Exercice 1 (sur 8 pts). *Du cours*

1. Pour chaque affirmation, indiquer si elle est **vraie ou fausse** en **justifiant**.

i. Dans un tas, le plus petit élément est une feuille.

ii. Dans un arbre binaire de recherche, le plus petit élément est une feuille.

iii. Un tableau trié par ordre décroissant est un tas.

2. Parmi les tableaux suivants, lesquels sont des tas : $T_1 = [7,3,6,1,2,4,5]$, $T_2 = [7,6,3,1,2,4,5]$, $T_3 = [7,6,5,1,2,3,4]$?

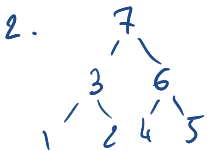
1.i. VRAI car le tas est décroissant.

ii. FAUX

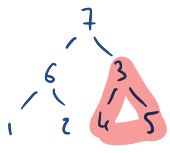


iii. VRAI car $T[\lfloor \frac{i-1}{2} \rfloor] \geq T[i]$ pour tout $i > 0$.

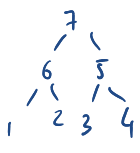
2.



ok

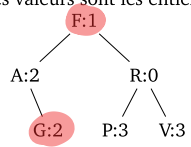


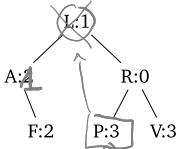
Non



ok

3. On considère les deux arbres binaires ci-dessous, dans lesquels les clés sont les lettres et les valeurs sont les entiers.





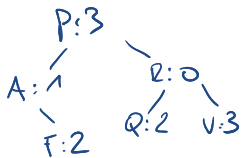
i. Lequel des deux est un arbre binaire de recherche ? *Donner explicitement le problème pour celui qui n'en est pas un.*

ii. On note D le dictionnaire représenté par l'arbre qui est un ABR, et on applique les opérations suivantes : $D_{[Q]} \leftarrow 2$; $D_{[A]} \leftarrow 1$; SUPPRIMER(D, L). Dessiner l'arbre obtenu après avoir appliqué les trois opérations.

4. Qu'est-ce qu'une famille universelle de fonctions de hachage ?

3.i. Celui de gauche n'est pas un ABR car G est dans le sous-arbre gauche de F.

ii.



4. Une famille H de fonctions de hachage $h : U \to \{0, \dots, m-1\}$ est universelle si pour tout $k_1 \neq k_2 \in U$ $\Pr_{h \in H}[h(k_1) = h(k_2)] \leq 1/m$.

Exercice 2 (sur 6 pts). *Structure bizarre*

1. Justifier qu'en réalisant le TAD Liste bizarre à partir du TAD Liste du cours, les complexités de NVLISTEBIZARRE et AJOUTER sont celles annoncées. *On ne demande rien pour* RÉDUIRE.
2. Montrer que le coût amorti d'une opération AJOUTER est borné par une constante.
3. Montrer que le coût amorti d'une opération RÉDUIRE est borné par une constante.
4. Que peut-on en déduire sur le coût total de k opérations AJOUTER et RÉDUIRE ?

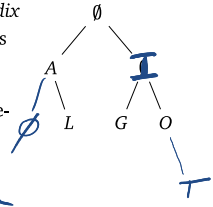
1. NVLISTEBIZARRE et AJOUTER sont exactement les mêmes opérations que NVLISTE et AJOUTER du TAD "Liste".

2. Coût amorti d'AJOUTER : $a_i = \Theta(1) + 2cn_i - 2c(n_i - 1)$
$= \Theta(1) + 2c = \Theta(1)$

3. RÉDUIRE : $a_i \leq c \cdot n_{i-1} + 2c \cdot n_i - 2cn_{i-1}$
et $n_i = \left\lfloor \frac{n_{i-1}}{2} \right\rfloor$ donc $a_i \leq c \cdot n_{i-1} + 8c \frac{n_{i-1}}{8} - 2cn_{i-1} \leq 0$

4. On conclut que le coût de k opérations est $\Theta(k)$: $\sum_{i=0}^{n-1} c_i = \sum_{i=0}^{n-1} a_i + \phi_{i-1} - \phi_i = \Theta(k) + \phi_0$

Exercice 3 (sur 7 pts). *Arbre radix*



1. Dessiner l'arbre radix obtenu si on insère les couples $(1, I)$ et $(111, T)$ dans l'arbre ci-dessus.
2. **i.** Pourquoi peut-on supposer qu'aucune feuille n'est vide ?
   ii. Exprimer la hauteur de l'arbre en fonction de la longueur des clés contenues dans le dictionnaire.

2. i. Une feuille vide ne contient aucune valeur et n'a pas d'enfants. Si on la supprime le nouvel arbre radix représente le même dictionnaire.

ii. Hauteur de l'arbre = max des longueurs des clés du dictionnaire

3. Proposer une réalisation en pseudo-code de l'opération RECHERCHER(D, c) du TAD dictionnaire et analyser sa complexité. *Rappel : si c n'est pas dans D, il faut renvoyer une erreur.*

4. Décrire un algorithme, basé sur un parcours d'arbre radix, qui prend en entrée un tableau de mots binaires et les affiche dans l'ordre lexicographique, en temps $O(N)$ où N est la somme des longueurs des mots du tableau. *Préciser le parcours d'arbre utilisé. L'ordre lexicographique sur les mots binaires est* $0, 00, 000, \ldots, 01, 010, 0100, \ldots, 011, 0110, \ldots, 1, 10, 100, \ldots, 11, 110, \ldots$ *Formellement,* $w < w'$ *s'il existe* j *tel que* $w_{[0,j-1]} = w'_{[0,j-1]}$ *et soit* w *est de taille* j*, soit* $w_{[j]} = 0$ *et* $w'_{[j]} = 1$.

3 RECHERCHER (Δ, c) : —> $\Theta(\#c)$

Pour i = 0 à #c-1 :

| Si $c_{[i]} = 0$: $\Delta \leftarrow$ ENFANTG(Δ)
| Sinon : $\Delta \leftarrow$ ENFANTD(Δ)
| Si Δ est l'arbre vide : Renvoyer une erreur

Si la racine de Δ est le nœud vide : Renvoyer une erreur

Renvoyer la valeur de la racine de Δ.