

BACCALAURÉAT GÉNÉRAL

CORRECTION DE L'ÉPREUVE D'ENSEIGNEMENT DE SPÉCIALITÉ

SESSION 2024

NUMÉRIQUE ET SCIENCES INFORMATIQUES

Durée de l'épreuve : 3 heures 30

Le sujet est composé de trois exercices indépendants.

Questions	Contenu et notions	Capacités exigibles	Éléments de réponses et commentaires
1.	Citer un attribut et une méthode	N1 : donner les attributs et les méthodes d'une <code>class</code> .	attributs : <code>itineraire</code> , <code>longueur</code> , <code>largeur</code> , <code>grille</code> ; méthode : <code>remplir_grille</code>
2.	Quelles sont les valeurs contenues dans chacune des variables <code>a</code> et <code>b</code>	N1 : déterminer la valeur prise par une variable	<code>a = 7</code> et <code>b=4</code>
3.	Compléter la méthode <code>remplir_grille</code>	N2 : compléter une fonction simple Python à « trous »	<pre> 18 self.itinéraire 20 j = j + 1 22 i = i + 1 </pre>
4.	Écrire la méthode <code>get_dimension</code>	N2 : Écrire une méthode de class simple	<pre> def get_dimension(self) : return (self.longueur, self.largeur) </pre>
5.	Écrire la méthode <code>tracer_chemin</code>	N2 : Écrire une méthode de class	<pre> def tracer_chemin(self): self.remplir_grille() for ligne in self.grille: print(ligne) </pre>
6.		N3 : compléter une grande partie du code d'une fonction	<pre> while i != m and j != n: direction = choice(['B', 'D']) itineraire = itineraire + direction if direction == 'B': i = i + 1 else: j = j + 1 </pre>
7.	Justifier que $N(1, n)$	N2	Un tableau de dimension $1 \times n$ est une ligne. Il y a un seul chemin possible : se déplacer n fois à droite.
8.	Justifier que $N(m, n) = N(m - 1, n) + N(m, n - 1)$.	N2	Pour arriver à la cellule de coordonnées (m, n) soit le dernier déplacement est B, soit D. On en déduit que pour atteindre la case (n, m) , il faut venir soit de $((n - 1), m)$ soit de $(n, (m - 1))$. Le nombre de chemins pour atteindre (n, m) est donc la somme du nombre de chemins pour atteindre $((n - 1), m)$ et $(n, (m - 1))$.

Exercice 1	6 points		
9.	Écrire une fonction récursive	N3 : écrire un programme non classique à partir de rien.	<pre>def nombre_chemins(m, n): if m == 0 or n == 0 : return 1 return nombre_chemins(m-1, n) + nombre_chemins(m, n-1)</pre>

Exercice 2		6 points	
Questions	Contenu et notions	Capacités exigibles	Éléments de réponses et commentaires
1	Système d'exploitation : Utiliser les commandes de base en ligne de commande (programme de première)	N1	<code>ls documents</code> ou réponse équivalente
2	Système d'exploitation : Utiliser les commandes de base en ligne de commande (programme de première)	N2	La commande déplace le dossier <code>multimedia</code> et tout son contenu dans le dossier <code>documents</code>
3	Arbres. Identifier des situations nécessitant une structure de données arborescente	N1	l'arborescence de fichiers de la figure 1 n'est pas un arbre binaire (le dossier Documents possède trois fils). Or la modélisation proposée représente un arbre binaire où chaque dossier ne peut avoir au plus qu'un fils gauche et un fils droit. Cette modélisation n'est donc pas adaptée
4	Algorithme sur les arbres binaires : parcourir un arbre de différente façon	N1	parcours (en profondeur) en ordre préfixe; en profondeur non exigé
5	Algorithme sur les arbres binaires : parcourir un arbre de différente façon	N2	home - documents - multimedia - cours - administratif - personnel - images - videos - films
6	Structure de données : implémentation	N2	<pre>def est_vide(self): return self.fils == []</pre>
7	Vocabulaire de la programmation objet	N2	<pre>films = Dossier("films", []) videos = Dossier("videos", [films]) images = Dossier("images", []) multimedia = Dossier("multimedia", [images, videos])</pre>

Exercice 2		6 points	
8	Structure de données : implémentation	N3	<pre>def parcours(self): print(self.nom) for f in self.fils: f.parcours()</pre>
9	Analyse des algorithmes. Terminaison d'un algorithme récursif.	N3	Les appels récursifs à la méthode parcours se font toujours sur une arborescence de taille/hauteur strictement plus petite, donc cet algorithme récursif termine. Remarquons que le cas de base est le cas d'une feuille : la liste fils est vide et il n'y a plus d'appels récursifs.
10	Algorithme sur les arbres binaires : parcourir un arbre de différente façon	N2	<pre>def parcours(self): for f in self.fils: f.parcours() print(self.nom)</pre>
11	Recurivite, système d'exploitation (programme de première)	N2	la méthode parcours est récursive et affiche tous les descendants alors que la commande ls n'affiche que les fils (descendants directs).
12	Structure de données : implémentation	N2	<pre>def mkdir(self, nom_dossier_fils): nouveau_dossier = Dossier(nom_dossier_fils, []) self.fils.append(nouveau_dossier)</pre>
13	Recherche d'un élément dans un arbre	N3	<pre>def contient(self, nom_dossier): if self.nom == nom_dossier: return True for f in self.fils: if f.contient(nom_dossier): return True return False</pre>

Exercice 2		6 points	
14	Choisir une structure de donnée adaptée à la situation à modéliser	N3	On peut parcourir l'arborescence comme précédemment (Q13) en prenant soin de mémoriser le parent pour parcourir les fils. Voici un programme Python (non demandé) permettant de résoudre ce problème. <pre>def parent(dossier, racine): def cherche(courant, pere): <i>"""Cherche `dossier` dans l'arborescence de racine `courant` ayant pour parent `pere`."""</i> if courant == dossier: return pere for fils in courant.fils: res = cherche(fils, courant) if res is not None: return res return None return cherche(racine, None)</pre>
15	Choisir une structure de donnée adaptée à la situation à modéliser	N3	On ajoute un attribut parent qui contient le repertoire parent du dossier. On peut ainsi y accéder en coût constant et cette approche est plus efficace que l'approche précédente qui nécessitait, dans le pire cas, de parcourir tout l'arbre.

Exercice 3		8 points	
Questions	Contenu et notions	Capacités exigibles	Éléments de réponses et commentaires
1	Décimal à binaire	N1	On écrit 11 en base 2, ainsi $11 = 2^3 + 2^1 + 2^0$, il s'agit donc d'un conducteur, chef d'équipe et équipier.
2	Binaire à décimal	N1	Un chef d'agrès conducteur est conducteur, chef d'agrès, chef d'équipe et équipier, ainsi sa qualification sera codé par $2^3 + 2^2 + 2^1 + 2^0 = 15$.
3	Binaire, réflexion	N1	4 code pour l'aptitude chef d'agrès seule, mais un chef d'agrès possède nécessairement l'aptitude équipier et chef d'équipe, ainsi il faut ajouter les autres aptitudes induites et sa qualification sera a minima 7.
4	Binaire, réflexion	N1	Il reste 4 bits non utilisés, ainsi on peut encore encoder 4 aptitudes.
5	représentation d'information en machine	N2	Il y a quatre aptitudes codées sur dix caractères chacun codés sur 1 octet, soit 40 octets nécessaires. Avec le codage choisi, la qualification utilise 1 octet, le gain est donc de 39 octets, soit $39/40 \approx 98\%$.
6	SQL	N1	Une clef primaire d'une table est un ensemble d'attributs qui identifie de manière certaine un objet de la table. Une clef étrangère est un ensemble d'attributs qui est la clef primaire d'une autre table.
7	SQL	N1	La table agres ne possède pas d'entrée avec l'identifiant 1. La contrainte définie dans la table moyen par la clef étrangère idagres lève donc une erreur.
8	SQL	N1	<code>UPDATE Intervention SET heure='10:44:06' WHERE id=3;</code>
9	SQL	N1	La requête affiche Charlot, Red et Kevin. C'est à dire les noms des personnels inactifs.
10	SQL	N2	[Ajouter à la correction] <code>SELECT 'nom'</code> <code>FROM personnel</code> <code>WHERE qualif >= 16 AND actif = 1 ;</code>
11	SQL	N2	La requête A renvoie 2, tandis que la requête B renvoie 1. La requête A renvoie le nombre d'agrès prêts à partir en intervention le 27 mars 2024, tandis que la requête B renvoie le nombre d'agrès partis en intervention ce jour-là.

Exercice 3	8 points		
12	SQL	N2	Le nom (sans répétition) des chefs d'agres assignés à un véhicule le 15 février 2024 est obtenu avec la requête <pre>SELECT DISTINCT nom FROM personnel AS p INNER JOIN agres AS a ON a.idchefagres = p.matricule WHERE a.jour = '2024-02-15' ;</pre>
13	SQL	N2	Le nom (sans répétition) des chefs d'agres engagés en intervention le 11 juin 2024 est obtenu avec la requête [Requête difficile] <pre>SELECT DISTINCT nom FROM personnel AS p INNER JOIN agres AS a ON a.idchefagres = p.matricule INNER JOIN moyen AS m ON m.idagres = a.id WHERE a.jour = '2024-06-11' ;</pre>
14	SQL	N2	[Requête faisable] <pre>ORDER BY nb LIMIT 1;</pre>