Nom :_____

Code permanent : _____

Numéro de place : _____

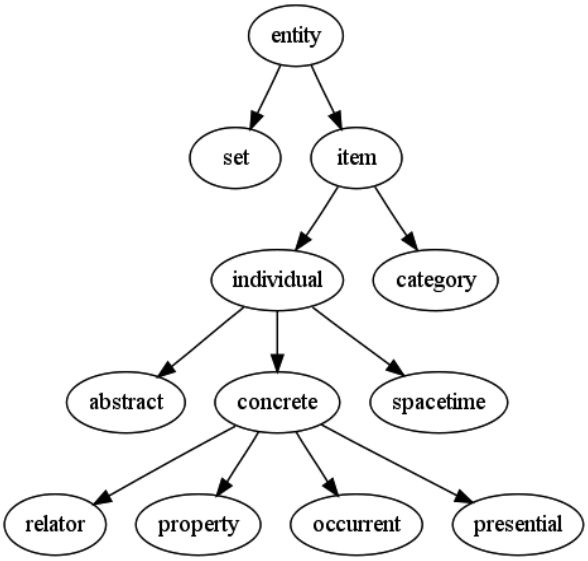
Directives pédagogiques :

- Inscrivez votre nom, prénom, code permanent et numéro de votre place.
- Lisez attentivement toutes les questions et **répondez directement sur le questionnaire**.
- Seule l'utilisation d'un crayon est permise, **aucune documentation, calculatrice, téléphone cellulaire, ordinateur, ou autre objet permis**.
- Cet examen contient 10 questions pour 160 points au total.
- Tout point au-dessus de 150 sera considéré comme bonus. Cependant, faites attention au temps car le barème est établi à 1 point par minute. Vous pouvez laisser tomber 10 points sans être pénalisé.
- Cet examen contient 19 pages, incluant 3 pages détachables à la fin pour vos brouillons.
- Pour les questions à développement, **écrivez lisiblement et détaillez vos réponses**.
- Vous avez 150 minutes pour compléter cet examen.

BONNE CHANCE !

1	/ 10
2	/ 10
3	/ 15
4	/ 15
5	/ 25
6	/ 15
7	/ 15
8	/ 20
9	/10
10	/25
Total	/150

1. (10) Soit l'arbre taxinomique de l'ontologie formelle générale :



a) (1) Quel noeud est la racine ?

entity

b) (2) Quels sont les noeuds internes ?

entity, item, individual, concrete

c) (1) Combien de descendants le noeud « individual » possède-t-il ?

7

d) (1) Combien d'ancêtres le noeud « spacetime » possède-t-il ?

3

e) (1) Combien de frères et soeurs le noeud « item » possède-t-il ?

1

f) (1) Quels sont les noeuds du sous-arbre dont la racine est « individual » ?

individual, abstract, concrete, spacetime, relator, property, occurrent, presential

g) (2) Quelle est la profondeur du noeud « category » ?

2

h) (1) Quelle est la hauteur de cet arbre ?

4

2. (10) Dans quel ordre les noeuds de l'arbre du numéro (1) seront visités par un parcours :

a) (5) préfixé (preorder)

entity, set, item, individual, abstract, concrete, relator, property, occurrent, presential, spacetime, category

b) (5) postfixé (postorder)

set, abstract, relator, property, occurrent, presential, concrete, spacetime, individual, category, item, entity

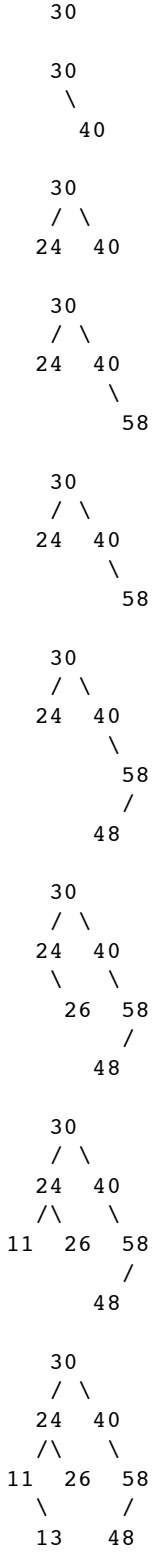
3. (15) Montrez comment implanter une pile (opérations `push` et `pop`) en utilisant une queue avec priorités (clés min prioritaires et opérations `enqueue` et `dequeue`) et seulement une autre variable de type entier. (hint: les éléments d'une queue avec priorités sont composés d'une clé et d'une valeur).

```
# solution avec un entier
Stack:

    q = PriorityQueue()
    counter = larger int

    push( v ):
            q.enqueue( ( counter— , v ) )

    pop( ):
            return q.dequeue()
```

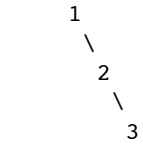
```
# solution sans même un entier
Stack:

    q = PriorityQueue()

    push( v ):
            q.enqueue( ( time.time() * -1, v ) )

    pop( ):
            return q.dequeue()
```

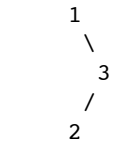
4. (15) Considérer un arbre binaire de recherche (ABR).

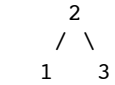
a) (8) Dessinez l'ABR après chaque insertion des clés 30, 40, 24, 58, 48, 26, 11, 13 (dans cet ordre).

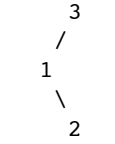


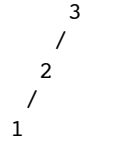
b) (7) Dessinez tous les ABR possibles contenant les clés 1, 2 et 3.









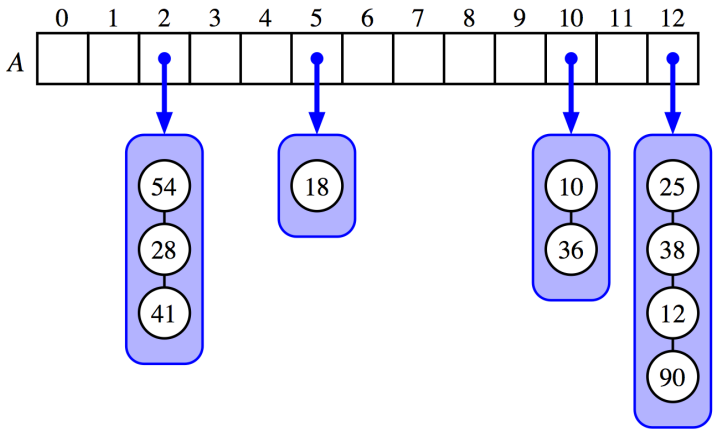


5. (20) Donnez étape par étape en dessinant les arbres AVL résultant des opérations suivantes :

   a) (10) Insertion des clés suivantes et dans cet ordre dans un arbre AVL vide au départ : 14, 17, 11, 7, 53, 4, 13, 12 et 8.

b) (10) De l'arbre AVL que vous avez obtenu en (a), suppression des clés suivantes et dans cet ordre : 53, 11 et 8.

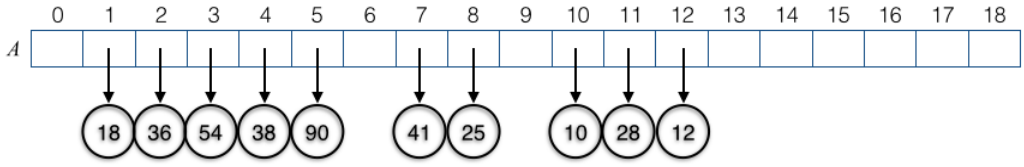
6. (15) Considérez la table de hachage avec chaînage externe suivante :



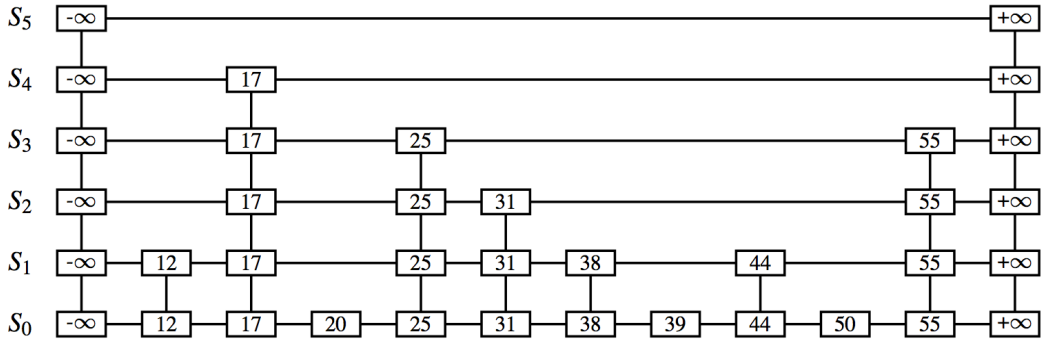
a) (5) Quel est le pire cas en temps d'exécution pour insérer *n* éléments dans cette table initialement vide et avec une résolution des collisions par listes simplement chaînées ?

n x temps pour insérer dans une liste simplement chaînée O(n)
=> $O(n^2)$

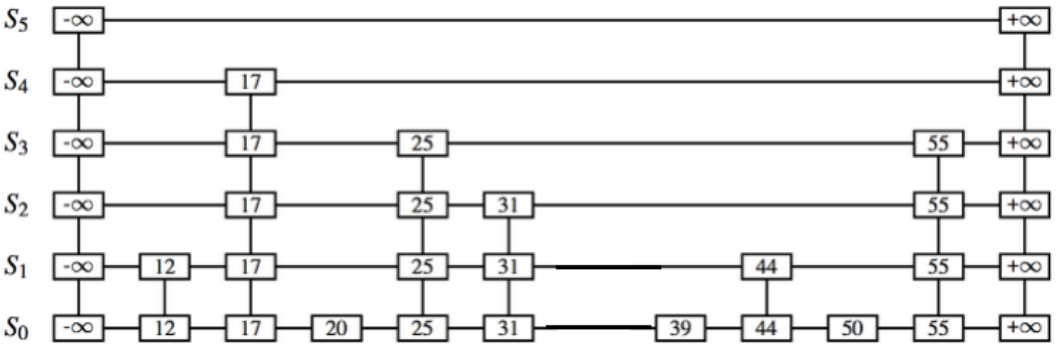
b) (10) Dessiner la nouvelle table après re-hachage dans une table de taille 19 avec une nouvelle fonction de hachage `h(k) = k mod 17`.



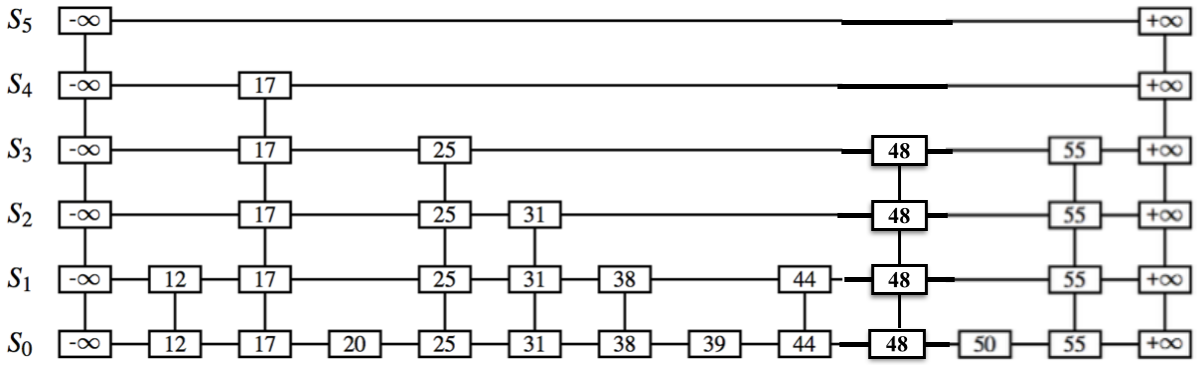
7. (15) Considérez la « skip list » suivante :



a) (5) Dessinez la skip list résultante après l'opération `del S[38]`.



b) (10) À partir de la « skip list » ci-haut (celle de départ et non pas celle que vous avez obtenue en (a)), dessinez la « skip list » résultante après l'opération `S[48] = 'x'` si les résultats du pile-ou-face sont `FACE`, `FACE`, `FACE`, `PILE`, `FACE`.



8. (20) Considérez les arbres rouge-noir (ARN). Pour chaque énoncé suivant, donnez une justification s'il est vrai ou un contre exemple s'il est faux.

a) (5) Un sous-arbre d'un ARN est lui aussi un ARN.

Faux. La racine d'un sous-arbre RN peut être rouge, mais pas la racine d'un ARN.

b) (5) Un noeud d'un ARN qui n'a pas de frère (ou soeur) est rouge.

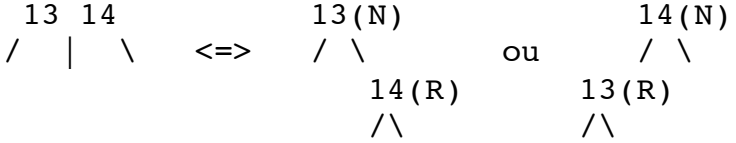
Vrai. Sinon la propriété de profondeur de tous les noeuds de zéro ou un enfant ne serait pas respectée (black depth).

c) (5) Chaque ARN possède un arbre (2,4) associé unique.

Vrai. Un ARN est avant tout un arbre (2,4) et pour les 2-noeud et 4-noeud il existe une seule représentation équivalente RN, alors que pour le 3-noeud il existe deux représentations équivalentes RN.

d) (5) Chaque arbre (2,4) possède un ARN associé unique.

Faux. Il existe 2 représentations RN pour le 3-noeud d'un arbre (2,4).

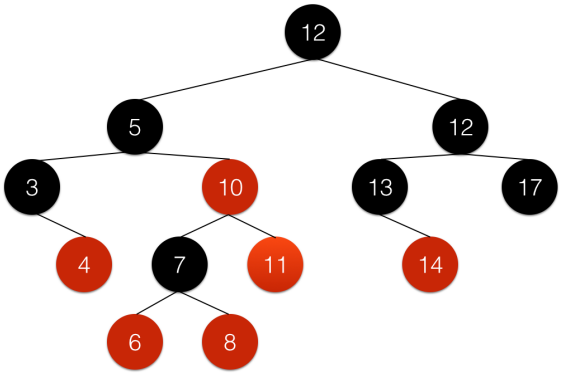


9. (10) Expliquez pourquoi le parcours dans l'ordre (inorder) des clés d'un ABR, arbre AVL, arbre « splay » ou ARN donne la même chose.

Le parcours dans l'ordre (inorder) de tout arbre binaire de recherche (ABR), gauche-racine-droite, parcourt les clés dans l'ordre croissant de par la structure même d'un ABR et les arbres AVL, splay et Rouge-Noir sont avant tout des ABR.

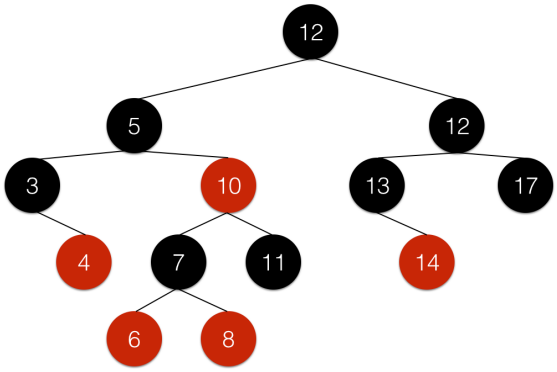
10. (25) Dites si les ARN suivants sont valides et dans chaque cas s'il ne l'est pas quelle propriété est violée.

a) (5)



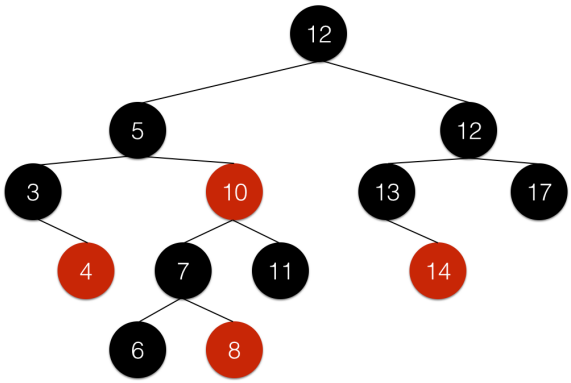
Non valide. Les enfants d'un noeud rouge doivent être noirs, ce qui n'est pas le cas pour le noeud 10 qui a un enfant rouge, 11. ***La propriété rouge (red property) est violée***.

b) (5)



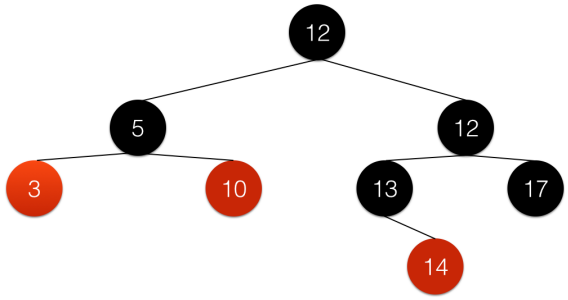
Valide.

c) (5)



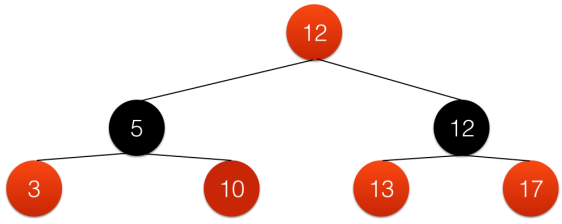
Non valide. La profondeur noire (black depth) du noeud 6 qui n'a aucun enfant est 4, soit un de plus que tous les autres noeuds qui n'ont aucun ou un enfant, 13, 14, 17, 3, 4, 11 et 8. ***La propriété de profondeur est violée***.

d) (5)



Non valide. La profondeur noire (black depth) des noeuds 3 et 10 qui n'ont aucun enfant est 2, soit un de moins que les noeuds 14 et 17. ***La propriété de profondeur est violée***.

e) (5)



Non valide. ***La propriété que la racine d'un arbre rouge-noir doit être noire est violée***.

Brouillon :

Brouillon :

Brouillon :