Bac Général Maths + NSI (Mathématiques et Numérique et sciences informatiques)

CORRIGÉ NON OFFICIEL

Proposition de corrigé rédigée par FormaV avec l'aide d'outils d'intelligence artificielle. Elle peut contenir des erreurs et ne remplace ni le corrigé ni le barème officiels de l'examen.

Les sujets officiels des sessions passées et nos sujets d'entraînement sont classés sur https://bac-maths-nsi.fr/annales-sujets-et-corriges

Document mis en ligne par FormaV®. Reproduction ou diffusion sans autorisation interdite.

BACCALAURÉAT GÉNÉRAL • CORRIGÉ DÉTAILLÉ

Numérique et sciences informatiques (NSI)

Session 2023 • Métropole • Jour 1

Code sujet **Non indiqué sur les pages fournies** Non indiquée sur les pages fournies

Proposition de corrigé pédagogique • FormaV

Se repérer dans le corrigé

Exercice 1 — Bases de données et SQL
8 réponses • sujet p. 1–1

Exercice 2 — Réseaux et protocoles de routage
7 réponses • sujet p. 2–3

Exercice 3 — Algorithmique et programmation orientée objet : coloration de régions
12 réponses • sujet p. 4–6

Les numéros, parties et pages renvoient au sujet original. Chaque réponse rappelle la consigne, explique la démarche et donne le résultat.

FormaV - document d'entraînement non officiel

Exercice 1

3 points

Bases de données et SQL

Question 1. (a)

Sujet p. 1

> Dans cette question sur le modèle relationnel, donner la caractéristique qu'un attribut doit avoir pour être choisi comme clé primaire.

Une clé primaire sert à identifier sans ambiguïté chaque ligne d'une relation. La valeur de l'attribut choisi doit donc être unique pour chaque ligne et ne peut pas être absente.

> **RÉPONSE À RETENIR**
>
> Un attribut choisi comme clé primaire doit avoir une valeur unique et non nulle pour chaque enregistrement.

Question 1. (b)

Sujet p. 1

> Expliquer le rôle des deux clés étrangères de la relation Commandes.

Dans `Commandes`, `#idClient` référence l'identifiant d'un client dans `Clients` : il associe la commande à son client. `#idMeuble` référence l'identifiant d'un meuble dans `Meubles` : il indique quel meuble est concerné par la commande.

> **RÉPONSE À RETENIR**
>
> `#idClient` référence `Clients(id)` et `#idMeuble` référence `Meubles(id)`.

Question 1. (c)

Sujet p. 1

> Donner le schéma relationnel de la relation Meubles en précisant la clé primaire et les éventuelles clés étrangères.

Les attributs de `Meubles` sont ceux du tableau de la figure 1. L'attribut `id` est sa clé primaire. Aucune clé étrangère n'est indiquée pour cette relation dans le sujet.

> **RÉPONSE À RETENIR**
>
> `Meubles(id, intitule, prix, stock, description)`, avec `id` comme clé primaire et aucune clé étrangère indiquée.

Question 2.

Sujet p. 1

> En vous basant uniquement sur les données du tableau de la figure 1, donner le résultat de la requête SQL : SELECT id, stock, description FROM Meubles WHERE intitule = 'skap'.

Dans l'extrait de la figure 1, les deux lignes dont `intitule` vaut `'skap'` ont pour identifiants 62 et 63. La requête ne sélectionne que les colonnes `id`, `stock` et `description`, dans cet ordre.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

id	stock	description
62	2	Armoire blanche 3 portes
63	3	Armoire noire 3 portes

Question 3.

Sujet p. 1

> Donner la requête SQL permettant d'afficher les noms et prénoms des clients habitant à Paris.

On sélectionne les colonnes `nom` et `prenom` de `Clients`, puis on ne conserve que les lignes dont la ville est Paris.

RÉPONSE À RETENIR

```
SELECT nom, prenom
FROM Clients
WHERE ville = 'Paris';
```

Question 4.

Sujet p. 1

> Le magasin vient de recevoir des meubles dont l'intitulé est 'hylla' et dont l'attribut id dans la relation Meubles vaut 98. Le stock de ces meubles est alors de 50. Recopier et compléter la requête SQL fournie (UPDATE, SET, WHERE) afin de mettre à jour la base de données.

On met à jour la relation `Meubles` en affectant 50 à `stock`. La condition sur l'identifiant 98 cible le meuble concerné et limite la modification à sa ligne.

RÉPONSE À RETENIR

```
UPDATE Meubles
SET stock = 50
WHERE id = 98;
```

Question 5.

Sujet p. 1

> Le magasin vient d'ajouter à son catalogue un nouveau meuble dont les caractéristiques sont id 65, intitulé 'matta', prix 95.99, stock 25 et description 'Tapis vert à pois rouges'. Donner la requête SQL qui permet d'ajouter cet article à la relation Meubles.

`INSERT INTO` ajoute une ligne à la relation. Les valeurs de `VALUES` correspondent, dans l'ordre, aux attributs indiqués après le nom de la relation.

RÉPONSE À RETENIR

```
INSERT INTO Meubles (id, intitule, prix, stock, description)
VALUES (65, 'matta', 95.99, 25, 'Tapis vert à pois rouges');
```

Question 6.

Sujet p. 1

> Donner la requête SQL permettant de récupérer le nom et le prénom des différents clients qui ont passé une commande le 30 avril 2021. Dans la relation Commandes, les dates sont des chaînes de caractères, par exemple '21/08/2002'.

La jointure associe chaque commande à son client grâce à `Clients.id = Commandes.idClient`. La date est comparée à la chaîne au format jour/mois/année. La sous-requête ne conserve qu'une occurrence de chaque identifiant de client ayant commandé ce jour-là : un client ayant plusieurs commandes ce jour-là n'est donc renvoyé qu'une fois. Deux clients distincts portant les mêmes nom et prénom restent deux clients distincts dans le résultat.

RÉPONSE À RETENIR

```
SELECT Clients.nom, Clients.prenom
FROM Clients
JOIN (
    SELECT DISTINCT idClient
    FROM Commandes
    WHERE date = '30/04/2021'
) AS ClientsDuJour ON Clients.id = ClientsDuJour.idClient;
```

Exercice 2

3 points

Réseaux et protocoles de routage

Question 1.

Sujet p. 2

> Justifier brièvement que le réseau de l'entreprise sera encore fonctionnel même si l'une des liaisons (Liaison1, Liaison2, …, Liaison6) est coupée.

Les liaisons entre routeurs forment des cycles : R1, R2 et R3 forment un premier cycle ; R2, R3, R4 et R5 en forment un autre. Chaque liaison numérotée appartient donc à un cycle : si elle est coupée, on peut encore relier ses deux extrémités en empruntant les autres liaisons du cycle. Les sites restent raccordés à leurs routeurs par leurs liaisons directes.

RÉPONSE À RETENIR

Le réseau reste connecté si l'une des liaisons Liaison1 à Liaison6 est coupée : il existe toujours un chemin de remplacement entre les routeurs concernés.

Question 2.

Sujet p. 2

> Les tables de routage des routeurs étant configurées manuellement comme dans la figure 2, indiquer le chemin suivi par les paquets lorsqu'une information est envoyée de SiteB à SiteC.

Le paquet arrive d'abord au routeur R2, auquel SiteB est directement raccordé. Pour atteindre SiteC, la table de R2 indique R3 comme routeur suivant. La table de R3 indique ensuite R4, celle de R4 indique R5, et la table de R5 indique que SiteC est local.

RÉPONSE À RETENIR

Chemin suivi : SiteB → R2 → R3 → R4 → R5 → SiteC.

Question 3.

Sujet p. 2

> Afin d'optimiser la maintenance du réseau, les tables de routage sont configurées automatiquement en utilisant le protocole RIP. Pour RIP, le chemin est construit de façon à minimiser le nombre de routeurs traversés. Recopier et compléter la table de routage RIP du routeur R1 (figure 3).

RIP minimise le nombre de routeurs traversés, sans tenir compte des coûts des liaisons. Depuis R1, SiteB est atteint par R2 en un saut. SiteC est atteint au plus court par R2 puis R5, soit deux sauts ; SiteD est atteint par R3 puis R4, soit deux sauts. Dans la colonne « Suivant », on inscrit le premier routeur après R1.

RÉPONSE À RETENIR

Destination	Suivant	Nombre de sauts
SiteA	Local	0
SiteB	R2	1
SiteC	R2	2
SiteD	R3	2

FormaV - document d'entraînement non officiel

Question 4.

Sujet p. 3

> La liaison Liaison2 a un débit très inférieur aux autres liaisons. Expliquer pourquoi le choix du protocole RIP n'est pas judicieux.

RIP minimise le nombre de routeurs traversés, mais ne prend pas en compte le débit des liaisons. Il peut donc choisir un chemin ayant moins de sauts, même s'il emprunte une liaison très lente. Par exemple, pour aller de R1 à SiteC, le chemin R1 → R2 → R5 utilise deux sauts et passe par Liaison2. Le chemin R1 → R3 → R4 → R5 utilise trois sauts, mais évite cette liaison lente.

RÉPONSE À RETENIR

RIP n'est pas judicieux ici, car il peut privilégier un chemin plus court en nombre de sauts qui passe par Liaison2, malgré son débit très inférieur aux autres.

Question 5. (a)

Sujet p. 3

> Pour le protocole OSPF, le chemin est construit de façon à minimiser le coût. Le coût d'une liaison est donné par la relation coût = $10^{10}/D$, où D est son débit en bit/s. En utilisant les coûts des liaisons donnés dans la figure 4, indiquer la liaison dont le débit est le plus faible.

La relation donnée est coût = $\frac{10^{10}}{D}$. Comme le numérateur est fixe et positif, un coût plus élevé correspond à un débit plus faible. Le coût maximal du tableau est 1 000 000, celui de Liaison2.

RÉPONSE À RETENIR

Le débit le plus faible est celui de Liaison2.

Question 5. (b)

Sujet p. 2, 3

> Donner la liste des quatre chemins possibles pour aller de SiteA à SiteC sans utiliser deux fois le même routeur et calculer le coût de chacun de ces chemins. Le coût d'un chemin est la somme des coûts des liaisons à parcourir ; par convention, le coût d'une liaison directe entre un routeur et un site est 0.

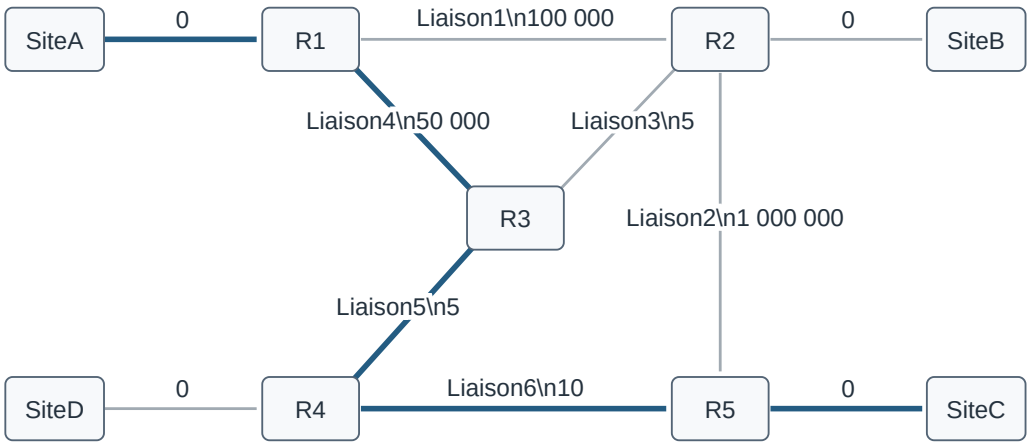
SiteA est directement relié à R1 et SiteC à R5 ; ces liaisons directes ont un coût nul. Il suffit donc de rechercher les chemins simples de R1 à R5 dans le graphe. On additionne les coûts des liaisons numérotées de chacun des quatre chemins :

- SiteA → R1 → R2 → R5 → SiteC : Liaison1 et Liaison2, soit 100 000 + 1 000 000 = 1 100 000.
- SiteA → R1 → R2 → R3 → R4 → R5 → SiteC : Liaison1, Liaison3, Liaison5 et Liaison6, soit 100 000 + 5 + 5 + 10 = 100 020.
- SiteA → R1 → R3 → R2 → R5 → SiteC : Liaison4, Liaison3 et Liaison2, soit 50 000 + 5 + 1 000 000 = 1 050 005.
- SiteA → R1 → R3 → R4 → R5 → SiteC : Liaison4, Liaison5 et Liaison6, soit 50 000 + 5 + 10 = 50 015.

Ces quatre chemins sont tous les chemins de R1 à R5 qui ne réutilisent pas un routeur.

FormaV - document d'entraînement non officiel

Réseau pondéré et chemin minimal de SiteA à SiteC



Les six liaisons numérotées portent exactement les coûts de la figure 4. Les liaisons directes entre un site et son routeur coûtent 0. Le chemin mis en évidence a un coût total de 50 015.

RÉPONSE À RETENIR

Chemin	Calcul du coût	Coût total
SiteA → R1 → R2 → R5 → SiteC	100 000 + 1 000 000	1 100 000
SiteA → R1 → R2 → R3 → R4 → R5 → SiteC	100 000 + 5 + 5 + 10	100 020
SiteA → R1 → R3 → R2 → R5 → SiteC	50 000 + 5 + 1 000 000	1 050 005
SiteA → R1 → R3 → R4 → R5 → SiteC	50 000 + 5 + 10	50 015

Le chemin de coût minimal est SiteA → R1 → R3 → R4 → R5 → SiteC, de coût 50 015.

Question 5. (c)

Sujet p. 2, 3

> Recopier et compléter la table de routage OSPF du routeur R1 (figure 5).

Pour chaque destination, on compare les coûts des chemins possibles depuis R1. Vers SiteB, la liaison directe Liaison1 coûte 100 000 ; passer par R3 puis R2 coûte Liaison4 + Liaison3 = 50 000 + 5 = 50 005. Le chemin minimal commence donc par R3. Vers SiteC, le chemin minimal passe par R3, R4 et R5 et coûte Liaison4 + Liaison5 + Liaison6 = 50 000 + 5 + 10 = 50 015. Vers SiteD, le chemin minimal passe par R3 puis R4 et coûte Liaison4 + Liaison5 = 50 000 + 5 = 50 005. Les liaisons directes entre routeur et site coûtent 0 ; « Suivant » désigne le premier routeur après R1.

RÉPONSE À RETENIR

Destination	Suivant	Coût total du chemin
SiteA	Local	0
SiteB	R3	50 005
SiteC	R3	50 015
SiteD	R3	50 005

FormaV - document d'entraînement non officiel

Exercice 3

6 points

Algorithmique et programmation orientée objet : coloration de régions

Partie 1

Question 1.

Sujet p. 4

> En vous appuyant sur l'extrait de code de la classe Region, associer les noms nom, tab_voisines, tab_couleurs_disponibles et couleur_attribuee au terme qui leur correspond parmi : objet, attribut, méthode ou classe.

Les quatre noms désignent des données associées à une instance de la classe `Region`. En programmation orientée objet, ces données sont des attributs. Aucun de ces noms ne désigne une méthode, une classe ou un objet.

RÉPONSE À RETENIR

`nom`, `tab_voisines`, `tab_couleurs_disponibles` et `couleur_attribuee` sont tous des attributs.

Question 2.

Sujet p. 4

> Indiquer le type du paramètre nom_region de la méthode __init__ de la classe Region.

La documentation du constructeur précise que le paramètre `nom_region` est de type `str`.

RÉPONSE À RETENIR

`nom_region` est de type `str` (chaîne de caractères).

Question 3.

Sujet p. 5

> Donner une instruction permettant de créer une instance nommée ge de la classe Region correspondant à la région dont le nom est « Grand Est ».

On crée une instance en appelant la classe `Region` avec le nom de la région comme argument. Le constructeur place ce nom dans l'attribut `nom`.

RÉPONSE À RETENIR

```
ge = Region("Grand Est")
```

Question 4.

Sujet p. 4, 5

> Recopier et compléter la ligne 6 de la méthode de la classe Region renvoie_premiere_couleur_disponible, qui renvoie la première couleur du tableau des couleurs disponibles, supposé non vide.

Le premier élément d'une liste Python est à l'indice `0`. Il suffit donc de renvoyer l'élément d'indice 0 du tableau des couleurs disponibles. Avec le tableau donné à l'initialisation, ce serait `'rouge'`.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

```
return self.tab_couleurs_disponibles[0]
```

Question 5.

Sujet p. 4, 5

Recopier et compléter la ligne 6 de la méthode de la classe Region renvoie_nb_voisines, qui renvoie le nombre de régions voisines.

`len` renvoie le nombre d'éléments d'un tableau. Pour obtenir le nombre de voisines, on l'applique à `self.tab_voisines`.

RÉPONSE À RETENIR

```
return len(self.tab_voisines)
```

Question 6.

Sujet p. 4, 5

Compléter à partir de la ligne 6 la méthode est_coloriee de la classe Region. Elle renvoie True si une couleur a été attribuée à cette région et False sinon.

À l'initialisation, `couleur_attribuee` vaut `None`. Une région est coloriée si et seulement si cet attribut ne vaut pas `None`. Cette comparaison renvoie un booléen.

RÉPONSE À RETENIR

```
return self.couleur_attribuee is not None
```

Question 7.

Sujet p. 4, 5

Compléter à partir de la ligne 8 la méthode retire_couleur de la classe Region. Elle retire couleur du tableau des couleurs disponibles de la région si elle s'y trouve et ne fait rien sinon.

La méthode `remove` provoque une erreur si l'élément est absent. Il faut donc tester l'appartenance de la couleur avant de la retirer. Par exemple, retirer `'rouge'` du tableau initial laisse `['vert', 'bleu', 'jaune', 'orange', 'marron']`. Si la couleur n'est pas présente, le tableau reste inchangé.

RÉPONSE À RETENIR

```
if couleur in self.tab_couleurs_disponibles:
    self.tab_couleurs_disponibles.remove(couleur)
```

Question 8.

Sujet p. 4, 5

> Compléter à partir de la ligne 7, en utilisant une boucle, la méthode est_voisine de la classe Region. Elle renvoie True si la région passée en paramètre est une voisine et False sinon.

On parcourt la liste des voisines. Si l'une d'elles est l'instance passée en paramètre, la méthode renvoie `True`. Si la boucle se termine sans trouver cette instance, la méthode renvoie `False`. La comparaison par identité convient ici, car aucune méthode de comparaison personnalisée n'est définie dans le sujet.

Avec l'instance `ge` créée à la question 3, le tableau des voisines est vide à l'initialisation : la boucle n'effectue aucun passage et la méthode renvoie `False`.

RÉPONSE À RETENIR

```
for voisine in self.tab_voisines:
    if voisine is region:
        return True
return False
```

Partie 2

Question 9.

Sujet p. 6

> Recopier et compléter à partir de la ligne 6 la méthode renvoie_tab_regions_non_coloriees de la classe Pays. Elle renvoie un tableau dont les éléments sont les régions du pays sans couleur attribuée.

On parcourt `self.tab_regions` et on ajoute à une nouvelle liste chaque région pour laquelle `est_coloriee()` renvoie `False`. Le tableau initial des régions n'est pas modifié.

Pour une instance nouvellement créée telle que `ge`, `couleur_attribuee` vaut `None`, donc `est_coloriee()` renvoie `False`. Si `ge` figure dans `self.tab_regions`, elle sera donc ajoutée à la liste renvoyée. Le sujet ne donne pas le contenu de `self.tab_regions` : la liste complète dépend des données de l'instance.

RÉPONSE À RETENIR

```
def renvoie_tab_regions_non_coloriees(self):
    regions_non_coloriees = []
    for region in self.tab_regions:
        if not region.est_coloriee():
            regions_non_coloriees.append(region)
    return regions_non_coloriees
```

Question 10. (a)

Sujet p. 6

> Expliquer dans quel cas la méthode renvoie_max de la classe Pays renvoie None.

`region_max` est initialisée à `None` et n'est modifiée que si la boucle parcourt au moins une région non coloriée. Si la liste des régions non coloriées est vide, la boucle ne s'exécute pas et la méthode renvoie donc `None`.

RÉPONSE À RETENIR

`renvoie_max` renvoie `None` lorsqu'il n'existe aucune région non coloriée, c'est-à-dire lorsque toutes les régions sont coloriées.

Question 10. (b)

Sujet p. 6

> Dans le cas où la méthode renvoie_max ne renvoie pas None, indiquer les deux particularités de la région renvoyée.

La méthode parcourt uniquement les régions non coloriées. Elle ne remplace la région retenue que si une région possède strictement plus de voisines que le maximum courant. La région renvoyée est donc non coloriée et possède le plus grand nombre de voisines parmi les régions non coloriées. En cas d'égalité, elle conserve la première région rencontrée ayant ce nombre maximal de voisines.

RÉPONSE À RETENIR

- La région renvoyée n'est pas coloriée.
- Elle possède le plus grand nombre de voisines parmi les régions non coloriées.

Question 11.

Sujet p. 4, 5, 6

> Coder la méthode colorie(self) de la classe Pays, qui choisit une couleur pour chaque région du pays de la façon suivante : récupérer la région non coloriée qui possède le plus de voisines ; tant que cette région existe, lui attribuer la première couleur disponible dans son tableau de couleurs disponibles ; pour chaque région voisine, retirer cette couleur de son tableau de couleurs disponibles si elle y est présente ; puis récupérer à nouveau la région non coloriée qui possède le plus de voisines.

À chaque passage, la méthode sélectionne une région non coloriée ayant le plus grand nombre de voisines, lui attribue sa première couleur disponible, puis retire cette couleur des tableaux de couleurs disponibles de ses voisines lorsqu'elle y figure. Elle répète ensuite la recherche. Chaque passage colore une région distincte : si les appels nécessaires réussissent, il y a au plus autant de passages que de régions, puis `renvoie_max` renvoie `None`.

Trace sur les valeurs d'initialisation fournies : si une région nouvellement créée est sélectionnée avant toute suppression de couleur, son tableau commence par `'rouge'`. La méthode lui attribue donc cette couleur et la retire du tableau de chaque voisine où elle est présente. Le sujet ne fournit pas la liste des régions du pays ni leurs voisinages ; on ne peut pas détailler une exécution complète avec des régions précises.

La méthode suppose que la région sélectionnée possède au moins une couleur disponible : si son tableau est vide, la méthode de la question 4 ne peut pas renvoyer de première couleur. Enfin, cette stratégie suit la règle demandée, mais ne garantit pas en général de trouver une coloration utilisant le nombre minimal de couleurs.

RÉPONSE À RETENIR

```
def colorie(self):
    region = self.renvoie_max()
    while region is not None:
        couleur = region.renvoie_premiere_couleur_disponible()
        region.couleur_attribuee = couleur
        for voisine in region.tab_voisines:
            voisine.retire_couleur(couleur)
        region = self.renvoie_max()
```

Sujet : https://aeif.fr/wp-content/uploads/2025/07/Metropole_mars_2023_1.pdf
© FormaV EI • Proposition de correction, distincte d'un corrigé officiel.

Copyright © 2027 FormaV®. Tous droits réservés.

FormaV® est une marque déposée auprès de l'INPI sous le numéro 5227897.

Ce document a été élaboré par FormaV® avec le plus grand soin afin d'accompagner chaque apprenant vers la réussite de ses examens. Son contenu (textes, graphiques, méthodologies, tableaux, exercices, concepts, mises en forme) constitue une œuvre protégée par le droit d'auteur.

Toute copie, partage, reproduction, diffusion ou mise à disposition, même partielle, gratuite ou payante, est strictement interdite sans accord préalable et écrit de FormaV®, conformément aux articles L.111-1 et suivants du Code de la propriété intellectuelle. Dans une logique anti-plagiat, FormaV® se réserve le droit de vérifier toute utilisation illicite, y compris sur les plateformes en ligne ou sites tiers.

En utilisant ce document, vous vous engagez à respecter ces règles et à préserver l'intégrité du travail fourni. La consultation de ce document est strictement personnelle.

Merci de respecter le travail accompli afin de permettre la création continue de ressources pédagogiques fiables et accessibles.