Epreuve finale 2022/2023 - Master TC

Département Informatique - Faculté FSEA - Université Oran1

Algorithmique Avancée et Complexité

Exercice 1 (10 pts)

Soit le tableau T=[1, 2, 0, 3, -1, 4]

Q1)(4p) Dérouler le tri rapide sur T.

Q2)(4p) Dérouler le tri par tas sur T.

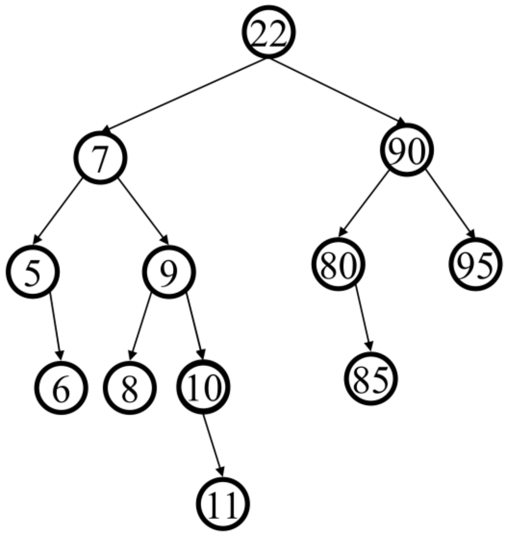
Soit l'algorithme CONSTRUIRETAS(A[1..n]) qui permet de forcer la propriété d'un tas sur un tableau A de taille n (i.e., la 1ère étape du tri par tas).

Q3)(1p) Proposer un invariant pour CONSTRUIRETAS, puis démontrer que l'algorithme est correct.

Q4)(1p) Donner et démontrer la complexité de CONSTRUIRETAS en exploitant l'égalité

$$\sum_{j=0..+\infty} \frac{j}{2^j} = 2.$$

Exercice 2 (8 pts) Soit la structure AVL d'un arbre binaire de recherche A.



Q1)(3p) Démontrer que cet arbre est un arbre AVL.

Q2)(3p) Illustrer toutes les étapes pour supprimer la clé 95 et conserver la structure AVL ?

Q3)(2p) Démontrer la complexité de l'insertion de *n* clés dans un AVL vide.

Exercice 3 (2 pts)

Soit le problème de la recherche de la plus longue sous-séquence commune entre deux séquences *x* et *y*. Soit l'impression de la plus longue sous-séquence commune par programmation dynamique :

```
IMPRIMER-PLSC(b, X, i, j)
si i = 0 ou j = 0
    alors retourner
si b[i, j] = « ↖ »
    alors IMPRIMER-PLSC(b, X, i − 1, j − 1)
          imprimer x_i
sinon si b[i, j] = « ↑ »
    alors IMPRIMER-PLSC(b, X, i − 1, j)
sinon IMPRIMER-PLSC(b, X, i, j − 1)
```

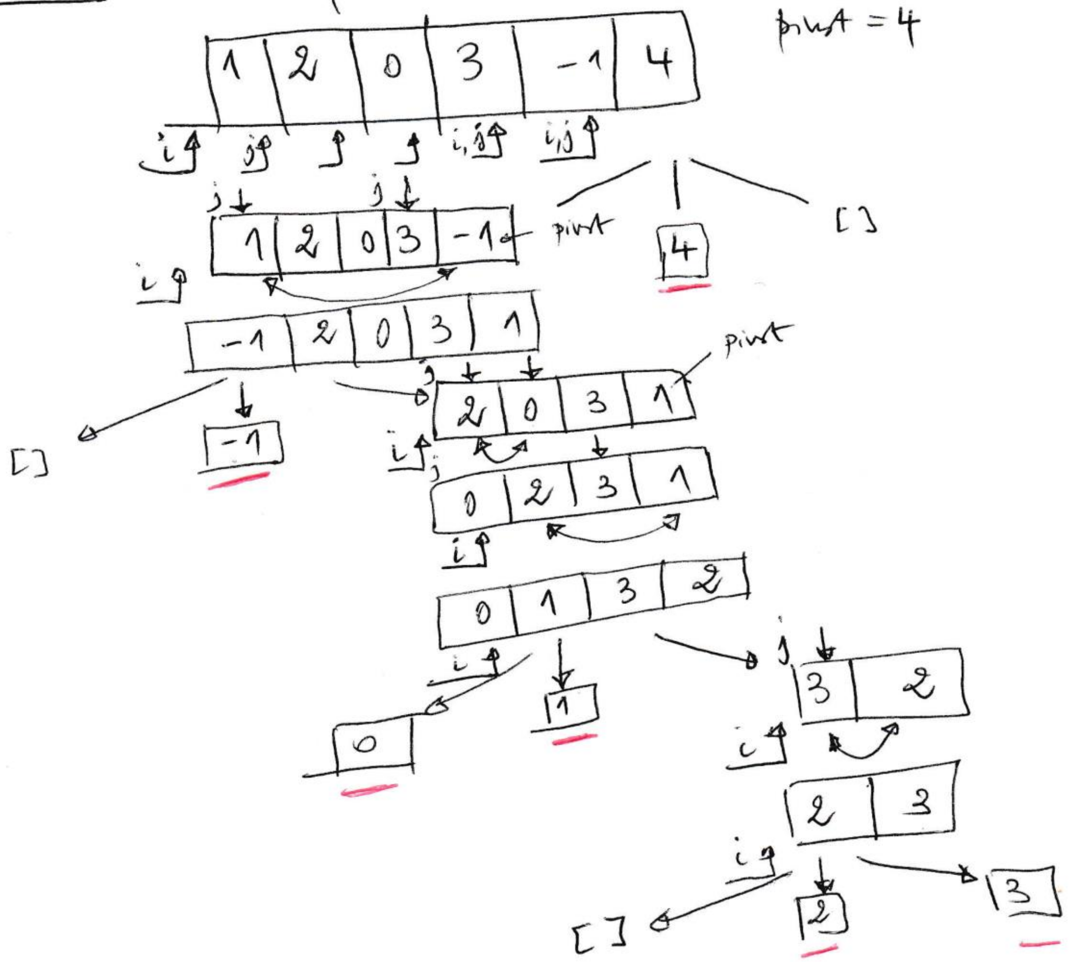
avec

$$c[i,j] = \begin{cases} 0 & \text{si i=0 ou j=0,} \\ \text{c[i-1, j-1]} + 1 & \text{si i,j}>0\text{, et } x_i = y_j, \\ \max(\text{c[i, j-1], c[i-1, j]}) & \text{si i,j}>0\text{, et } x_i \neq y_j. \end{cases}$$

Q1) (2p) Proposer un algorithme pour imprimer le nombre de sous-séquences communes les plus longues.

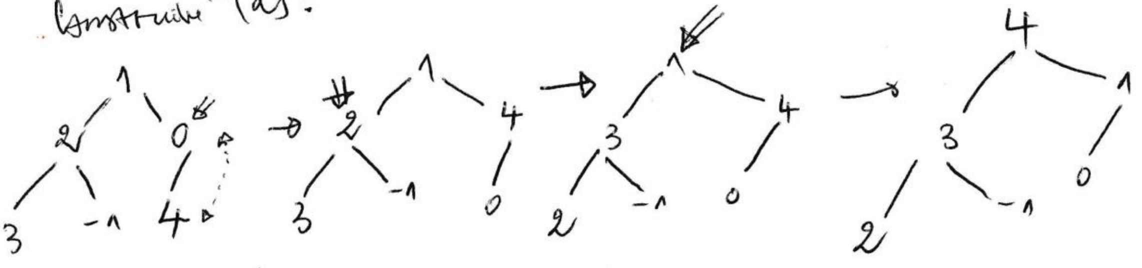
Exercice n°1 | Q_1) Tri rapide

pivot = 4



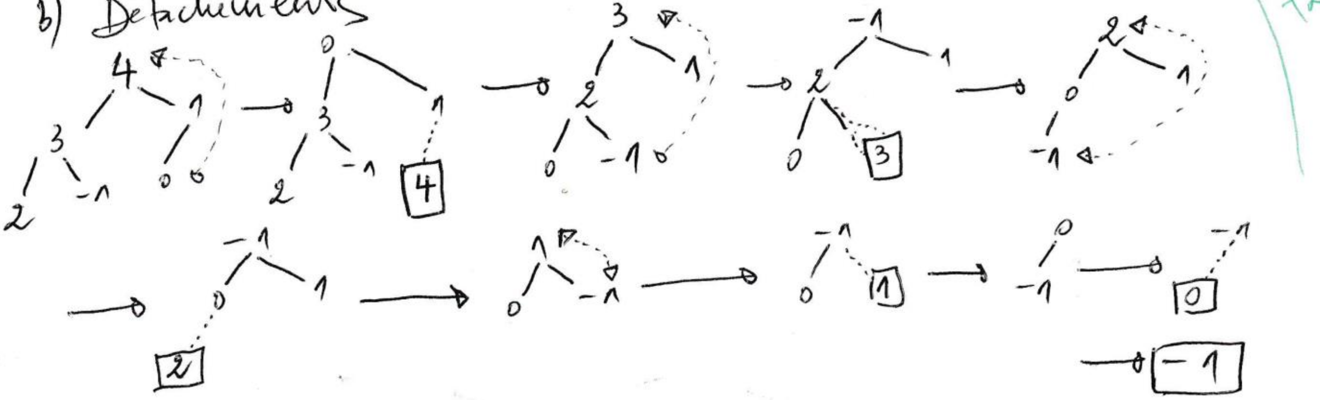
Q_2) a) Tri par tas

Construire tas :



+2

b) Détachements



+2

Q3) ConstruireTas (A)

```
for i ← ⌊n/2⌋ to 1 do
    Entasser(A, i)
endfor
```

Q4) Invariant de ConstruireTas :

Invariant proposé : $I(i)$: Tous les éléments de $A[i+1..n]$ respectent la propriété du tas.

<u>Preuve</u> : - À l'entrée de la boucle $i = \lfloor n/2 \rfloor$, les éléments $A[\lfloor n/2 \rfloor, \ldots n]$ sont des feuilles, donc respectent la propriété de tas.

- Supposons que la propriété est vraie à i, démontrons qu'elle restera vraie à $i+1$.

↳ Après l'itération i, l'algorithme aurait entasser l'élément $A[i] \Rightarrow A[i..n]$ respectent le tas.

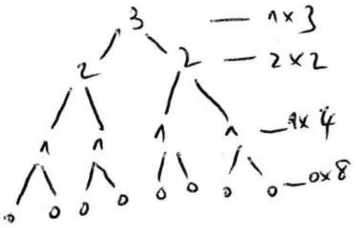
D'où $I(i+1)$ à l'entrée de l'itération $i+1$.

⟹ L'invariant I est donc valide.

Terminaison : l'algorithme sort de la boucle avec $i=0$.

En remplaçant dans l'invariant $I(0) \Rightarrow A[1,..n]$ respectent la propriété de Tas. D'où la <u>correction</u>.

Q5) <u>Preuve</u> Soit le coût des entassements illustré



À partir de l'illustration, on peut généraliser soit $T(n)$, où T est la complexité de l'algo.

$$T(n) = \sum_{j=0..h} j \times 2^{h-j} = \sum_{j=0..h} j \cdot \frac{2^h}{2^j} = 2^h \cdot \sum_{j=0..h} \frac{j}{2^j}$$

$$\leq 2^h \cdot \sum_{j=0..\infty} \frac{j}{2^j} = 2^h \cdot 2 = 2^{h+1}$$

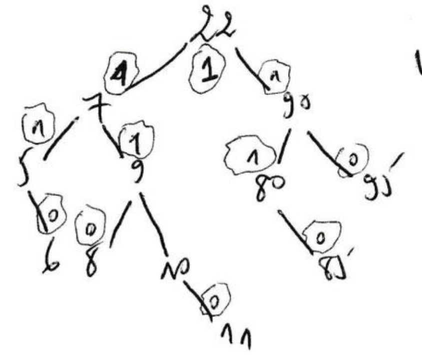
Dans un Tas complet $n = 2^{h+1} - 1 \Rightarrow T(n) \leq n+1$

Nous avons aussi $T(n) \geq \frac{n}{2}$ à partir du nombre d'itération $n/2$.

$$\Rightarrow T(n) = \Phi(n)$$

Exercice 2

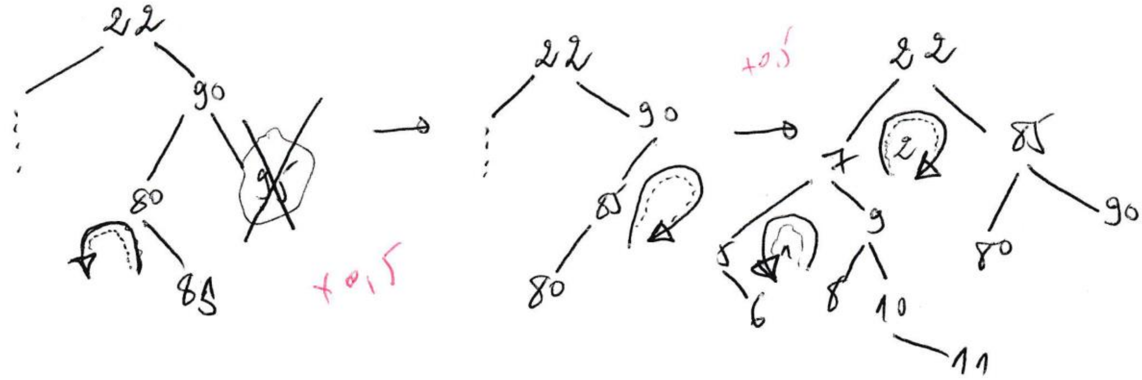
Q1)

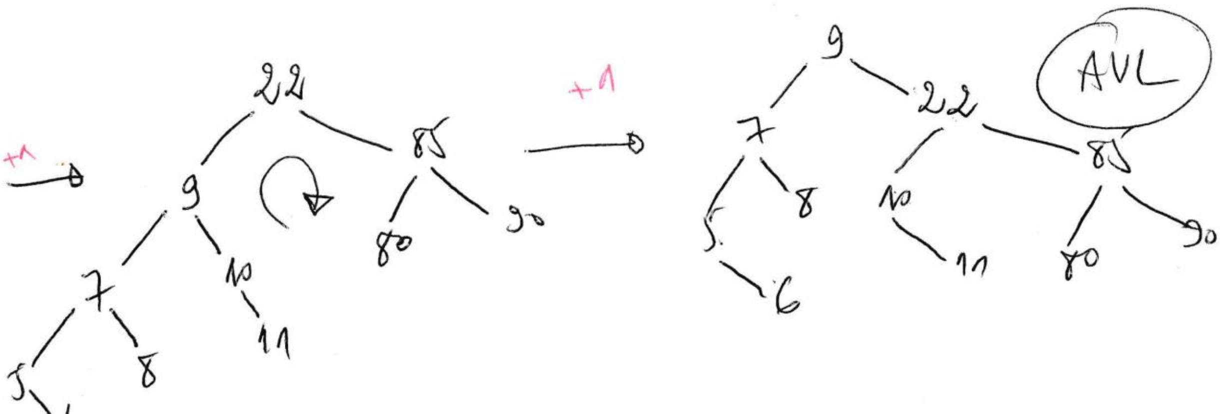


La différence de hauteur des deux sous-arbres de tous les noeuds est toujours ≤ 1

$\Rightarrow$ D'où AVL.

Q2)





Q3) Soit T la complexité de l'insertion de n clés.

$$T(n) = \sum_{i=1..n} \log(i).$$

On peut démontrer que $\lfloor n/2 \rfloor \log(n/2) \leq \sum_{i=1..n} \log(i) \leq n \log(n)$

D'où $T(n) = \Theta(n \log(n))$ (voir exercice 2.7.4 – Q6).

Execice 3

Q1

```
imprimer_nbre_PLSC(b, X, i, j)
    if i=0 or j=0 then
        return 1;
    else if b[i,j] = "↖" then
        return imprimer_nbre_PLSC(b, c, X, i-1, j-1)
    else
        if c[i, j-1] = c[i-1, j] then
            a ← imprimer_nbre_PLSC(b, c, X, i-1, j)
            b ← ---------------, i, j-1)
            return a+b
        else if b[i,j] = ↑ then
            return imprimer_nbre_PLSC( ---, i-1, j)
        else return imprimer_nbre_PLSC( ---, i, j-1)
        endif
    endif
endif
```