

Exercice sur les arbres binaires de recherche

Voici une liste aléatoire de 15 éléments. Notez que vous pouvez faire cet exercice en prenant une autre liste aléatoire évidemment, il y a peu de chances que vous obteniez le même résultat.

25	60	35	10	5	20	65	45	70	40	50	55	30	15
----	----	----	----	---	----	----	----	----	----	----	----	----	----

On s'intéresse aux arbres binaires de recherche.

A-Rappelez les propriétés des arbres binaires de recherche.

B-Rappelez ce qu'est l'opération d'adjonction aux feuilles.

C-Construire l'arbre binaire de recherche par adjonction des valeurs aux feuilles, dans l'ordre de la liste. On s'attachera particulièrement à expliquer le raisonnement.

D-Donner la liste infixée de l'arbre obtenu, en justifiant votre raisonnement. Quelle propriété a-t-elle Est-ce toujours le cas

E-Rappelez ce qu'est l'opération d'adjonction à la racine.

F-Construire l'arbre binaire de recherche par adjonction des valeurs à la racine, dans l'ordre de la liste initiale. On s'attachera particulièrement à expliquer le raisonnement.

G-Donner la liste préfixée de l'arbre obtenu, en justifiant votre raisonnement.

H-Donner l'arbre obtenu par adjonction des valeurs aux feuilles, dans l'ordre de la liste préfixée que vous venez d'obtenir. Quelle propriété a-t-il vis à vis de l'arbre obtenu à la question F En est-il toujours ainsi

I-Rappelez ce qu'est l'opération de suppression.

J-Donner l'arbre obtenu par suppression de 30 dans l'arbre de la question H, en justifiant votre raisonnement.

Corrigé

Question A.

Un arbre binaire de recherche est tel que tout nœud a une clé supérieure à celles des nœuds de son sous arbre gauche et inférieure à celles des nœuds de son sous arbre droit. On peut encore dire que la clé d'un nœud est comprise entre la plus grande clé de son sous arbre gauche et la plus petite clé de son sous arbre droit. En d'autres termes, pour tout nœud o , on a :

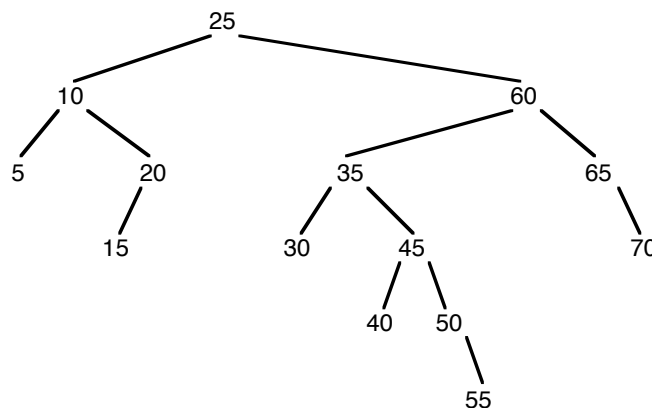
$$(g(o) \neq \emptyset \square \max(g(o)) \leq \text{la_clé}(o)) \square (d(o) \neq \emptyset \square \text{la_clé}(o) \leq \min(d(o)))$$

L'intérêt des arbres binaires de recherche est que la complexité moyenne de la recherche d'un nœud est proportionnelle à la hauteur moyenne des nœuds de l'arbre.

Question B.

L'adjonction aux feuilles consiste à rechercher où devrait être la feuille si elle était déjà dans l'arbre et à la mettre à cet endroit.

Question C.



25 est introduit dans un arbre vide. 60 étant plus grand que 25 est mis à droite de 25. 35 étant plus grand que 25 et plus petit que 60 est mis à gauche de 60. 10 est plus petit que 25 est mis à gauche de 25. 5 est plus petit que 25 et plus petit que 10, il est mis à gauche de 10. 20 est plus petit que 25 et plus grand que 10, il est mis à droite de 10. 65 est plus grand que 25 et plus grand que 60, il est mis à droite de 60. 45 est plus grand que 25, plus petit que 60 et plus grand que 35, il est mis à droite de 35. 70 est plus grand que 25, plus grand que 60 et plus grand que 65. Il est mis à droite de 65. 40 est plus grand que 25, plus petit que 60, plus grand que 35 et plus petit que 45, il est mis à gauche de 45. 50 est plus grand que 25, plus petit que 60, plus grand que 35 et plus grand que 45, il est mis à droite de 45. 55 est plus grand que 25, plus petit que 60, plus grand que 35, plus grand que 45, plus grand que 50, il est mis à droite de 50. 30 est plus grand que 25, plus petit que 60 et plus petit que 35, il est mis à gauche de 35. Enfin, 15 est plus petit que 25, plus grand que 10 et plus petit que 20, il est mis à gauche de 20. On obtient l'arbre suivant.

Question D.

La liste infixée de l'arbre est obtenue en faisant une exploration à main gauche, et en sortant le nœud après l'exploration de son sous arbre gauche et avant l'exploration de son sous arbre droit. C'est donc la suivante:

5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70

Cette liste est une liste ordonnée. C'est une des propriétés d'un arbre binaire de recherche que sa liste infixée soit une liste triée, qui découle directement de la propriété énoncée dans la question A.

Question E.

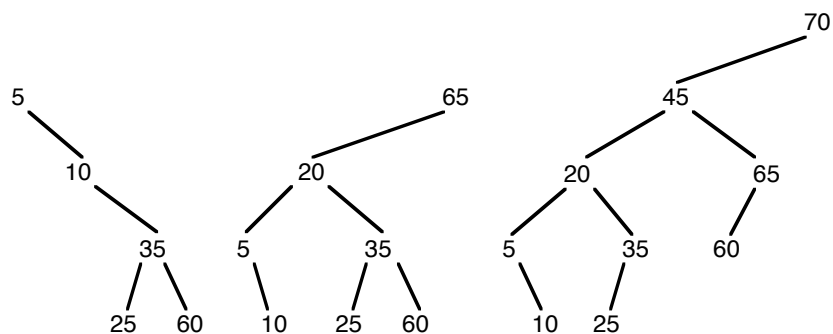
L'adjonction à la racine consiste à reconstruire l'arbre binaire de recherche en mettant la nouvelle valeur à la racine. En parcourant l'arbre initial depuis sa racine et selon un chemin guidé par la valeur ajoutée, on va placer successivement chaque nœud rencontré avec l'un de ses sous arbres gauche ou droit dans l'arbre résultat.

Question F.

Initialement, l'arbre contient 25. L'adjonction de 60 conduira à mettre 25 (<60) à gauche de 60. L'adjonction de 35 conduit d'abord à mettre 60 (>35) et son sous arbre droit à droite de 35 et à mettre ensuite 25 (<35) à gauche de 35. L'adjonction de 10 conduit d'abord à mettre 35 (>10) et son sous arbre droit à droite de 10, puis à mettre 25 (>10) comme sous arbre gauche de 35. L'ajout de 5 conduit à mettre 10 (>5) et son sous arbre droit à droite de 5. On obtient le premier arbre de la suite.

L'ajout de 20 conduit à mettre 5 (<20) et son sous arbre gauche à gauche de 20, puis 10 (<20) et son sous arbre gauche à droite de 5, puis 35 (>20) et son sous arbre droit à droite de 20 et enfin 25 (>20) et son sous arbre droit à gauche de 35. L'ajout de 65 conduit à mettre 20 (<65) et son sous arbre gauche à gauche de 65, puis 35 (<65) et son sous arbre gauche à droite de 20 et enfin 60 (<65) et son sous arbre gauche à droite de 35. On obtient le deuxième arbre de la suite.

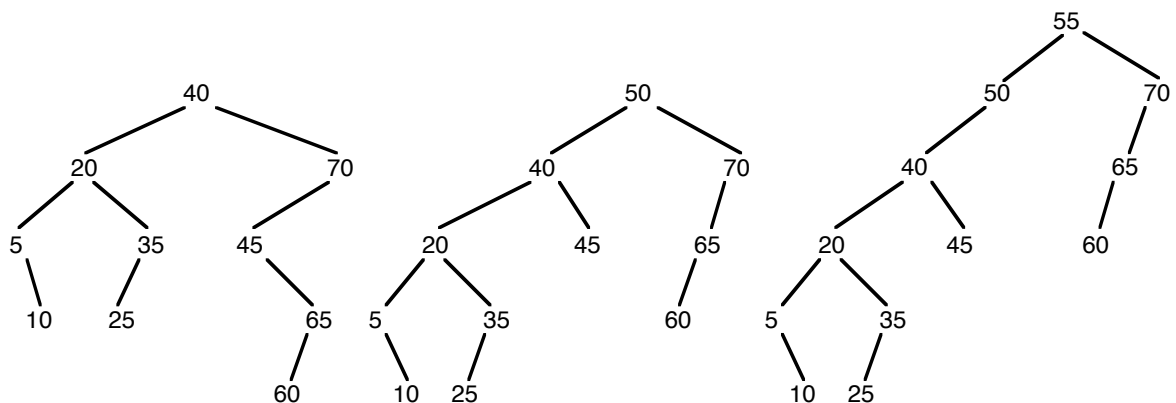
L'ajout de 45 conduit à mettre 65 (>45) et son arbre droit à droite de 45, puis 20 (<45) et son arbre gauche à gauche de 45, puis 35 (<45) et son arbre gauche à droite de 20 et enfin 60 (>45) et son arbre droit à gauche de 65. L'ajout de 70 conduit à mettre 45 (<70) et son arbre gauche à gauche de 70, puis 65 (<70) et son arbre gauche à droite de 45. On obtient le troisième arbre de la suite.



L'ajout de 40 conduit à mettre 70 (>40) et son sous arbre droit à droite de 40, puis 45 (>40) et son sous arbre droit à gauche de 70, puis 20 (<40) et son sous arbre gauche à gauche de 40 et enfin 35 (<40) et son sous arbre gauche à droite de 20. On obtient le premier arbre de la suite ci-dessous.

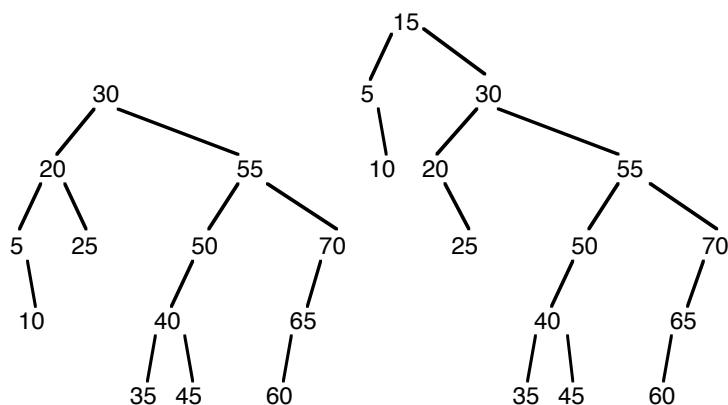
L'ajout de 50 conduit à mettre 40 (<50) et son sous arbre gauche à gauche de 50, puis 70 (>50) et son sous arbre droit à droite de 40, puis 45 (<50) et son sous arbre gauche à droite de 40, puis 65 (>50) et son sous arbre droit à gauche de 70 et enfin 60 (>50 et son sous arbre droit à gauche de 65. On obtient le deuxième arbre de la suite ci-dessous.

L'ajout de 55 conduit à mettre 50 (<55) et son arbre gauche à droite de 55, puis 70 (>55) et son arbre droit à droite de 55, puis 65 (>55) et son arbre droit à gauche de 70 et enfin 60 (>55) et son arbre droit à gauche de 65. On obtient le troisième arbre.



L'ajout de 30 conduit à mettre 55 (>30) et son sous arbre droit à droite de 30, puis 50 (>30) et son sous arbre droit à gauche de 55, puis 40 (>30) et son sous arbre droit à gauche de 50, puis 20 (<30) et son sous arbre gauche à gauche de 30, puis 35 (>30) et son sous arbre droit à gauche de 40 et enfin 25 (<30) et son sous arbre gauche à droite de 20. On obtient le premier arbre de la liste ci-dessous.

L'ajout de 15 conduit à mettre 30 (>15) et son sous arbre droit à droite de 15, puis 20 (>15) et son sous arbre droit à gauche de 30, puis 5 (<15) et son sous arbre gauche à gauche de 15 et enfin 10 (<15) et son sous arbre gauche à droite de 5. On obtient le deuxième arbre de la suite ci-dessous.



Question G.

La liste préfixée s'obtient en faisant l'exploration à main gauche et en sortant le nœud avant ses descendants, c'est-à-dire lorsque l'on descend sur ce nœud depuis son père. La liste préfixée est donc la suivante:

15, 5, 10, 30, 20, 25, 55, 50, 40, 35, 45, 70, 65, 60

Question H.

La construction de l'arbre à partir de sa liste préfixée redonne l'arbre initial, ce qui est toujours le cas. En effet, dans la liste préfixe, tout nœud est avant ses descendants. Au moment où on ajoute un nœud en feuille, tous ses ascendants sont déjà dans l'arbre et aucun de ses descendants.

Question I.

La suppression dans un arbre binaire de recherche consiste d'abord à rechercher le nœud à supprimer. S'il a deux fils, on le remplace par l'extrémité du bord gauche du sous arbre droit, et on supprime cette extrémité. Le nœud à supprimer, que ce soit le nœud initial ou l'extrémité du bord droit a ainsi au plus un fils. Il suffit alors de remonter le fils restant à la place du nœud à supprimer.

Question J.

La suppression de 30 dans le dernier arbre obtenu dans la question H, commence par sa recherche. Il s'agit du fils droit de 15, et il a deux fils. On remplace sa valeur par l'extrémité du bord gauche du sous arbre droit, donc par 35, et on supprime cette extrémité qui n'a pas de fils. L'arbre obtenu est le suivant.

