

ALGORITHMES ET STRUCTURES DE DONNÉES

IFT-2008/GLO-2100

Chapitre 9 : Arbres rouge et noir

Thierry Eude, Kim Rioux-Paradis

Département d'informatique et de génie logiciel



Plan

- Arbre rouge et noir
 - Propriétés
 - Exemples
 - Insertions
 - Suppressions

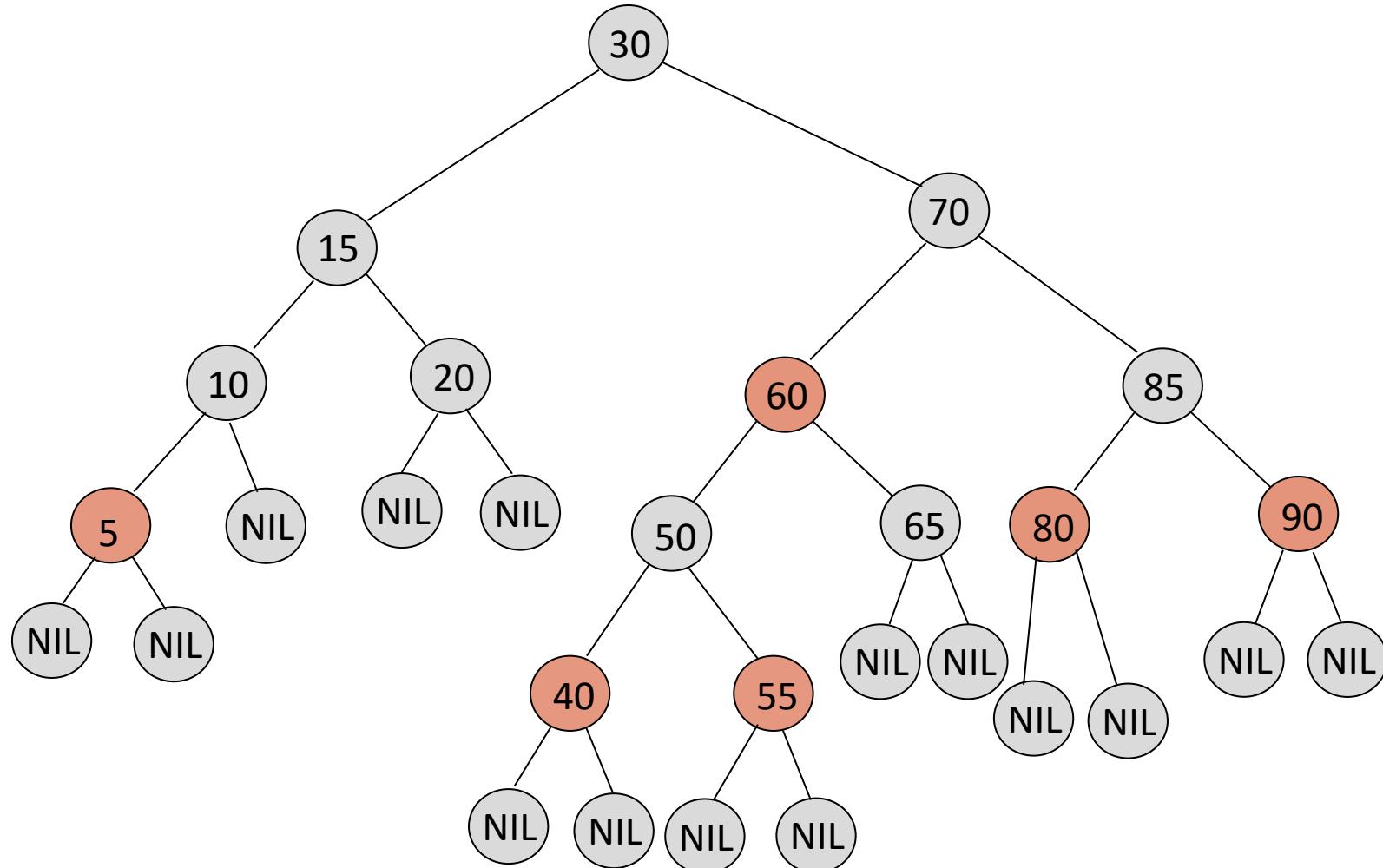
Propriétés d'un arbre rouge et noir

- Arbre binaire ordonné.
- Chaque noeud est rouge ou noir.
- La racine est noire.
- Les feuilles (pointeurs NULL) sont noires.
- Les enfants d'un noeud rouge sont noirs.
- À partir de n'importe quel noeud, tous les chemins de la racine jusqu'à un pointeur NULL doivent avoir le même nombre de noeuds noirs

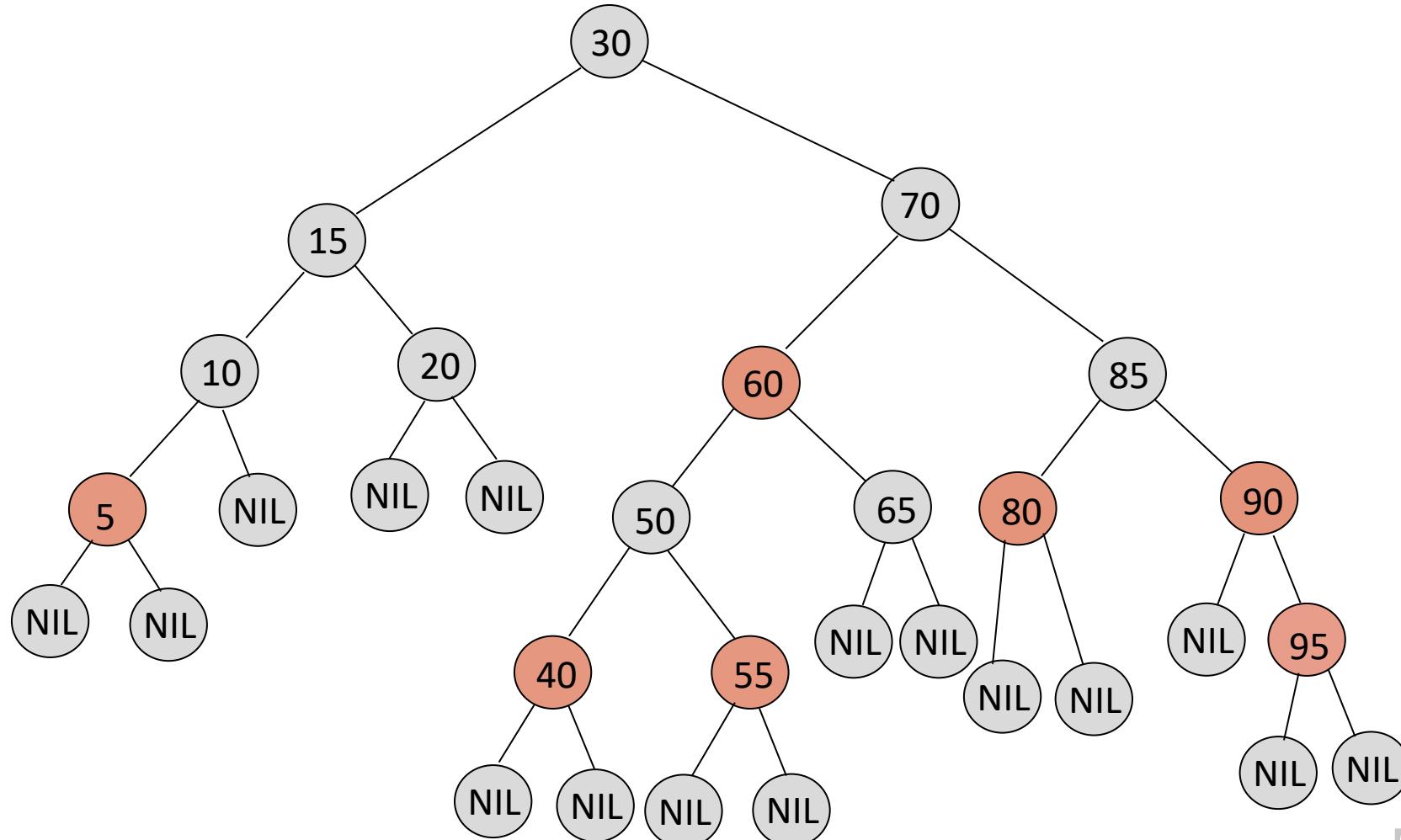
Propriétés d'un arbre rouge et noir

- Remarque :
 - Comme la racine est noire et il ne peut y avoir plus de deux noeuds rouges consécutifs, la longueur de tout chemin de la racine à une feuille ne peut être supérieure à 2 fois le nombre de noeuds noirs dans ce chemin.
 - La hauteur d'un arbre rouge et noir est maximum $2\log_2(n + 1)$.

Exemples

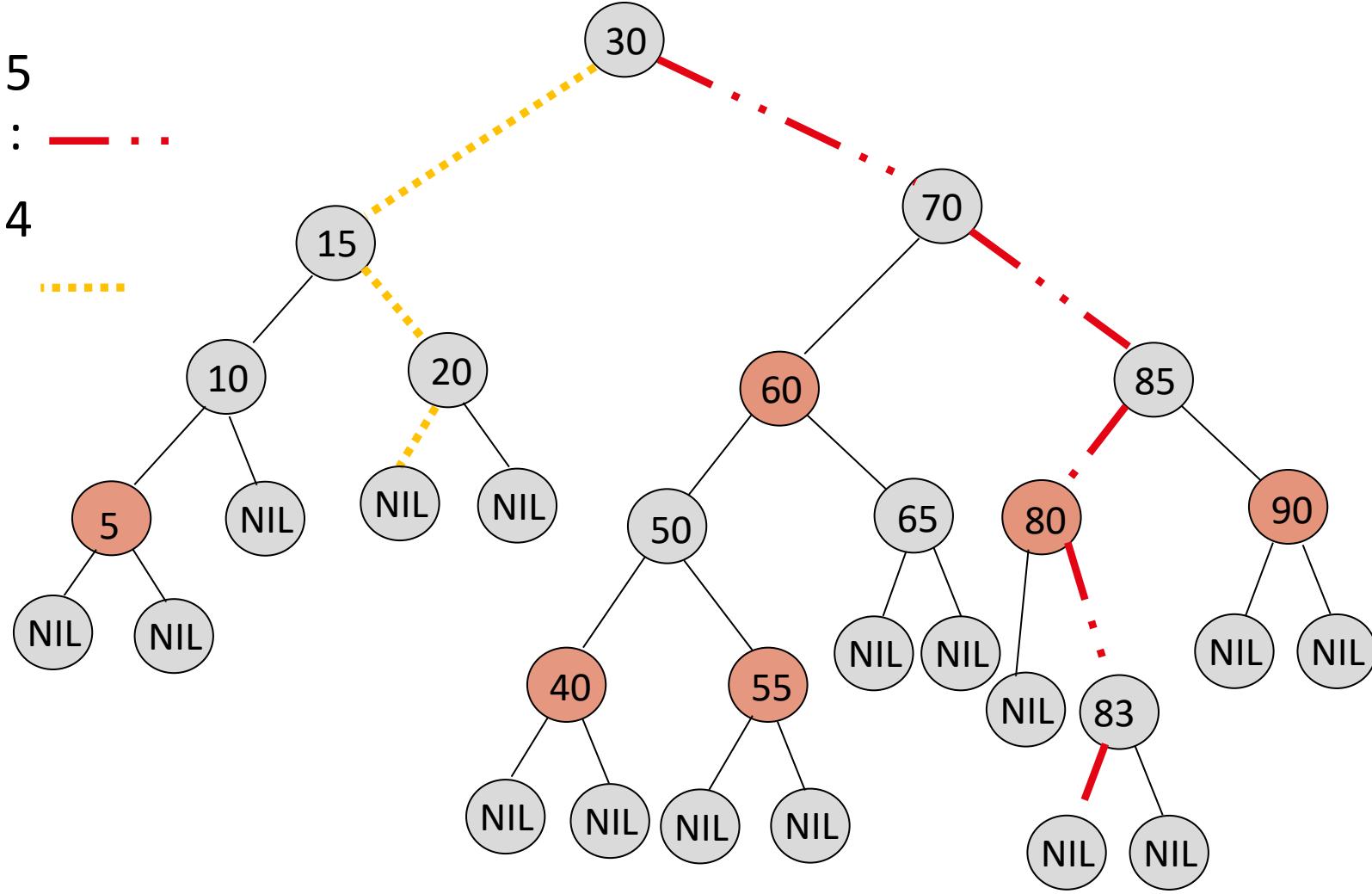


Exemples – Erreur deux noeuds rouges



Exemples – Erreur nombre de noeuds noirs sur un chemin différent

- Chemin avec 5 noeuds noirs : — · ·
- Chemin avec 4 noeuds noir : · · · ·



Insertion

- Nous verrons 3 cas possible lors de l'insertion et les manipulations à faire.

Insertion

- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - nœud → parent ← *∅;
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;

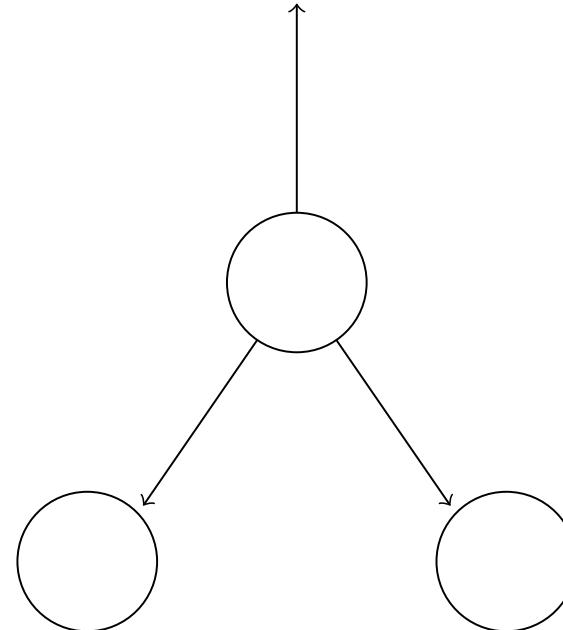
...

Insertion

- InsertionRougeNoir(int valeur)

- NœudPtr nœud = new Nœud;
- nœud → donnée ← valeur;
- nœud → parent ← *∅;
- nœud → gauche ← *F;
- nœud → droit ← *F;
- nœud → couleur ← Rouge;

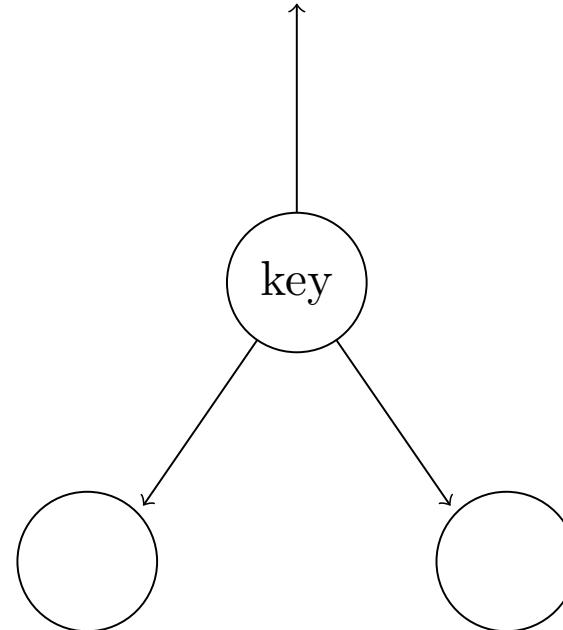
...



Insertion

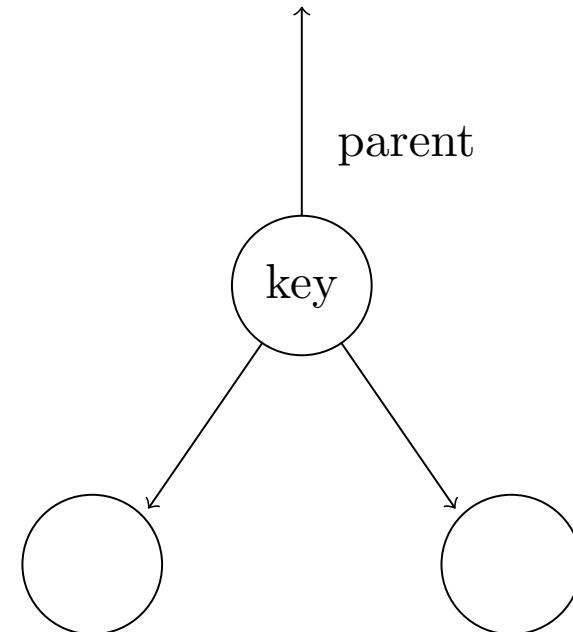
- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - nœud → parent ← *Ø;
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;

...



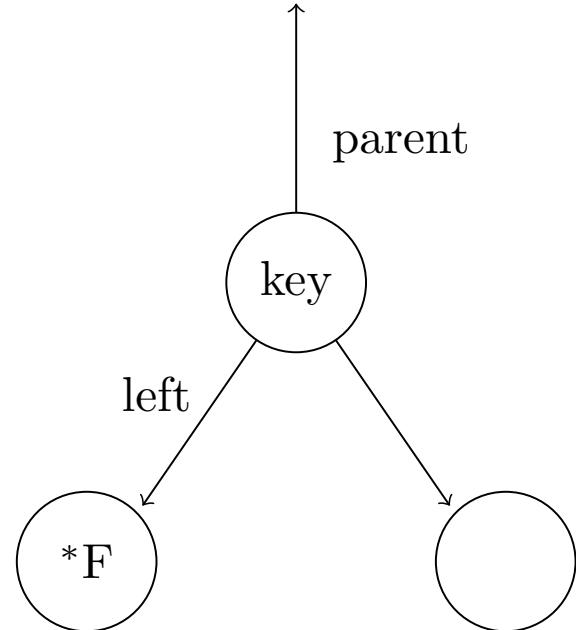
Insertion

- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - **nœud → parent ← *Ø;**
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;
- ...



Insertion

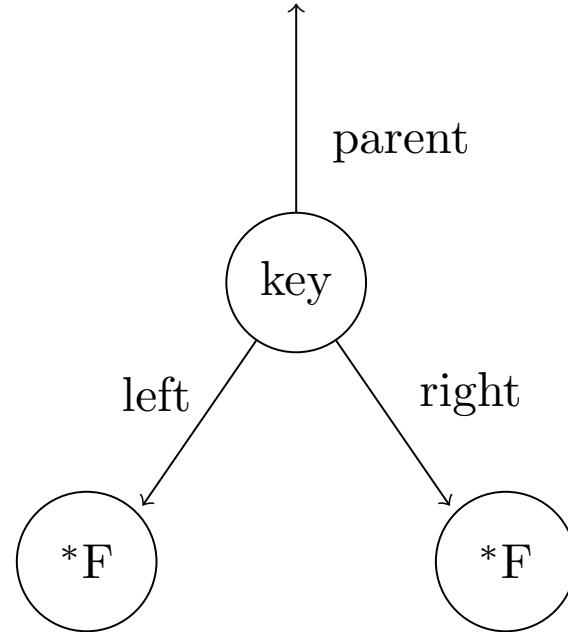
- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - nœud → parent ← *Ø;
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;
- ...



Insertion

- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - nœud → parent ← *Ø;
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;

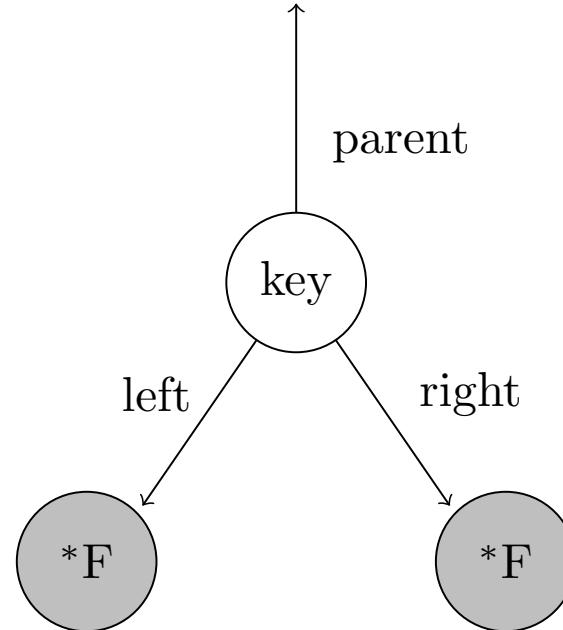
...



Insertion

- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - nœud → parent ← *Ø;
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;

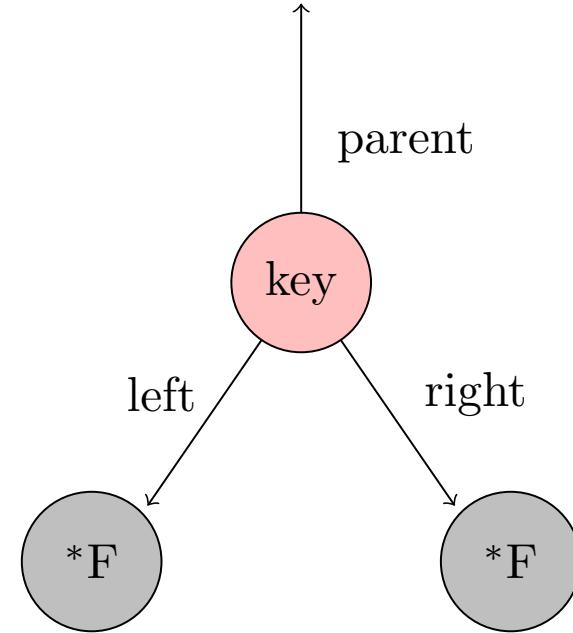
...



Insertion

- InsertionRougeNoir(int valeur)
 - NœudPtr nœud = new Nœud;
 - nœud → donnée ← valeur;
 - nœud → parent ← *Ø;
 - nœud → gauche ← *F;
 - nœud → droit ← *F;
 - nœud → couleur ← Rouge;

...



Insertion

- InsertionRougeNoir(int valeur)

...

- NœudPtr y \leftarrow * Ø;
- NœudPtr x \leftarrow * racine;
- Tant que (x != *F)
 - ✓ y \leftarrow x;
 - ✓ Si (nœud \rightarrow donnée < x \rightarrow donnée)
 - x \leftarrow x \rightarrow gauche;
- ✓ Sinon
 - x \leftarrow x \rightarrow droit;

...

Insertion

- InsertionRougeNoir(int valeur)

...

```
➤ NœudPtr y ← * Ø;  
➤ NœudPtr x ← * racine;
```

➤ Tant que (x != *F)

✓ y ← x;

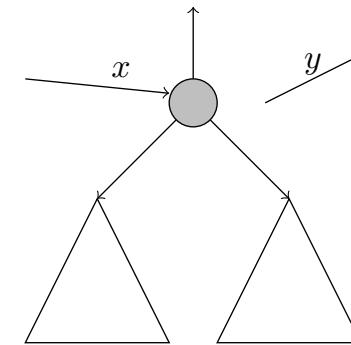
✓ Si (nœud → donnée < x → donnée)

• x ← x → gauche;

✓ Sinon

• x ← x → droit;

...



Insertion

- InsertionRougeNoir(int valeur)

...

- NœudPtr y $\leftarrow * \emptyset;$
- NœudPtr x $\leftarrow * \text{racine};$

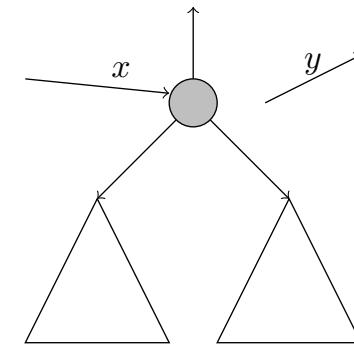
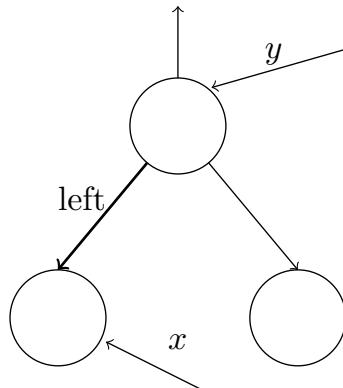
- Tant que ($x \neq *F$)

- $y \leftarrow x;$

- Si (nœud \rightarrow donnée $< x \rightarrow$ donnée)
 - $x \leftarrow x \rightarrow \text{gauche};$

- Sinon

- $x \leftarrow x \rightarrow \text{droit};$



Insertion

- InsertionRougeNoir(int valeur)

...

- NœudPtr y $\leftarrow * \emptyset;$
- NœudPtr x $\leftarrow * \text{racine};$

- Tant que ($x \neq *F$)

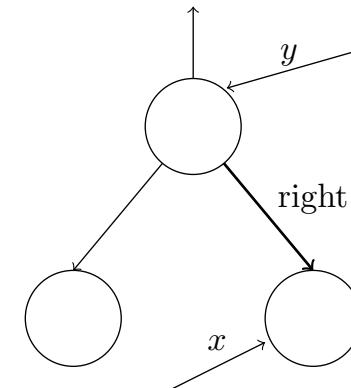
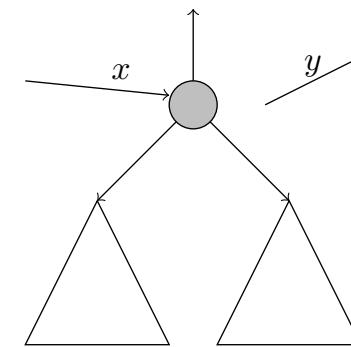
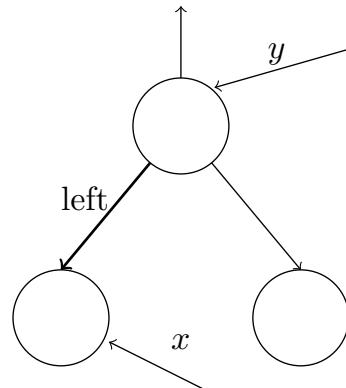
- $y \leftarrow x;$

- Si (nœud \rightarrow donnée $< x \rightarrow$ donnée)
 - $x \leftarrow x \rightarrow \text{gauche};$

- ✓ Sinon

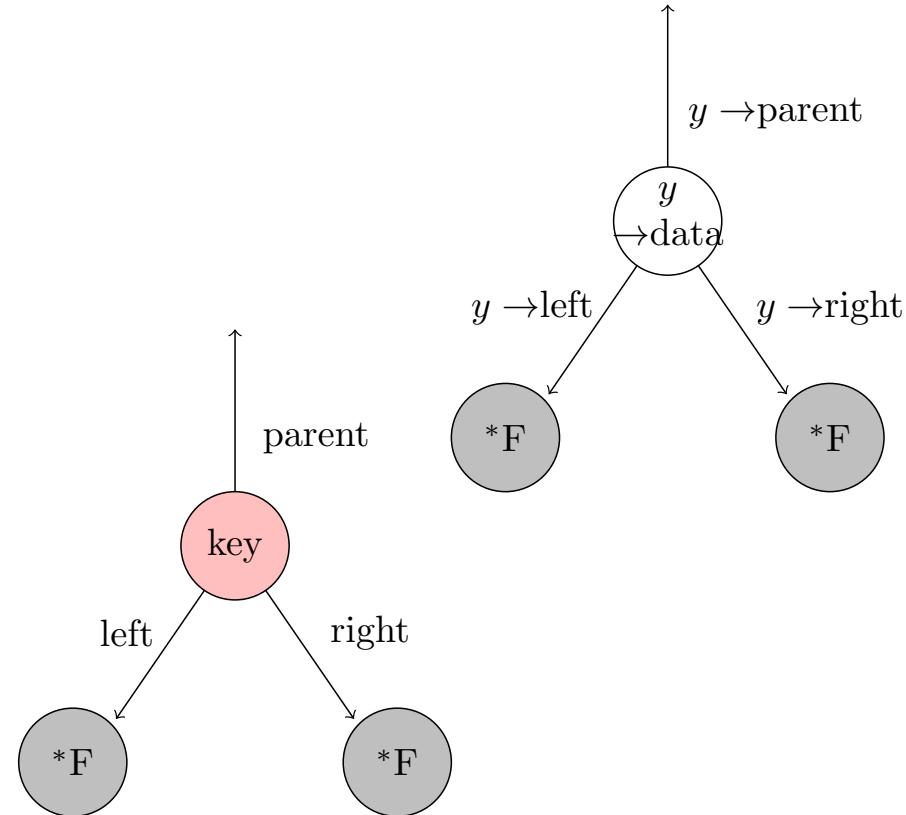
- $x \leftarrow x \rightarrow \text{droit};$

...



Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - $y \rightarrow$ gauche ← nœud;
 - ✓ Sinon
 - $y \rightarrow$ droit ← nœud;
 - Si ($nœud \rightarrow parent == * \emptyset$)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si($nœud \rightarrow parent \rightarrow parent == * \emptyset$)
 - ✓ retourne ;
 - insertionColoriage(nœud)

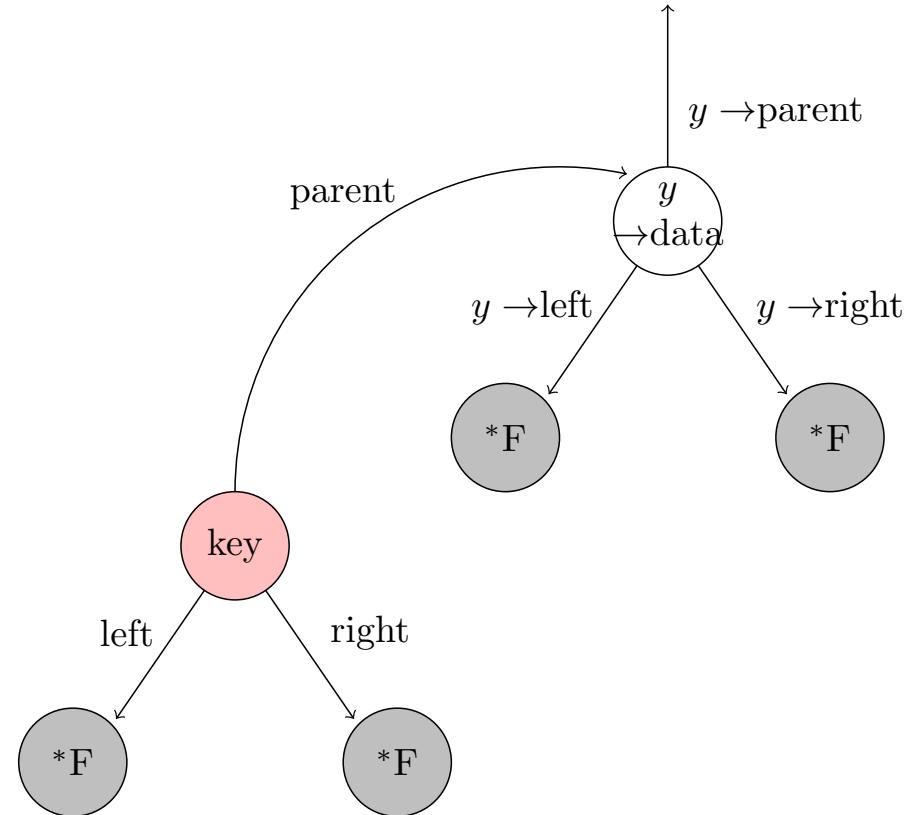


Insertion

- InsertionRougeNoir(int valeur)

...

- nœud → parent ← y;
- Si($y = * \emptyset$)
 - ✓ racine ← nœud;
- Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - $y \rightarrow$ gauche ← nœud;
 - ✓ Sinon
 - $y \rightarrow$ droit ← nœud;
- Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
- Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;
- insertionColoriage(nœud)

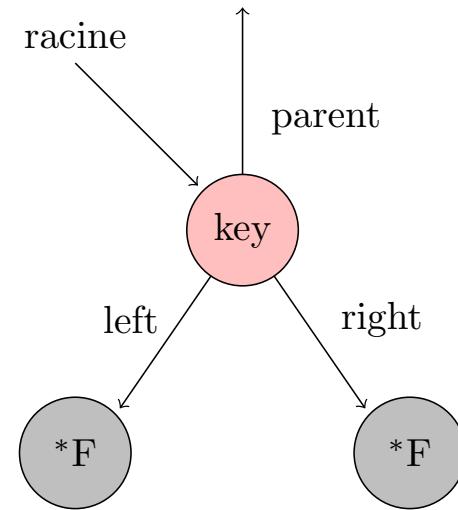


Insertion

- InsertionRougeNoir(int valeur)

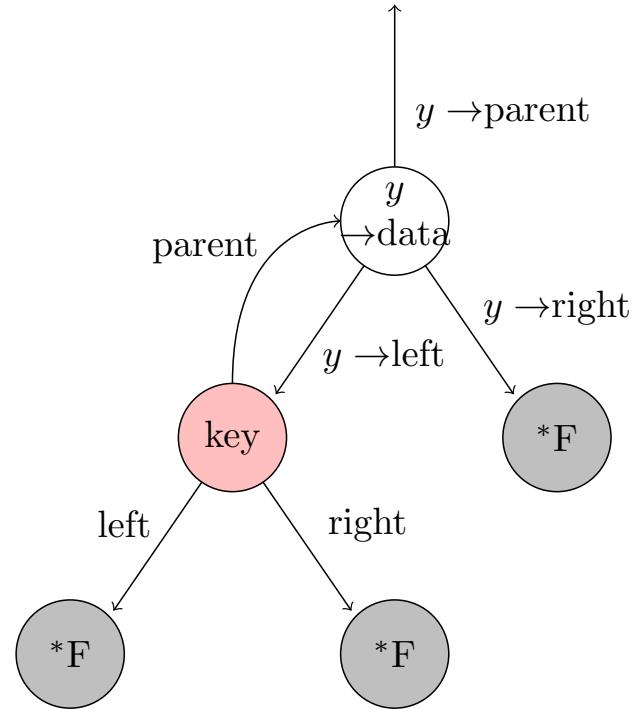
...

- nœud → parent ← v;
- Si($y = * \emptyset$)
 - ✓ racine ← nœud;
- Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - $y \rightarrow$ gauche ← nœud;
 - ✓ Sinon
 - $y \rightarrow$ droit ← nœud;
- Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
- Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;
- insertionColoriage(nœud)



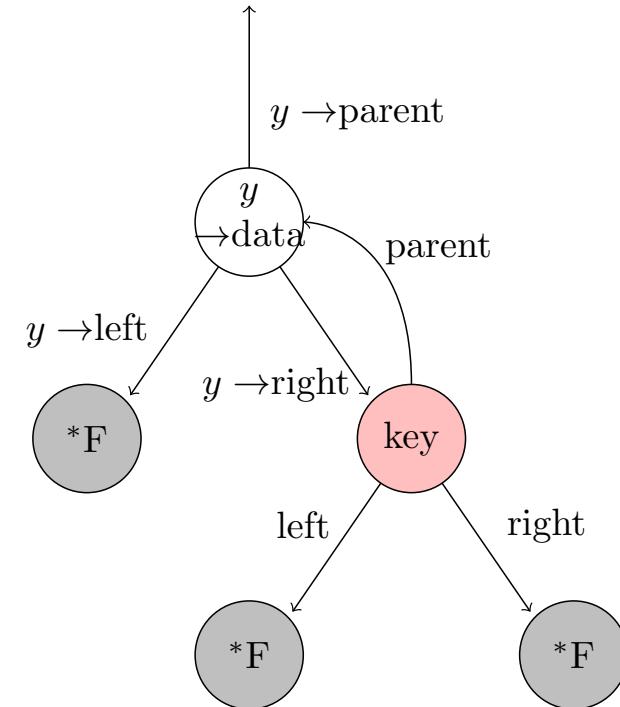
Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - $y \rightarrow \text{gauche} \leftarrow \text{nœud}$;
 - ✓ Sinon
 - $y \rightarrow \text{droit} \leftarrow \text{nœud}$;
 - Si ($\text{nœud} \rightarrow \text{parent} == * \emptyset$)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si($\text{nœud} \rightarrow \text{parent} \rightarrow \text{parent} == * \emptyset$)
 - ✓ retourne ;
 - insertionColoriage(nœud)



Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;
 - Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;
 - insertionColoriage(nœud)



Insertion

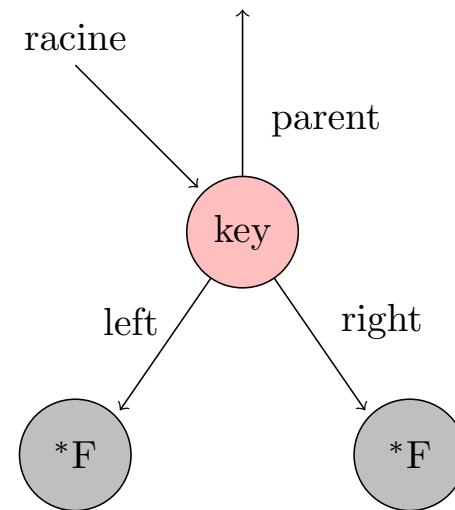
- InsertionRougeNoir(int valeur)

...

- nœud → parent ← y;
- Si($y = * \emptyset$)
 - ✓ racine ← nœud;
- Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;

- Si ($nœud \rightarrow parent == * \emptyset$)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si($nœud \rightarrow parent \rightarrow parent == * \emptyset$)
 - ✓ retourne ;

- insertionColoriage(nœud)



Insertion

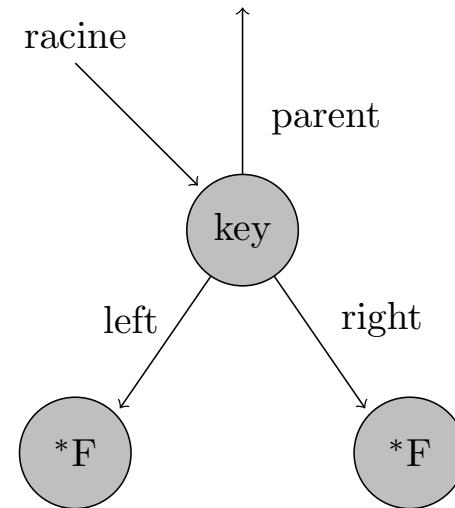
- InsertionRougeNoir(int valeur)

...

- nœud → parent ← y;
- Si($y = * \emptyset$)
 - ✓ racine ← nœud;
- Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;

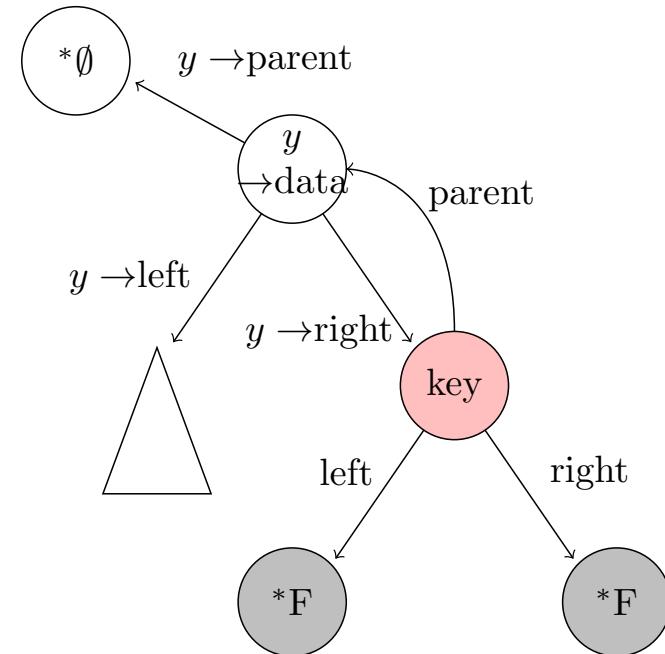
- Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;

- insertionColoriage(nœud)



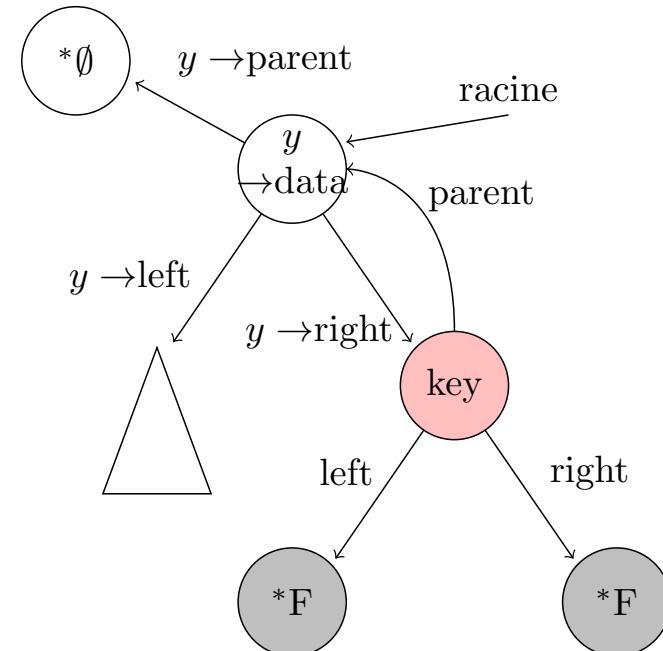
Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;
 - Si ($nœud \rightarrow parent == * \emptyset$)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si($nœud \rightarrow parent \rightarrow parent == * \emptyset$)
 - ✓ retourne ;
 - insertionColoriage(nœud)



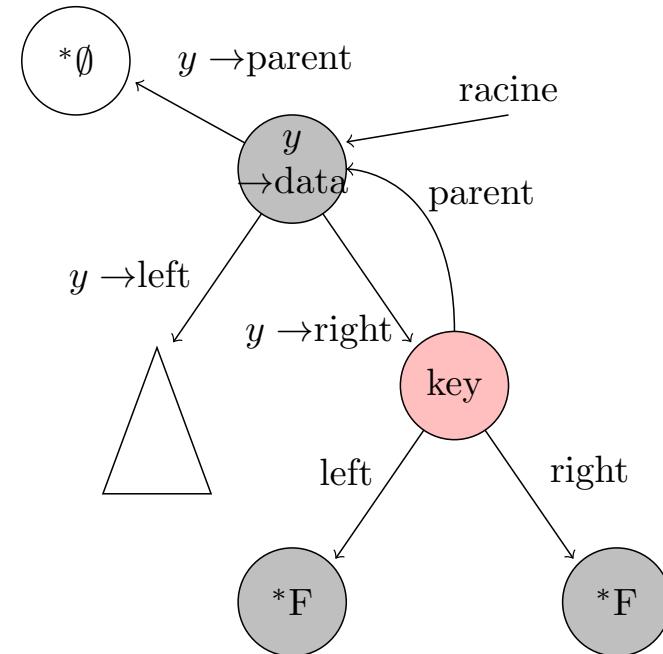
Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;
 - Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;
 - insertionColoriage(nœud)



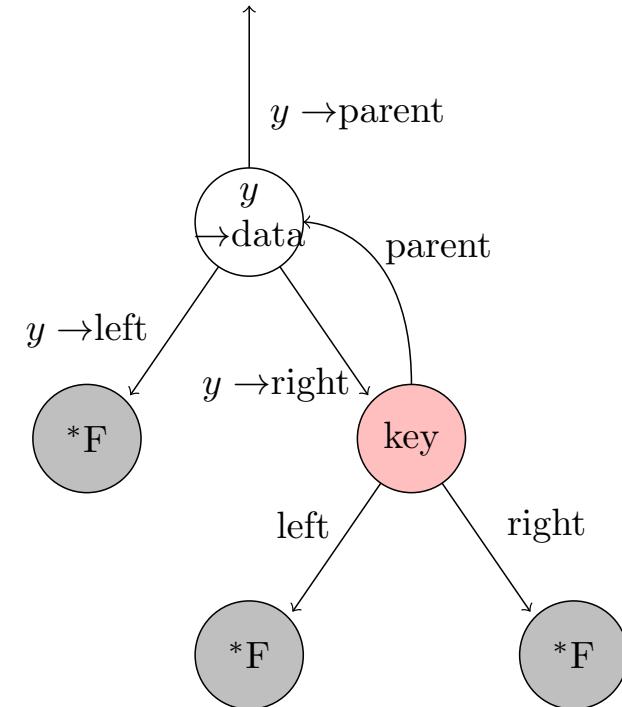
Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;
 - Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;
 - insertionColoriage(nœud)



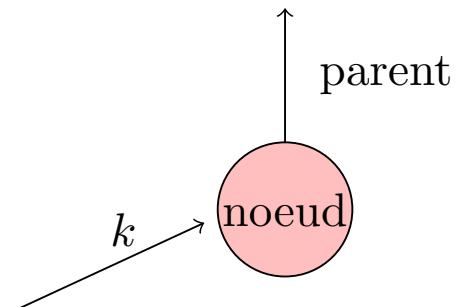
Insertion

- InsertionRougeNoir(int valeur)
 - ...
 - nœud → parent ← y;
 - Si($y = * \emptyset$)
 - ✓ racine ← nœud;
 - Sinon
 - ✓ Si(nœud → donnée < y → donnée)
 - y → gauche ← nœud;
 - ✓ Sinon
 - y → droit ← nœud;
 - Si (nœud → parent == * \emptyset)
 - ✓ nœud → couleur ← Noir;
 - ✓ retourne ;
 - Si(nœud → parent → parent == * \emptyset)
 - ✓ retourne ;
 - insertionColoriage(nœud)



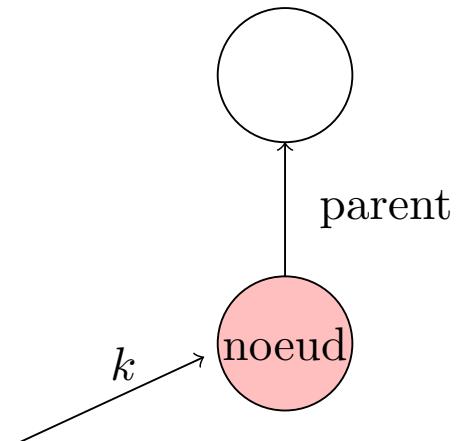
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



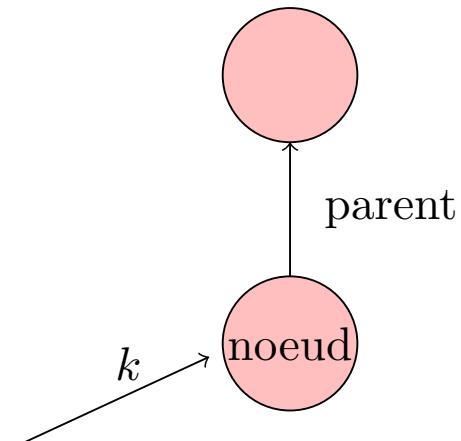
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



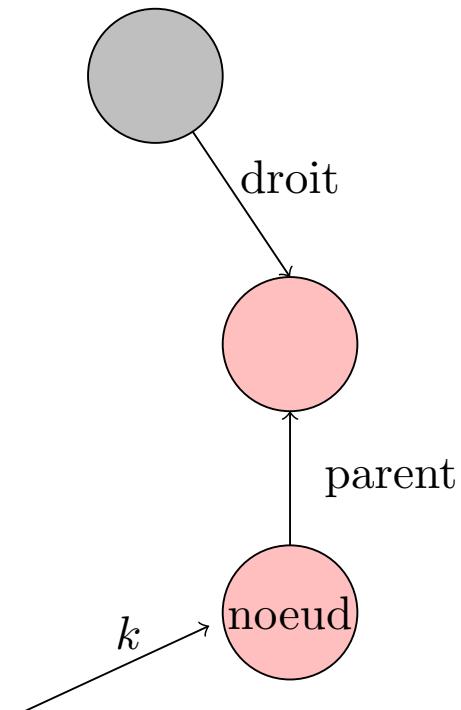
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



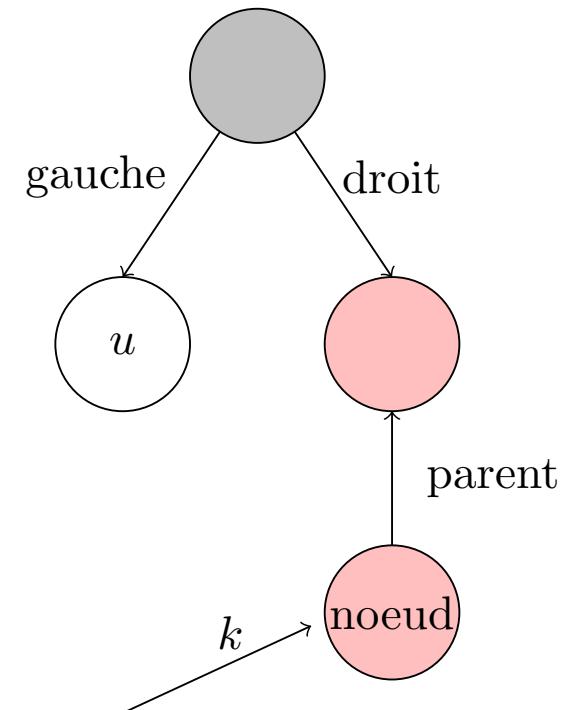
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftrightarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftrightarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftrightarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftrightarrow Rouge$;
 - $nœud \leftrightarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



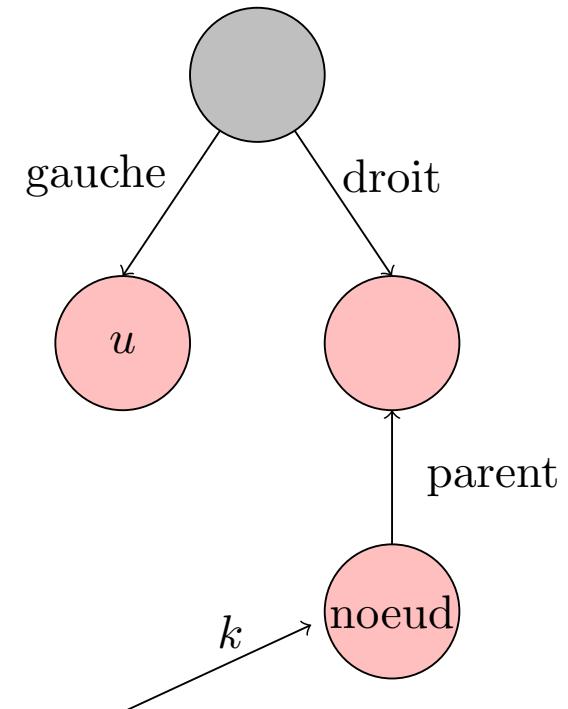
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u ;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



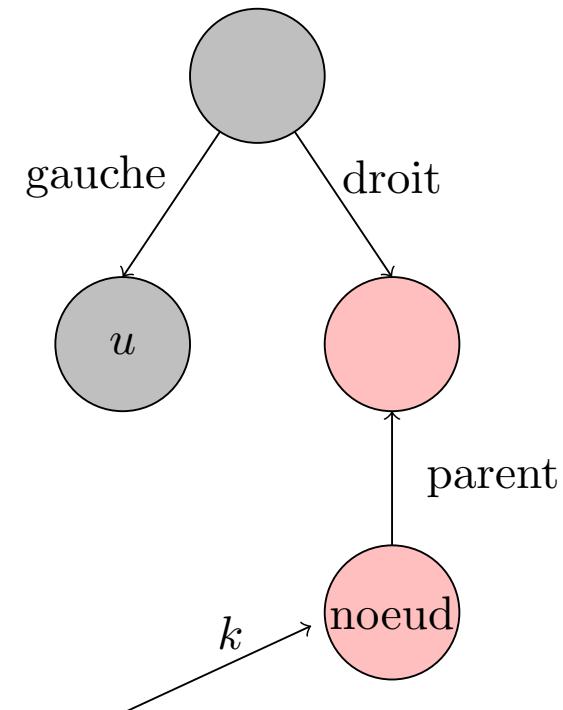
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u ;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



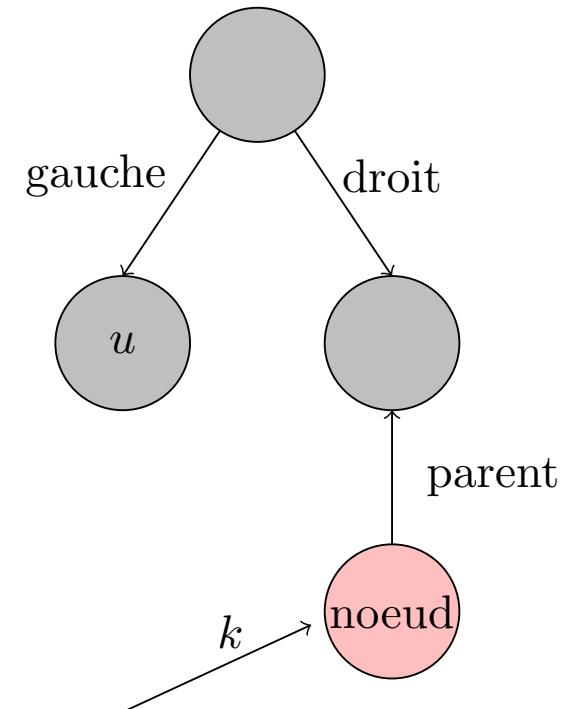
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u ;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



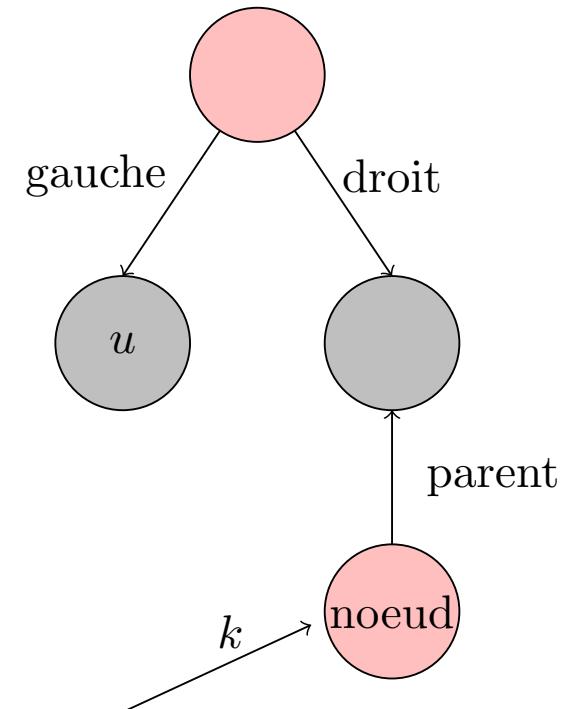
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



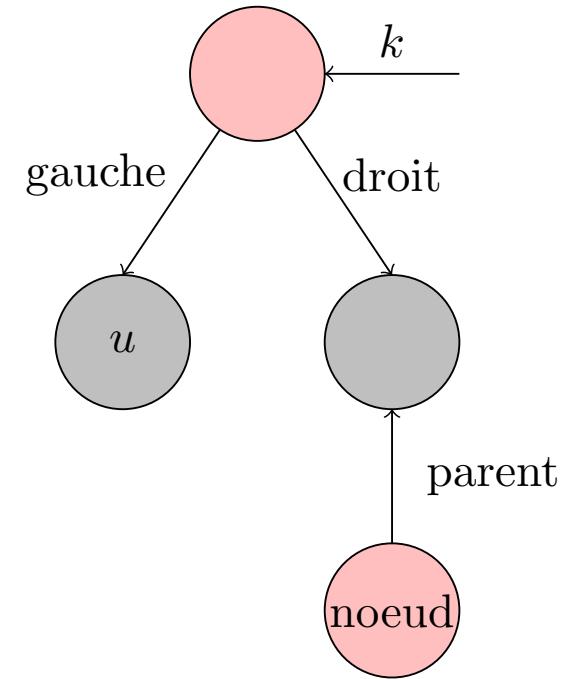
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



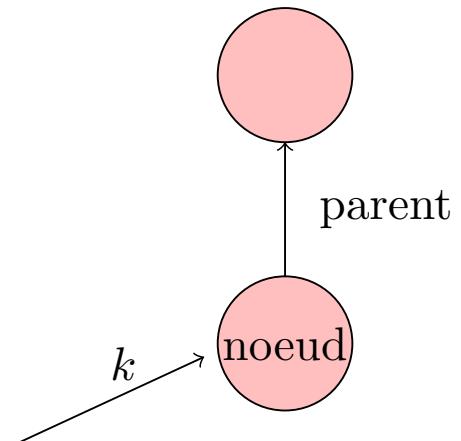
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u ;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur ==$ Rouge)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur ==$ Rouge)
 - $u \rightarrow couleur \leftarrow$ Noir;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow$ Noir;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow$ Rouge;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



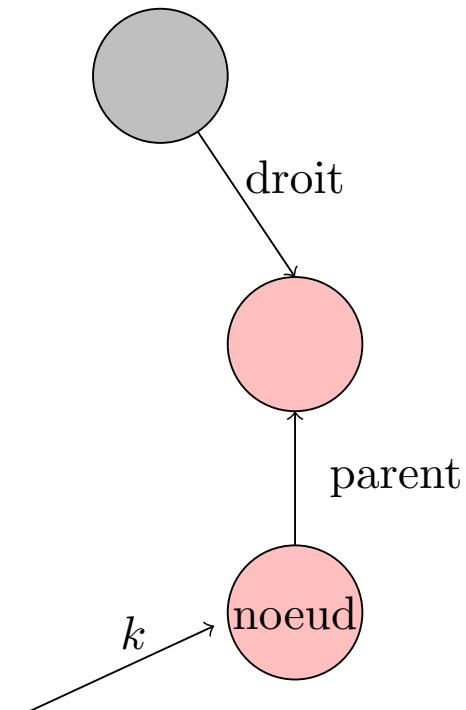
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche($nœud$);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)



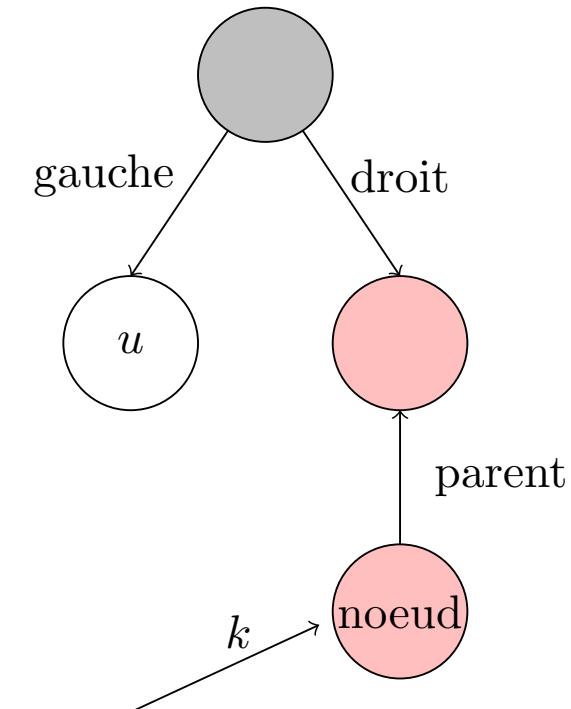
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche($nœud$);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ...



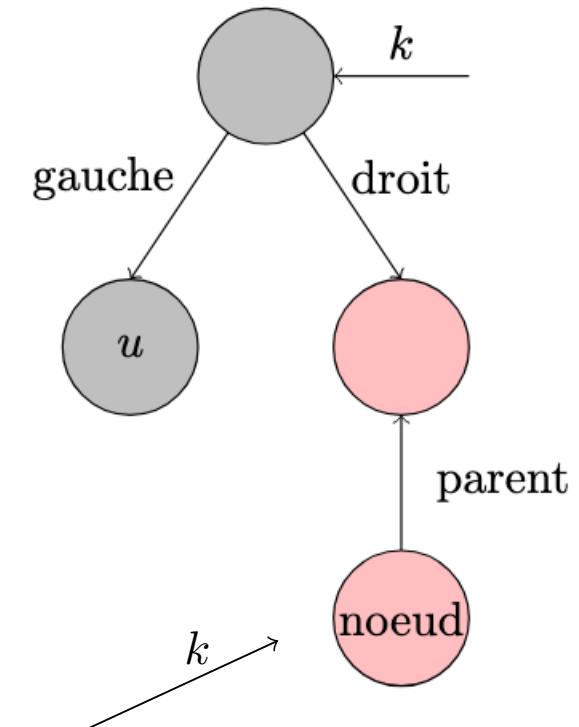
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche;$
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent;$
 - ZigZigGauche(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir;$
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge;$
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ..



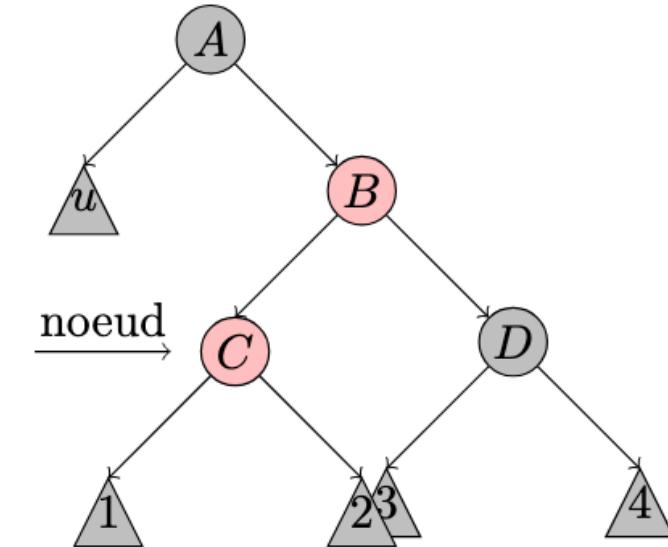
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Si ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche($nœud$);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ..



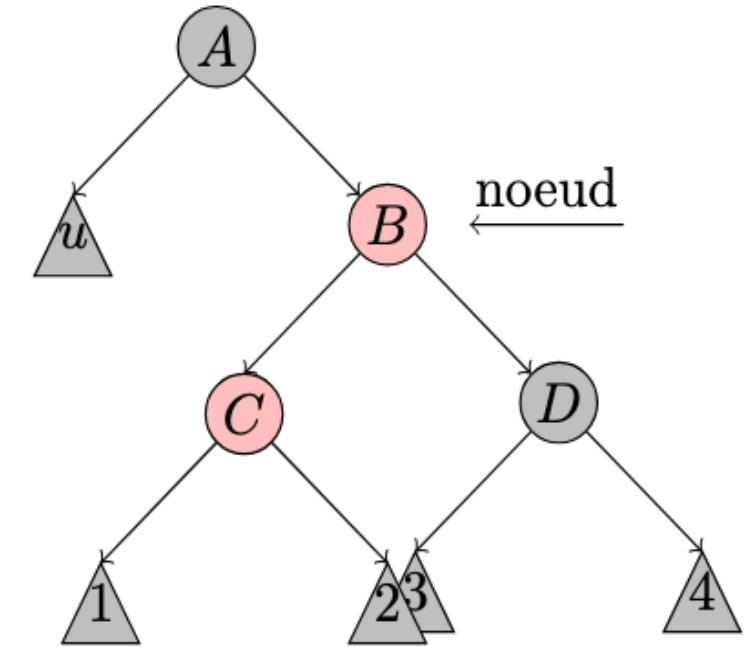
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → gauche)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit(nœud → parent → parent)
- ✓ ..



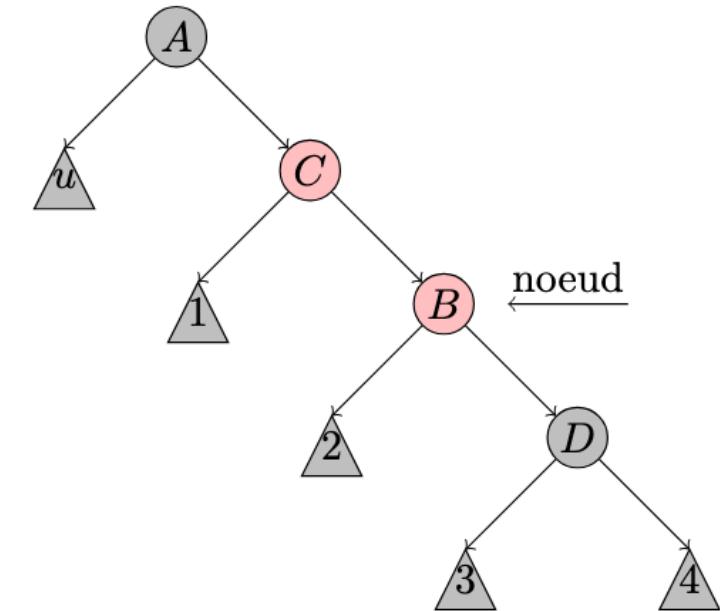
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → gauche)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit(nœud → parent → parent)
- ✓ ..



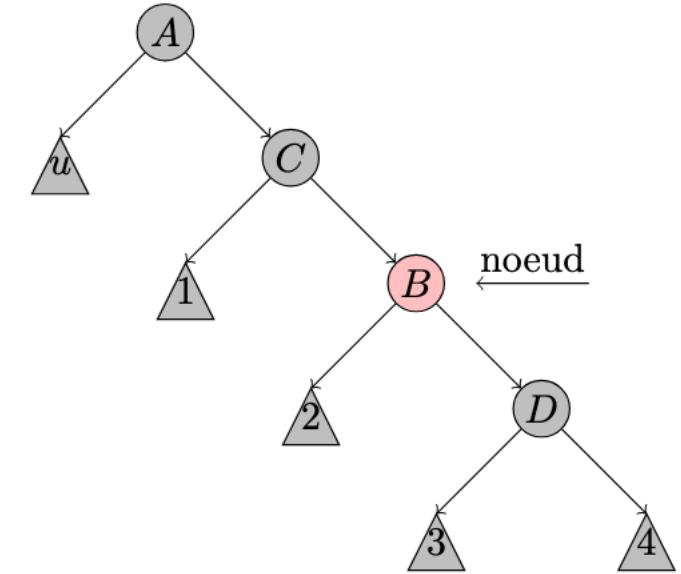
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ..



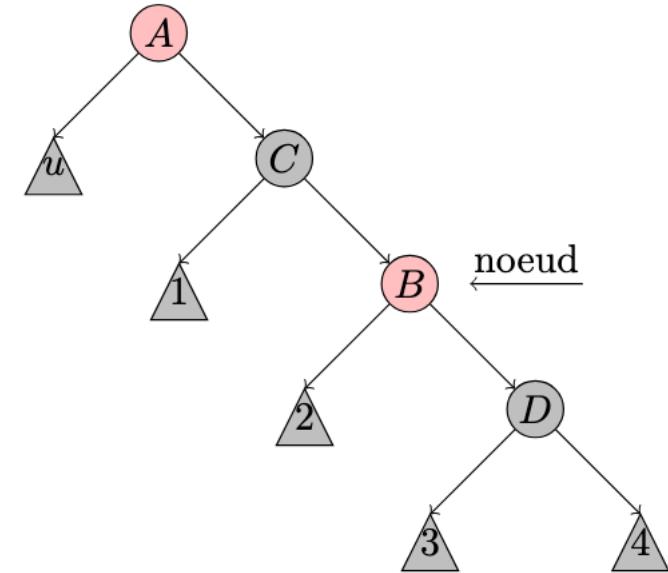
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → gauche)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ..



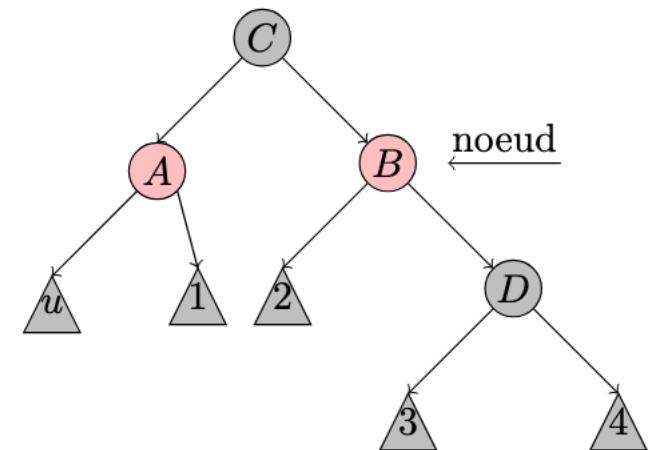
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche($nœud$);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ..



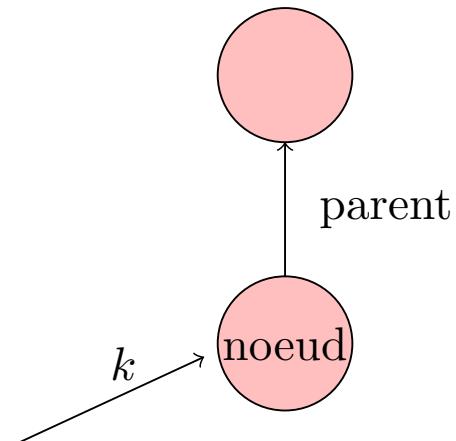
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent] → droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow gauche$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigGauche(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigDroit($nœud \rightarrow parent \rightarrow parent$)
 - ✓ ..



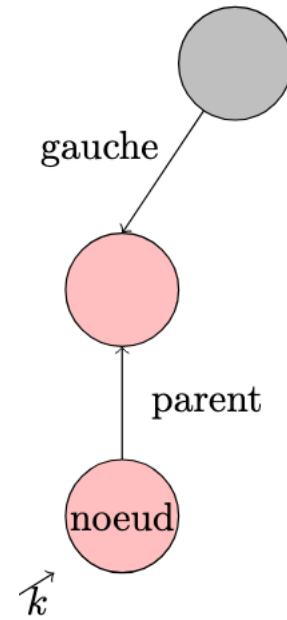
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ droit)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



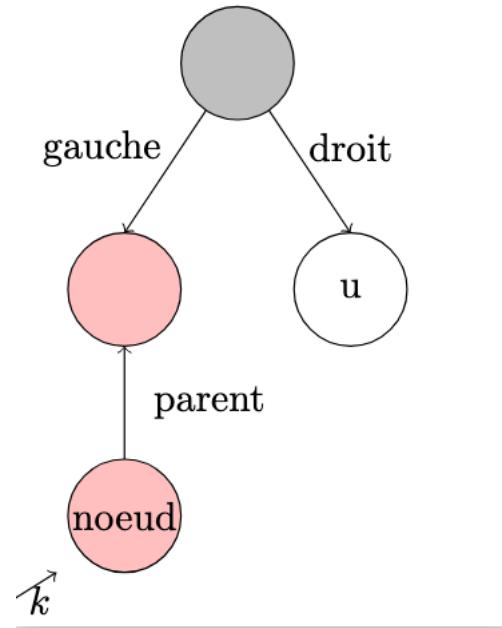
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Sinon ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftrightarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftrightarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftrightarrow Rouge$;
 - $nœud \leftrightarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



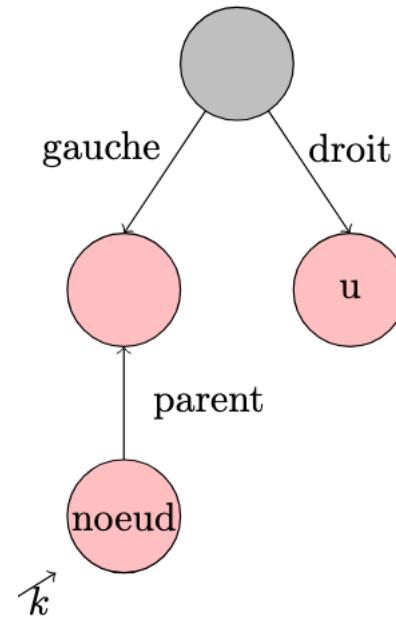
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Sinon (nœud → parent == [(nœud → parent) → parent] → gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit;$
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir;$
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir;$
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge;$
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent;$
 - Sinon
 - ...



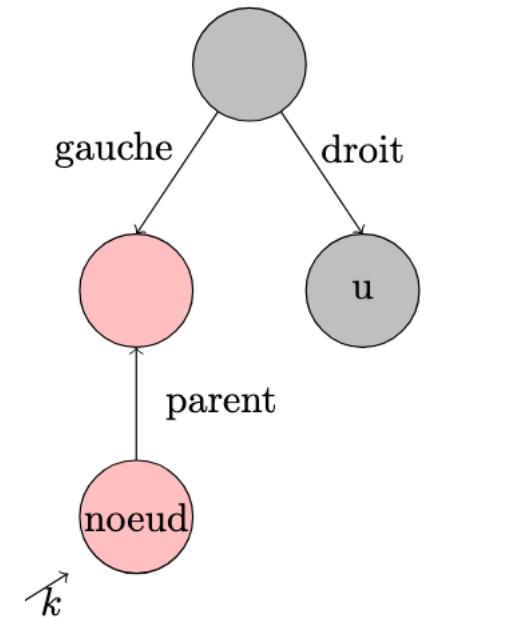
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Sinon ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



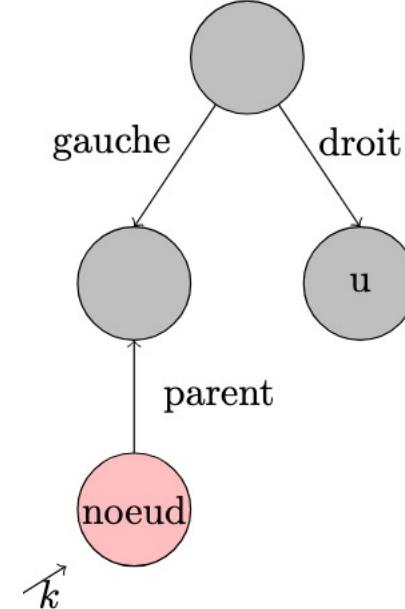
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u ;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Sinon ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



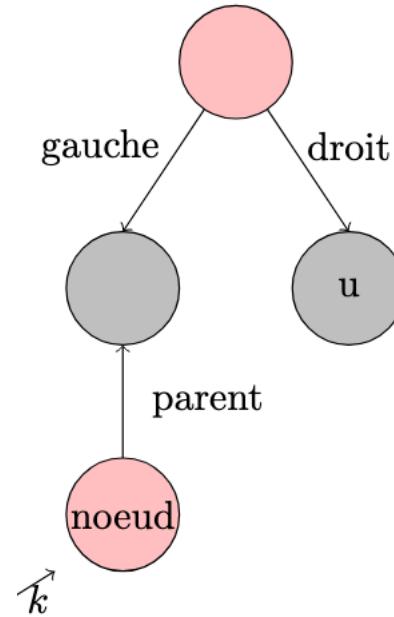
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Sinon ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



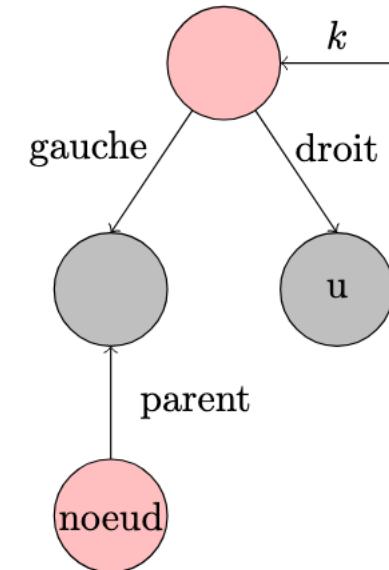
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Sinon ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



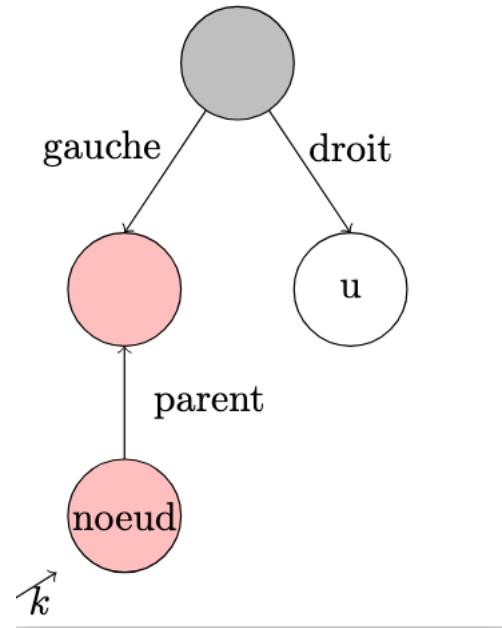
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que ($nœud \rightarrow parent \rightarrow couleur == Rouge$)
 - ✓ Sinon ($nœud \rightarrow parent == [(nœud \rightarrow parent) \rightarrow parent] \rightarrow gauche$)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge$;
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent$;
 - Sinon
 - ...



InsertionColoriage

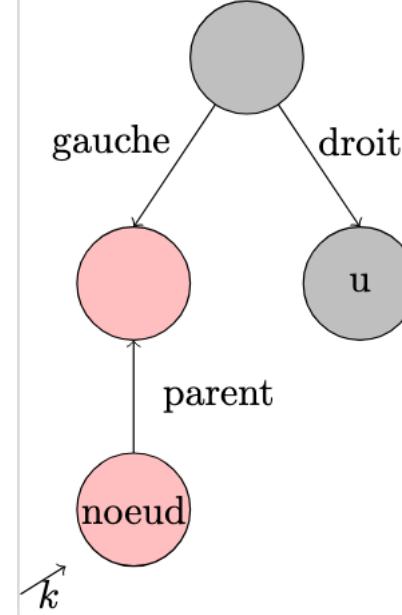
- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Sinon (nœud → parent == [(nœud → parent) → parent] → gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit;$
 - Si ($u \rightarrow couleur == Rouge$)
 - $u \rightarrow couleur \leftarrow Noir;$
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir;$
 - $[(nœud \rightarrow parent) \rightarrow parent] \rightarrow couleur \leftarrow Rouge;$
 - $nœud \leftarrow nœud \rightarrow parent \rightarrow parent;$
 - Sinon



InsertionColoriage

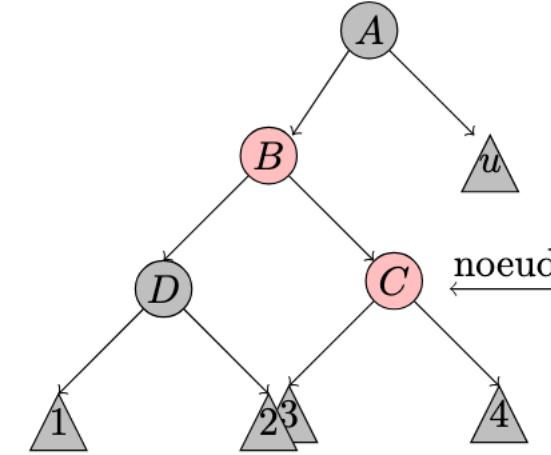
- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit;$
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon

- Si (nœud == nœud → parent → droit)
 - $nœud \leftarrow nœud \rightarrow parent;$
 - ZigZigDroit(nœud);
- $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir;$
- $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge;$
- ZigZigGauche(nœud → parent → parent)



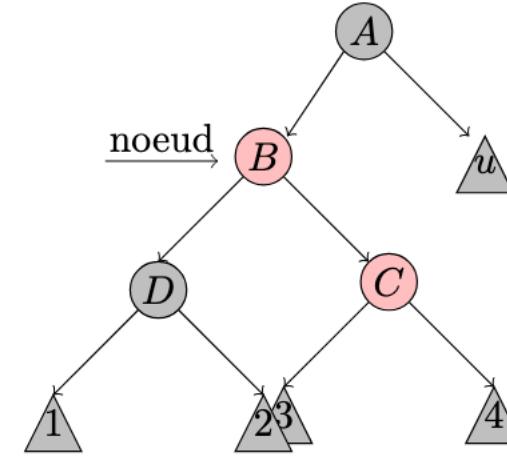
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → droit)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigDroit(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigGauche(nœud → parent → parent)



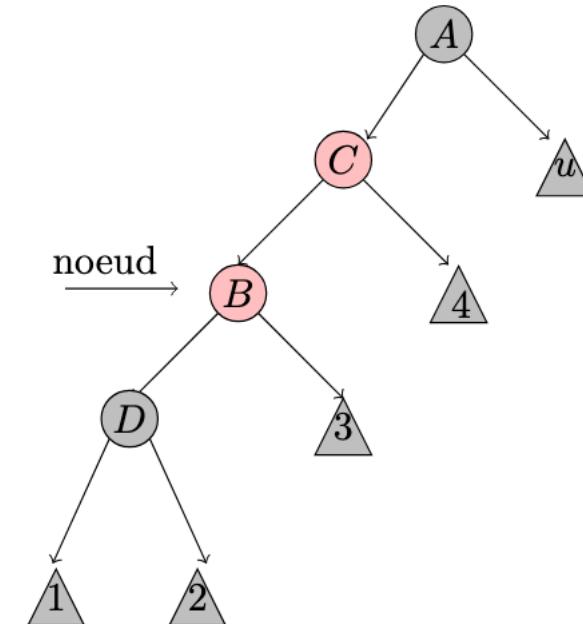
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ gauche)
 - $u \leftrightarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → droit)
 - $nœud \leftrightarrow nœud \rightarrow parent$;
 - ZigZigDroit(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigGauche(nœud → parent → parent)



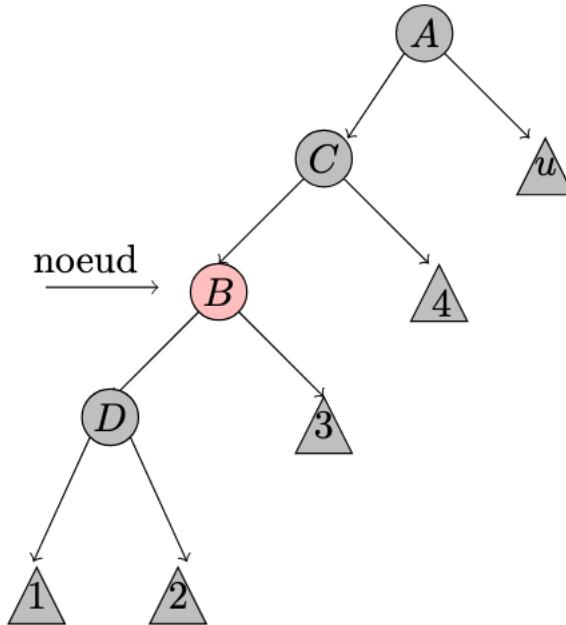
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → droit)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigDroit(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigGauche(nœud → parent → parent)



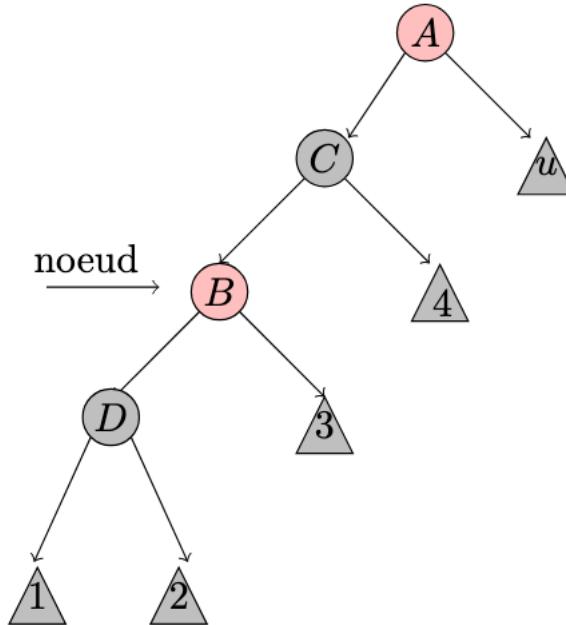
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → droit)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigDroit(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigGauche(nœud → parent → parent)



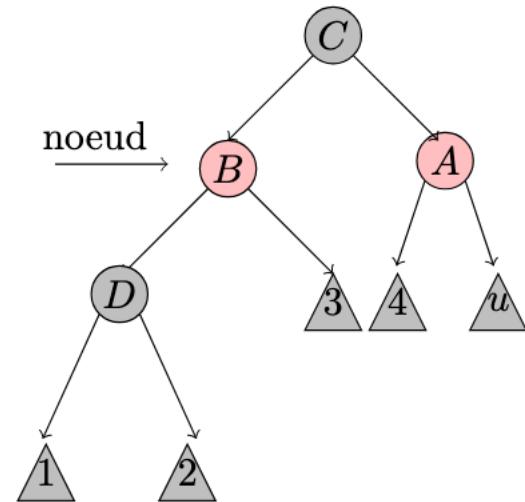
InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent]→ gauche)
 - $u \leftarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit;$
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si (nœud == nœud → parent → droit)
 - $nœud \leftarrow nœud \rightarrow parent;$
 - ZigZigDroit(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir;$
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge;$
 - ZigZigGauche(nœud → parent → parent)

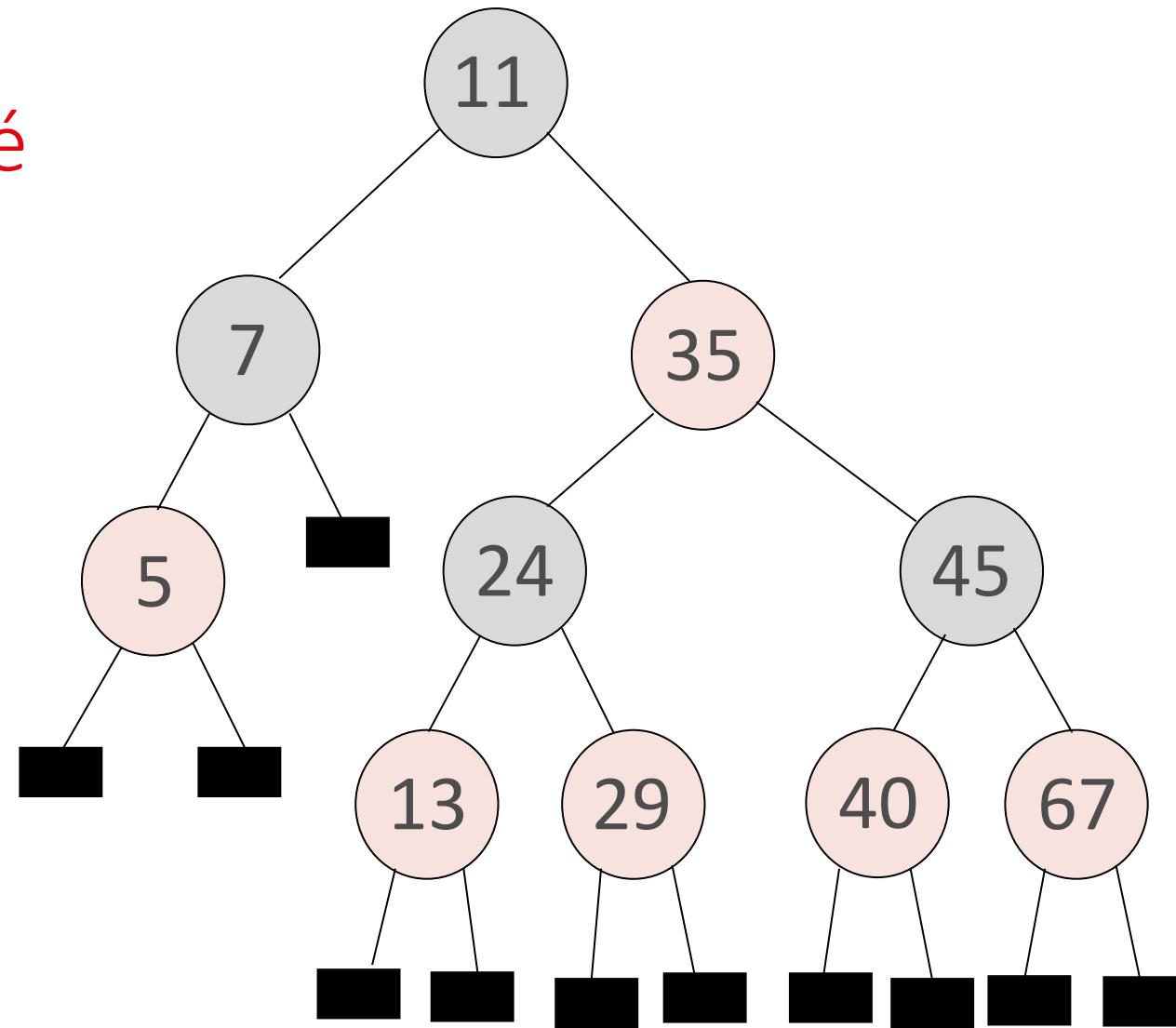


InsertionColoriage

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que (nœud → parent → couleur == Rouge)
 - ✓ Si (nœud → parent == [(nœud → parent) → parent] → gauche)
 - $u \leftrightarrow [(nœud \rightarrow parent) \rightarrow parent] \rightarrow droit$;
 - Si ($u \rightarrow couleur == Rouge$) {...}
 - Sinon
 - Si ($nœud == nœud \rightarrow parent \rightarrow droit$)
 - $nœud \leftarrow nœud \rightarrow parent$;
 - ZigZigDroit(nœud);
 - $nœud \rightarrow parent \rightarrow couleur \leftarrow Noir$;
 - $nœud \rightarrow parent \rightarrow parent \rightarrow couleur \leftarrow Rouge$;
 - ZigZigGauche($nœud \rightarrow parent \rightarrow parent$)

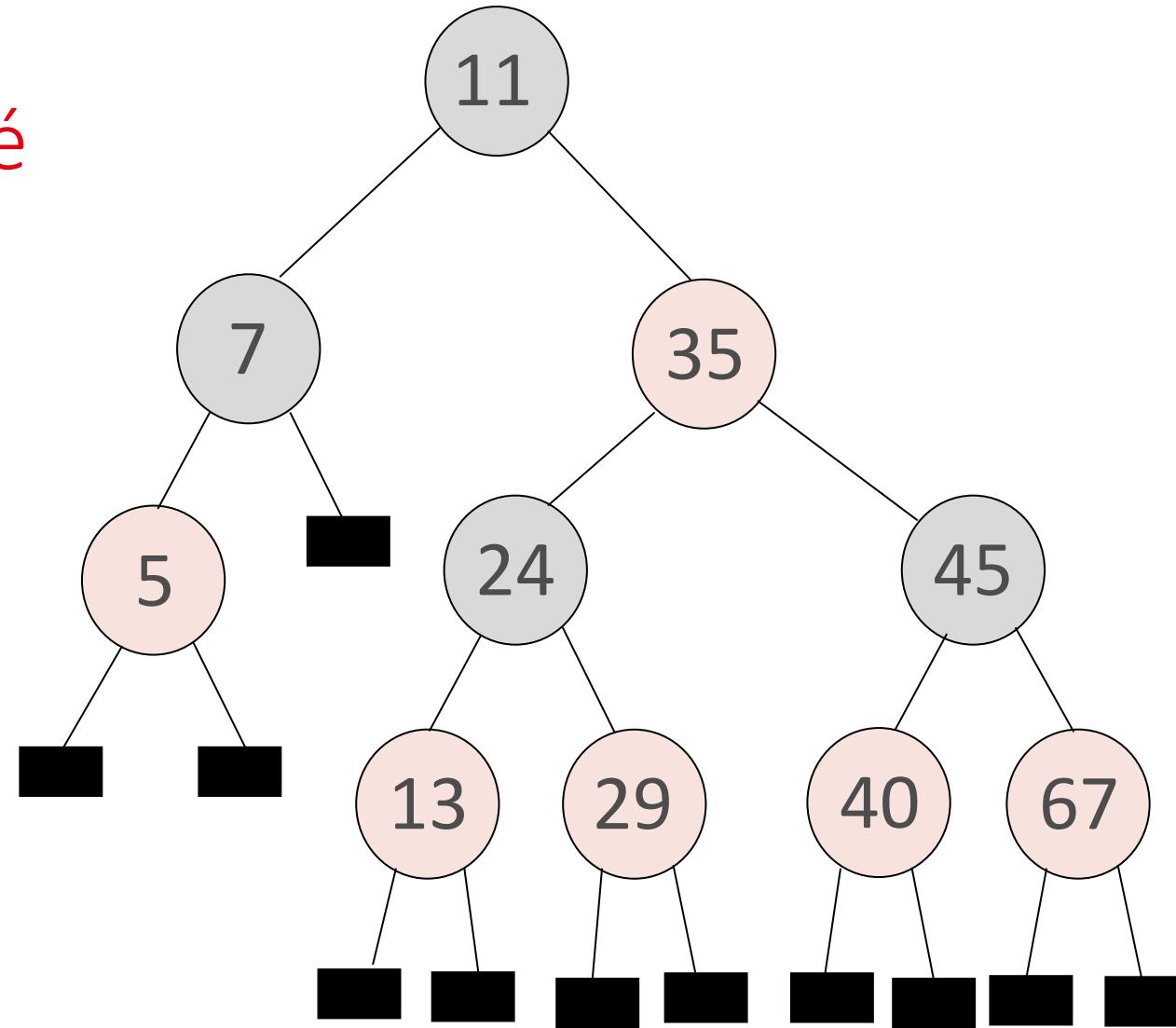


Exemple détaillé



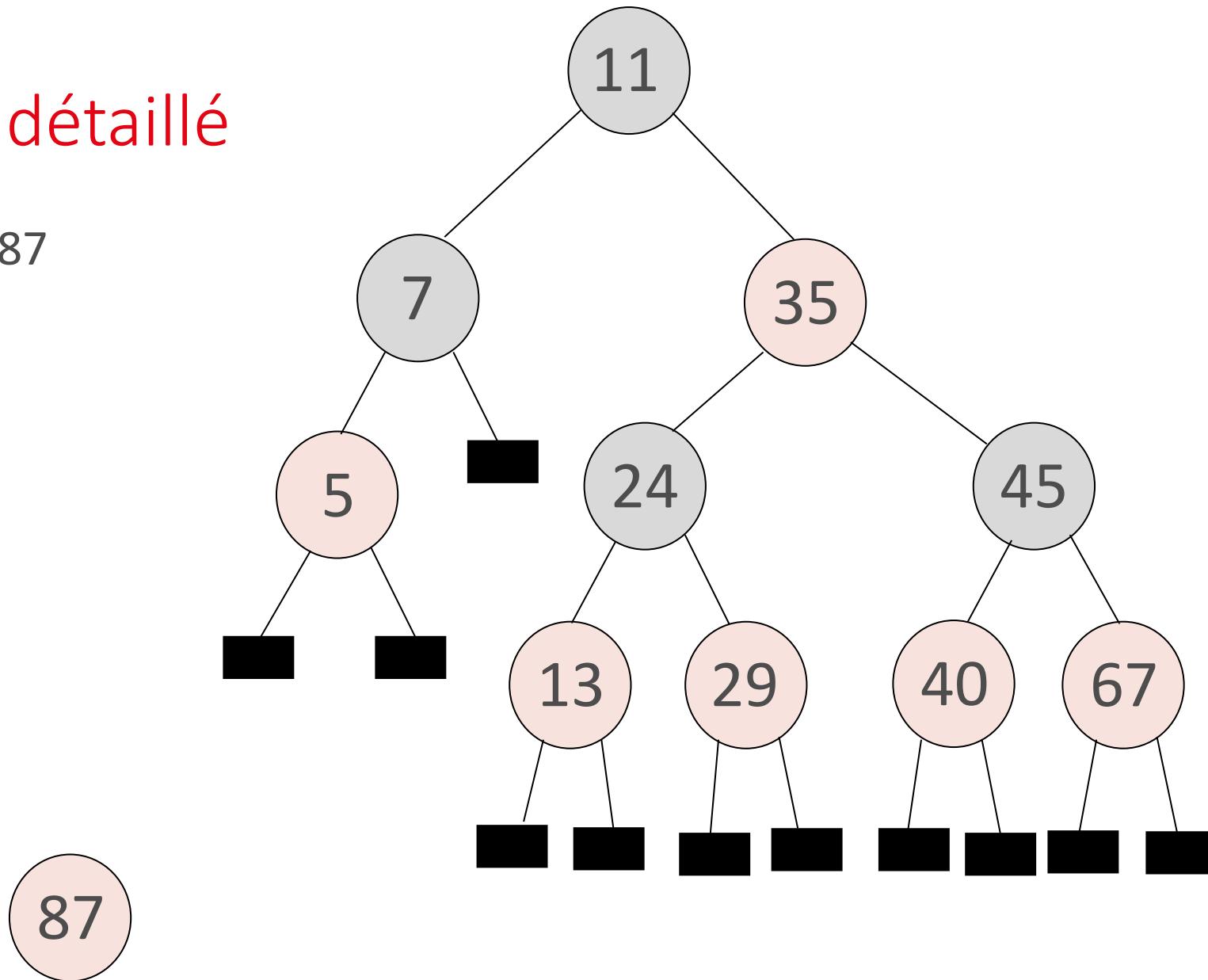
Exemple détaillé

Insertion de 87



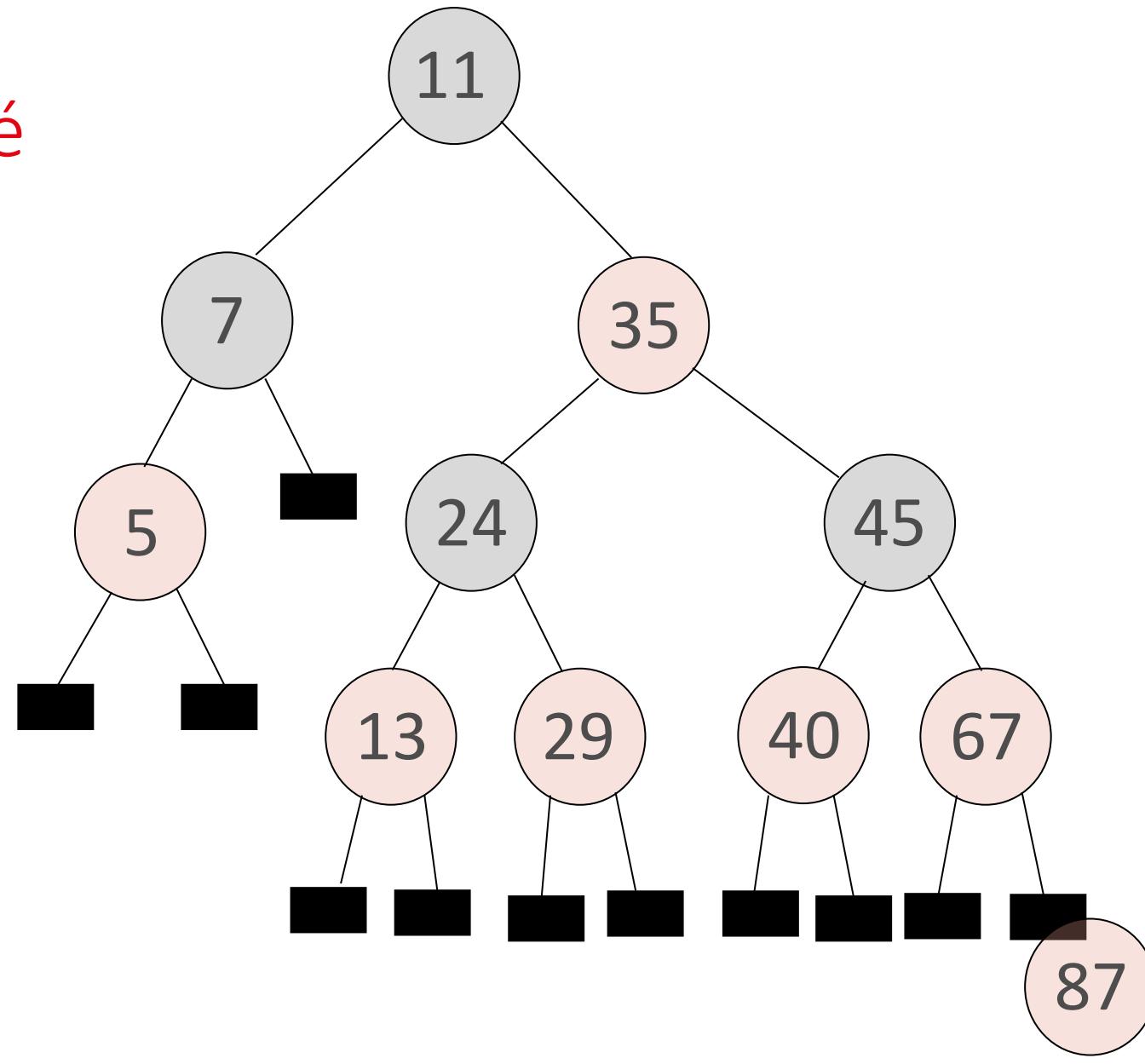
Exemple détaillé

Insertion de 87

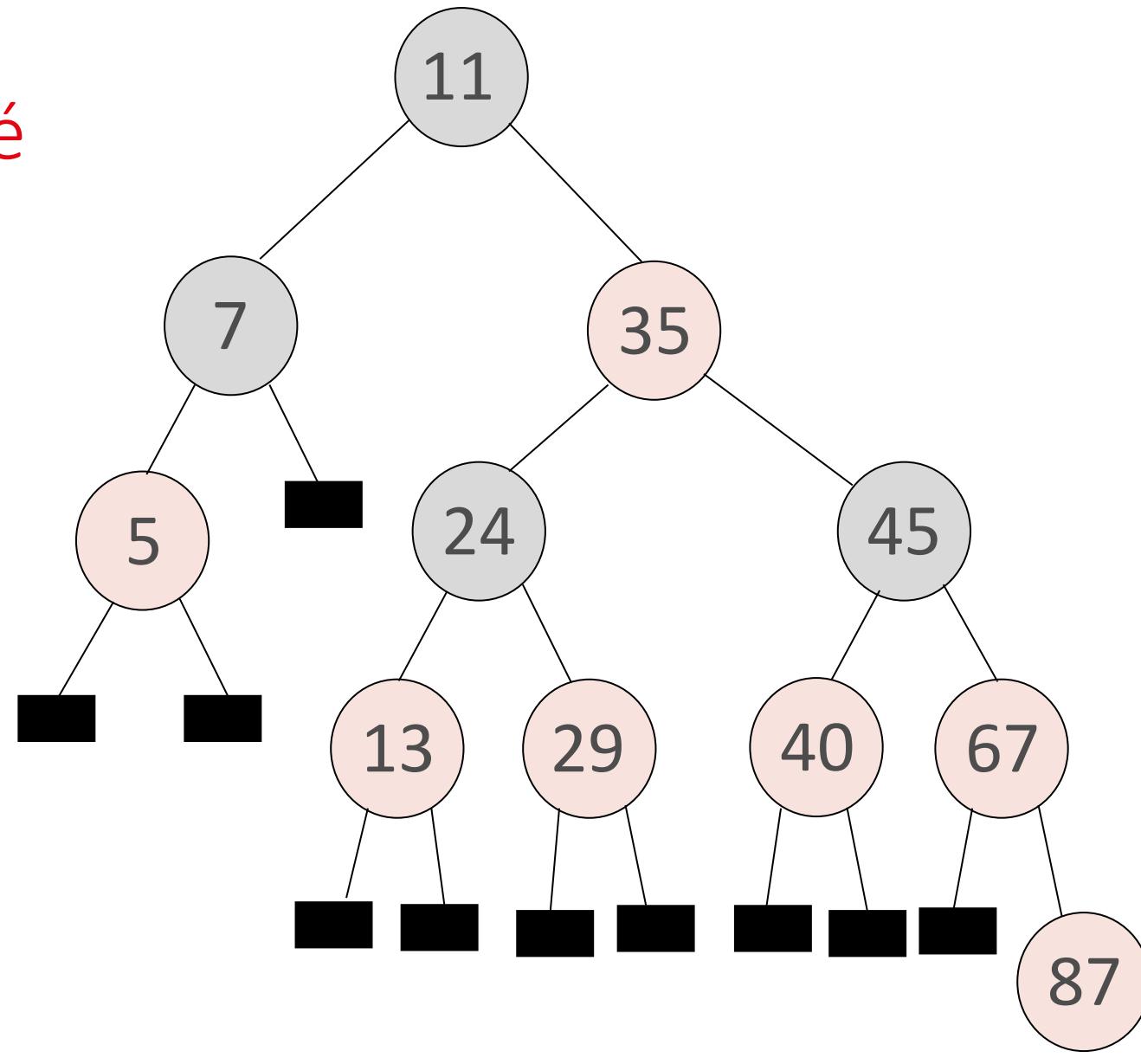


Exemple détaillé

Insertion de 87

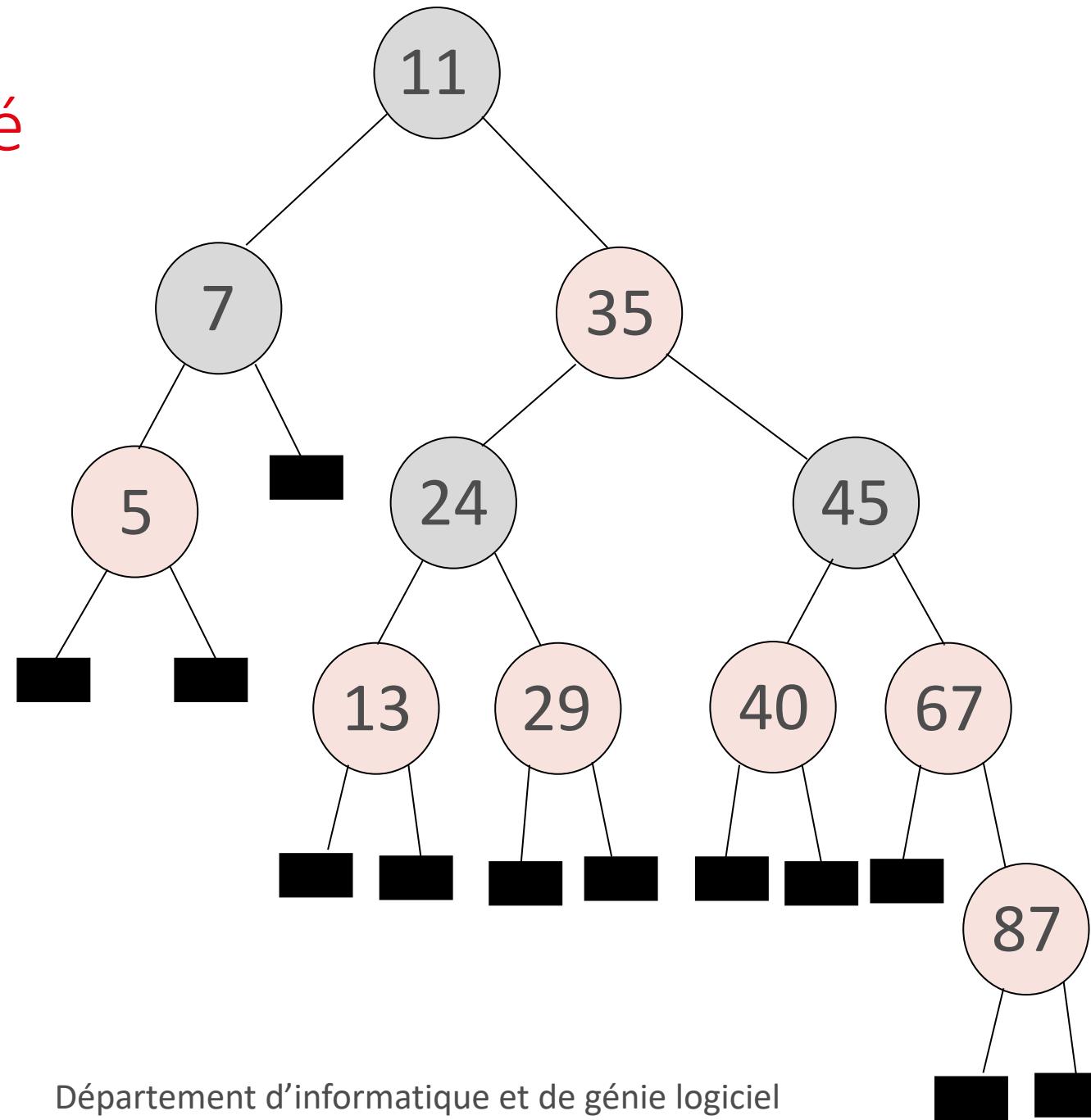


Exemple détaillé



Exemple détaillé

Insertion de 87

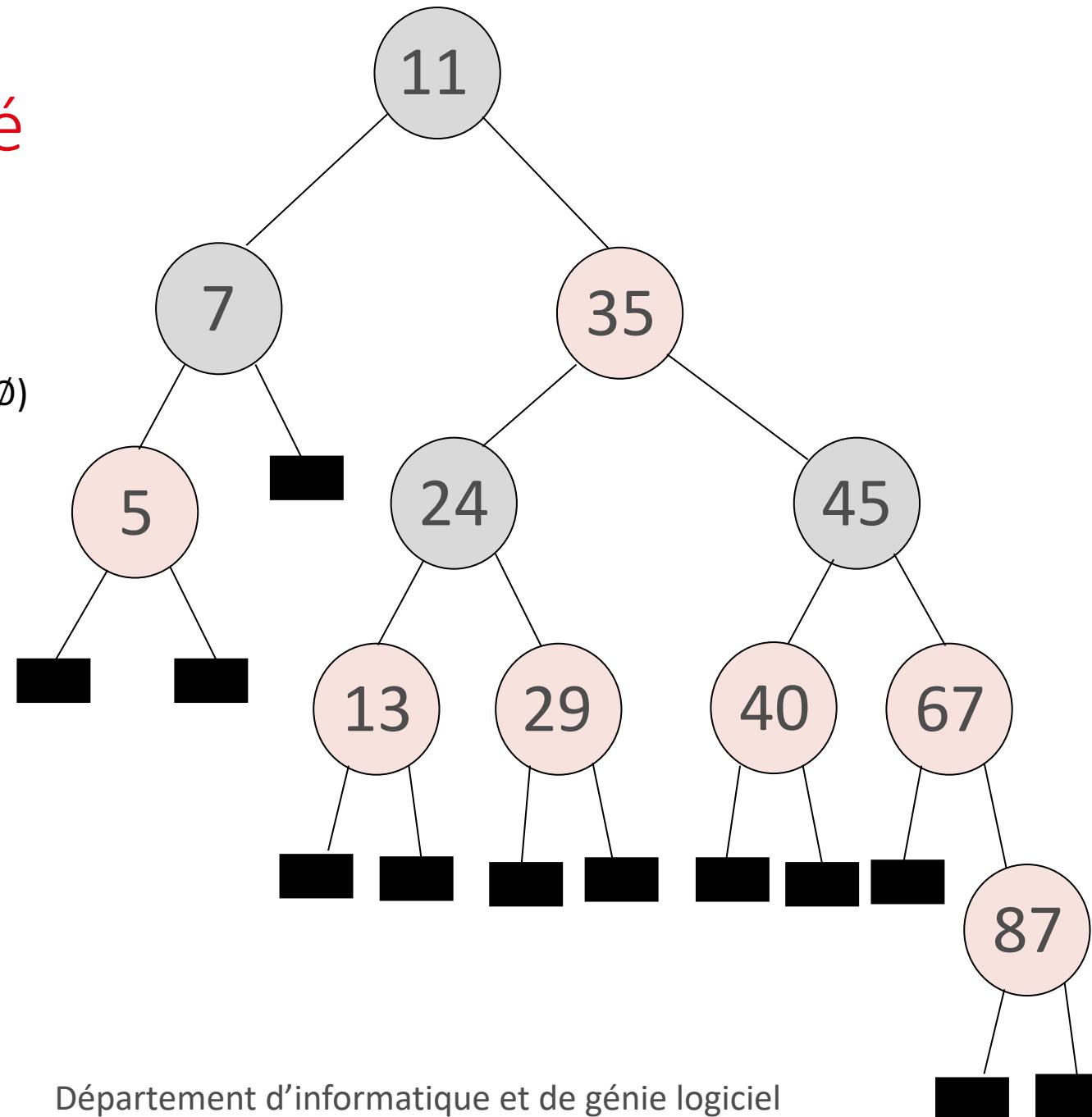


Exemple détaillé

```
Si (nœud → parent == * Ø)
    nœud → couleur ← Noir;
    retourne;
```

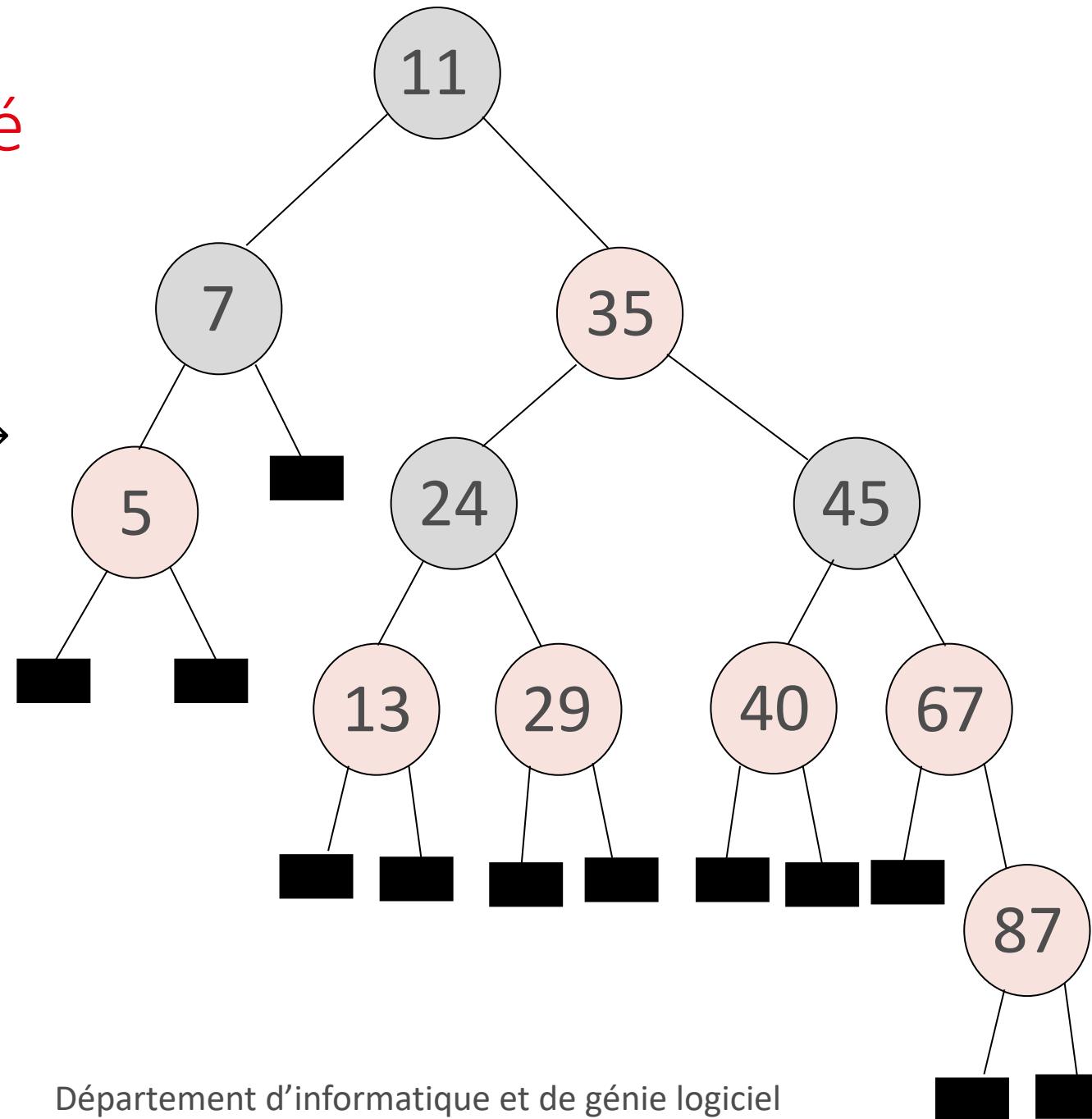
```
Si(nœud → parent → parent == * Ø)
    retourne;
```

```
insertionColoriage(nœud)
```



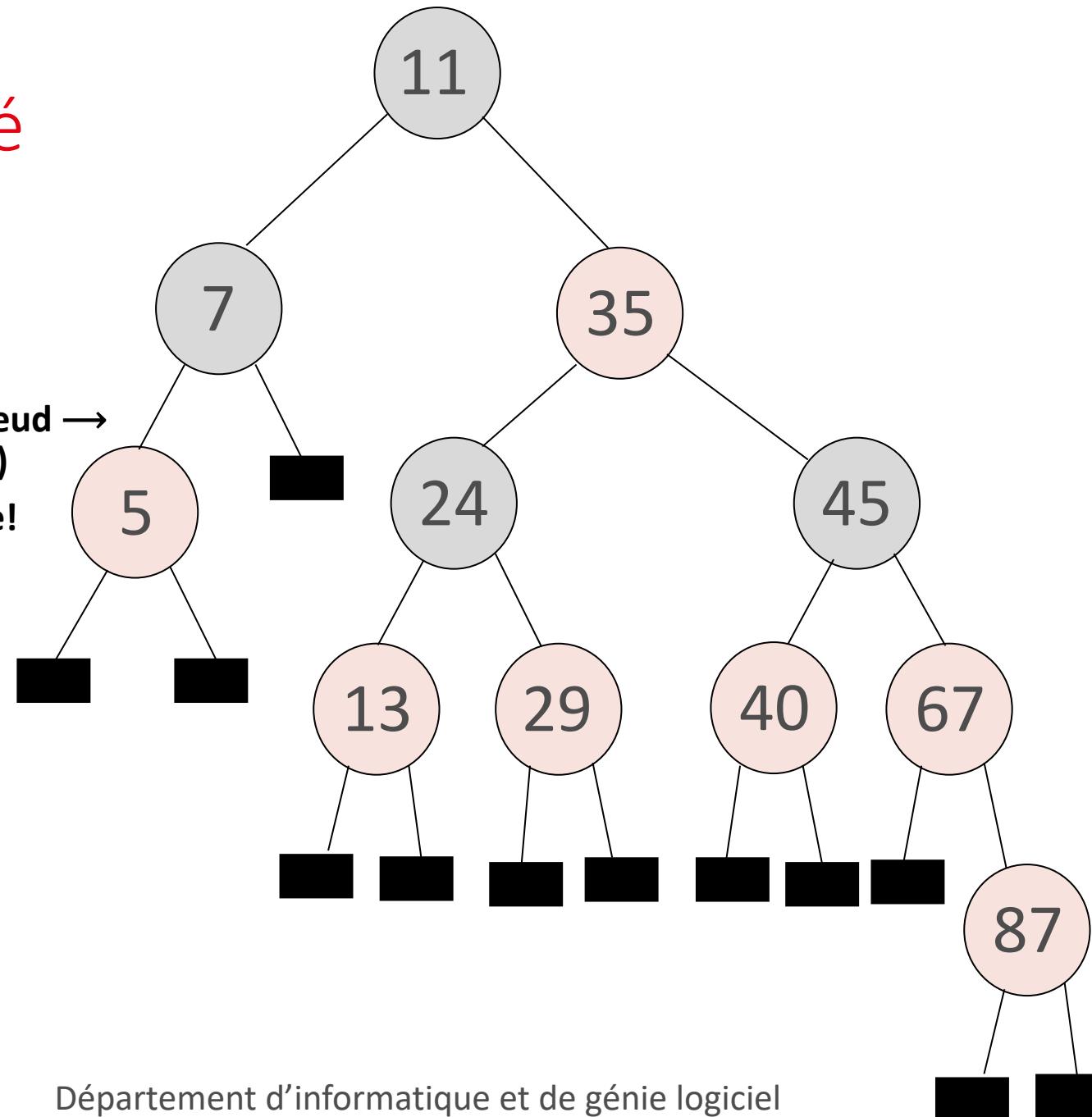
Exemple détaillé

- InsertionColoriage(nœud)
 - Nœud u;
 - Tant que
(nœud → parent → couleur
== Rouge)



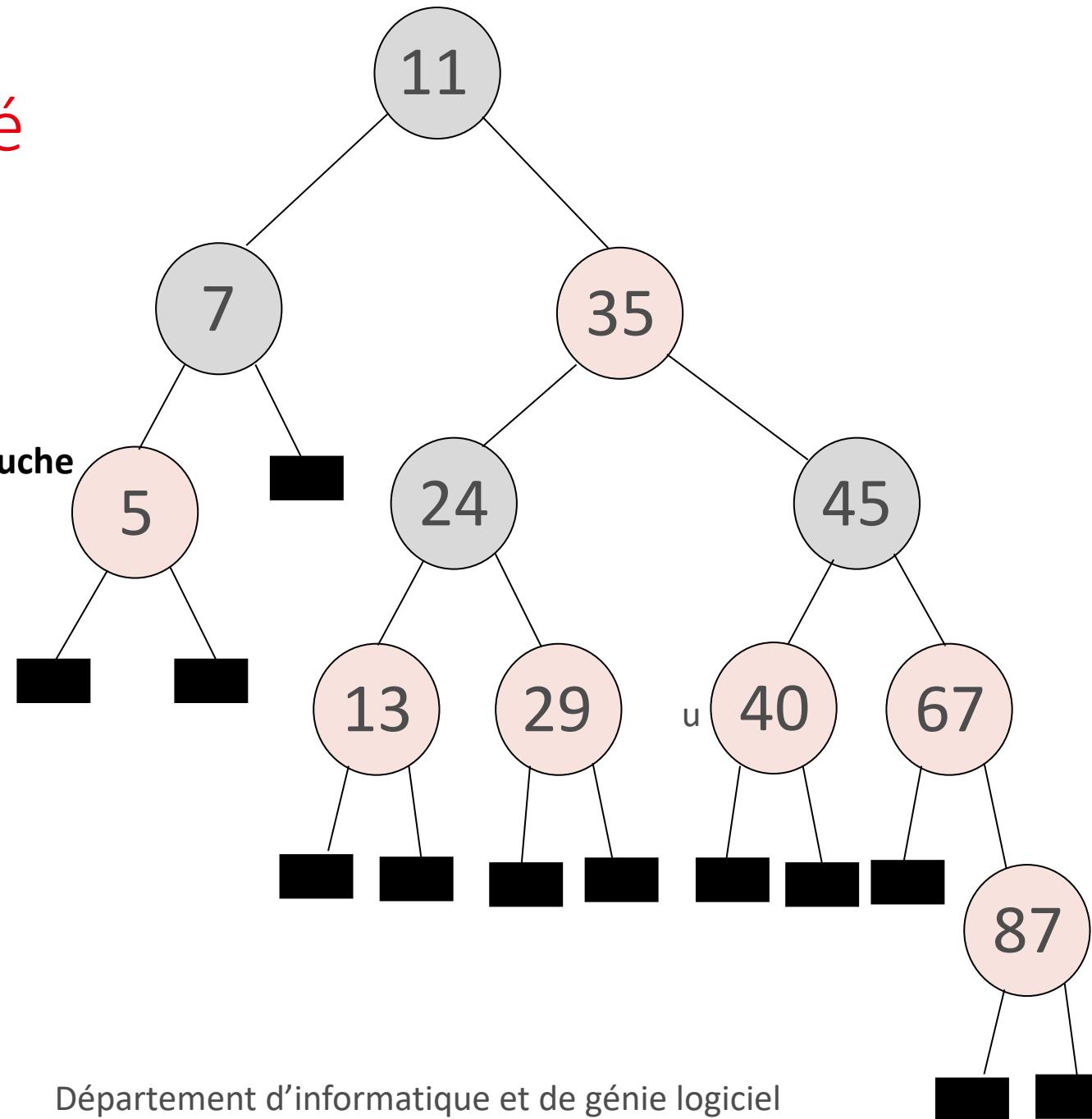
Exemple détaillé

- InsertionColoriage(nœud)
- Nœud u;
- Si $(\text{nœud} \rightarrow \text{parent} == [\text{nœud} \rightarrow \text{parent}) \rightarrow \text{parent}] \rightarrow \text{droit})$
 - Oui donc cas à droite!



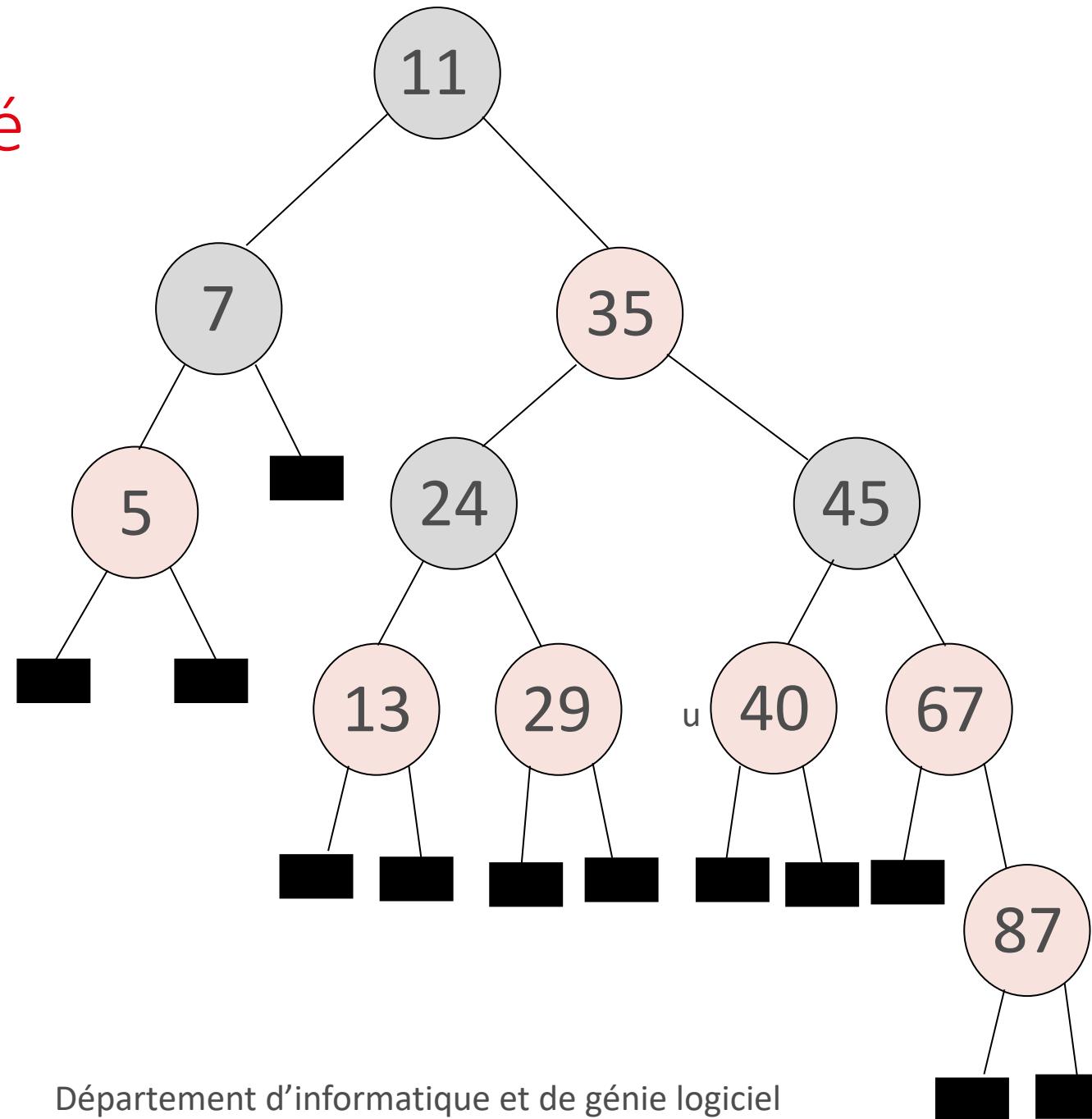
Exemple détaillé

- InsertionColoriage(nœud)
- Nœud u;
- $u \leftarrow [(\text{nœud} \rightarrow \text{parent}) \rightarrow \text{parent}] \rightarrow \text{gauche}$



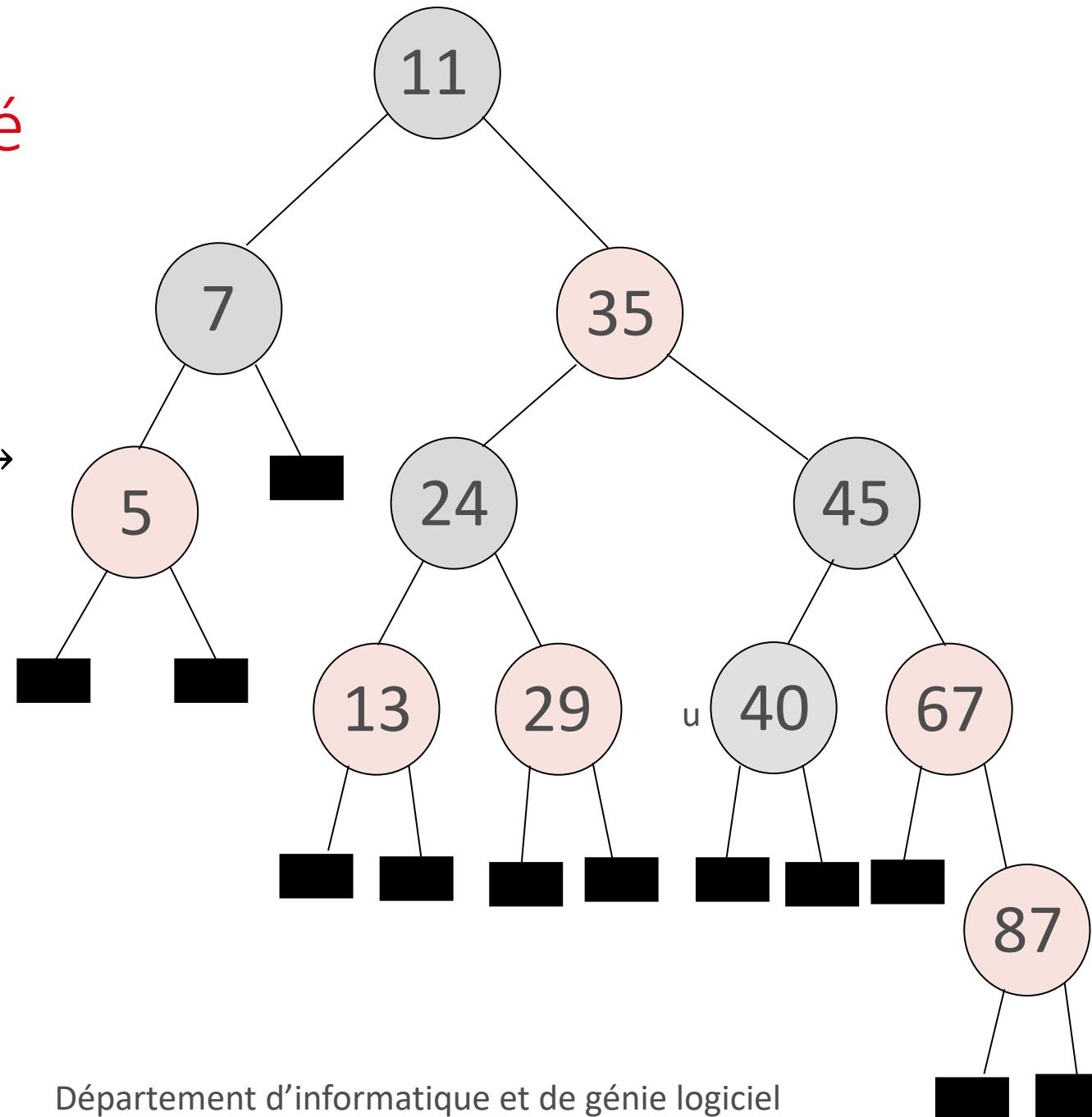
Exemple détaillé

- InsertionColoriage(nœud)
- Nœud u;
- Si ($u \rightarrow \text{couleur} == \text{Rouge}$)



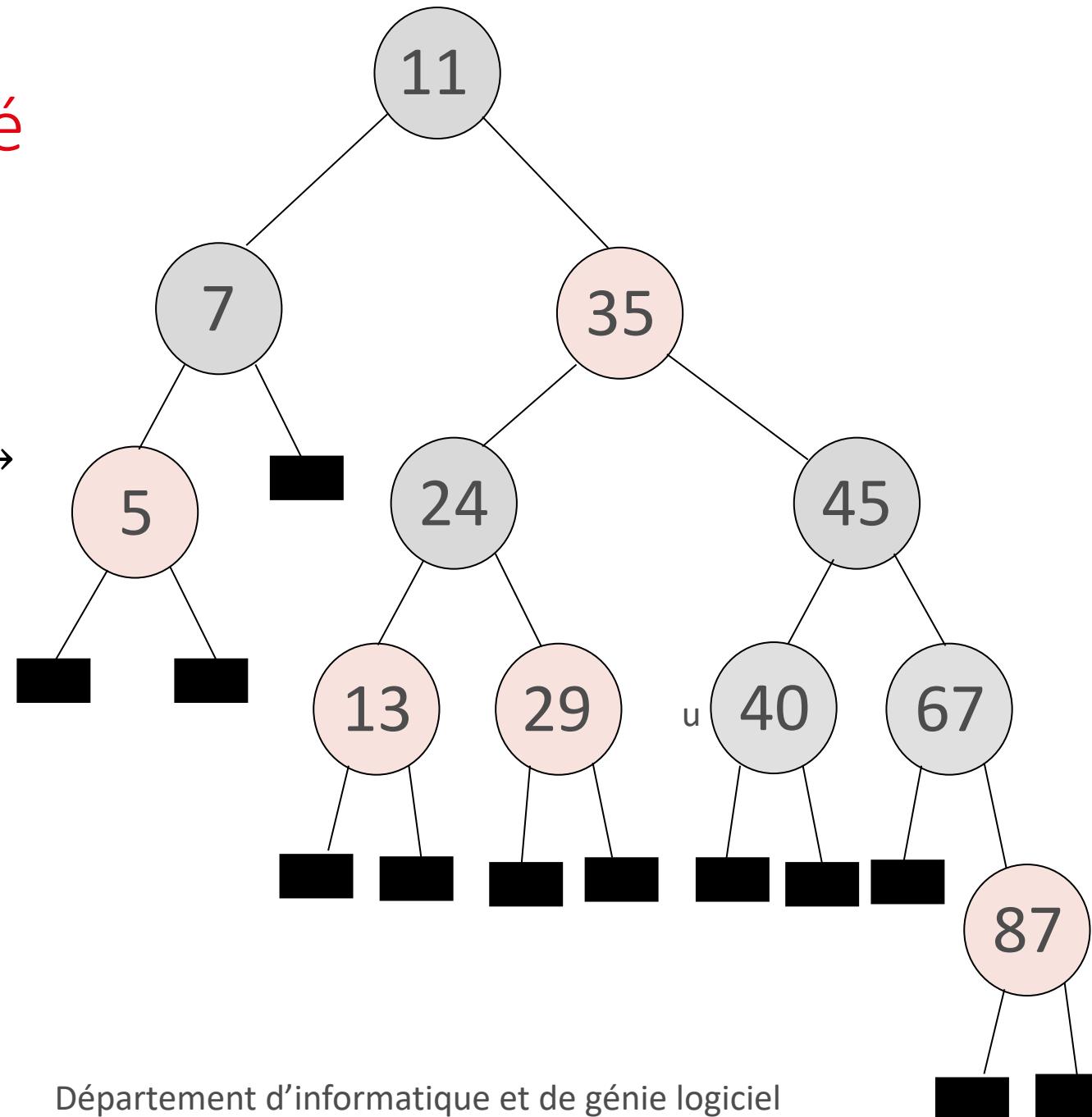
Exemple détaillé

- $u \rightarrow \text{couleur} \leftarrow \text{Noir};$
- noeud \rightarrow parent \rightarrow couleur \leftarrow Noir
- $[(\text{noeud} \rightarrow \text{parent}) \rightarrow \text{parent}] \rightarrow \text{couleur} \leftarrow \text{Rouge};$
- noeud \leftarrow noeud \rightarrow parent \rightarrow parent;



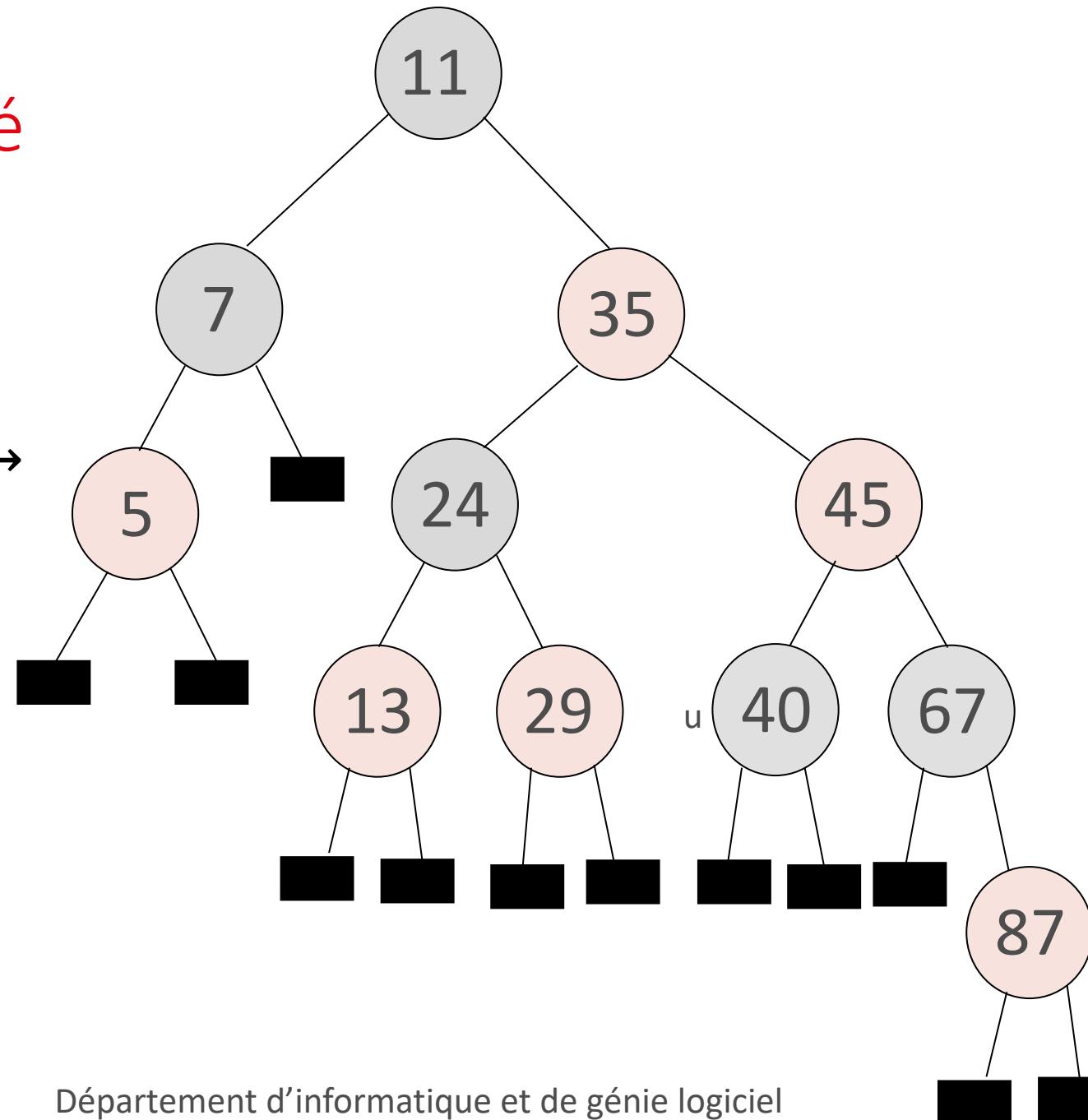
Exemple détaillé

- $u \rightarrow \text{couleur} \leftarrow \text{Noir};$
- **nœud \rightarrow parent \rightarrow couleur \leftarrow Noir**
- $[(\text{nœud} \rightarrow \text{parent}) \rightarrow \text{parent}] \rightarrow \text{couleur} \leftarrow \text{Rouge};$
- $\text{nœud} \leftarrow \text{nœud} \rightarrow \text{parent} \rightarrow \text{parent};$



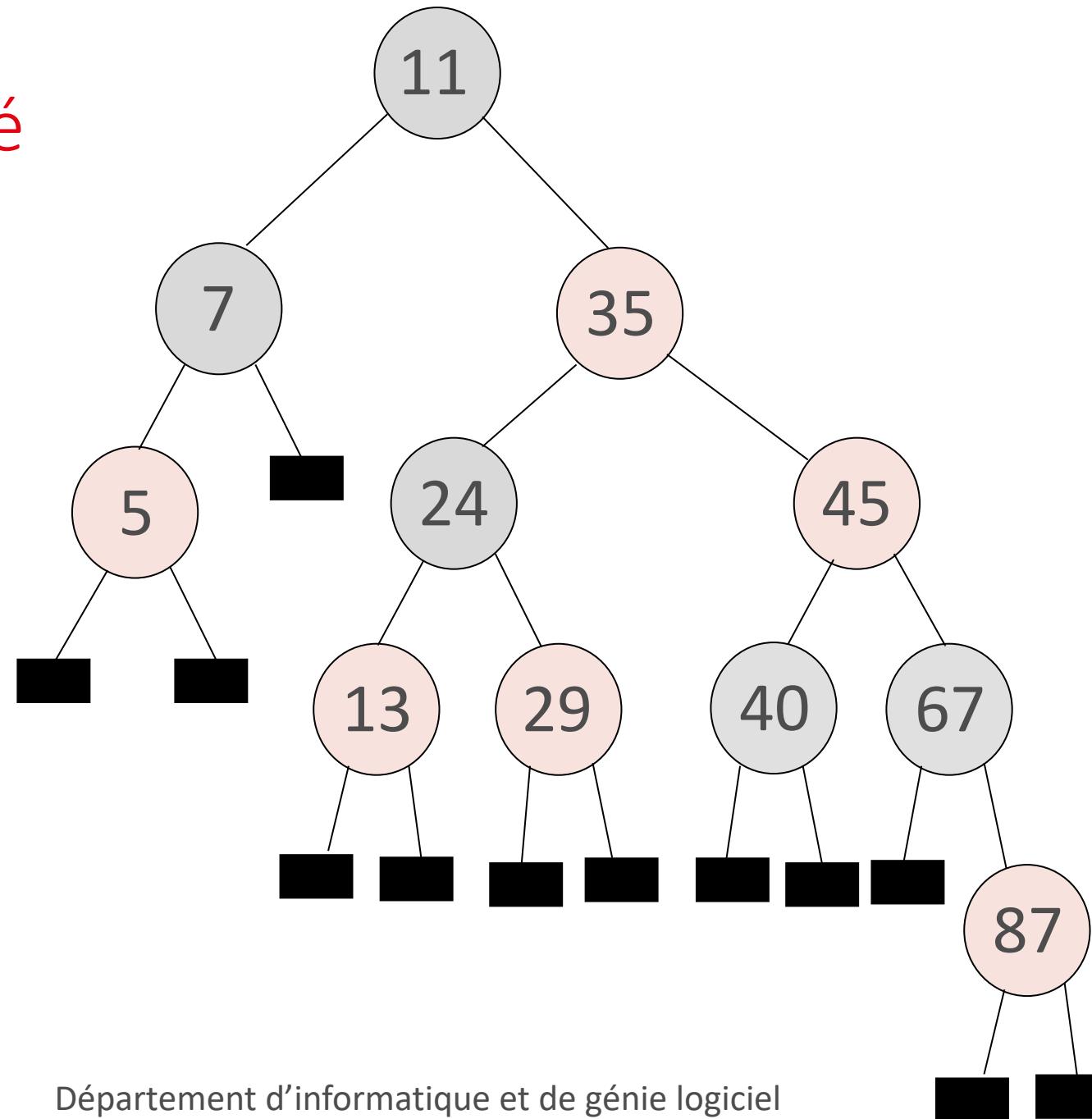
Exemple détaillé

- $u \rightarrow \text{couleur} \leftarrow \text{Noir}$;
- $\text{nœud} \rightarrow \text{parent} \rightarrow \text{couleur} \leftarrow \text{Noir}$
- $[(\text{nœud} \rightarrow \text{parent}) \rightarrow \text{parent}] \rightarrow \text{couleur} \leftarrow \text{Rouge}$;
- $\text{nœud} \leftarrow \text{nœud} \rightarrow \text{parent} \rightarrow \text{parent}$;
Ici on repart de 45



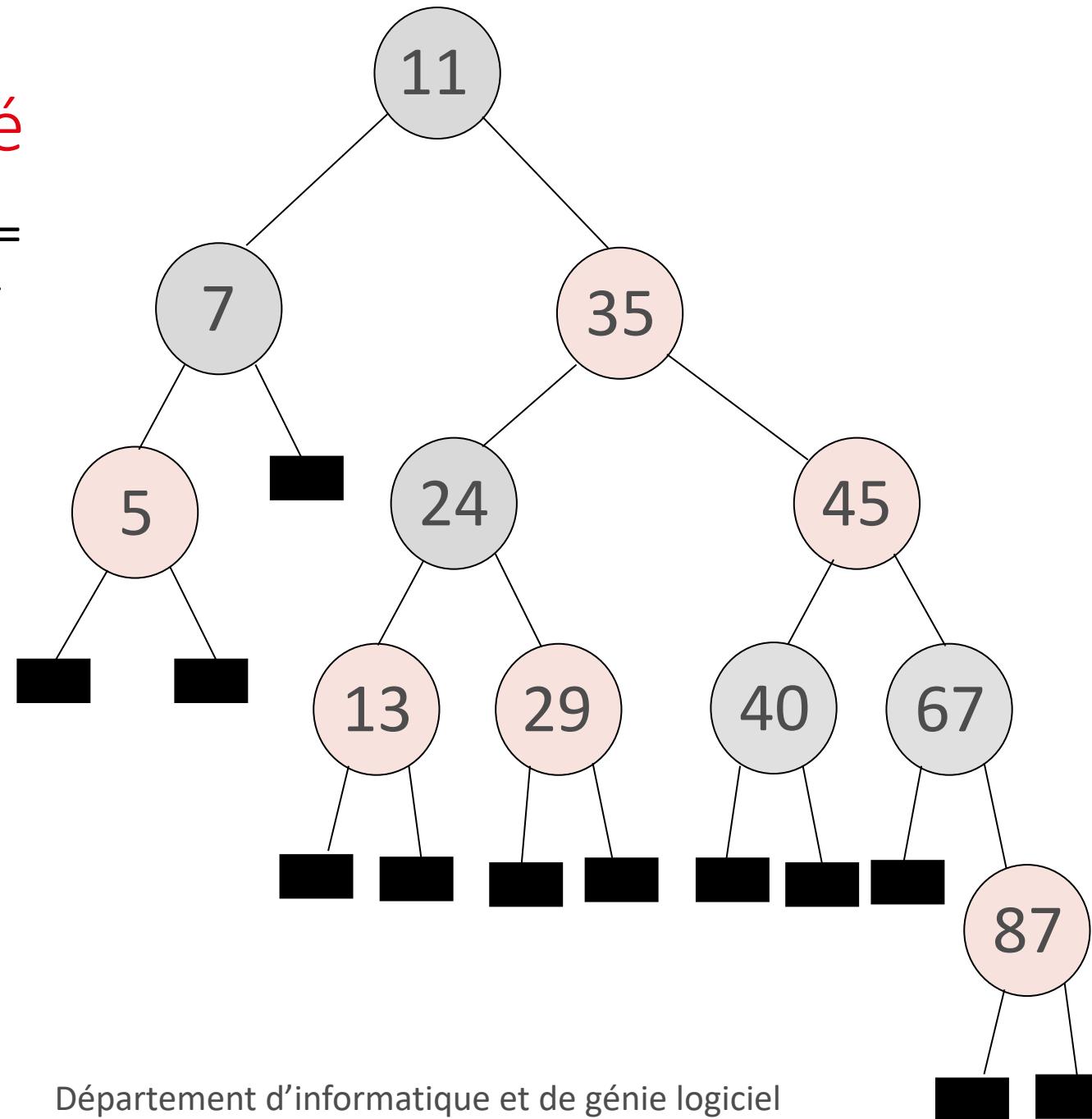
Exemple détaillé

- Tant que
(nœud → parent →
couleur
== Rouge)



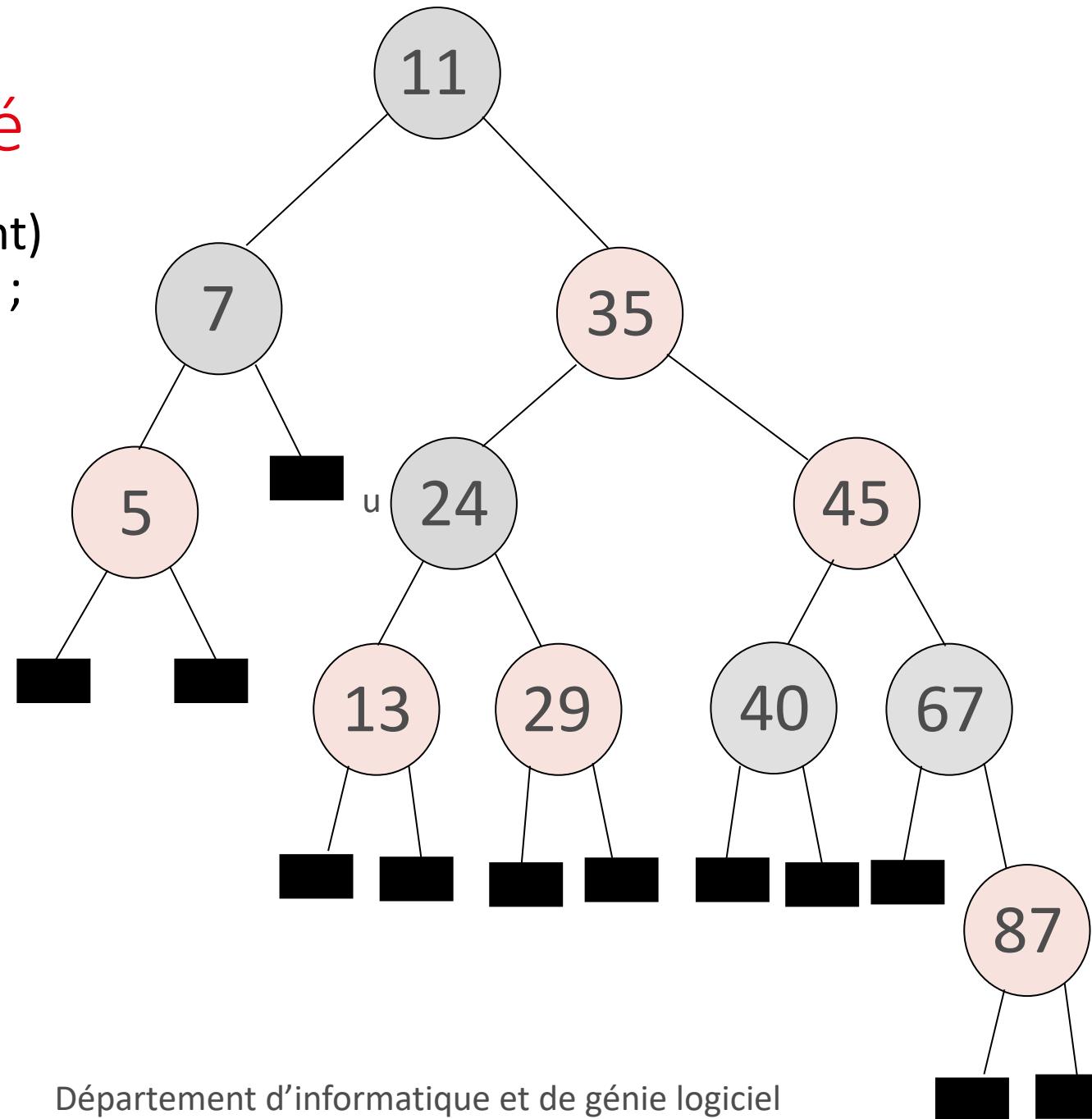
Exemple détaillé

- Si (nœud → parent == [(nœud → parent) → parent]→ droit)



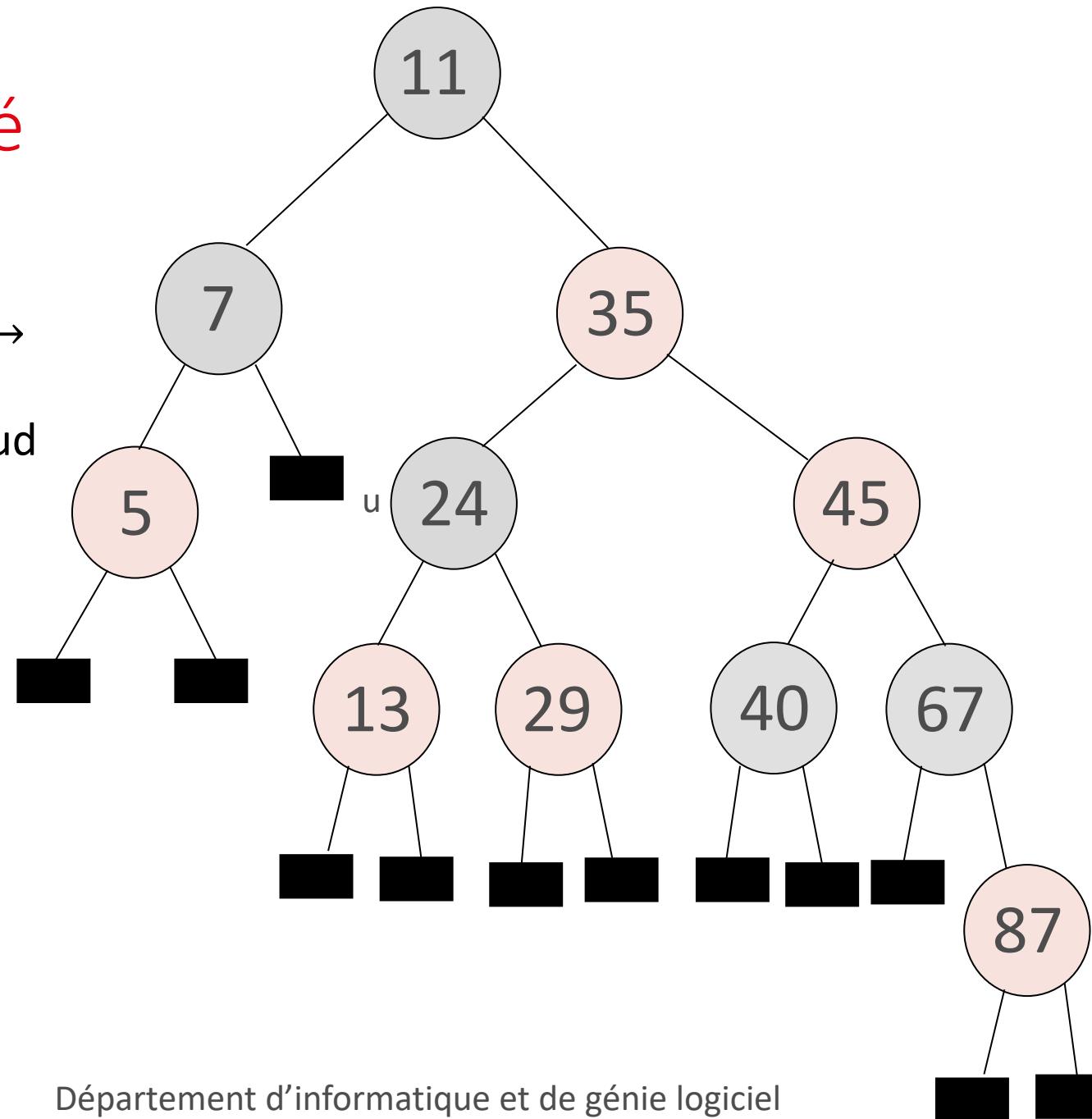
Exemple détaillé

- $u \leftarrow [(\text{nœud} \rightarrow \text{parent}) \rightarrow \text{parent}] \rightarrow \text{gauche} ;$



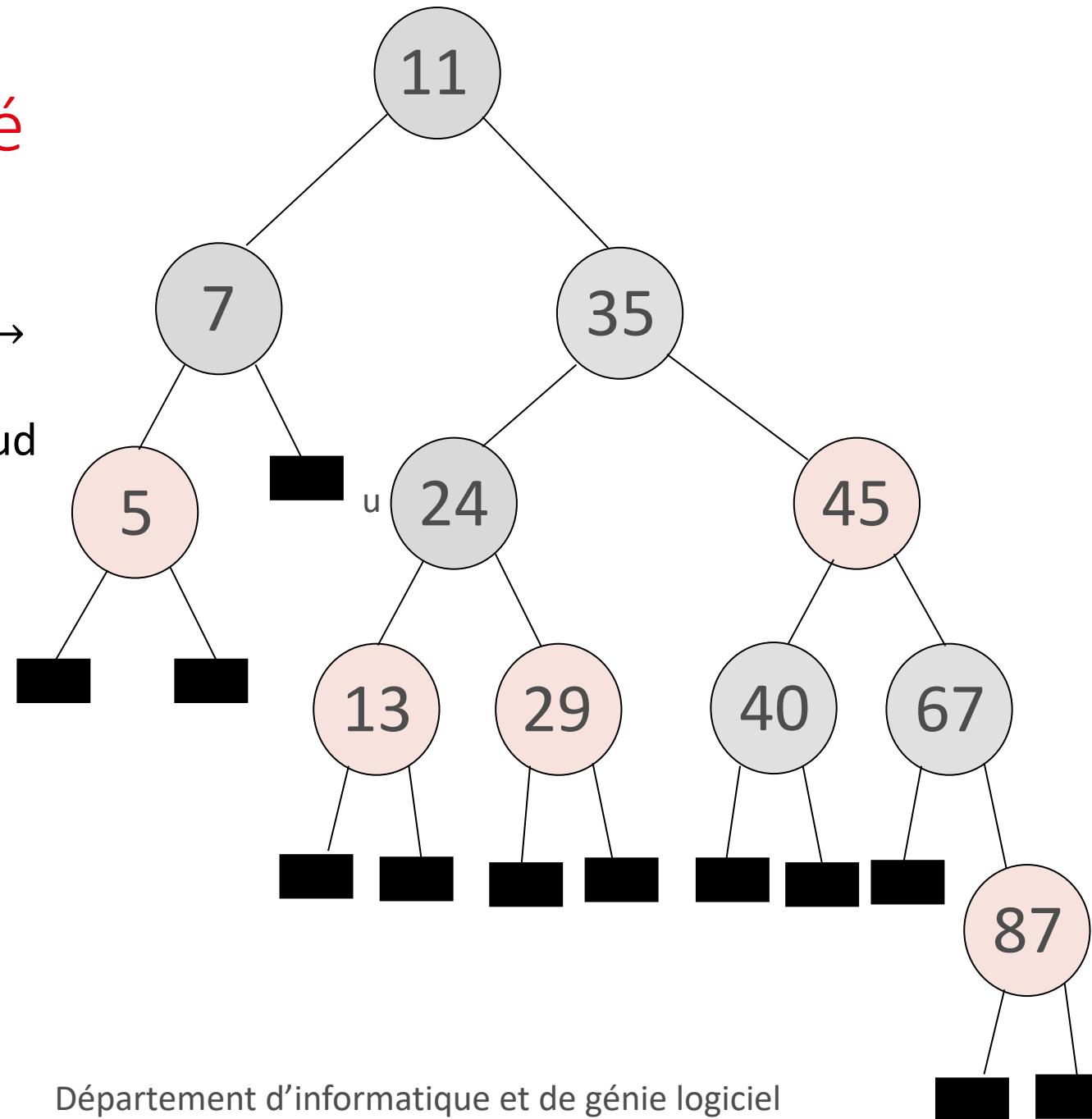
Exemple détaillé

- Si (**nœud == nœud → parent → gauche**)
 - **nœud ← nœud → parent;**
 - **ZigZigGauche(nœud);**
- **nœud → parent → couleur ↔ Noir;**
- **nœud → parent → parent → couleur ← Rouge;**
- **ZigZigDroit(nœud → parent → parent)**



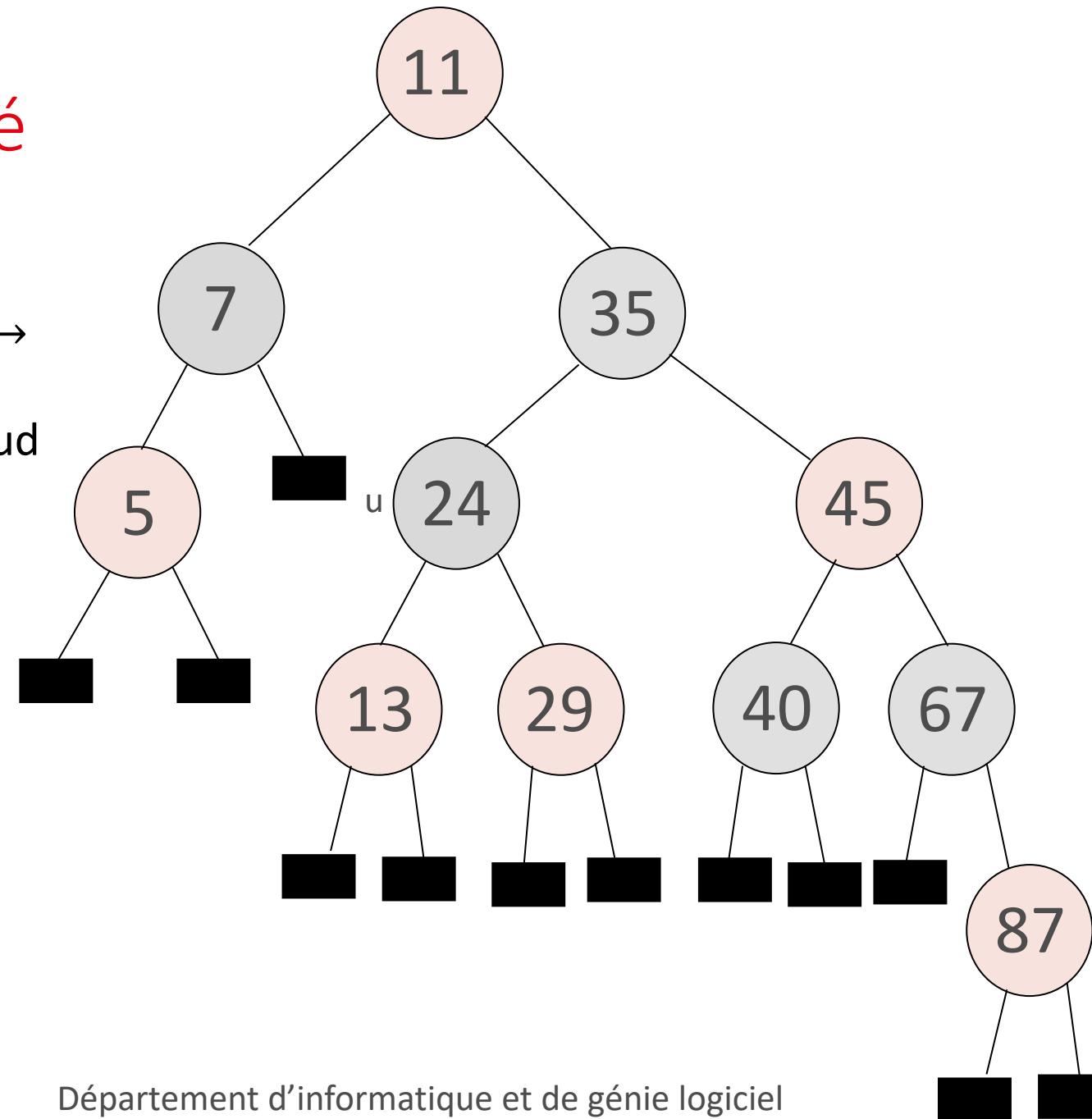
Exemple détaillé

- Si ($nœud == nœud \rightarrow$
 $parent \rightarrow \text{gauche}$)
 - $nœud \leftarrow nœud \rightarrow$
 $parent;$
 - ZigZigGauche($nœud$)
- $nœud \rightarrow parent \rightarrow$
 $\text{couleur} \leftarrow \text{Noir};$
- $nœud \rightarrow parent \rightarrow$
 $parent \rightarrow \text{couleur} \leftarrow$
 $\text{Rouge};$
- ZigZigDroit($nœud \rightarrow$
 $parent \rightarrow parent)$



Exemple détaillé

- Si ($nœud == nœud \rightarrow$
 $parent \rightarrow \text{gauche}$)
 - $nœud \leftarrow nœud \rightarrow$
 $parent;$
 - $\text{ZigZigGauche}(nœud)$;
- $nœud \rightarrow parent \rightarrow$
 $\text{couleur} \leftrightarrow \text{Noir};$
- **$nœud \rightarrow parent \rightarrow$**
 $parent \rightarrow couleur \leftarrow$
Rouge;
- $\text{ZigZigDroit}(nœud \rightarrow$
 $parent \rightarrow parent)$



Exemple détaillé

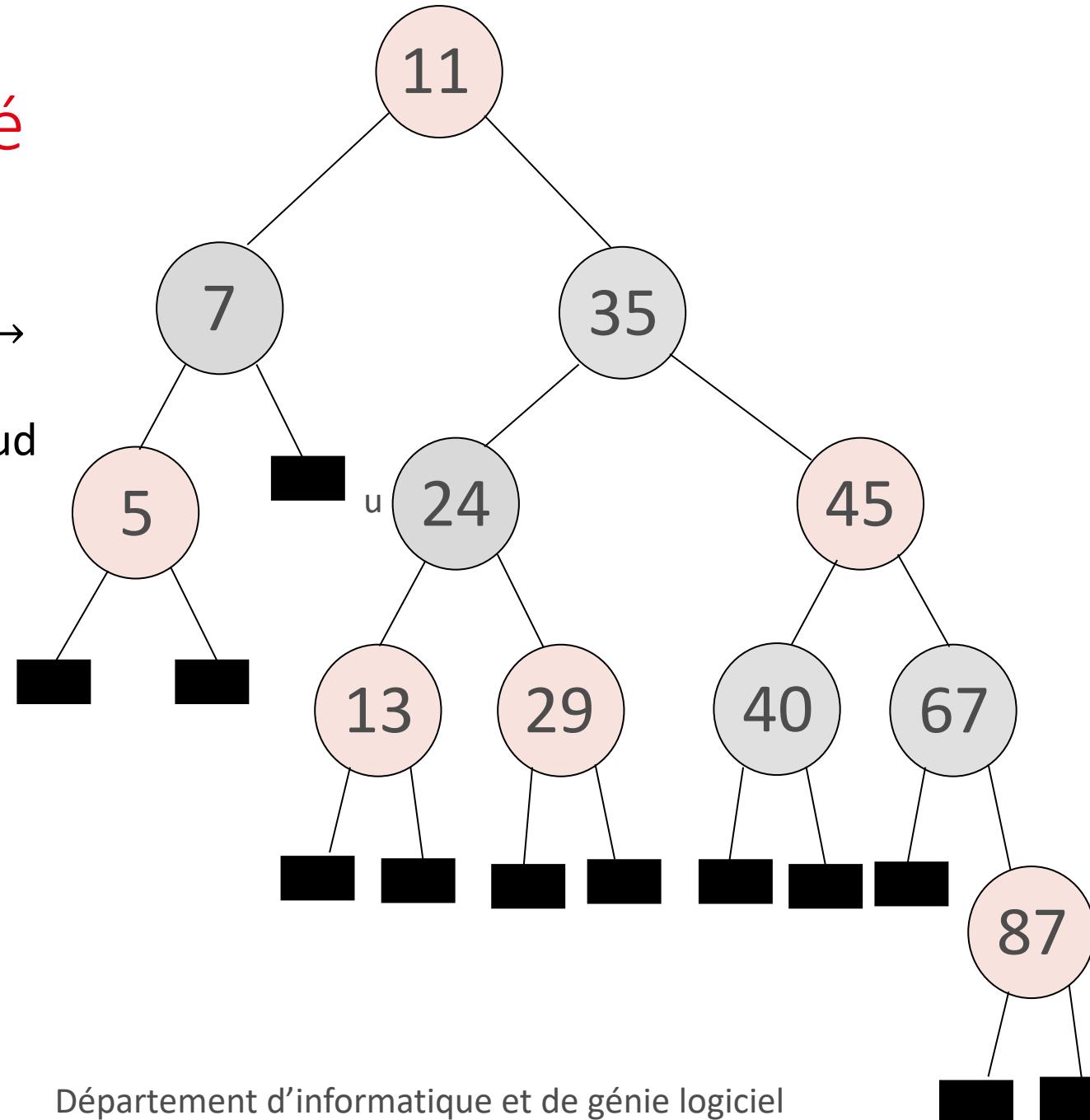
- Si ($nœud == nœud \rightarrow$
 $parent \rightarrow \text{gauche}$)

➤ $nœud \leftarrow nœud \rightarrow$
 $parent;$
➤ **ZigZigGauche($nœud$)**;

- $nœud \rightarrow parent \rightarrow$
 $\text{couleur} \leftrightarrow \text{Noir};$

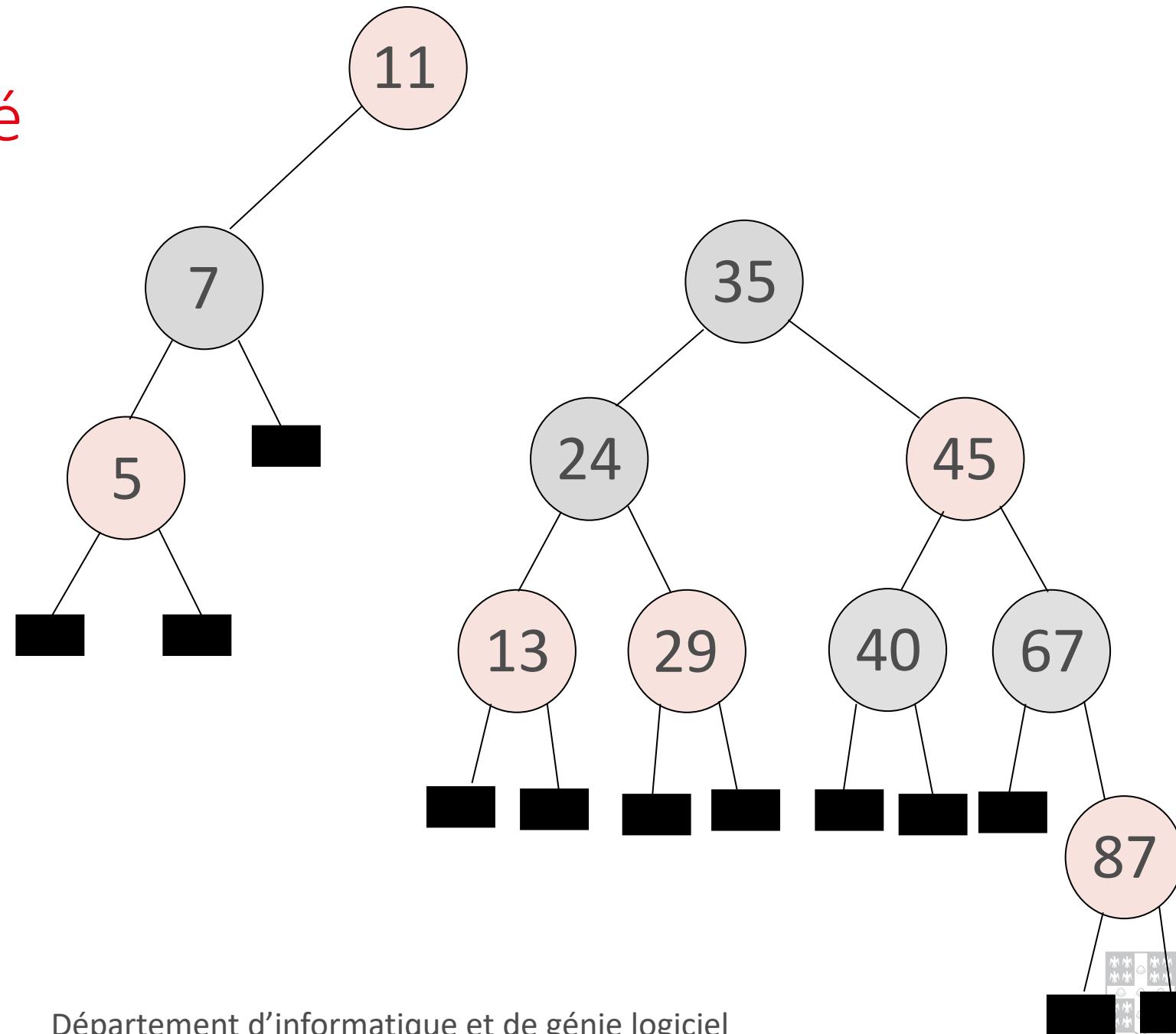
- $nœud \rightarrow parent \rightarrow$
 $parent \rightarrow \text{couleur} \leftarrow$
 $\text{Rouge};$

- **ZigZigDroit($nœud \rightarrow$
 $parent \rightarrow parent$)**



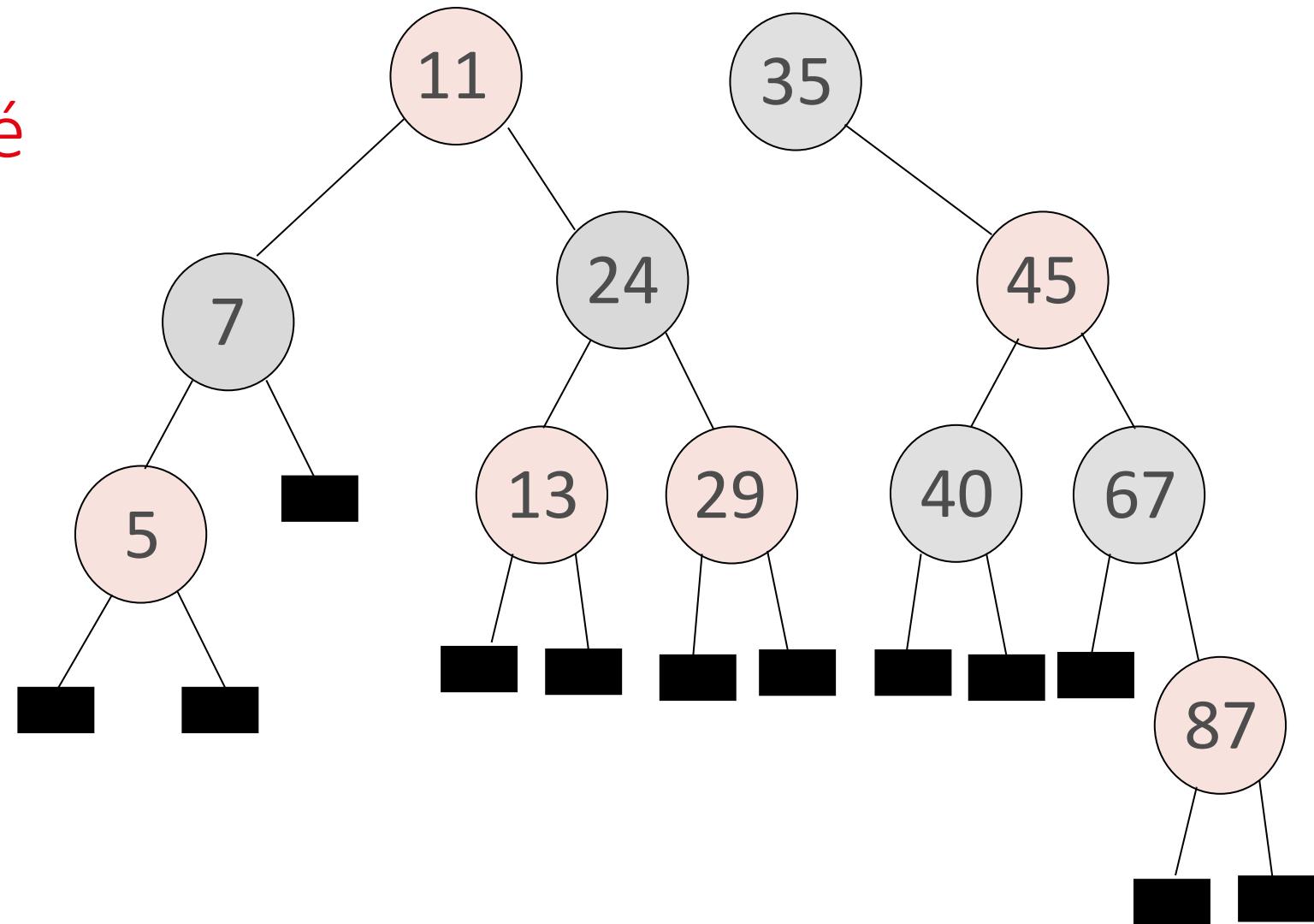
Exemple détaillé

- **ZigZigDroit(nœud → parent → parent)**



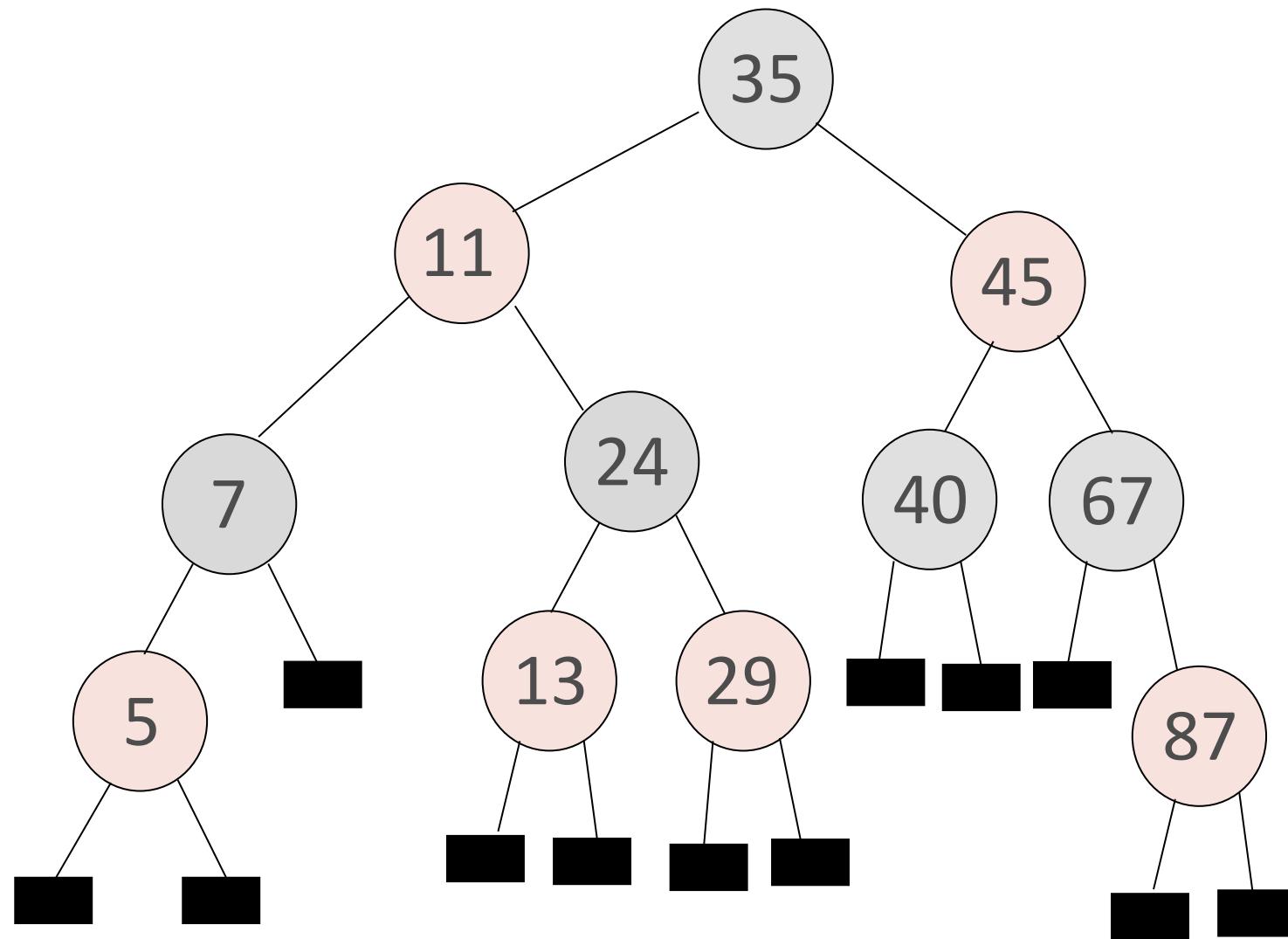
Exemple détaillé

- **ZigZigDroit(nœud → parent → parent)**

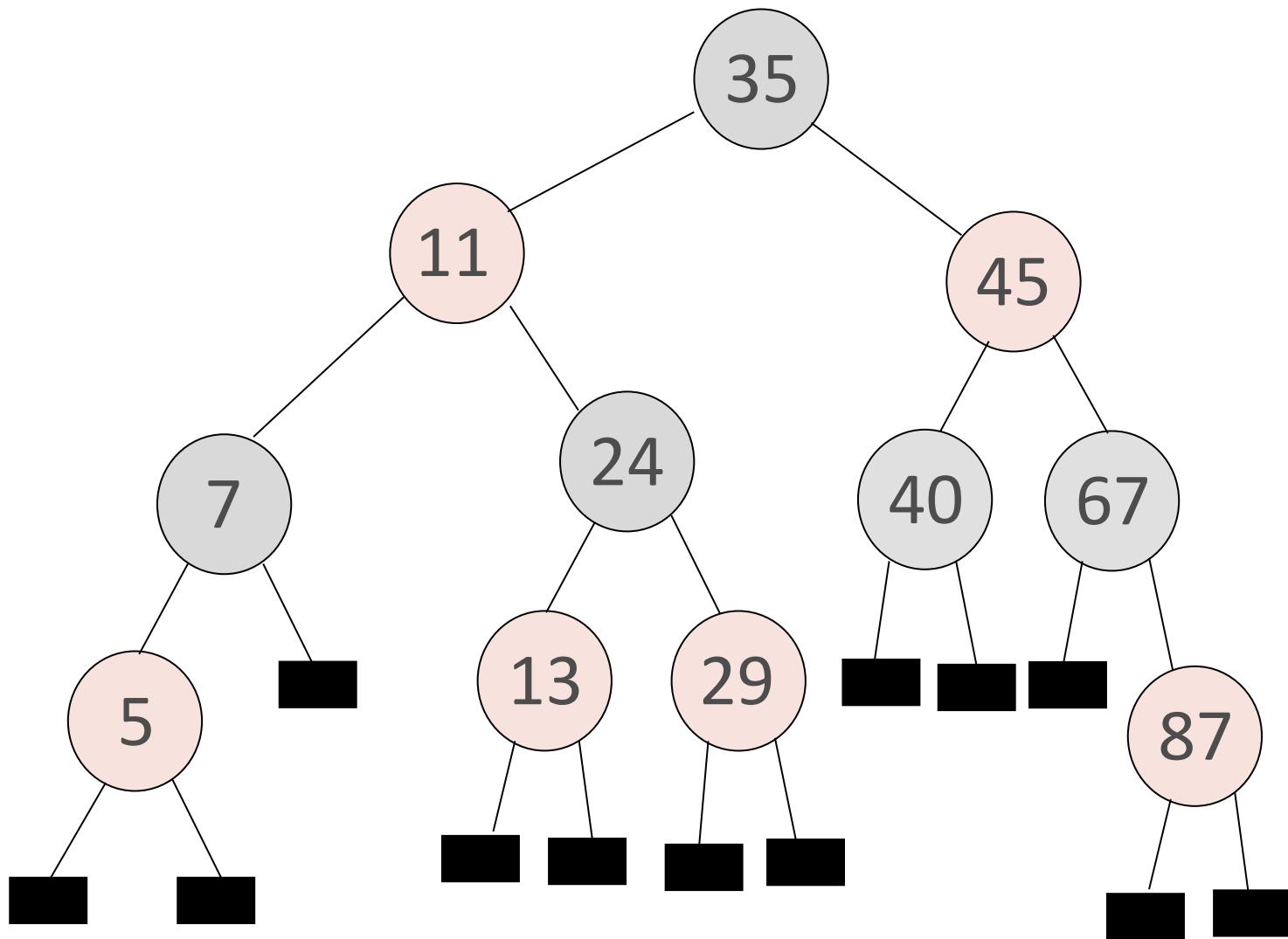


Exemple détaillé

- **ZigZigDroit(nœud → parent → parent)**



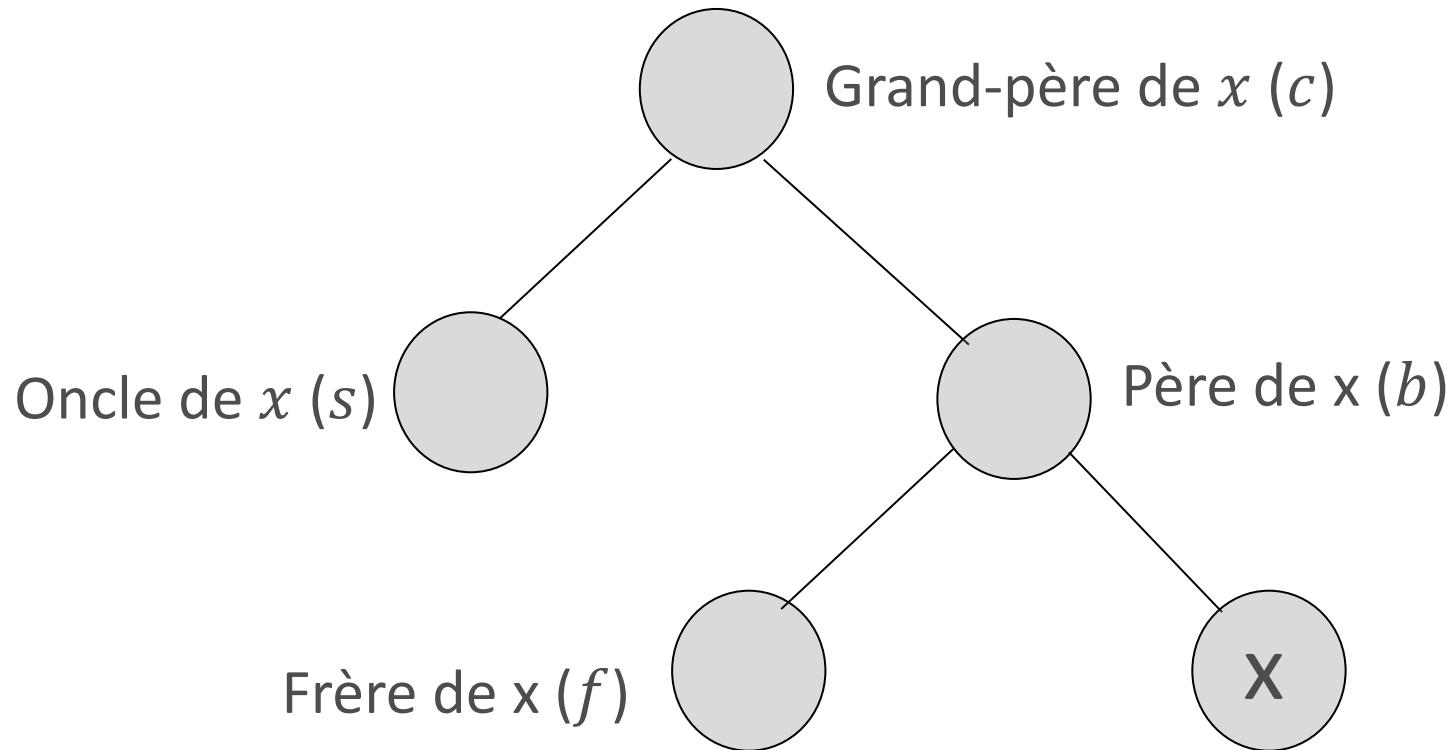
Exemple détaillé



Suppressions

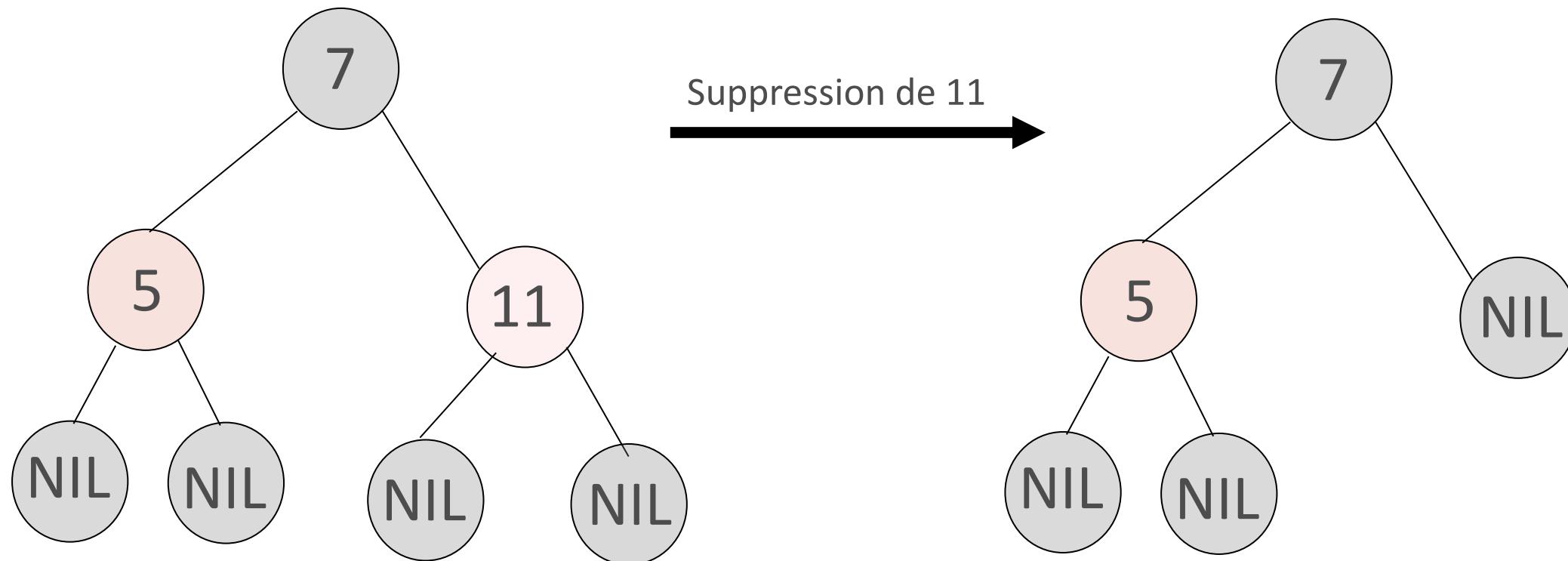
- Supprimer le noeud à la bonne position comme dans un arbre AVL.
- Ensuite, on doit gérer le coloriage et le balancement.
- On verra 3 cas pour la suppression et 6 pour fixer le coloriage.

Supression – Terminologie



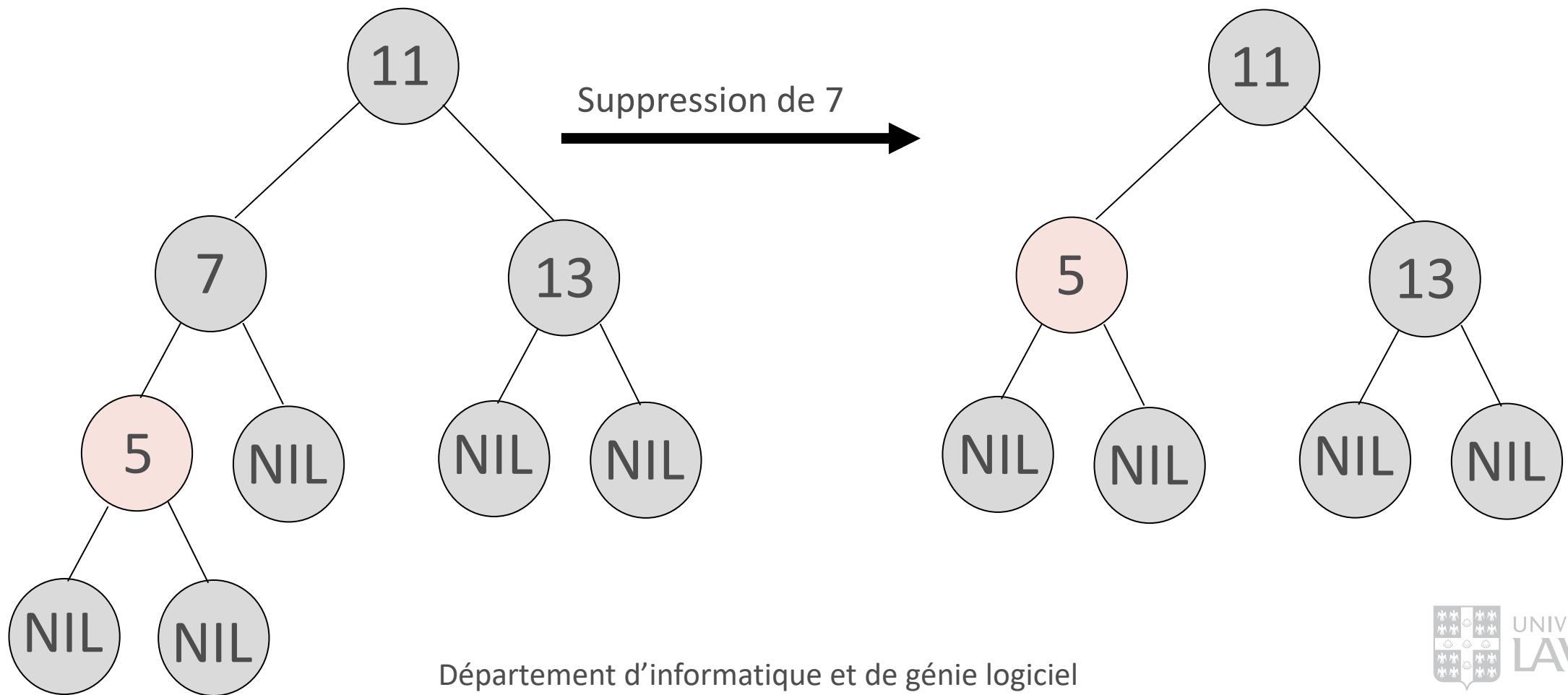
Suppression – Cas 1 : Le noeud à supprimer a deux feuilles comme enfant

- On supprime le noeud et on remplace ce noeud par une feuille.



Suppression – Cas 2 : Le noeud à supprimer a une feuille et un noeud interne

- On remplace le noeud à supprimer par son seul enfant.

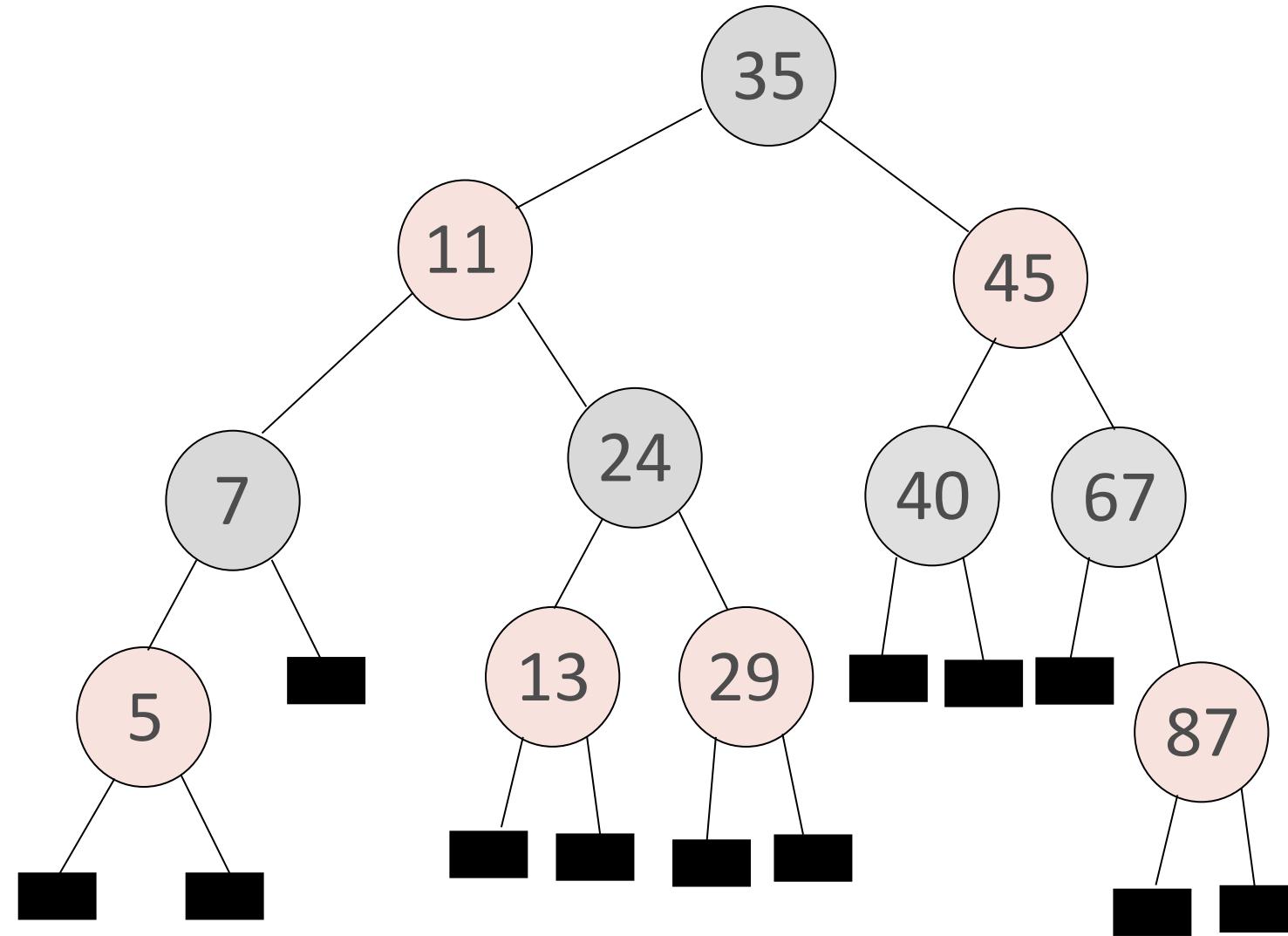


Suppression – Cas 3 : Le noeud à supprimer a deux noeuds internes

- Comme pour les arbres AVL, il faut donc d'abord retrouver son successeur minimal à droite à l'aide d'une boucle simple (une fois à droite, plein de fois à gauche).
- Ensuite, on doit les échanger et on supprime le noeud.

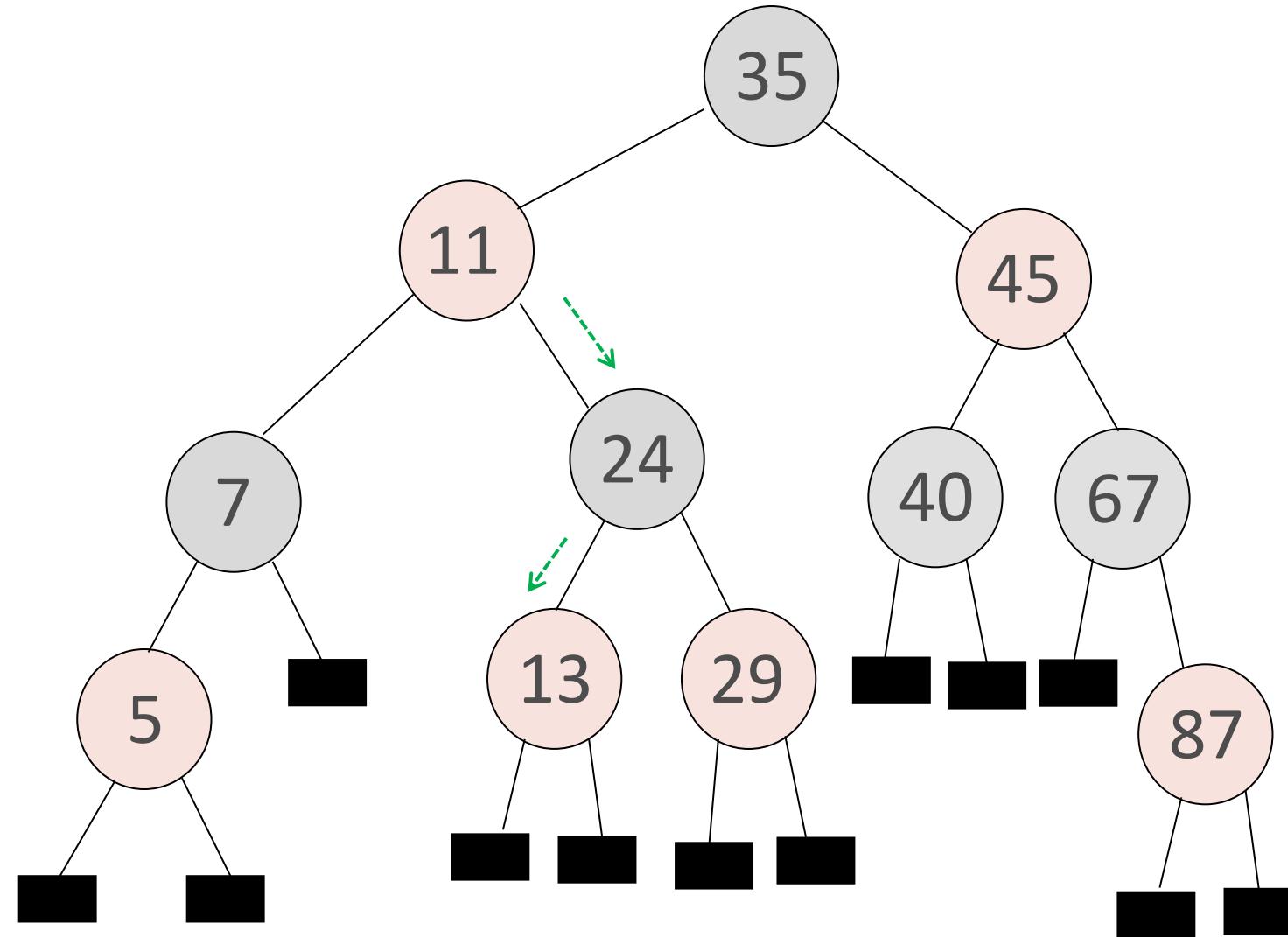
Suppression – Cas 3 : Le noeud à supprimer a deux noeuds internes

- Supprimer 11



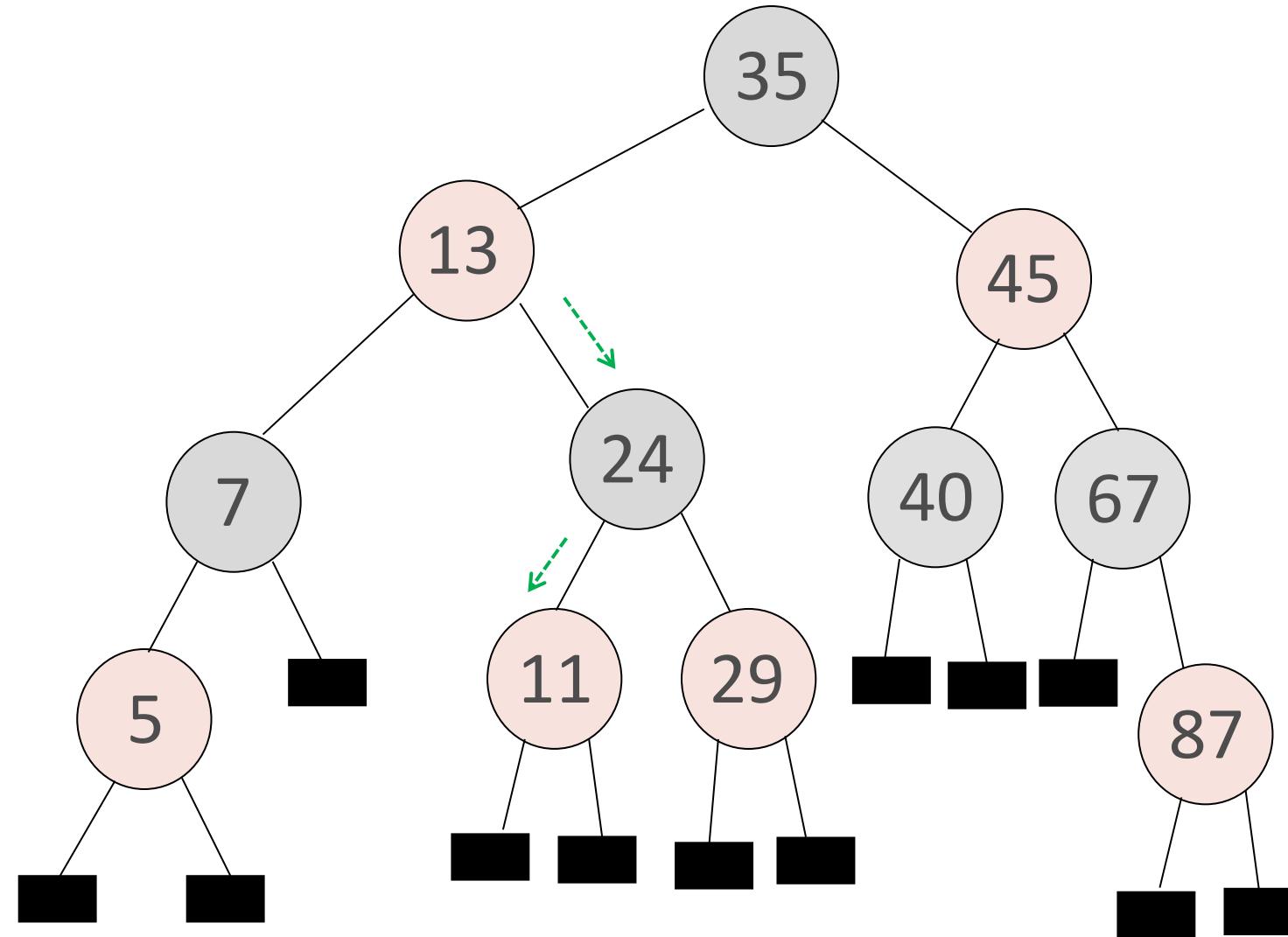
Suppression – Cas 3 : Le noeud à supprimer a deux noeuds internes

- Supprimer 11



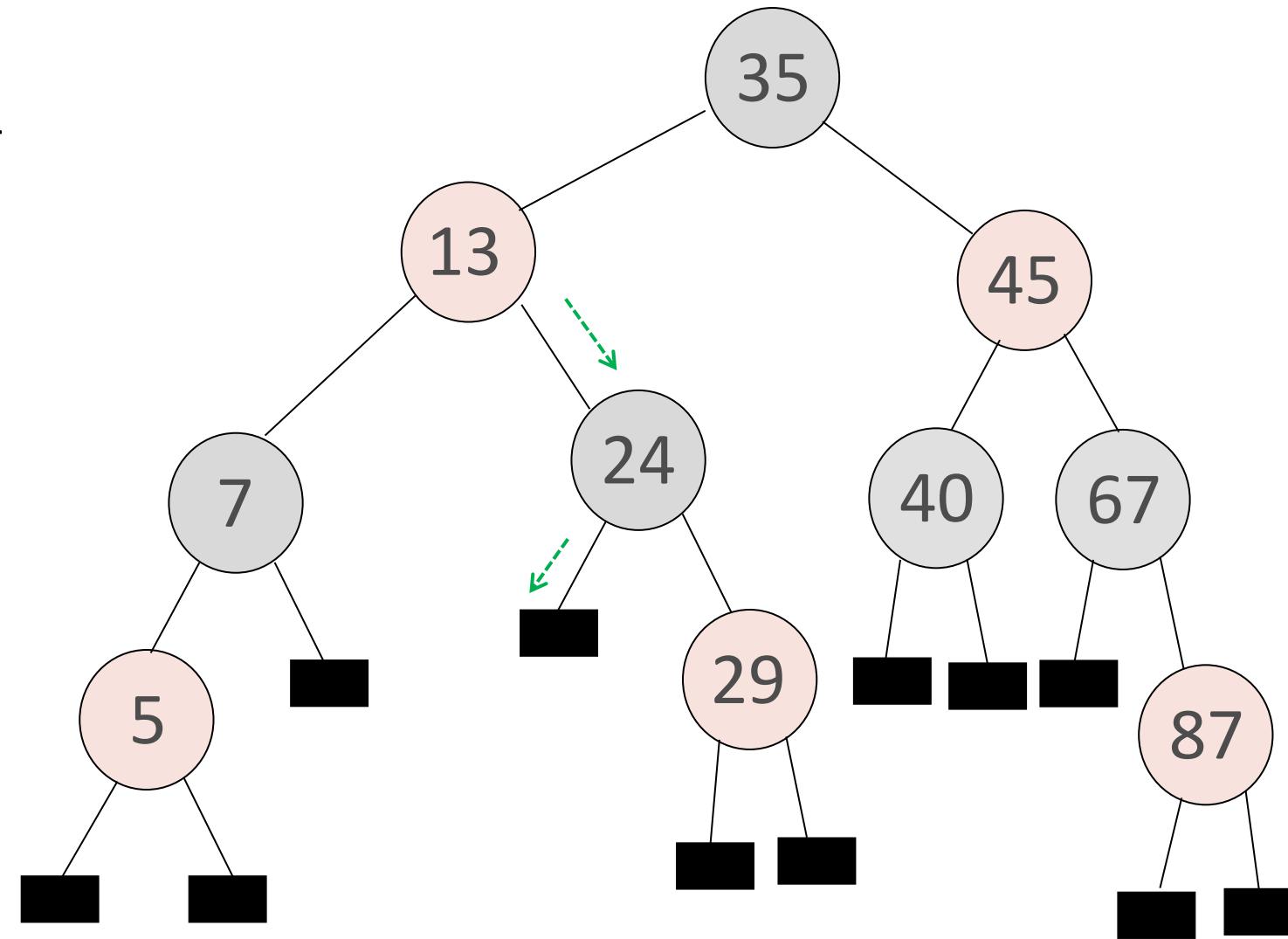
Suppression – Cas 3 : Le noeud à supprimer a deux noeuds internes

- Supprimer 11



Suppression – Cas 3 : Le noeud à supprimer a deux noeuds internes

- Supprimer 11

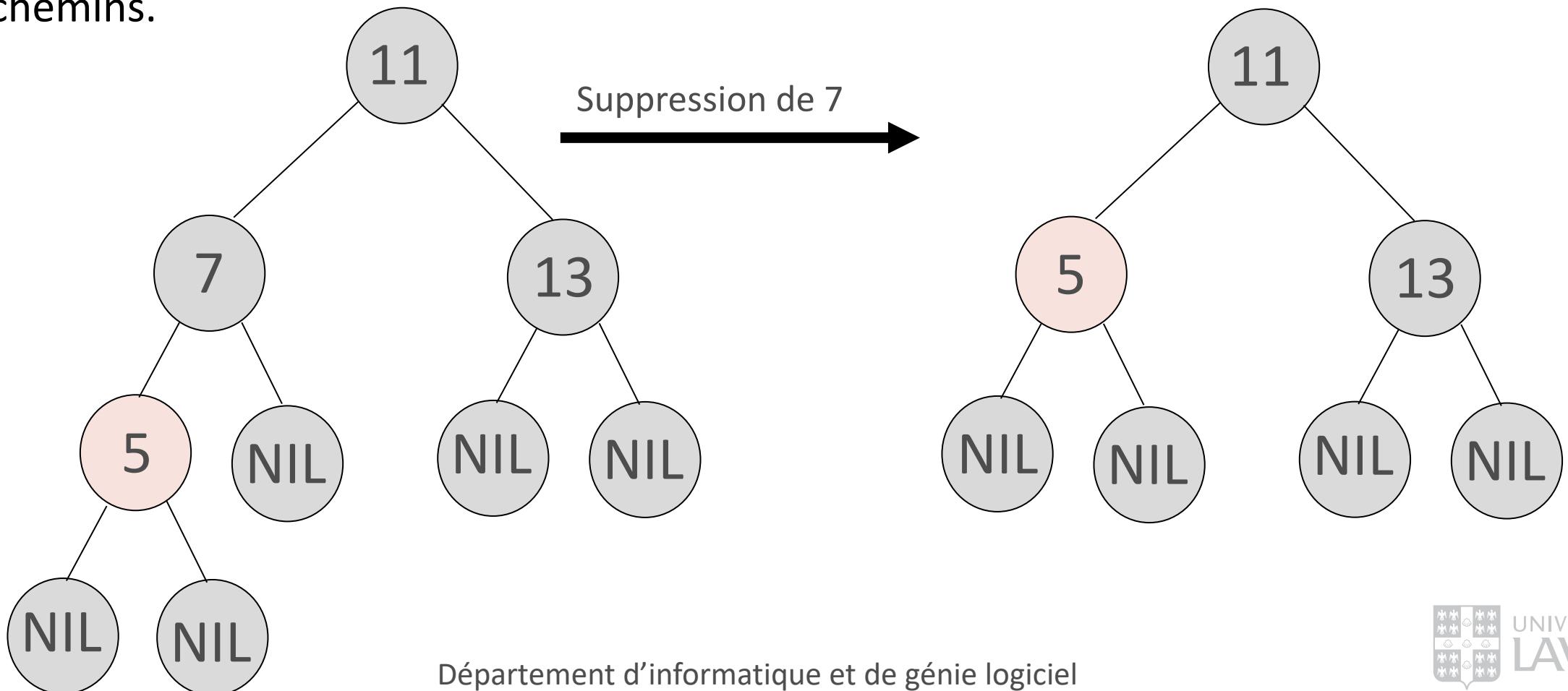


Supression – Fixer les propriétés

- Lorsqu'on fait ce changement et qu'on supprimer un noeud, il se peut que certaines propriétés de l'arbre rouge-noire soit violées.
- Il faut faire des changements pour respecter ces propriétés.
- On verra plusieurs cas.
- Noter que dans tous les cas, le noeud supprimé aura 0 ou 1 enfant, puisque dans le cas où il y avait deux enfants, nous avons fait un changement de noeud.

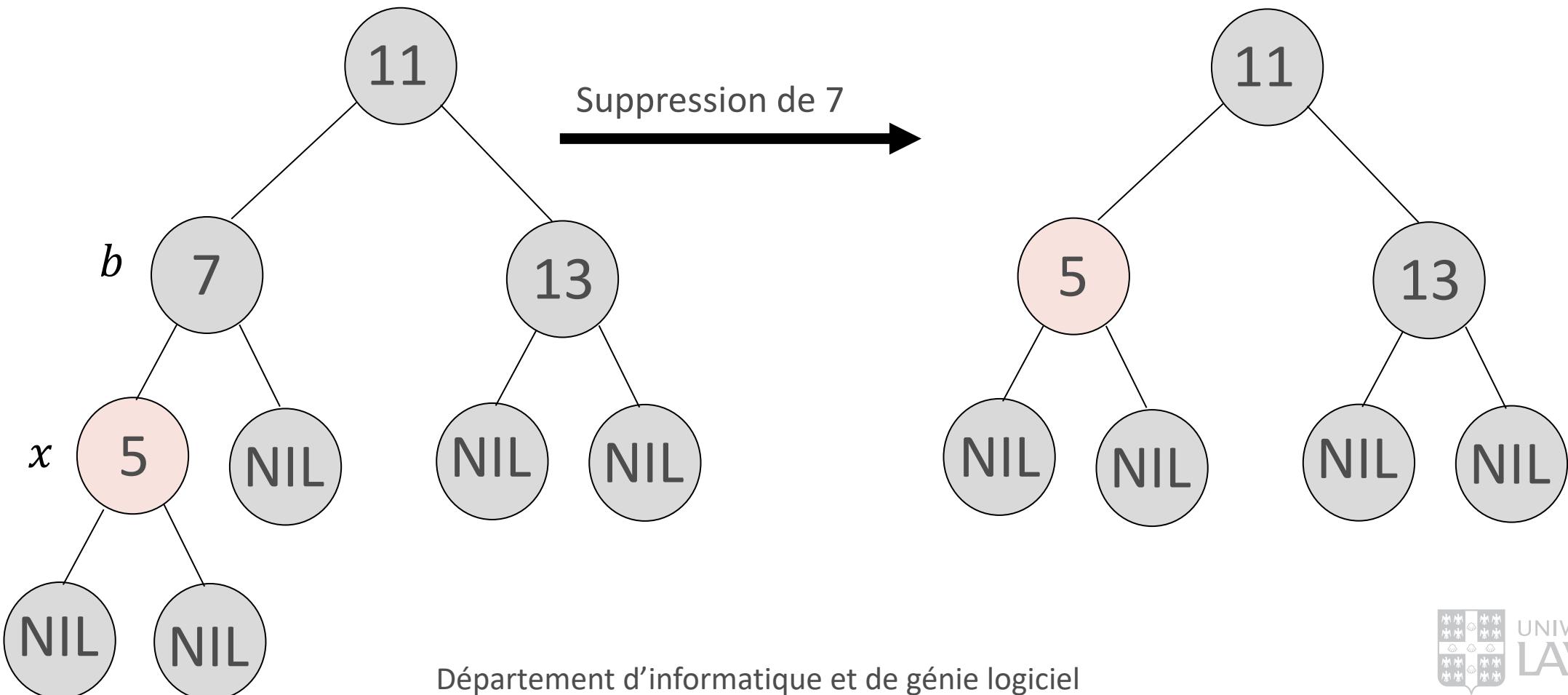
Supression – Fixer les propriétés

- Par exemple, ici, la suppression de 7 change le nombre de noeuds noirs sur des chemins.



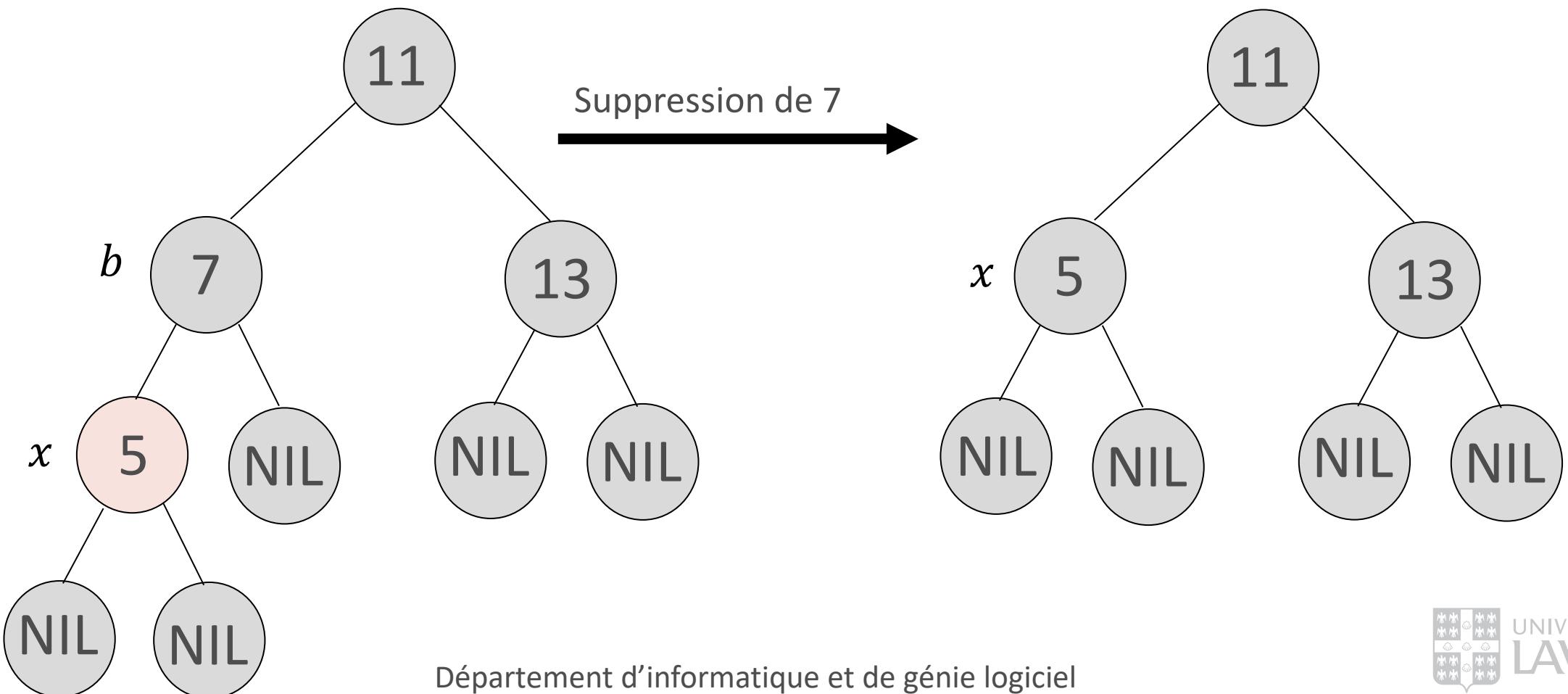
Supression – Fixer les propriétés : Cas 1

- Cas 1 : x ou b est rouge (comme on ne peut pas avoir deux noeuds consécutifs rouge, on au maximum un des deux noeuds rouges)



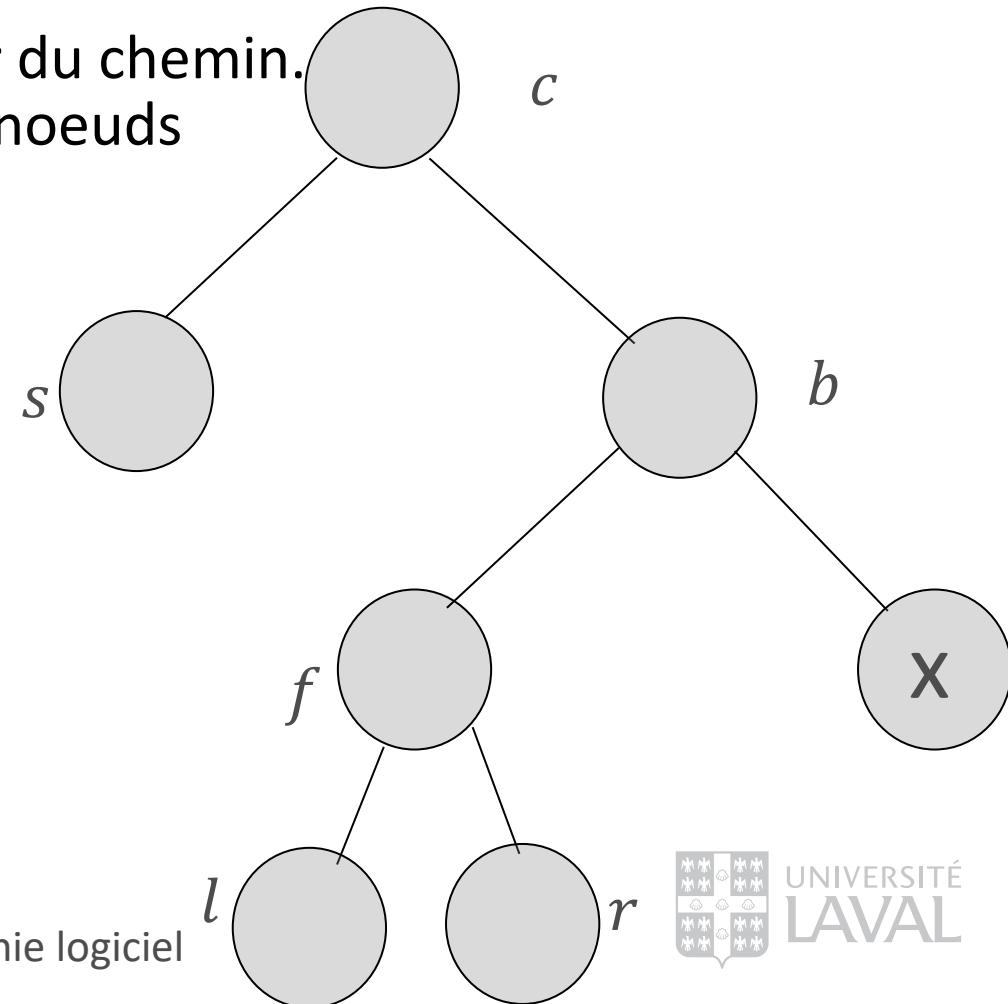
Supression – Fixer les propriétés : Cas 1

- Cas 1 : Dans ce cas, on remplace la couleur de l'enfant par noir. Ici, le noeud contenant 5 devient donc noir.



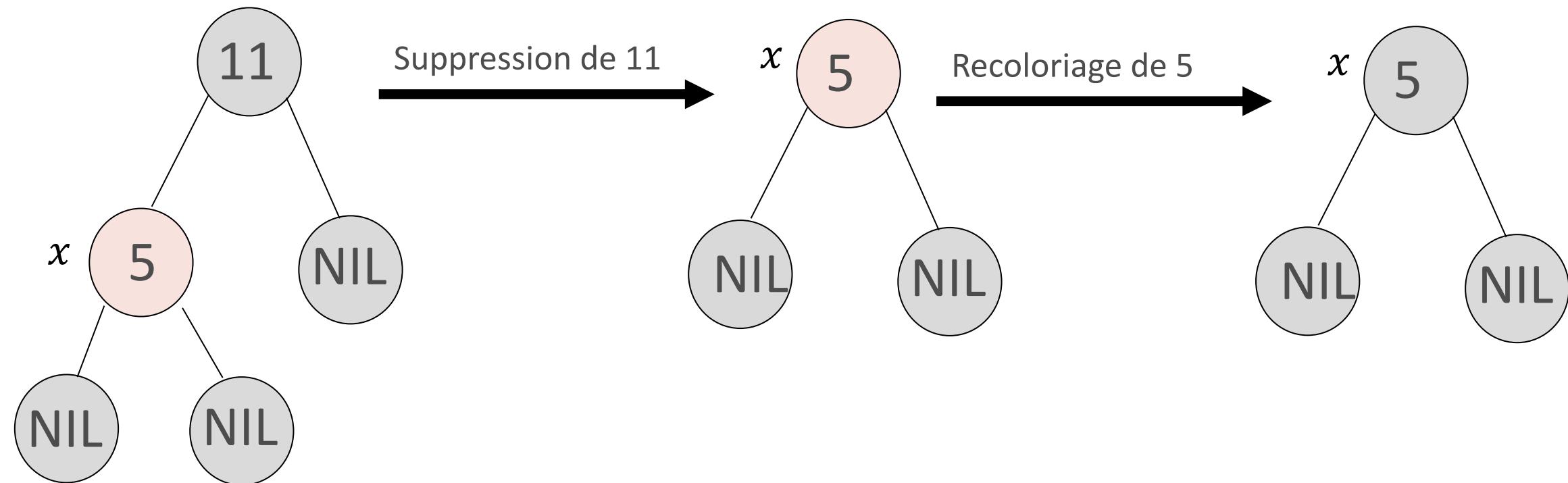
Supression – Fixer les propriétés : Cas 2

- Les deux noeuds sont noirs.
- Dans ce cas, retirer un noeud enlève un noeud noir du chemin.
On doit donc fixer la propriété pour ramener deux noeuds noirs sur chaque chemin.
- On verra plusieurs sous-cas :
 - Si x est la racine
 - Si f est noir et ses enfants sont rouges.
 - Si f est noir un de ses enfants est rouge.
 - Si f est noir et ses enfants sont noirs.
 - Si f est rouge



Supression – Fixer les propriétés : Cas 2a, x est la racine

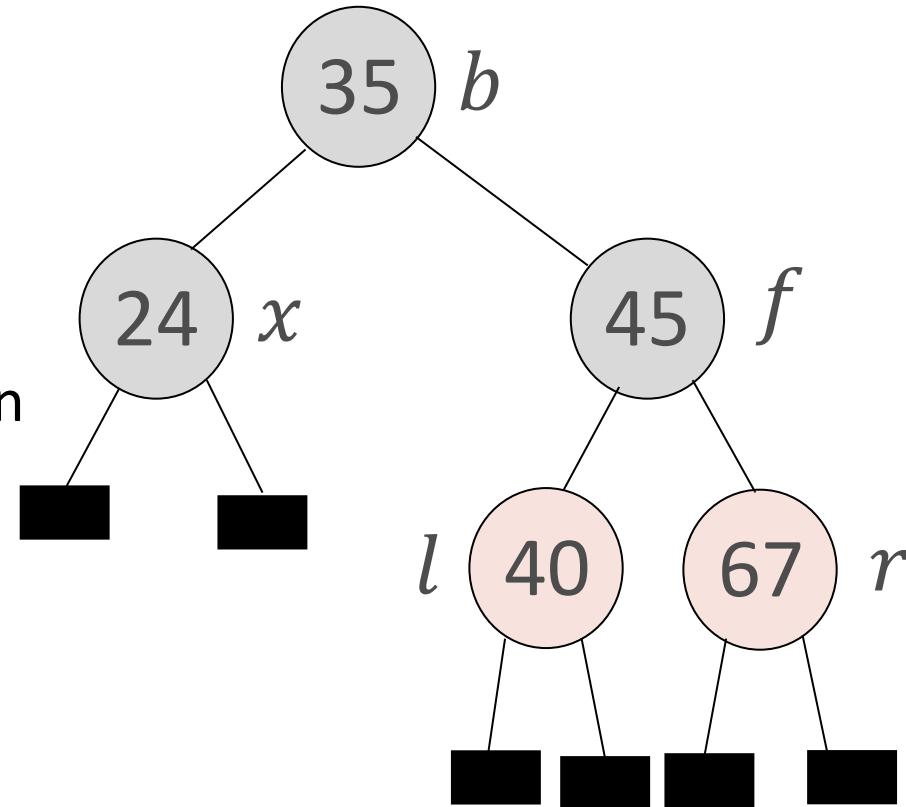
- Si le noeud x est devenu la racine, on recolorie la racine en noir



Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution

- Supprimer 24
- Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un **ZigZigDroit** sur r .
- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution :

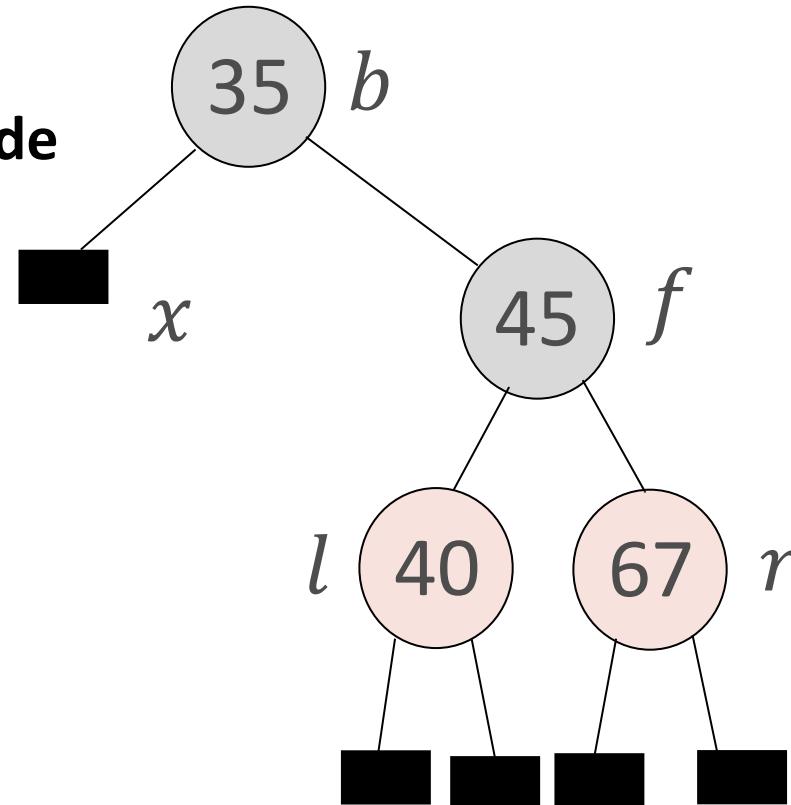
- Supprimer 24 (on a un noeud noir de moins sur le chemin vers x).

- Rebalance #1

- ✓ f est à droite de b

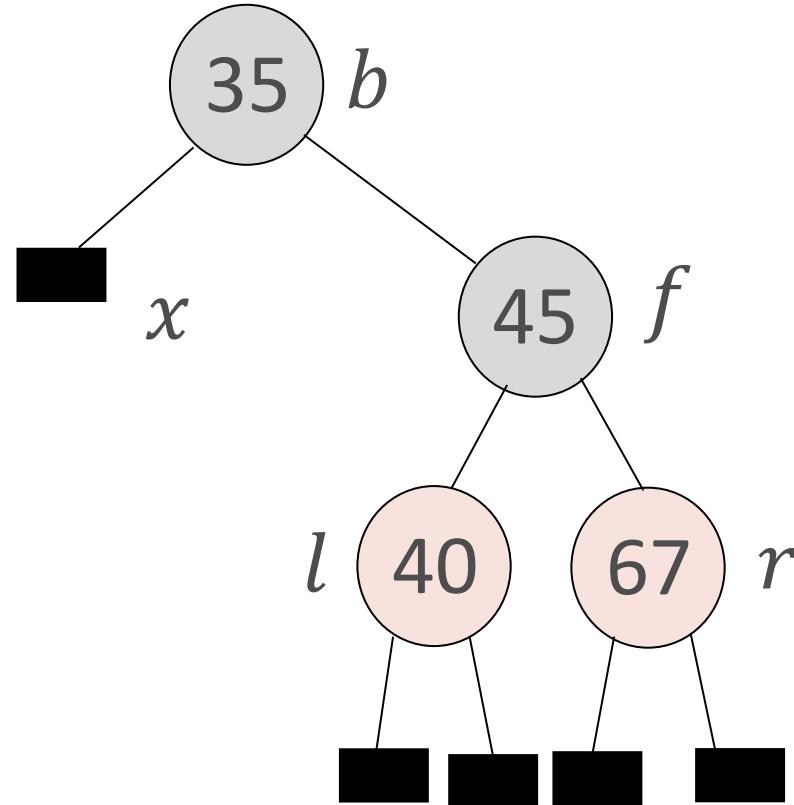
- ✓ On rebalance en faisant un **ZigZigDroit** sur r .

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

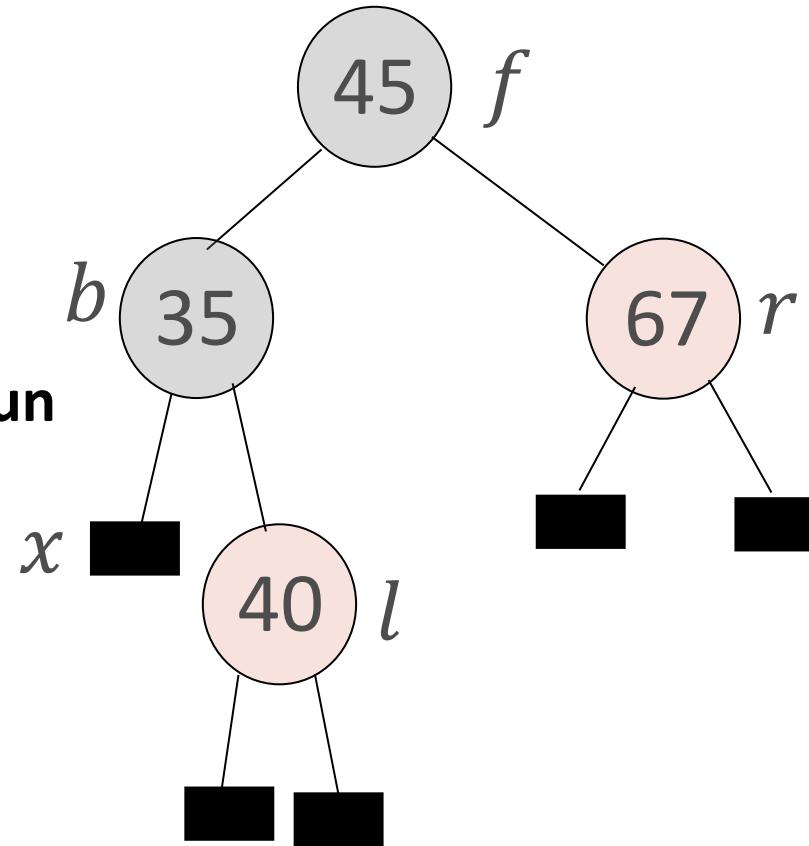
- Résolution :
 - Supprimer 24
 - **Rebalance #1**
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un **ZigZigDroit** sur r .
 - Colorier r en noir



Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution :

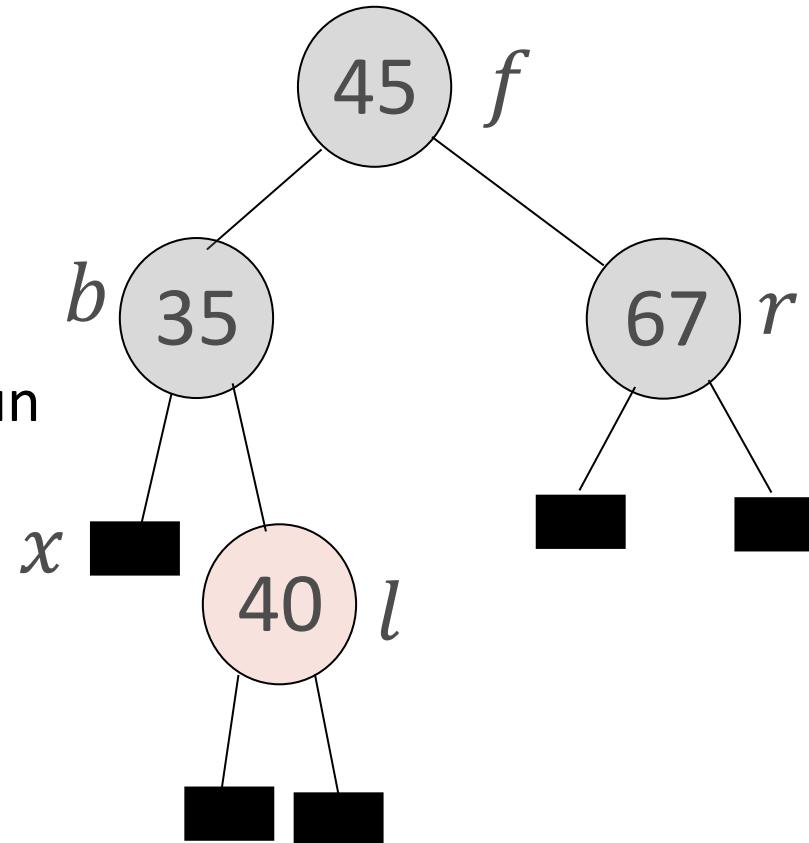
- Supprimer 24
- **Rebalance #1**
 - ✓ f est à droite de b
 - ✓ **On rebalance en faisant un ZigZigDroit sur r .**
- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

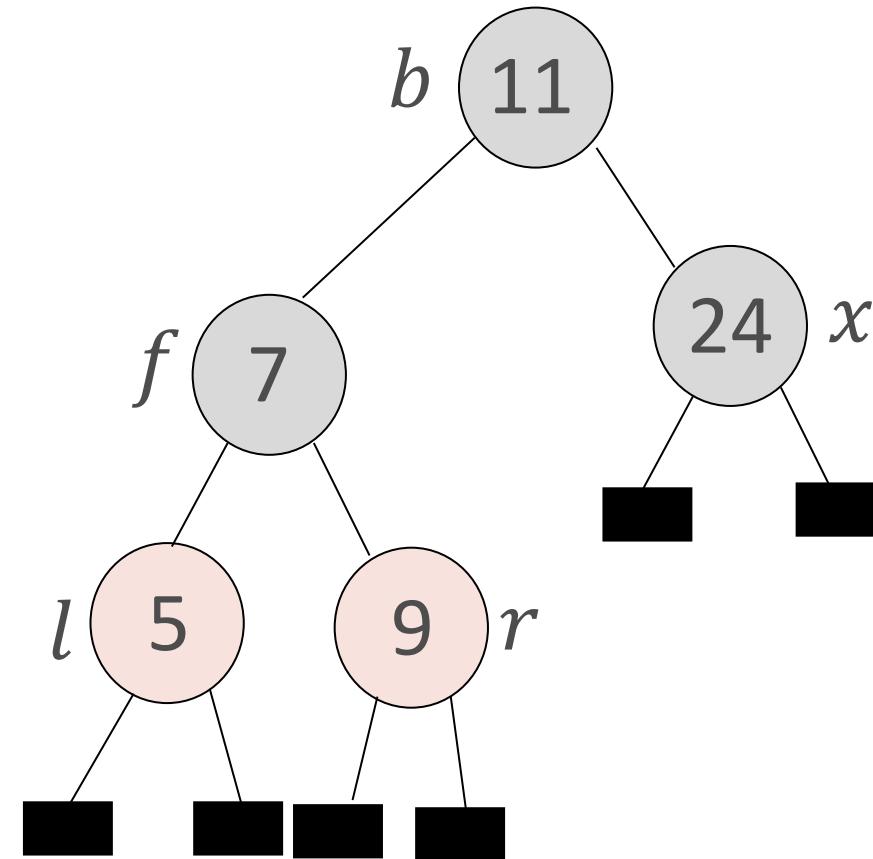
- Résolution :

- Supprimer 24
- Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un ZigZigDroit sur r .
- Colorier r en noir



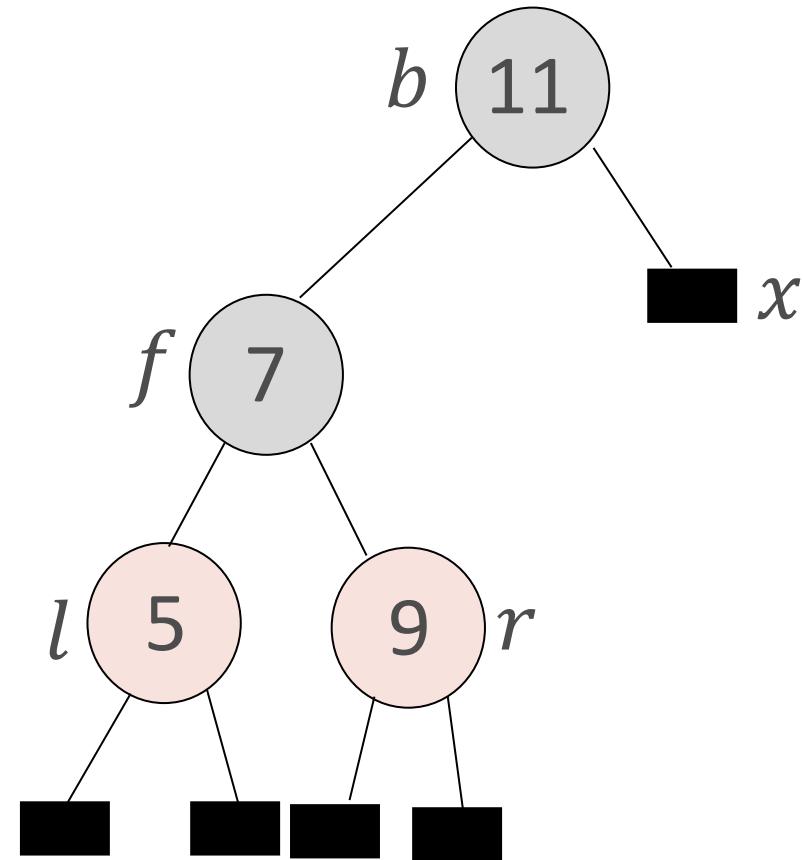
Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution
 - Supprimer 24
 - Rebalancement #1
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un **ZigZigGauche** sur l .
 - Colorier l en noir



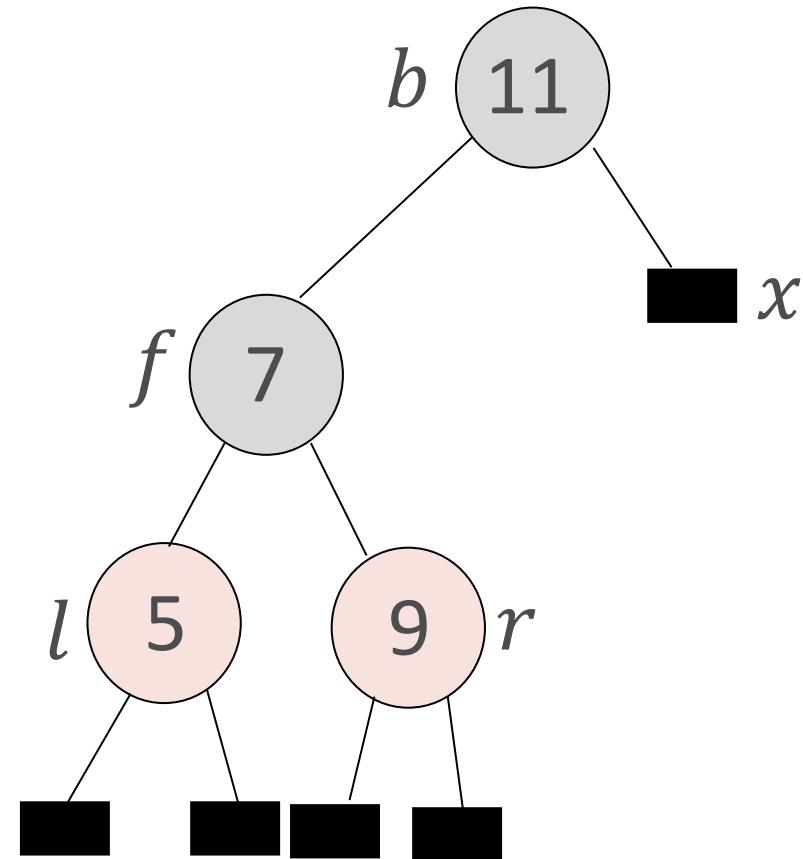
Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution
 - Supprimer 24
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un **ZigZigGauche** sur l .
 - Colorier l en noir



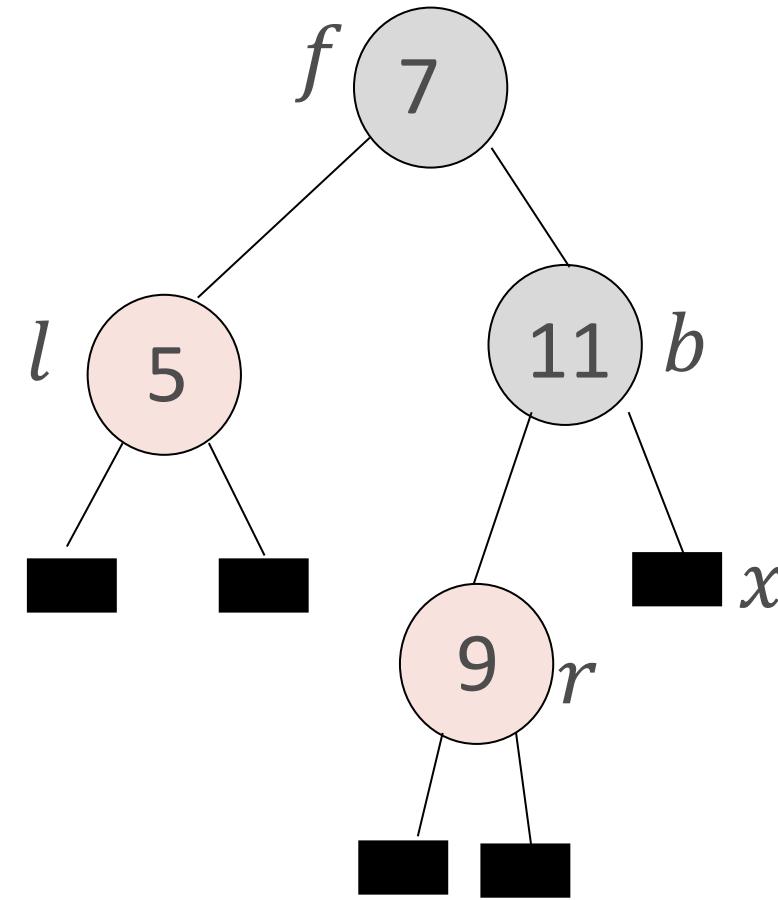
Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution
 - Supprimer 24
 - **Rebalance #2**
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un **ZigZigGauche** sur l .
 - Colorier l en noir



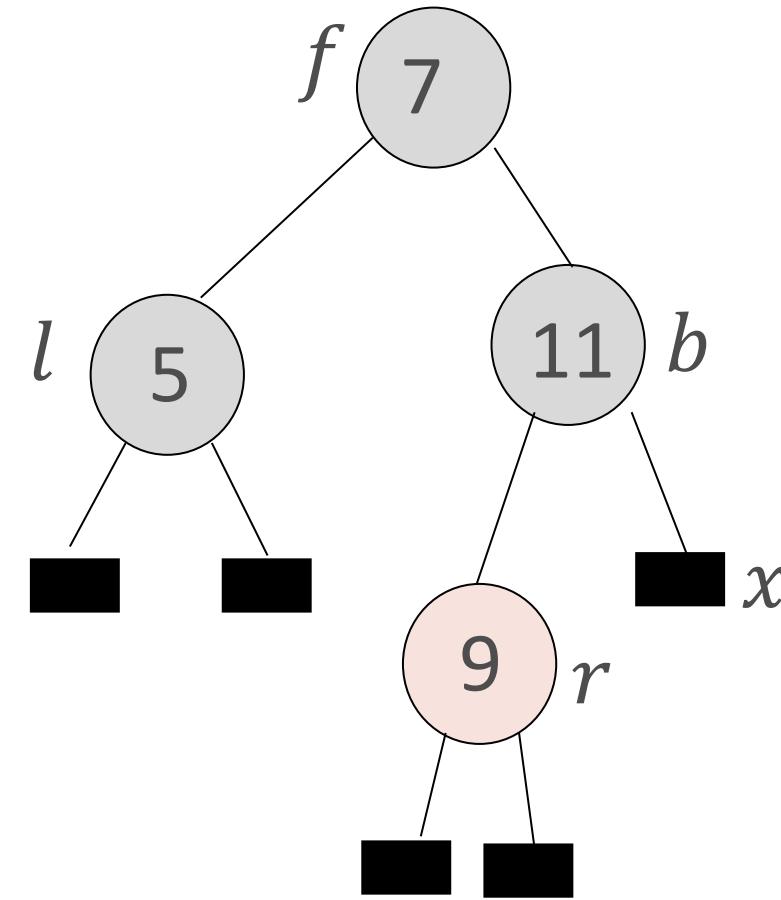
Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution
 - Supprimer 24
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ **On rebalance en faisant un ZigZigGauche sur l .**
 - Colorier l en noir



Suppression – Fixer les propriétés : Cas 2b, Si f est noir et ses enfants sont rouges.

- Résolution
 - Supprimer 24
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un **ZigZigGauche** sur l .
 - Colorier l en noir



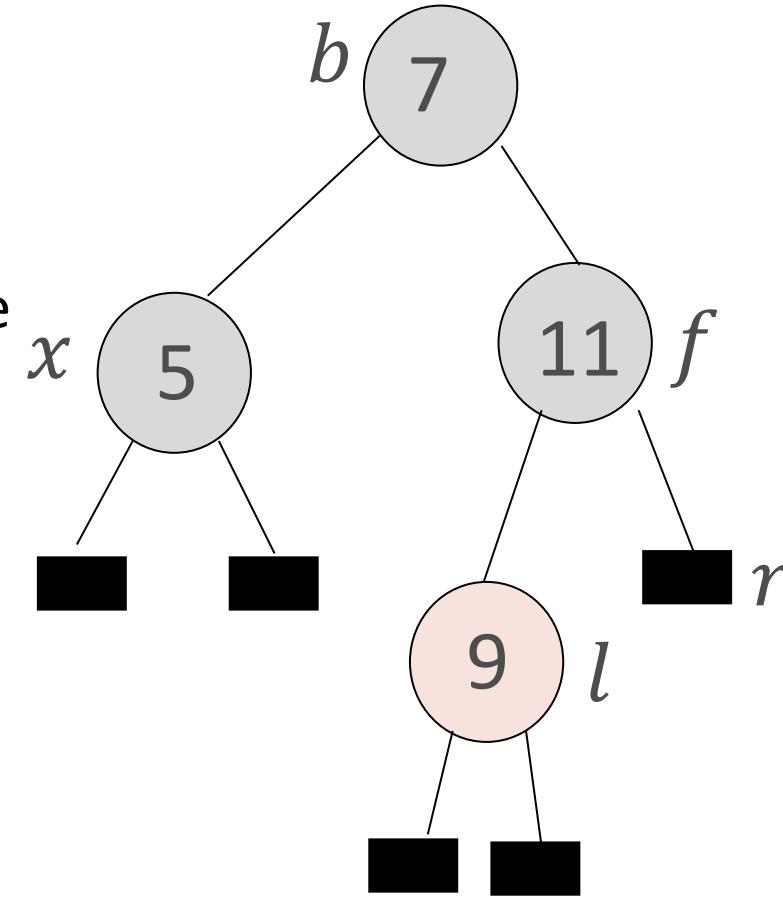
Supression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

- Supprimer 5
- Rebalancement #1
 - ✓ f est à droite de b et le noeud rouge est l .
 - ✓ On rebalance en faisant un **ZigZagDroit** sur b .

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

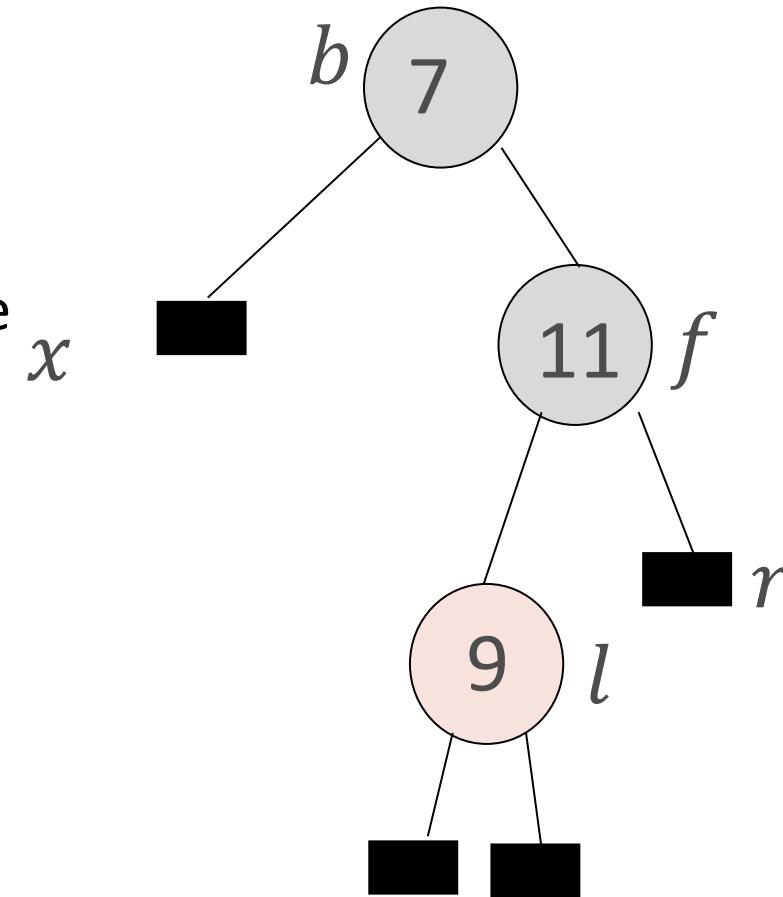
- Supprimer 5

- Rebalancement #1

- ✓ f est à droite de b et le noeud rouge est l .

- ✓ On rebalance en faisant un **ZigZagDroit** sur b .

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

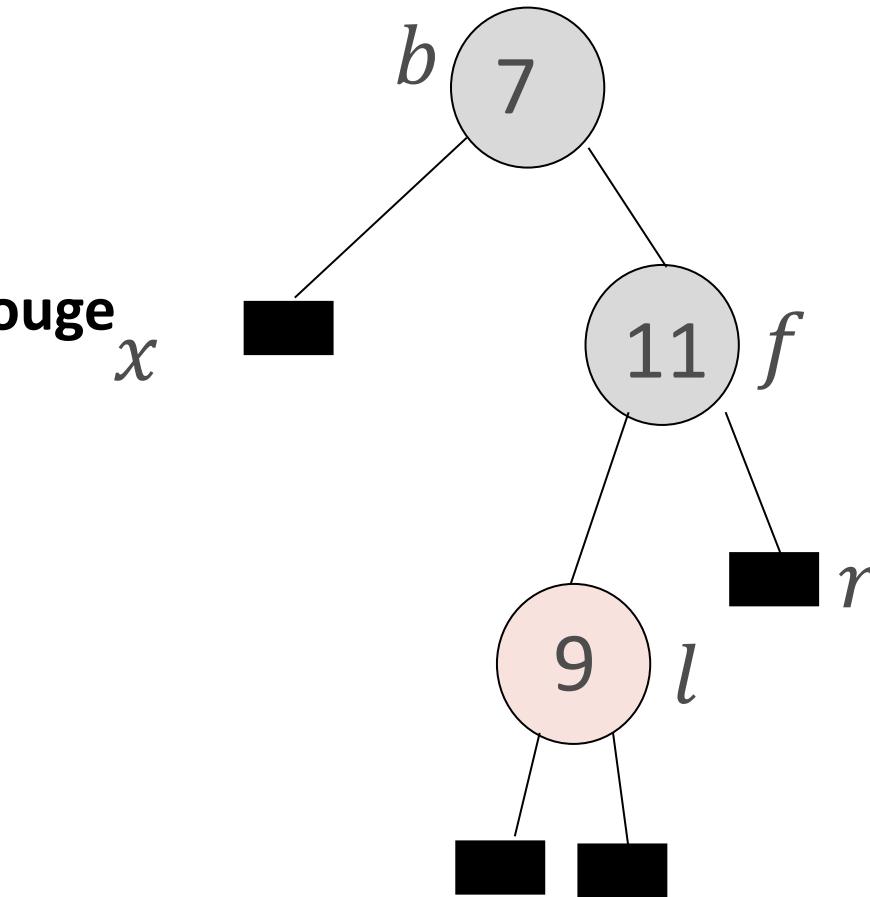
- Résolution

- Supprimer 5
- **Rebalance #1**

- ✓ f est à droite de b et le noeud rouge est l .

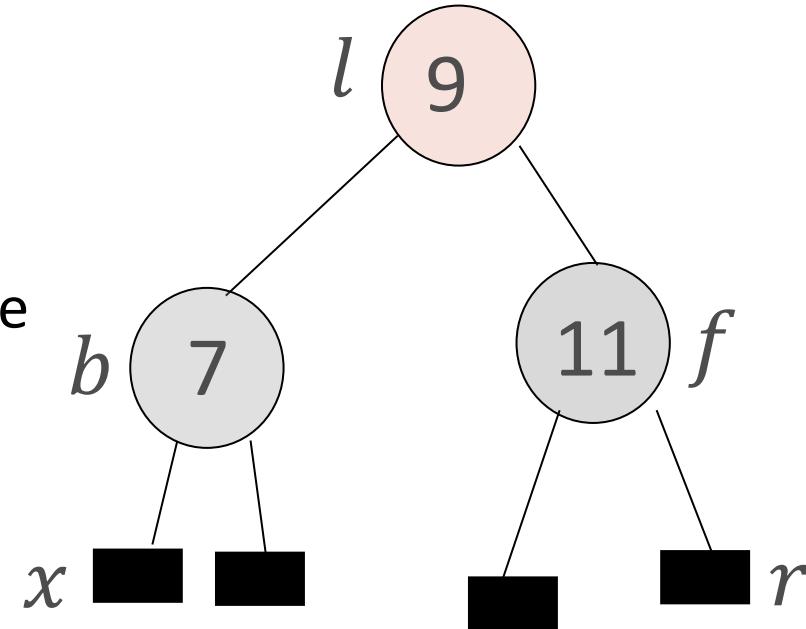
- ✓ On rebalance en faisant un ZigZagDroit sur b .

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

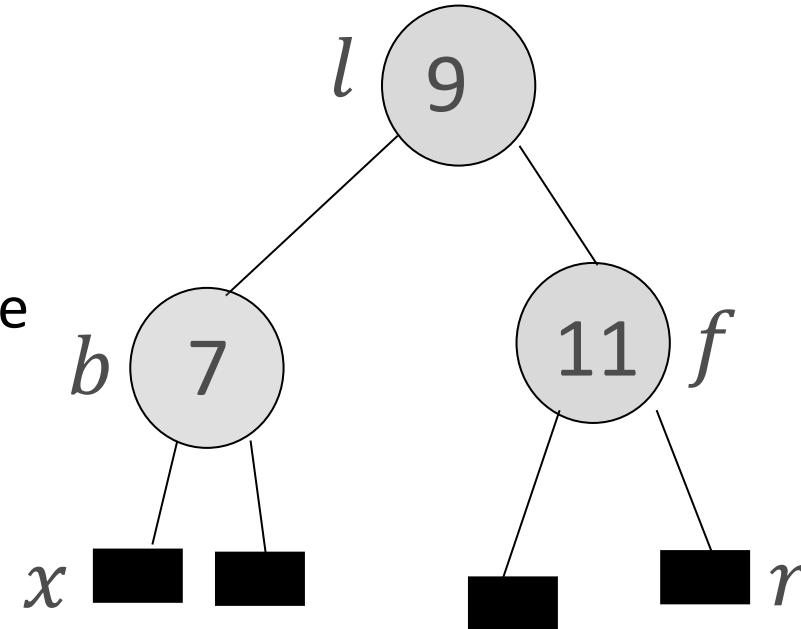
- Résolution
 - Supprimer 5
 - Rebalancement #1
 - ✓ f est à droite de b et le noeud rouge est l .
 - ✓ **On rebalance en faisant un ZigZagDroit sur b .**
 - Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

- Supprimer 5
- Rebalancement #1
 - ✓ f est à droite de b et le noeud rouge est l .
 - ✓ On rebalance en faisant un ZigZagDroit sur b .
- Colorier l en noir

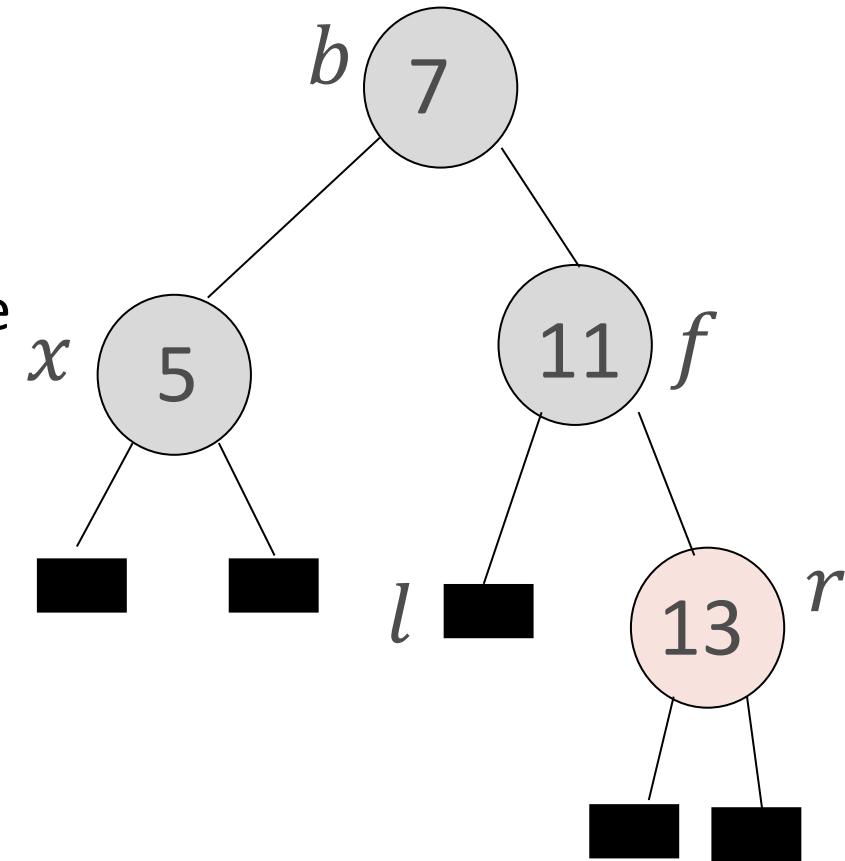


Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

- Supprimer 5
- Rebalancement #2
 - ✓ f est à droite de b et le noeud rouge est r .
 - ✓ On rebalance en faisant un **ZigZigDroit** sur b .

- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

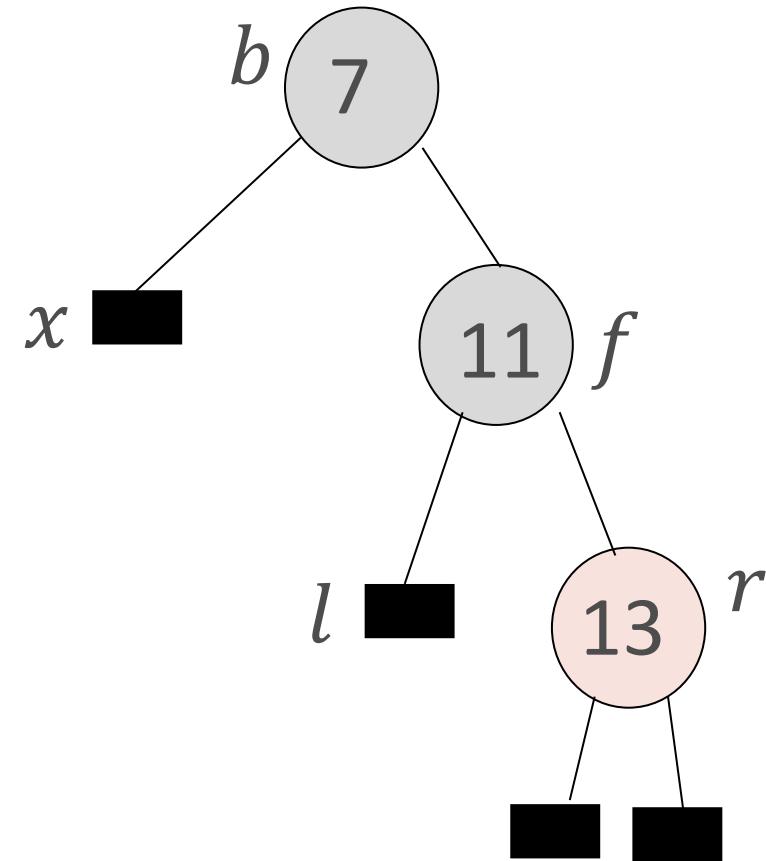
- Supprimer 5

- Rebalancement #2

- ✓ f est à droite de b et le noeud rouge est r .

- ✓ On rebalance en faisant un **ZigZigDroit** sur b .

- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

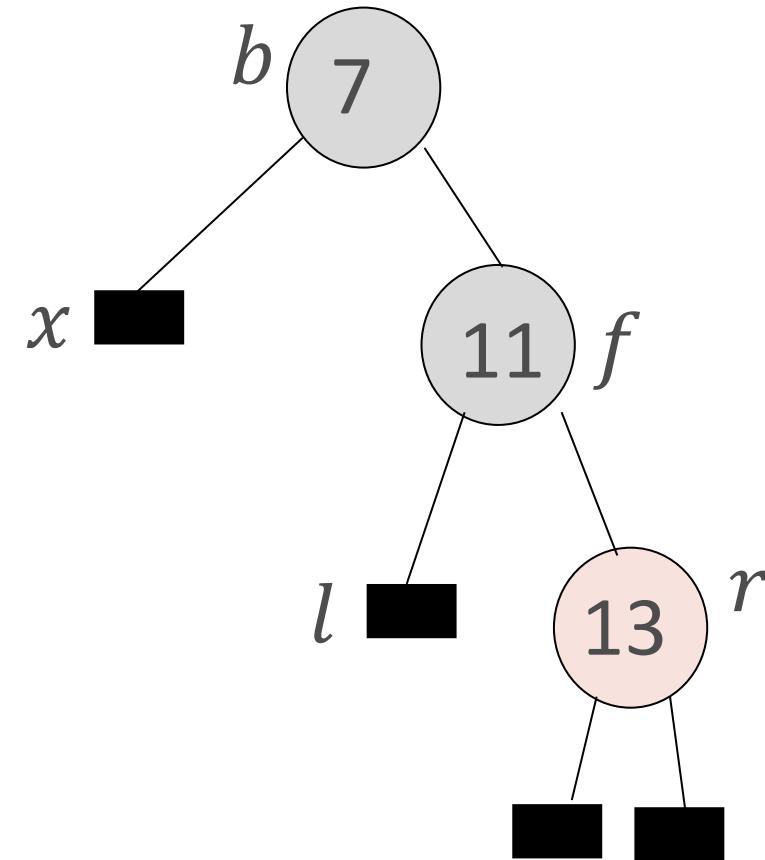
- Supprimer 5

- **Rebalance #2**

- ✓ f est à droite de b et le noeud rouge est r .

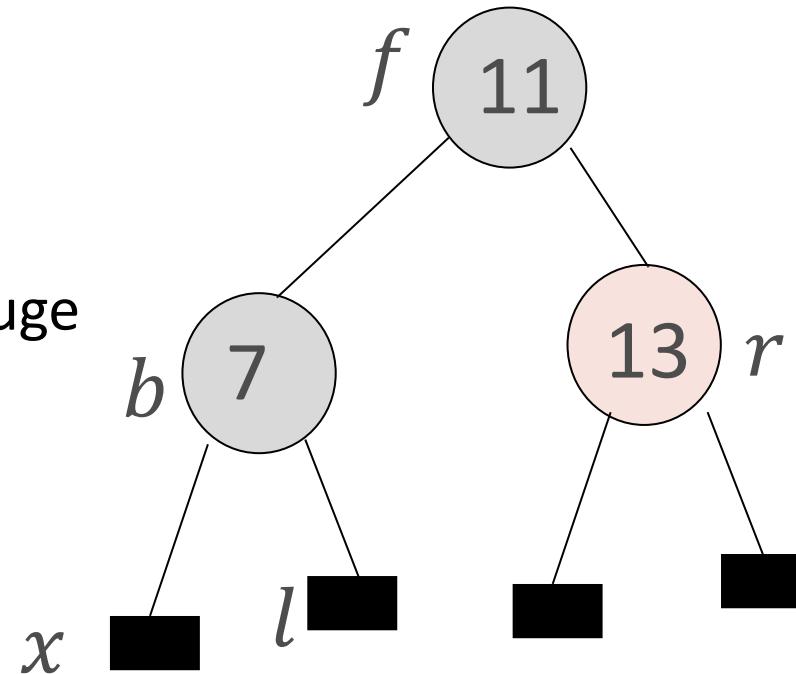
- ✓ On rebalance en faisant un **ZigZigDroit** sur b .

- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

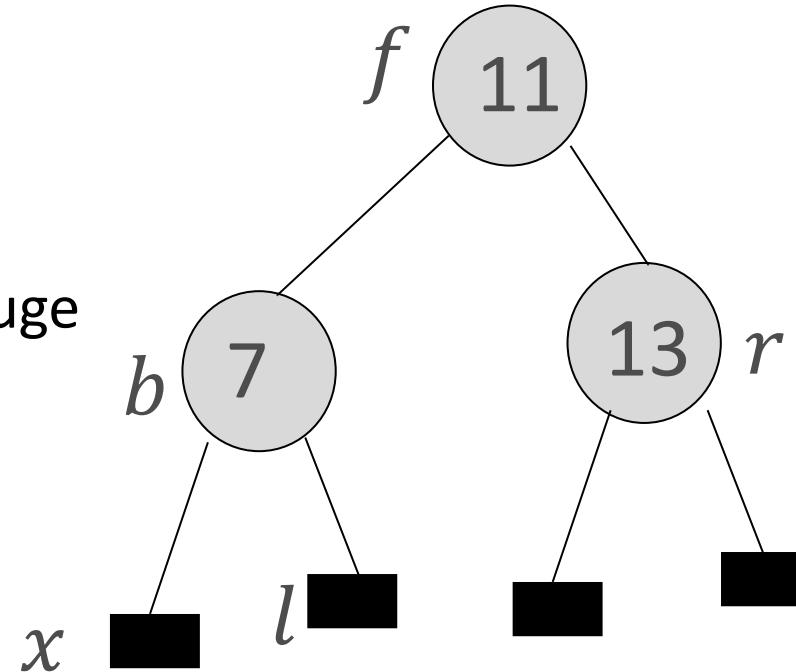
- Résolution
 - Supprimer 5
 - Rebalancement #2
 - ✓ f est à droite de b et le noeud rouge est r .
 - ✓ **On rebalance en faisant un ZigZigDroit sur b .**
 - Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

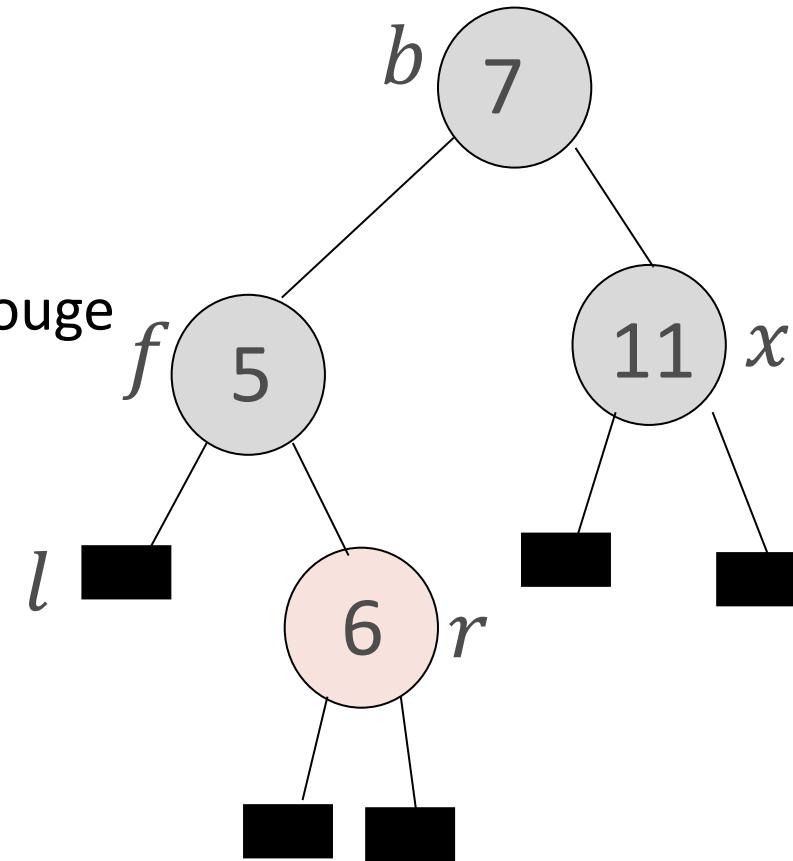
- Supprimer 5
- Rebalancement #2
 - ✓ f est à droite de b et le noeud rouge est r .
 - ✓ On rebalance en faisant un ZigZigDroit sur b .
- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

- Supprimer 11
- Rebalancement #3
 - ✓ f est à gauche de b et le noeud rouge est r .
 - ✓ On rebalance en faisant un **ZigZagGauche** sur b .
- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

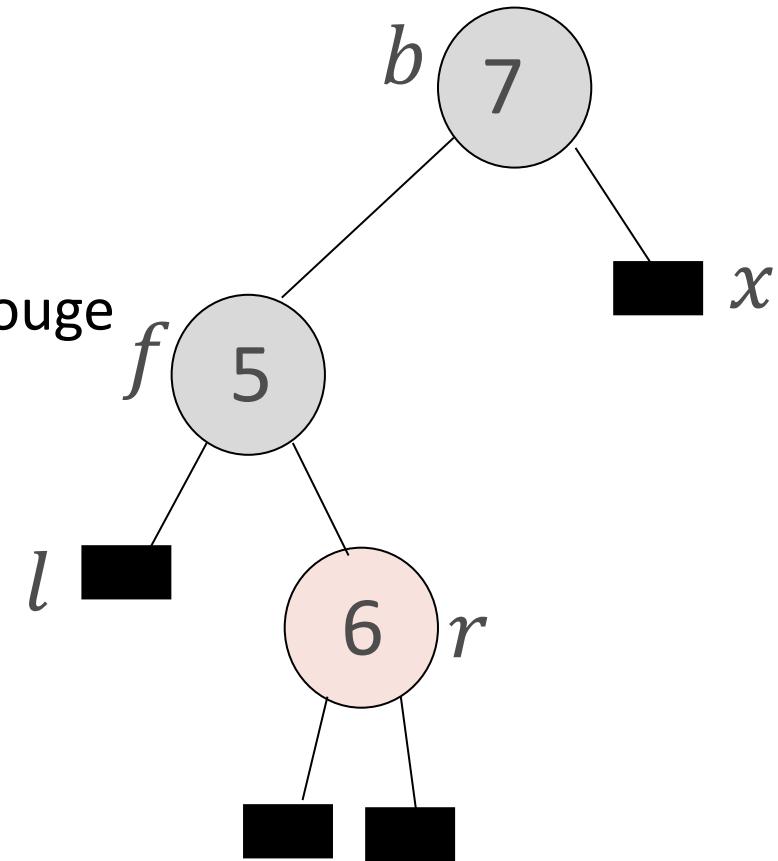
- Supprimer 11

- Rebalancement #3

- ✓ f est à gauche de b et le noeud rouge est r .

- ✓ On rebalance en faisant un **ZigZagGauche** sur b .

- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

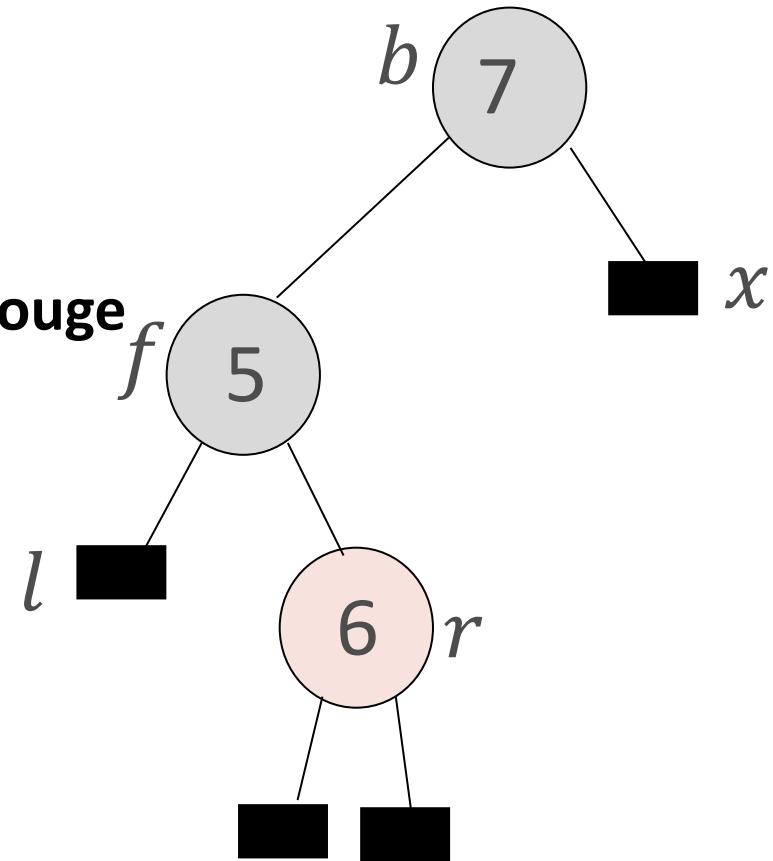
- Supprimer 11

- **Rebalance #3**

- ✓ f est à gauche de b et le noeud rouge est r .

- ✓ On rebalance en faisant un **ZigZagGauche** sur b .

- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

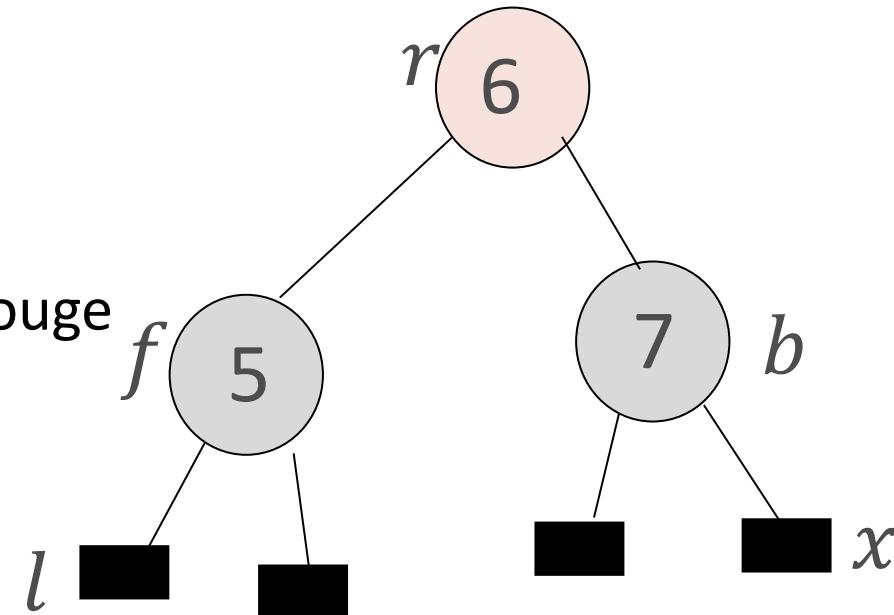
- Supprimer 11

- Rebalancement #3

- ✓ f est à gauche de b et le noeud rouge est r .

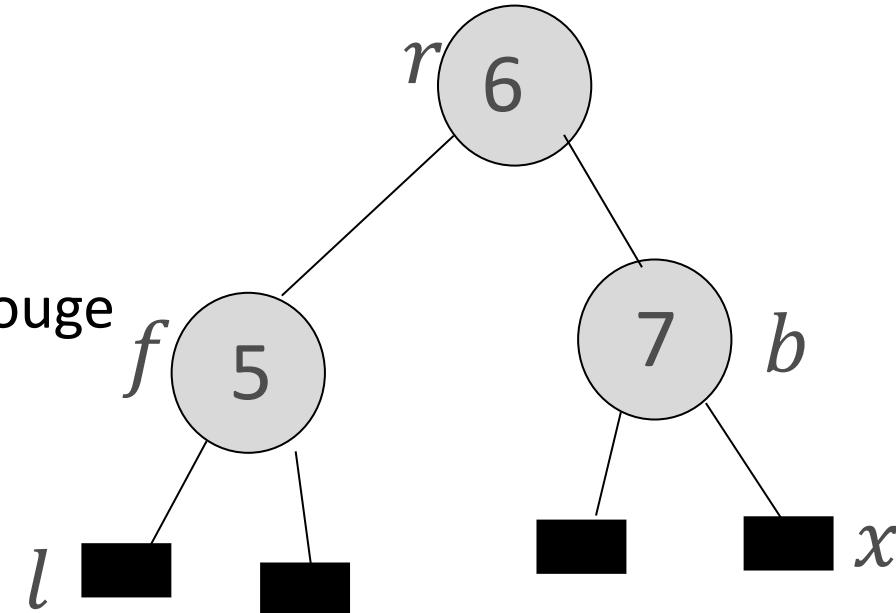
- ✓ **On rebalance en faisant un ZigZagGauche sur b .**

- Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

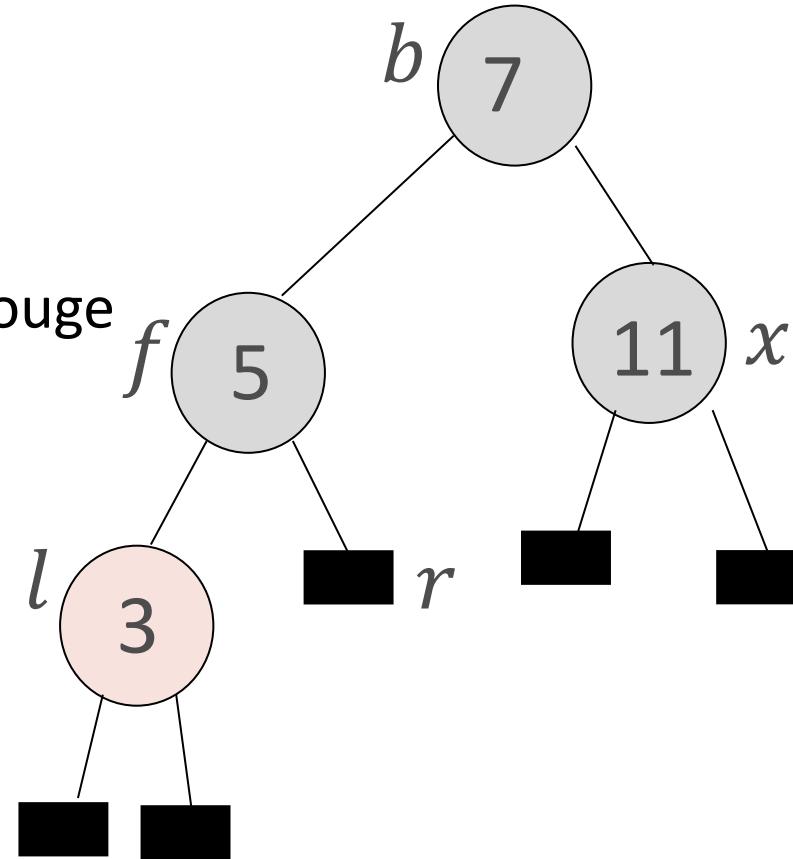
- Résolution
 - Supprimer 11
 - Rebalancement #3
 - ✓ f est à gauche de b et le noeud rouge est r .
 - ✓ On rebalance en faisant un ZigZagGauche sur b .
 - Colorier r en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

- Supprimer 11
- Rebalancement #4
 - ✓ f est à gauche de b et le noeud rouge est l .
 - ✓ On rebalance en faisant un **ZigZigGauche** sur b .
- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

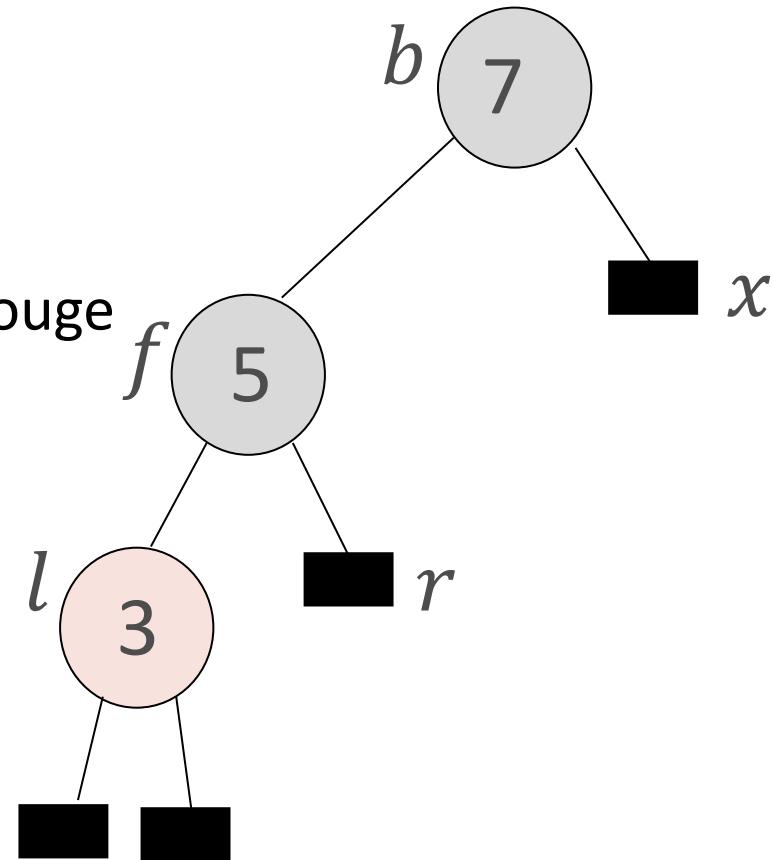
- Supprimer 11

- Rebalancement #4

- ✓ f est à gauche de b et le noeud rouge est l .

- ✓ On rebalance en faisant un **ZigZigGauche** sur b .

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

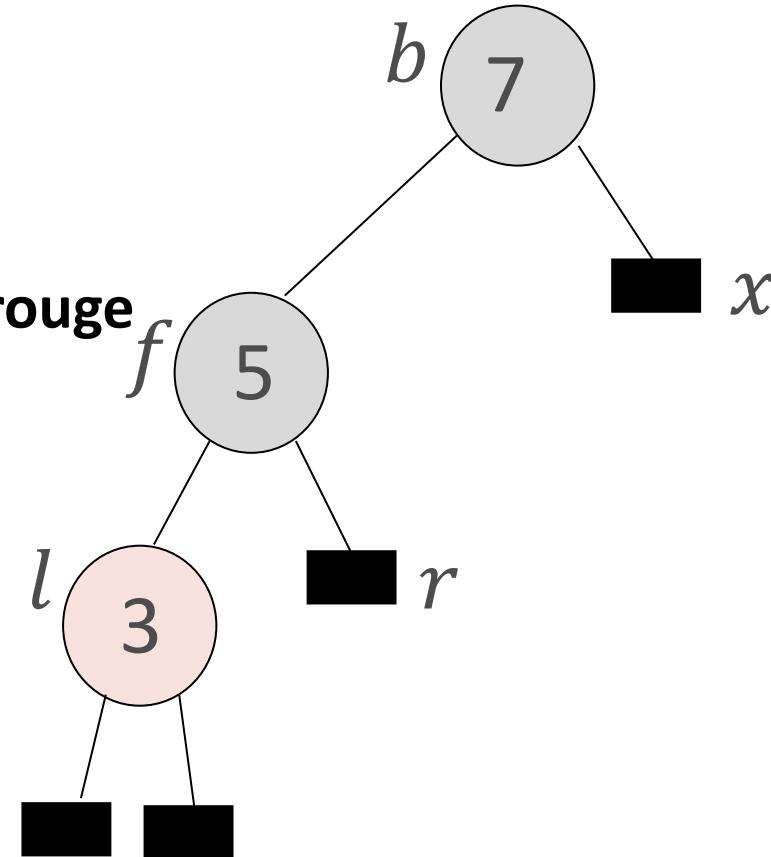
- Résolution

- Supprimer 11
 - **Rebalance #4**

- ✓ f est à gauche de b et le noeud rouge est l .

- ✓ On rebalance en faisant un **ZigZigGauche** sur b .

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

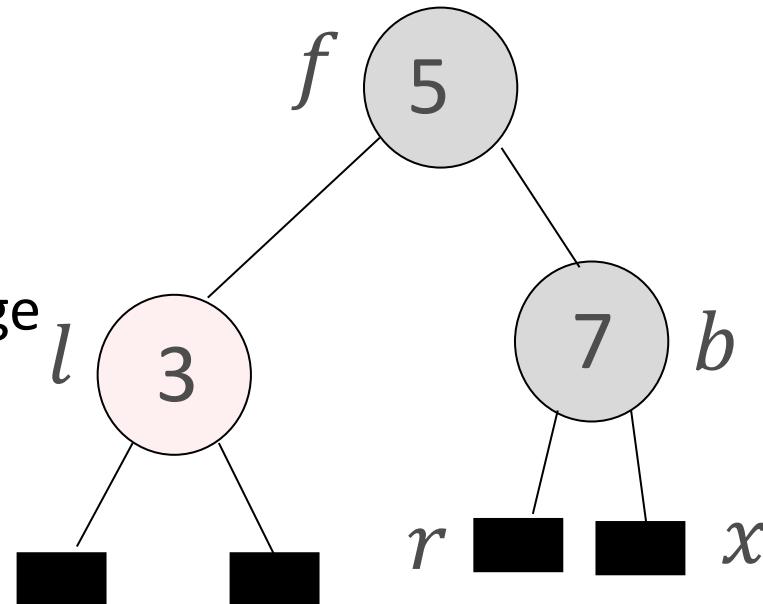
- Supprimer 11

- Rebalancement #4

- ✓ f est à gauche de b et le noeud rouge est l .

- ✓ **On rebalance en faisant un ZigZigGauche sur b .**

- Colorier l en noir



Suppression – Fixer les propriétés : Cas 2c, Si f est noir un de ses enfants est rouge.

- Résolution

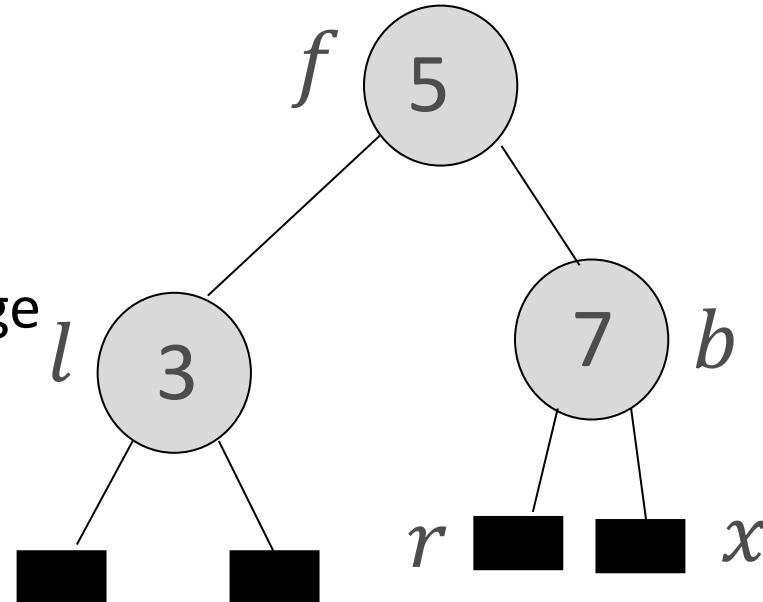
- Supprimer 11

- Rebalancement #4

- ✓ f est à gauche de b et le noeud rouge est l .

- ✓ On rebalance en faisant un ZigZigGauche sur b .

- Colorier l en noir



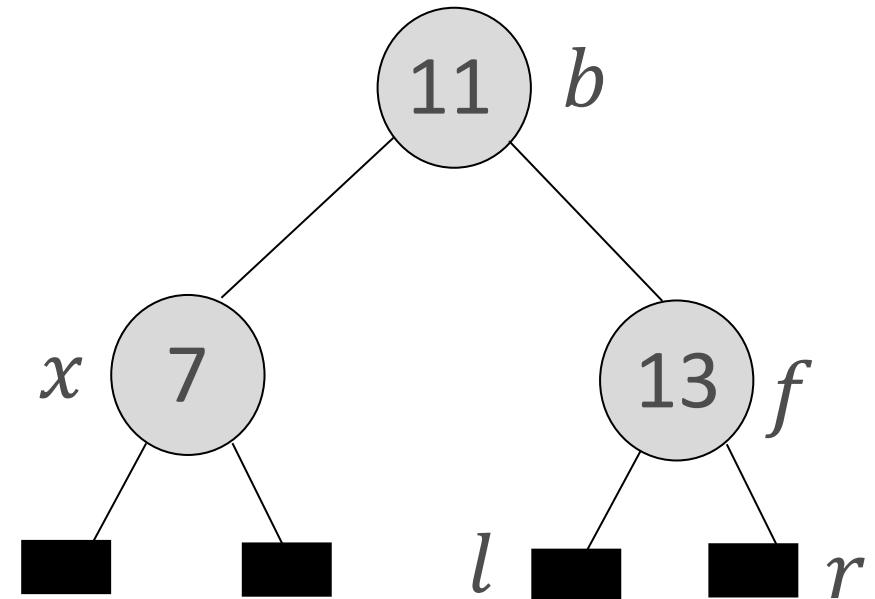
Supression, simplification des deux derniers cas

- On peut mettre les deux cas b et c dans le même cas et faire un ZigZig si le noeud rouge est à la bonne position.

Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

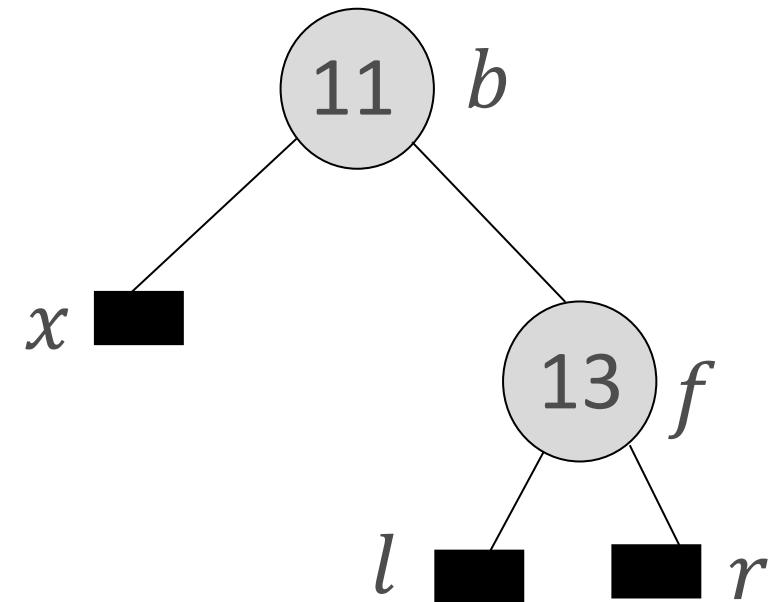
- Supprimer 7
- Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On colorie f rouge
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

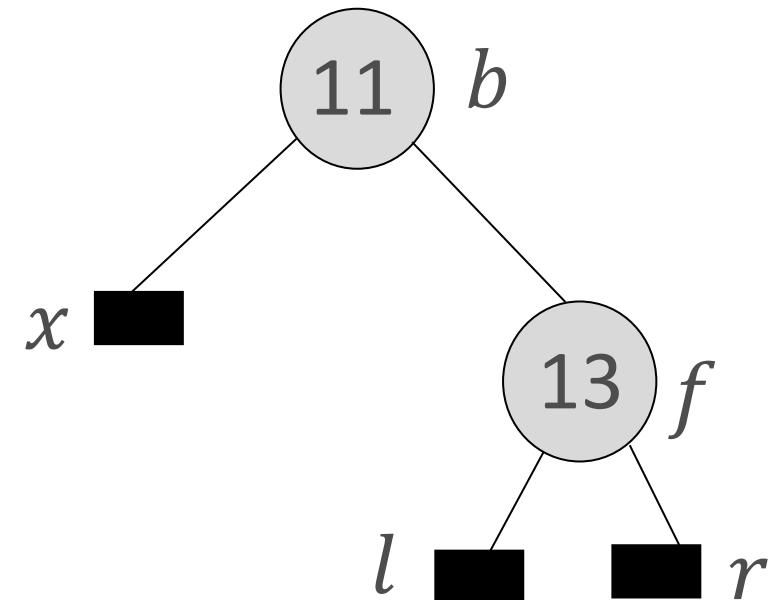
- Supprimer 7 (on a un noeud noir de moins sur le chemin vers x).
- Rebalance #1
 - ✓ f est à droite de b
 - ✓ On colorie f rouge
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

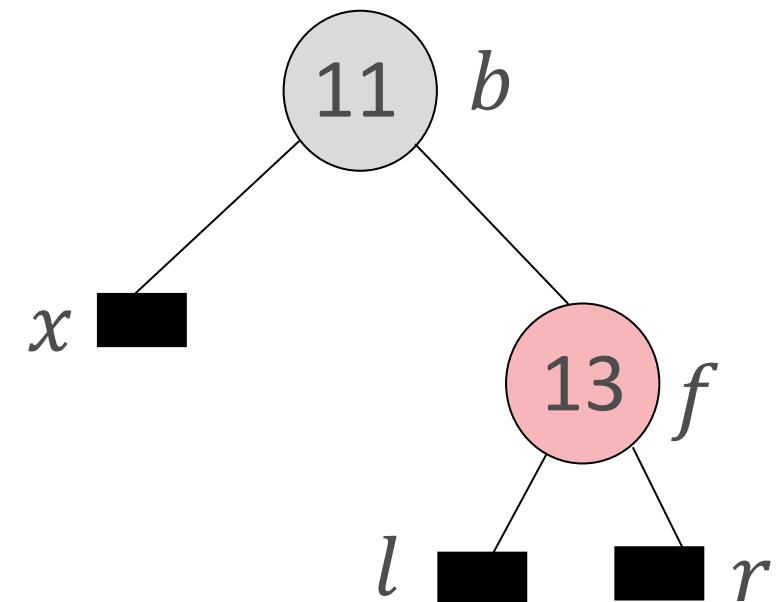
- Supprimer 7
- **Rebalance #1**
 - ✓ f est à droite de b
 - ✓ On colorie f rouge
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

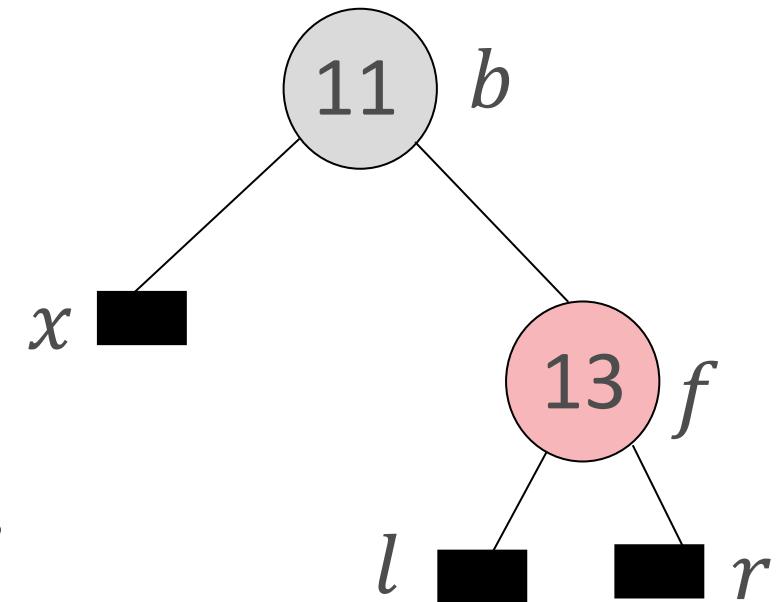
- Résolution

- Supprimer 7
- Rebalancement #1
 - ✓ f est à droite de b
 - ✓ **On colorie f rouge**
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

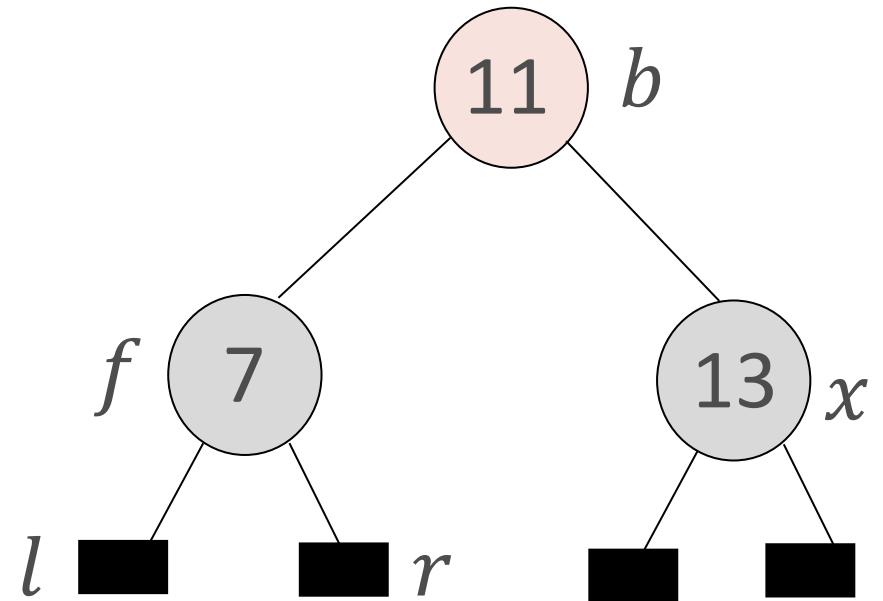
- Résolution
 - Supprimer 7
 - Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On colorie f rouge
 - Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
(On a pas ajouté le noir supprimé sur le chemin).
 - Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

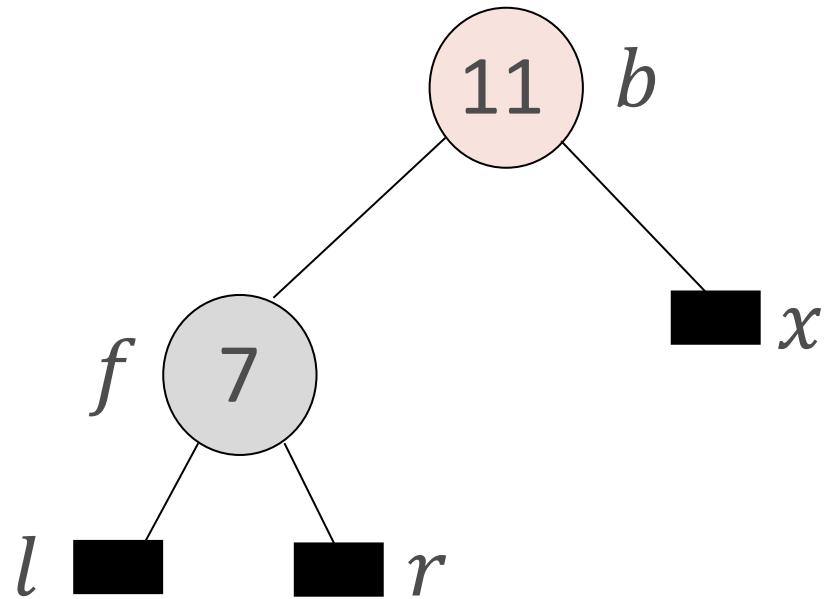
- Supprimer 13
- Rebalancement #1
 - ✓ f est à gauche de b
 - ✓ On colorie f rouge
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

- Supprimer 13
- Rebalancement #1
 - ✓ f est à gauche de b
 - ✓ On colorie f rouge
- On refait les étapes récursivement pour b tant que son parent est noir.

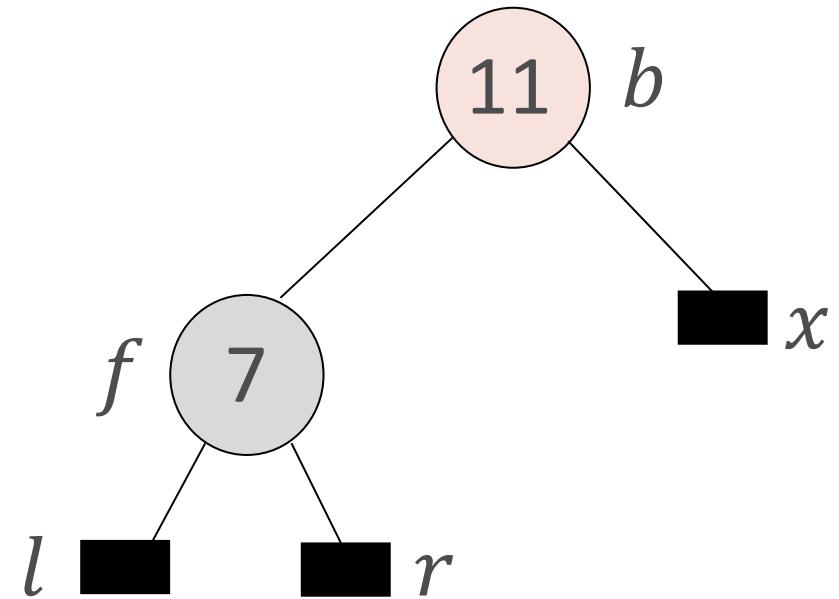


Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

- Supprimer 13
- **Rebalance #1**
 - ✓ f est à gauche de b
 - ✓ On colorie f rouge

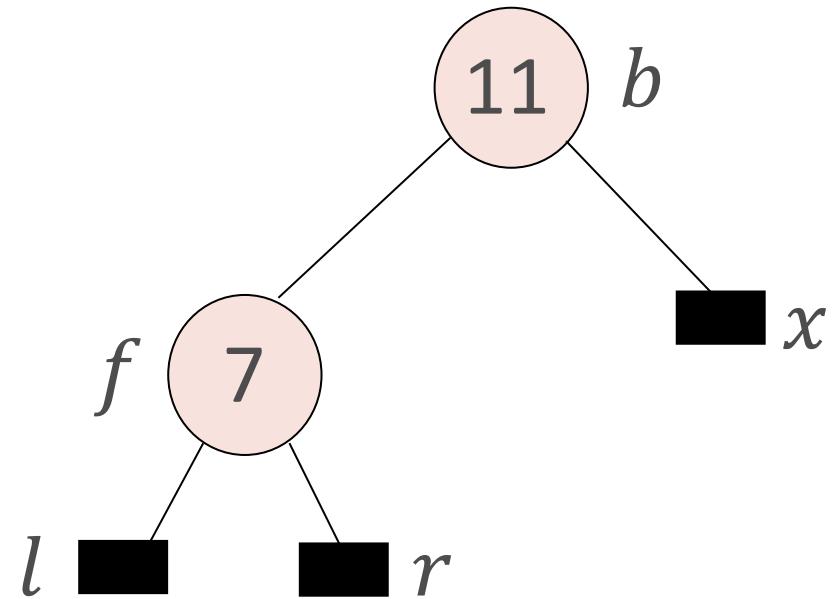
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

- Résolution

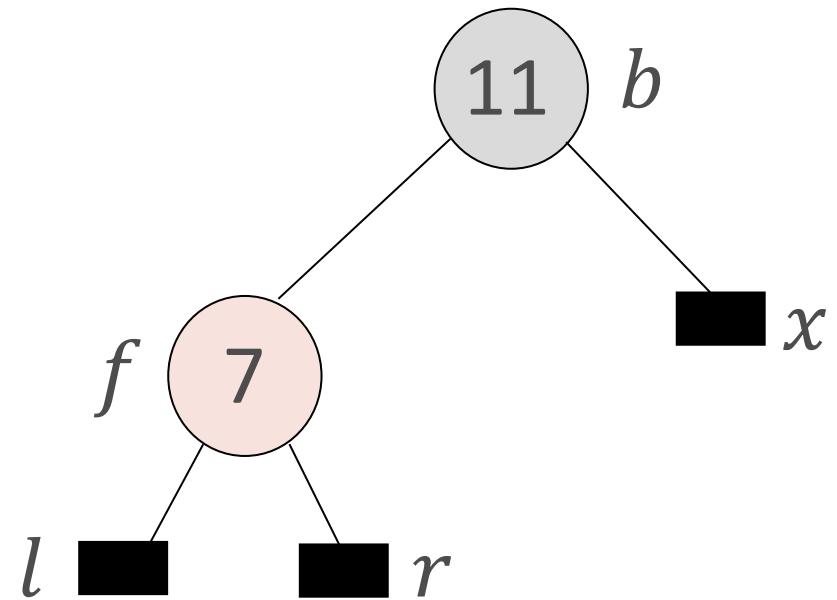
- Supprimer 13
- Rebalancement #1
 - ✓ f est à gauche de b
 - ✓ **On colorie f rouge**
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- Si b est rouge, on le colorie noir.



Suppression – Fixer les propriétés : Cas 2d, Si f est noir et ses enfants sont noirs.

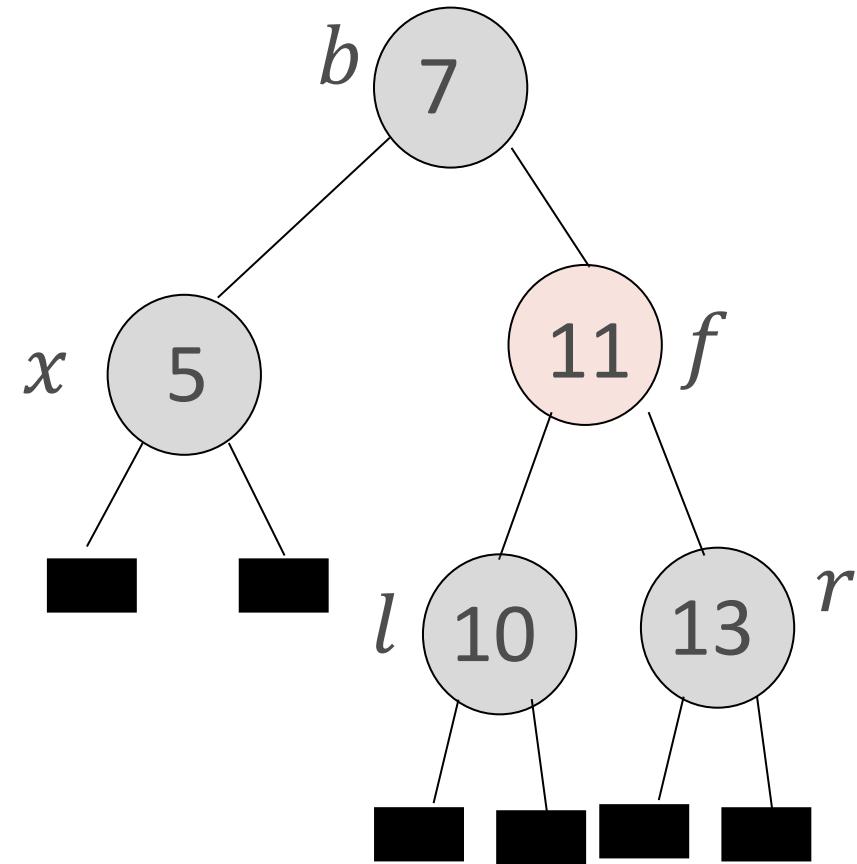
- Résolution

- Supprimer 13
- Rebalancement #1
 - ✓ f est à gauche de b
 - ✓ On colorie f rouge
- Si b est noir, on refait les cas de suppressions pour b tant que b est noir.
- **Si b est rouge, on le colorie noir.
(On a ajouté un noir sur le chemin,
on a donc le même nombre de noeud
noir qu'avant la suppression).**



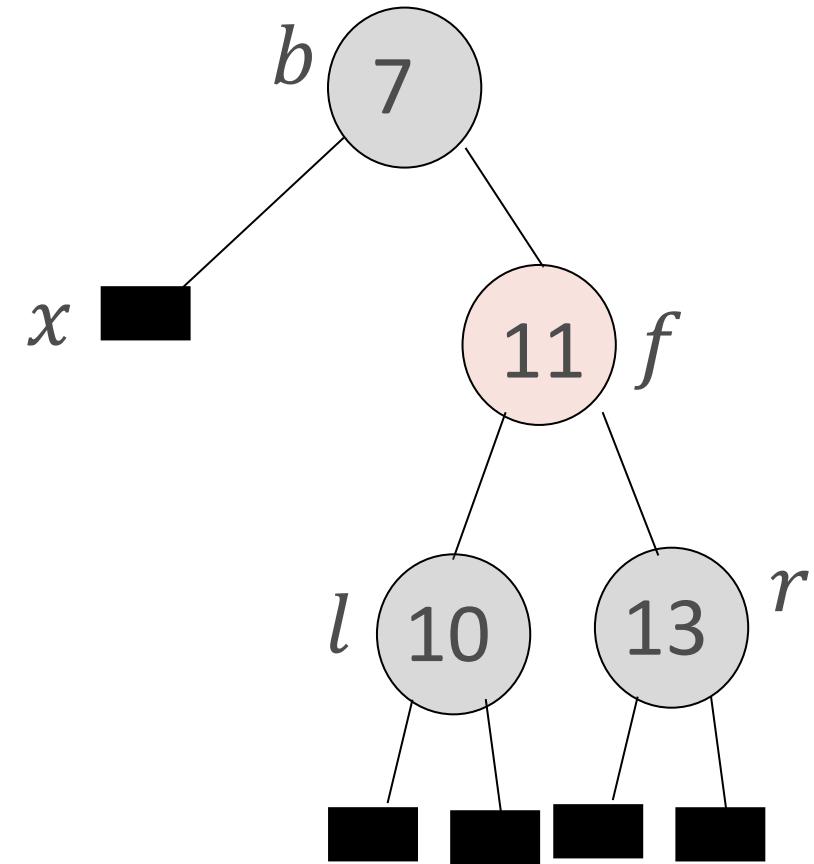
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 5.
 - Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un ZigZigDroit sur b .
 - Colorier f noir
 - Colorier l rouge



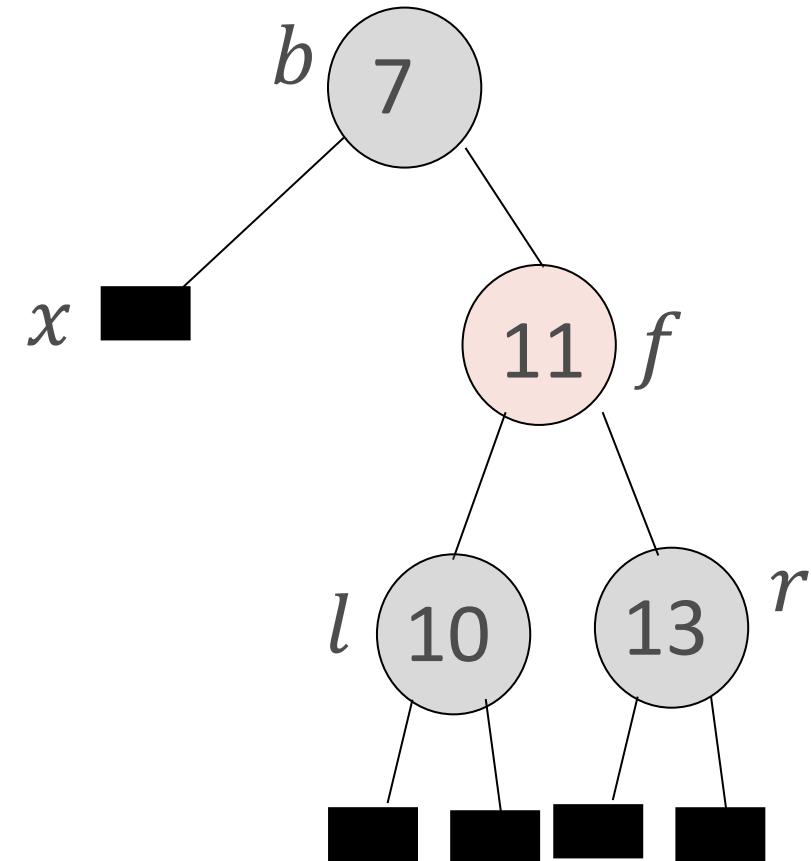
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 5.
 - Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un ZigZigDroit sur b .
 - Colorier f noir
 - Colorier l rouge



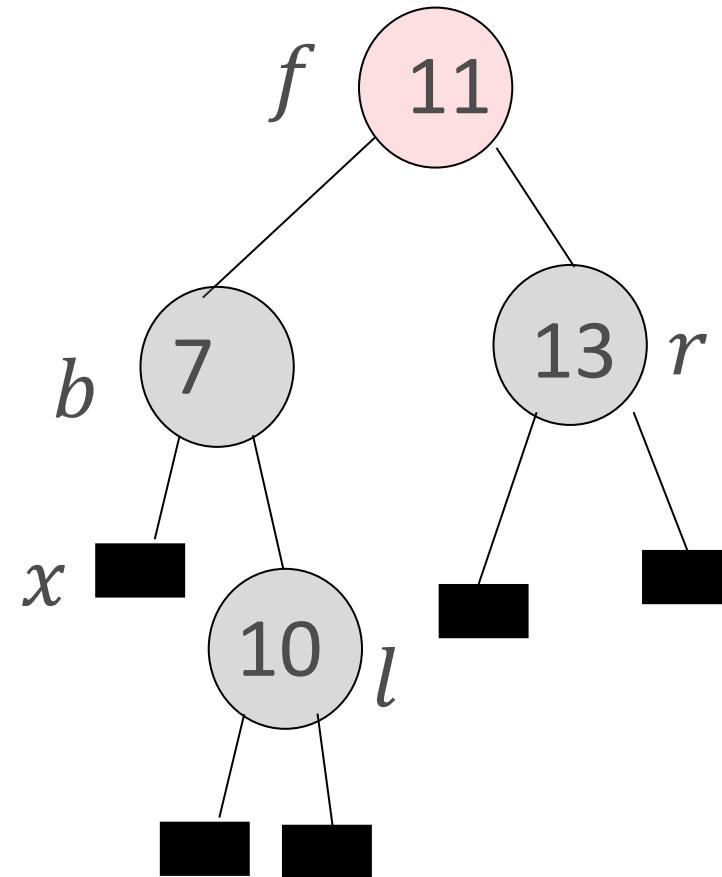
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 5.
 - **Rebalance #1**
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un ZigZigDroit sur b .
 - Colorier f noir
 - Colorier l rouge



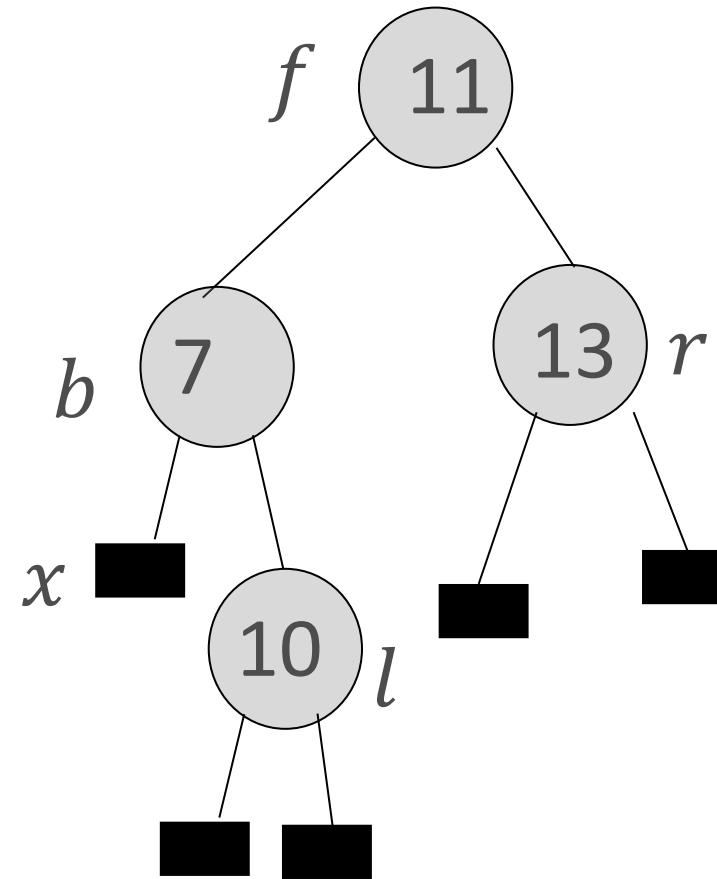
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 5.
 - Rebalancement #1
 - ✓ f est à droite de b
 - ✓ **On rebalance en faisant un ZigZigDroit sur b .**
 - Colorier f noir
 - Colorier l rouge



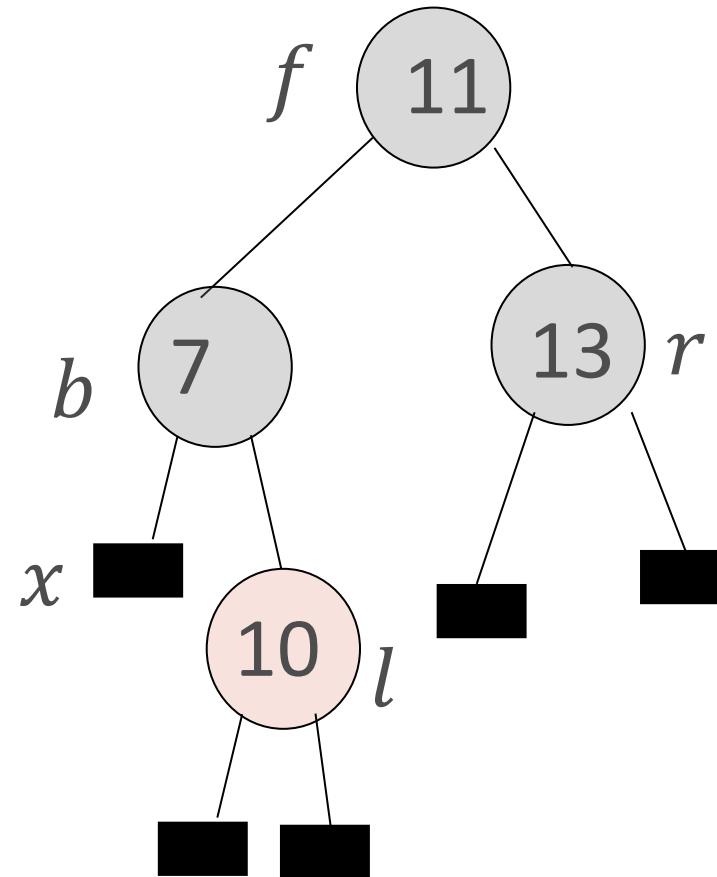
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 5.
 - Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un ZigZigDroit sur b .
 - Colorier f noir
 - Colorier l rouge



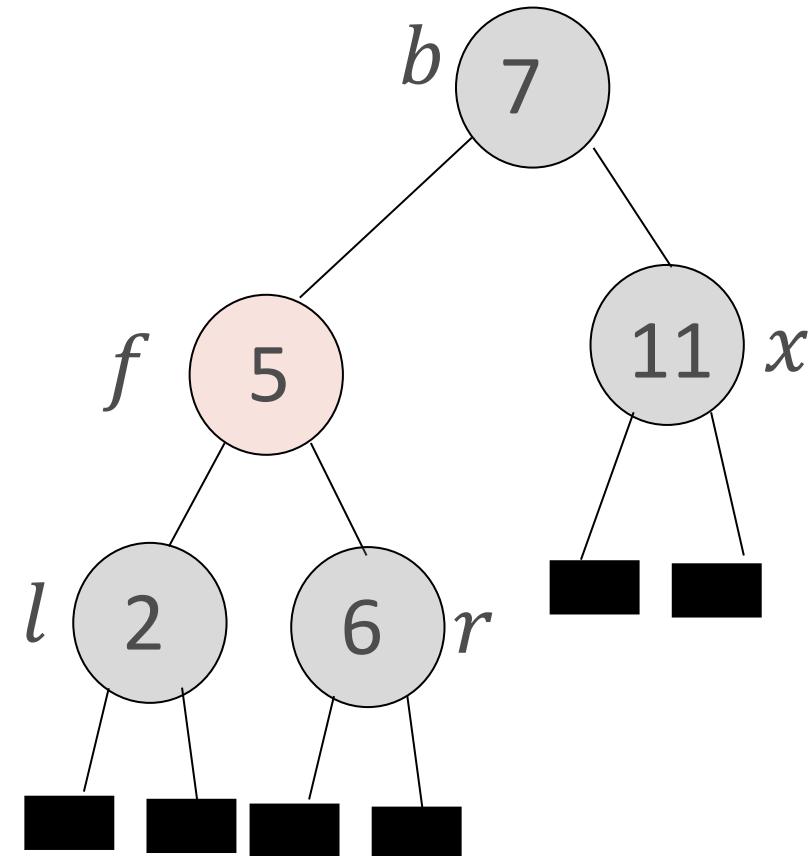
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 5.
 - Rebalancement #1
 - ✓ f est à droite de b
 - ✓ On rebalance en faisant un ZigZigDroit sur b .
 - Colorier f noir
 - Colorier l rouge



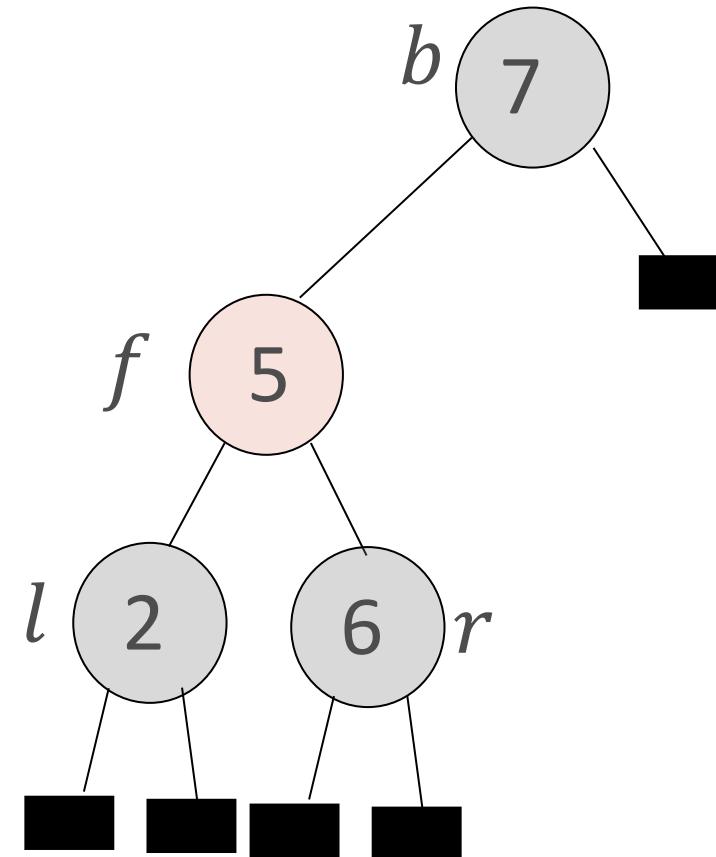
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 11.
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un ZigZigGauche sur b .
 - Colorier f noir
 - Colorier r rouge



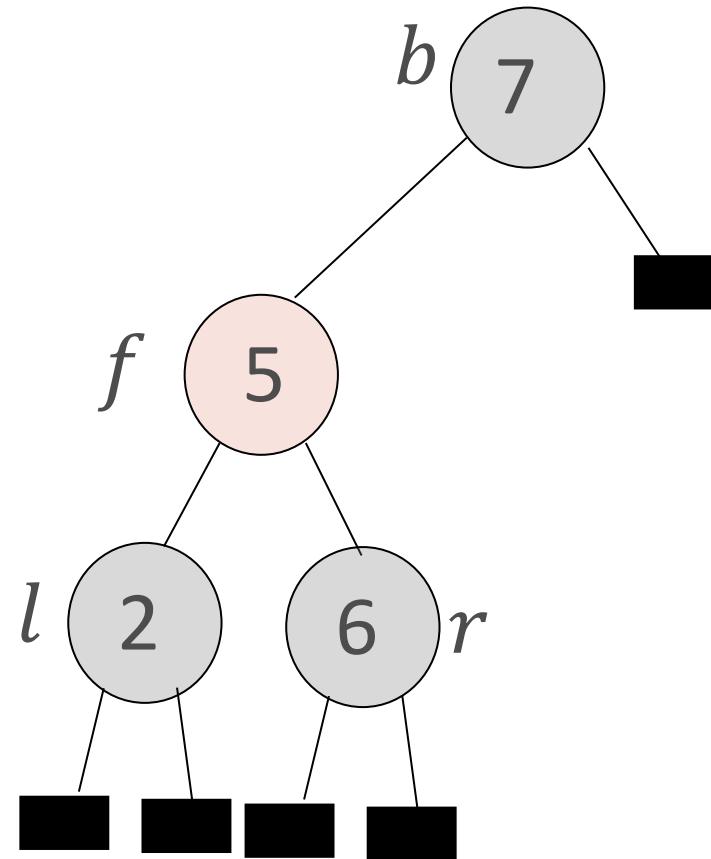
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 11.
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un ZigZigGauche sur b .
 - Colorier f noir
 - Colorier r rouge



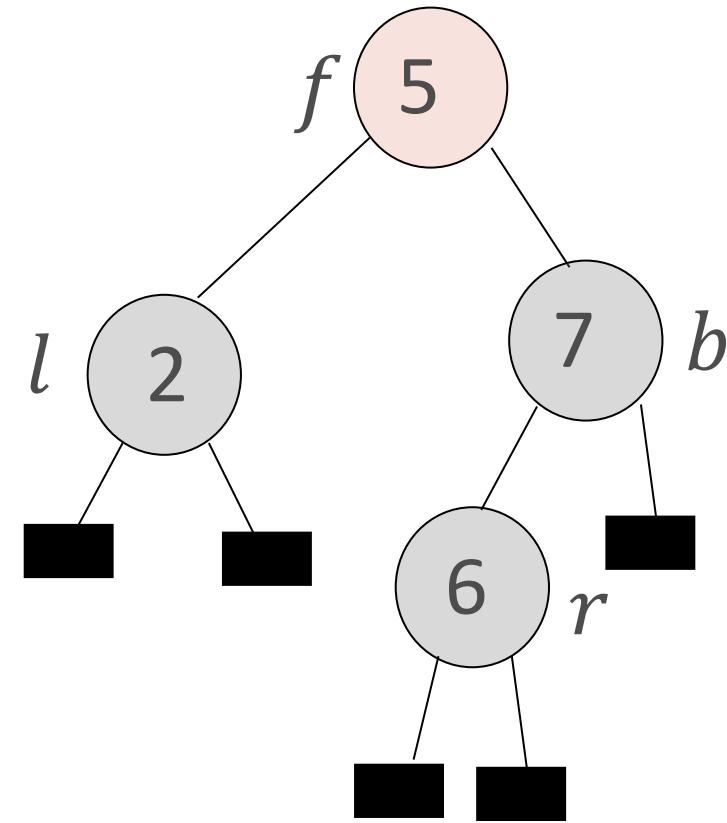
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 11.
 - **Rebalance #2**
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un ZigZigGauche sur b .
 - Colorier f noir
 - Colorier r rouge



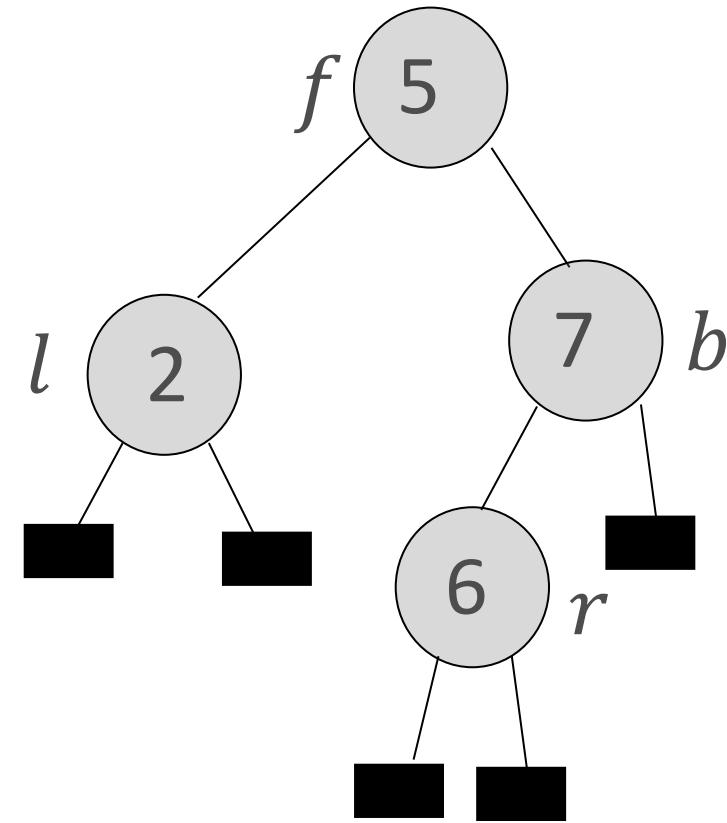
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 11.
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ **On rebalance en faisant un ZigZigGauche sur b .**
 - Colorier f noir
 - Colorier r rouge



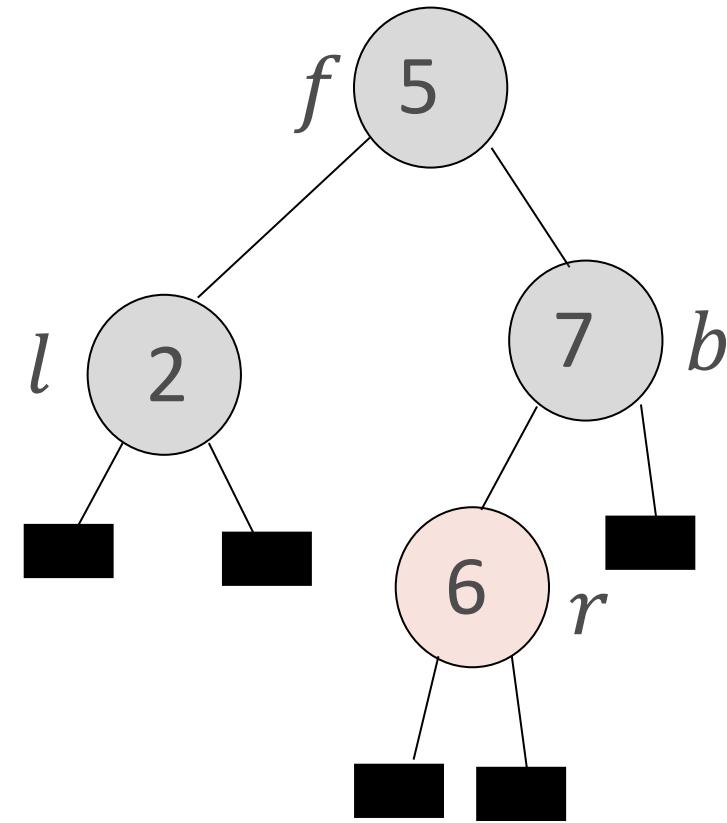
Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 11.
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un ZigZigGauche sur b .
 - Colorier f noir
 - Colorier r rouge



Supression – Fixer les propriétés : Cas 2e, f est rouge

- Résolution
 - Supprimer 11.
 - Rebalancement #2
 - ✓ f est à gauche de b
 - ✓ On rebalance en faisant un ZigZigGauche sur b .
 - Colorier f noir
 - **Colorier r rouge**



Synthèse

- Arbre rouge et noir plus complexe que AVL
- Moins de rotations que les arbres AVL
- Recherche plus longue pour red-black (l'arbre peut être un peu plus débalancé que AVL)