

Algorithmes et structures de données Travail Écrit 2

13.06.2022

Nom :

Prénom :

Note :

Durée du test : 1h30

Documents autorisés : 6 feuilles de notes personnelles (recto/verso : 12 pages)

Modalités d'évaluation : Le travail écrit est noté sur **50** points. La répartition des points par exercice est communiquée dans le tableau ci-dessous.

Exercice	Points
1	10
2	10
3	10
4	10
5	5
6	5

IMPORTANT : veuillez indiquer vos nom et prénom sur chacune des feuilles.

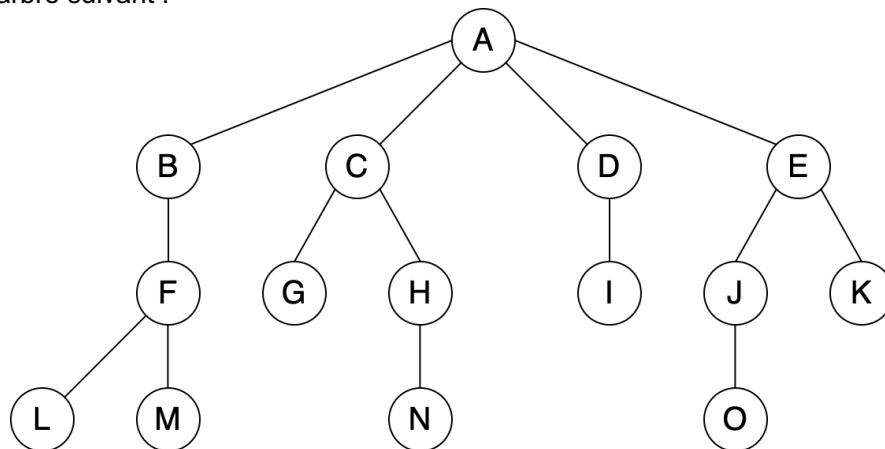
Ne rien écrire sur cette page

Nom :

Prénom :

1. Arbres [10 pts]

a. Soit l'arbre suivant :



- Effectuez le parcours pré-ordonné : (1pt)

ABFLMCGHNDIEJOK ABFLMCGHNDIEJOK ✓

- Effectuez le parcours post-ordonné : (1pt)

LMFBGNHCIDJKEA LMFBGNHCIDJKEA ✓

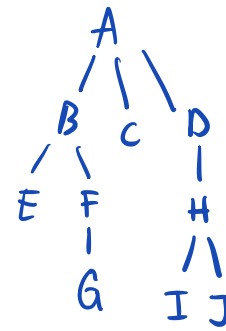
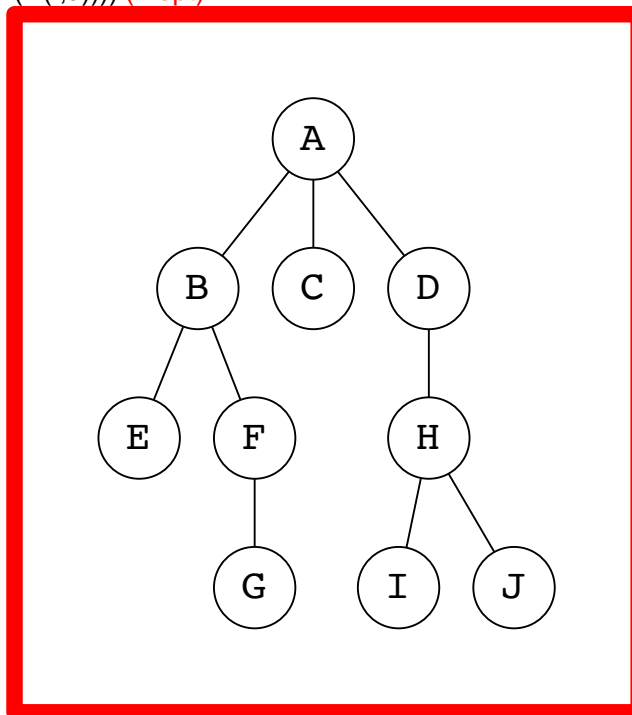
- Effectuez le parcours en largeur : (1pt)

ABCDEFGH IJKLMNO ABCDEFGH IJKLMNO ✓

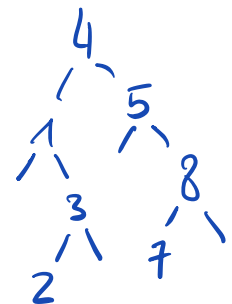
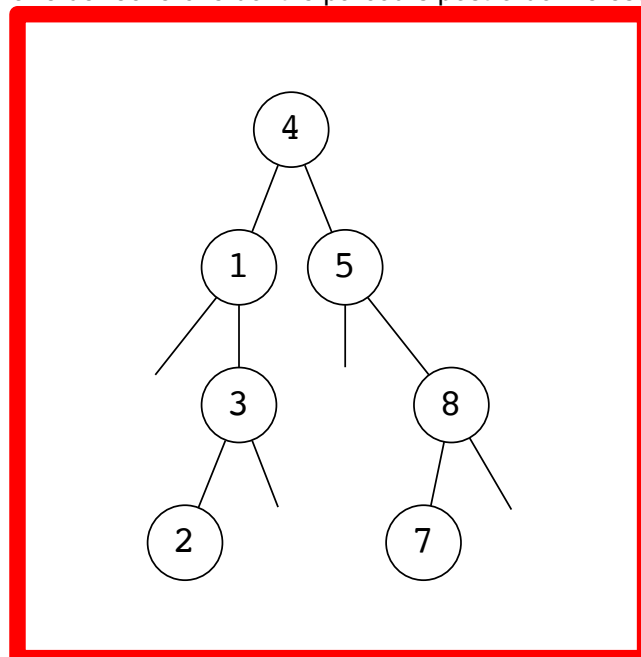
- Dans le parcours en largeur, quel est le contenu de la file d'attente après le traitement du nœud I (après avoir retiré I de la file) : (1pt)

JKLMN ?

- b. Dessinez l'arbre quelconque dont la représentation sous forme de liste imbriquée est (A(B(E,F(G)),C,D(H(I,J)))) (1.5pt)



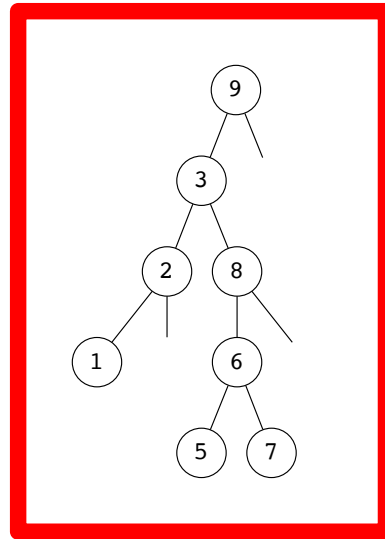
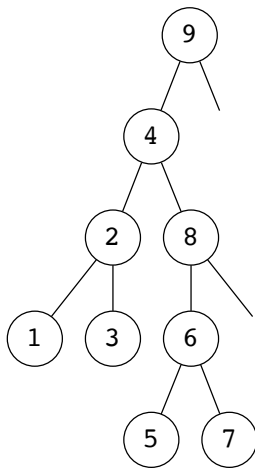
- c. Dessinez l'arbre binaire de recherche dont le parcours post-ordonné est 2,3,1,7,8,5,4 (1.5pt)



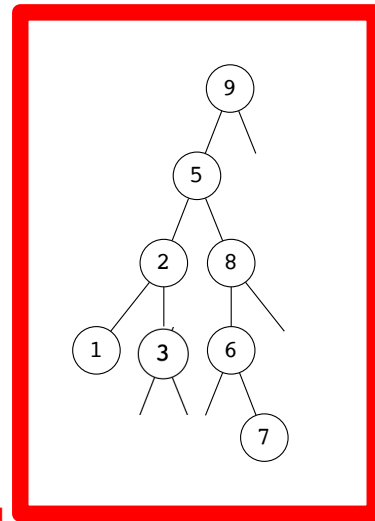
Nom :

Prénom :

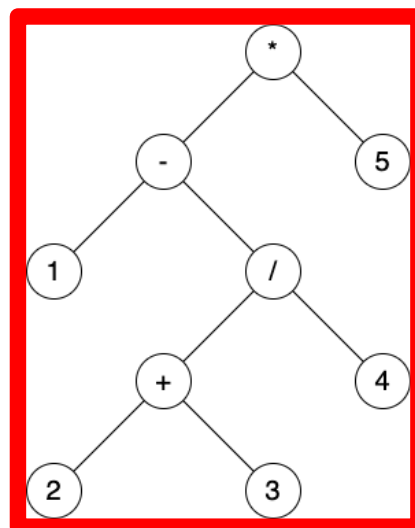
- d. Soit l'arbre binaire de recherche suivant, dessinez l'arbre résultant de la suppression du nœud de clé 4 (1.5pt)



ou



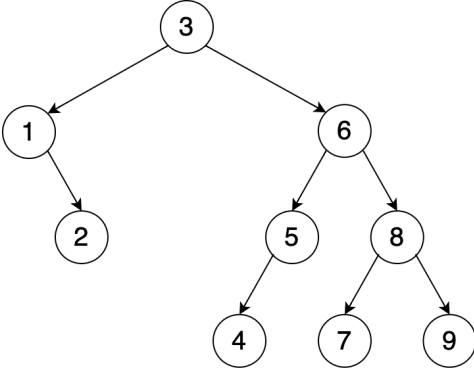
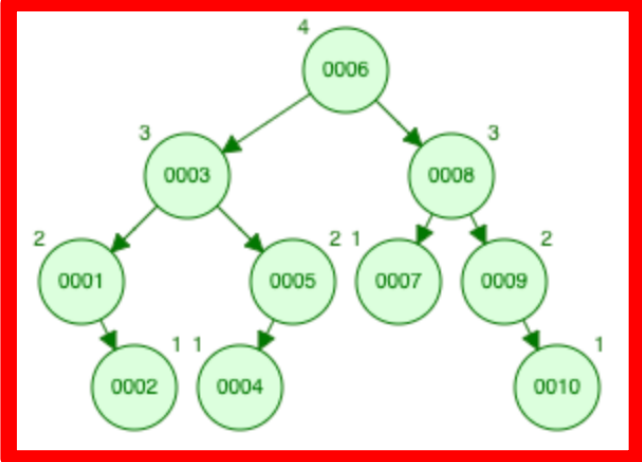
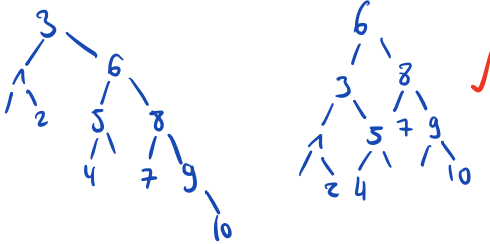
- e. Dessinez l'arbre binaire dont la notation postfixe est : 1 2 3 + 4 / - 5 * (1.5pt)



2. Arbres AVL [10 points]

Pour chacun des cinq arbres AVL suivants, dessiner le résultat **final** de l'opération correspondante.

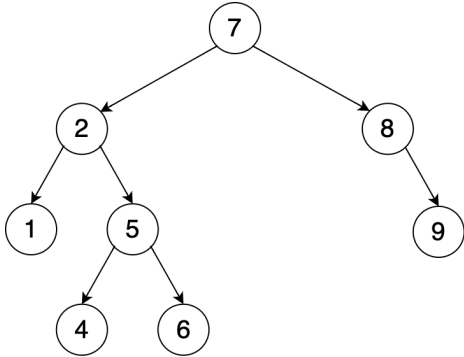
S'il faut supprimer un nœud avec 2 fils, privilégiez la recherche du remplaçant dans le sous-arbre **gauche**.

Insertion de 10	
	<p>Résultat final :</p>  

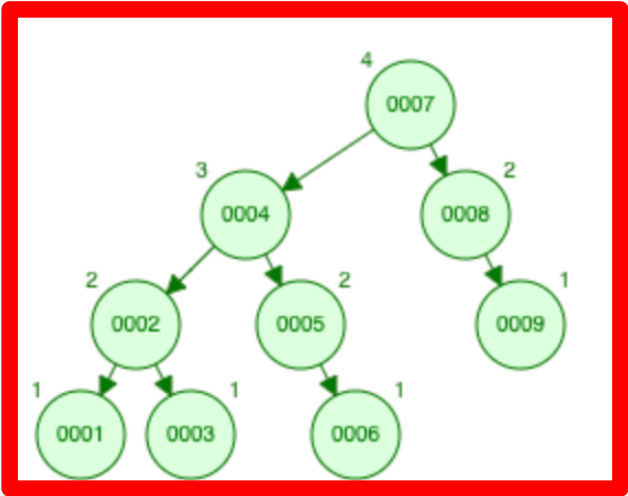
Nom :

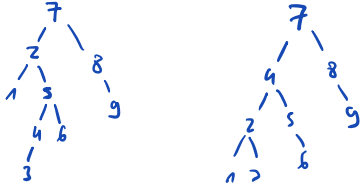
Prénom :

Insertion de 3

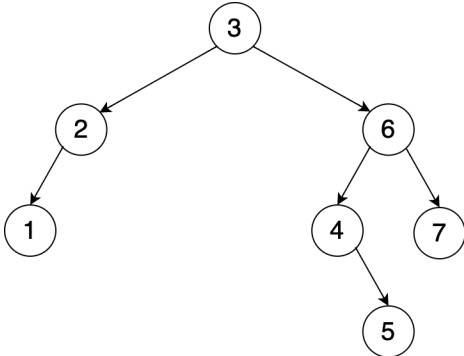


Résultat **final** :

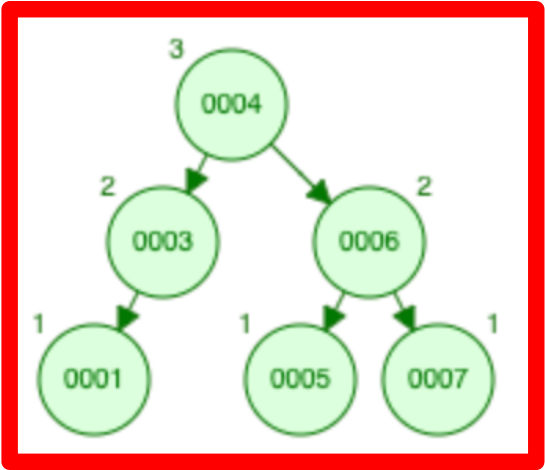


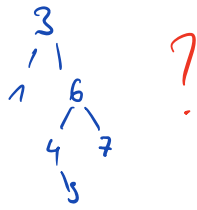


Suppression de 2

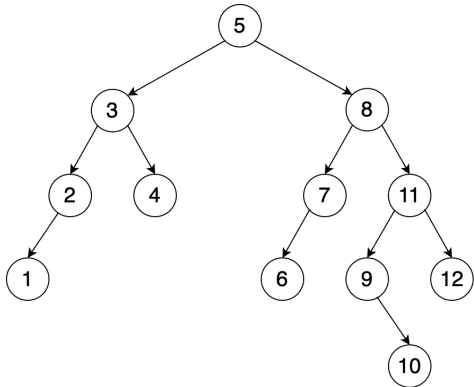


Résultat **final** :

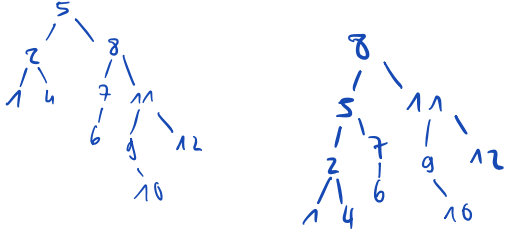
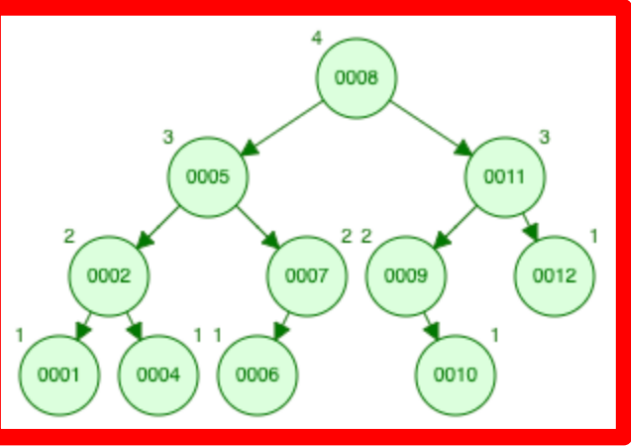




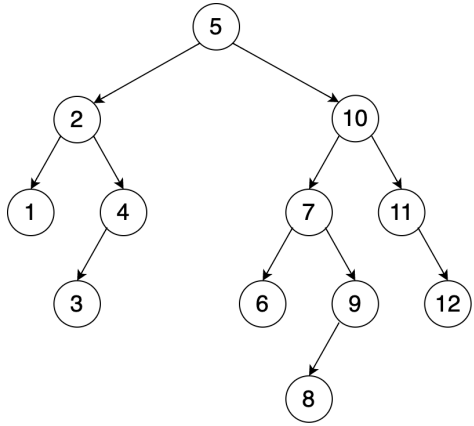
Suppression de 3



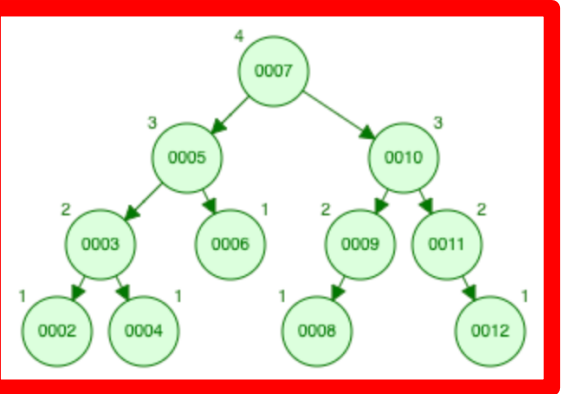
Résultat final :



Suppression de 1



Résultat final :



Profs Olivier Cuisenaire, Laura Elena Raileanu

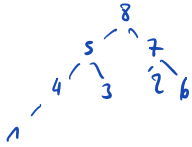
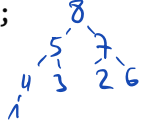


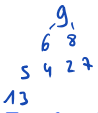
8/13

Nom :

Prénom :

3. Tas [10 pts]

Qu'affichent les codes suivants ?

<pre>std::vector<int> v{6, 3, 7, 1, 5, 2, 8, 4}; std::make_heap(v.begin(), v.end()); for (int e: v) std::cout << e << ' ';</pre> 	<p>8 5 7 4 3 2 6 1</p> <p>8 5 7 4 3 2 6 1</p>
<pre>std::vector<int> v{8, 4, 7, 1, 3, 2, 6}; v.push_back(5); std::push_heap(v.begin(), v.end()); for (int e: v) std::cout << e << ' ';</pre> 	<p>8 5 7 4 3 2 6 1</p> <p>8 5 7 4 3 2 6 1</p>
 <pre>std::vector<int> v{3, 7, 1, 2, 4, 8, 6, 5}; std::pop_heap(v.begin(), v.end()); for (int e: v) std::cout << e << ' ';</pre>	<p>7 5 1 2 4 8 6 3</p>
 <pre>std::vector<int> v{2, 5, 3, 4, 7, 1, 6}; std::sort_heap(v.begin(), v.end()); for (int e: v) std::cout << e << ' ';</pre>	<p>1 3 4 7 5 6 2</p>
 <pre>std::vector<int> v{9, 6, 8, 5, 4, 2, 7, 1, 3}; std::cout << std::boolalpha << std::is_heap(v.begin(), v.end());</pre>	<p>true</p> <p>true</p>

4. Structures linéaires et associatives [10 pts]

Qu'affichent les codes suivants ?

<pre>vector<int> sequence {0, 1, 2, 3, 4, 5}; sequence.reserve(7); auto it = next(sequence.begin(), 2); sequence.insert(it, 6); cout << *it;</pre>	
<pre>list<int> sequence {0, 1, 2, 3, 4, 5}; auto it = next(sequence.begin(), 2); sequence.insert(it, 6); cout << *it;</pre>	
<pre>list<int> sequence {0, 1, 2, 3, 4, 5}; sequence.splice(next(sequence.begin(),1), sequence, next(sequence.begin(),3), sequence.end()); for(auto e : sequence) cout << e;</pre>	
<pre>list<int> sequence {5,3,1,2,4,6}; auto it = sequence.begin(); cout << *it; sequence.sort(); cout << *it;</pre>	
<pre>forward_list<int> sequence{0, 1, 0, 2, 0, 0, 3, 0 }; for (auto it = sequence.begin(); next(it) != sequence.end();) { if (*next(it) == 0) sequence.erase_after(it); else ++it; } for(auto e : sequence) cout << e;</pre>	

Nom :

Prénom :

<pre>stack<int> v; for(int i : { 3, 7, 9, 1}) v.push(i); while(not v.empty()) { cout << v.top(); v.pop(); }</pre>	
<pre>queue<int> v; for(int i : { 3, 7, 9, 1}) v.push(i); while(not v.empty()) { cout << v.front(); v.pop(); }</pre>	
<pre>priority_queue<int> v; for(int i : { 3, 7, 9, 1}) v.push(i); while(not v.empty()) { cout << v.top(); v.pop(); }</pre>	
<pre>set<int> ensemble {0,1,3,2,1,2,0}; for(auto e : ensemble) cout << e;</pre>	
<pre>map<int,int> tableau; for(int i : { 1, 3, 5, 3, 1}) tableau[i] = i*i; cout << tableau.size(); for(int i : { 1, 2, 3}) cout << tableau[i]; cout << tableau.size();</pre>	

5. Complexités en mémoire [5 pts]

Quelle quantité de mémoire (en octets) est utilisée par la structure S ? Votre réponse peut ignorer les termes négligeables pour N grand, mais doit inclure le coefficient multiplicatif exact pour le terme principal. Par exemple, si la réponse exacte est $34 \cdot N^2 + 12 \cdot N + 1024$, la réponse attendue est $34 \cdot N^2$.

Pour cet exercice, supposez un système 64 bits où `sizeof(int) = 4`, `sizeof(void*) = 8`, et `sizeof(size_t) = 8`.

<pre>array<array<int,N>,N> S;</pre>	
<pre>vector<vector<int>> S(N,vector<int>(10));</pre>	
<pre>vector<forward_list<int>> S(N);</pre>	
<pre>vector<list<int>> S(N); for(size_t i = 0; i < N; ++i) for(int j = 0; j < N; ++j) S[i].push_back(j);</pre>	
<pre>vector<forward_list<int>> S[N]; for(size_t i = 0; i < N; ++i) for(int_t j = 0; j < i; ++j) S[i].push_front(j);</pre>	

Nom :

Prénom :

6. Complexités temporelles [5 pts]

Quelle est la complexité (en temps/nombre d'opération) des fonctions suivantes ?

<pre>void f1(vector<int> & v) { auto N = v.size(); while(not v.empty()) v.erase(v.begin()); }</pre>	
<pre>void f2(list<int> & v) { auto N = v.size(); while(not v.empty()) v.erase(v.begin()); }</pre>	
<pre>set<int> f3(int N) { set<int> s; for(int i = 0; i < N; ++i) s.insert(rand()); return s; }</pre>	
<pre>set<int> f4(int N) { int mod = log(N); set<int> s; for(int i = 0; i < N; ++i) s.insert(rand() % mod); return s; }</pre>	
<pre>void f5(vector<int> & v) { auto N = v.size(); make_heap(v.begin(), v.end()); for (int i = 0; i < N; ++i) { v.push_back(rand()); push_heap(v.begin(), v.end()); } }</pre>	