

CSI2510 Automne 2017 Examen Final

Prof. Lucia Moura et Prof. Robert Laganieri

14 Décembre, 9h30 – 3 heures 40 points (40% de votre note finale)

NOM de FAMILLE: _____

Prénom: _____

Numéro d'étudiant: _____

Signature _____

Instructions :

Aucune documentation permise.

Calculatrice interdite.

**Les questions à choix multiples se répondent
sur un papier Scantron.**

15 pages.

Pour toutes les questions où la complexité
Grand O est demandée, toujours donner la
meilleure valeur possible.

Tous les logarithmes sont en base 2.

QUESTION	résultat
Questions 1-15	/15
Question 16 (AVL)	/2
Question 17 (AVL)	/2
Question 18 (graphe)	/6
Question 19 (Dijkstra)	/4
Question 20 (tri)	/2
Question 21 (tri fusion)	/3
Question 22 (arbres)	/6
TOTAL	

Question 1 [1 point] Quelle la complexité asymptotique Grand O de la fonction suivante:
What is the big-oh characterization of the following function:

$$f(N) = 1 + 2 + 4 + 8 + \dots + 2^{N-1} + 2^N \quad ?$$

- A) $O(N^2)$ B) $O(\log N)$ C) $O(N)$ D) $O(2^N)$ E) $O(N^N)$

Question 2 [1 point] Une fonction $f(N)$ est $O(N)$ et une autre fonction $g(N)$ est $O(N^2)$, quelle serait alors la limite asymptotique Grand O de la fonction $f(N)+g(N)$?

If a function $f(N)$ is $O(N)$ and a second function $g(N)$ is $O(N^2)$, what would be the complexity of the function $f(N)+g(N)$?

- A) $O(N)$ B) $O(N^2)$ C) $O(N^3)$ D) $O(\log N)$ E) impossible à déterminer

Question 3 [1 point] La fonction suivante permet de trouver l'index des 2 nombres dans un tableau ayant le produit le plus élevé. Quelle est la complexité Grand O de cet algorithme en termes du nombre de multiplications effectuées?

The following function returns the index of the two numbers having the maximum product value. What is the big-oh complexity of this algorithm in terms of the number of multiply operations?

```
1.  int x= -1; int y= -1;
2.  int maxProd = -1;
3.  for (i=0; i<n-1; i++)
4.      for (j=i+1; j<n; j++) {
5.          int prod = tab[i] * tab[j]);
6.          if (prod > maxProd) {
7.              maxProd = prod;
8.              x=i;
9.              y=j;
10.         }
11. return x, y;
```

- A) $O(N^2)$ B) $O(\log N)$ C) $O(N)$ D) $O(2^N)$ E) $O(N^N)$

Question 4 [1 point] Dessiner l'arbre binaire fait de noeuds contenant des lettres et dont le parcours inordre donne ZLCXE et le parcours postordre donne ZCLEX.

Draw the binary tree where each node contains a letter and its inorder traversal prints ZLCXE and its postorder traversal prints ZCLEX.

Laquelle des séquences suivantes est le résultat du parcours préordre de cet arbre?

Which one is the result of of a preorder traversal of the tree you just draw?

- A) XLEZC B) XELCZ C) LEXCZ D) ZLCXE E) XLZCE

Question 5 [1 point] Soit un **monceau max** représenté par le tableau $a[] = [10, 5, 3, 1, 4, 2]$, quel sera le tableau résultant après l'opération de retrait `removeMax()`?

*Consider the **max heap** that uses the following array $a[] = [10, 5, 3, 1, 4, 2]$ to store its elements. Which is the array after one `removeMax()` operation?*

- A) [5, 3, 1, 4, 2, -]
B) [2, 5, 3, 1, 4, -]
C) [5, 4, 3, 1, 2, -]
D) [5, 2, 3, 1, 4, -]
E) [5, 4, 3, 2, 1, -]

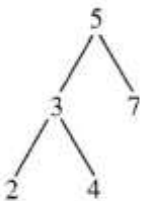
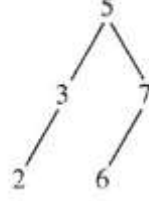

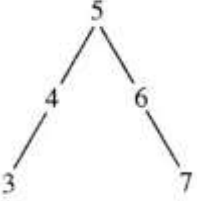
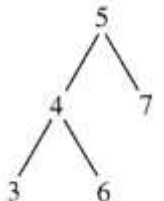
Question 6 [1 point] Soit un **monceau max** représenté par le tableau $a[] = [10, 5, 3, 1, 4, 2]$, quel sera le tableau résultant après l'insertion de la clé 9 dans ce monceau?

Consider the **max heap** that uses the following array $a[] = [10, 5, 3, 1, 4, 2]$ to store its elements. Which is the array after inserting key 9 into the heap?

- A) [10,5,3,1,4,2,9]
- B) [10,9,5,1,4,2,3]
- C) [10,9,1,4,5,2,3]
- D) [10,5,9,1,4,2,3]
- E) aucune de ces réponses

Question 7 [1 point] Lequel des arbres suivants est un **arbre binaire de recherche**?

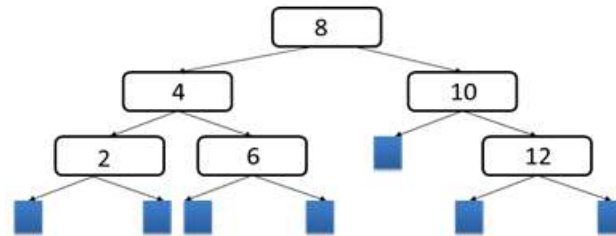
Which ones of the following is a **binary search tree**? (Here no dummy leaves are used)

I	II	III	IV	V
				

- A) I, II, III, IV
- B) I, II, IV, V
- C) I, II, V
- D) I and V
- E) Tout ces arbres sont des arbres binaires de recherche

Question 8 [1 point] Laquelle des clés ci-dessous exigera une opération de re-balancement lorsqu'insérée dans l'arbre AVL suivant?

Which of the following keys when inserted into the following AVL tree will cause a rebalancing operation?



- A) Insertion de 1
- B) Insertion de 7
- C) Insertion de 9
- D) Insertion de 11
- E) Aucune de ces réponses

Question 9 [1 point] Soit la fonction de hachage $h(k) = k \bmod 9$ utilisée sur une table de hachage de dimension 9 et sur laquelle les collisions sont résolues en utilisant le **sondage linéaire**. Quel sera le résultat de l'insertion, dans l'ordre, des éléments suivants dans une table initialement vide ? 11, 2, 3, 7, 13, 12, 9

*Consider the following hash function $h(k) = k \bmod 9$, and a hash table of size 9, where collisions are resolved using **linear probing**. What is the table resulting from the insertions (in the following order) into an initially empty table: 11, 2, 3, 7, 13, 12, 9*

- A)

Index:	0	1	2	3	4	5	6	7	8
Hash Table	11	2	3	7	13	12	9		
- B)

Index:	0	1	2	3	4	5	6	7	8
Hash Table	9		2	3	11	13	12	7	
- C)

Index:	0	1	2	3	4	5	6	7	8
Hash Table	9		11	2	3	12	13	7	
- D)

Index:	0	1	2	3	4	5	6	7	8
Hash Table	9		11	2	3	13	12	7	
- E)

Index:	0	1	2	3	4	5	6	7	8
Hash Table	9	11	2	3	13	12		7	

Question 10 [1 point] Soit la fonction de hachage $h(k) = k \bmod 9$ appliquée sur la table de hachage suivante de dimension 9.

Index:	0	1	2	3	4	5	6	7	8
Hash Table			11	3	13	5		7	

A quelle position dans la table la clé $k=12$ se retrouvera si le **sondage quadratique** est utilisé afin de résoudre les collisions? C'est-à-dire $h_j(k) = (h(k) + j^2) \bmod 9$, pour les sondages $j=0,1,2,\dots$

*In which index of the hash table we must insert key $k=12$ if collisions are resolved using **quadratic probing** ? Remember that in quadratic probing we attempt to enter keys into positions: $h_j(k) = (h(k) + j^2) \bmod 9$, for $j=0,1,2,\dots$*

- A) Index 0
- B) Index 1
- C) Index 6
- D) Index 8
- E) Il est impossible d'insérer la clé $k=12$.

Question 11 [1 point] Laquelle de ces réponses donne la meilleure estimation de la complexité au pire cas de tri-fusion, tri par monceau et du tri rapide?

*Select the answer that gives the most precise estimate of the **worst-case time complexity** of Mergesort, Heapsort and Quicksort, respectively.*

	Tri fusion	Tri par monceau	Tri rapide
A)	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$
B)	$O(n \log n)$	$O(\log n)$	$O(n \log n)$
C)	$O(n \log n)$	$O(n \log n)$	$O(n^2)$
D)	$O(n^2)$	$O(n \log n)$	$O(n \log n)$

- E) aucune de ces réponses

Question 12 [1 point] Des éléments se trouvent stockés dans un tableau **en ordre croissant**. Ignorant que ce tableau est déjà trié, vous appliquez un algorithme de tri standard afin de trier ces n éléments **en ordre croissant**.

*Suppose you have an array of n elements sorted in **increasing** order. You now use a sorting algorithm to sort them in **increasing** order (the algorithm does not know the input is in increasing order and runs as usual)*

Quelle sera alors la complexité du tri de ces n éléments avec les algorithmes ci-dessous?

This question is about the most precise big-Oh complexity for the problem above using:

- Quicksort (with pivot in partition being the first element of the sequence)
- Insertion sort
- Selection sort

	Tri rapide (le pivot est le 1er élément)	Tri par Insertion	Tri par Selection
A)	$O(n \log n)$	$O(n)$	$O(n)$
B)	$O(n^2)$	$O(n)$	$O(n^2)$
C)	$O(n \log n)$	$O(n^2)$	$O(n^2)$
D)	$O(n^2)$	$O(n^2)$	$O(n^2)$

E) aucune de ces réponses

Question 13 [1 point] Soit une classe de graphes en Anneau, où tous les nœuds sont connectés à exactement 2 voisins; l'ensemble des n nœuds formant un long anneau circulaire. Soit un algorithme de parcours de graphe ayant une complexité $\Theta(n+m)$ où m est le nombre total d'arêtes. Exprimer la complexité de cet algorithme de parcours en fonction de n , lorsque n est très grand.

Consider a class of graphs, called Ring, with n vertices where each vertex is adjacent to exactly 2 other vertices. Consider a graph traversal with time complexity $\Theta(n+m)$ where n is the number of vertices and m is the number of edges in the graph. What is the time complexity of the graph traversal on the given type of graph as a function of n , assuming n is very large?

- A) $\Theta(n)$
- B) $\Theta(n \log n)$
- C) $\Theta(n^2)$
- D) $\Theta(n^3)$
- E) Aucune de ces réponses.

Question 14 [1 point] Soit la fonction `stableSort(A,i)` permettant de trier un tableau de chaînes de caractères W formées de 4 lettres. Ce tri permet de trier ces chaînes en utilisant seulement la lettre à la position i de chaque chaîne. De plus, ce tri est stable. Le code ci-dessous est appliqué.

*Consider procedure `stableSort(A,i)` which sorts a global array of strings W with 4 letters by simply using the letter at position i of each string S as the key for sorting in increasing order of $S[i]$. Moreover we know that the algorithm is **stable**. Consider the piece of code:*

```
for i=0 to 3 do stableSort(A, i)
```

Si le tableau de chaînes à trier est le suivant, quel sera le résultat obtenu?

If before the piece of code the array has the Strings in the following order

$W = [\text{DEBY}, \text{ANNA}, \text{ANDY}, \text{BOBI}, \text{MEBA}, \text{ANGU}, \text{IKBA}]$

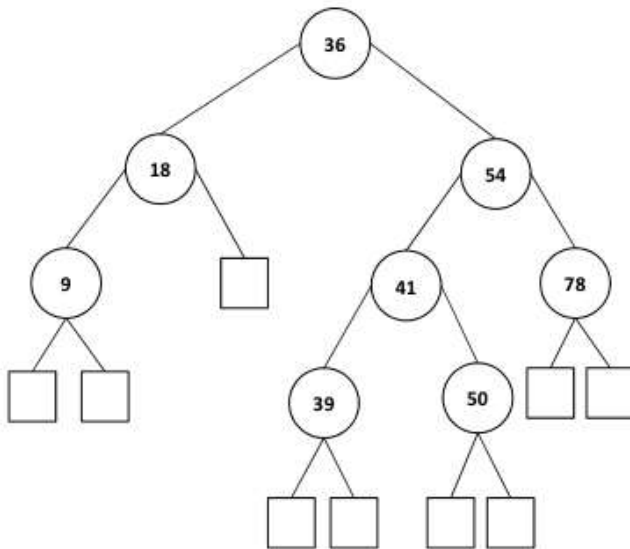
- A) $W = [\text{DEBY}, \text{ANNA}, \text{ANDY}, \text{BOBI}, \text{MEBA}, \text{ANGU}, \text{IKBA}]$
- B) $W = [\text{ANNA}, \text{ANDY}, \text{ANGU}, \text{DEBY}, \text{IKBA}, \text{BOBI}, \text{MEBA}]$
- C) $W = [\text{ANDY}, \text{ANGU}, \text{ANNA}, \text{DEBY}, \text{IKBA}, \text{BOBI}, \text{MEBA}]$
- D) $W = [\text{MEBA}, \text{IKBA}, \text{ANNA}, \text{BOBI}, \text{ANGU}, \text{DEBY}, \text{ANDY}]$
- E) None of the above.

Question 15 [1 point] Laquelle des structures de données ne peut pas être utilisée dans la réalisation du type abstrait de données Map (un dictionnaire avec clé unique)?

All the alternatives are data structures that can be used to implement the MAP Abstract Data Type, with the exception of

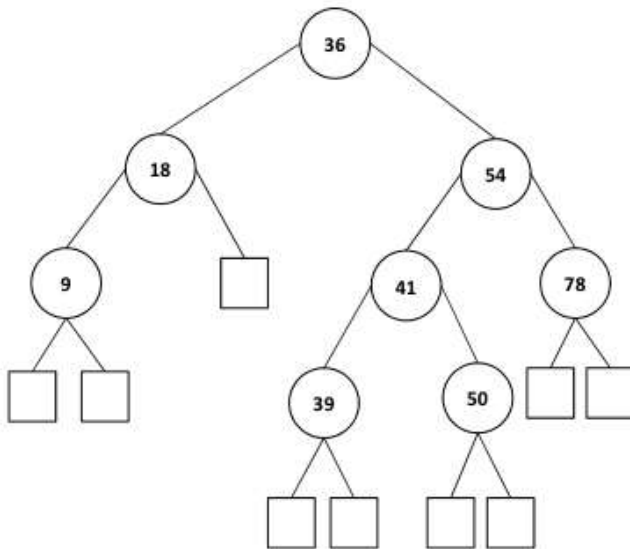
- A) Table de hachage
- B) Arbre binaire de recherche
- C) Arbre AVL
- D) Liste avec Tableau
- E) File à deux bouts (Deque)

Question 16 [2 point] Soit l'arbre AVL ci-dessous.
Consider the following AVL tree.



Insérer la clé 38 en utilisant l'algorithme AVL vu en classe et montrer l'arbre résultant.
Insert key 38 using the algorithm seen in class and show the resulting tree :

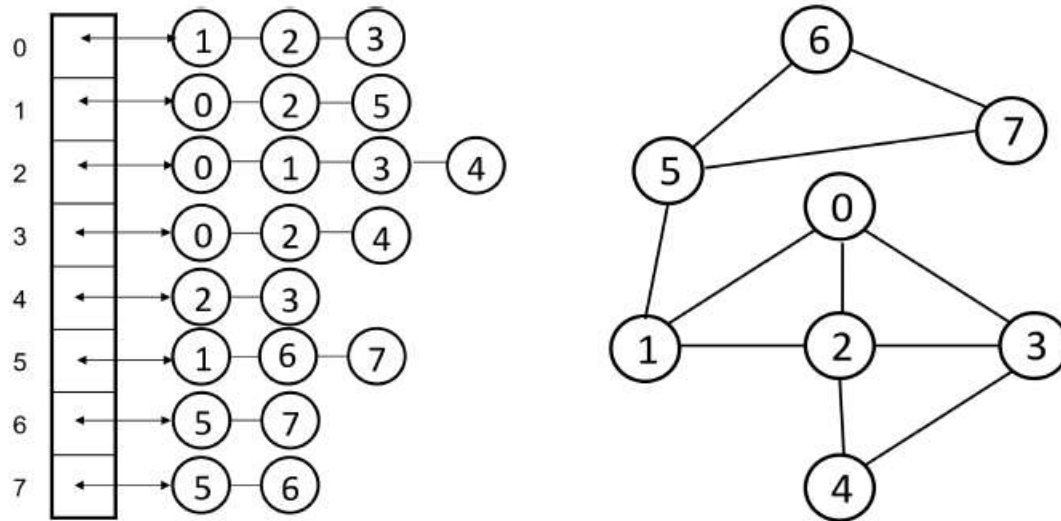
Question 17 [2 point] Soit l'arbre AVL ci-dessous.
Consider the following AVL tree.



Retirer la clé 9 en utilisant l'algorithme AVL vu en classe et montrer l'arbre résultant.
Delete key 9 using the algorithm seen in class and show the resulting tree :

Question 18 [6 points] Soit le graphe ci-dessous et sa représentation avec une liste d'adjascence

Consider the graph given below and consider the given adjacency list representation of the graph.



Prenant en considération les noeuds dans leur ordre d'apparition dans la liste d'adjascence et partant du noeud 1,

*Taking into account the order of the vertices in each adjacency list, and **starting the search from vertex 1**,*

A) donner, dans l'ordre, la liste des noeuds qui seront visités dans le cas d'un parcours de recherche en profondeur à partir du noeud 1.

*Give, in order, the list of visited nodes in the case of a **depth-first search** starting from vertex 1:*

B) donner, dans l'ordre, la liste des noeuds qui seront visités dans le cas d'un parcours de recherche en largeur à partir du noeud 1.

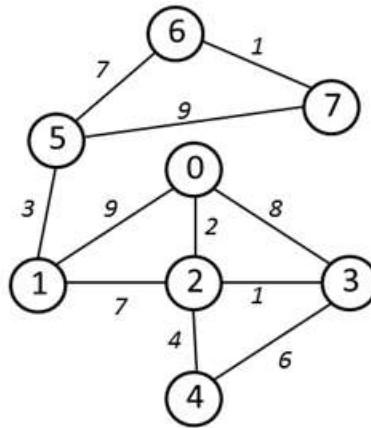
*Give, in order, the list of visited nodes in the case of a **breath-first search** starting from vertex 1:*

C) Dessiner la matrice d'adjascence correspondant au graphe montré ci-dessus.

Below draw the adjacency matrix representation of the graph :

Question 19 [4 points] Utiliser l'algorithme de Dijkstra afin de trouver l'arbre des plus courts chemins à partir du noeud 3 du graphe ci-dessous.

Use Dijkstra's algorithm to obtain a tree of shortest paths starting for starting vertex 3 for the graph below.



Remplir la table ci-dessous montrant comment les valeurs de distance sont mises à jour en cours d'exécution. La première ligne de cette table est déjà remplie correspondant au moment où le nœud 3 s'ajoute au nuage de sommets sélectionnés. L'arête suivante à venir est la (3,2). Continuer à remplir ce tableau jusqu'à ce que tous les sommets aient été ajoutés au nuage de sommets. La colonne de droite monte toutes les arêtes faisant partie de l'arbre des plus courts chemins.

Fill the following table to show the updates of the distances associated to the vertices along its execution. We already filled the iteration when 3 is added to the cloud. The next vertex to join the cloud is vertex 2 and at this time edge (3,2) is added to the tree of shortest paths. You need to keep filling the table with distance updates until all vertices are added to the cloud. On the right column you show edges that join the tree of shortest paths at each iteration.

itération	Sommet ajouté au sommet	Dist[0]	Dist[1]	Dist[2]	Dist[3]	Dist[4]	Dist[5]	Dist[6]	Dist[7]	Arêtes ajoutés à l'arbre des plus courts chemins
0	3	8	infinity	1	0	6	infinity	infinity	infinity	--
1	2									(3,2)
2										
3										
4										
5										
6										
7										

Question 20 [2 points]

Un développeur tente de programmer un algorithme de tri mais malheureusement il obtient des résultats incorrects.

*A developer tries to implement a sorting algorithm but unfortunately it is **incorrect**:*

```
1.  public static void selection(int[] data){
2.      for (int i = 0; i < data.length - 1; i++) {
3.          int minIndex = i;
4.          for (int j = 0; j < data.length; j++) {
5.              if (data[j] < data[minIndex])
6.                  minIndex = j;
7.          }
8.          // swap data between i and minIndex
9.          int temp = data[i];
10.         data[i] = data[minIndex];
11.         data[minIndex] = temp;
12.     }
13. }
```

Lorsque cet algorithme est testé avec l'entrée suivante:

When the algorithm is tested with the following array:

10, 34, 2, 56, 7, 67, 88, 42

le résultat suivant est obtenu:

the following result is obtained:

34, 10, 56, 7, 67, 88, 2, 42

Corriger ce programme en apportant une modification à une et une seule ligne de code.
(donner le numéro de la ligne modifiée)

Correct the code by proposing a change to one and only one line of code. In the space below give the line number and the modified line:

Question 21 [3 points] Dessiner l'arbre de récursivité de l'algorithme de tri fusion. Ici est montrée la première subdivision du tableau; vous devez montrer toutes les étapes de subdivision et de fusion jusqu'à ce que le tableau soit complètement trié.

Draw the Mergesort recursion tree with the following array:

9	1	6	3	2	8	4	7
---	---	---	---	---	---	---	---

9	1	6	3
---	---	---	---

2	8	4	7
---	---	---	---

Question 22 [6 points] Soit un arbre binaire de recherche et le programme ci-dessous. Pour un nœud p dans cet arbre binaire, $\text{left}(p)$ est l'enfant gauche de p et $\text{right}(p)$ est l'enfant droit du nœud p . La fonction $\text{compare}(x,y)$ est une méthode comparateur retournant 0 si $x=y$, -1 si $x<y$ et 1 si $x>y$.

Consider a binary search tree and the piece of code below. For a node p in the binary search tree, $\text{left}(p)$ is the node that is the left child of p and $\text{right}(p)$ is the node that is the right child of p . The function $\text{compare}(x,y)$ is a standard comparator method that returns 0, if $x=y$, -1 if $x<y$, and 1 if $x>y$.

```
int private
Position<Entry<K,V>> Mystery(Position<Entry<K,V>> p, K key) {
    if (isExternal(p))
        return p; // case that algorithm Mystery is not successful
    int comp = compare(key, p.getElement());
    if (comp == 0)
        return p; // case that algorithm Mystery is successful
    else if (comp < 0)
        return Mystery(left(p), key);
    else
        return Mystery(right(p), key);
}
```

Partie A) Que fait la méthode $\text{Mystery}(p, \text{key})$?

What does method $\text{Mystery}(p, \text{key})$ do?

Partie B) Quel sera la complexité (Grand O), **au pire cas**, l'appel $\text{Mystery}(\text{root}, \text{key})$ avec root étant la racine d'un arbre composé de n nœuds dans les cas suivants :

For a call $\text{Mystery}(\text{root}, \text{key})$, where root is the root of a tree containing n nodes, what is the worst-case time complexity (big-Oh) of the algorithm in the following cases

L'arbre est un arbre binaire de recherche: _____

L'arbre est un arbre AVL: _____