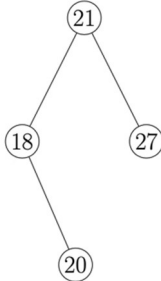
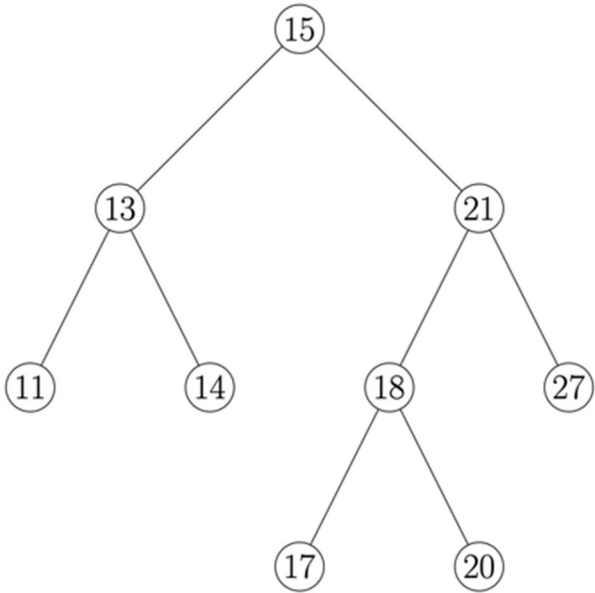


## **NUMERIQUE ET SCIENCES INFORMATIQUES – CORRIGÉ – JOUR 2**

Recommandations :

- Si un candidat traite plus de 3 exercices, le correcteur corrige l'ensemble de la copie et seules les trois meilleures notes des exercices seront retenues.
- Pour les parties de code rédigées en Python ou en SQL, les erreurs de syntaxe élémentaires (oubli des guillemets, des « : », etc.) ne seront pas pénalisées si, sur le fond, la question est correctement traitée.
- Une partie d'une question ou d'un code traitée correctement sera valorisée.

Exercice 1 (4 points)				
	Contenus et notions	Capacités exigibles	Éléments de réponses et commentaires	Barème
1.				1,75
a)	Algorithmes sur les arbres binaires et sur les arbres binaires de recherche.	Calculer la taille d'un arbre.	La taille de cet arbre est 8.	0,25
b)	Algorithmes sur les arbres binaires et sur les arbres binaires de recherche.	Calculer la hauteur d'un arbre.	La hauteur de cet arbre est 4	0,25
c)	Arbres binaires : nœuds, racines, feuilles, sous-arbres gauches, sous-arbres droits.		<p>Le sous-arbre droit du nœud de valeur 15 est:</p>  <pre> graph TD     21((21)) --- 18((18))     21 --- 27((27))     18 --- 20((20)) </pre>	0,25
d)	Algorithmes sur les arbres binaires et sur les arbres binaires de recherche.		L'arbre de la figure 1 est un arbre binaire de recherche car pour chaque nœud, la valeur de ce nœud est plus grande que les valeurs figurant dans son sous-arbre gauche et plus petite que les valeurs figurant dans son sous-arbre droit.	0,5

e)	Algorithmes sur les arbres binaires et sur les arbres binaires de recherche.	Insérer une clé.	<p>Le nouvel arbre est :</p>  <pre> graph TD     15((15)) --- 13((13))     15 --- 21((21))     13 --- 11((11))     13 --- 14((14))     21 --- 18((18))     21 --- 27((27))     18 --- 17((17))     18 --- 20((20)) </pre>	0,5
<b>2.</b>				<b>1</b>
a)	Vocabulaire de la programmation objet.	Accéder aux attributs et méthodes d'une classe.	Réponse (C)	0,5
b)	Vocabulaire de la programmation objet. Récursivité	Écrire un programme récursif.	<code>Noeud(ins(v, abr.gauche), abr.valeur, abr.droit)</code>	0,5

<b>3.</b>				<b>1,25</b>
a)	Récurtivité	Analyser le fonctionnement d'un programme récursif.	Le nombre d'appels à la fonction nb_sup est 17 en comptant l'appel initial. (16 accepté.)	0,5
b)	Récurtivité	Écrire un programme récursif.	<p>On peut proposer le code suivant :</p> <pre>def nb_sup (v, abr):     if abr is None:         return 0     else:         if abr.valeur &gt;= v:             return 1+nb_sup(v,abr.gauche)+nb_sup(v,abr.droit)         else:             return nb_sup(v,abr.droit)</pre> <p>ou encore:</p> <pre>def nb_sup(v, abr):     if abr is None:         return 0     else:         if abr.valeur &gt; v:             return 1+nb_sup(v,abr.gauche)+nb_sup(v,abr.droit)         elif abr.valeur == v:             return 1+nb_sup(v,abr.droit)         else:             return nb_sup(v,abr.droit)</pre>	0,75

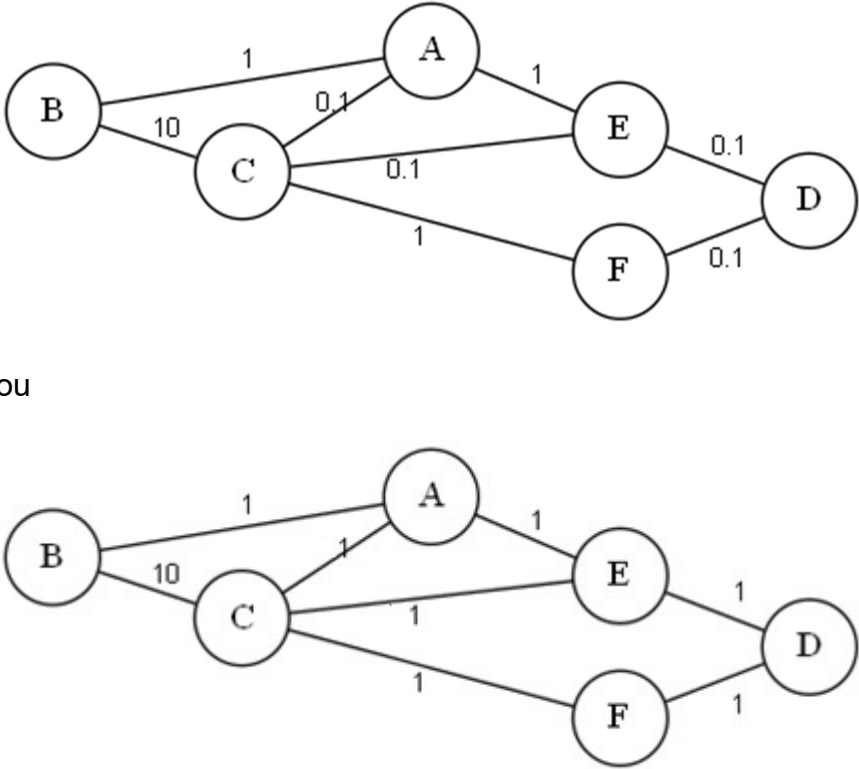
<b>Exercice 2 (4 points)</b>			
<b>Question</b>	<b>Capacités exigibles</b>	<b>Éléments de réponses et commentaires</b>	<b>Point(s)</b>
Question 1 : structures linéaires			<b>1,25</b>
1 a	Distinguer des structures par le jeu des méthodes qui les caractérisent.	498742 48742 4842 442 42 (tête de pile à gauche)	0,75
1 b	Distinguer des structures par le jeu des méthodes qui les caractérisent.	Pile B	0.5
Question2 : structures de données / Interface et implémentation.			<b>1</b>
2	Spécifier une structure de données par son interface.	<pre>if a%2 != c%2:     empiler(p, b) empiler(p, a)</pre>	1
Question3 : structures linéaires / Constructions élémentaires			<b>1,25</b>
3a		3	0,5
3b	Concevoir des solutions algorithmiques, mettre en évidence un corpus de constructions élémentaires.	<pre>while taille(p) &gt;= 3:     e = depiler(q)     empiler(p, e)</pre> <p>On tiendra compte de la cohérence avec la réponse de la question 3a</p>	0,75
Question 4 : structures linéaires / Récursivité / Constructions élémentaires			

			<b>0,5</b>
4	Concevoir des solutions algorithmiques	<p>Sans tenir compte des effets de bord sur la pile :</p> <pre>if taille(p) == taille(q) :     return jouer(q)</pre> <p>On attribuera le demi-point à toute réponse cohérente avec l'énoncé.</p>	0,5

Exercice 3 (4 points)					
Question	Contenus et notions	Capacités exigibles	Éléments de réponses et commentaires		Barème
1.					1
1.a	Architecture d'un réseau	Mettre en œuvre un réseau	Adresse du réseau: 192.168.1.0		0.25
1.b			Adresse diffusion: 192.168.1.255		0.25
1.c			Nombre de machines : $2^8 - 2 = 254$ machines		0.25
1.d			Toutes les adresses comprises entre 192.168.1.4 et 192.168.1.254 ou 192.168.1.2 (les adresses 192.168.1.1 et 192.168.1.3 étant déjà attribuées)		0.25
2.					0.5
2.a	Protocoles de routage (RIP)	Identifier les routes pouvant être empruntées par un paquet	A - E - D A - E - C - F - D A - B - C -E - D A - B - C -F - D A - C - E - D A - C - F - D  Dès lors que cinq chemins sont corrects, on attribuera le quart de point.		0.25
2.b			La multiplicité des chemins permet de pouvoir transmettre la trame en cas de coupure de la liaison, routeur en panne ou d'encombrement.		0.25
3.					1.25
3.a	Protocoles de routage (RIP)	Identifier la route empruntée par un paquet	Routeur A		0,25
destination			passé par		
B			B		
C			C		
D			E		
E			E		

			F	C								
3.b		Identifier la route empruntée par un paquet	Message de B – D: B – C – E – D									0.25
3.c		Déterminer le coût des routes	Routeur A		Routeur B		Routeur C					0.5
			destination	passé par	destination	passé par	destination	passé par				
			B	B	A	A	A	A				
			C	C	C	A	B	A				
			D	C	D	A	D	E ou F				
			E	C	E	A	E	E				
			F	C	F	A	F	F				
3.d		Identifier la route empruntée par un paquet	B – A – C - E – D ou B – A – C - F – D									0.25
4.			Pour ces questions, suivant les connaissances de l'élève, on acceptera les deux cas, la version décimale ou la version entière de la métrique.									1.25
4.a	Protocoles de routage (OSPF)	Déterminer le coût des routes	Coût Ethernet : 10 Coût Fast Ethernet : 1 Coût fibre : 0.1 ou 1  On attribuera le quart de point si l'élève laisse les réponses sous forme fractionnaire.									0.25




4.b			 <p>ou</p>	0.25
4.c		Déterminer le coût des routes	<p> <math>B - A - E - D</math> coût: <math>1 + 1 + 0.1 = 2.1</math> ou 3  <math>B - A - C - E - D</math> coût: <math>1 + 0.1 + 0.1 + 0.1 = 1.3</math> ou 4  <math>B - A - C - F - D</math> coût: <math>1 + 0.1 + 1 + 0.1 = 2.2</math> ou 4  <math>B - C - A - E - D</math> coût: <math>10 + 0.1 + 1 + 0.1 = 11.2</math> ou 13  <math>B - C - E - D</math> coût: <math>10 + 0.1 + 0.1 = 10.2</math> ou 12  <math>B - C - F - D</math> coût: <math>10 + 1 + 0.1 = 11.1</math> ou 12 </p>	0.5
4.d		Identifier la route empruntée par un paquet	<p> <math>B - A - C - E - D</math> coût: <math>1 + 0.1 + 0.1 + 0.1 = 1.3</math>  Ou <math>B - A - E - D</math> pour l'élève qui a obtenu 3  Car celui-ci a le coût le plus faible. </p>	0.25

### Exercice 4 (4 points)

	Contenus et notions	Capacités exigibles	Éléments de réponses et commentaires		Barème														
1.					1.25														
1.a	Langage SQL	Identifier les composants d'une requête.	<table><tr><td>Titre</td></tr><tr><td>Hey Jude</td></tr><tr><td>I Want To Hold Your Hand</td></tr></table> <p>Ou "tous les titres de chanson interprétées par les Beatles"</p>		Titre	Hey Jude	I Want To Hold Your Hand	0.25											
Titre																			
Hey Jude																			
I Want To Hold Your Hand																			
1.b	Langage SQL	Construire des requêtes d'interrogation à l'aide des clauses du langage SQL : SELECT, FROM, WHERE, JOIN.	<pre>SELECT nom FROM interpretes WHERE pays = "Angleterre";</pre>		0.25														
1.c	Langage SQL	Identifier les composants d'une requête.	<table><tr><td>titre</td><td>annee</td></tr><tr><td>I Want To Hold Your Hand</td><td>1963</td></tr><tr><td>Like A Rolling Stone</td><td>1965</td></tr><tr><td>Respect</td><td>1967</td></tr><tr><td>Hey Jude</td><td>1968</td></tr><tr><td>Imagine</td><td>1970</td></tr><tr><td>Smell Like Teen Spirit</td><td>1991</td></tr></table>	titre	annee	I Want To Hold Your Hand	1963	Like A Rolling Stone	1965	Respect	1967	Hey Jude	1968	Imagine	1970	Smell Like Teen Spirit	1991		0.25
titre	annee																		
I Want To Hold Your Hand	1963																		
Like A Rolling Stone	1965																		
Respect	1967																		
Hey Jude	1968																		
Imagine	1970																		
Smell Like Teen Spirit	1991																		

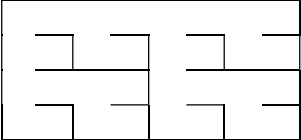
			Ou « cela renvoie les titres et date des œuvres triées de la date la plus ancienne à la plus récente »	
1.d	Langage SQL	Construire des requêtes d'interrogation à l'aide des clauses du langage SQL : SELECT, FROM, WHERE, JOIN.	<pre>SELECT COUNT(*) FROM morceaux;</pre> <p>Ou toute variante correcte.</p>	0.25
1.e	Langage SQL	Construire des requêtes d'interrogation à l'aide des clauses du langage SQL : SELECT, FROM, WHERE, JOIN.	<pre>SELECT titre FROM morceaux ORDER BY titre;</pre>	0.25
<b>2.</b>				<b>1.5</b>
2.a	Modèle relationnel	Identifier les concepts définissant le modèle relationnel.	La clé étrangère de la table morceaux est <code>id_interprete</code> car elle fait référence à la clé primaire de la table interpretes.	0.5

2.b	Base de données relationnelle	Savoir distinguer la structure d'une base de données de son contenu.	<table><tr><td colspan="2">morceaux</td></tr><tr><td>Id_morceau</td><td>Int</td></tr><tr><td>Titre</td><td>Char</td></tr><tr><td>Annee</td><td>Int</td></tr><tr><td>Id_interprete</td><td>Int</td></tr></table>  <table><tr><td colspan="2">interpretes</td></tr><tr><td>Id_interprete</td><td>Int</td></tr><tr><td>Nom</td><td>Char</td></tr><tr><td>pays</td><td>Char</td></tr></table> <p>On accepte aussi :</p> <pre>morceux(<u>Id_morceau</u> :Int, Titre :Char, Annee :Int, #Id_interprete :Int) interpretes(<u>Id_interprete</u> :Int, Nom:Char, pays :Char)</pre> <p><b>Les types ne sont pas attendus.</b></p>	morceaux		Id_morceau	Int	Titre	Char	Annee	Int	Id_interprete	Int	interpretes		Id_interprete	Int	Nom	Char	pays	Char	0.5
morceaux																						
Id_morceau	Int																					
Titre	Char																					
Annee	Int																					
Id_interprete	Int																					
interpretes																						
Id_interprete	Int																					
Nom	Char																					
pays	Char																					
2.c	Base de données relationnelle	Repérer des anomalies dans le schéma d'une base de données.	On essaie d'insérer une nouvelle ligne avec la valeur 1 comme clé primaire. Or cette valeur est déjà utilisée et une clé primaire ne peut pas avoir deux fois la même valeur. Il va donc y avoir une erreur de clés dupliquées.	0.5																		
3.				0,75																		
3.a	Langage SQL	Construire des requêtes d'insertion et de mise à jour à l'aide de : UPDATE, INSERT, DELETE.	<pre>UPDATE morceaux SET annee = 1971 WHERE id_morceau = 3 ;</pre> <p>Ou toute variante correcte.</p>	0.25																		

3.b	Langage SQL	Construire des requêtes d'insertion et de mise à jour à l'aide de : UPDATE, INSERT, DELETE.	INSERT INTO interpretes VALUES (6, 'The Who', 'Angleterre');  Ou toute variante correcte.	0.25
3.c	Langage SQL	Construire des requêtes d'insertion et de mise à jour à l'aide de : UPDATE, INSERT, DELETE.	INSERT INTO morceaux VALUES (7, 'My Generation', 1965, 6);  Ou toute variante correcte.	0.25
4.	Langage SQL	Construire des requêtes d'interrogation à l'aide des clauses du langage SQL : SELECT, FROM, WHERE, JOIN.	SELECT titre FROM morceaux JOIN interpretes ON morceaux.id_interprete = interpretes.id_interprete WHERE pays = "États-Unis";  Ou toute variante correcte.	0.5

Exercice 5 (4 points)				
	Contenus et notions	Capacités exigibles	Éléments de réponses et commentaires	Barème
1	Vocabulaire de la POO	Accéder aux méthodes d'une classe	<code>cellule = Cellule(True, False, True, True)</code>	0.25
2	Tableau indexé  Vocabulaire de la POO	Utiliser des tableaux de tableaux pour représenter des matrices : notation a [i] [j].  Itérer sur les éléments d'un tableau.  Accéder aux attributs et méthodes d'une classe.	<pre> for i in range(hauteur):      ligne = []      for j in range(longueur):          cellule = Cellule(True, True, True, True)          ligne.append(cellule)      grille.append(ligne) </pre>	1
3	Dictionnaires par clés et valeurs  Vocabulaire de la POO	Construire une entrée de dictionnaire.  Accéder aux attributs et méthodes d'une classe.	<code>cellule2.murs['S'] = False</code>	0.25

4	<p>Tableau indexé</p> <p>Constructions élémentaires</p> <p>Dictionnaires par clés et valeurs</p> <p>Vocabulaire de la POO</p>	<p>Lire et modifier les éléments d'un tableau grâce à leurs index.</p> <p>Affectation conditionnelle</p> <p>Construire une entrée de dictionnaire.</p> <p>Accéder aux attributs et méthodes d'une classe.</p>	<pre>elif c1_col - c2_col == 1 and c1_lig == c2_lig :      cellule1.murs['O'] = False      cellule2.murs['E'] = False</pre>	1
5	<p>Tableau indexé,</p> <p><b>Premiere / Langages et programmation</b></p> <p>Constructions élémentaires</p> <p>Vocabulaire de la POO</p>	<p>Itérer sur les éléments d'un tableau.</p> <p>Appels de fonctions</p> <p>Écrire la définition d'une classe.</p> <p>Accéder aux attributs et méthodes d'une classe.</p>	<pre>def creer_labyrinthe(self, ligne , colonne, haut, long):      if haut == 1 : #Cas de base          for k in range(long):              self.creer_passage(ligne, k, ligne, k+1)      elif long == 1: #Cas de base          for k in range(haut):</pre>	0.75

			<pre> self.creer_passage(k, colonne, k+1, colonne)  else: # Appels récursifs  On acceptera self.creer_passage(k+1, colonne, k, colonne) </pre>	
6	<b>Terminale</b> / <b>Algorithmique :</b> Méthode « diviser pour régner ».	Écrire un algorithme utilisant la méthode « diviser pour régner »		<b>0.75</b>  On valorisera toute ébauche de réponse montrant une compréhension de la méthode diviser pour régner.