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

Session 2024 • Centres étrangers • Jour 1

Code sujet **24-NSIJ1G11** 3 heures 30

Proposition de corrigé pédagogique • FormaV

Se repérer dans le corrigé

Exercice 1 — Déplacements aléatoires d’un virus dans un réseau informatique
9 réponses • sujet p. 2–3

Exercice 2 — Adresses IP et protocoles de routage
10 réponses • sujet p. 4–6

Exercice 3 — Base de données d’un camping, requêtes SQL et facturation en Python
15 réponses • sujet p. 7–11

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

6 points

Déplacements aléatoires d'un virus dans un réseau informatique

Question 1.

Sujet p. 2

> On représente le réseau informatique de la figure 1, composé des ordinateurs numérotés 0, 1, 2, 3 et 4, par un graphe stocké sous forme de listes de voisins. Compléter la définition de la variable voisins donnée dans le sujet.

Le graphe de la figure est non orienté : chaque voisin doit donc apparaître dans les listes des deux sommets reliés. Le sommet 0 est relié aux sommets 1, 2, 3 et 4 ; les arêtes supplémentaires sont 1–2 et 1–3. Ainsi, les voisins de 3 sont 0 et 1, et le seul voisin de 4 est 0.

RÉPONSE À RETENIR

```
voisins = [[1, 2, 3, 4],
           [0, 2, 3],
           [0, 1],
           [0, 1],
           [0]]
```

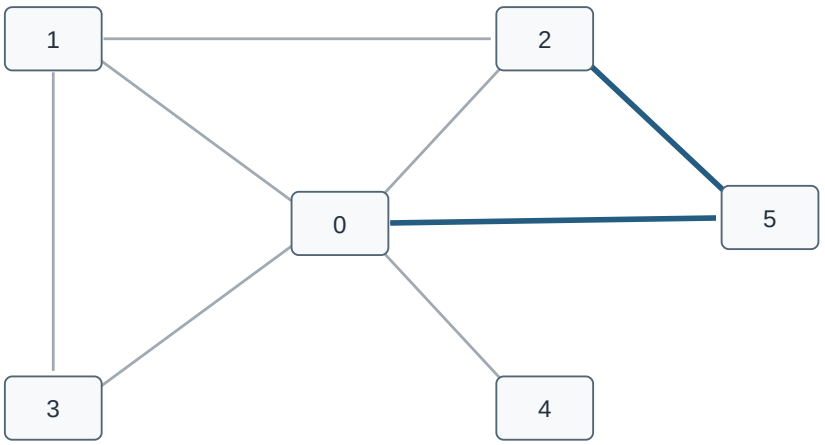
Question 2.

Sujet p. 2

> On ajoute au réseau actuel un sixième ordinateur, numéroté 5. Cet ordinateur n'est accessible que des ordinateurs numérotés 0 et 2. Dessiner le nouveau graphe.

On conserve les arêtes du graphe initial et on ajoute le sommet 5, relié exactement aux sommets 0 et 2. Le graphe étant non orienté, cela ajoute les arêtes 0–5 et 2–5.

Réseau après l'ajout de l'ordinateur 5



Les deux arêtes mises en évidence sont celles ajoutées avec l'ordinateur 5.

RÉPONSE À RETENIR

Le nouveau graphe comporte les arêtes 0–1, 0–2, 0–3, 0–4, 1–2, 1–3, 0–5 et 2–5.

Question 3.

Sujet p. 2

Donner la nouvelle définition de la variable voisins correspondant au réseau après l'ajout de l'ordinateur 5.

Les listes des sommets 0 et 2 gagnent chacune le sommet 5. La liste de 5 contient ses deux seuls voisins, 0 et 2. Les autres listes restent inchangées.

RÉPONSE À RETENIR

```
voisins = [[1, 2, 3, 4, 5],
           [0, 2, 3],
           [0, 1, 5],
           [0, 1],
           [0],
           [0, 2]]
```

Question 4.

Sujet p. 2

Compléter la fonction voisin_alea qui prend en paramètre un graphe voisins sous forme de listes de voisins et un entier s représentant un sommet, et qui renvoie un entier représentant un voisin de s choisi aléatoirement. On pourra utiliser la fonction random.randrange(n), qui renvoie un nombre aléatoire entre 0 inclus et n exclus.

On tire un indice entre 0 et la taille de la liste des voisins de `s` moins 1, puis on renvoie le voisin situé à cet indice. Le module `random` doit être importé. Cette méthode suppose que le sommet possède au moins un voisin, ce qui est le cas des sommets du réseau considéré.

Par exemple, pour `s = 4`, la liste des voisins est `[0]` : le seul indice possible est 0 et la fonction renvoie 0.

RÉPONSE À RETENIR

```
import random

def voisin_alea(voisins, s):
    indice = random.randrange(len(voisins[s]))
    return voisins[s][indice]
```

Question 5.

Sujet p. 3

Justifier que la fonction marche_alea donnée dans le sujet est une fonction récursive.

Une fonction est récursive lorsqu'elle s'appelle elle-même. Lorsque `n` est positif, `marche_alea` s'appelle à nouveau avec `n - 1`. Les appels successifs atteignent donc le cas de base `n == 0`, qui renvoie le sommet courant sans nouvel appel. Cela garantit la terminaison pour un entier `n` positif ou nul.

RÉPONSE À RETENIR

`marche_alea` est récursive car sa définition contient un appel à `marche_alea` elle-même. Son cas de base est `n == 0`.

FormaV - document d'entraînement non officiel

Question 6.

Sujet p. 2, 3

> Décrire ce que modélise la fonction marche_alea, en rapport avec le contexte de l'exercice.

La fonction part du sommet `i`. Tant qu'il reste des étapes, elle choisit au hasard un voisin du sommet courant, puis réduit d'une unité le nombre d'étapes restant. Au cas de base, elle renvoie le sommet atteint. On suppose que `n` est positif ou nul.

RÉPONSE À RETENIR

`marche_alea(voisins, i, n)` modélise la position finale du virus après `n` déplacements aléatoires entre ordinateurs voisins, en partant de l'ordinateur `i`.

Question 7.

Sujet p. 3

> Compléter la fonction simule, qui simule n_tests fois le déplacement d'un virus pendant n_pas étapes, démarrant au sommet i, et qui renvoie une liste contenant en position j le nombre de fois que le virus a terminé son parcours au sommet j, divisé par n_tests.

On initialise un compteur par sommet. Pour chaque test, on détermine le sommet final du parcours et on incrémente son compteur. La division finale par `n_tests` transforme les compteurs en fréquences. On suppose `n_tests > 0` et `n_pas >= 0`.

Une exécution illustrative sur le réseau à six sommets : pour un test partant de 4 et deux étapes, le parcours possible 4 → 0 → 5 incrémente le compteur du sommet 5. Comme le choix est aléatoire, d'autres parcours sont possibles.

La complétion directe avec la fonction récursive donnée consiste à appeler `marche_alea`. Toutefois, 1 000 appels récursifs peuvent dépasser la limite de récursion usuelle de Python. Pour le nombre d'étapes de l'appel indiqué dans le sujet, la version ci-dessous utilise une marche itérative équivalente et évite ce risque de pile.

RÉPONSE À RETENIR

```
def marche_alea_iterative(voisins, i, n):
    sommet = i
    for _ in range(n):
        sommet = voisin_alea(voisins, sommet)
    return sommet

def simule(voisins, i, n_tests, n_pas):
    results = [0] * len(voisins)
    for _ in range(n_tests):
        sommet_final = marche_alea_iterative(voisins, i, n_pas)
        results[sommet_final] += 1
    for j in range(len(results)):
        results[j] = results[j] / n_tests
    return results
```

Pour les fréquences fournies à la question 8, la somme vaut `1` : `0.328 + 0.195 + 0.18 + 0.12 + 0.059 + 0.118 = 1`, comme attendu pour des fréquences de sommets finaux.

FormaV - document d'entraînement non officiel

Question 8.

Sujet p. 2, 3

> L'appel simule(voisins, 4, 1000, 1000) renvoie la valeur [0.328, 0.195, 0.18, 0.12, 0.059, 0.118]. Déduire de ce résultat l'ordinateur du réseau qu'il est le plus rentable de protéger.

Chaque valeur donne la fréquence estimée des parcours qui se terminent sur le sommet correspondant à sa position. On compare les six valeurs : la plus grande est `0.328`, située en position 0. La simulation suggère donc que le virus termine le plus souvent ses parcours sur cet ordinateur.

> **RÉPONSE À RETENIR**
>
> L'ordinateur à protéger en priorité est l'ordinateur **0**.

Question 9.

Sujet p. 3

> Au début, on suppose que le virus n'est présent que sur un ordinateur. À chaque étape, il contamine tous ses voisins non déjà contaminés. Un graphe voisins représente un réseau et s représente un sommet de départ. Proposer un algorithme pour déterminer le temps, en étapes, que met un virus à se propager dans l'intégralité d'un réseau.

On peut propager le virus par vagues, comme dans un parcours en largeur : à l'étape 0, seul `s` est contaminé ; à chaque étape suivante, on contamine les voisins des sommets de la vague précédente qui ne l'étaient pas déjà. Ainsi, après `k` étapes, les sommets contaminés sont ceux situés à une distance d'au plus `k` arêtes de `s`.

Exemple avec le graphe initial et `s = 4` : à l'étape 0, on a {4} ; à l'étape 1, on ajoute 0 ; à l'étape 2, on ajoute 1, 2 et 3. Tout le réseau est alors contaminé : il faut 2 étapes.

Chaque sommet est ajouté à l'ensemble des contaminés au plus une fois. Si la propagation s'arrête alors que certains sommets restent non contaminés, le réseau n'est pas entièrement accessible depuis `s` ; le code renvoie alors `-1`. Avec des listes de voisins et un ensemble pour tester l'appartenance, la durée est en `O(V + E)`, où `V` est le nombre de sommets et `E` le nombre d'arêtes. On suppose que le sommet `s` est valide.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

```
def temps_propagation(voisins, s):
    contamines = {s}
    frontiere = [s]
    etapes = 0

    while frontiere:
        suivante = []
        for sommet in frontiere:
            for voisin in voisins[sommet]:
                if voisin not in contamines:
                    contamines.add(voisin)
                    suivante.append(voisin)
        if suivante:
            etapes += 1
        frontiere = suivante

    if len(contamines) == len(voisins):
        return etapes
    return -1
```

Dans un réseau connexe, le résultat est la plus grande distance entre `s` et un sommet du réseau. Dans le graphe initial de la figure, avec `s = 4`, la fonction renvoie 2.

Exercice 2

6 points

Adresses IP et protocoles de routage

A

Question 1.

Sujet p. 4, 5

> Les machines du réseau local L1 indiquent un masque de sous-réseau sur 24 bits en notation CIDR, soit 255.255.255.0 en notation décimale pointée. Donner le masque de sous-réseau en notation décimale pointée des machines du réseau L2 (masque de sous-réseau de 16 bits).

Un masque sur 16 bits comporte 16 bits à 1, puis 16 bits à 0. En regroupant les bits par octets, on obtient `11111111.11111111.00000000.00000000`, soit deux octets à 255 suivis de deux octets à 0.

RÉPONSE À RETENIR

Le masque de sous-réseau de L2 est **255.255.0.0**.

Question 2.

Sujet p. 4, 5

> Concernant le réseau local L2 représenté, donner l'adresse du réseau.

Sur l'image, une adresse de L2 est `172.16.255.254/16`. Avec le masque /16, les deux premiers octets forment la partie réseau et les deux derniers la partie machine. Pour obtenir l'adresse du réseau, on met à 0 tous les bits de la partie machine.

RÉPONSE À RETENIR

L'adresse du réseau L2 est **172.16.0.0**.

Question 3.

Sujet p. 4, 5

> Concernant le réseau local L2 représenté, donner l'adresse de diffusion.

Pour l'adresse de diffusion du réseau /16, on conserve la partie réseau `172.16` et on met à 1 tous les bits de la partie machine. Les deux derniers octets valent alors 255.

RÉPONSE À RETENIR

L'adresse de diffusion de L2 est **172.16.255.255**.

Question 4.

Sujet p. 4, 5

> Concernant le réseau local L2 représenté, donner le nombre maximum de machines pouvant être connectées à ce réseau.

Un masque /16 laisse $32 - 16 = 16$ bits pour la partie machine, soit 2^{16} adresses possibles. L'adresse du réseau et l'adresse de diffusion ne sont pas attribuables à des machines : on les retranche.

RÉPONSE À RETENIR

Le nombre maximal de machines connectées est $\mathbf{2^{16} - 2 = 65\,534}$.

B

Question 5.

Sujet p. 4, 5

> À l'aide des extraits des tables de routage ci-dessus, donner un chemin (c'est-à-dire nommer les routeurs traversés) suivi par un message envoyé du réseau L1 vers le réseau L2.

Le message envoyé de L1 arrive d'abord au routeur A. D'après sa table, A transmet à la passerelle `53.10.10.2`, qui est l'interface de H sur la liaison A–H. La table de H indique ensuite la passerelle `53.10.10.6`, l'interface de D sur la liaison H–D. D est directement connecté à L2.

> **RÉPONSE À RETENIR**
>
> Un chemin est **L1 → A → H → D → L2**.

Question 6.

Sujet p. 4, 5

> La liaison entre les routeurs H et D est rompue. Sachant que le protocole de routage RIP est utilisé (distance en nombre de sauts), donner les nouveaux chemins que pourra suivre un message allant de L1 vers L2.

Après la rupture de H–D, on cherche les chemins ayant le nombre minimal de sauts vers D, qui est directement connecté à L2. Les distances minimales jusqu'à D sont : D à 0 saut ; C et E à 1 ; B, F et H à 2 ; A et G à 3. Depuis A, deux premiers choix conduisent à une route minimale : B puis C, ou H puis C. Dans chacun de ces cas, le trajet de A à D comporte trois liaisons entre routeurs.

> **RÉPONSE À RETENIR**
>
> Les deux chemins minimaux de L1 vers L2 sont **L1 → A → B → C → D → L2** et **L1 → A → H → C → D → L2**.

Question 7.

Sujet p. 4, 5

> Choisir un des chemins de la question précédente. Donner les routeurs dont la règle de routage à destination de L2 est obligatoirement modifiée. Après avoir examiné tous les routeurs, écrire sur votre copie les règles de routage modifiées en conséquence.

Choisissons le chemin A–H–C–D. Après la rupture de H–D, H ne peut plus transmettre directement à D : sa règle vers L2 doit donc être modifiée. Les autres routeurs peuvent conserver leur passerelle : A peut toujours passer par H ; B par C ; C par D ; E par D ; F par E ; G par H. Après mise à jour, H passe par C. Les distances minimales jusqu'à D restent cohérentes avec ces choix.

FormaV - document d'entraînement non officiel

Routeur	Prochain routeur choisi vers L2	Nombre de sauts jusqu'à D
A	H	3
B	C	2
C	D	1
D	Directement connecté à L2	0
E	D	1
F	E	2
G	H	3
H	C	2

RÉPONSE À RETENIR

La règle obligatoirement modifiée est celle de **H** : pour la destination L2, sa passerelle devient **53.10.10.10** (l'interface de C sur la liaison H–C) et son interface de sortie est **53.10.10.9** (l'interface de H sur cette liaison). Les règles des autres routeurs peuvent rester inchangées, comme dans le tableau ci-dessus.

Question 8.

Sujet p. 6

La liaison entre les routeurs H et D est rétablie. Pour tenir compte du débit des liaisons, on utilise le protocole OSPF, avec le coût d'une liaison donné par coût = 10^9 / BP, où BP est la bande passante de la connexion en bits par seconde. Calculer le coût des liaisons pour les 3 valeurs de bande passante qui apparaissent dans le tableau.

On applique la formule `coût = 10^9 / BP`, avec BP en bits par seconde. Les trois débits du tableau sont 1 Gbit/s = 10^9 bit/s, 100 Mbit/s = 10^8 bit/s et 10 Gbit/s = 10^{10} bit/s.

RÉPONSE À RETENIR

Bande passante	Calcul	Coût
1 Gbit/s	10^9 / 10^9	**1**
100 Mbit/s	10^9 / 10^8	**10**
10 Gbit/s	10^9 / 10^{10}	**0,1**

Question 9.

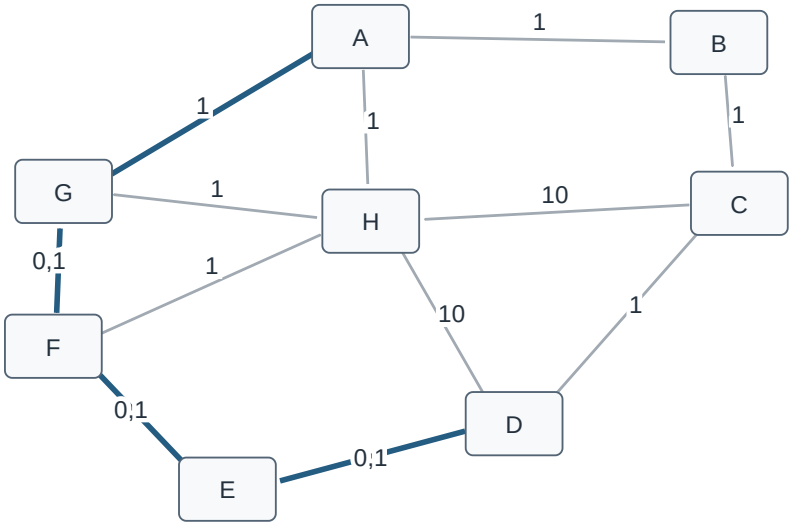
Sujet p. 4, 6

À partir des coûts des liaisons, déterminer le chemin que suivra un message allant de L1 vers L2 et donner son coût.

Les coûts des liaisons sont ceux calculés à la question 8. En partant de A, les liaisons vers B, G et H coûtent chacune 1. Le chemin A–G–F–E–D a pour coût cumulé 1 + 0,1 + 0,1 + 0,1 = 1,3. Pour vérifier qu'il est minimal, on compare notamment les autres routes vers D : A–B–C–D coûte 1 + 1 + 1 = 3 ; A–H–D coûte 1 + 10 = 11 ; A–H–C–D coûte 1 + 10 + 1 = 12. Les coûts étant positifs, le chemin par G, F et E est le chemin de coût minimal.

FormaV - document d'entraînement non officiel

Réseau pondéré : chemin OSPF minimal de L1 à L2



Les coûts des arêtes correspondent aux trois débits indiqués dans le sujet. Le chemin minimal A–G–F–E–D est mis en évidence.

RÉPONSE À RETENIR

Le message suit **L1 → A → G → F → E → D → L2**. Son coût total est **1,3**.

Question 10.

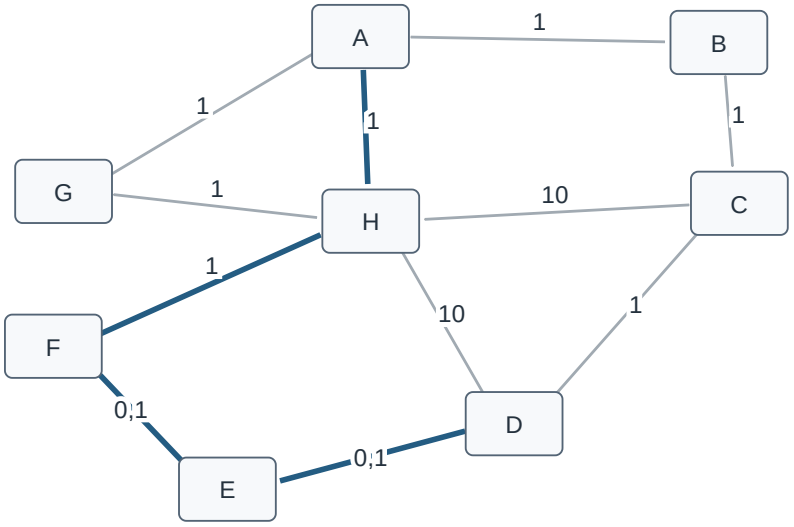
Sujet p. 4, 6

La liaison entre les routeurs G et F est rompue. Déterminer le nouveau chemin suivi par un message allant de L1 vers L2 et donner son coût.

La liaison G–F étant rompue, on retire cette arête du graphe. Le chemin A–H–F–E–D coûte 1 + 1 + 0,1 + 0,1 = 2,2. À titre de comparaison, A–B–C–D coûte 3 et A–G–H–F–E–D coûte 3,2. Les autres routes vers D passant notamment par H–D ou C–H sont plus coûteuses. Le chemin A–H–F–E–D est donc minimal dans le réseau restant.

FormaV - document d'entraînement non officiel

Réseau après la rupture de G–F : chemin OSPF minimal



Les coûts des arêtes sont ceux déduits des débits du sujet. L'arête G–F est absente car rompue ; le chemin minimal A–H–F–E–D est mis en évidence.

> **RÉPONSE À RETENIR**
>
> Le nouveau chemin est **L1 → A → H → F → E → D → L2**, de coût total **2,2**.

FormaV - document d'entraînement non officiel

Exercice 3

8 points

Base de données d'un camping, requêtes SQL et facturation en Python

A

Question 1.

Sujet p. 7

> Citer deux avantages à utiliser une base de données relationnelle plutôt qu'un fichier texte ou un fichier tableur.

Une base relationnelle organise les informations en tables qui peuvent être reliées entre elles. Cela limite la répétition des mêmes données et donc les risques d'incohérence lors des mises à jour. Elle permet aussi d'interroger efficacement les données, par exemple en filtrant des lignes ou en reliant plusieurs tables.

RÉPONSE À RETENIR

Deux avantages possibles : limiter les redondances et les incohérences ; retrouver et croiser efficacement des informations grâce aux requêtes.

Question 2.

Sujet p. 7

> Quelle doit être la caractéristique d'un attribut pour pouvoir être utilisé en tant que clé primaire ?

Une clé primaire identifie une ligne de façon unique. Sa valeur doit donc être différente pour chaque ligne et ne pas être absente.

RÉPONSE À RETENIR

Un attribut utilisé comme clé primaire doit avoir une valeur unique et non nulle pour chaque enregistrement.

Question 3.

Sujet p. 7

> Dans la relation Reservation, quel est le rôle des clés étrangères id_client et id_emplacement ?

Une clé étrangère référence la clé primaire d'une autre relation. Dans `Reservation`, `id_client` désigne le client ayant effectué la réservation et `id_emplacement` désigne l'emplacement réservé. Ces références permettent de relier les relations et de préserver la cohérence des données.

RÉPONSE À RETENIR

`Reservation.id_client` référence `Client.id_client` ; `Reservation.id_emplacement` référence `Emplacement.id_emplacement`.

Question 4.

Sujet p. 7, 8

> Donner le schéma relationnel de la relation Emplacement en précisant la clé primaire et le type de chacun des attributs.

Le sujet ne fixe pas les types SQL. On peut choisir des types adaptés aux valeurs indiquées : un entier pour l'identifiant, du texte pour le nom et la localisation, et un type décimal pour le tarif, qui comporte des centimes dans l'extrait. La clé primaire est `id_emplacement`.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Un schéma possible est : `Emplacement(id_emplacement INTEGER [clé primaire], nom VARCHAR, localisation VARCHAR, tarif_journalier DECIMAL(5,2))`.

Question 5.

Sujet p. 7, 8

> À partir de l'extrait du contenu de la relation Emplacement, donner le résultat de la requête : SELECT id_emplacement, nom, localisation FROM Emplacement WHERE tarif_journalier = 25;

La clause `WHERE` retient les emplacements dont le tarif journalier vaut 25. Parmi les lignes de l'extrait, il s'agit des emplacements d'identifiants 1, 4 et 6. La requête ne conserve ensuite que les colonnes demandées.

RÉPONSE À RETENIR

id_emplacement	nom	localisation
1	myrtille	A4
4	mandarine	B1
6	melon	A2

Question 6.

Sujet p. 7, 8

> Écrire une requête permettant de donner le nom et le prenom de tous les clients habitant à 'Strasbourg'.

On sélectionne les colonnes `nom` et `prenom` de `Client`, puis on filtre les clients selon la valeur de leur ville.

RÉPONSE À RETENIR

```
SELECT nom, prenom
FROM Client
WHERE ville = 'Strasbourg';
```

Aucun extrait de la relation `Client` n'est fourni ; les lignes renvoyées ne peuvent donc pas être déterminées à partir du sujet.

Question 7.

Sujet p. 7, 8

> Écrire une requête permettant d'ajouter un nouveau client : id_client 42 ; nom 'CODD' ; prénom 'Edgar' ; adresse '28 rue des Capucines' ; ville 'Lyon' ; pays 'France' ; numéro de téléphone '0555555555'.

La requête précise les colonnes et fournit leurs valeurs dans le même ordre. Le numéro de téléphone est écrit comme une chaîne de caractères afin de conserver son zéro initial.

RÉPONSE À RETENIR

```
INSERT INTO Client (id_client, nom, prenom, adresse, ville, pays, telephone)
VALUES (42, 'CODD', 'Edgar', '28 rue des Capucines', 'Lyon', 'France', '0555555555');
```


FormaV - document d'entraînement non officiel

Question 8.

Sujet p. 7, 8

> Écrire une requête SQL permettant de récupérer les informations ci-dessous concernant la réservation dont l'identifiant id_reservation est 18 : Client.nom ; Client.prenom ; Reservation.nombre_personne ; Reservation.date_arrivee ; Reservation.date_depart ; Emplacement.tarif_journalier.

Les clés étrangères de `Reservation` permettent de relier la réservation au client et à l'emplacement. On effectue les jointures sur les identifiants correspondants, puis on filtre sur l'identifiant de réservation 18.

RÉPONSE À RETENIR

```
SELECT Client.nom, Client.prenom,
       Reservation.nombre_personne,
       Reservation.date_arrivee,
       Reservation.date_depart,
       Emplacement.tarif_journalier
FROM Reservation
JOIN Client
  ON Reservation.id_client = Client.id_client
JOIN Emplacement
  ON Reservation.id_emplacement = Emplacement.id_emplacement
WHERE Reservation.id_reservation = 18;
```

Aucun extrait des relations `Client` et `Reservation` n'est fourni ; les valeurs renvoyées ne peuvent donc pas être calculées à partir du sujet.

B

Question 9.

Sujet p. 8, 9

> Expliquer pourquoi le terme self est utilisé comme paramètre pour les méthodes des classes Client, Reservation et Emplacement.

`self` désigne l'objet sur lequel la méthode est appelée. Lors d'un appel à une méthode d'instance, Python transmet automatiquement cet objet comme premier argument. La méthode peut ainsi accéder à ses attributs, par exemple avec `self.nom`.

RÉPONSE À RETENIR

`self` est une référence à l'instance concernée ; elle permet à ses méthodes d'accéder aux attributs de cet objet et, si nécessaire, de les modifier.

Question 10.

Sujet p. 8, 9

> Instancier une variable client01 de la classe Client représentant un client se nommant CODD Edgar, habitant au 28 rue des Capucines à Lyon, France, ayant pour numéro de téléphone le 0555555555.

On fournit les arguments dans l'ordre du constructeur de `Client` : nom, prénom, adresse, ville, pays et téléphone. Le téléphone est une chaîne pour préserver le zéro initial.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

```
client01 = Client('CODD', 'Edgar', '28 rue des Capucines',
                  'Lyon', 'France', '0555555555')
```

Question 11.

Sujet p. 8, 9

> Compléter la ligne 5 de la fonction montant_a_regler. Elle reçoit un tuple constitué de trois objets, respectivement des classes Client, Reservation et Emplacement, et doit renvoyer le montant dû pour ce client, cet emplacement et cette durée de séjour. Au tarif journalier de location de l'emplacement, il faut ajouter une taxe de séjour de 2,20 € par jour et par personne.

La durée du séjour est donnée par `reservation.nb_jours()`. Chaque jour, le montant comprend le tarif de l'emplacement et une taxe de 2,20 € par personne. Le coût quotidien est donc multiplié par le nombre de jours.

RÉPONSE À RETENIR

```
return (emplacement.tarif_journalier + 2.20 * reservation.nombre_personne) *
reservation.nb_jours()
```

Pour l'exemple du sujet, cela donne `(30 + 2.20 * 4) * 12 = 465.6` euros, soit le même calcul que `30 * 12 + 4 * 2.20 * 12`.

Question 12.

Sujet p. 10, 11

> Expliquer pourquoi une erreur se produit à l'exécution de la fonction facture_est_valide donnée dans le sujet.

La fonction `split` renvoie des sous-chaînes : `annee` est donc une chaîne de caractères. À la ligne 25, cette chaîne est comparée aux entiers `2018` et `2024`. En Python, cette comparaison entre une chaîne et des entiers provoque une `TypeError`. La fonction `que_des_chiffres` comporte aussi un défaut distinct : la chaîne `"012345789"` omet le chiffre 6, ce qui peut faire rejeter à tort une chaîne contenant ce chiffre, mais ne provoque pas cette erreur d'exécution.

RÉPONSE À RETENIR

L'erreur d'exécution vient de la comparaison `2018 <= annee <= 2024` : `annee` est une chaîne alors que les bornes sont des entiers.

Question 13.

Sujet p. 10, 11

> Proposer une correction du code pour que cette erreur ne se produise plus.

Après avoir vérifié que l'année ne contient que des chiffres, on la convertit en entier avant de la comparer aux bornes. Il faut également corriger la chaîne de caractères utilisée par `que_des_chiffres`, qui doit contenir tous les chiffres de 0 à 9.

RÉPONSE À RETENIR

```
def que_des_chiffres(chaine):
    for car in chaine:
        if car not in '0123456789':
            return False
    return True

# Dans facture_est_valide, après le contrôle des chiffres de l'année :
if len(annee) != 4 or not (2018 <= int(annee) <= 2024):
    return False
```

La conversion par `int` est faite après le contrôle des caractères, et la longueur de l'année est aussi vérifiée.

Question 14. (mois, ligne 28)

Sujet p. 10, 11

> Compléter le code afin de vérifier les mois (ligne 28) dans la fonction facture_est_valide. Les mois MMM sont les trois premières lettres du mois en anglais, selon la liste fournie dans le programme. Pour cette vérification, il est possible d'insérer une ou plusieurs lignes.

La liste `calendrier` contient les abréviations des douze mois en anglais. On vérifie que la valeur de `mois` appartient à cette liste ; sinon, le numéro de facture est invalide.

RÉPONSE À RETENIR

```
if mois not in calendrier:
    return False
```

Question 14. (numéro, ligne 30)

Sujet p. 10, 11

> Compléter le code afin de vérifier le numéro (ligne 30) dans la fonction facture_est_valide. La partie xxx du format 'AAAA-MMM-xxx' désigne trois chiffres. Pour cette vérification, il est possible d'insérer une ou plusieurs lignes.

La partie `numero` doit comporter exactement trois caractères, et chacun doit être un chiffre. On utilise la fonction `que_des_chiffres` corrigée à la question 13.

RÉPONSE À RETENIR

```
if len(numero) != 3 or not que_des_chiffres(numero):
    return False
```

Si les vérifications précédentes et celle-ci sont satisfaites, la fonction peut atteindre `return True`.

Sujet : https://www.sujetdebac.fr/annales-pdf/2024/spe-numerique-informatique-2024-centres-etranger-1-sujet-officiel.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.