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.

Numérique et sciences informatiques

Session 2022 • Métropole • Remplacement

Code sujet **22-NSIJ1ME3** 3 heures 30

Proposition de corrigé pédagogique • FormaV

Se repérer dans le corrigé

Exercice 1 — Algorithmique, arbres binaires de recherche et parcours
9 réponses • sujet p. 2–3

Exercice 2 — Classes, listes, itération et récursivité appliquées à des annonces immobilières
8 réponses • sujet p. 4–6

Exercice 3 — Bases de données relationnelles et langage SQL à partir du catalogue du satellite GAIA
11 réponses • sujet p. 7–8

Exercice 4 — Architecture matérielle, gestion de processus et réseaux
8 réponses • sujet p. 9–10

Exercice 5 — Files et programmation Python
5 réponses • sujet p. 11–13

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

4 points

Algorithmique, arbres binaires de recherche et parcours

Partie A : Préambule

Question 1) — Arbre 1

Sujet p. 2

> Parmi les arbres proposés, recopier le numéro correspondant à un arbre binaire de recherche. Option Arbre 1 : racine 3 ; fils gauche 2, avec enfants 1 et 3 ; fils droit 4, avec enfants 4 et 5.

Dans un ABR, chaque clé du sous-arbre gauche doit être inférieure ou égale à la racine, et chaque clé du sous-arbre droit doit lui être strictement supérieure. Il faut également vérifier la même règle à chaque nœud.

Dans l'arbre 1, le sous-arbre gauche de la racine 3 contient 1, 2 et 3 : ces clés sont toutes inférieures ou égales à 3. Son sous-arbre droit contient 4, 4 et 5 : elles sont toutes strictement supérieures à 3. Les sous-arbres de 2 et de 4 respectent eux aussi les règles : en particulier, le 3 est à droite de 2 et le 4 est à gauche de 4, ce qui est autorisé.

> **RÉPONSE À RETENIR**
>
> **Arbre 1 est un ABR.**

Question 1) — Arbre 2

Sujet p. 2

> Parmi les arbres proposés, recopier le numéro correspondant à un arbre binaire de recherche. Option Arbre 2 : racine 4 ; fils gauche 2, avec enfants 1 et 3 ; fils droit 4, avec enfants 4 et 5.

La racine de l'arbre 2 vaut 4. Or son sous-arbre droit contient un nœud de valeur 4. Toutes les clés du sous-arbre droit de la racine doivent être strictement supérieures à 4 ; cette condition n'est donc pas respectée.

> **RÉPONSE À RETENIR**
>
> **Arbre 2 n'est pas un ABR.**

Question 1) — Arbre 3

Sujet p. 2

> Parmi les arbres proposés, recopier le numéro correspondant à un arbre binaire de recherche. Option Arbre 3 : racine 3 ; fils gauche 2, avec enfants 1 et 1 ; fils droit 4, avec enfants 3 et 5.

La racine de l'arbre 3 vaut 3. Or son sous-arbre droit contient un nœud de valeur 3. Une clé du sous-arbre droit doit être strictement supérieure à la racine : la valeur 3 ne convient pas.

> **RÉPONSE À RETENIR**
>
> **Arbre 3 n'est pas un ABR.** Le seul numéro à retenir parmi les trois arbres est donc **1**.

FormaV - document d'entraînement non officiel

Partie B : Analyse

Question 2.a)

Sujet p. 3

> Dans un ABR, où se trouve le plus petit élément ? Justifier.

Dans un ABR non vide, on part de la racine et on suit les fils gauches aussi longtemps qu'ils existent. En effet, chaque clé située dans un sous-arbre gauche est inférieure ou égale à celle de son parent. Le nœud le plus à gauche contient donc une clé minimale. Si des clés égales sont présentes, plusieurs nœuds peuvent porter cette valeur minimale.

> **RÉPONSE À RETENIR**
>
> Le plus petit élément se trouve au nœud le plus à gauche de l'ABR. Un arbre vide, bien sûr, ne contient pas de plus petit élément.

Question 2.b)

Sujet p. 3

> En utilisant les fonctions de la structure de données ABR présentées dans le sujet, écrire une fonction récursive RechercheValeur prenant en arguments la clé recherchée et l'arbre ABR considéré. Cette fonction retourne un booléen (vrai ou faux) indiquant si la clé est présente dans l'arbre ou non.

On traite d'abord le cas d'un arbre vide : la clé ne peut pas y être présente. Sinon, on compare la clé recherchée à la racine. Si elles sont égales, la recherche réussit ; si la clé est plus petite, on poursuit dans le sous-arbre gauche ; si elle est plus grande, dans le sous-arbre droit. Chaque appel porte sur un sous-arbre plus petit, jusqu'à trouver la clé ou atteindre un arbre vide : la recherche termine donc sur un ABR fini.

```
def RechercheValeur(cle, a):
    if est_vide(a):
        return False
    elif cle == racine(a):
        return True
    elif cle < racine(a):
        return RechercheValeur(cle, sous_arbre_gauche(a))
    else:
        return RechercheValeur(cle, sous_arbre_droit(a))
```

Les comparaisons sont cohérentes avec la règle de l'ABR : les clés inférieures ou égales à la racine se cherchent à gauche, et les clés strictement supérieures à droite.

> **RÉPONSE À RETENIR**
>
> `RechercheValeur(cle, a)` renvoie `True` si la clé est dans l'ABR et `False` sinon.

Question 3.a)

Sujet p. 3

> Pour l'ABR représenté, dire à quel type de parcours correspond le résultat suivant, où les clés sont triées dans l'ordre croissant : 1 – 2 – 3 – 5 – 6 – 7 – 8 – 9 – 10.

Un parcours infixe visite successivement le sous-arbre gauche, la racine, puis le sous-arbre droit. Dans un ABR, cette visite donne les clés dans l'ordre croissant : ici, le résultat fourni est bien la suite croissante des clés de l'arbre.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Le résultat correspond au **parcours infixe** (gauche, racine, droite).

Question 3.b)

Sujet p. 3

Donner le parcours préfixe de cet arbre.

Le parcours préfixe visite d'abord la racine, puis le sous-arbre gauche, puis le sous-arbre droit. On part donc de 7 ; on parcourt ensuite l'arbre de gauche (2, puis 1, puis 5 avec ses enfants 3 et 6), puis celui de droite (10, puis 8 et 9).

RÉPONSE À RETENIR

Parcours préfixe : 7 – 2 – 1 – 5 – 3 – 6 – 10 – 8 – 9.

Question 3.c)

Sujet p. 3

Donner le parcours suffixe de cet arbre.

Le parcours suffixe visite d'abord le sous-arbre gauche, puis le sous-arbre droit, et termine par la racine. À gauche, on obtient 1 – 3 – 6 – 5 – 2 ; à droite, 9 – 8 – 10 ; on ajoute enfin la racine 7.

RÉPONSE À RETENIR

Parcours suffixe : 1 – 3 – 6 – 5 – 2 – 9 – 8 – 10 – 7.

Question 3.d)

Sujet p. 3

Donner le parcours en largeur de cet arbre.

Le parcours en largeur visite les nœuds niveau par niveau, de gauche à droite. On visite d'abord la racine 7, puis 2 et 10, puis 1, 5 et 8, et enfin 3, 6 et 9.

RÉPONSE À RETENIR

Parcours en largeur : 7 – 2 – 10 – 1 – 5 – 8 – 3 – 6 – 9.

FormaV - document d'entraînement non officiel

Exercice 2

4 points

Classes, listes, itération et récursivité appliquées à des annonces immobilières

Partie A : Analyse du code et complétion

Question 1.a)

Sujet p. 4, 5

> Combien d'éléments contient la liste v ?

La liste est initialisée par `v = []`. Les cinq instructions `v.append(...)` ajoutent chacune une villa : la liste contient donc cinq éléments.

RÉPONSE À RETENIR

La liste `v` contient **5 éléments**.

Question 1.b)

Sujet p. 4, 5

> Que retourne l'instruction v[1].nom() ?

La deuxième instruction `v.append(...)` ajoute « Les goélands » : c'est donc l'objet désigné par `v[1]`, puisque les indices commencent à 0. L'intention de la question est que l'appel fournisse ce nom.

Toutefois, dans le code imprimé, l'initialiseur de `Villa` affecte la chaîne du nom à `self.nom`, alors que la classe définit aussi une méthode `nom()`. L'attribut d'instance masque la méthode : littéralement, `v[1].nom()` provoque une erreur de type `TypeError`, car une chaîne de caractères n'est pas appelable.

RÉPONSE À RETENIR

Réponse visée : « Les goélands ».

Avec le code Python exactement imprimé : `v[1].nom()` ne retourne pas cette chaîne et provoque une `TypeError`. Pour obtenir le nom avec le code fourni, il faut écrire `v[1].nom`, ou modifier la classe pour éviter le conflit entre l'attribut et la méthode.

Question 1.c)

Sujet p. 4, 5

> Compléter sur votre copie la méthode surface(), qui renvoie la surface habitable du logement : def surface(self): return …

La surface habitable est la somme des superficies du séjour et des deux chambres. Dans le code imprimé, chaque objet `Piece` stocke sa superficie dans l'attribut numérique `sup` ; cet attribut masque également la méthode `sup()`. Pour que la méthode complétée soit exécutable avec les classes telles qu'elles sont écrites, on additionne donc les attributs `sup`, sans parenthèses.

```
def surface(self):
    return self.sejour.sup + self.ch1.sup + self.ch2.sup
```

Si l'on corrigeait la classe `Piece` pour que sa méthode d'accès soit réellement appelable, la somme pourrait alors utiliser des appels à cette méthode.

RÉPONSE À RETENIR

Avec le code imprimé, compléter la méthode ainsi :

```
def surface(self):
    return self.sejour.sup + self.ch1.sup + self.ch2.sup
```

Pour « Les quatre vents », le calcul donne $40 + 10 + 20 = 70$ m².

Question 2.

Sujet p. 4, 5

> Rédiger sur votre copie la portion du programme qui réalise cette sélection : parcourir séquentiellement la liste v et afficher à l'écran le nom de chaque villa équipée d'une cuisine équipée.

On parcourt chaque villa de la liste et on teste si `equip()` renvoie `"eq"`. Dans le code imprimé, le nom de la villa est stocké dans l'attribut `nom`, qui masque la méthode portant ce même nom ; pour obtenir un programme exécutable sans modifier la classe, on affiche donc `villa.nom`, sans parenthèses.

```
for villa in v:
    if villa.equip() == "eq":
        print(villa.nom)
```

RÉPONSE À RETENIR

Avec les données de l'énoncé, les trois villas équipées sont affichées dans l'ordre de la liste :

```
Les quatre vents
Les goélands
Les oliviers
```

Partie B : Récursivité

Question 3. — Proposition 1

Sujet p. 5

> Parmi les propositions suivantes, recopier celle qui caractérise un appel récursif. Option : « appel d'une fonction par elle-même. »

Un appel est récursif lorsqu'une fonction s'appelle elle-même, directement ou indirectement. Cette proposition caractérise bien la récursivité.

RÉPONSE À RETENIR

Proposition correcte : « appel d'une fonction par elle-même. »

Question 3. — Proposition 2

Sujet p. 5

> Parmi les propositions suivantes, recopier celle qui caractérise un appel récursif. Option : « appel dont l'exécution est un processus itératif. »

La répétition d'instructions par un processus itératif ne caractérise pas un appel récursif : une fonction récursive s'appelle elle-même.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Cette proposition n’est pas celle qui caractérise un appel récursif : « appel dont l’exécution est un processus itératif. »

Question 3. — Proposition 3

Sujet p. 5

> Parmi les propositions suivantes, recopier celle qui caractérise un appel récursif. Option : « appel d’une fonction comportant une boucle. »

Une fonction peut comporter une boucle sans s’appeler elle-même. La présence d’une boucle ne suffit donc pas à caractériser la récursivité.

RÉPONSE À RETENIR

Cette proposition n’est pas celle qui caractérise un appel récursif : « appel d’une fonction comportant une boucle. »

Question 4.

Sujet p. 4, 6

> Écrire sur votre copie le code Python de la fonction max_surface(v). Selon l’algorithme décrit, si la liste contient une seule villa, il faut la renvoyer. Si elle en contient au moins deux, comparer les surfaces de v[0] et v[1], supprimer v[0] si sa surface est inférieure à celle de v[1], sinon supprimer v[1], puis recommencer avec la liste tronquée jusqu’à ce qu’une seule villa reste.

Le cas d’arrêt est celui d’une liste d’une seule villa : la fonction renvoie alors cette villa. Sinon, elle compare les deux premières surfaces. Selon la règle donnée, elle supprime la première villa si sa surface est strictement plus petite ; dans le cas contraire, elle supprime la deuxième. Elle rappelle ensuite la fonction sur la liste raccourcie.

À chaque appel récursif, la liste perd un élément. Pour une liste non vide, cette diminution garantit que le cas d’arrêt sera atteint. La fonction modifie la liste passée en argument ; cette modification est conforme à l’algorithme décrit.

```
def max_surface(v):
    if len(v) == 1:
        return v[0]
    if v[0].surface() < v[1].surface():
        del v[0]
    else:
        del v[1]
    return max_surface(v)
```

Trace avec les données du sujet : les surfaces sont respectivement 70, 80, 65, 60 et 60 m². La comparaison de 70 et 80 élimine la première villa. Puis 80 est comparé successivement à 65, à 60 et à 60 ; à chaque étape, la villa de surface inférieure est éliminée. La fonction termine donc avec « Les goélands ».

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Le code demandé est :

```
def max_surface(v):
    if len(v) == 1:
        return v[0]
    if v[0].surface() < v[1].surface():
        del v[0]
    else:
        del v[1]
    return max_surface(v)
```

Cette fonction suppose que v contient au moins une villa. Avec la liste fournie, elle renvoie **« Les goélands »**, dont la surface habitable est **80 m²**.

Exercice 3

4 points

Bases de données relationnelles et langage SQL à partir du catalogue du satellite GAIA

Partie A : Schéma relationnel

Question 1.

Sujet p. 7

> Justifier que l'attribut Num_Objet peut être choisi comme clé primaire de la table Gaia.

Une clé primaire identifie de façon unique chaque ligne d'une table. Le sujet précise que `Num_Objet` identifie chaque objet cartographié de manière unique. Dans l'extrait, ses valeurs ne sont pas répétées.

> **RÉPONSE À RETENIR**
>
> `Num_Objet` peut être choisi comme clé primaire de `Gaia`, car il identifie chaque objet de manière unique.

Question 2.

Sujet p. 8

> Proposer le schéma relationnel de la table Type en soulignant la clé primaire.

La clé primaire de `Type` est `Type_Objet`, comme l'indique l'énoncé. Le libellé est associé à ce code.

> **RÉPONSE À RETENIR**
>
> `Type(Type_Objet : String, Libelle_Objet : String)`, avec `Type_Objet` comme clé primaire.

Partie B : Instructions SQL

Question 3. — a)

Sujet p. 7, 8

> Parmi les commandes suivantes, laquelle ne provoque pas d'erreur ? Option a) INSERT INTO Gaia VALUES ('8', 4, 'WISEA J085510', 'Naine Brune', 'WISEA J085510', 133.781, -7.244, 439.000, 'WISEA J085510');

La valeur donnée pour la clé étrangère `Type_Objet` est `'Naine Brune'`. Or les valeurs de cette clé dans la table `Type` sont les codes `'LM'`, `'Planet'`, `'*'`, `'BD'` et `'WD'`. `'Naine Brune'` n'y figure pas : la contrainte de clé étrangère n'est donc pas respectée. De plus, la valeur de `Num_Objet` est écrite comme une chaîne plutôt que comme un entier ; son éventuelle conversion dépend du système SQL.

> **RÉPONSE À RETENIR**
>
> L'option a) provoque une erreur de clé étrangère.

Question 3. — b)

Sujet p. 7, 8

> Parmi les commandes suivantes, laquelle ne provoque pas d'erreur ? Option b) INSERT INTO Gaia VALUES (8, 4, 'WISEA J085510', 'Naine Brune', 'WISEA J085510', 133.781, -7.244, 439.000, 'WISEA J085510');

Cette option présente les chaînes entre apostrophes et les valeurs numériques sous forme numérique. Toutefois, elle fournit `'Naine Brune'` pour `Type_Objet`, alors que cet attribut référence `Type.Type_Objet` et que `'Naine Brune'` n'est pas une valeur de cette clé dans la table `Type`.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

L'option b) est la seule qui paraît correctement formée du point de vue de l'écriture des chaînes et des types numériques, mais elle provoque tout de même une erreur de clé étrangère telle que l'énoncé définit les tables. En conséquence, aucune des quatre commandes proposées ne satisfait toutes les contraintes du schéma.

Question 3. — c)

Sujet p. 7, 8

> Parmi les commandes suivantes, laquelle ne provoque pas d'erreur ? Option c) INSERT INTO Gaia VALUES (8, 4, WISEA J085510, 'Naine Brune', WISEA J085510, 133.781, -7.244, 439.000, WISEA J085510);

Les noms de système et d'objet sont des chaînes de caractères, mais ils ne sont pas entourés d'apostrophes. En particulier, `WISEA J085510` n'est pas une chaîne SQL correctement délimitée. De plus, la valeur `'Naine Brune'` ne correspond pas à une clé présente dans `Type.Type_Objet`.

RÉPONSE À RETENIR

L'option c) provoque une erreur de syntaxe SQL et ne respecterait pas non plus la contrainte de clé étrangère.

Question 3. — d)

Sujet p. 7, 8

> Parmi les commandes suivantes, laquelle ne provoque pas d'erreur ? Option d) INSERT INTO Gaia VALUES (8, 4, 'WISEA J085510', 'Naine Brune', 'WISEA J085510', '133.781', -7.244, 439.000, 'WISEA J085510');

Comme dans les autres options, `'Naine Brune'` ne figure pas parmi les valeurs de la clé référencée `Type.Type_Objet` : la clé étrangère n'est pas respectée. En outre, `Ascension_Droite` est de type `Real`, mais sa valeur est écrite entre apostrophes ; la conversion de cette chaîne en nombre n'est pas garantie par le schéma et dépend du système SQL.

RÉPONSE À RETENIR

L'option d) provoque une erreur de clé étrangère ; l'acceptation de la valeur textuelle de `Ascension_Droite` dépend également du système SQL.

Question 4.

Sujet p. 8

> Expliquer pourquoi le code SQL suivant ne fonctionne pas : INSERT INTO Type VALUES ('BD', 'Trou Noir');

La table `Type` contient déjà une ligne dont la clé primaire `Type_Objet` vaut `'BD'`, associée au libellé `'Naine Brune'`. L'insertion proposée essaie d'ajouter une seconde ligne portant cette même clé primaire, ce qui est interdit.

RÉPONSE À RETENIR

La commande échoue à cause du doublon de clé primaire `'BD'`. Pour modifier le libellé de cette ligne, il faudrait utiliser une commande `UPDATE`, et non insérer une nouvelle ligne.

Question 5.

Sujet p. 7, 8

> Indiquer le résultat de la requête suivante exécutée sur l'extrait présenté : SELECT Nom_Objet, Parallaxe FROM

Gaia WHERE Type_Objet = 'Planet';

La condition sélectionne les lignes où `Type_Objet` vaut `'Planet'`. Dans l'extrait, ce sont les objets 2 et 11. La projection ne conserve que leur nom et leur parallaxe.

RÉPONSE À RETENIR

Nom_Objet	Parallaxe
Proxima Cen b	768,067
Lalande 21185 b	392,753

Question 6.

Sujet p. 7, 8

Écrire la requête qui permet de récupérer le nom du système, le nom de l'objet et le libellé du type pour des objets ayant une parallaxe supérieure à 400 millisecondes d'arc et étant des « Etoile ».

La table `Gaia` contient le code du type, tandis que son libellé se trouve dans `Type`. Il faut donc joindre les deux tables sur `Type_Objet`, puis sélectionner les lignes dont le libellé est `'Etoile'` et dont la parallaxe est strictement supérieure à 400.

```
SELECT Gaia.Nom_Systeme, Gaia.Nom_Objet, Type.Libelle_Objet
FROM Gaia
JOIN Type ON Gaia.Type_Objet = Type.Type_Objet
WHERE Gaia.Parallaxe > 400
  AND Type.Libelle_Objet = 'Etoile';
```

Dans l'extrait, le code `'*'` correspond au libellé `'Etoile'`. Les objets 3 et 4 ont chacun une parallaxe de 743,000 ; ils satisfont donc le filtre. L'objet 12 a une parallaxe de 379,210 et est exclu.

RÉPONSE À RETENIR

Nom_Systeme	Nom_Objet	Libelle_Objet
alf Cen	alf Cen A	Etoile
alf Cen	alf Cen B	Etoile

Question 7.a)

Sujet p. 8

Écrire une requête permettant d'insérer un nouveau type « ST » de libellé « Etoile ».

La table `Type` comporte le code du type puis son libellé. On insère donc le code `'ST'` et le libellé `'Etoile'`, dans cet ordre.

```
INSERT INTO Type VALUES ('ST', 'Etoile');
```

RÉPONSE À RETENIR

Cette commande insère le type `ST` de libellé `Etoile`.

Question 7.b)

Sujet p. 7, 8

> On veut remplacer la valeur '*' de Type_Objet dans les tables Gaia et Type par la valeur 'ST'. Expliquer la démarche à suivre et les commandes à exécuter pour finaliser la modification.

Pour respecter la clé étrangère, il faut d'abord que le nouveau code `'ST'` existe dans `Type`. Après l'insertion de la ligne `('ST', 'Etoile')`, on remplace dans `Gaia` les références à `'*'` par `'ST'`. Une fois ces références remplacées, on peut supprimer l'ancienne ligne de `Type`. Si la commande demandée en 7.a) a déjà été exécutée, il ne faut pas la répéter.

```
INSERT INTO Type VALUES ('ST', 'Etoile');

UPDATE Gaia
SET Type_Objet = 'ST'
WHERE Type_Objet = '*';

DELETE FROM Type
WHERE Type_Objet = '*';
```

RÉPONSE À RETENIR

La valeur `'*'` est ainsi remplacée par `'ST'` dans les références de `Gaia`, et l'ancienne ligne est retirée de `Type`. Si l'insertion de `'ST'` a déjà été faite en 7.a), commencer directement par la commande `UPDATE`.

Exercice 4

4 points

Architecture matérielle, gestion de processus et réseaux

Partie A : Architecture matérielle

Question 1. — a)

Sujet p. 9

> Parmi les schémas proposés, lequel représente le mieux une architecture de Von Neumann ? Option a) : schéma montrant la mémoire, l'UC, l'UAL, les entrées et les sorties.

Dans le schéma a), l'unité centrale regroupe l'unité de commande (UC) et l'unité arithmétique et logique (UAL), et communique avec la mémoire ainsi qu'avec les entrées et les sorties. C'est la représentation usuelle de l'architecture de Von Neumann, où instructions et données sont stockées dans une mémoire commune.

> **RÉPONSE À RETENIR**
>
> **Le schéma a) est celui qui représente le mieux une architecture de Von Neumann.**

Question 1. — b)

Sujet p. 9

> Parmi les schémas proposés, lequel représente le mieux une architecture de Von Neumann ? Option b) : schéma montrant l'UAL au-dessus de l'UC, les entrées et les sorties de part et d'autre, et la mémoire sous l'UC.

Dans le schéma b), l'UAL est représentée séparément au-dessus de l'UC, tandis que la mémoire est reliée sous l'UC. Il ne représente donc pas aussi clairement que le schéma a) une unité centrale composée de l'UC et de l'UAL, reliée à une mémoire commune.

> **RÉPONSE À RETENIR**
>
> **Le schéma b) n'est pas celui à retenir ; la réponse est le schéma a).**

Partie B : Le réseau

Question 2.

Sujet p. 9, 10

> Proposer une adresse IPv4 pour l'ordinateur PC02.

Le réseau de LAN01 est `192.168.10.0/24`. Une adresse de PC02 doit appartenir à ce sous-réseau, sans être l'adresse du réseau (`192.168.10.0`) ni celle de diffusion (`192.168.10.255`). Le sujet ne précise pas quelles adresses sont déjà attribuées : on peut donc proposer une adresse d'hôte, sous réserve qu'elle soit disponible.

> **RÉPONSE À RETENIR**
>
> Par exemple, `192.168.10.10`, si cette adresse n'est pas déjà attribuée à une autre machine.

FormaV - document d'entraînement non officiel

Question 3.

Sujet p. 9, 10

> Compte tenu du masque de sous-réseau de LAN01, combien de machines peuvent être connectées ?

Le préfixe `/24` laisse `32 - 24 = 8` bits pour la partie machine. Il y a donc `2^8 = 256` adresses dans le sous-réseau. L'adresse du réseau et l'adresse de diffusion ne pouvant pas être attribuées à des machines, on les soustrait : `256 - 2 = 254`.

> **RÉPONSE À RETENIR**
>
> **254 machines peuvent être connectées à LAN01.**

Question 4.

Sujet p. 9, 10

> Quel est le rôle d'un switch ?

Un switch relie des équipements d'un même réseau local. Il transmet les trames vers le port correspondant à l'équipement destinataire, en s'appuyant sur les adresses MAC.

> **RÉPONSE À RETENIR**
>
> **Un switch connecte des équipements d'un réseau local et achemine les trames vers leur destinataire.**

Question 5.

Sujet p. 9, 10

> Quel est le rôle d'un routeur ?

Un routeur relie des réseaux différents. Il examine l'adresse IP de destination et consulte sa table de routage pour déterminer vers quel prochain routeur ou quelle interface transmettre le paquet.

> **RÉPONSE À RETENIR**
>
> **Un routeur achemine les paquets entre des réseaux différents à l'aide de sa table de routage.**

Question 6.

Sujet p. 9, 10

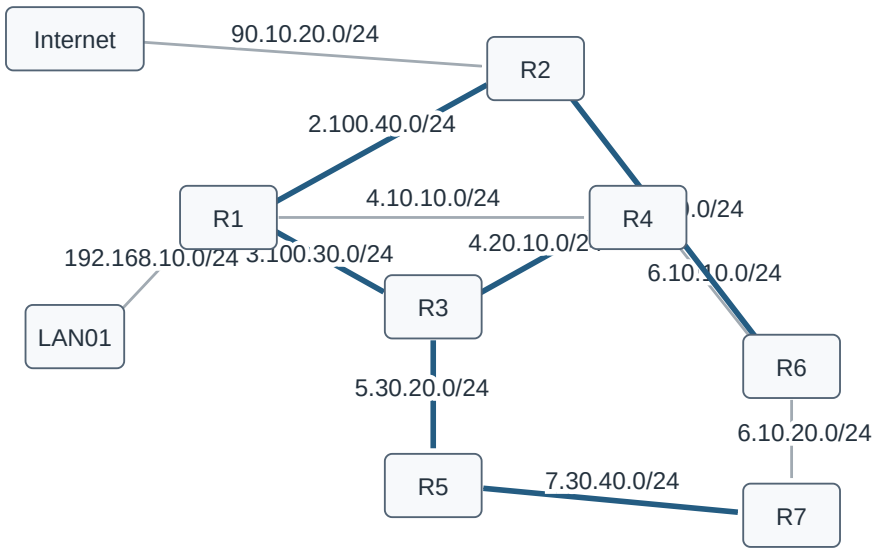
> Reproduire et compléter sur votre copie les lignes incomplètes de la table de routage RIP du routeur R1 en suivant la logique du protocole RIP. Les lignes concernées sont celles des destinations 4.20.10.0/24, 7.30.40.0/24 et 6.10.30.0/24.

Dans la table, les réseaux directement accessibles à R1 par ses routeurs voisins ont une métrique de 1. Avec RIP, la métrique augmente de 1 à chaque routeur traversé. Le réseau `4.20.10.0/24`, entre R3 et R4, est donc atteint depuis R1 avec une métrique de 2, que l'on passe par R3 ou par R4. Le réseau `6.10.30.0/24`, entre R2 et R6, est atteint en passant par R2 : sa métrique est 2. Pour `7.30.40.0/24`, le chemin le plus court passe par R3 puis R5 avant d'atteindre le réseau entre R5 et R7 ; la métrique est donc 3.

Les passerelles de ces routes sont les adresses déjà indiquées pour les routeurs voisins dans la table. Pour le réseau `4.20.10.0/24`, les deux passerelles possibles donnent la même métrique.

FormaV - document d'entraînement non officiel

Topologie du réseau et chemins retenus depuis R1



Les étiquettes des arêtes reprennent les sous-réseaux du schéma du sujet. Les chemins mis en évidence correspondent aux routes proposées dans le tableau.

RÉPONSE À RETENIR

Destination réseau	Passerelle	Métrique
`4.20.10.0/24`	`3.100.30.2` ou `4.10.10.2`	2
`7.30.40.0/24`	`3.100.30.2`	3
`6.10.30.0/24`	`2.100.40.1`	2

Question 7.

Sujet p. 9, 10

> À la suite d'un problème technique, la liaison de R1 vers R2 n'est plus active. Écrire sur votre copie la ligne correspondant à la destination « Internet », telle qu'elle serait modifiée selon le protocole RIP.

La route vers le réseau d'Internet `90.10.20.0/24` passe par R2, avec la passerelle `2.100.40.1` et une métrique de 2. Si la liaison de R1 vers R2 devient inactive, cette route n'est plus accessible par cette passerelle. En RIP, une métrique de 16 signifie que la destination est inaccessible.

RÉPONSE À RETENIR

`90.10.20.0/24` | `2.100.40.1` | `16` — la route est déclarée inaccessible.

Exercice 5

4 points

Files et programmation Python

Question 1. — Situation 1

Sujet p. 11

> Laquelle des deux situations est associée à une structure de file ? Situation 1 : « Je cuisine des crêpes. Dès qu'une crêpe est faite, je la place sur un plat. Chaque nouvelle crêpe est placée sur la crêpe précédente. Quand je vais manger une de ces crêpes, je commencerai par la crêpe située en haut de mon tas. »

Une file applique le principe « premier arrivé, premier servi ». Ici, la crêpe mangée en premier est celle du haut du tas, donc la dernière qui a été posée. C'est le principe d'une pile, et non d'une file.

RÉPONSE À RETENIR

La situation 1 n'est pas associée à une file : elle décrit une pile, selon le principe « dernier arrivé, premier servi ».

Question 1. — Situation 2

Sujet p. 11

> Laquelle des deux situations est associée à une structure de file ? Situation 2 : « Je dispose d'une imprimante placée en réseau dans ma salle de classe équipée d'ordinateurs, tous en réseau. Tous les élèves présents ont accès à cette imprimante, via le réseau. À la fin de la séance, les élèves envoient leur production à l'impression. Les documents sont imprimés dans l'ordre d'arrivée. »

Les documents sont imprimés dans l'ordre de leur arrivée : le premier document envoyé est le premier traité. Cela correspond au principe FIFO d'une file (« premier arrivé, premier servi »).

RÉPONSE À RETENIR

La situation 2 est associée à une file.

Question 2.a)

Sujet p. 12, 13

> On suppose que la file F contient, de la queue à la tête : Client4, Prioritaire, Client3, Client2, Client1. On considère la file V définie dans le code fourni. Quel sera le contenu de V, de F et de la variable val à la suite de ces instructions Python ?

Sur le schéma de la page 13, la queue est à gauche et la tête à droite. Comme `defiler(F)` retire l'élément en tête, les valeurs retirées successivement sont `Client1`, `Client2`, `Client3`, puis `Prioritaire`. Ce dernier élément n'est pas enfilé dans `V`, car il fait échouer le test de la boucle. Il reste alors `Client4` dans `F`.

Étape	`val`	`F` (queue → tête)	`V` (queue → tête)
Après le premier retrait	Client1	Client4, Prioritaire, Client3, Client2	vide
Après le premier passage dans la boucle	Client2	Client4, Prioritaire, Client3	Client1
Après le deuxième passage	Client3	Client4, Prioritaire	Client2, Client1
Après le troisième passage et le retrait suivant	Prioritaire	Client4	Client3, Client2, Client1

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

À la fin, `V` contient, de la queue à la tête : **Client3, Client2, Client1**. `F` contient **Client4** et `val` vaut **« Prioritaire »**.

Question 2.b)

Sujet p. 12, 13

> Compléter le programme de la fonction longueur_file(F), qui doit renvoyer le nombre d'éléments d'une file donnée en paramètre. À la fin du programme, la file F doit avoir retrouvé son état d'origine.

À chaque tour de la première boucle, on retire un élément de `F`, on augmente `n` de 1 et on place l'élément dans `V`. Une fois `F` vide, on transfère les éléments de `V` vers `F`. Comme ils sont retirés et réenfilés dans leur ordre FIFO, `F` retrouve son ordre initial. Chaque boucle termine, car elle retire un élément de la file qu'elle parcourt à chaque passage.

```
def longueur_file(F):
    V = creer_file_vide()
    n = 0

    while not est_vide(F):
        n = n + 1
        val = defiler(F)
        enfiler(V, val)

    while not est_vide(V):
        val = defiler(V)
        enfiler(F, val)

    return n
```

Pour la file de la question 2.a, les cinq retraits portent successivement `n` à 1, 2, 3, 4 puis 5. Le transfert des cinq éléments de `V` vers `F` rétablit la file initiale.

RÉPONSE À RETENIR

`longueur_file(F)` renvoie le nombre d'éléments de la file et la laisse dans son état d'origine. Pour la file fournie dans le sujet, le résultat est **5**.

Question 2.c)

Sujet p. 12, 13

> Écrire une fonction compter_prio qui prend en paramètre une file F et renvoie le nombre de personnes prioritaires dans la file d'attente, à l'instant t. La file F doit être identique à celle du départ en fin d'exécution de la fonction.

On retire chaque élément de `F`, on compte ceux dont la valeur est exactement `"Prioritaire"`, puis on place chaque élément dans une file temporaire `V`. On transfère ensuite `V` vers `F` pour restaurer la file et son ordre. Les deux boucles terminent, car chacune retire un élément de la file qu'elle parcourt à chaque passage.

```
def compter_prio(F):
    V = creer_file_vide()
    n = 0

    while not est_vide(F):
        val = defiler(F)
        if val == "Prioritaire":
            n = n + 1
        enfiler(V, val)

    while not est_vide(V):
        val = defiler(V)
        enfiler(F, val)

    return n
```

Pour la file de l'énoncé, le parcours rencontre successivement `Client1`, `Client2`, `Client3`, `Prioritaire` et `Client4`. Le compteur vaut donc 1 après le passage de `Prioritaire` et reste à 1 ; le transfert final restaure la file.

> **RÉPONSE À RETENIR**
>
> `compter_prio(F)` renvoie le nombre de personnes prioritaires et conserve l'état initial de `F`. Pour la file donnée dans le sujet, le résultat est **1**.

Sujet : https://aeif.fr/wp-content/uploads/2022/11/2022_Metropole_Septembre.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.