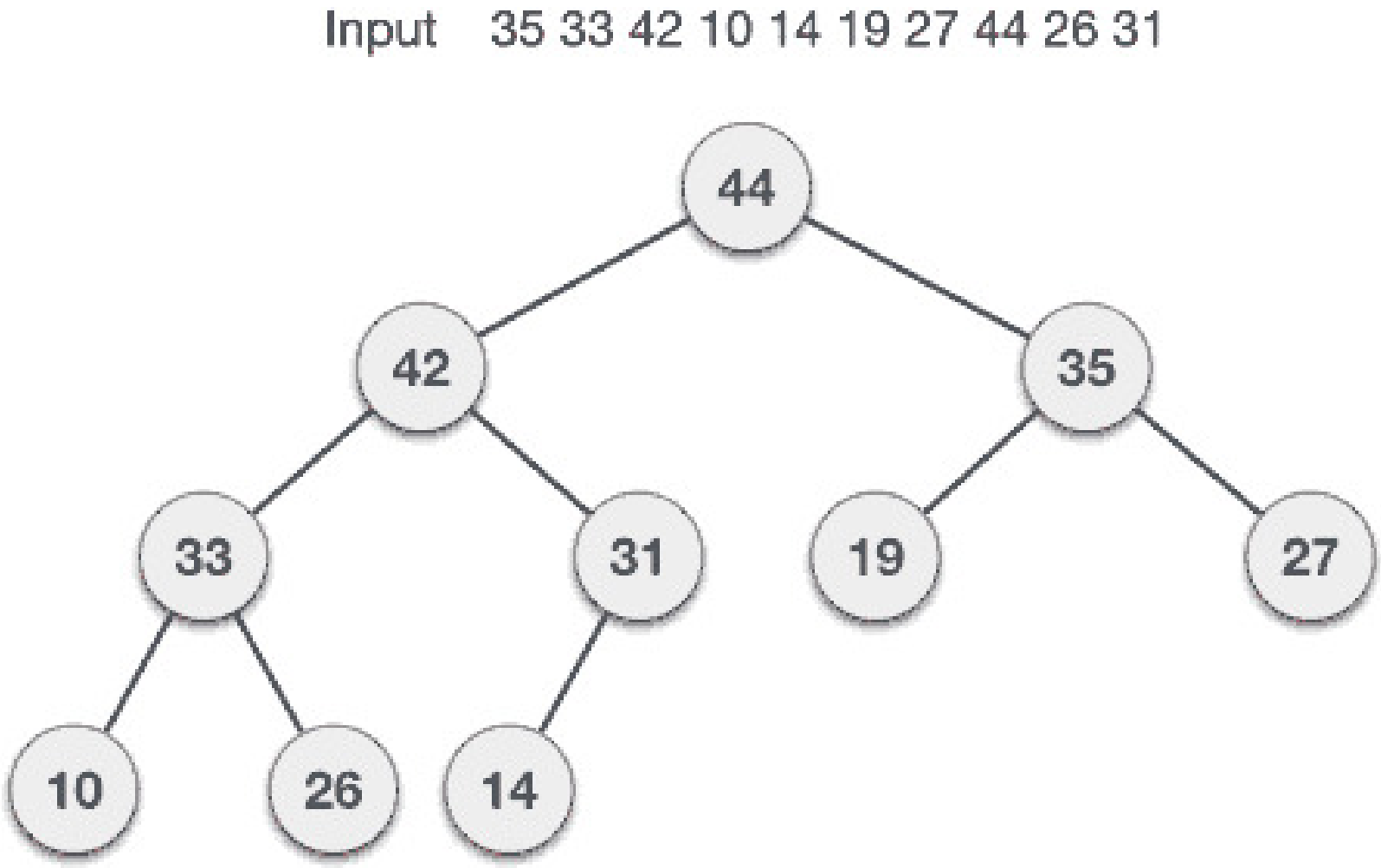
Exemple de construction de Max Heap



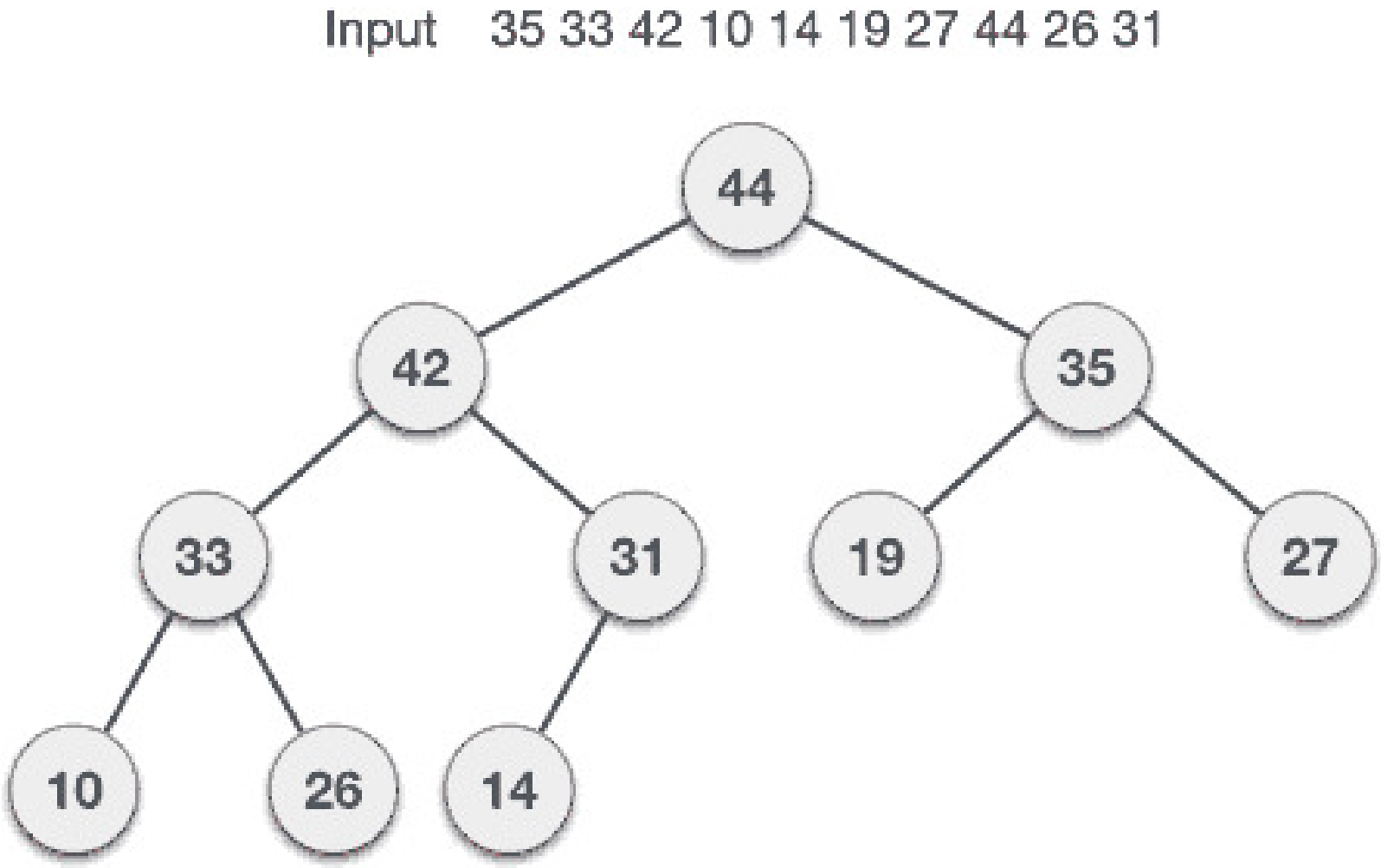
Exemple de construction de Max Heap



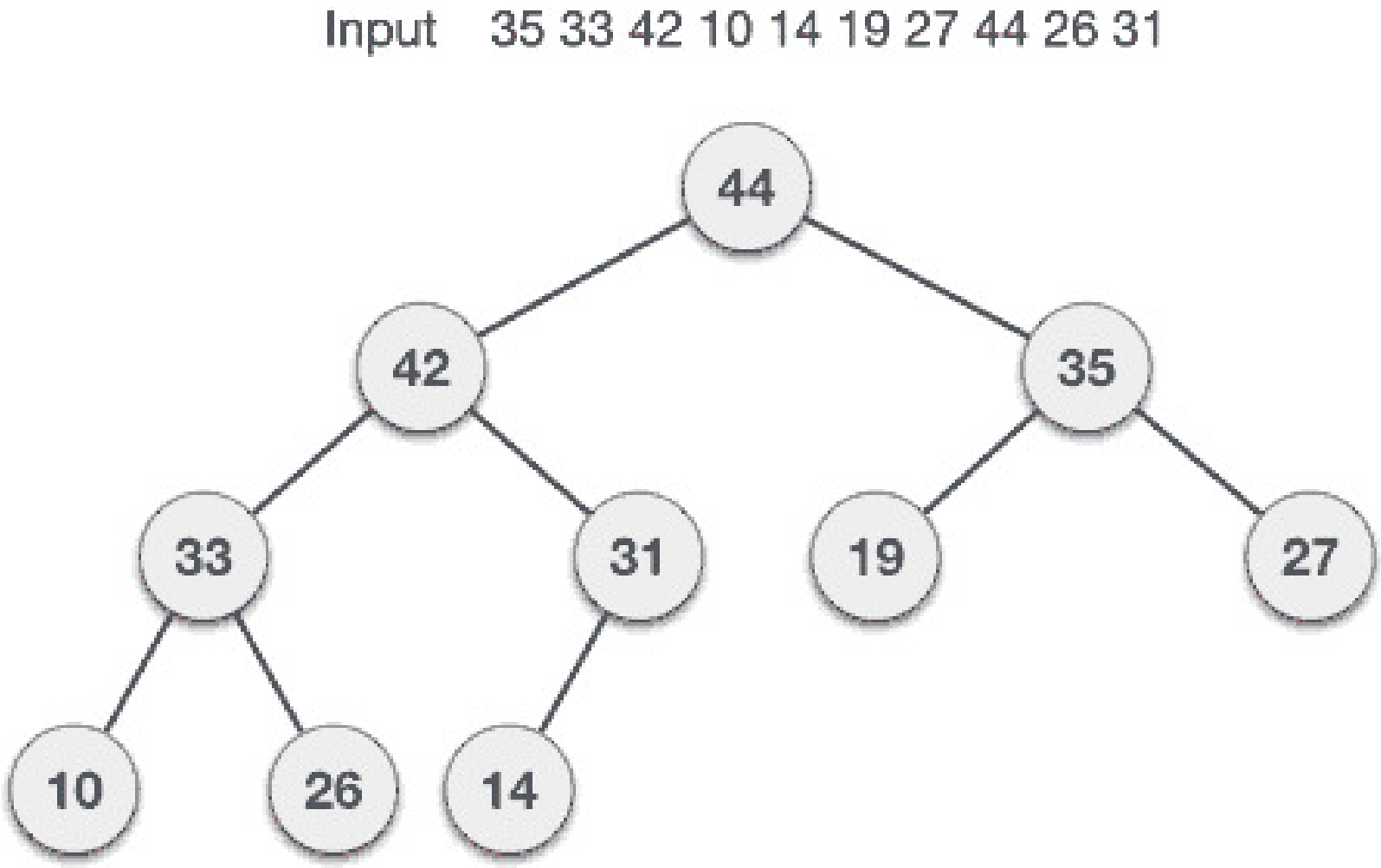
Exemple de construction de Max Heap



Exemple de construction de Max Heap



Exemple de construction de Max Heap



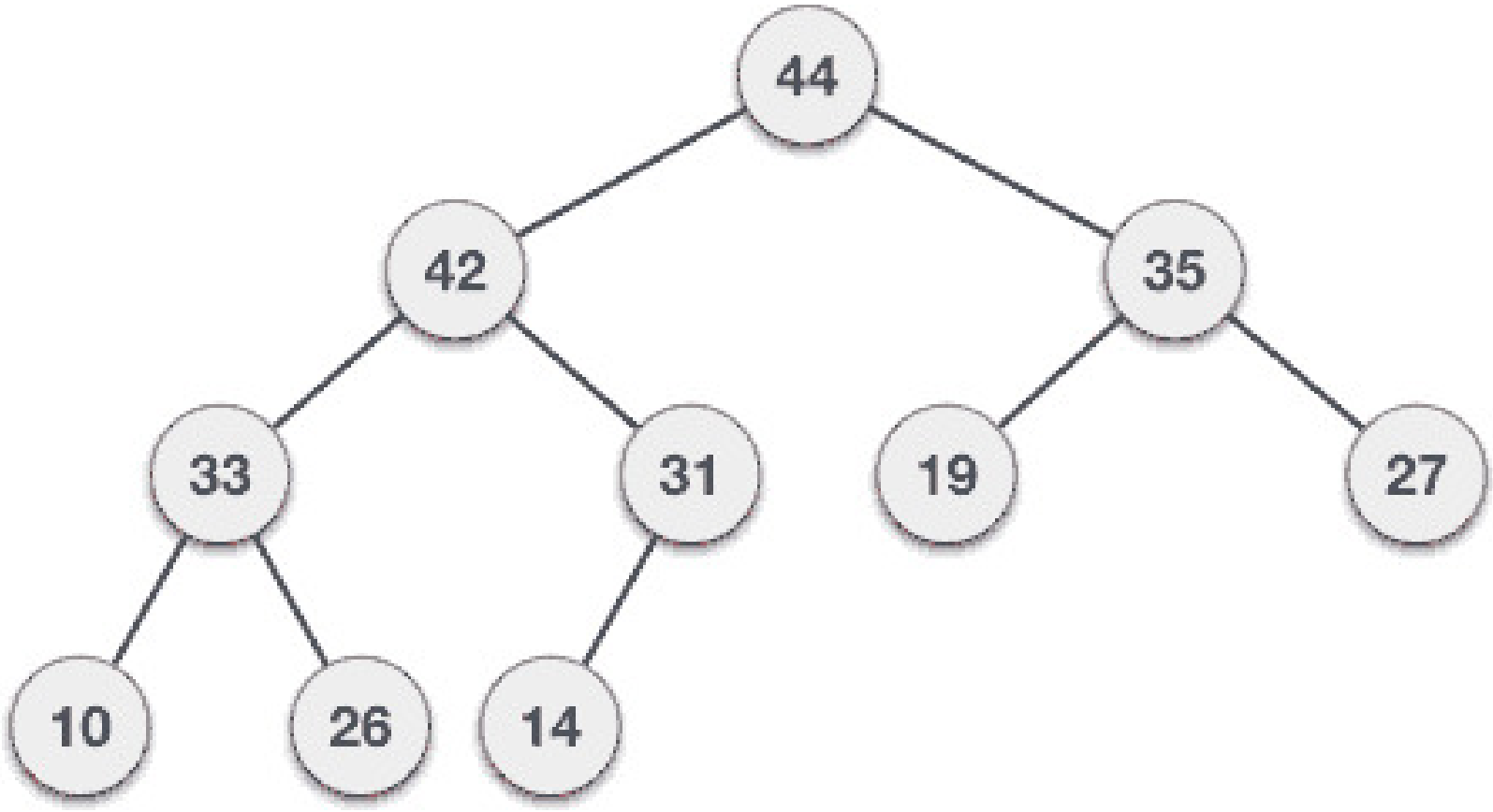
Algorithme de suppression dans un Max Heap

Déterminons un algorithme de suppression dans le tas maximum. La suppression dans le tas Max (ou Min) se produit toujours à la racine pour supprimer la valeur maximale (ou minimale).

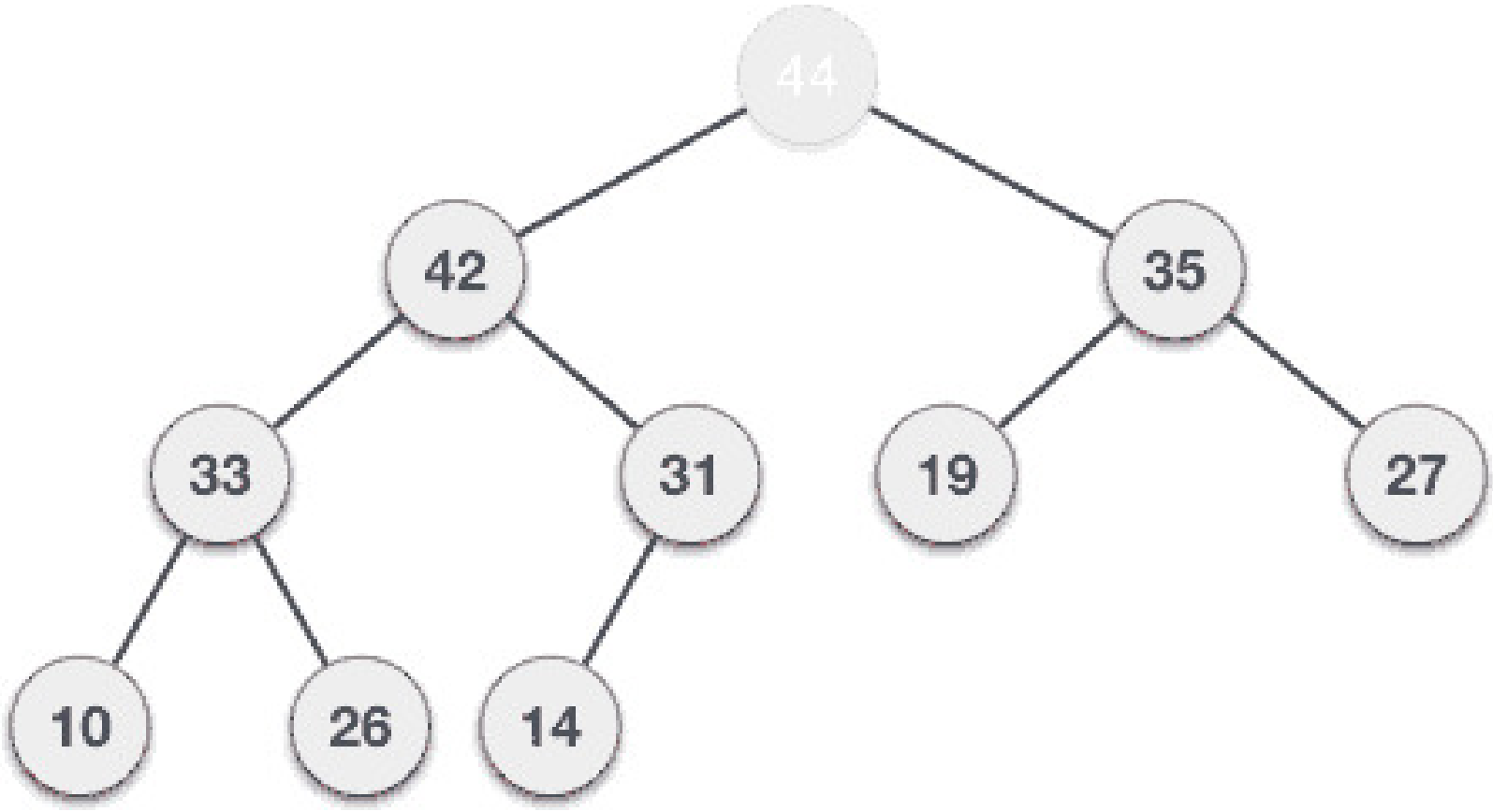
Algorithme

- Suppression du nœud racine.
- Déplacer le dernier élément du dernier niveau vers la racine.
- Comparer la valeur de ce noeud enfant avec son parent.
- Si la valeur du parent est inférieure à celle de l'enfant, échangez-les.
- Répétez les étapes 3 et 4 jusqu'à ce que la propriété "Heap" soit respectée.

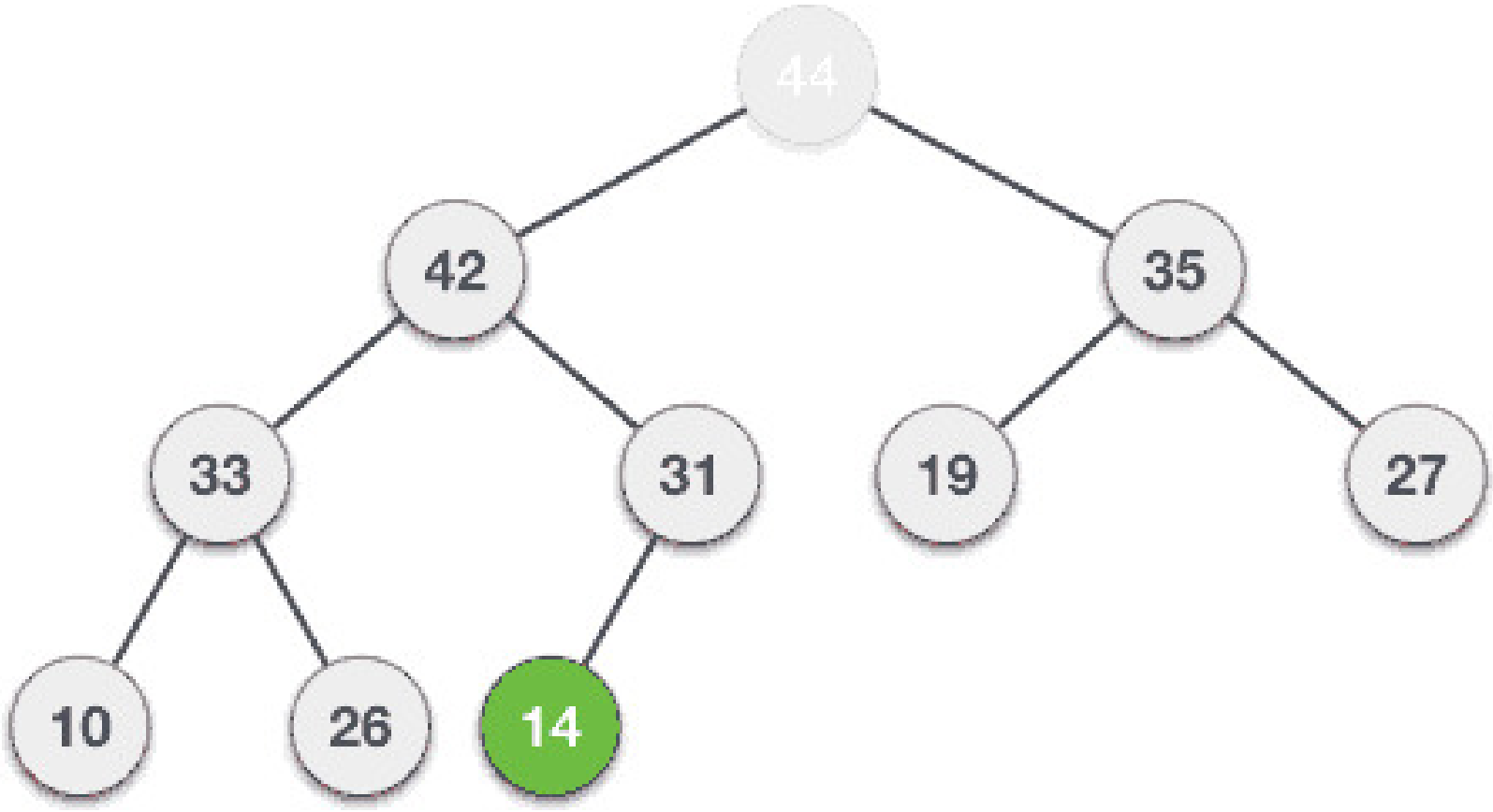
Exemple de suppression dans un Max Heap



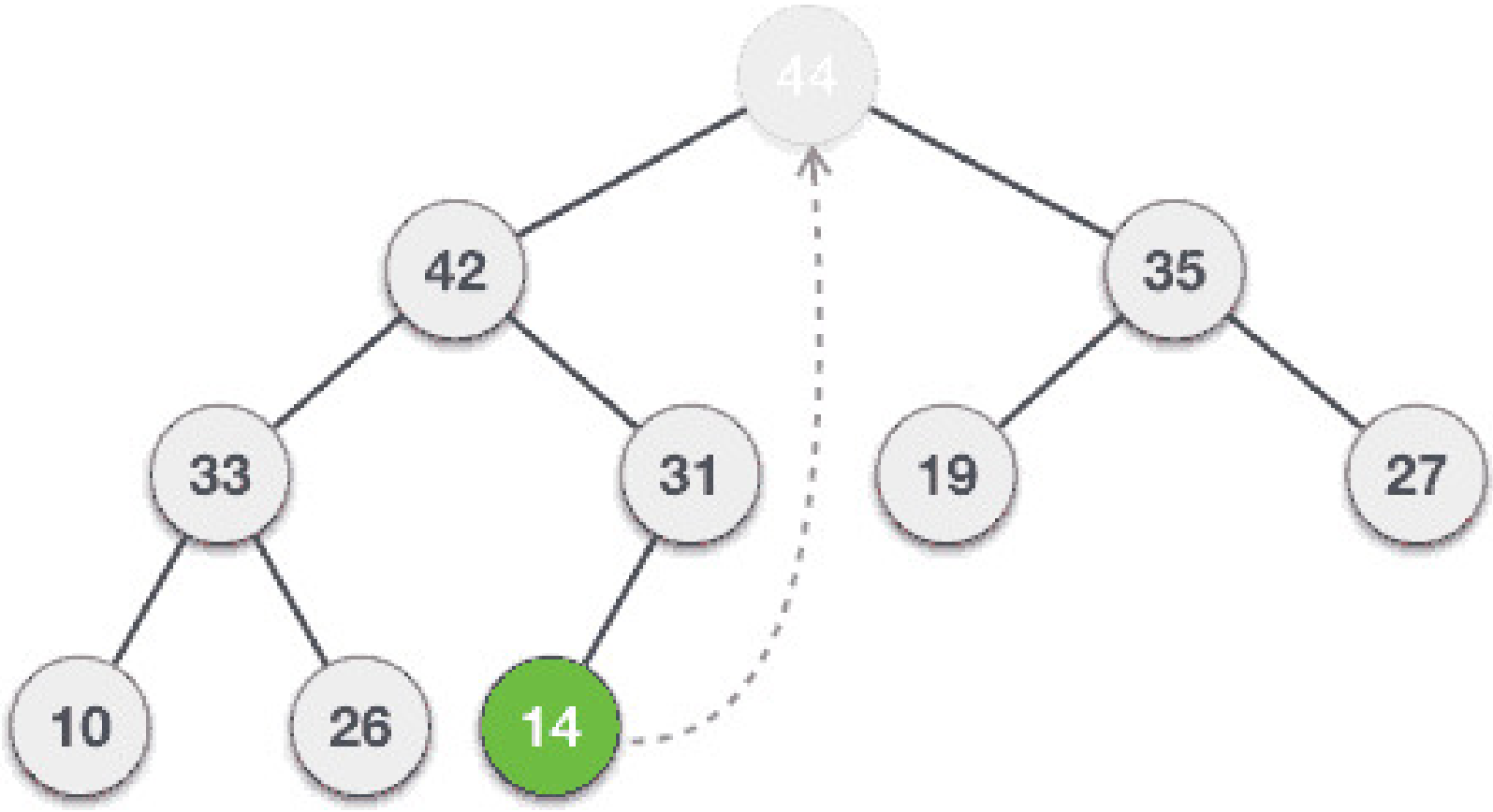
Exemple de suppression dans un Max Heap



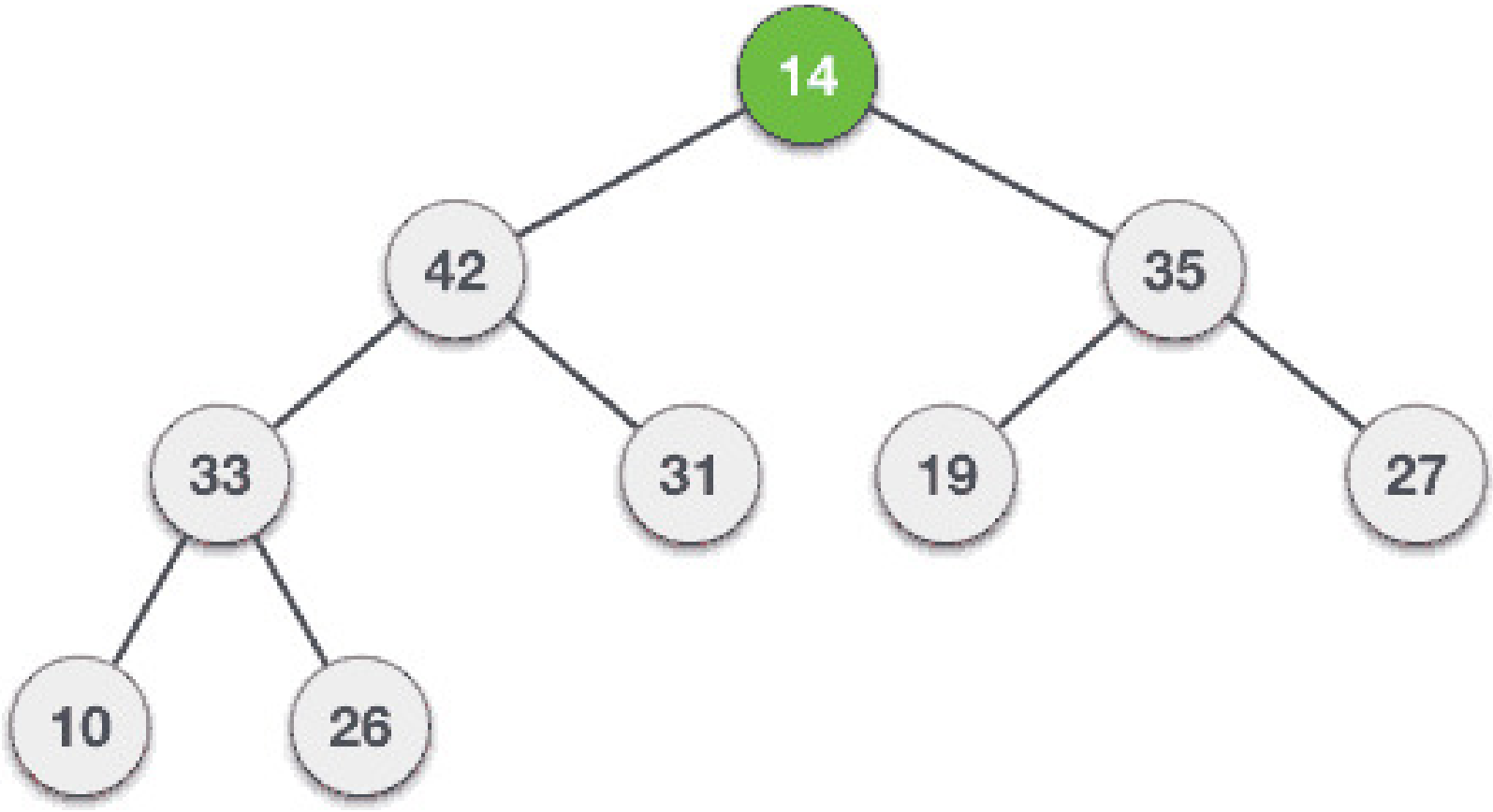
Exemple de suppression dans un Max Heap



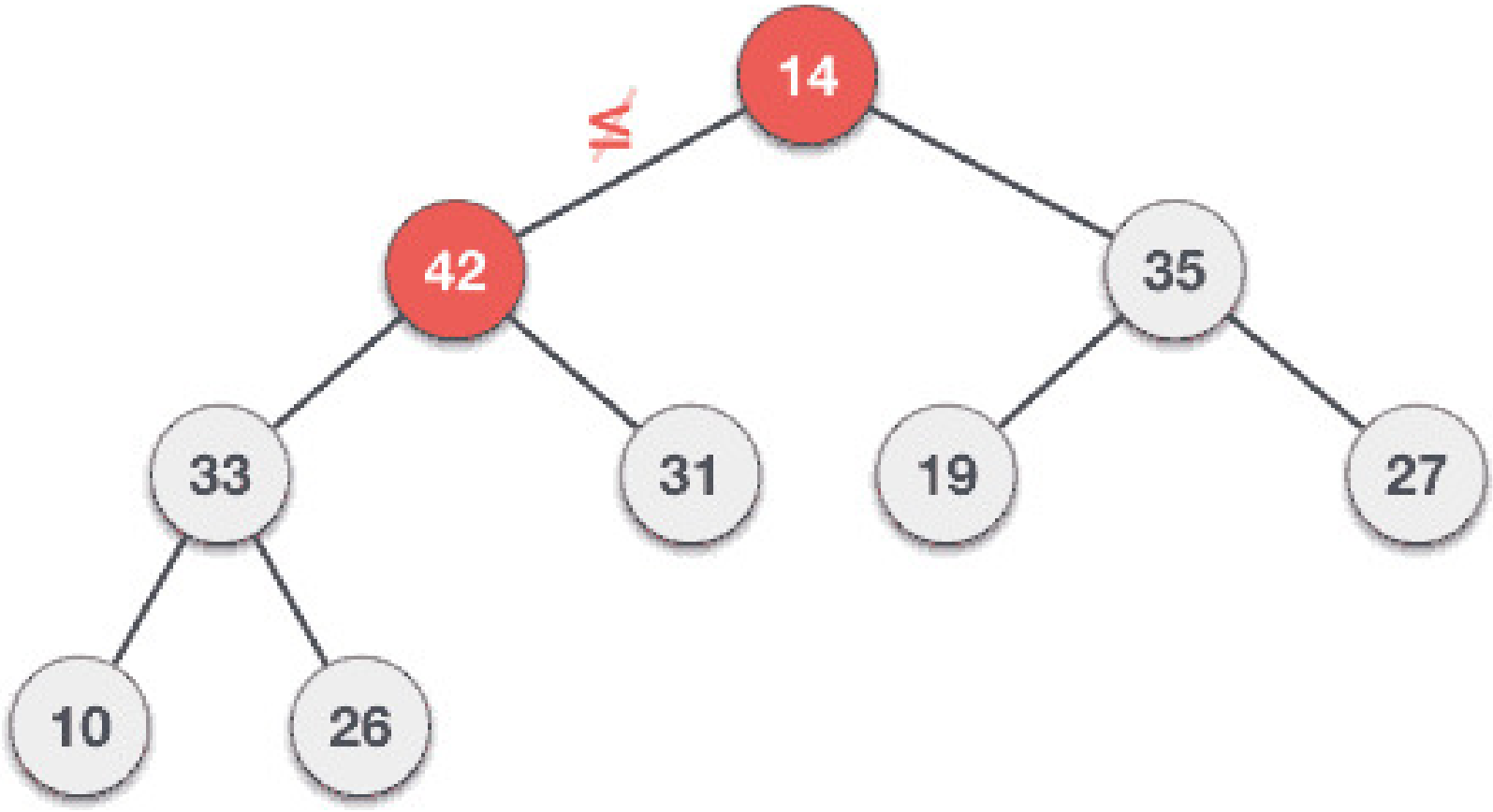
Exemple de suppression dans un Max Heap



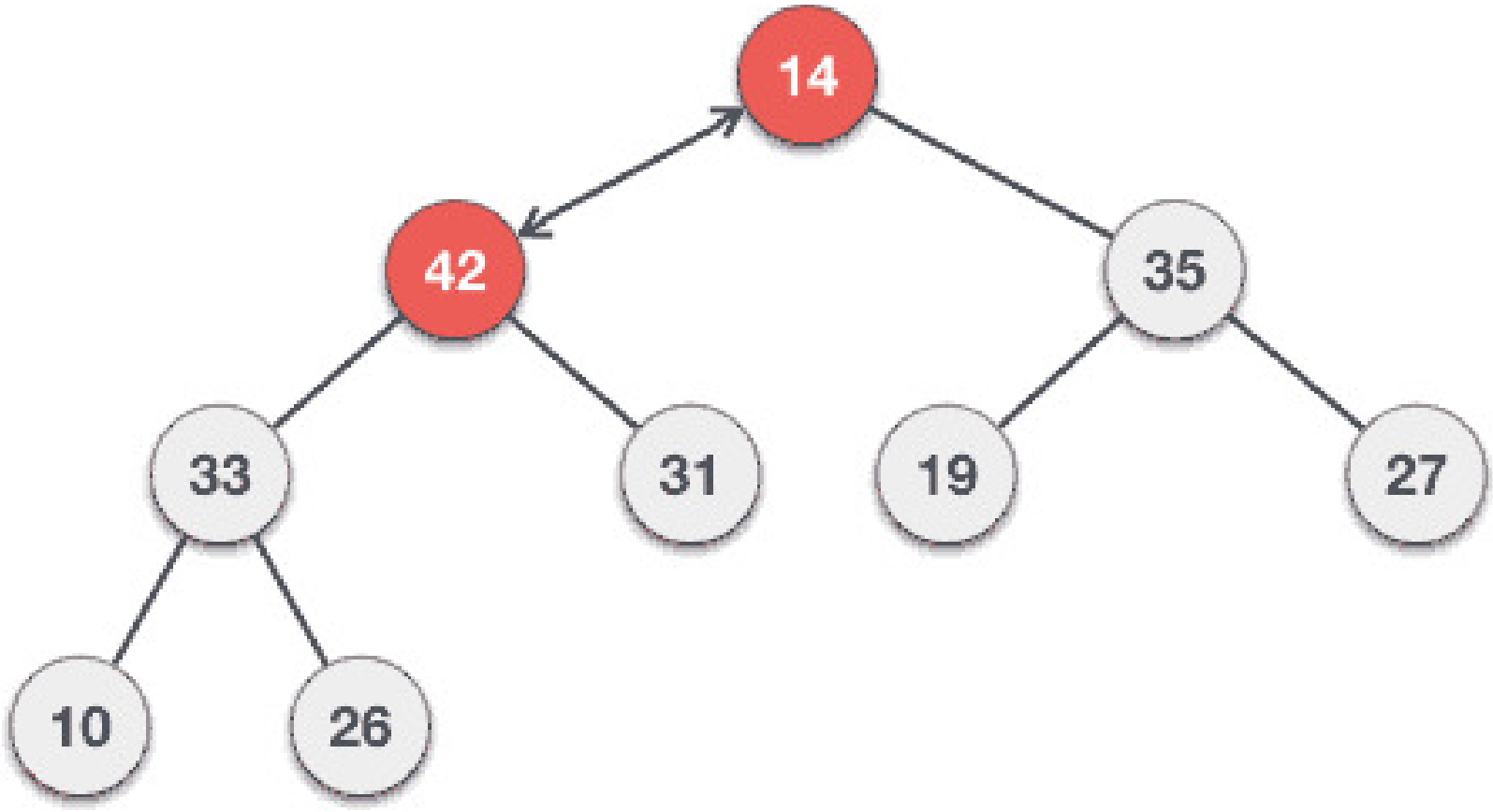
Exemple de suppression dans un Max Heap



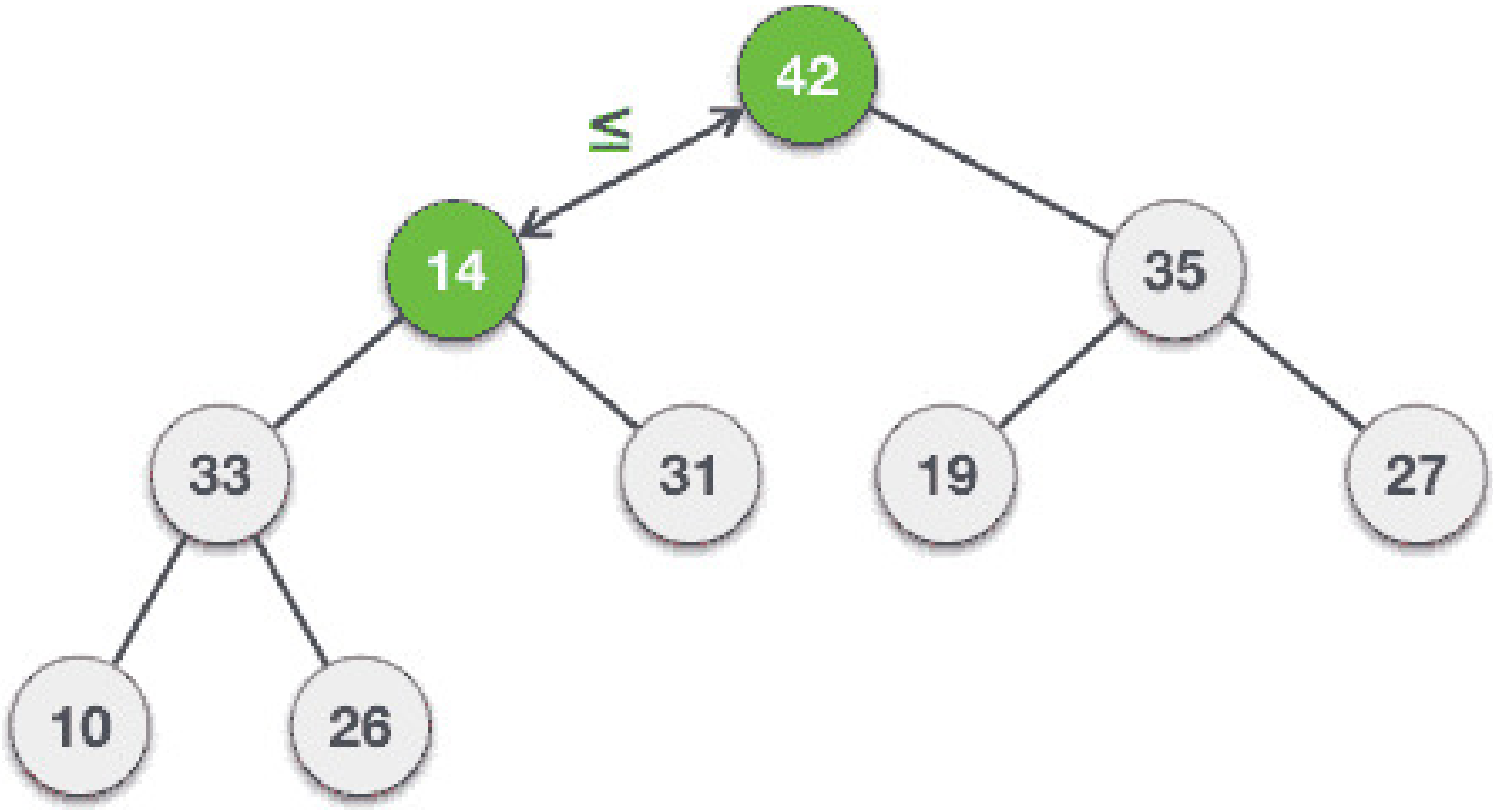
Exemple de suppression dans un Max Heap



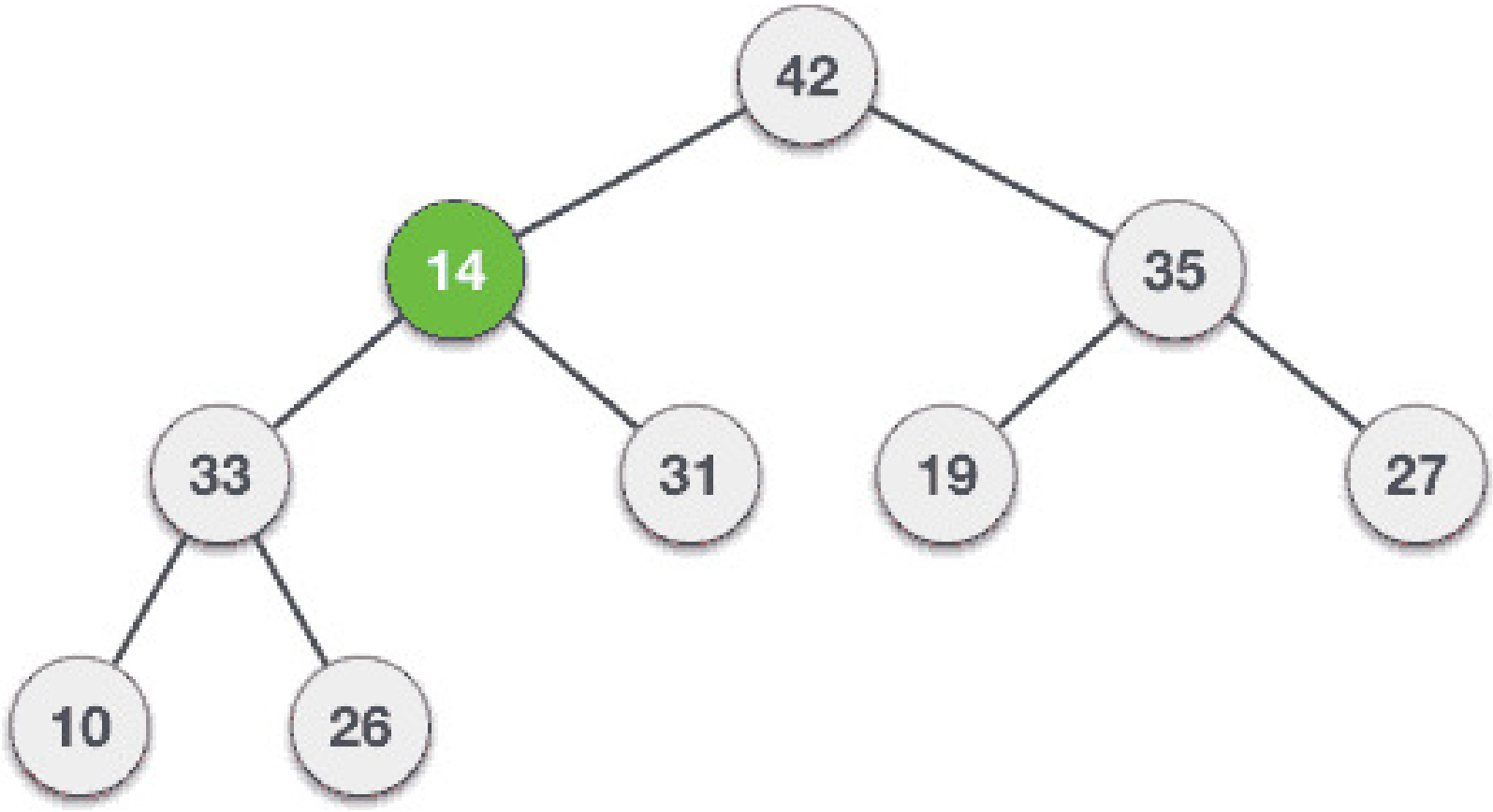
Exemple de suppression dans un Max Heap



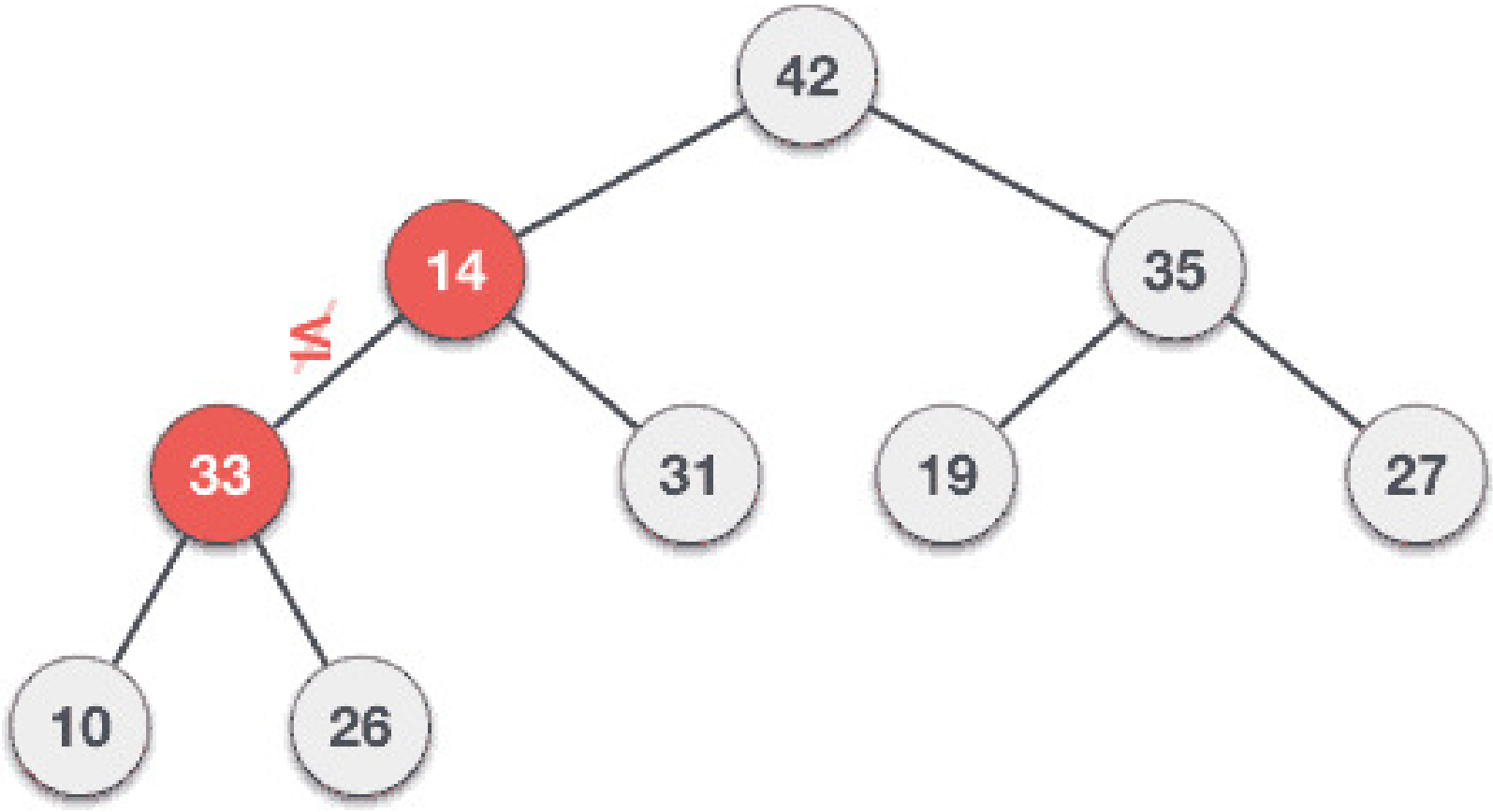
Exemple de suppression dans un Max Heap



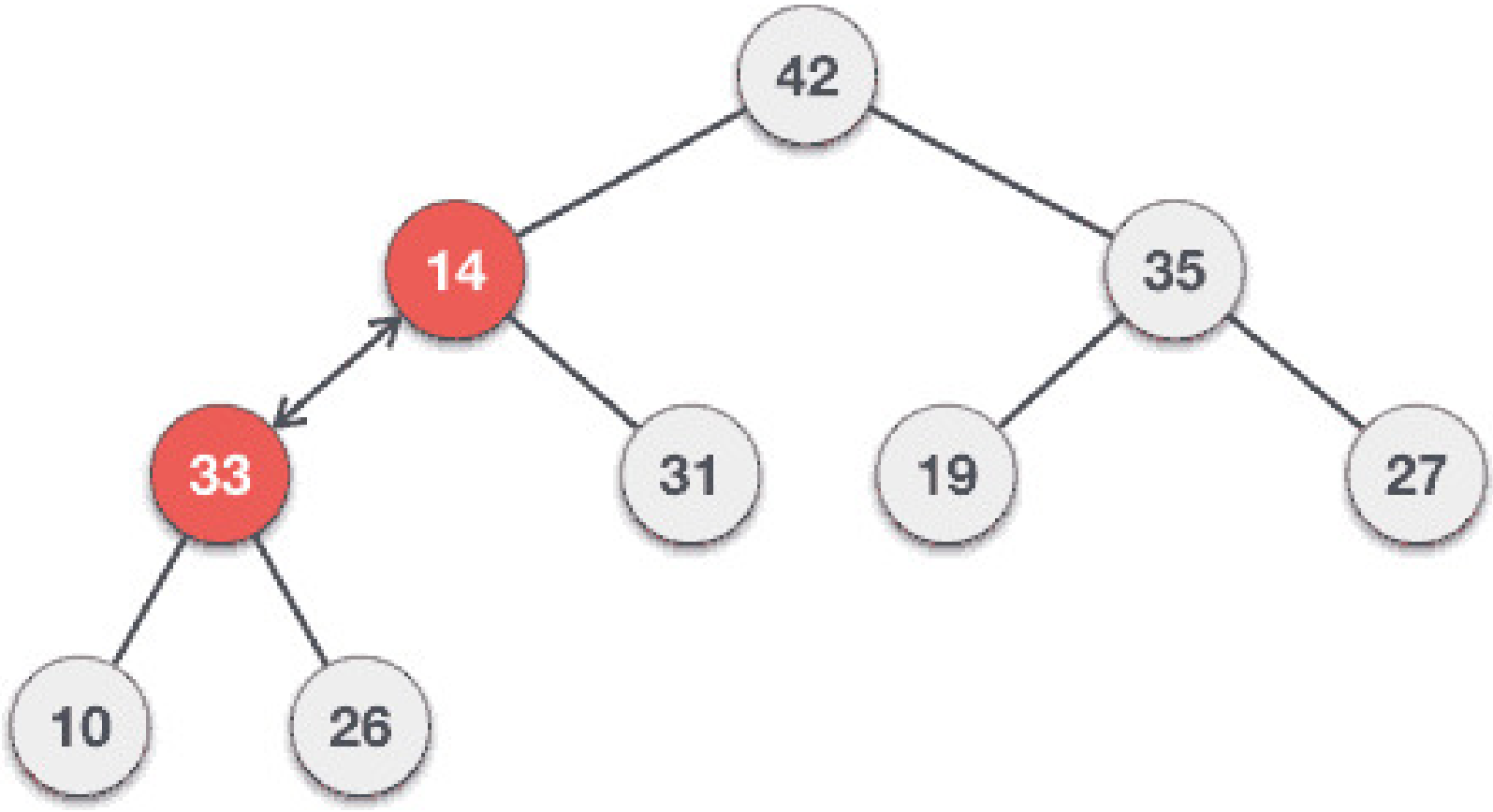
Exemple de suppression dans un Max Heap



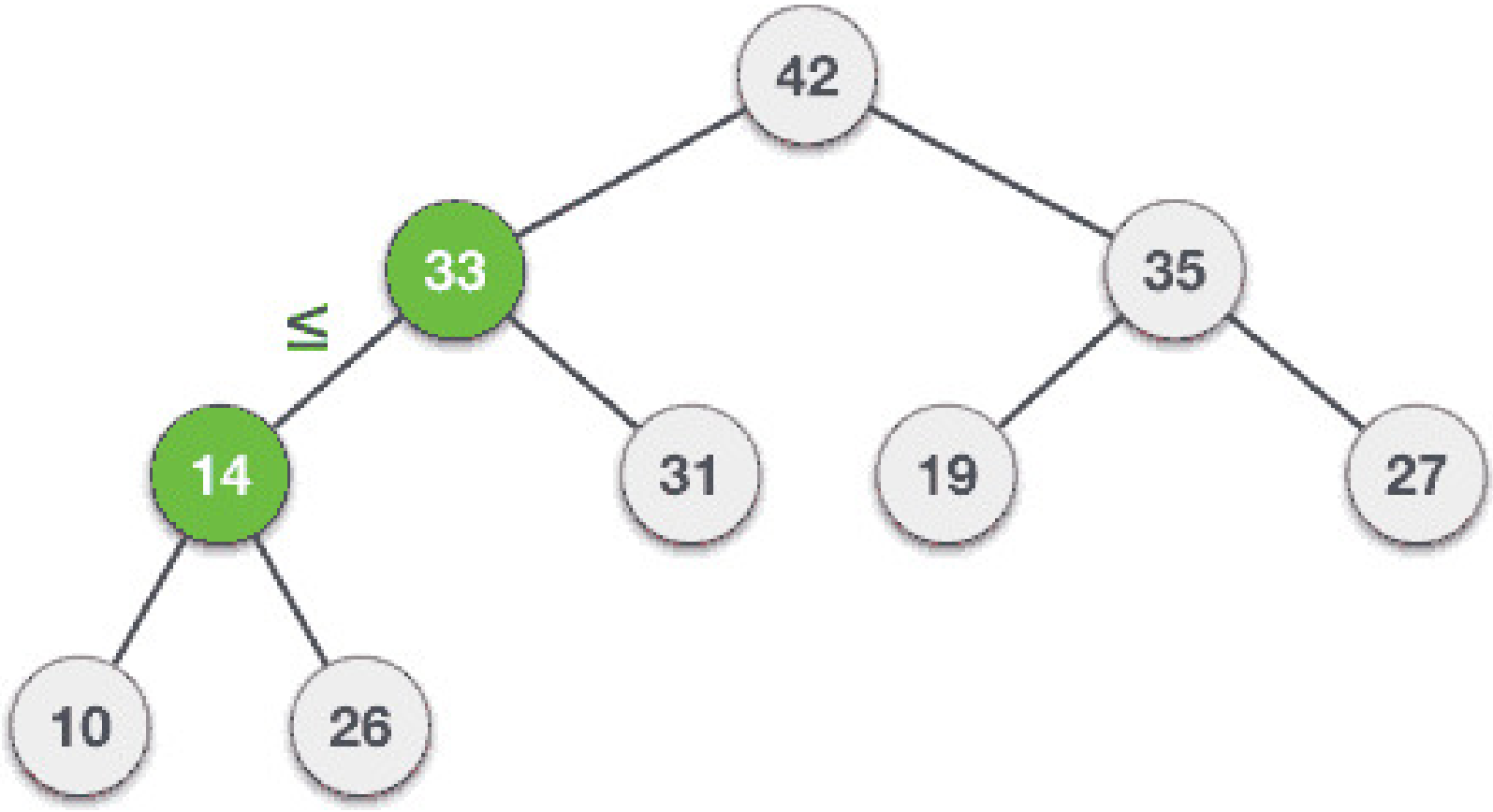
Exemple de suppression dans un Max Heap



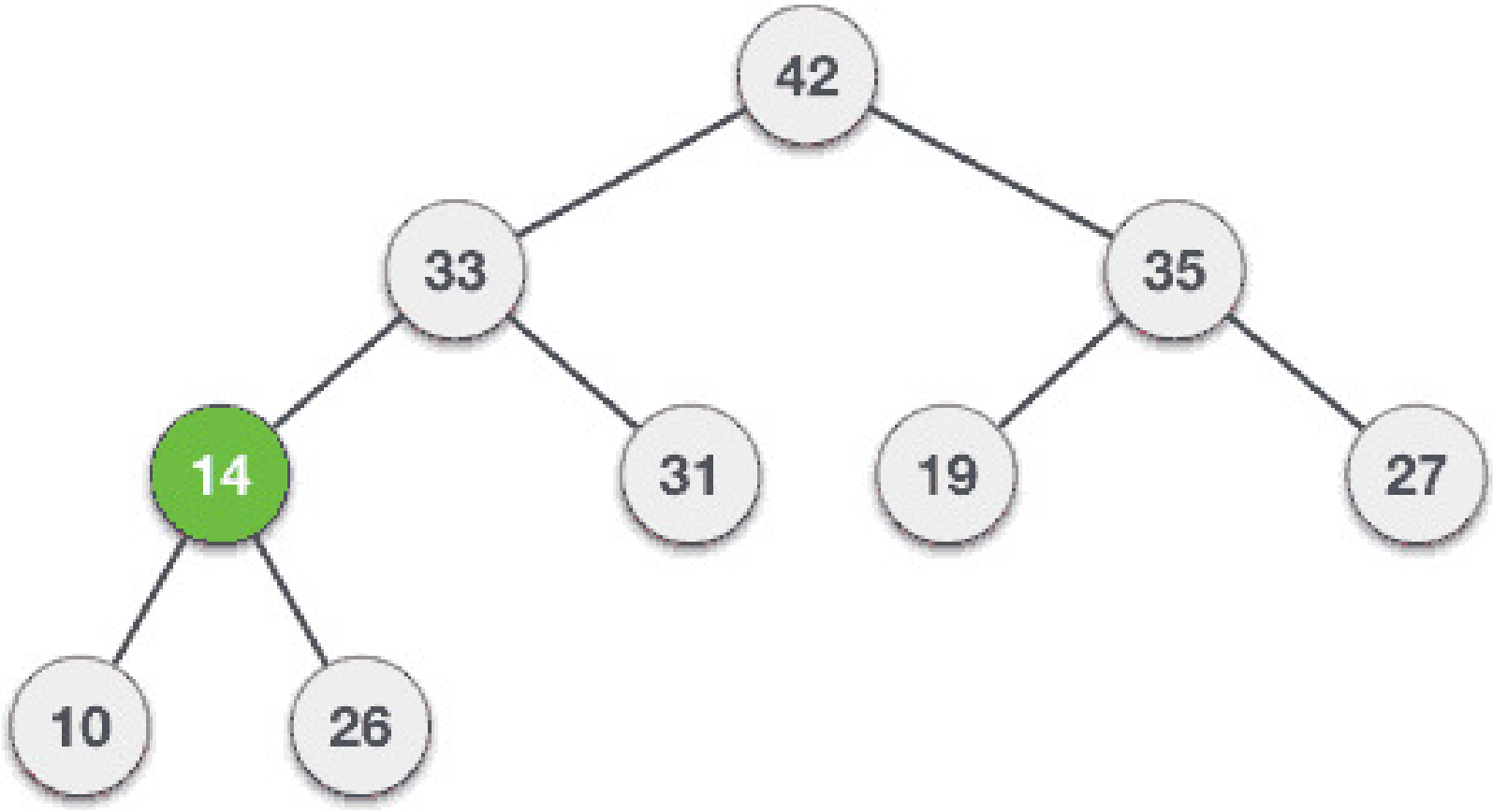
Exemple de suppression dans un Max Heap



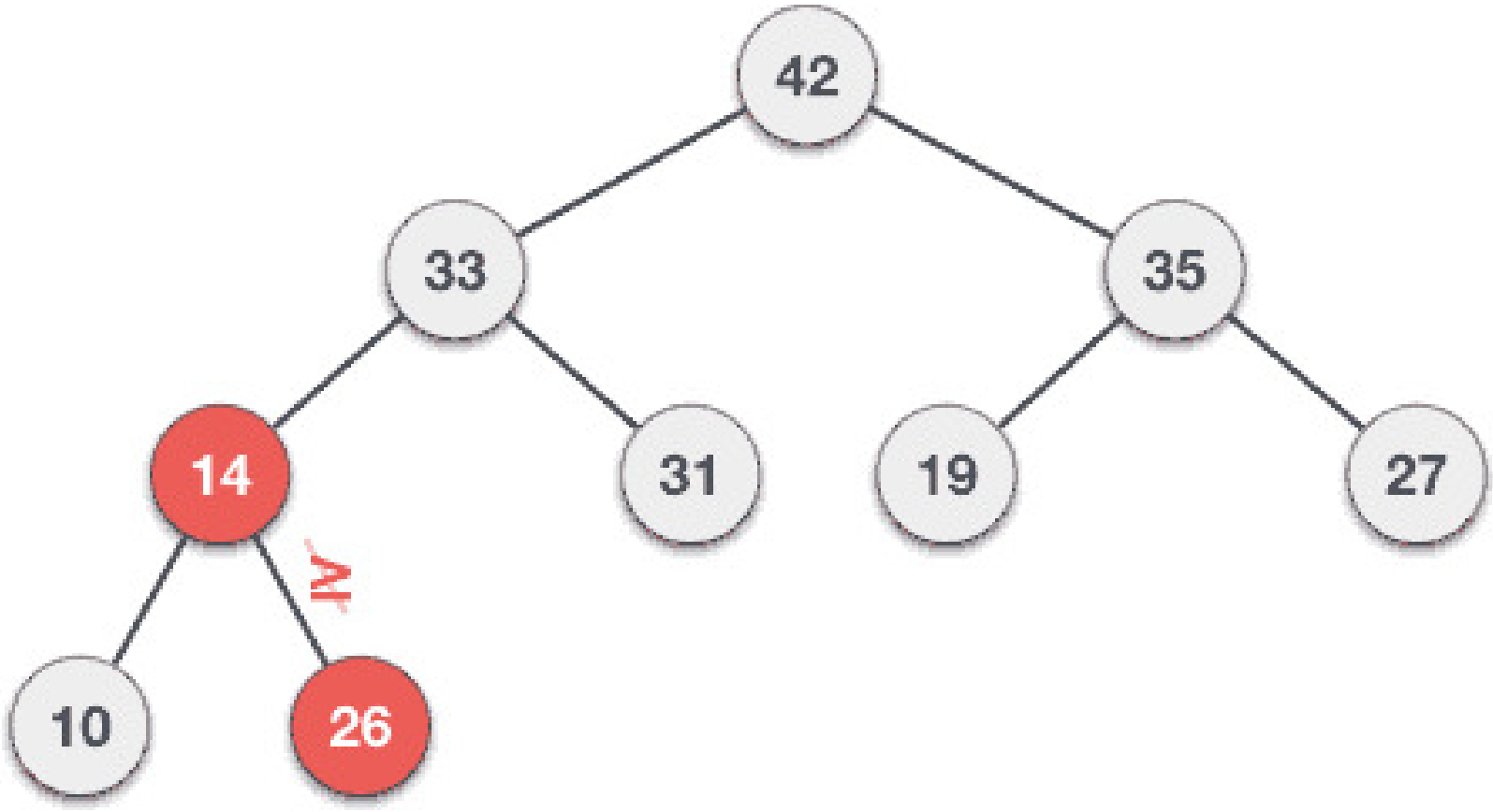
Exemple de suppression dans un Max Heap



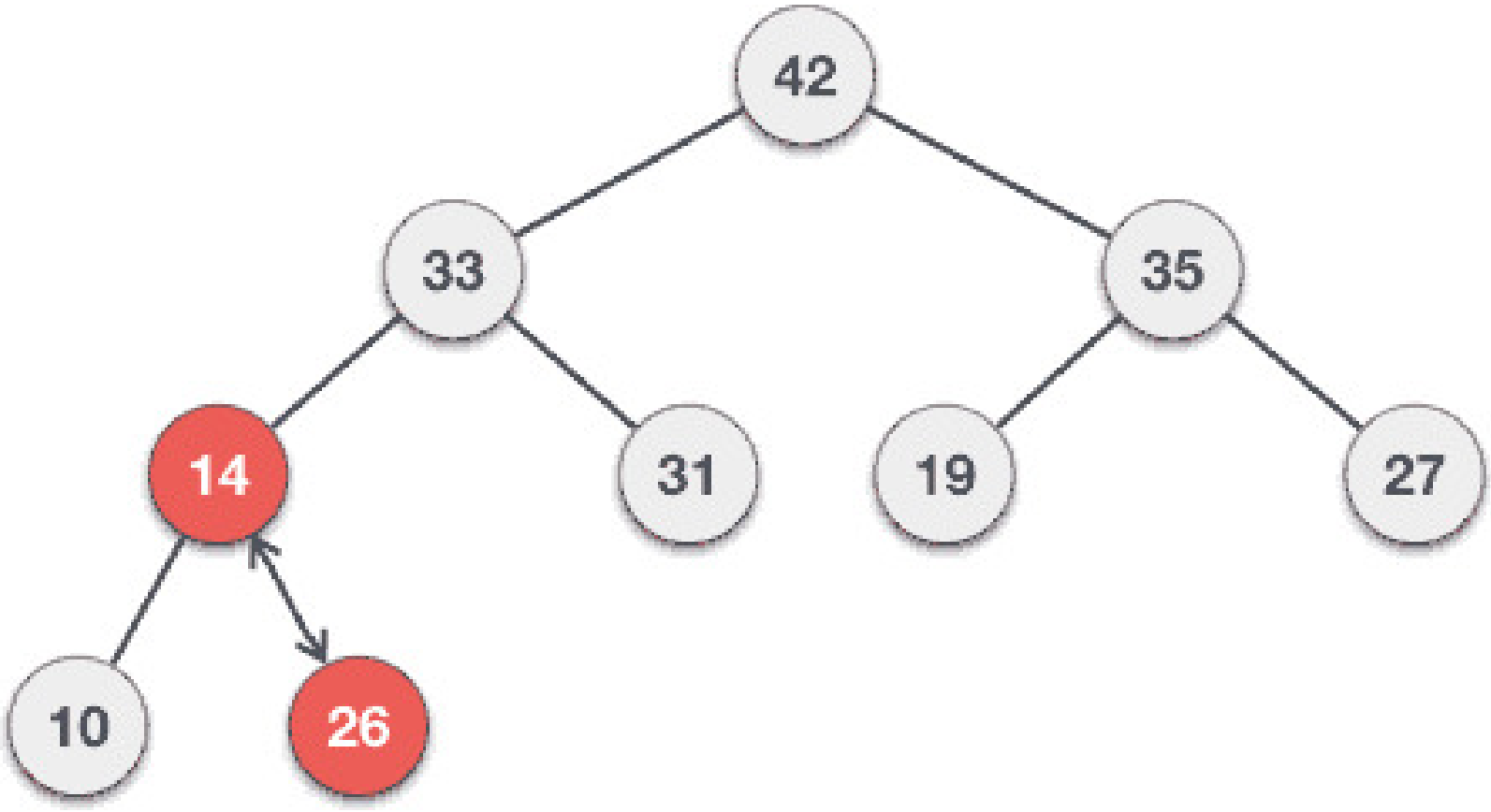
Exemple de suppression dans un Max Heap



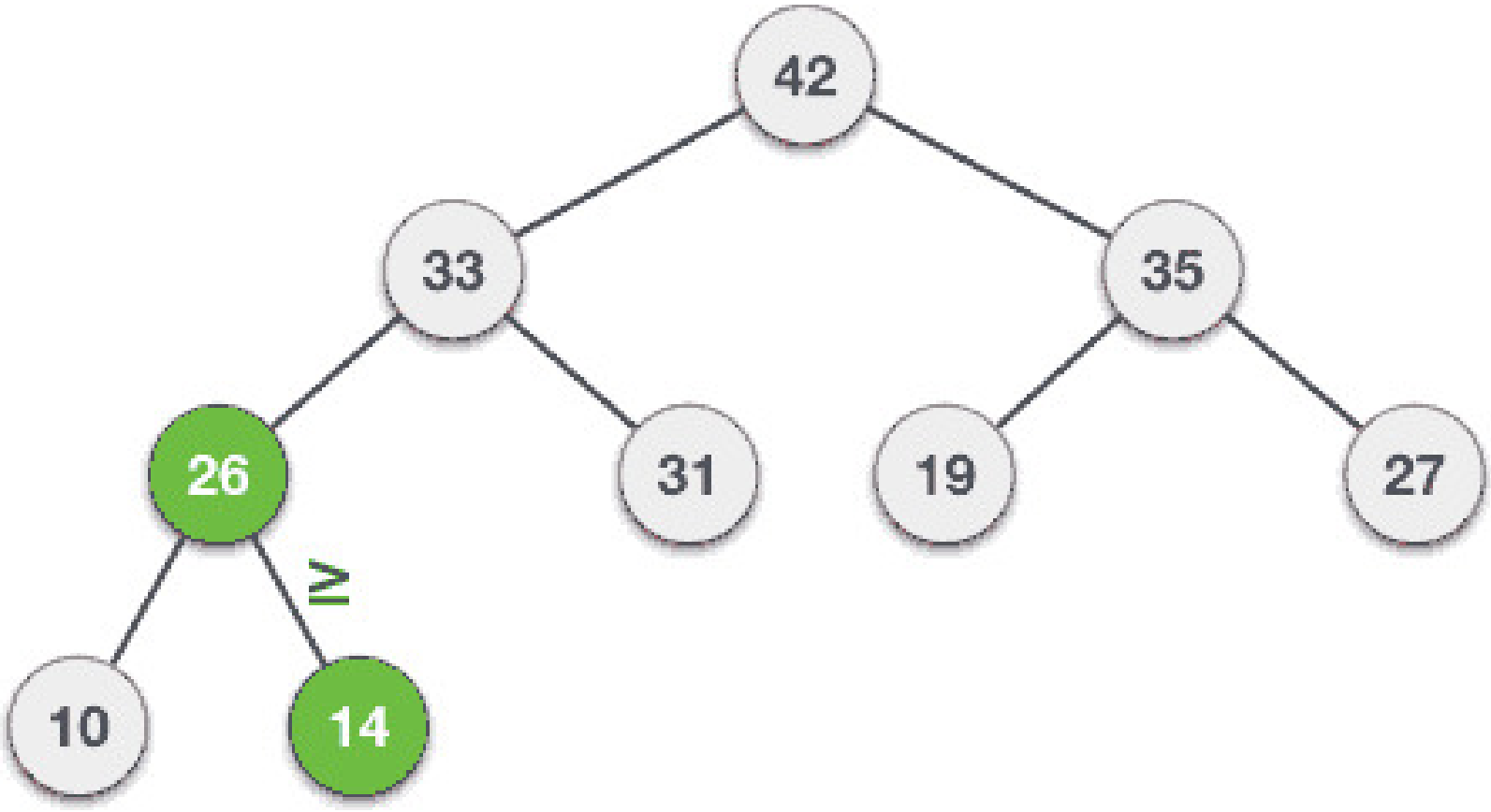
Exemple de suppression dans un Max Heap



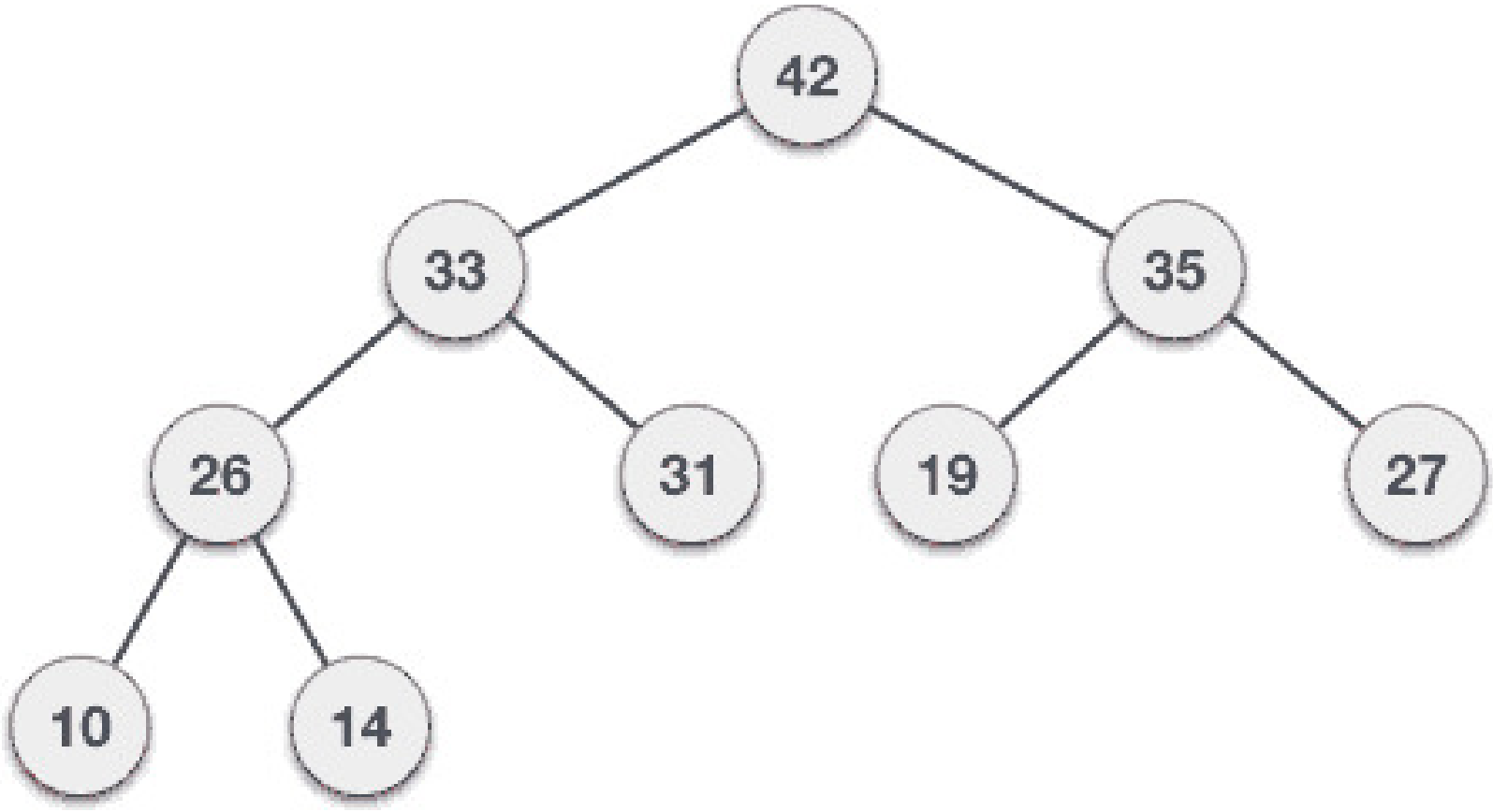
Exemple de suppression dans un Max Heap



Exemple de suppression dans un Max Heap



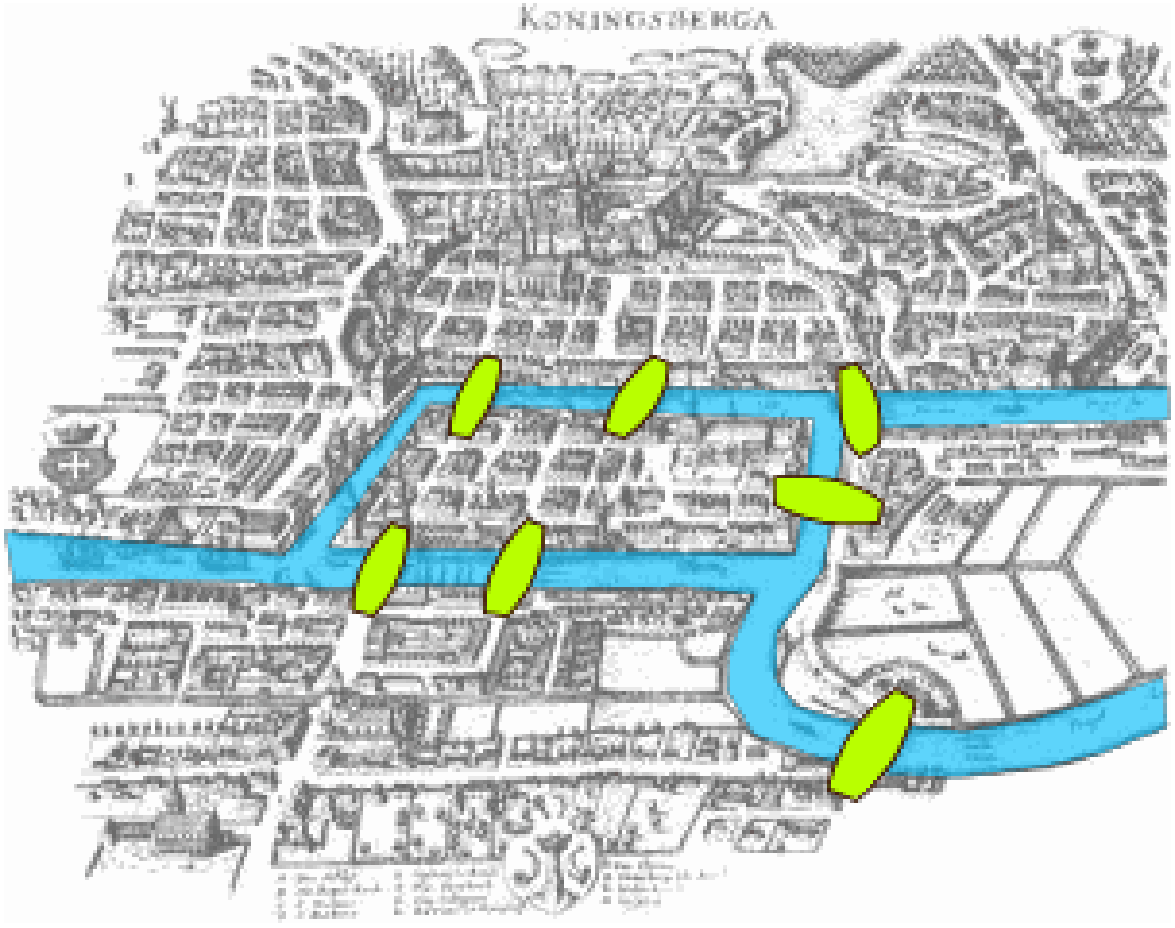
Exemple de suppression dans un Max Heap



Les graphes

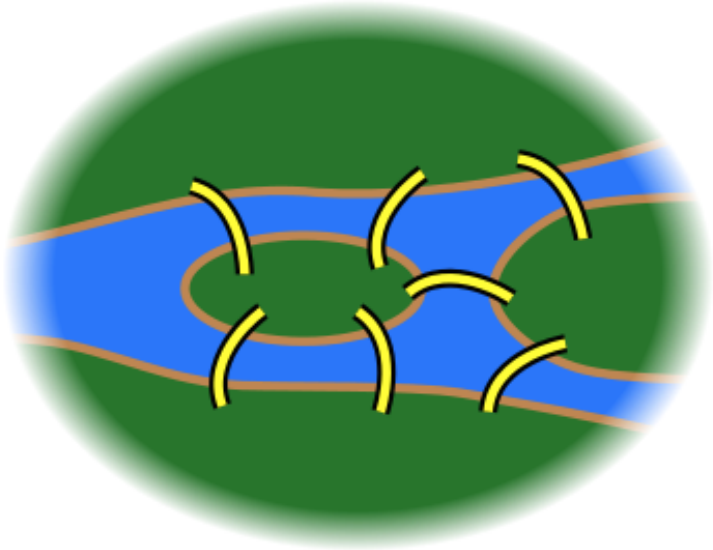
Un peu d'histoire

L'histoire de la théorie des graphes débute avec les travaux d'Euler au XVIIIe siècle.



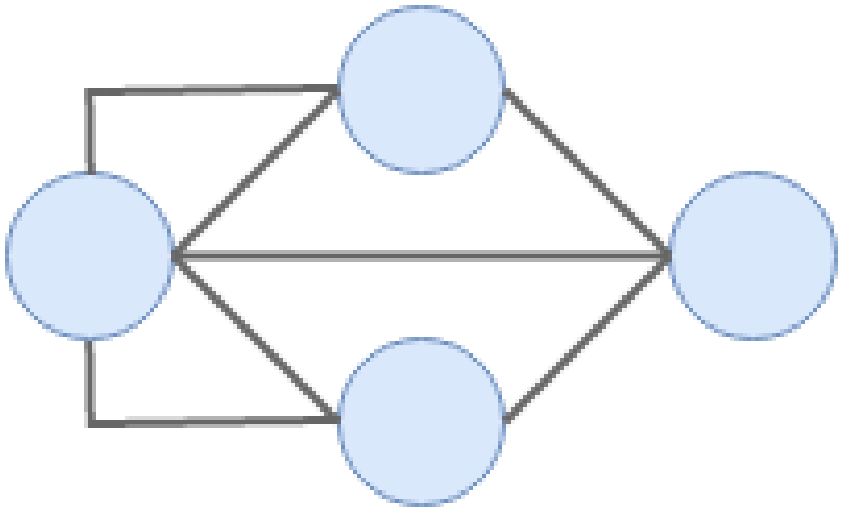
Un peu d'histoire

L'histoire de la théorie des graphes débute avec les travaux d'Euler au XVIIIe siècle.



Un peu d'histoire

L'histoire de la théorie des graphes débute avec les travaux d'Euler au XVIIIe siècle.



Définitions

Graphe

un graphe est un couple G = (V, E) tel que :

- V un ensemble de sommets (aussi appelés *nœuds* ou *vertex*);
- $E \subseteq \{\{x, y\} | (x, y) \in V^2 \wedge x \neq y\}$ un ensemble d'arêtes (aussi appelées *liens*), qui sont des paires de sommets (c.-à-d. qu'une arête est associée à deux sommets distincts).

Sommets et arêtes

Sommets

- Un élément simple;
- on le dessine habituellement par un point;
- **L'ensemble des sommets** de G est noté $V(G)$ et on note $n = |V(G)|$

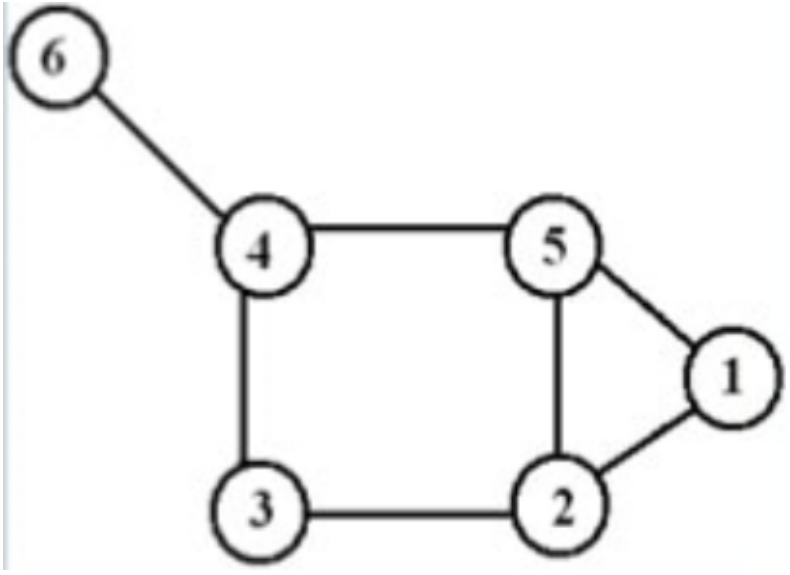
Arêtes

- Un ensemble de deux éléments;
- On le dessine habituellement par une ligne entre les deux points;
- **L'ensemble des arêtes** de G est noté $E(G)$ et on note $m = |V(G)|$

Le voisinage

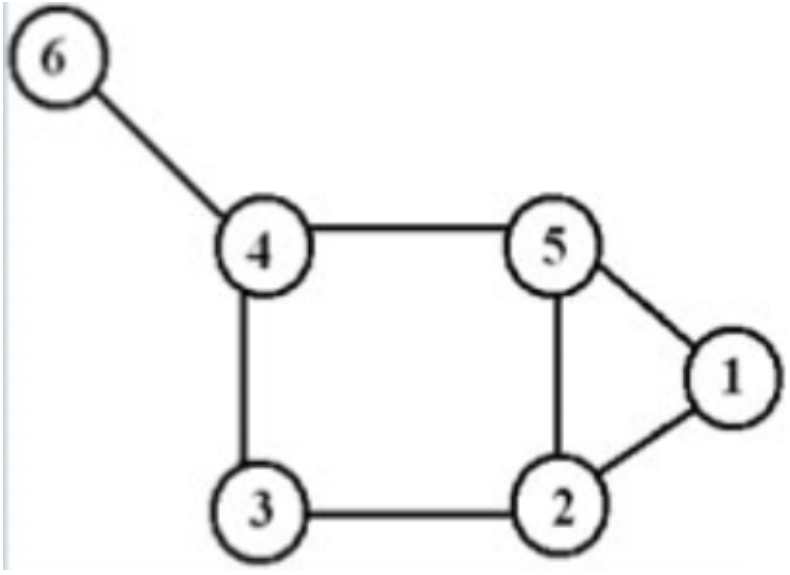
Pour chaque sommet v, l'ensemble des noeuds qui sont connectés par une arête à ce noeud v est appelé le voisinage et est noté $N(v)$

Exemple de graphe



- n = 6 ; m = 7
- V(G) = {1, 2, 3, 4, 5, 6}
- E(G) = { (1,2), (1,5), (2,3), (2,5), (3,4), (4,5), (4,6) }
- N(4) = ?

Exemple de graphe



- n = 6 ; m = 7
- V(G) = {1, 2, 3, 4, 5, 6}
- E(G) = { (1,2), (1,5), (2,3), (2,5), (3,4), (4,5), (4,6) }
- N(4) = {6, 5, 3}

Graphe simple (non dirigé)

Degré d'un sommet
On définit le degré d'un sommet $d(v)$ comme étant le nombre de voisins de V, on a donc : $d(v) = |N(v)|$

Boucle (*loop*)
Une boucle est une arête (v,v) , v étant un sommet.

Définition d'un graphe simple
Un graphe simple est un graphe non dirigé sans boucle.

Somme des degrés d'un graphe
Dans un graphe simple G, on a :

$$\sum_{v_i \in V(G)} deg(v_i) = ?$$

Graphe simple (non dirigé)

Degré d'un sommet
On définit le degré d'un sommet $d(v)$ comme étant le nombre de voisins de V, on a donc : $d(v) = |N(v)|$

Boucle (*loop*)
Une boucle est une arête (v,v) , v étant un sommet.

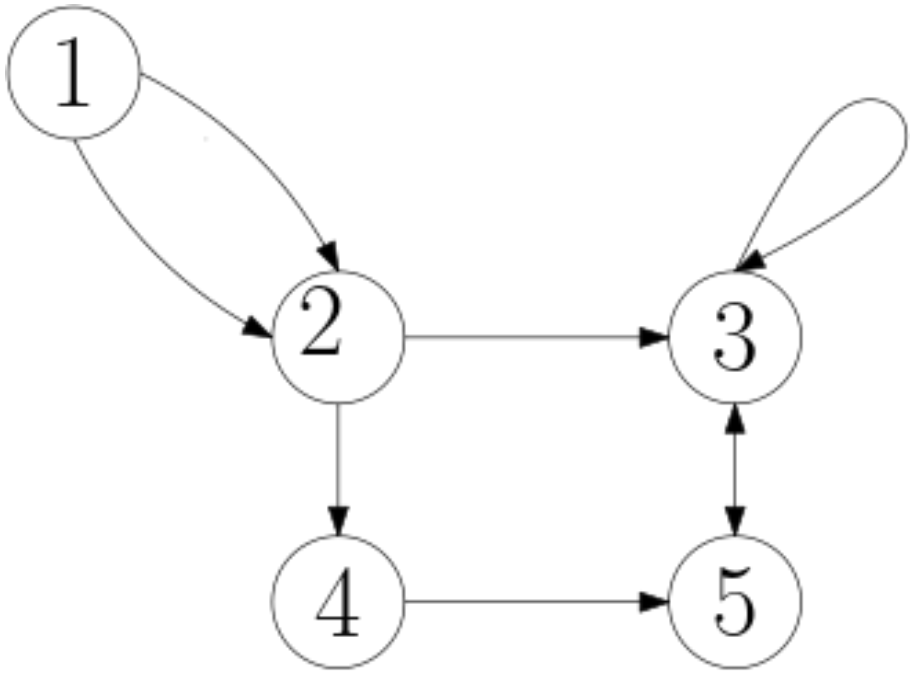
Définition d'un graphe simple
Un graphe simple est un graphe non dirigé sans boucle.

Somme des degrés d'un graphe
Dans un graphe simple G, on a :

$$\sum_{v_i \in V(G)} deg(v_i) = 2|E|$$

Graphe dirigé (*digraph*)

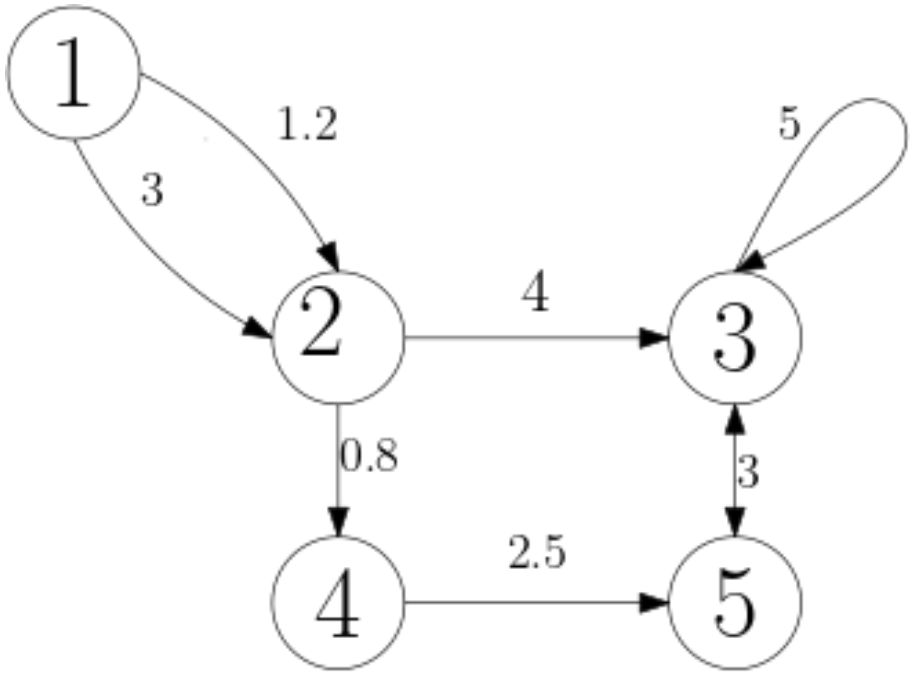
- Les arêtes sont orientées;
- les boucles sont autorisées;
- les multi-arêtes sont autorisées.



Les graphes pondérés

Définition

Un graphe est dit pondéré si chaque arête à un poids associé.

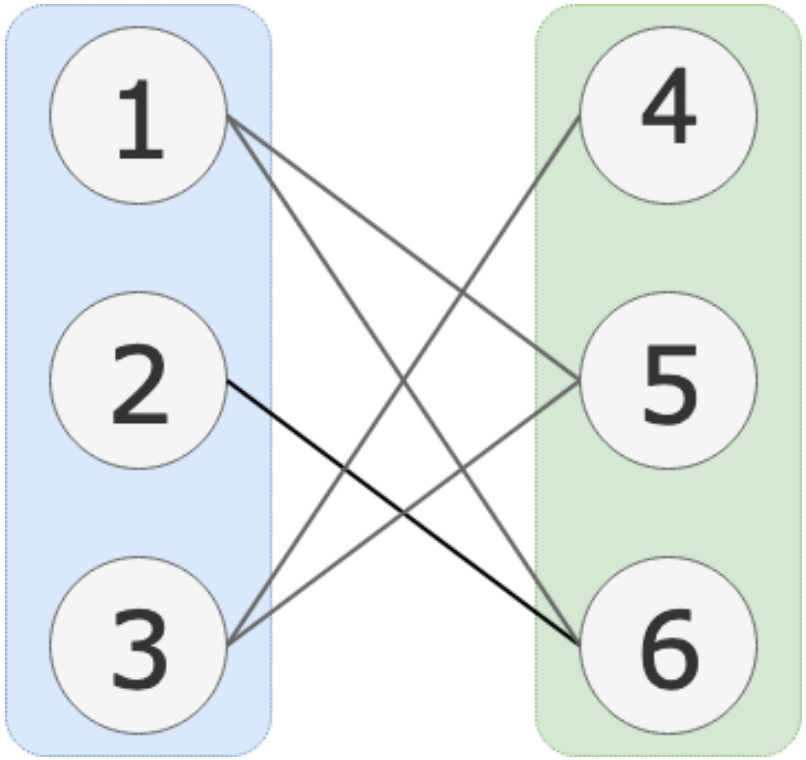


Graphe biparti

Définition

Un graphe est dit **biparti** si l'ensemble des sommets V peut être partitionné en deux ensembles V_1 et V_2 tel que $(u, v) \in E$ implique :

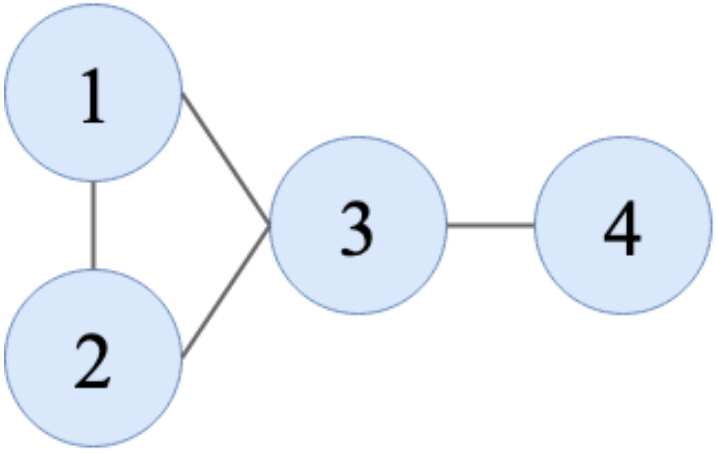
- soit $u \in V_1$ et $v \in V_2$
- soit $v \in V_1$ et $u \in V_2$

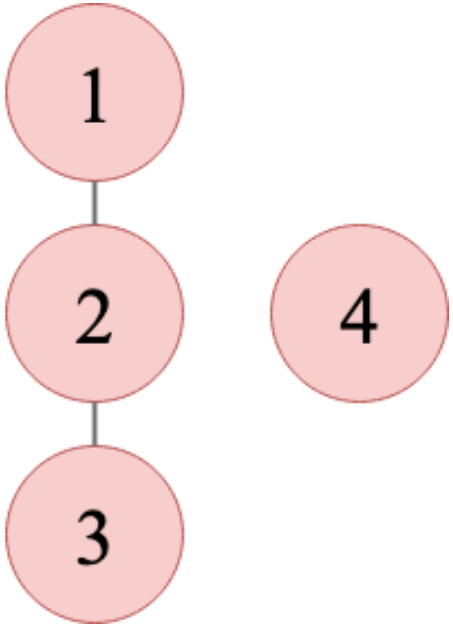


Quelques définitions

Graphe connexe

Graphe dans lequel on peut relier, directement ou non, n'importe quel sommet à n'importe quel autre sommet du graphe par une chaîne d'arêtes.





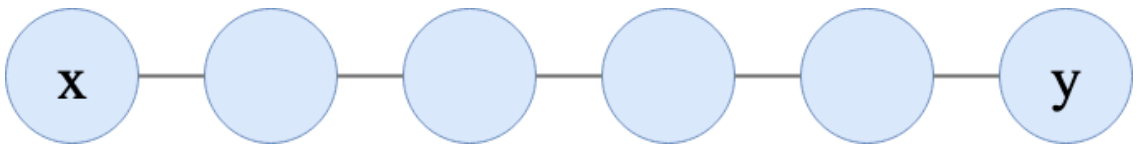
Quelques définitions

Chaîne

une **chaîne** reliant x à y, notée μ(x,y), est définie par une suite finie d'arêtes consécutives, reliant x à y.

Vocabulaire

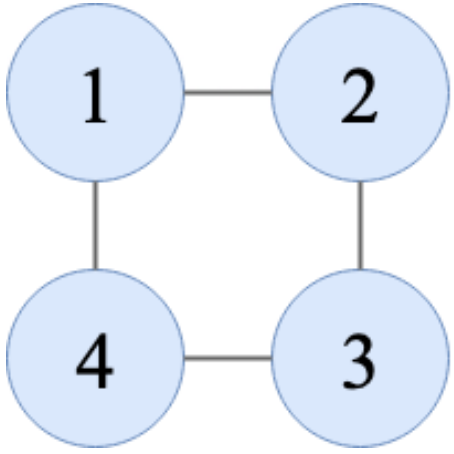
- Une **chaîne élémentaire** est une chaîne ne passant pas deux fois par un même sommet, c'est-à-dire dont tous les sommets sont distincts;
- Une **chaîne simple** est une chaîne ne passant pas deux fois par une même arête, c'est-à-dire dont toutes les arêtes sont distinctes.

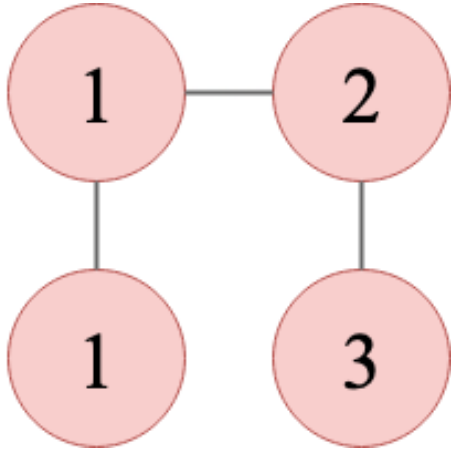


Quelques définitions

Cycle

Dans un graphe non orienté, un **cycle** est une suite d'arêtes consécutives dont les deux sommets extrémités sont identiques.

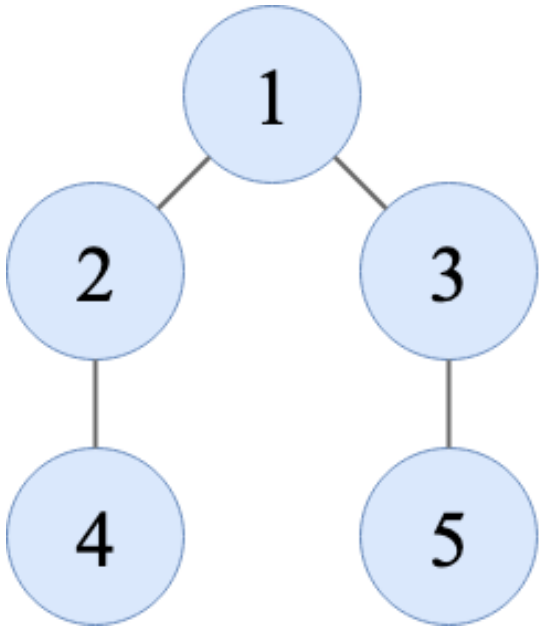


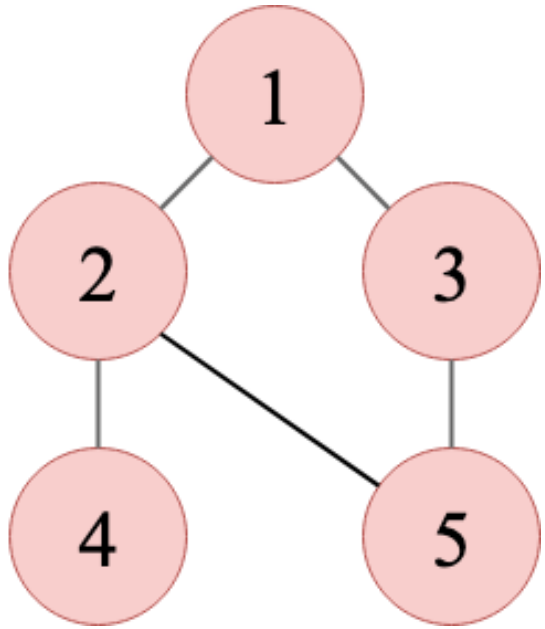


Note: Un **cycle** est une chaîne simple dont les deux extrémités sont identiques.

Définition

Un graphe simple est un **arbre** si il est connexe et sans cycle.

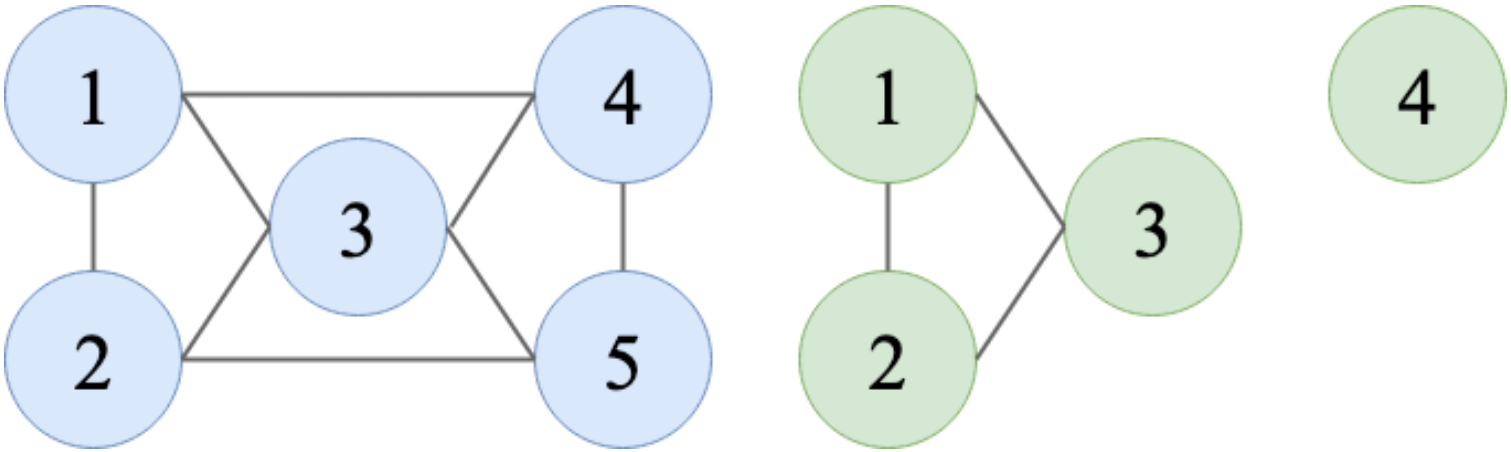




Sous-graphe

Définition

Un **sous-graphe** G' d'un graphe G contient un sous-ensemble des sommets et des arêtes de G.



Graphe planaire

Définition

- Une **représentation** d'un graphe G est une fonction qui associe chaque sommet $v \in V(G)$) un point $f(v)$ dans le plan et chaque arête uv à un courbe entre $f(u)$ et $f(v)$;

Graphe planaire

Définition

- Une **représentation** d'un graphe G est une fonction qui associe chaque sommet $v \in V(G)$) un point $f(v)$ dans le plan et chaque arête uv à un courbe entre $f(u)$ et $f(v)$;
- Un point $f(e) \cap f(e')$ autre qu'un point commun est un **croisement**;

Graphe planaire

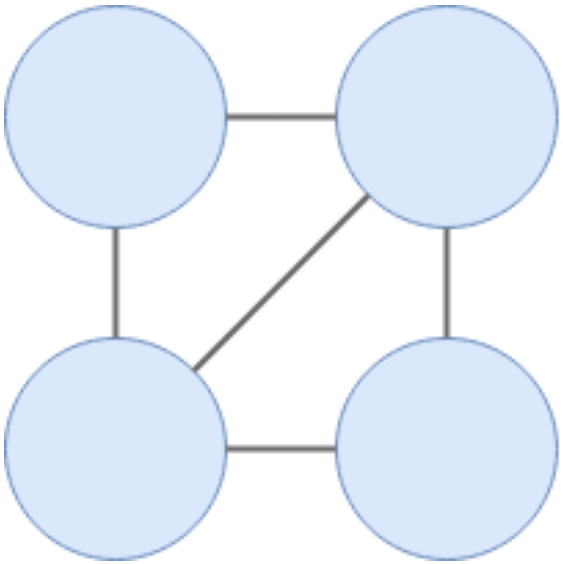
Définition

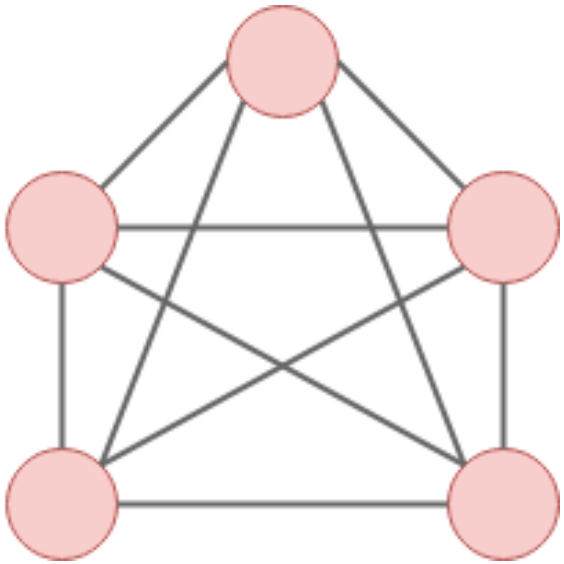
- Une **représentation** d'un graphe G est une fonction qui associe chaque sommet $v \in V(G)$) un point $f(v)$ dans le plan et chaque arête uv à un courbe entre $f(u)$ et $f(v)$;
- Un point $f(e) \cap f(e')$ autre qu'un point commun est un **croisement**;
- Un graphe est **planaire** si il a une représentation sans croisement

Graphe planaire

Définition

- Une **représentation** d'un graphe G est une fonction qui associe chaque sommet $v \in V(G)$ un point $f(v)$ dans le plan et chaque arête uv à un courbe entre $f(u)$ et $f(v)$;
- Un point $f(e) \cap f(e')$ autre qu'un point commun est un **croisement**;
- Un graphe est **planaire** si il a une représentation sans croisement





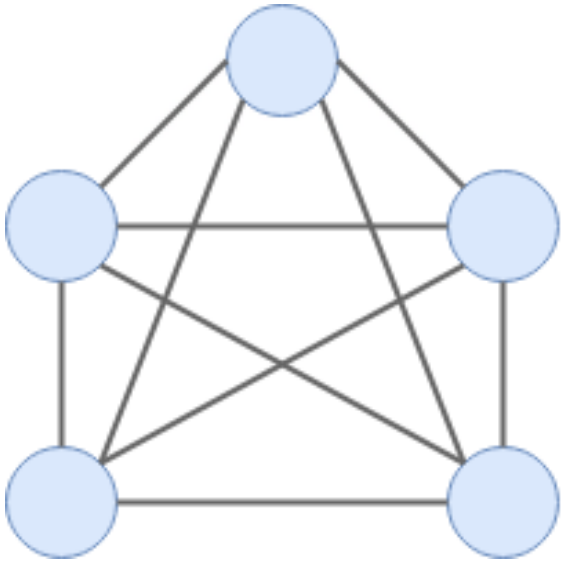
Graphe complet

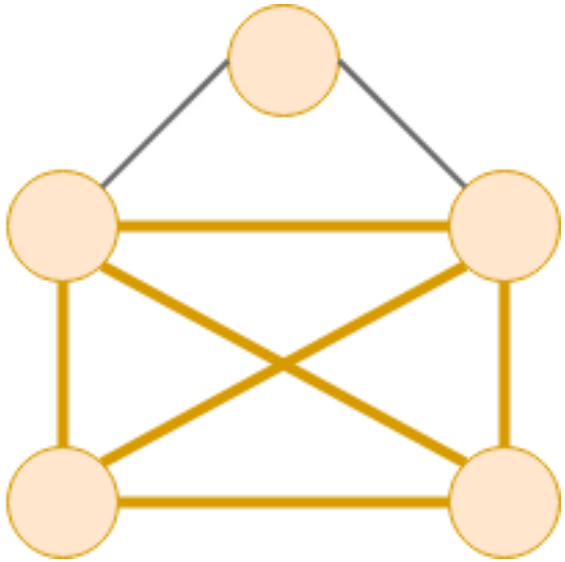
Définition

Un graphe **complet** est un graphe dont tous les sommets sont adjacents.

À *isomorphisme* près, il n'existe qu'un seul graphe complet non orienté d'ordre n, que l'on note $\mathcal{K}_n$.

on appelle **clique** un sous-ensemble de sommets induisant un sous-graphe complet de G.





Graphe eulérien

Définition

Un **parcours eulérien** d'un graphe non orienté est un chemin qui passe par toutes les arêtes, une fois par arête. Si un tel chemin revient au sommet de départ, on parle de **circuit eulérien**.

Un graphe qui admet un circuit eulérien est dit eulérien.

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

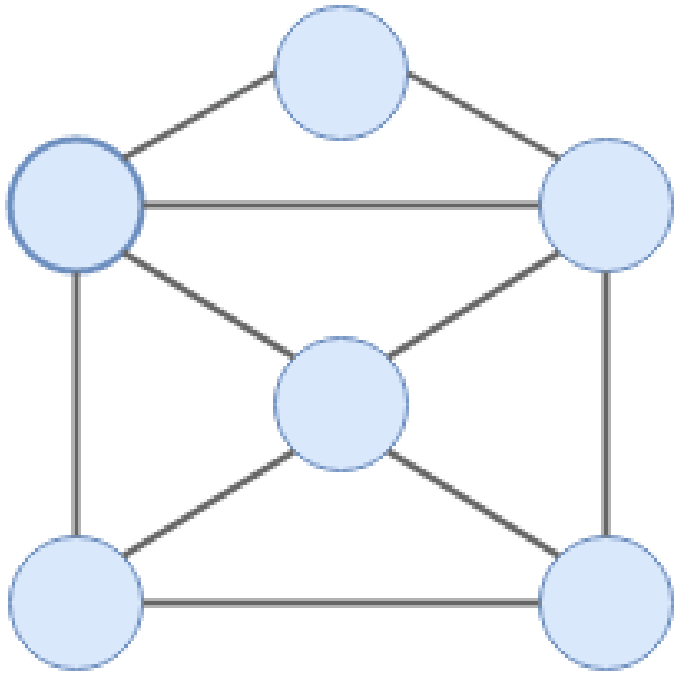
Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.

Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.

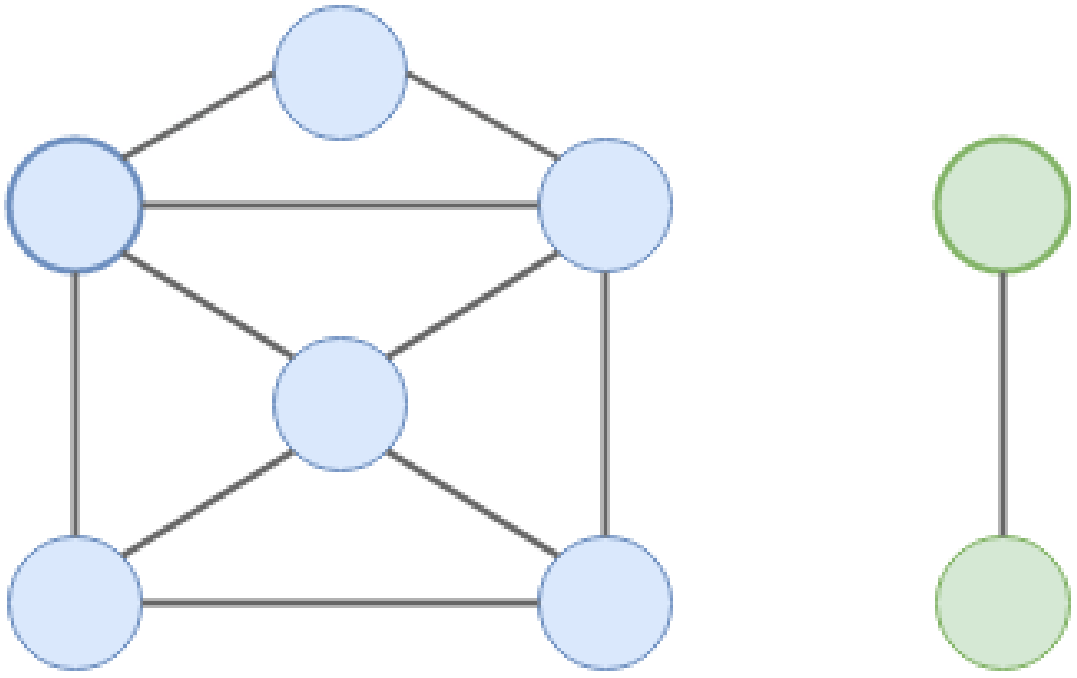


Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.

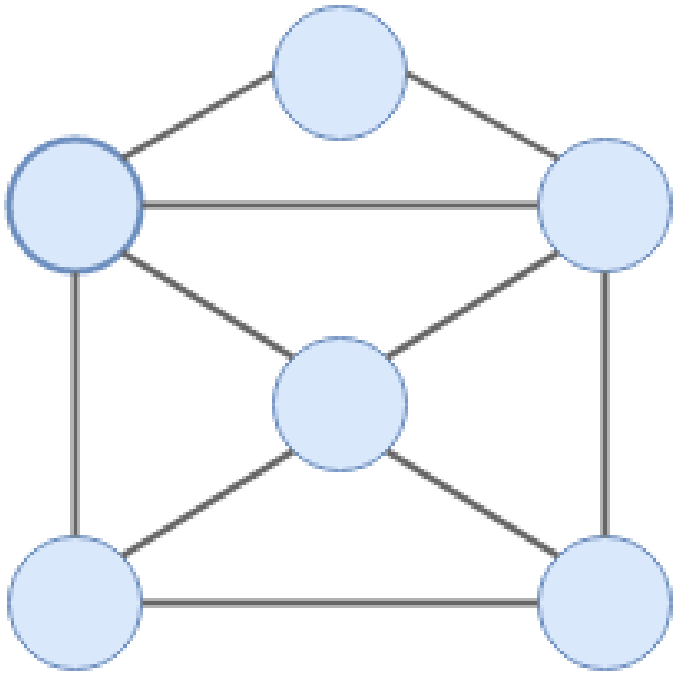


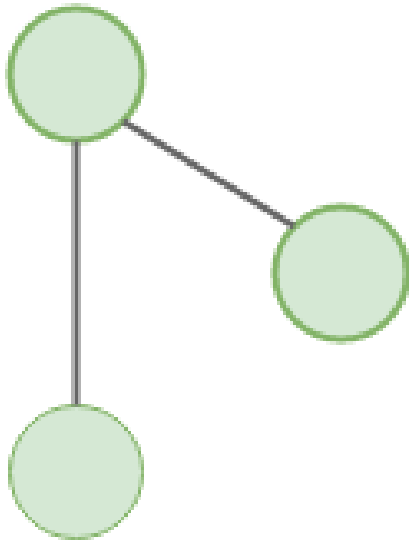
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



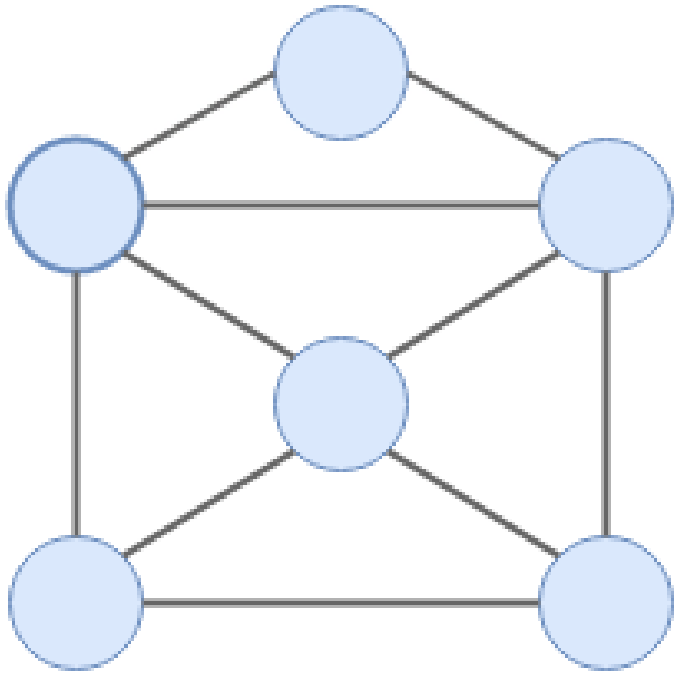


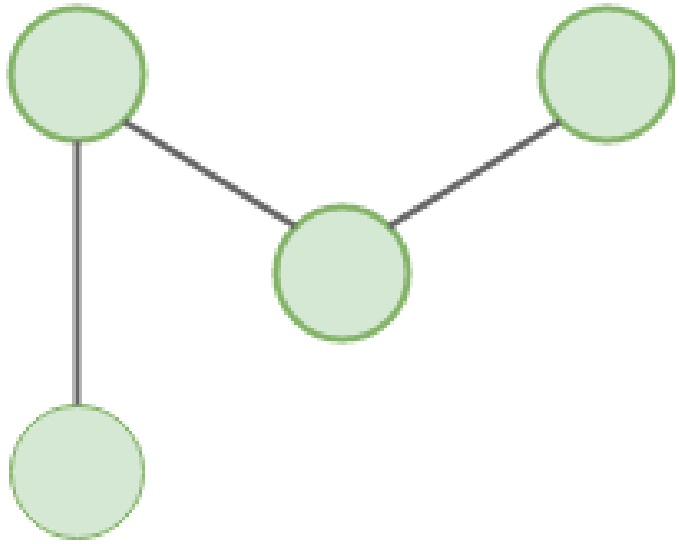
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



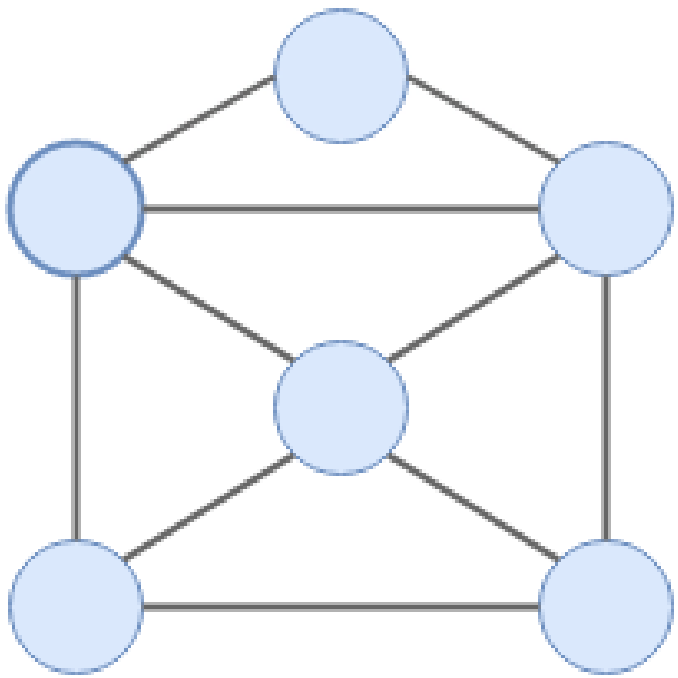


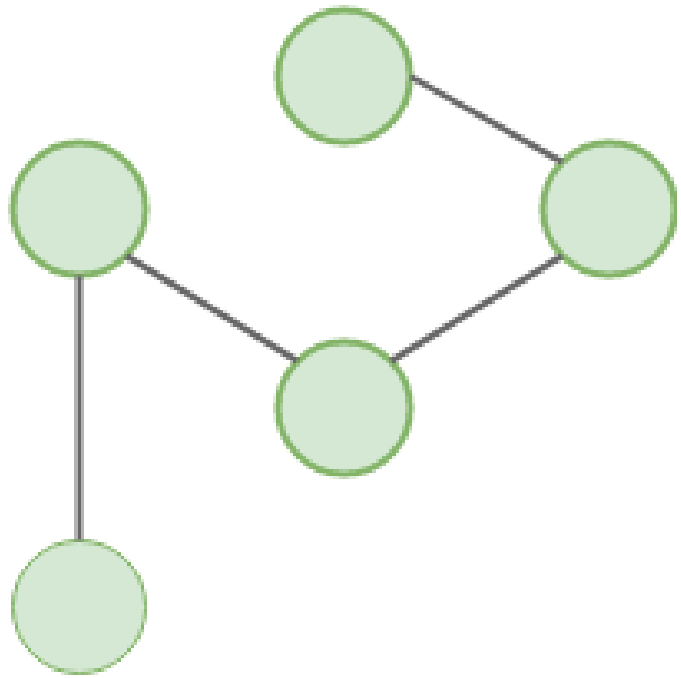
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



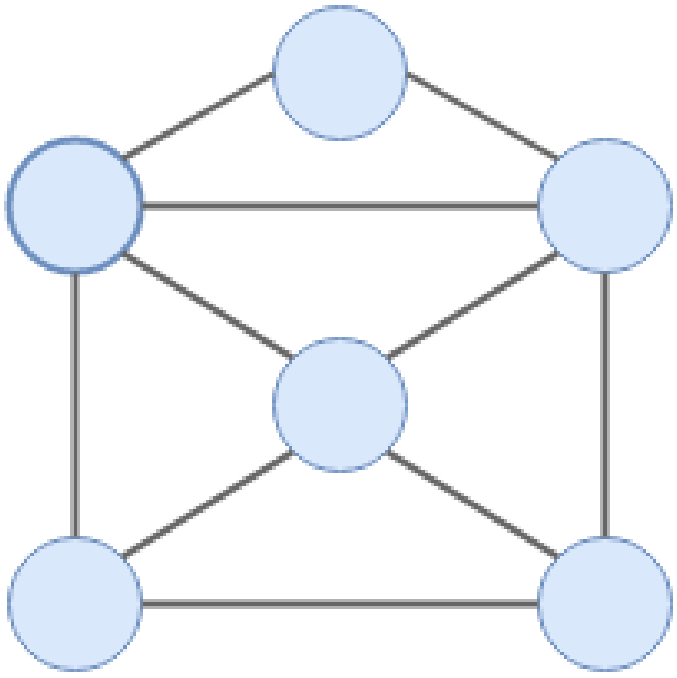


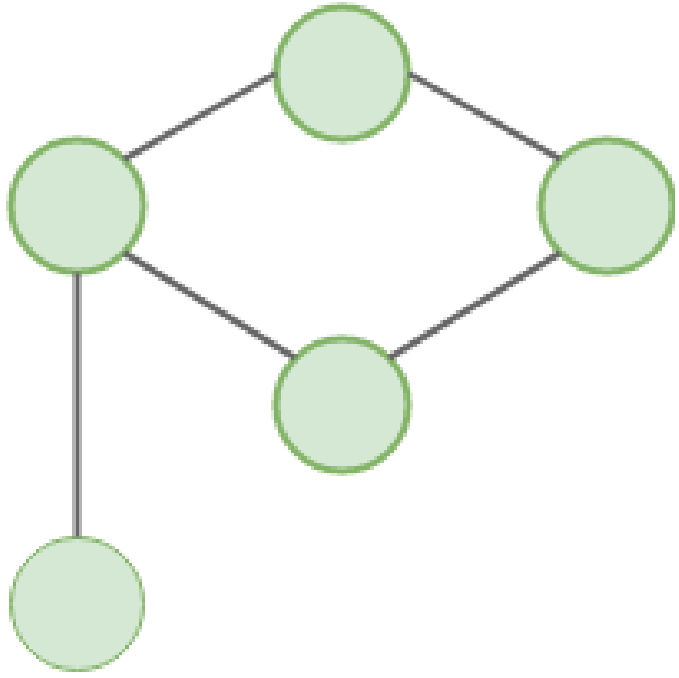
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



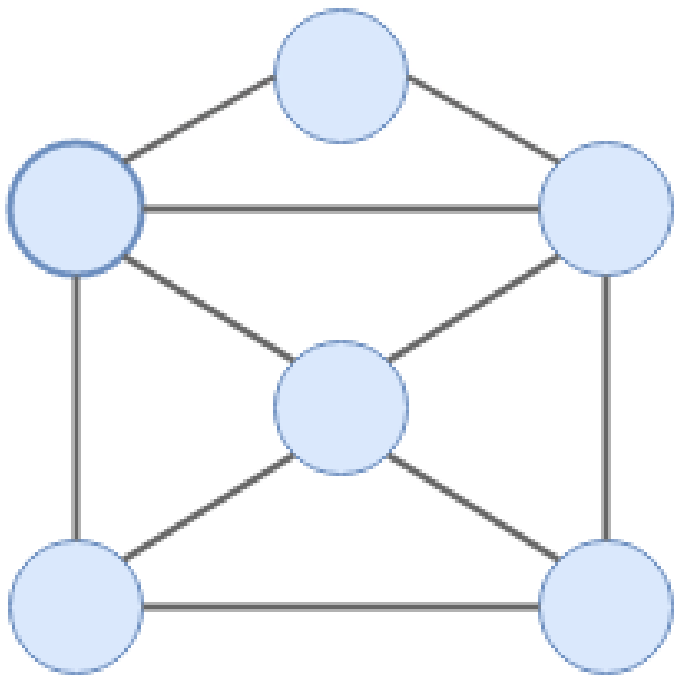


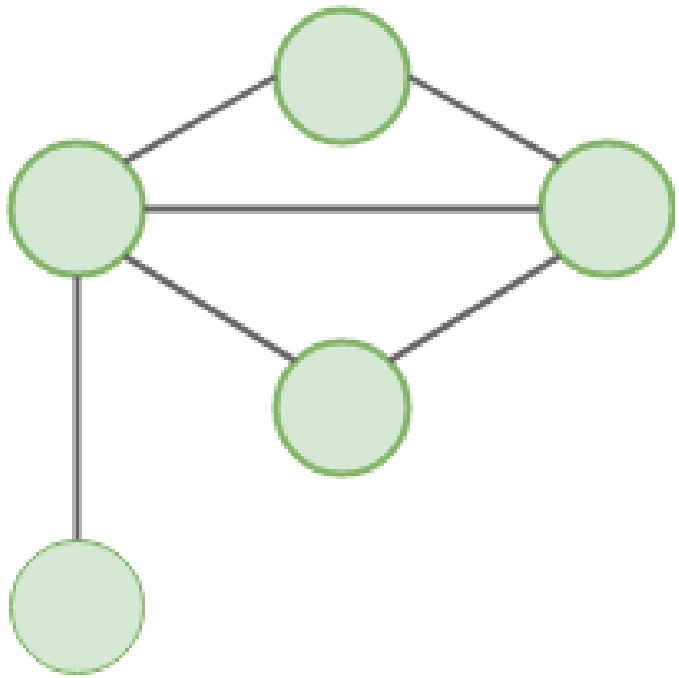
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



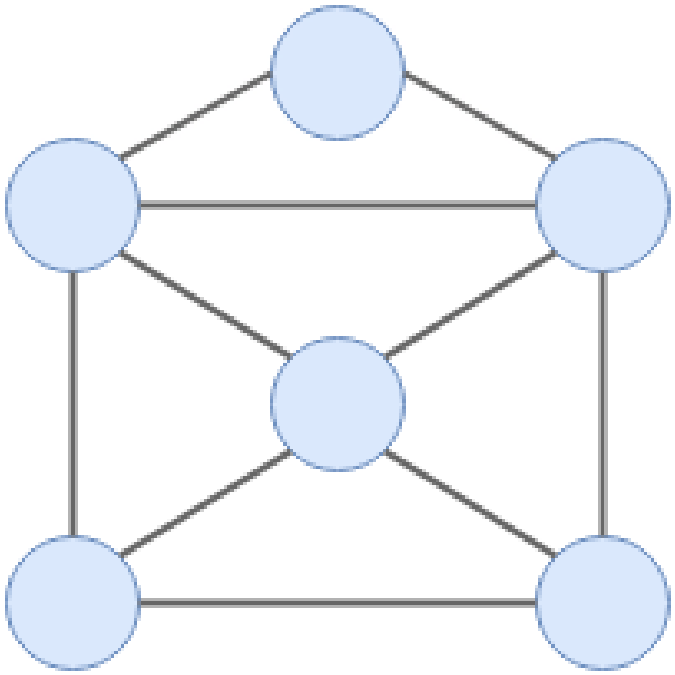


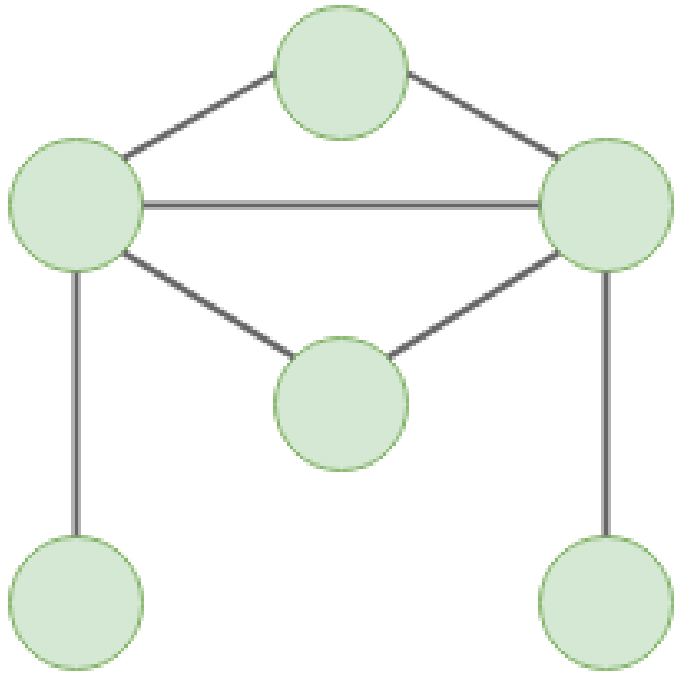
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



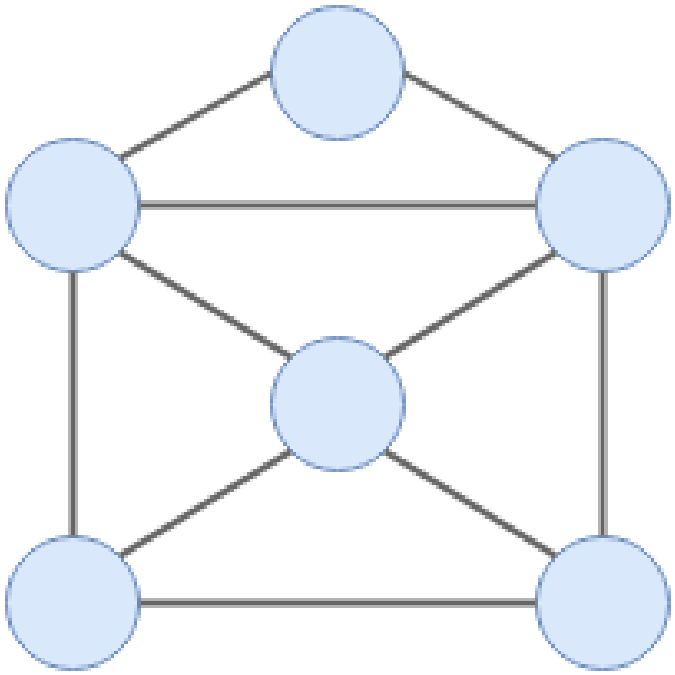


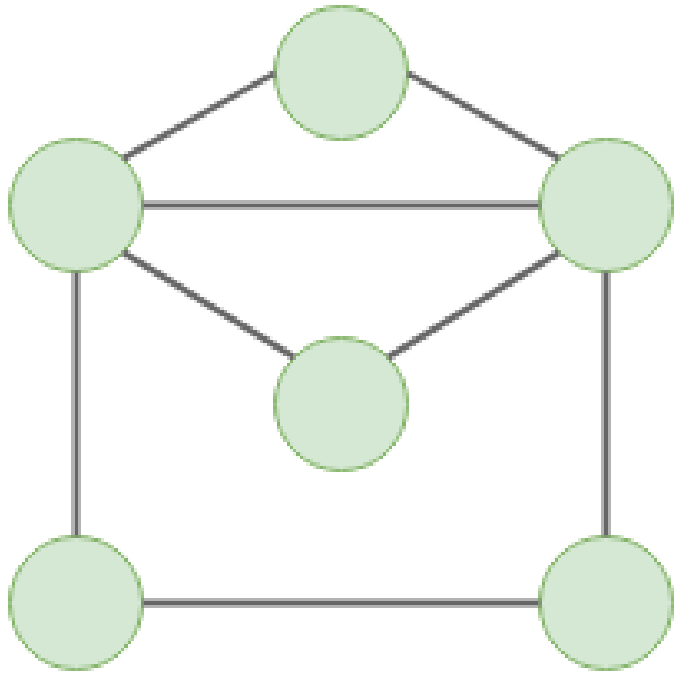
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



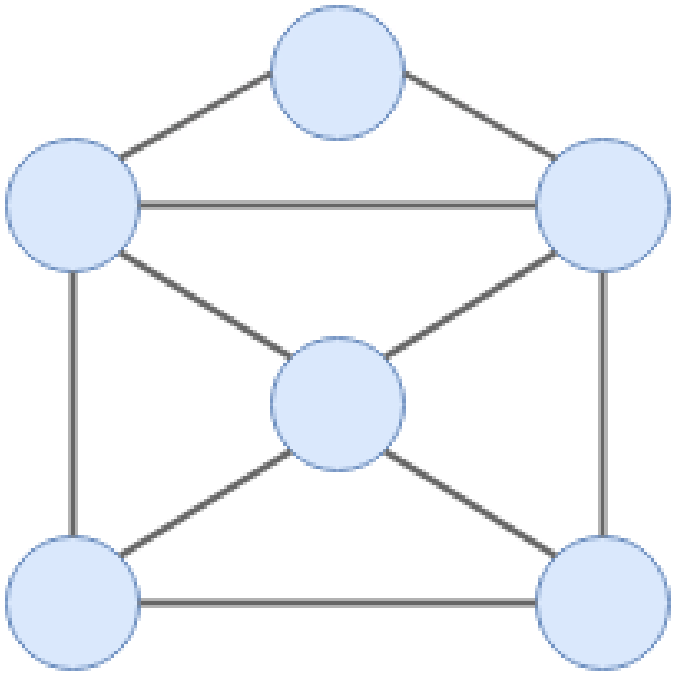


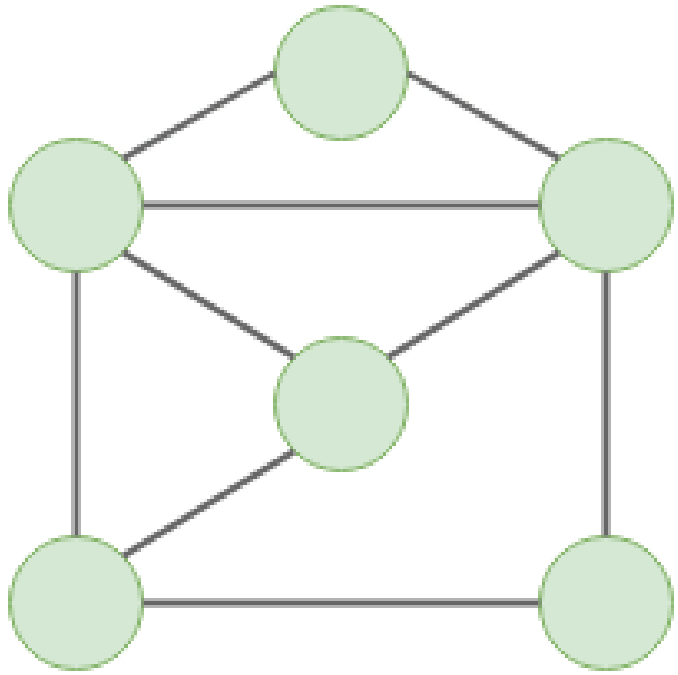
Graphe eulérien

Intuition

La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.



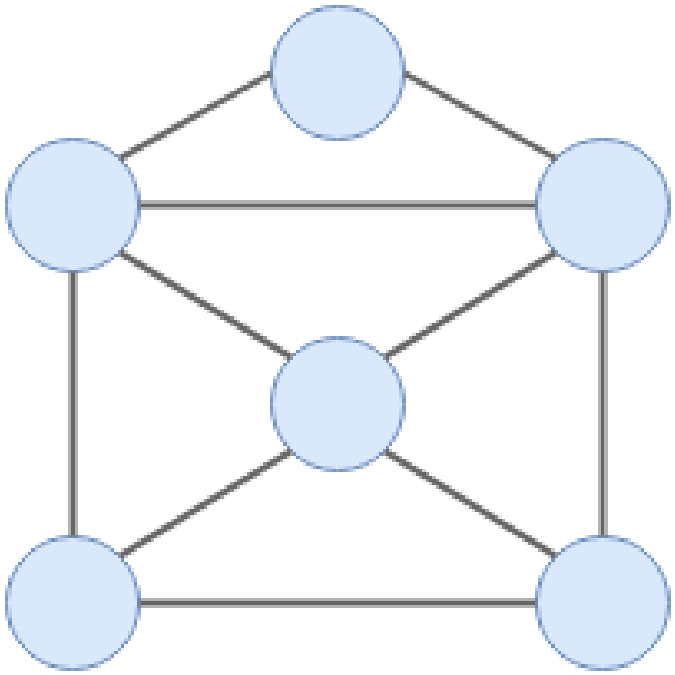


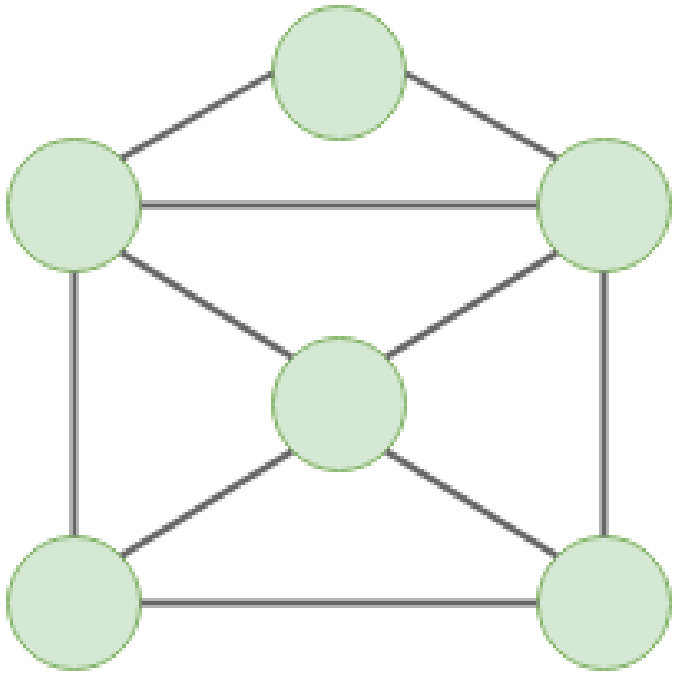
Graphe eulérien

Intuition

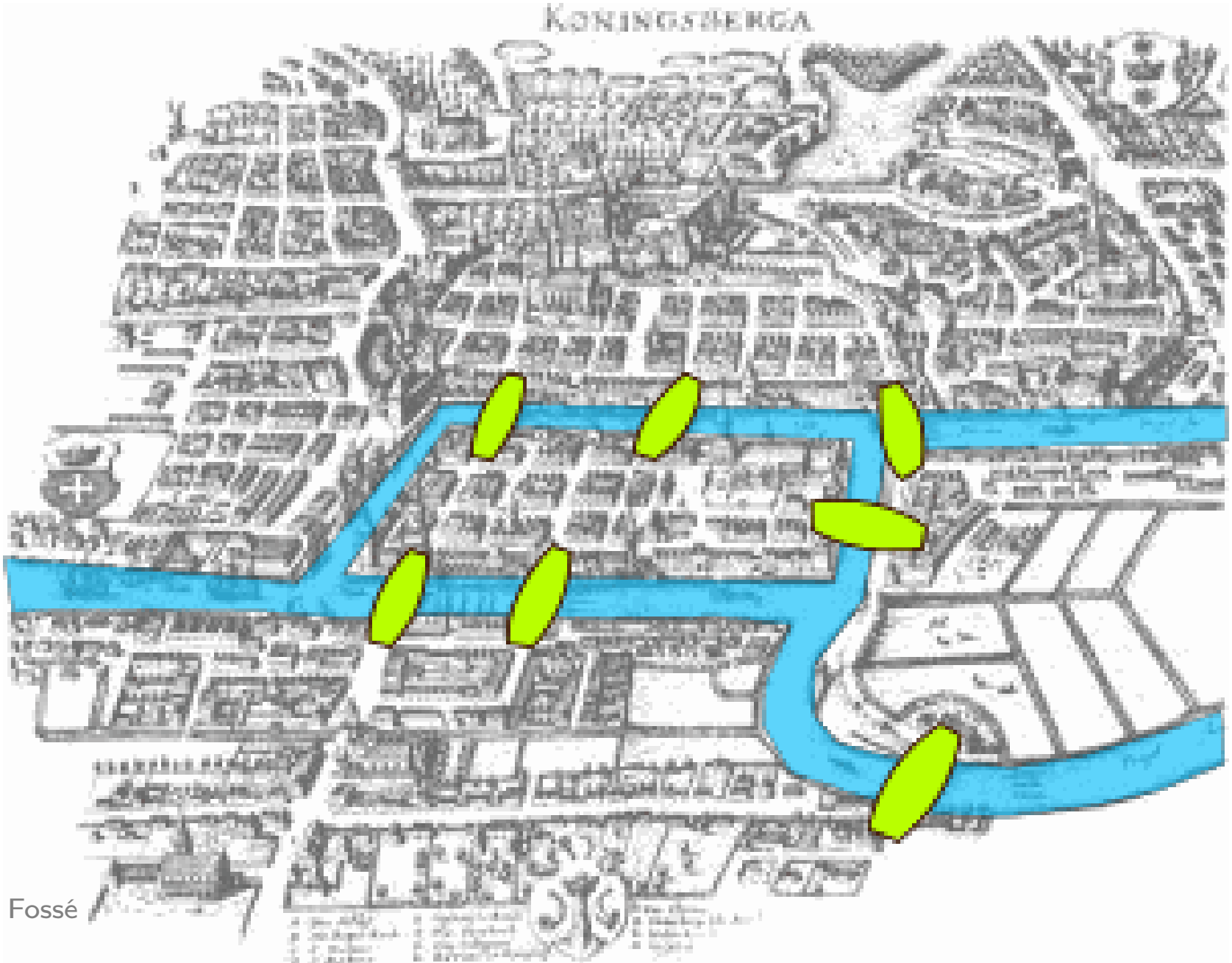
La notion de parcours eulérien s'illustre avec le problème du **dessin de l'enveloppe**. Il s'agit de tracer une enveloppe sans lever le crayon et sans dessiner plusieurs fois un même trait.

Un parcours eulérien correspond à un tracé d'un graphe sur une feuille sans lever le crayon.

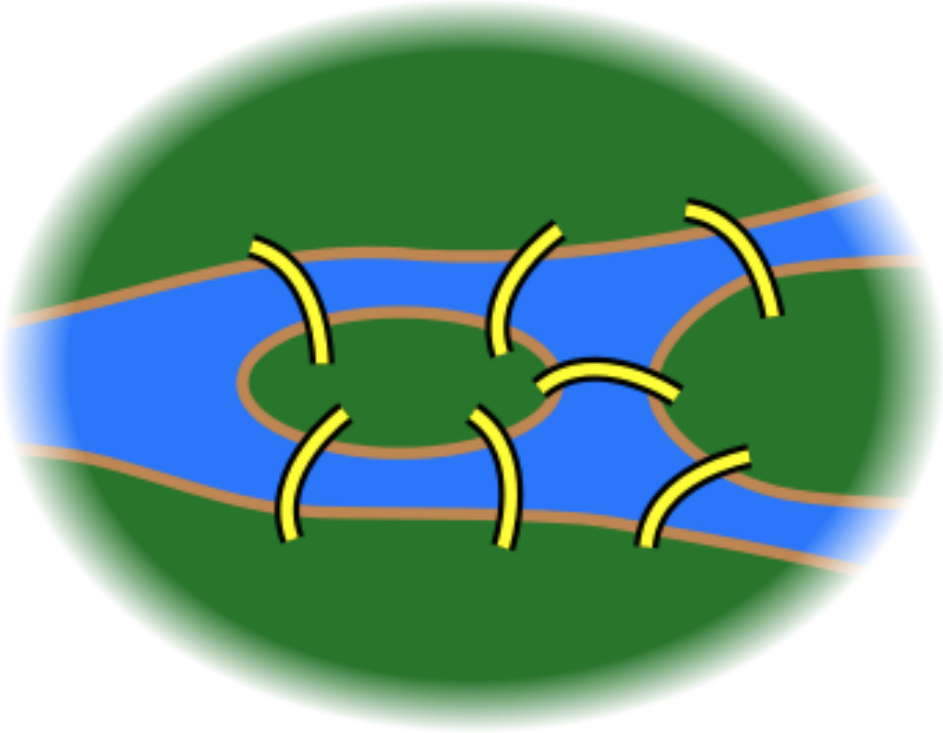




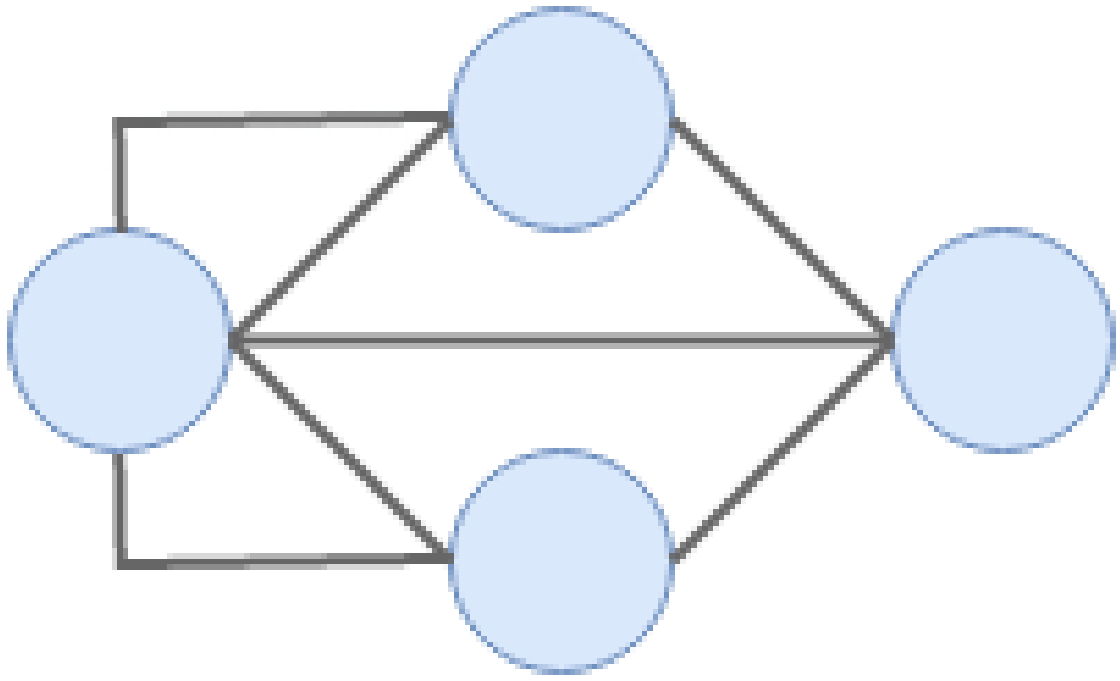
Et notre histoire de ponts ?



Et notre histoire de ponts ?



Et notre histoire de ponts ?



Est-ce-que ce graphe est eulérien ?

Théorème d'Euler

Théorème

Le théorème d'Euler se décline en deux caractérisations :

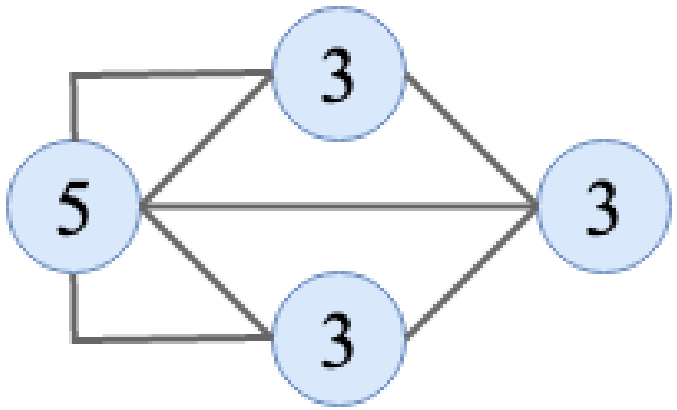
- Un graphe connexe admet un **parcours eulérien** si et seulement si ses sommets sont tous de degré pair sauf au plus deux.
- Un graphe connexe admet un **circuit eulérien** si et seulement si tous ses sommets sont de degré pair.

Théorème d'Euler

Théorème

Le théorème d'Euler se décline en deux caractérisations :

- Un graphe connexe admet un **parcours eulérien** si et seulement si ses sommets sont tous de degré pair sauf au plus deux.
- Un graphe connexe admet un **circuit eulérien** si et seulement si tous ses sommets sont de degré pair.



Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Un peu d'histoire

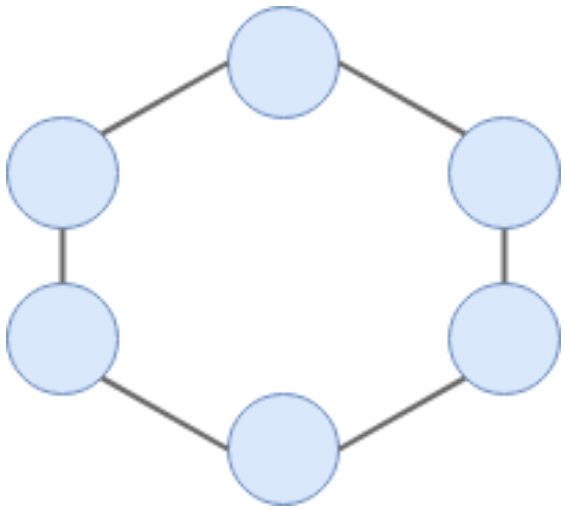
Au IX^e siècle, le poète indien *Rudrata* a été le premier à trouver le problème des cycles/chemins hamiltoniens sur un échiquier mais avec un cavalier. Il s'agit du problème du cavalier.

Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Quelques exemples

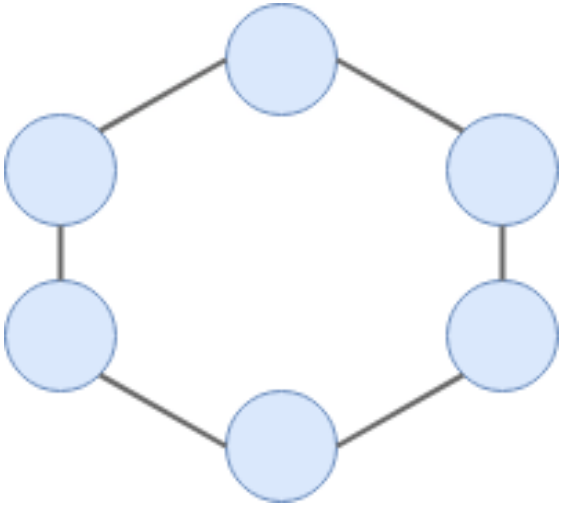


Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Quelques exemples



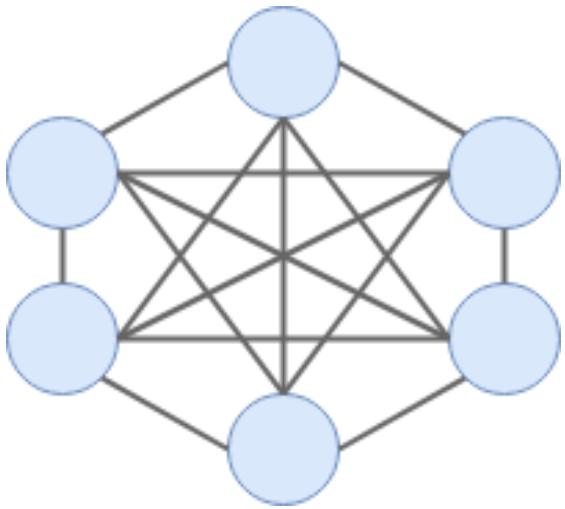
☺

Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Quelques exemples

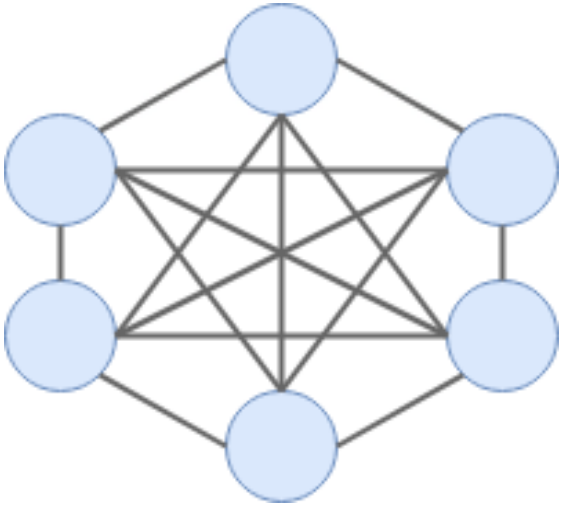


Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Quelques exemples



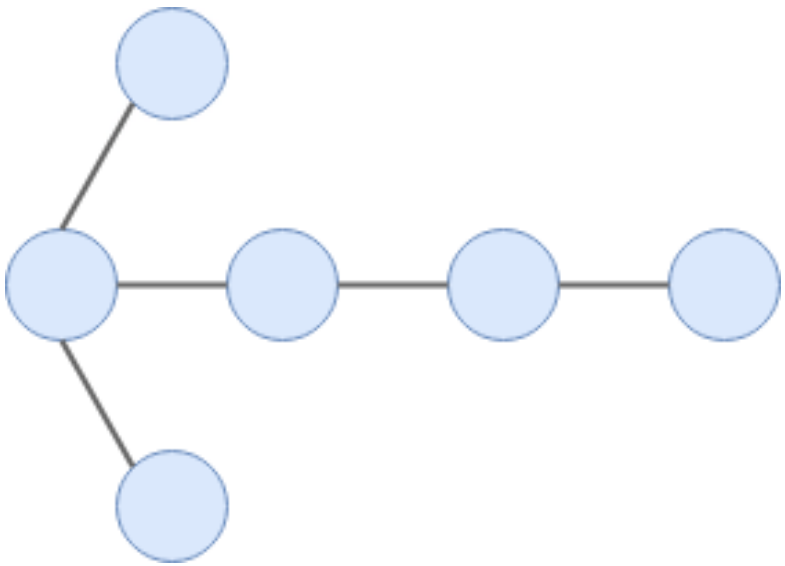
☺

Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Quelques exemples

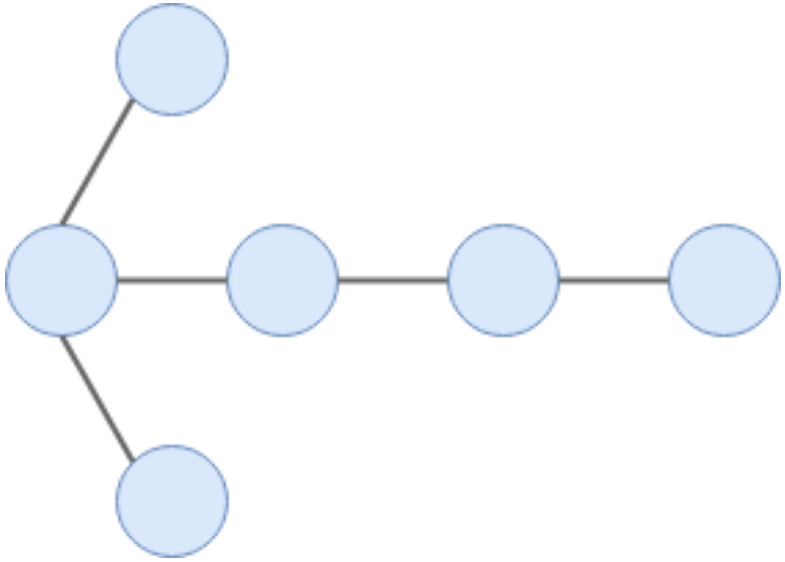


Graphe hamiltonien

Définitions

- Une **chaîne hamiltonienne** est une chaîne qui passe une fois et une seule par chaque sommet du graphe.
- Un **cycle hamiltonien** est une chaîne hamiltonienne qui est cyclique.

Quelques exemples



☹

Caractérisation des graphes hamiltoniens

Il n'existe pas de caractérisation pour les graphes hamiltoniens !

Caractérisation des graphes hamiltoniens

Théorème de Dirac (1952)
Un graphe simple à n sommets ($n \geq 3$) dont chaque sommet est au moins de degré $\frac{n}{2}$ est hamiltonien.

Caractérisation des graphes hamiltoniens

Théorème de Dirac (1952)
Un graphe simple à n sommets ($n \geq 3$) dont chaque sommet est au moins de degré $\frac{n}{2}$ est hamiltonien.

Théorème de Ore (1960)
Un graphe simple à n sommets ($n \geq 3$) tel que la somme des degrés de toute parie de sommets non adjacents vaut au moins n est hamiltonien.

Le théorème de *Dirac* est un cas particulier du théorème de *Ore* : en effet, si chaque sommet est de degré au moins $\frac{n}{2}$, alors la somme des degrés de n'importe quelle paire de sommets, adjacents ou pas, est au moins n.

Caractérisation des graphes hamiltoniens

Théorème de Ore (1960)
Un graphe simple à n sommets ($n \geq 3$) tel que la somme des degrés de toute parie de sommets non adjacents vaut au moins n est hamiltonien.

Théorème de Pósa (1962)
Un graphe simple à n sommets ($n \geq 3$) tel que :

1. pour tout entier k tel que $1 \leq k < \frac{n-1}{2}$ le nombre de sommets de degré inférieur ou égal à k est inférieur à k
2. le nombre de sommets de degré inférieur ou égal à $\frac{n-1}{2}$ est inférieur ou égal à $\frac{n-1}{2}$

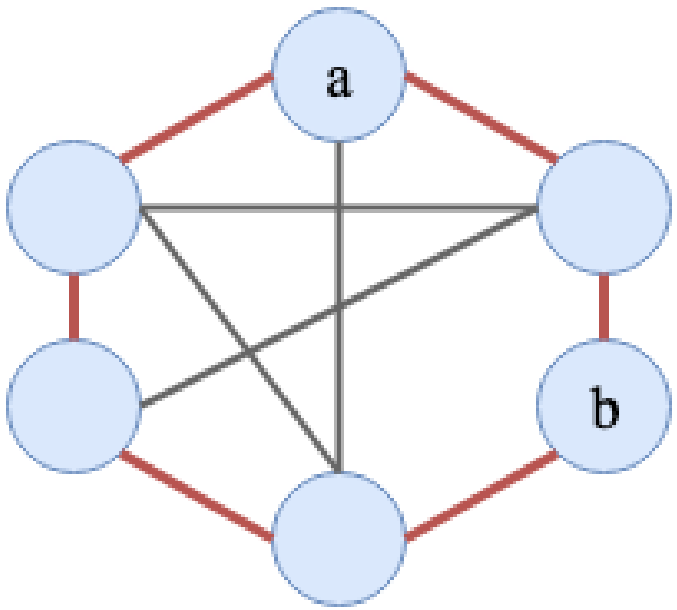
est hamiltonien.

On remarque que la condition 2 est comprise dans la condition 1 quand n est pair, et n'a donc d'intérêt que lorsque n est impair.

Prenons un exemple

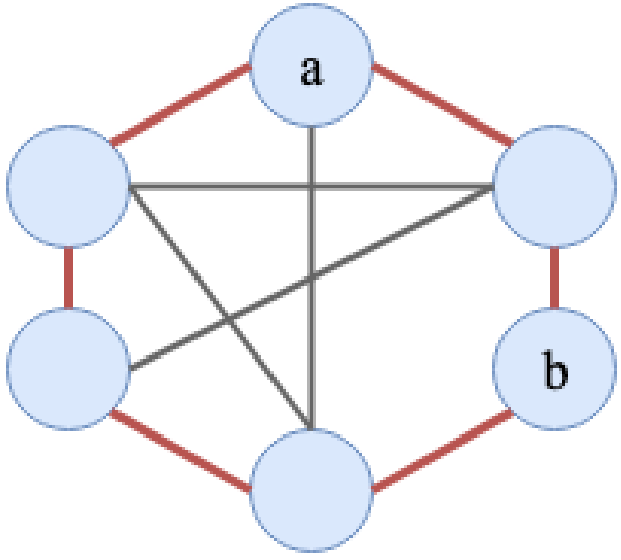
Exemple

Le graphe ci-contre possède 6 sommets. Il est hamiltonien : les sommets ont été ordonnés de manière à mettre en évidence un cycle hamiltonien, c'est le cycle extérieur rouge.



Prenons un exemple

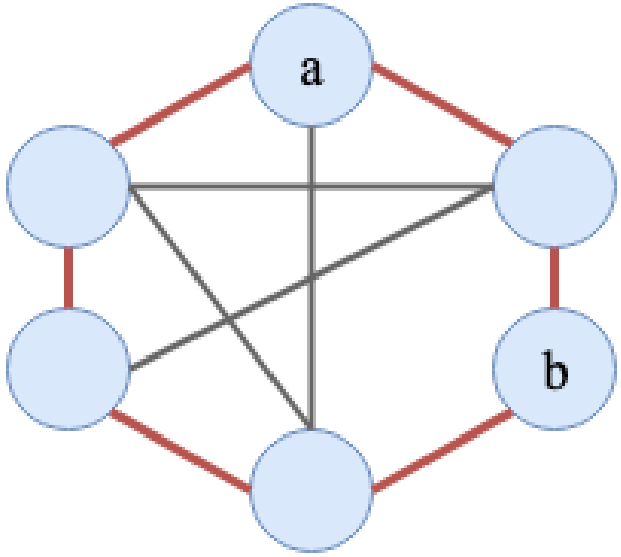
Théorème de Dirac (1952)
Un graphe simple à n sommets ($n \geq 3$) dont chaque sommet est au moins de degré $\frac{n}{2}$ est hamiltonien.



Le théorème de *Dirac* ne permet pas de prouver qu'il est hamiltonien.
Pour cela, tous les sommets devraient au moins être de degré 3 , or il y a un sommet de degré 2.

Prenons un exemple

Théorème de Ore (1960)
Un graphe simple à n sommets ($n \geq 3$) tel que la somme des degrés de toute parie de sommets non adjacents vaut au moins n est hamiltonien.

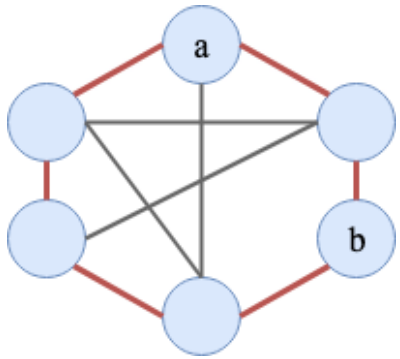


Le théorème de *Ore* n'est pas plus utile, car les sommets non adjacents a et b vérifient deg(a) + deg(b) = 3 + 2 = 5, or on devrait avoir au moins 6.

Prenons un exemple

Théorème de Pósa (1962)

1. pour tout entier k tel que $1 \leq k < \frac{n-1}{2}$ le nombre de sommets de degré inférieur ou égal à k est inférieur à k
2. le nombre de sommets de degré inférieur ou égal à $\frac{n-1}{2}$ est inférieur ou égal à $\frac{n-1}{2}$



En revanche, le théorème de *Pósa* permet de déterminer que le graphe est hamiltonien, car il y a 0 sommet de degré 1 et 1 sommet de degré 2: la condition 1 est remplie ($0 < 1$ et $0 + 1 < 2$).

Algorithme de Dijkstra

Problème du plus court chemin

Énoncé du problème

Étant donné un graphe non-orienté, dont les arêtes sont munies de poids, et deux sommets de ce graphe, trouver un chemin entre les deux sommets dans le graphe, de poids minimum.

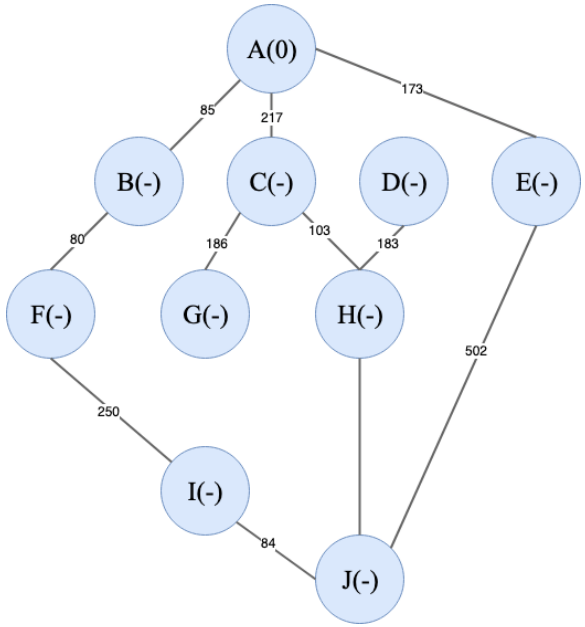
Principe de l'algorithme

1. On choisit le sommet accessible de distance minimale comme sommet à explorer;
2. A partir de ce sommet, on explore ses voisins et on met à jour les distances pour chacun. On ne met à jour la distance que si elle est inférieure à celle que l'on avait auparavant;
3. On répète jusqu'à ce que l'on arrive au point d'arrivée ou jusqu'à ce que tous les sommets aient été explorés.

Prenons un exemple

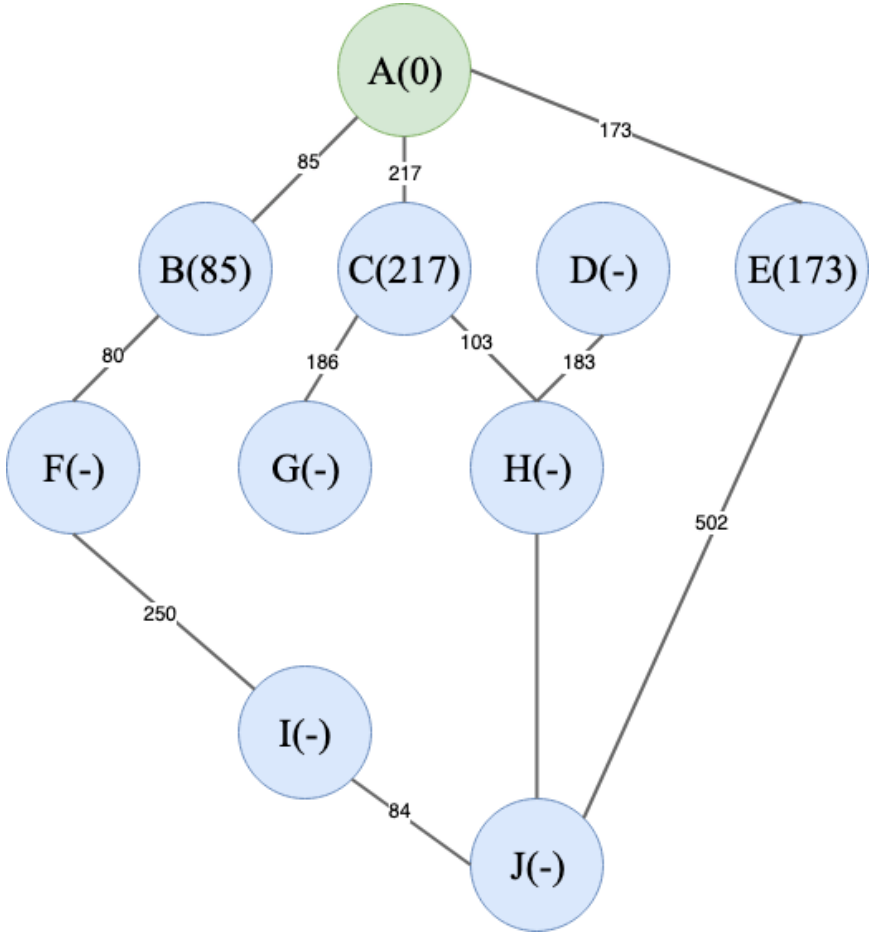
Énoncé de l'exemple

L'exemple ci-contre montre les étapes successives dans la résolution du chemin le plus court dans un graphe. Les nœuds symbolisent des villes identifiées par une lettre et les arêtes indiquent la distance entre ces villes. On cherche à déterminer le plus court trajet pour aller de la ville A à la ville J.



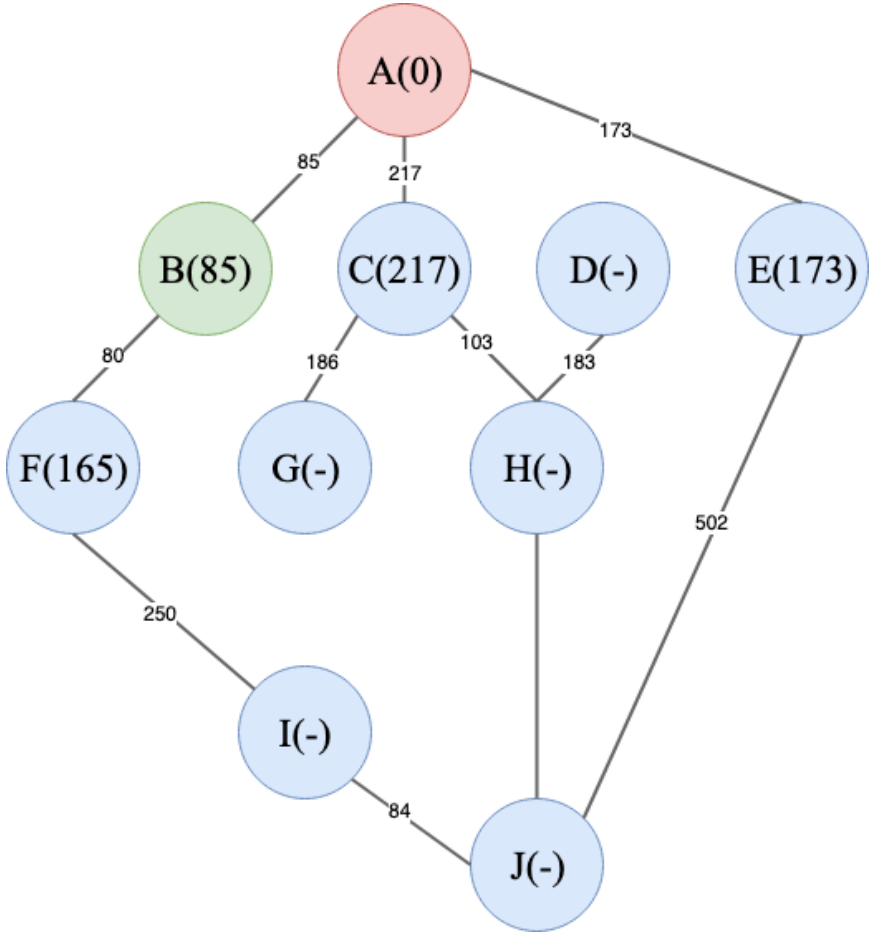
Prenons un exemple

Étape 1: on choisit la ville A. On met à jour les villes voisines de A qui sont B, C, et E. Leurs distances deviennent respectivement 85, 217, 173, tandis que les autres villes restent à une distance infinie.



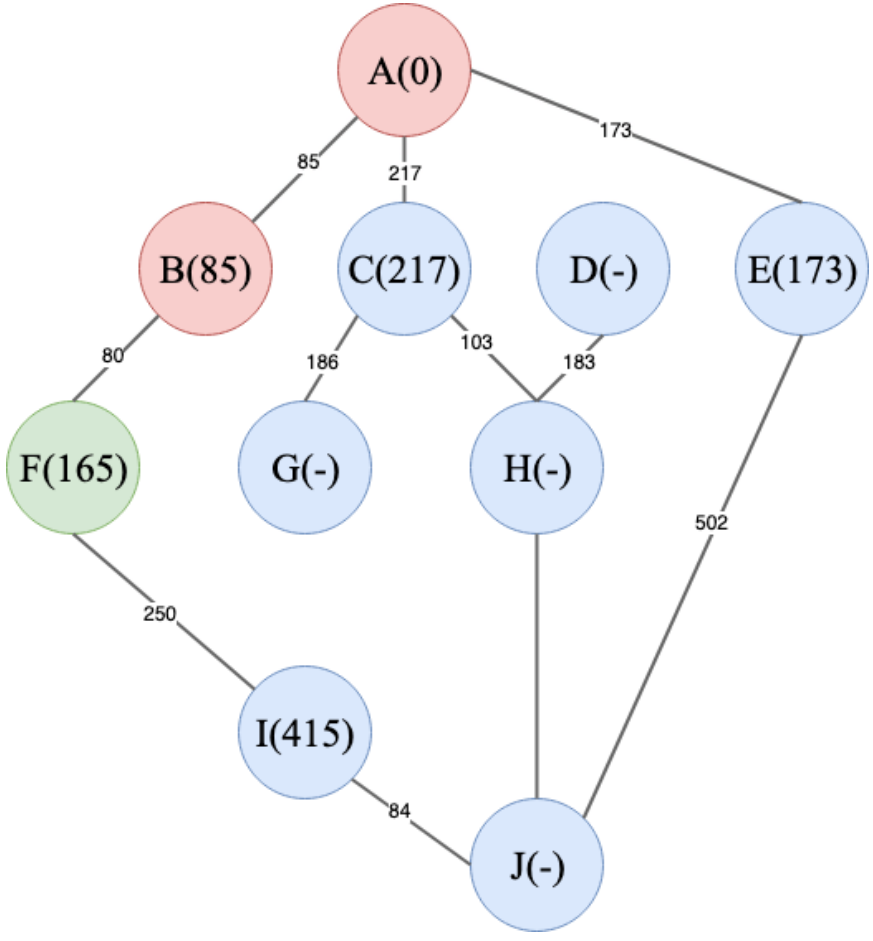
Prenons un exemple

Étape 2: on choisit la ville B. En effet, c'est la ville hors du sous-graphe qui est à la distance minimale (85). On met à jour le seul voisin (F). Sa distance devient 85+80 = 165.



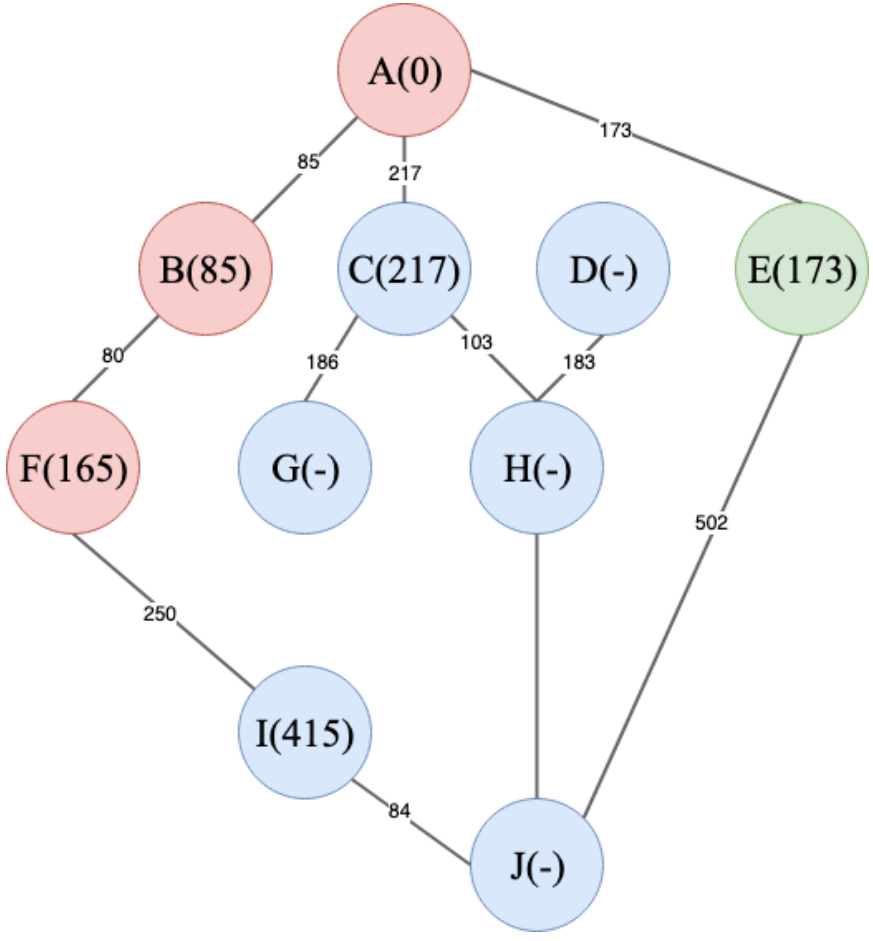
Prenons un exemple

Étape 3: on choisit F. On met à jour le voisin I (415).



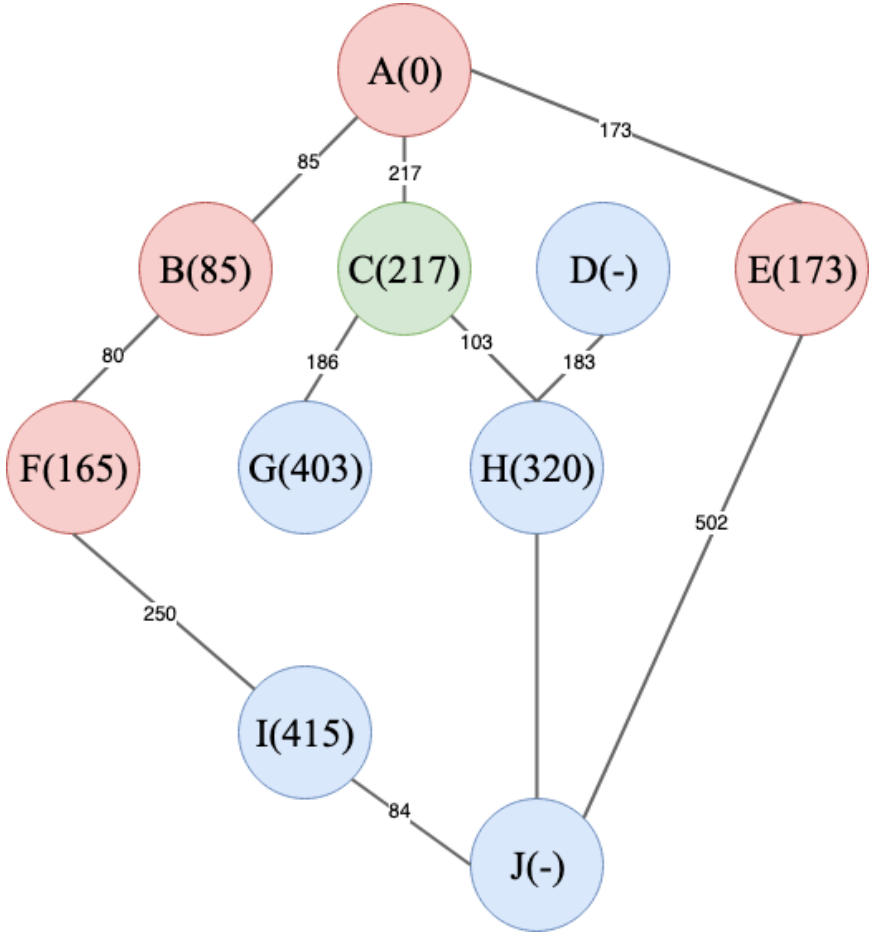
Prenons un exemple

Étape 4 : on choisit E. On met à jour le voisin J (675).



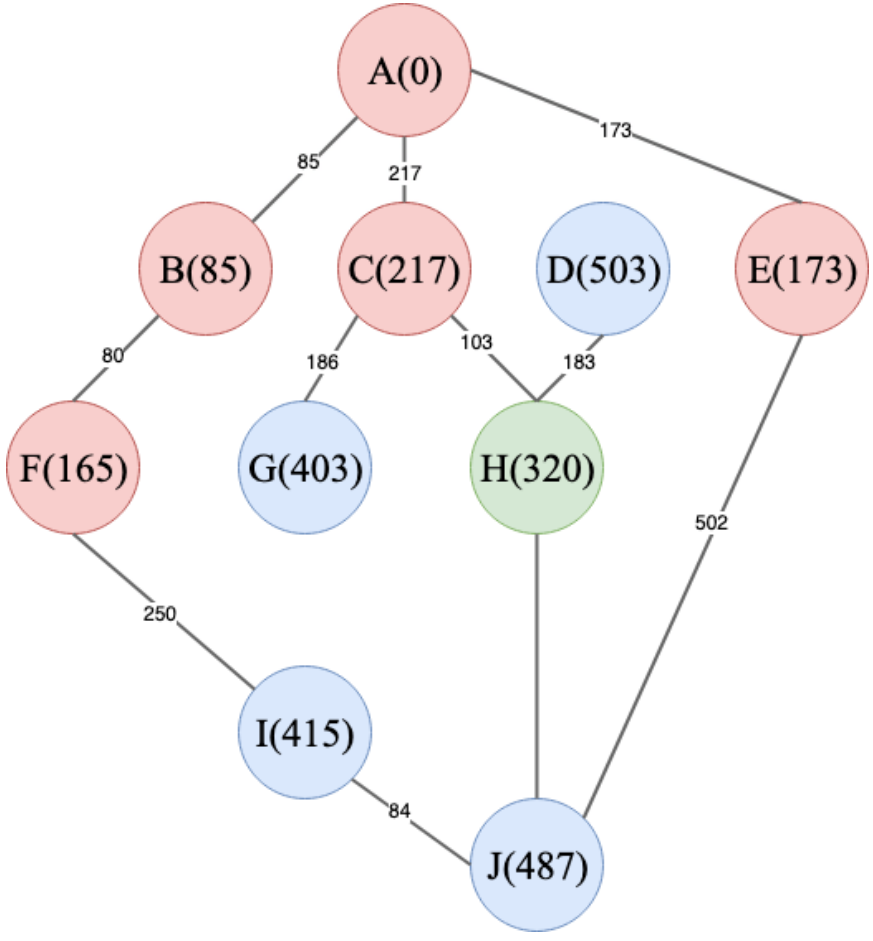
Prenons un exemple

Étape 5 : la distance la plus courte en dehors du sous-graphe est maintenant celle de la ville C. On choisit donc C. On met à jour la ville G (403) et la ville H (320).



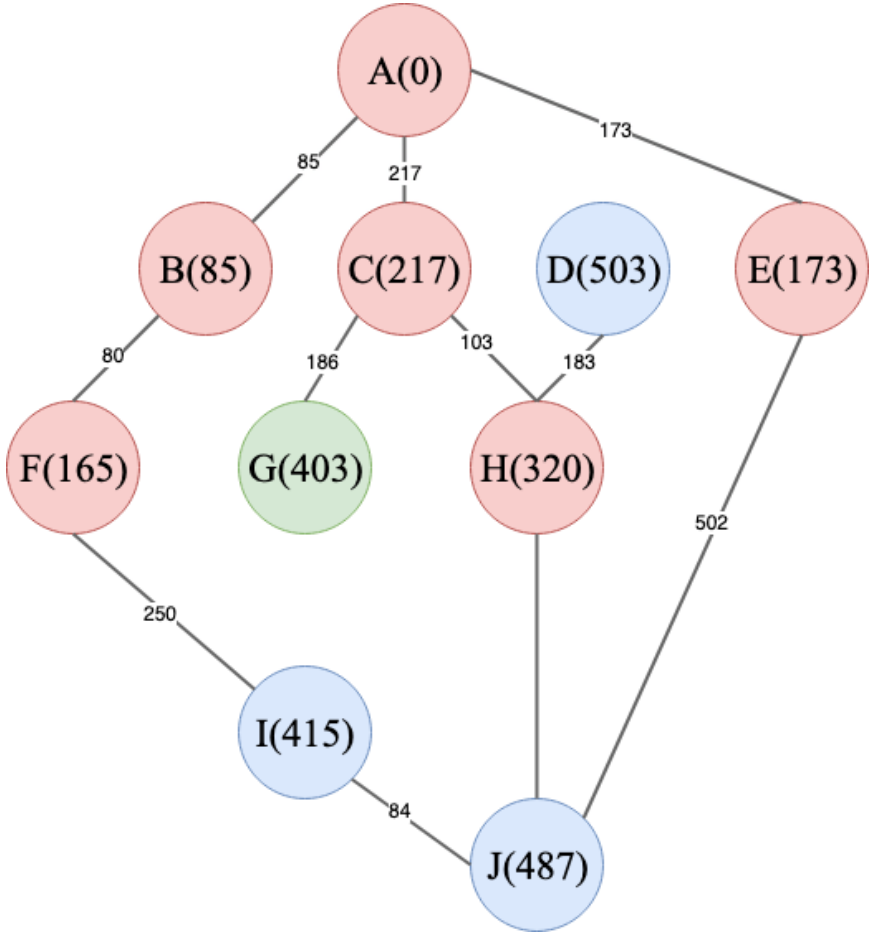
Prenons un exemple

Étape 6 : la distance la plus courte en dehors du sous-graphe est maintenant celle de la ville H (320). On choisit donc H. On met à jour la ville D (503) et la ville J (487< 675).



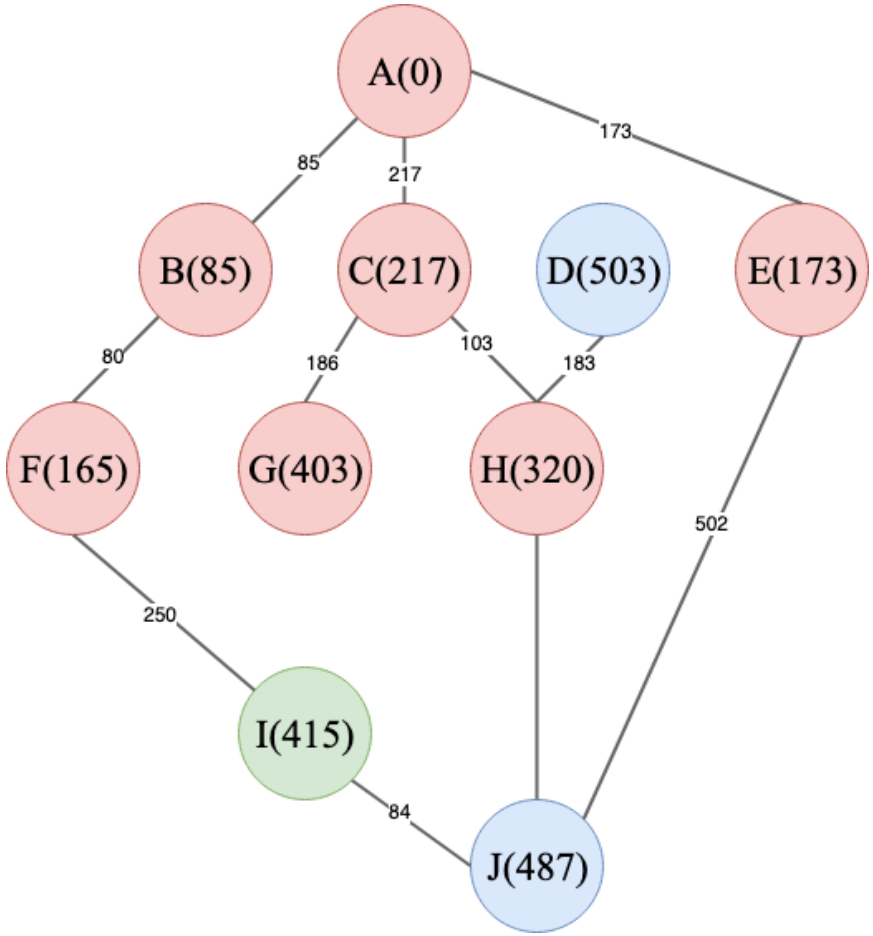
Prenons un exemple

Étape 7 : la distance la plus courte suivante est celle de la ville G. On choisit G. La mise à jour ne change aucune autre distance.



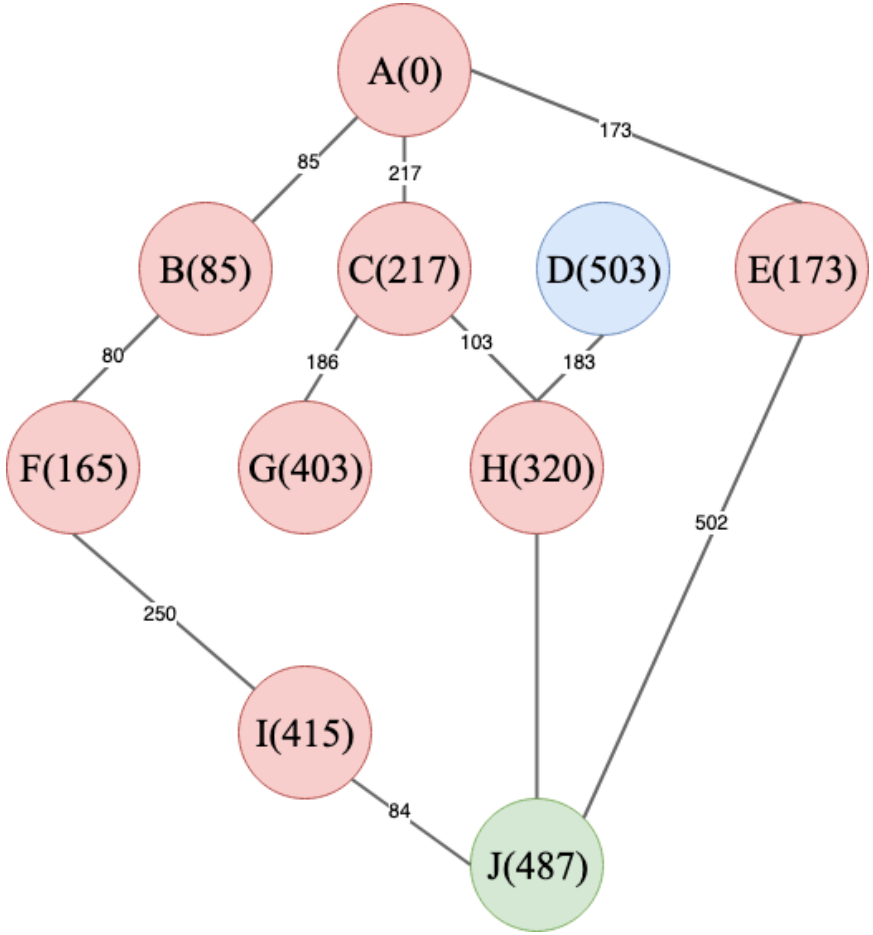
Prenons un exemple

Étape 8 : la distance la plus courte suivante est celle de la ville I. La distance de la ville voisine J n'est pas modifiée car la distance existante est inférieure à celle que l'on obtiendrait en passant par I ($415 + 84 > 487$).



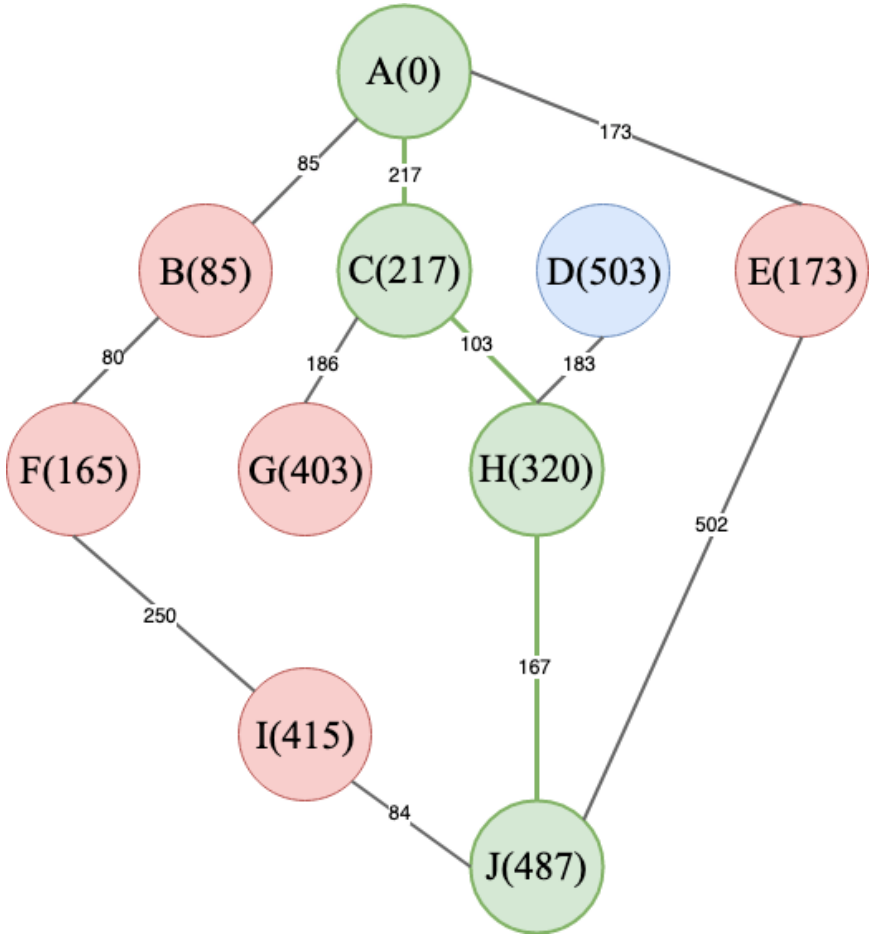
Prenons un exemple

Étape 9 : la ville dont la distance est la plus courte est J (487). On choisit J et on l'ajoute au sous-graphe. On s'arrête puisque la ville d'arrivée est maintenant dans le sous-graphe.



Prenons un exemple

En neuf étapes, on peut déterminer le chemin le plus court menant de A à J, il passe par C et H et mesure 487 km.



Coloration de graphes

Nombre chromatique

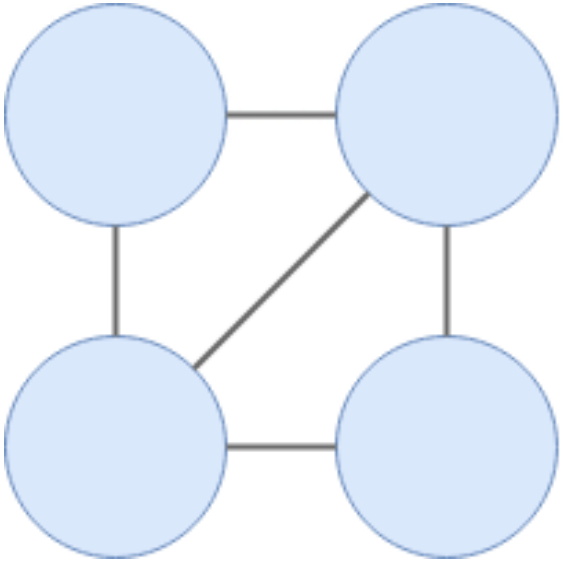
Définition

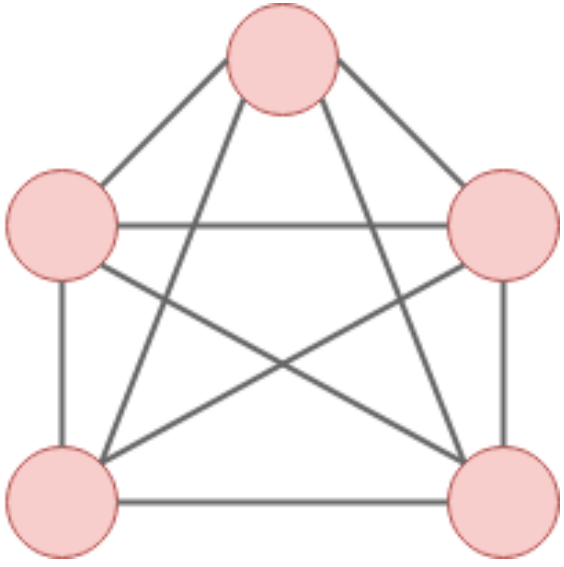
- Une **k-coloration** d'un graphe G est un étiquetage f : $V(G) \to \{1, 2, \ldots, k\}$;
- Les étiquettes sont des couleurs;
- Les sommets avec une couleur i sont une classe de couleurs;
- Une k-coloration f est **propre** si $f(x) \neq f(y)$ quand x et y sont adjacents;
- Le **nombre chromatique** $\chi(G)$ est le minimum k tel que G a une k-coloration propre;
- On note $\omega(G)$ le **nombre de clique** la taille de la plus grande clique d'un graphe;
- On note $\alpha(G)$ la taille du plus grand **ensemble indépendant**, c'est-à-dire un ensemble de sommets deux à deux non adjacent.

Exemple sur le nombre chromatique

Rappel

Une k-coloration f est **propre** si $f(x) \neq f(y)$ quand x et y sont adjacents;



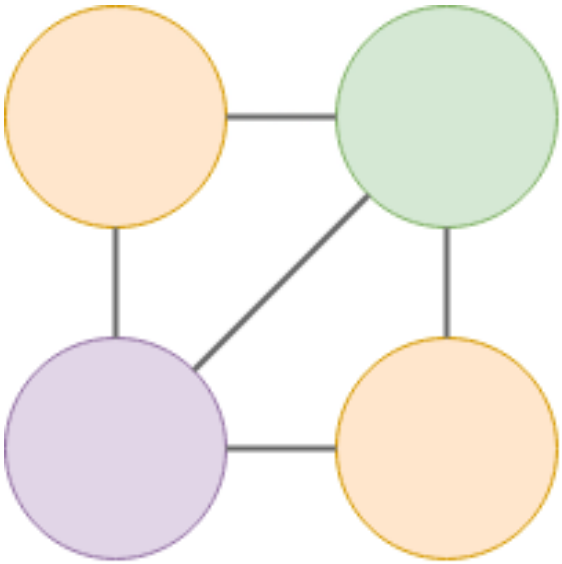


Quel est le nombre chromatique ?

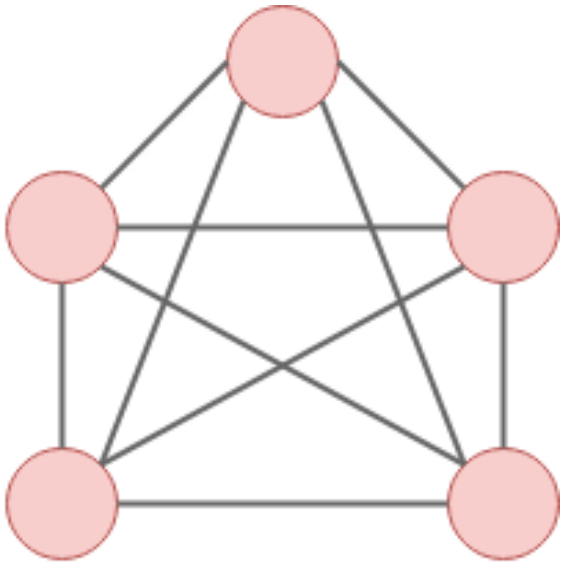
Exemple sur le nombre chromatique

Rappel

Une k-coloration f est **propre** si $f(x) \neq f(y)$ quand x et y sont adjacents;



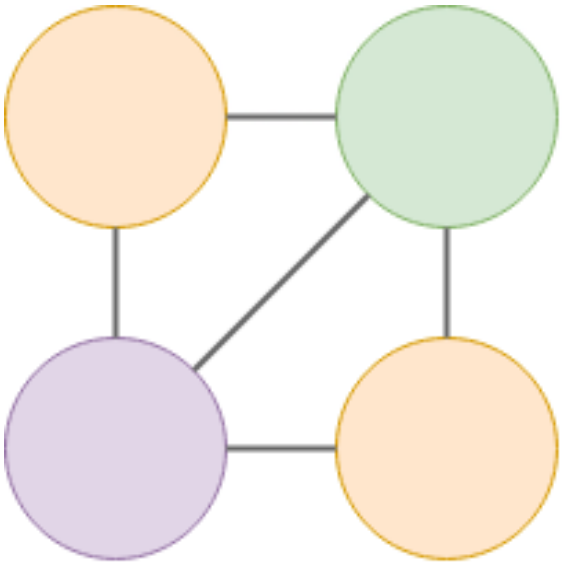
$\mathrm{k} = 3$



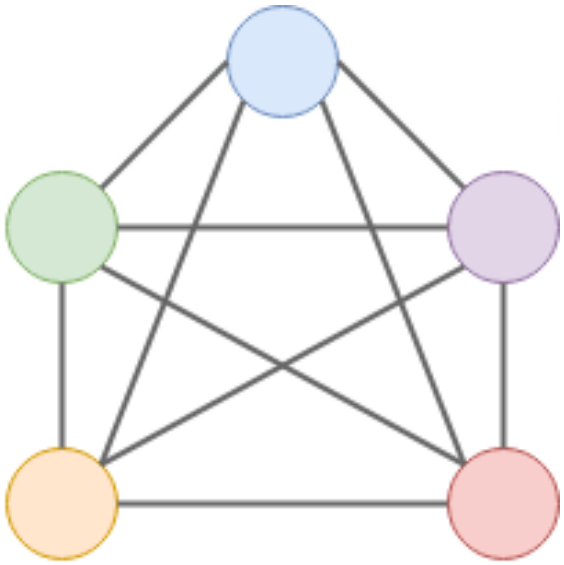
Exemple sur le nombre chromatique

Rappel

Une k-coloration f est **propre** si $f(x) \neq f(y)$ quand x et y sont adjacents;



k = 3

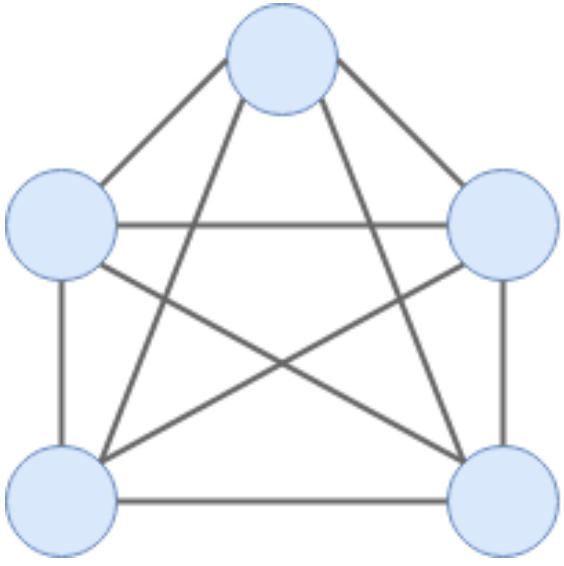


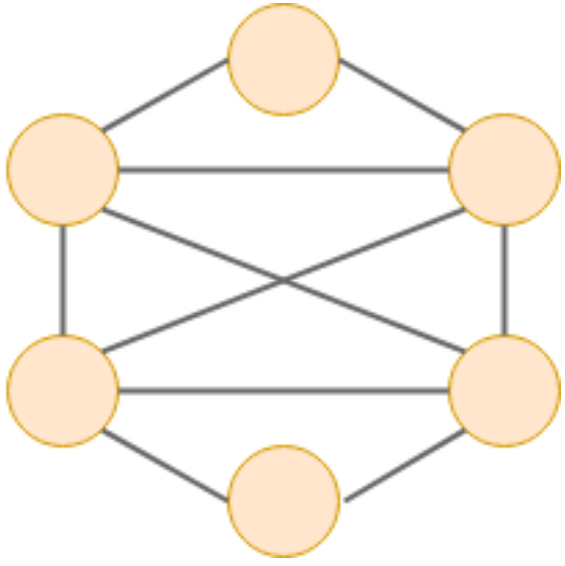
k = 5

Exemple sur le nombre de clique

Rappel

On note $\omega(G)$ le **nombre de clique** la taille de la plus grande clique d'un graphe

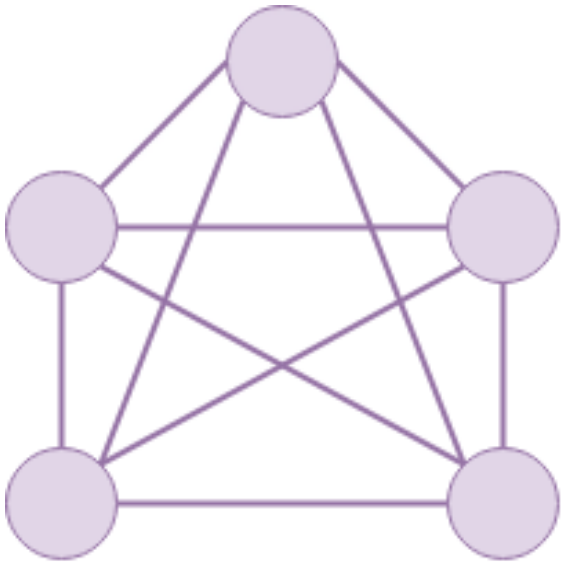




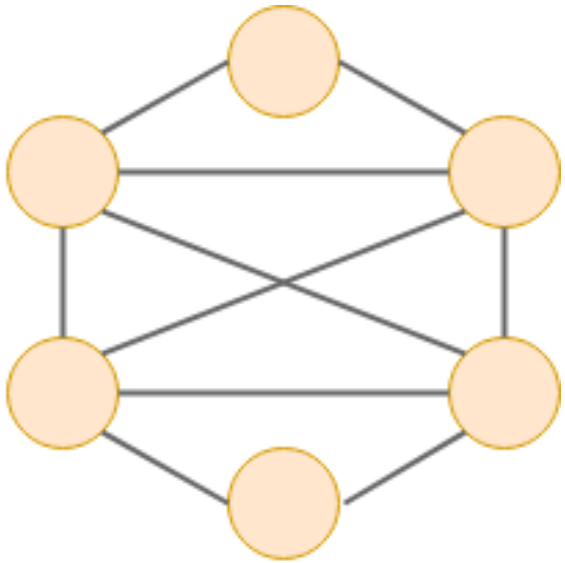
Exemple sur le nombre de clique

Rappel

On note $\omega(G)$ le **nombre de clique** la taille de la plus grande clique d'un graphe



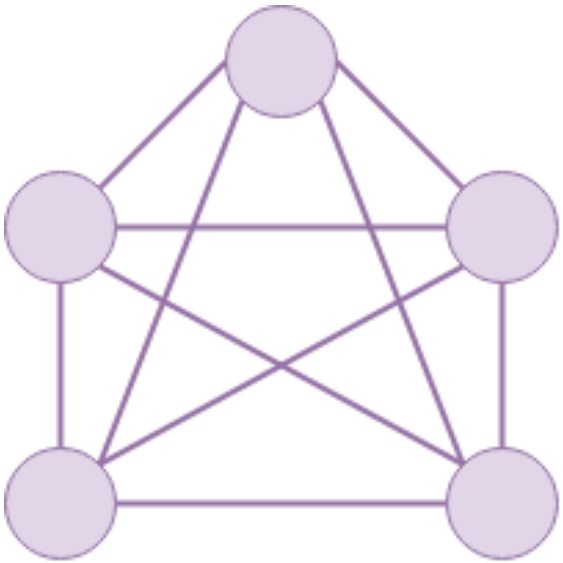
$w(G) = 5$



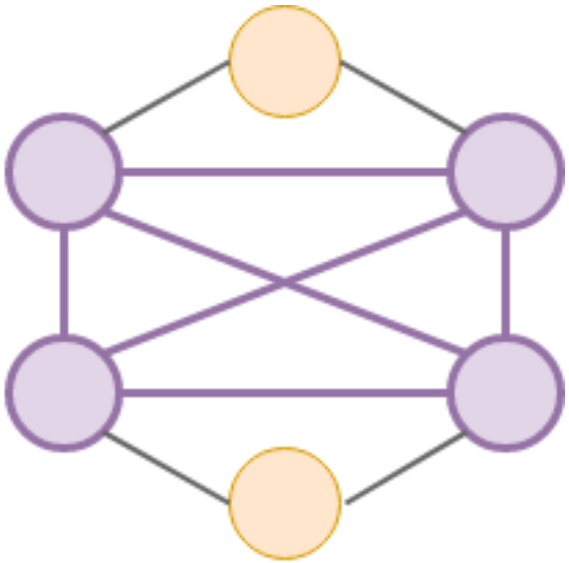
Exemple sur le nombre de clique

Rappel

On note $\omega(G)$ le **nombre de clique** la taille de la plus grande clique d'un graphe



w(G) = 5

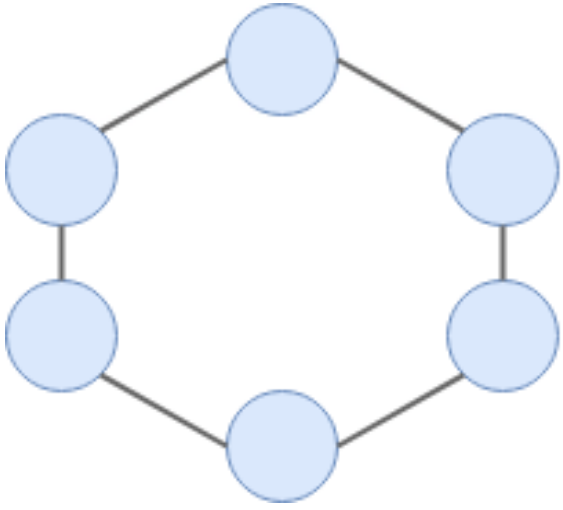


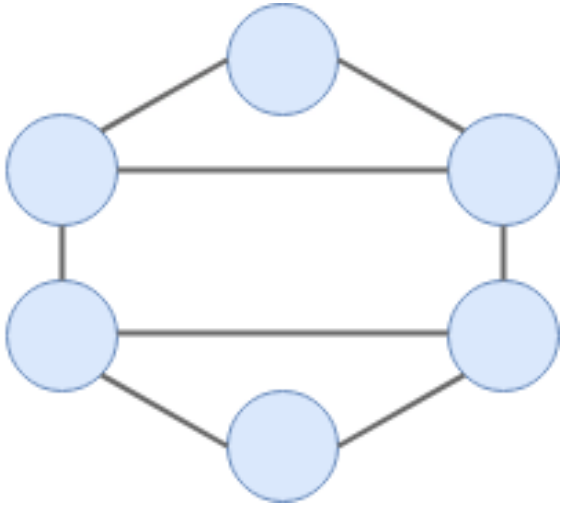
w(G) = 4

Exemple sur les ensembles indépendant

Rappel

On note $\alpha(G)$ la taille du plus grand **ensemble indépendant**, c'est-à-dire un ensemble de sommets deux à deux non adjacent.

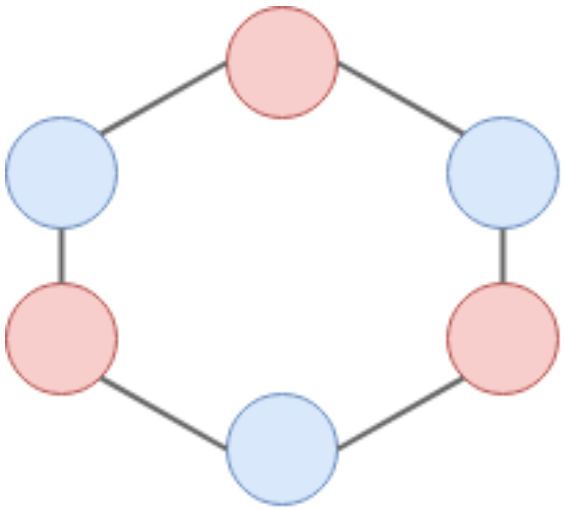




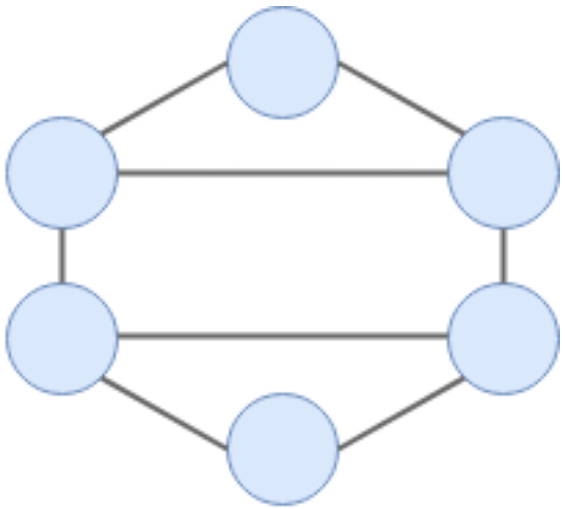
Exemple sur les ensembles indépendant

Rappel

On note $\alpha(G)$ la taille du plus grand **ensemble indépendant**, c'est-à-dire un ensemble de sommets deux à deux non adjacent.



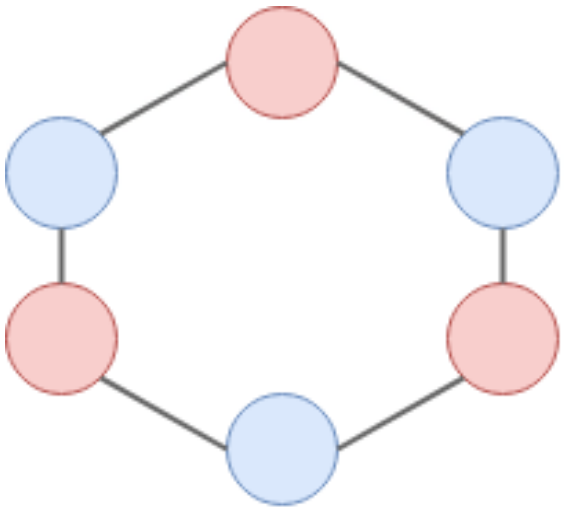
$a(G) = 3$



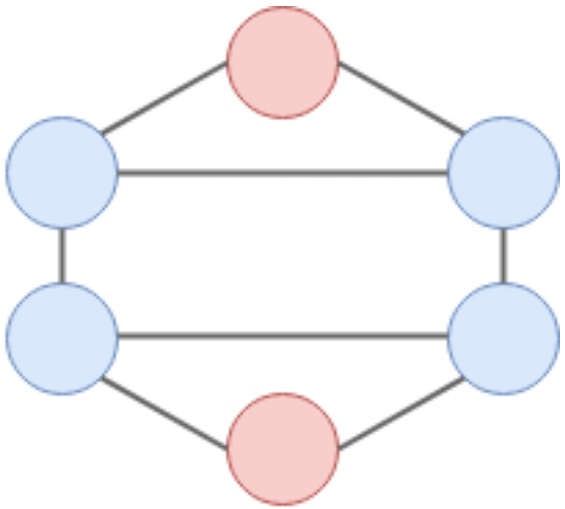
Exemple sur les ensembles indépendant

Rappel

On note $\alpha(G)$ la taille du plus grand **ensemble indépendant**, c'est-à-dire un ensemble de sommets deux à deux non adjacent.



a(G) = 3

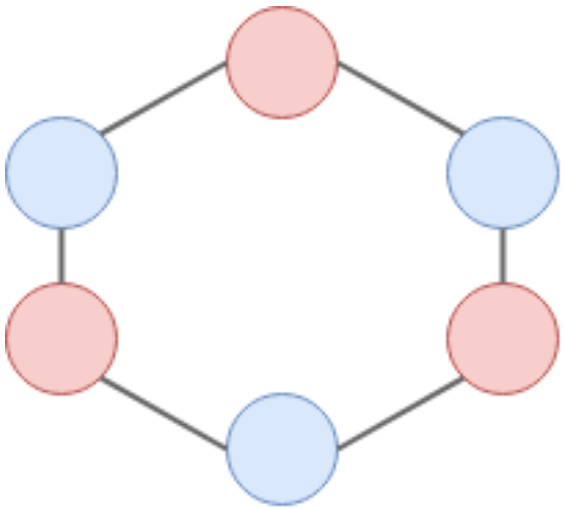


a(G) = 2

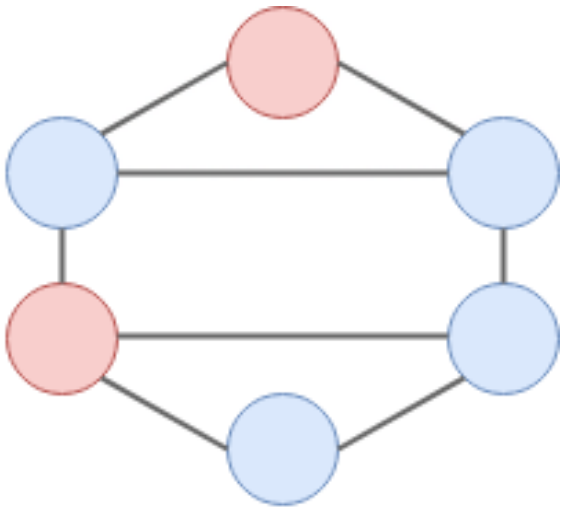
Exemple sur les ensembles indépendant

Rappel

On note $\alpha(G)$ la taille du plus grand **ensemble indépendant**, c'est-à-dire un ensemble de sommets deux à deux non adjacent.



a(G) = 3

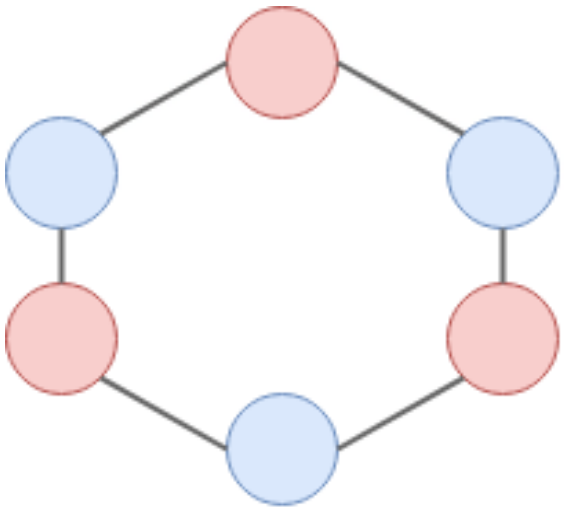


a(G) = 2

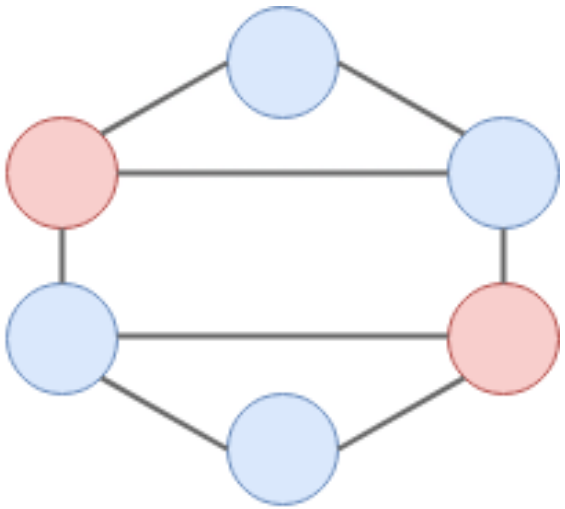
Exemple sur les ensembles indépendant

Rappel

On note $\alpha(G)$ la taille du plus grand **ensemble indépendant**, c'est-à-dire un ensemble de sommets deux à deux non adjacent.



a(G) = 3



a(G) = 2

Théorèmes sur la coloration

Théorème 1
Soit G un graphe à n sommets, on a la relation suivante:

$$\frac{n}{\alpha} \leq \chi$$

Proof.
Supposons que G est colorié avec χ couleurs. Comme chaque classe de couleurs est indépendant, la taille de chaque classe de couleur est au plus α. Soient les classes de couleur $V_1, V_2, \ldots, V_\alpha$. On obtient:

$$n = \sum_{i=1}^{\chi} |V_i| \leq \chi\alpha$$

□

Théorèmes sur la coloration

Théorème 2
Soit Δ le degré maximal dans un graphe, on a la relation suivante:

$$\chi \leq \Delta + 1$$

Proof.
Colorez le premier sommet avec la première couleur et faire pour les n-1 sommets restants:

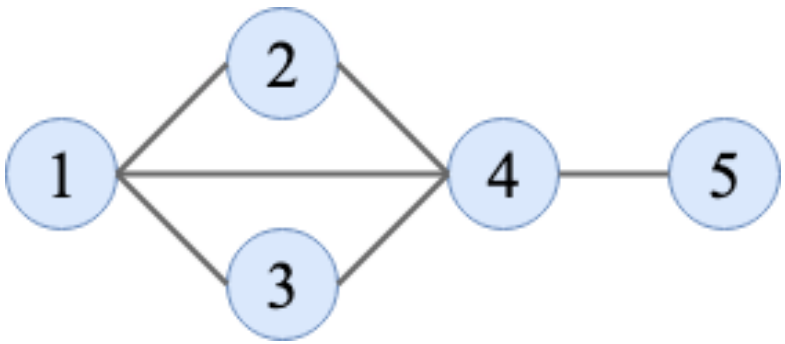
- On prend un sommet et on le colore avec la couleur la plus basse qui n'a pas été utilisée sur les sommets précédemment colorés qui lui sont adjacents;
- Si toutes les couleurs précédemment utilisées apparaissent sur les sommets adjacents à v, attribuez-lui une nouvelle couleur.

□

Théorèmes sur la coloration

Théorème 2
Soit Δ le degré maximal dans un graphe, on a la relation suivante:

$$\chi \leq \Delta + 1$$



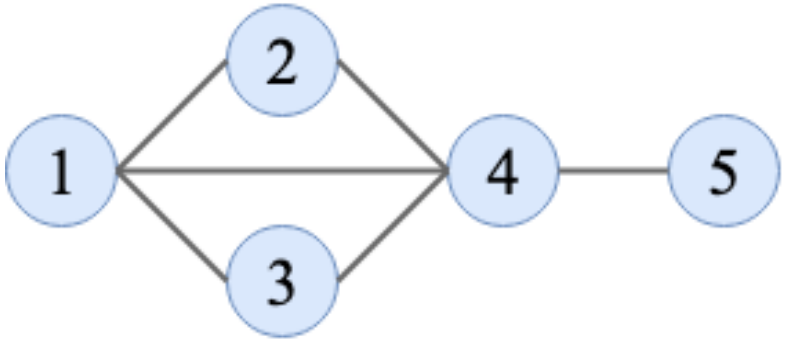
Quelle est la valeur de Δ ?

Théorèmes sur la coloration

Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

$$\chi \leq \Delta + 1$$

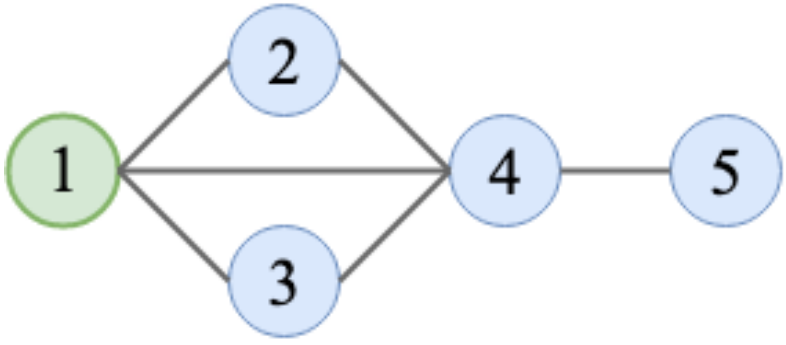


Théorèmes sur la coloration

Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

$$\chi \leq \Delta + 1$$

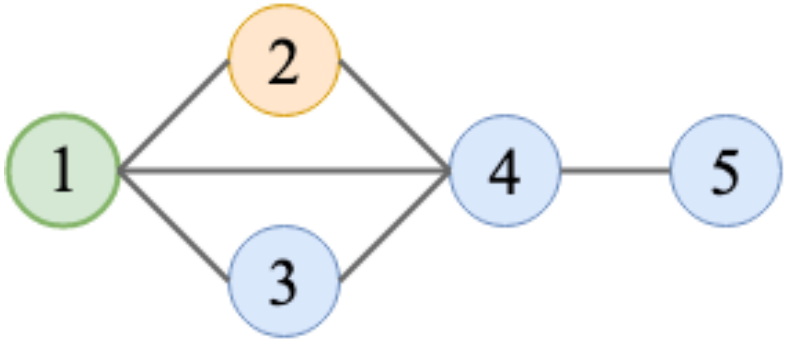


Théorèmes sur la coloration

Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

$$\chi \leq \Delta + 1$$

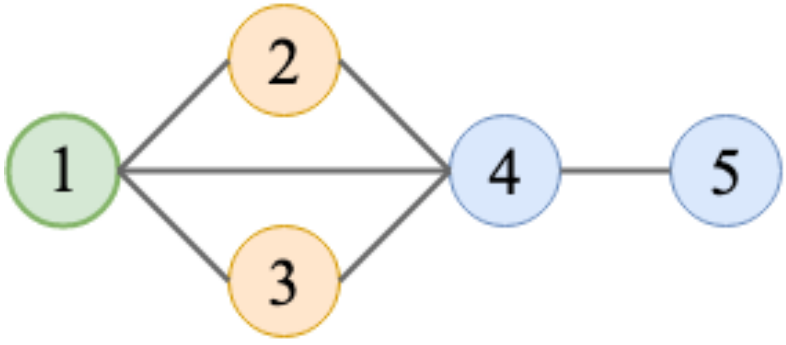


Théorèmes sur la coloration

Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

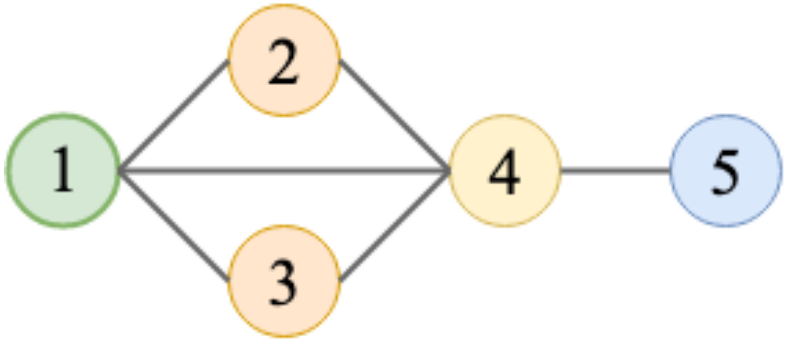
$$\chi \leq \Delta + 1$$



Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

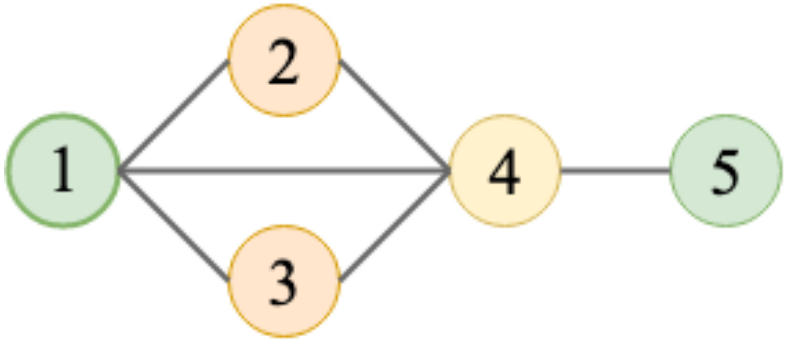
$$\chi \leq \Delta + 1$$



Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

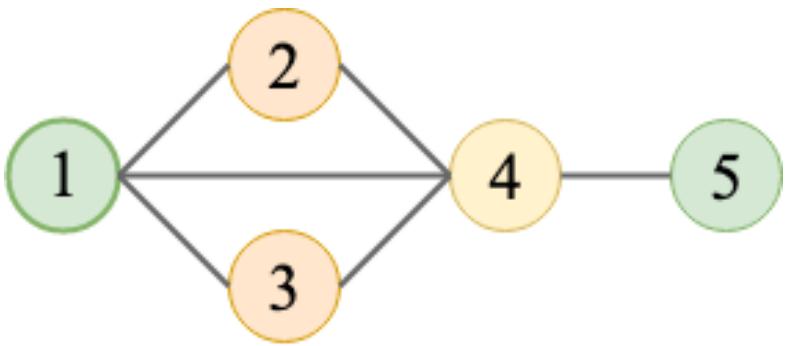
$$\chi \leq \Delta + 1$$



Théorème 2

Soit Δ le degré maximal dans un graphe, on a la relation suivante:

$$\chi \leq \Delta + 1$$



$k = 3$

Représentation des graphes

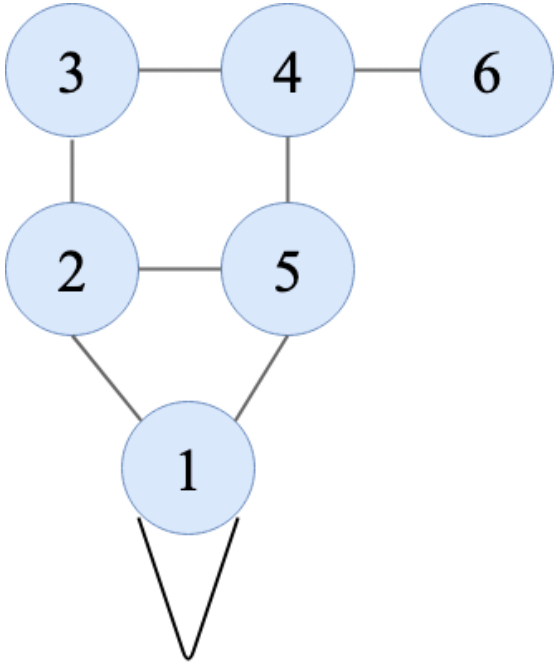
Matrice d'adjacence

Définition

Supposons que $G = (V, E)$ est un graphe simple, où $|V| = n$. Supposons aussi que les sommets de G sont numérotés arbitrairement $v_1, \ldots, v_n$. La matrice d'adjacence A de G se rapportant à cet ensemble de sommets est la matrice n $\times$ n *booléenne* A avec:

$$a_{ij} = \begin{cases} 1 & \text{si}(v_i, v_j) \in E \\ 0 & \text{sinon.} \end{cases}$$

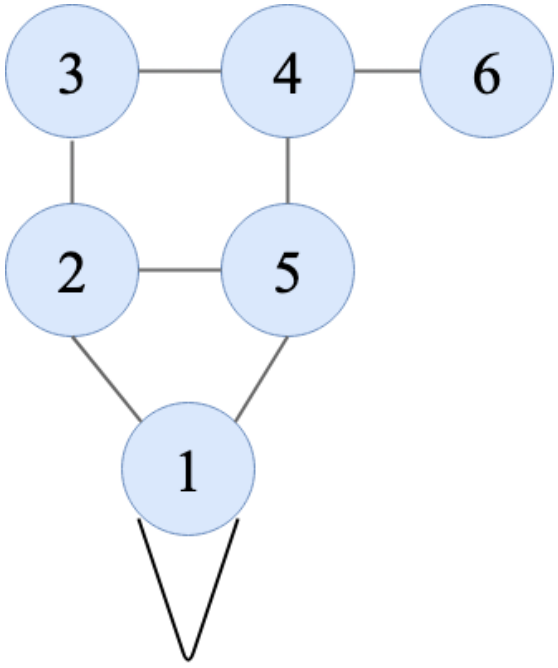
Exemple de matrice d'adjacence



$$\begin{pmatrix} & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ v_1 & 1 & 1 & 0 & 0 & 1 & 0 \\ v_2 & 1 & 0 & 1 & 0 & 1 & 0 \\ v_3 & 0 & 1 & 0 & 1 & 0 & 0 \\ v_4 & 0 & 0 & 1 & 0 & 1 & 1 \\ v_5 & 1 & 1 & 0 & 1 & 0 & 0 \\ v_6 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Liste d'arêtes

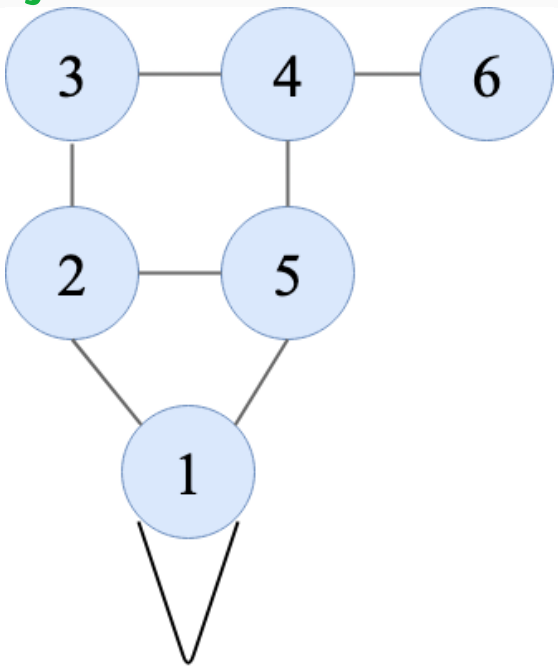
Liste d'arêtes



1 1
1 2
1 5
2 3
2 5
3 4
4 5
4 6

Liste d'adjacence

Liste d'adjacence



1 1 2 5

2 3 5

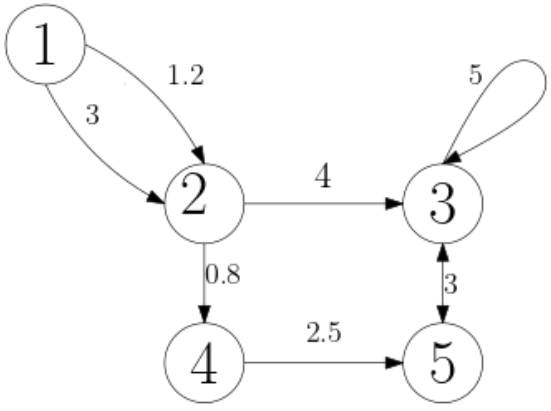
3 2 4

4 3 5

5 1 2 4

6 4

Liste d'arêtes pour les graphes pondérés



1 2 1.2

1 2 3

2 3 4

2 4 0.8

3 3 5

3 5 3

4 5 2.5

5 3 3