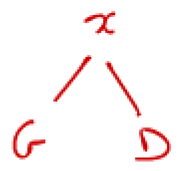
TD 6

Exercice 1

Arbre binaire : $T = \emptyset$ ou 

$h(T)$: hauteur de T

$n(T)$: nombre de noeuds de T

$f(T)$: nombre de feuilles de T

$n_1(T)$: nombre de noeuds à 1 fil

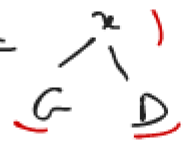
$n_2(T)$: nombre de noeuds à 2 fils

$n_i(T)$: nombre de noeuds internes

↳ $n_i(T) = n_1(T) + n_2(T)$

$a(T)$: nombre d'arêts

en fonction de x, G, D.

$T =$ 

$h(T) = 1 + \max(h(G), h(D))$

$n(T) = 1 + n(D) + n(G)$

$f(T) = f(G) + f(D)$

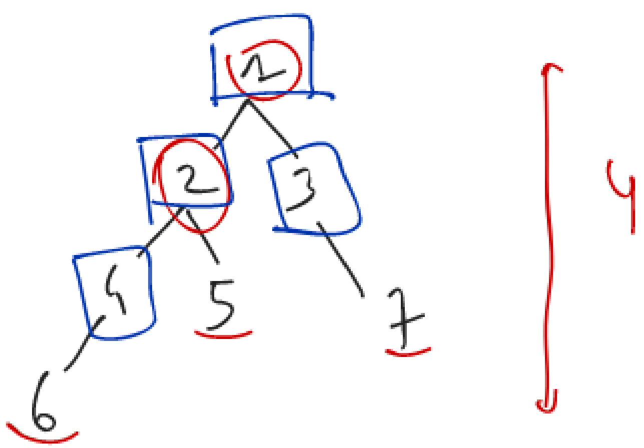
$n_1(T) = n_1(G) + n_1(D)$

$n_2(T) = 1 + n_2(G) + n_2(D)$

$n_i(T) = 1 + n_i(G) + n_i(D)$

$a(T) = 2 + a(G) + a(D)$

1) $h(T_0) = 4$, $f(T_0) = 3$, $n_1(T_0) = 2$, $n_2(T_0) = 2$



$n(T) = n_i(T) + f(T)$

$n_i(T) = n_1(T) + n_2(T)$

$\boxed{Q2}$ $h(T) \leq n(T) \leq 2^{h(T)} - 1$

T arbre binaire : $\emptyset$ ou (x, G, D) avec G et D arbres binaires

$\emptyset$ ou 

. Cas de base $\downarrow$ $\boxed{T=\emptyset}$

$h(\emptyset) = 0$

$n(\emptyset) = 0$

$2^{h(\emptyset)} - 1 = 1 - 1 = 0$

$\Big\}$ donc la propriété est vérifiée pour $T=\emptyset$

. Induction . Soit T un arbre binaire non vide,

$T =$ 

On suppose que la propriété est vraie pour G et D. Montrons-la pour T.

Méthode ♡

Mq $h(T) \leq n(T) \leq 2^{h(T)} - 1$

Méthode. On se ramène aux sous-arbres G et D pour appliquer l'hypothèse d'induction (HI)

$\boxed{\max(X,Y) \leq X+Y}$

. $h(T) = 1 + \max(h(G), h(D))$

$n(T) = 1 + n(G) + n(D)$

$h(T) = 1 + \max(h(G), h(D)) \leq 1 + h(G) + h(D) \underset{HI}{\leq} 1 + n(G) + n(D)$

Donc $h(T) \leq n(T)$.

. $n(T) = 1 + n(G) + n(D) \underset{HI}{\leq} \cancel{1} + 2^{h(G)} \cancel{-1} + 2^{h(D)} - 1$

$\leq 2^{\max(h(G), h(D))} \times 2 - 1$

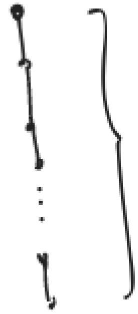
$= 2^{\max(h(G), h(D)) + 1} - 1$

$= 2^{h(T)} - 1$

Donc $n(T) \leq 2^{h(T)} - 1$

. Conclusion : la propriété est vraie pour tout arbre binaire.

$$h(T) \le n(T) \le 2^{h(T)} - 1$$



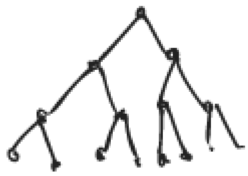
$h(T) = n(T)$

$\longrightarrow$ $\boxed{\log_2(n(T)+1) \le h(T) \le n(T)}$

car $n(T) \le 2^{h(T)} - 1$

donc $n(T) + 1 \le 2^{h(T)}$

donc $\log_2(n(T)+1) \le h(T)$



$\begin{cases} h(T) = \log_2(n(T)+1) \\ n(T) = 2^{h(T)} - 1 \end{cases}$

3) $\boxed{T \text{ non vide}}$: $(x,\emptyset,\emptyset)$, $(x,G,\emptyset)$, $(x,\emptyset,D)$, (x,G,D)

Mq $f(T) = n_2(T) + 1$

. Base : Soit $T = (x, \emptyset, \emptyset)$

Alors $f(T) = 1$ (x est une feuille car n'a pas de fils)

or $n_2(T) = 0$

Donc $f(T) = n_2(T) + 1$

. Induction : Soit T un ABV tq $T = (x,G,\emptyset)$ ou $(x,\emptyset,D)$ ou (x,G,D)

On suppose que la propriété vraie pour G et D.

. Si $T = (x,G,\emptyset)$ T =

$f(T) = f(G)$

$n_2(T) = n_2(G)$

x n'a qu'un seul fils, on ne le compte pas

Par HI : $f(G) = n_2(G) + 1$

↳ Donc $f(T) = n_2(T) + 1$

· Si $T = (x, \emptyset, D)$, $T =$ 

Même chose : $f(T) = n_2(T) + 1$

· Si $T = (x, G, D)$ 

- $f(T) = f(G) + f(D)$ car x n'est pas une feuille
- $n_2(T) = n_2(G) + n_2(D) + 1$ car x est un nœud interne à deux fils.

Par HI : $\begin{cases} f(G) = n_2(G) + 1 \\ f(D) = n_2(D) + 1 \end{cases}$

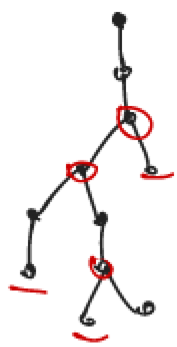
→ donc $f(T) = (n_2(G) + 1) + (n_2(D) + 1)$
$= [n_2(G) + n_2(D) + 1] + 1$

donc $f(T) = n_2(T) + 1$

· Conclusion : la propriété est vraie pour tout ABV.

* $\boxed{f(T) = n_2(T) + 1}$:

(à retenir)



* Mq $n(T) = n_1(T) + 2n_2(T) + 1$

On sait que $\boxed{n(T) = n_1(T) + n_2(T) + f(T)}$ *

On conclut grâce à $f(T) = n_2(T) + 1$

Analyse au brouillon :
on suppose $n(T) = n_1(T) + 2n_2(T) + 1$
alors $n(T) = n_1(T) + n_2(T) + f(T)$
d'après la formule
et on sait le montrer

* Mq $f(T) \le n_i(T) + 1$

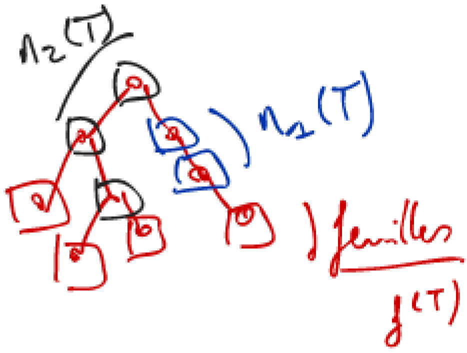
lien entre $\underbrace{n_i}_{\text{noeuds internes}}$, $\underbrace{n_1}_{\text{noeuds internes à 1 fils}}$ et $\underbrace{n_2}_{\text{noeuds internes à 2 fils}}$:

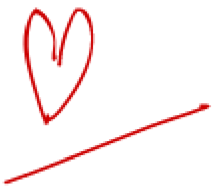
$$\boxed{n_i(T) = n_1(T) + n_2(T)}$$

Or $f(T) = n_2(T) + 1 \le n_i(T) + 1$

* $\underbrace{n_i(T) = n_1(T) + n_2(T)}_{\text{définition}}$

n_i : noeuds = noeuds qui ne sont pas des feuilles

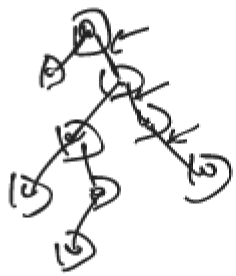




* $n(T) = f(T) + n_i(T)$

$$n(T) = f(T) + n_1(T) + n_2(T)$$

4) $n(T) = a(T) + 1$



Base : $T = (x, \emptyset, \emptyset)$ x

$a(T) = 0$

$n(T) = 1$

Donc $n(T) = a(T) + 1$

Induction : Soit T un ABC qui n'est pas juste une racine.

On suppose la propriété vraie pour les sous-arbres de T

- Si $T = (x, G, \emptyset)$

Alors $n(T) = 1 + n(G)$ HI : $n(G) = a(G) + 1$

$a(T) = 1 + a(G)$

Donc $n(T) = a(T) + 1$

- Si $T = (x, \emptyset, D)$: De même $n(T) = a(T) + 1$

- Si $T = (x, G, D)$:

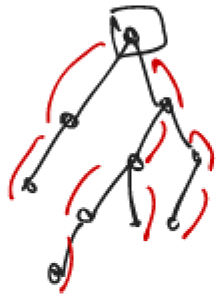
$n(T) = n(G) + n(D) + 1$ et HI : $n(G) = a(G) + 1$

$a(T) = a(G) + a(D) + 2$ $n(D) = a(D) + 1$

$$n(T) = n(G) + n(D) + 1 = \underbrace{a(G) + 1 + a(D) + 1}_{a(T)} + 1$$

Donc $n(T) = a(T) + 1$

- Conclusion : Pour tout ABV T, $n(T) = a(T) + 1$.



Ça se voit sans faire d'induction.

Exercice 2

$T = \emptyset$ ou (x, G, D)

$n(T) = 0$ ou $1 + n(G) + n(D)$

taille = nombre de noeuds

1) (a)
```
def ABtaille(T)
    if estABVide(T)
        return 0
    else
        return 1 + ABtaille(T.gauche) + ABtaille(T.droite)
```

$T = $ (x ; G, D)

Soit $c(n)$ la complexité de la fonction.

$c(n)$ nombre d'additions de la fonction

Mq $c(n) = \Theta(n)$.

$n_T = 1 + n_G + n_D$

Au brouillon, analyse :

(si on n'a pas d'idée)

On suppose $c(n) = \alpha n + A$, $n \geq 1$

On veut que les appels récursifs puissent démontrer $c(n)$ par récurrence

D'après la fonction

$T = $ (x ; G, D)

Raisonnement "Analyse-Synthèse"

Principe

- On suppose au brouillon le résultat vrai
- On remonte le raisonnement pour trouver des notions plus simples à démontrer

$$c(n_T) = 2 + \underbrace{c(n_G)}_{\alpha n_G + A} + \underbrace{c(n_D)}_{\alpha n_G + A}$$

$= 2 + \alpha n_G + A + \alpha n_D + A$, et $n_T = 1 + n_G + n_D$

$= 2 + 2A + \alpha(\underbrace{n_G + n_D + 1}_{n_T}) - \alpha$

$= \underbrace{2 + 2A - \alpha}_{} + \alpha n_T$, car on veut A et α tels que $c(n_T) = \alpha n_T + A$

on veut que ce soit $= A$ pour que la récurrence soit vérifiée

$A = 2 + 2A - \alpha \iff \alpha = 2 + A$. Par exemple $\begin{cases} A = 0 \\ \alpha = 2 \end{cases}$

on aurait pu juste chercher α

et $A = 0$ trouvé

Synthèse : Mq $c(n) = 2n$ par récurrence structurelle

· **Base** : $c(0) = 0$ ok

· **Induction** : On suppose $T =$  et que

$$\begin{cases} c(n_G) = 2n_G \\ c(n_D) = 2n_D \end{cases}$$

(ie la propriété est vraie pour G et D)

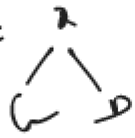
Alors $c(n_T) = 2 + c(n_G) + c(n_D)$

Donc $c(n_T) = 2 + 2n_G + 2n_D$
$= 2 \times (1 + n_G + n_D)$

Donc $c(n_T) = 2n_T$

Conclusion : Pour tout arbre binaire T, $c(n_T) = 2n_T$

Donc la complexité est en $\boxed{\Theta(n_T)}$.

(b) $T =$  $h(T) = 1 + \max(h(G), h(D))$

```
def ABhauteur(T):
    if estABvide(T):
        return 0
    else:
        return 1 + max(h(T.gauche), h(T.droit))
```

Soit $c(n)$ le nombre d'additions

Analyse : Si $c(n) = \alpha n$

la formule de récurrence est :

$$\underbrace{c(n_T)}_{\alpha n_T} = 1 + \underbrace{c(n_G)}_{\alpha n_G} + \underbrace{c(n_D)}_{\alpha n_D}$$

donc $\alpha n_T = 1 + \alpha n_G + \alpha n_D \quad (*)$

Quelle valeur de α pour que cette formule soit vraie ?

$n_T = 1 + n_G + n_D \quad (**)$

$(*)$ et $(**)$: $\alpha + \cancel{\alpha n_G} + \cancel{\alpha n_D} = 1 + \cancel{\alpha n_G} + \cancel{\alpha n_D}$: $\underline{\alpha = 1}$

Synthèse :

Mq $c(n_T) = n_T$ par induction structurelle sur T.

- Base : $c(0) = 0$ ✓
- Induction : Soit $T =$

On suppose que $\begin{cases} c(n_G) = n_G \\ c(n_D) = n_D \end{cases}$

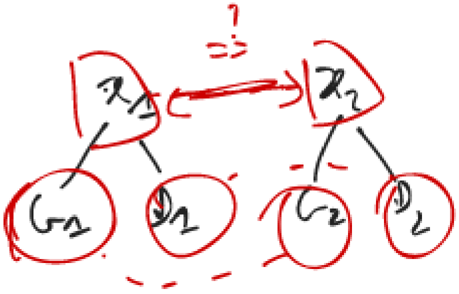
Alors, d'après la fonction : $c(n_T) = 1 + c(n_G) + c(n_D)$

Donc $c(n_T) = 1 + n_G + n_D$

Donc $c(n_T) = n_T$ //

. <u>Conclusion</u> : La propriété est vraie pour tout arbre binaire T.

2)
```
def ABegal(T1, T2):
    if estABvide(T1):
        if estABvide(T2):
            return True
        return False
    if ABvide(T2):
        return False
    return (T1.clef == T2.clef)
        and ABegal(T1.gauche, T2.gauche)
        and ABegal(T1.droit, T2.droit)
```



. <u>Meilleur cas</u> : les racines sont différentes, on ne fait <u>qu'une</u> comparaison car on renvoie directement que T_1 et T_2 sont différents

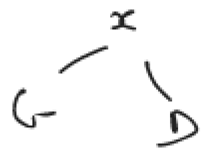
↳ Complexité $\Omega(1)$

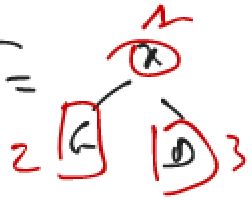
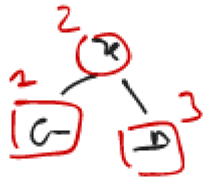
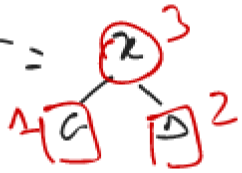
. <u>Pire cas</u> : On est obligé de tester tous les noeuds 1 par 1

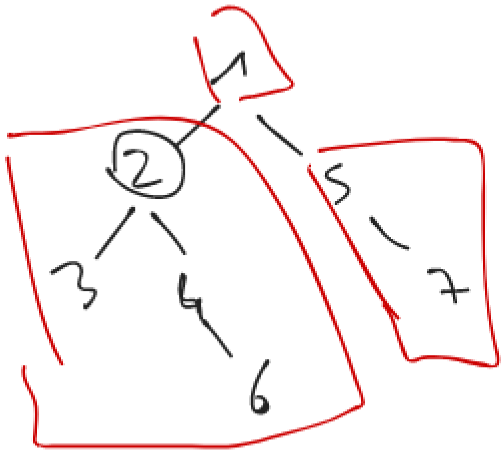
Donc tous les noeuds du plus petit des arbres

↳ Complexité en $O(\min(n_1, n_2))$

Exercice 3

$T = \emptyset$ ou 

- Parcours préfixe, infixe et suffixe de $\emptyset$: [ ]
- Parcours préfixe de $T =$  $P_T = [x] . P_G . P_D$
- Parcours infixe de $T =$  $I_T = I_G . [x] . I_D$
- Parcours suffixe de $T =$  $S_T = S_G . S_D . [x]$

$T =$ 

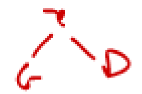
$P_T = (\boxed{1}, \underbrace{2, 3, 4, 6}_{I_G}, \underbrace{5, 7}_{I_D})$

$I_T = (3, 2, 4, 6, \boxed{1}, 5, 7)$

$S_T = (3, 6, 4, 2, 7, 5, \boxed{1})$

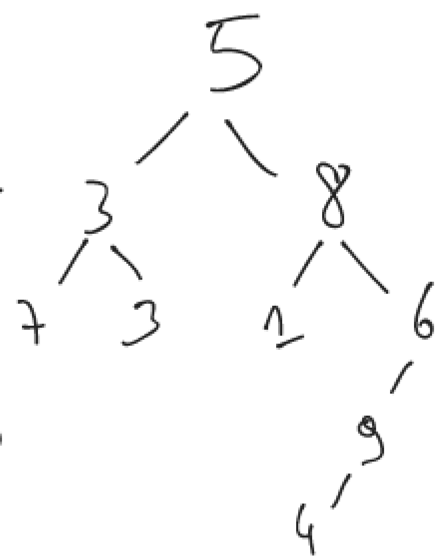
Q1

1) $P = (5 \mid \overbrace{3, 7, 2}^{G} \mid \overbrace{8, 1, 6, 9, 4}^{D})$ ← au hasard

$T =$ 

$P_T = [x] . P_G . P_D$

$T =$  $P_T = P$

$T =$ 

$P_T = P$

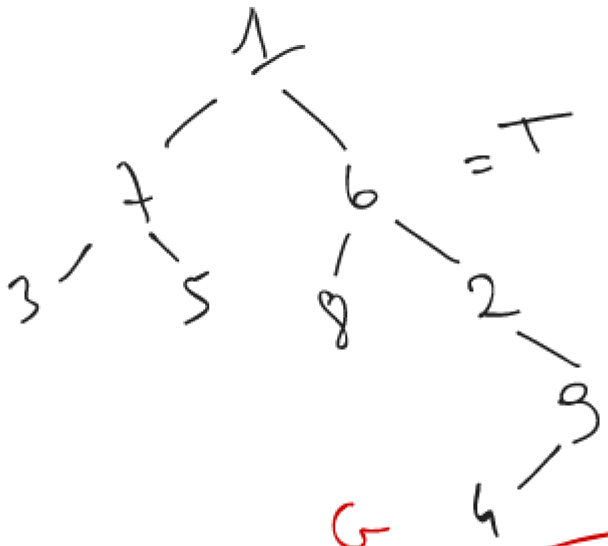
il y a plein de possibilités

2) $I = (\underbrace{3,7,5}_{G}, 1, \underbrace{8,6,2,4,9}_{D})$

au hasard

$T = $ x
G D

$I_T = I_G \cdot (x) \cdot I_D$



$= T$

Il y a plein de possibilités.

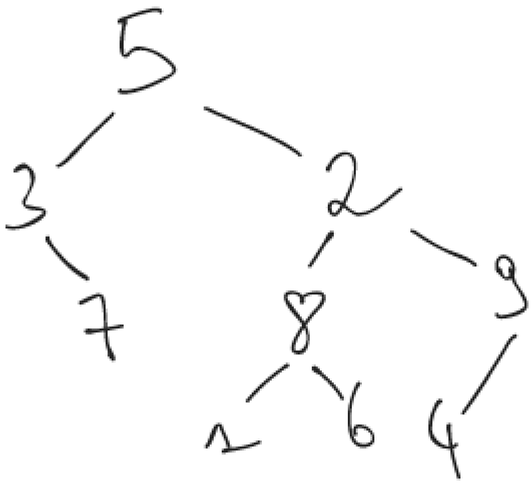
$I_T = I$.

3) $P = (5, \underbrace{3,7}_{G}, \underbrace{2, 8, 1, 6, 9, 4}_{D})$

$I = (\underbrace{3,7}_{G}, 5, \underbrace{1,8,6,2,4,9}_{D})$

$T = $ x
G D

- $P_T = [x] P_G P_D$
- $I_T = I_G (x) I_D$



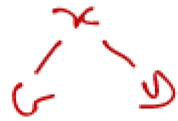
→ On n'a pas eu le choix pour la construction

Donc la solution est unique.

4) $P = (5, 3, 7, 2, 8, 1, 6, 9, 4)^{T_D}$

$I = (3, 7, 5, 4, 9, 2, 1, 8, 6)$

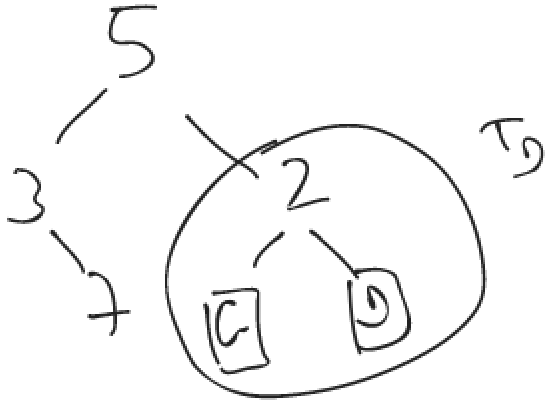
G $\quad T_D$

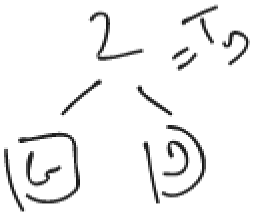
$T =$ 

$P_T = (x) . P_G . P_D$

$I_T = I_G \, (x) \, I_D$

Essayons de construire un arbre binaire à partir de P et I:



Si l'arbre  $= T_D$ peut être construit, alors par construction:

$P_{T_D} = (2, 8, 1, 6, 9, 4)$ (*) et $P_{T_D} = 2 . P_G . P_D$ (**)

$I_{T_D} = (4, 9, 2, 1, 8, 6)$ $\qquad I_{T_D} = I_G \; 2 \; I_D$

$I_G = (4, 9)$ donc G est composé des noeuds 4 et 9 donc P_G contient 4 et 9.

Impossible avec les relations (*) et (**).

Q2 P et I deux listes tq :

(pi1) les éléments de P : 2 à 2 distincts
(P ne contient pas 2x le même élément)

(pi2) $|P| = |I|$ et ont les éléments

$\left.\begin{array}{l} P \leadsto \text{parcours préfixe} \\ I \leadsto \text{parcours infixe} \end{array}\right\} \leadsto$ sur un même arbre).

1) $\Pi(n) =$ Si $|P| = n$, P et I vérifient (pi1) et (pi2)
et il existe T arbre binaire
tel que $\left| \begin{array}{l} P_T = P \\ I_T = I \end{array} \right.$

Alors T est unique

Montrons $\Pi(n)$ par récurrence forte sur $n \geq 0$.

· Base : $n = 0$, donc $P = []$
si $P_T = []$ alors T est l'arbre vide,
unique solution. ✓

· Induction : Soit $n > 0$, on suppose que $\forall m < n\ \Pi(m)$
Soit P et I tq $|P| = n$ et P et I vérifient (pi1) et (pi2)
On suppose qu'il existe T tq $\left| \begin{array}{l} P_T = P \\ I_T = I \end{array} \right.$

$|P| \geq 1$ donc T non vide donc $T =$

$P_T = (x) \cdot P_G \cdot P_D = P$)

$I_T = I_G \cdot (x) \cdot I_D = I$

. Mq G est unique grâce à l'HR.

(Pi1) : Par hypothèse (Pi1) sur P_T, les éléments de P_G sont 2 à 2 distincts

(Pi2) : P_G et I_G ont la même longueur par définition, et sont composés des mêmes éléments (parce que I existe)

$|P_G| < |P|$, donc on peut appliquer l'HR et G est unique

. De même, D est unique

. D'après le parcours préfixe, T = 

où x est le premier élément de P.

Donc T est unique → par le choix pour x

→ HR : G et D uniques

. Conclusion : La propriété est vraie pour tout arbre binaire T.

2) Hypothèse fondamentale : éléments 2 à 2 distincts

$T_1 =$  $T_2 =$  mais pourtant $P_{T_1} = P_{T_2} = I_{T_1} = I_{T_2}$

$= (1,1)$.

TD6 Exercice 6

Arbre binaire : $\emptyset$ ou 

G, D arbres binaires

Arbre binaire strict :

Arbre binaire → non vide
→ un noeud 0 ou 2 fils

⇒ T arbre binaire strict : $\left| \begin{array}{l} n(T) \geq 1 \\ n_1(T) = 0 \end{array} \right.$: $n_i(T) = n_2(T)$

Q1

T arbre binaire strict

$f(T) = n_i(T) + 1$

cf exercice 1 : $f(T) = n_2(T) + 1$

Or $n_2(T) = n_i(T)$ car T est strict (définition)

Donc $\boxed{f(T) = n_i(T) + 1}$

. $n(T) = n_i(T) + f(T)$

Donc $n(T) = n_i(T) + n_i(T) + 1$

Donc $\boxed{n(T) = 2n_i(T) + 1}$

. Si $n(T) = 100$, T peut-il être strict ?

Non car T strict ⇒ $n(T)$ impair

donc $n(T)$ pair ⇒ T n'est pas strict

Q2 Soit T arbre binaire non vide tq $f(T) = n_i(T) + 1$

$\rightarrow f(T) = n_2(T) + 1$ (à connaître)

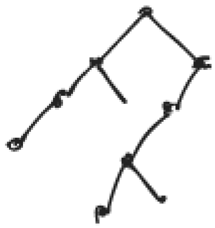
Donc $n_i(T) = n_2(T)$

$\rightarrow n_i(T) = n_1(T) + n_2(T)$ (à connaître)

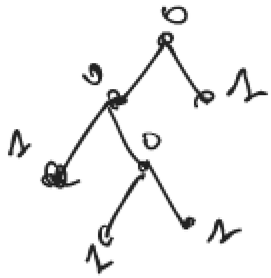
Donc $n_1(T) = 0$

Donc T est strict

arbre binaire



Q3 T :



$P(T)$: T arbre binaire strict étiqueté par $\{0,1\}$
avec noeud interne $\rightarrow 0$
feuille $\rightarrow 1$

Alors son parcours infixe est une liste $I_T = [\underbrace{1,0,1,\dots,0,1}_{n(T)}]$

. Base : $T = (x, \emptyset, \emptyset)$

x est une feuille donc $x = 1$

Donc $I_T = [1]$, Donc $P(T)$ est vérifié

. Induction :

Soit $T =$  un arbre strict

$I_T = [I_G, x, I_D]$, et G et D sont des arbres strict.

Par HI : $I_G = [1,0,1,\dots,0,1]$

$I_D = [1,0,1,\dots,0,1]$

$x = 0$ car c'est un noeud interne

Donc $I_T = [\underbrace{1,0,1,\dots,0,1}_{I_G}, \underbrace{0}_{x}, \underbrace{1,\dots,0,1}_{I_D}]$

Donc $P(T)$ est vrai

Conclusion : P(T) est vrai pour tout arbre binaire strict.