

La rigueur syntaxique du code est testée lors de l'épreuve pratique. Dans cette partie écrite, le correcteur pourra se montrer tolérant sur le fait que le code proposé par le candidat ne soit pas exécutable en raison d'erreurs de syntaxe minimales. Ainsi, par exemple, l'oubli du symbole « : » lors de la définition d'une fonction en Python ne sera pas sanctionné. Néanmoins, le correcteur veillera à sanctionner une ambiguïté résultant, par exemple, d'une mauvaise indentation lors de l'écriture d'un code impliquant deux boucles imbriquées.

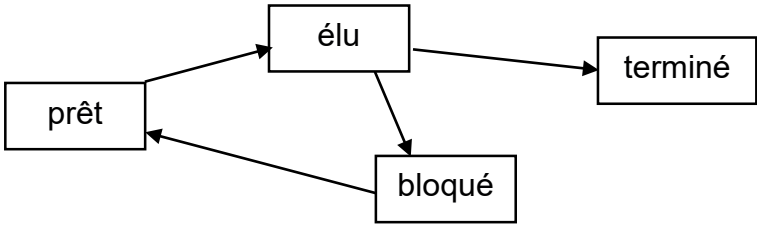
Il est fréquent que plusieurs réponses soient possibles, notamment dans l'écriture d'un code en Python. Ces éléments de correction ne proposent pour la plupart qu'une seule solution. Toute réponse correcte sera acceptée.

EXERCICE 1 (4 points)

Question	Éléments de correction	Barème	Commentaires
1.	Deux entrées ne peuvent posséder la même valeur pour l'attribut <code>idEleve</code> qui est une clé primaire.	0,5	
2.	L'attribut <code>idEleve</code> est une clé étrangère. Toutes les valeurs de la clé étrangère sont incluses dans l'ensemble des valeurs de la clé primaire.	0,5	
3.	<code>SELECT titre FROM Livres WHERE auteur='Molière';</code>	0,5	
4.	La requête renvoie le nombre d'élèves de la classe 'T2' inscrits au CDI.	0,5	On ne pénalise pas l'oubli de « inscrits au CDI ».
5.	<code>UPDATE Emprunts SET dateRetour = '2020-09-30' WHERE idEmprunt = 640 ;</code>	0,5	
6.	La requête renvoie la liste des élèves de T2 inscrits au CDI qui ont emprunté au moins un livre.	0,5	On ne pénalise pas l'oubli de « inscrits au CDI ».

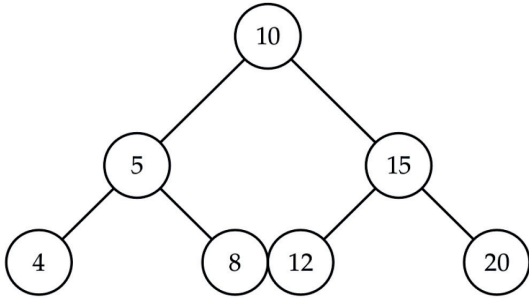
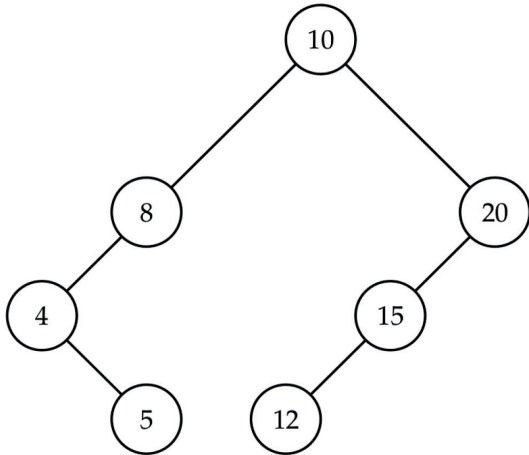
7.	<pre>SELECT DISTINCT nom, prenom FROM Eleves, Emprunts, Livres WHERE Emprunts.idEleve = Eleves.idEleve AND Emprunts.isbn = Livres.isbn AND Livres.titre = 'Les misérables'; ou SELECT DISTINCT nom, prenom FROM Emprunts JOIN Eleves ON Emprunts.idEleve = Eleves.idEleve JOIN Livres ON Emprunts.isbn = Livres.isbn WHERE Livres.titre = 'Les misérables';</pre>	1	
----	---	---	--

EXERCICE 2 (4 points)

1.a.	Élu correspond à l'état où le processus est en cours d'exécution sur le processeur.	0,5	
1.b.	<p>Ce schéma est le minimum attendu. On accepte tout complément cohérent : en particulier le passage de <i>élu</i> à <i>prêt</i> (préemption).</p>  <pre> graph LR prêt[prêt] --> élu[élu] élu --> terminé[terminé] élu --> bloqué[bloqué] bloqué --> prêt </pre>	0,5	
2.a.	Une file est une structure de données linéaire selon le principe « Premier entré, premier sorti » (FIFO).	0,25	

b.	<table><tr><td>C₁</td><td>élu</td><td colspan="4">bloqué</td><td colspan="2">prêt</td><td>élu</td><td>terminé</td></tr><tr><td>C₂</td><td>prêt</td><td colspan="3">élu</td><td colspan="5">terminé</td></tr><tr><td>C₃</td><td colspan="4">prêt</td><td>élu</td><td>bloqué</td><td>prêt</td><td>élu</td><td>terminé</td></tr><tr><td>C₄</td><td colspan="5">prêt</td><td colspan="2">élu</td><td colspan="2">terminé</td></tr></table> <div><div>0</div><div>20</div><div>40</div><div>60</div><div>80</div><div>100</div><div>120</div><div>140</div><div>160</div><div>180</div><div>200</div><div>220</div><div>240</div><div>260</div><div>280</div><div>300</div><div>320</div><div>340</div><div>360</div><div>380</div><div>400</div></div>	C ₁	élu	bloqué				prêt		élu	terminé	C ₂	prêt	élu			terminé					C ₃	prêt				élu	bloqué	prêt	élu	terminé	C ₄	prêt					élu		terminé		1,5	En cas d'erreur (ou décalage) on valorisera toute réponse cohérente.
C ₁	élu	bloqué				prêt		élu	terminé																																		
C ₂	prêt	élu			terminé																																						
C ₃	prêt				élu	bloqué	prêt	élu	terminé																																		
C ₄	prêt					élu		terminé																																			
3.a.	Le programme 2 pose problème lorsqu'il verrouille le fichier_2 et veut verrouiller le fichier_1 quand le programme 1 a déjà verrouillé le fichier_1 et veut verrouiller le fichier_2. Il y a alors interblocage (ou deadlock).	0,75																																									
3.b.	Il suffit d'échanger les deux premières lignes du programme 2.	0,5																																									

EXERCICE 3 (4 points)

Question	Éléments de correction	Barème	Commentaires
1.a.	Taille : 7	0,25	
1.b.	Hauteur : 4	0,25	
2.	 <pre> graph TD 10((10)) --- 5((5)) 10 --- 15((15)) 5 --- 4((4)) 5 --- 8((8)) 15 --- 12((12)) 15 --- 20((20)) </pre>	0,5	
3.	 <pre> graph TD 10((10)) --- 8((8)) 10 --- 20((20)) 8 --- 4((4)) 8 --- 15((15)) 4 --- 5((5)) 15 --- 12((12)) </pre>	0,75	

4.	<pre>def hauteur(self): return self.racine.hauteur()</pre>	0,5	
5.	<p>Dans la classe Arbre :</p> <pre>def taille(self): return self.racine.taille()</pre> <p>Dans la classe Noeud :</p> <pre>def taille(self): if self.gauche == None and self.droit == None : return 1 if self.gauche == None : return self.droit.taille()+1 elif self.droit == None : return self.gauche.taille()+1 else : return self.gauche.taille() + self.droit.taille() +1</pre>	1	
6.a.	$t_{\min} = 2^{h-1}$	0,25	
6.b.	<pre>def bien_construit(self): t = self.taille() h = self.hauteur() return 2**(h-1)<=t</pre>	0,5	On accepte toute réponse cohérente avec la question 6.a. ou une comparaison avec la valeur t_{\min}

EXERCICE 4 (4 points)

Question	Éléments de correction	Barème	Commentaires
1.	<p>La valeur de <code>lst[i2]</code> est perdue.</p> <pre>def echange(lst, i1, i2) : tmp = lst[i2] lst[i2] = lst[i1] lst[i1] = tmp</pre> <p>on acceptera <code>lst[i2], lst[i1] = lst[i2], lst[i1]</code></p>	0,5	
2.	0, 1, 9 et 10 peuvent être renvoyées par la fonction <code>randint</code> .	0,25	
3.a.	À chaque appel récursif, la valeur de <code>ind</code> passée en argument est décrémentée, de sorte que la condition d'arrêt <code>ind<=0</code> finira toujours par être satisfaite.	0,75	
3.b.	$n - 1$ appels récursifs	0,5	
3.c.	<pre>[0, 3, 4, 1, 2] [0, 3, 4, 1, 2] [3, 0, 4, 1, 2]</pre>	1	
3.d.	<pre>def melange_iter(lst): ind = len(lst)-1 while ind>0: j = randint(0, ind) echange(lst, ind, j) ind = ind-1</pre> <p>on acceptera : <code>for ind in range(len(lst)-1,0,-1)</code></p> <p>l'affichage du resultat n'est pas attendu donc on accepte avec ou sans <code>print</code></p>	1	

EXERCICE 5 (4 points)

Question	Éléments de correction	Barème	Commentaires
1.a	Si tous les éléments sont positifs : la somme de tous les éléments.	0,25	
1.b	Si tous les éléments sont négatifs : la somme réduite au plus grand élément.	0,25	
2.a.	<pre>def somme_sous_sequence(lst, i, j): s = 0 for ind in range(i, j+1): s = s+lst[ind] return s</pre>	0,5	
2.b.	Pour $n = 10$: 55 comparaisons	0,5	
2.c.	<pre>def pgsp(lst): n = len(lst) somme_max = lst[0] im, jm = 0, 0 for i in range(n): for j in range(i, n): s = somme_sous_sequence(lst, i, j) if s > somme_max : somme_max = s im, jm = i, j return somme_max, im, jm</pre>	0,5	

3.a.	<table><tr><td>$S(i)$</td><td>-8</td><td>-4</td><td>6</td><td>14</td><td>8</td><td>18</td><td>14</td><td>10</td></tr></table>	$S(i)$	-8	-4	6	14	8	18	14	10	0,5	
$S(i)$	-8	-4	6	14	8	18	14	10				
3.b.	<pre>def pgsp2(lst): sommes_max = [lst[0]] for i in range(1, len(lst)): si = lst[i] if sommes_max [i-1]>0 : si += sommes_max [i-1] sommes_max.append(si) return max(sommes_max)</pre>	1										
3.c.	<p>La complexité temporelle de la solution de la question 3 est linéaire alors que celle obtenue dans la question 1 est quadratique. On attend au minimum que l'élève identifie la diminution significative du nombre de comparaisons.</p>	0,5										