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 • Jour 1

Code sujet **Non indiqué dans le document** 3 h 30

Proposition de corrigé pédagogique • FormaV

Se repérer dans le corrigé

Exercice 1 — Algorithmique et programmation
9 réponses • sujet p. 1–1

Exercice 2 — Bases de données, SQL et réseaux
10 réponses • sujet p. 2–4

Exercice 3 — Piles, arbres binaires et algorithmique
11 réponses • sujet p. 5–7

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

Algorithmique et programmation

Question 1.

Sujet p. 1

> On saisit dans la console l'instruction tester_palindrome('kayak'). Combien de fois la fonction tester_palindrome est-elle appelée lors de l'exécution de cette instruction ? Compter l'appel initial.

Chaque appel compare les caractères aux deux extrémités. S'ils sont égaux, un nouvel appel est effectué sur la chaîne privée de son premier et de son dernier caractère.

Appel	Chaîne traitée	Action
1	`kayak`	`k` et `k` sont égaux ; appel sur `aya`.
2	`aya`	`a` et `a` sont égaux ; appel sur `a`.
3	`a`	La longueur est inférieure à 2 ; la fonction renvoie `True`.

Le comptage comprend bien l'appel initial sur `kayak`.

RÉPONSE À RETENIR

La fonction est appelée **3 fois**.

Question 2. (a)

Sujet p. 1

> Justifier que la fonction tester_palindrome est récursive.

Une fonction est récursive lorsqu'elle s'appelle elle-même. Ici, à la ligne 8, `tester_palindrome` appelle de nouveau `tester_palindrome`, cette fois sur la chaîne raccourcie à la ligne 7.

RÉPONSE À RETENIR

`tester_palindrome` est récursive car elle s'appelle elle-même à la ligne 8.

Question 2. (b)

Sujet p. 1

> Expliquer pourquoi l'appel à la fonction tester_palindrome se terminera quelle que soit la chaîne de caractères sur laquelle elle s'applique.

La chaîne considérée est finie. Si sa longueur est inférieure à 2, la fonction s'arrête immédiatement. Si ses caractères extrêmes diffèrent, elle renvoie `False` et s'arrête également. Dans le seul cas où un nouvel appel est effectué, la chaîne perd deux caractères : sa longueur diminue strictement. Cette diminution ne peut pas se poursuivre indéfiniment et finit par donner une longueur inférieure à 2.

RÉPONSE À RETENIR

L'appel se termine pour toute chaîne finie : chaque appel récursif réduit sa longueur de 2, jusqu'au cas de base, ou la fonction renvoie immédiatement `False`.

FormaV - document d'entraînement non officiel

Question 3. (a) — ZeroDivisionError

Sujet p. 1

> La saisie de tester_palindrome(53235) génère une erreur. Parmi les quatre propositions suivantes, indiquer le type d'erreur affiché : ZeroDivisionError.

Aucune division n'est effectuée. L'erreur se produit dès l'évaluation de `len(53235)`, avant toute autre opération.

> **RÉPONSE À RETENIR**
>
> `ZeroDivisionError` n'est pas le type d'erreur affiché.

Question 3. (a) — ValueError

Sujet p. 1

> La saisie de tester_palindrome(53235) génère une erreur. Parmi les quatre propositions suivantes, indiquer le type d'erreur affiché : ValueError.

L'erreur n'est pas due à une valeur incorrecte pour une opération donnée. Elle est due au fait qu'on demande la longueur d'un entier, alors que `len` s'applique à des séquences, comme les chaînes.

> **RÉPONSE À RETENIR**
>
> `ValueError` n'est pas le type d'erreur affiché.

Question 3. (a) — TypeError

Sujet p. 1

> La saisie de tester_palindrome(53235) génère une erreur. Parmi les quatre propositions suivantes, indiquer le type d'erreur affiché : TypeError.

À la ligne 2, Python tente d'évaluer `len(53235)`. Or `53235` est un entier et la fonction `len` ne s'applique pas à ce type de valeur. L'exécution s'arrête donc avant les tests des caractères.

> **RÉPONSE À RETENIR**
>
> La bonne réponse est **`TypeError`**.

Question 3. (a) — IndexError

Sujet p. 1

> La saisie de tester_palindrome(53235) génère une erreur. Parmi les quatre propositions suivantes, indiquer le type d'erreur affiché : IndexError.

Le programme n'atteint pas l'accès aux caractères. L'erreur survient auparavant, lors de l'appel de `len` sur l'entier `53235`.

> **RÉPONSE À RETENIR**
>
> `IndexError` n'est pas le type d'erreur affiché.

Question 3. (b)

Sujet p. 1

> Proposer sur la copie une ou plusieurs instructions à écrire entre la ligne 1 et la ligne 2 du code de la fonction tester_palindrome afin d'afficher clairement cette erreur à l'utilisateur.

On peut vérifier le type du paramètre dès le début de la fonction et lever une erreur avec un message explicite si ce n'est pas une chaîne. Cette vérification doit se trouver avant l'appel à `len`. Voici une version complète de la fonction, avec ce contrôle ajouté :

```
def tester_palindrome(chaine):
    if not isinstance(chaine, str):
        raise TypeError("chaine doit être une chaîne de caractères")
    if len(chaine) < 2:
        return True
    elif chaine[0] != chaine[-1]:
        return False
    else:
        chaine = chaine[1:-1]
        return tester_palindrome(chaine)
```

Avec `tester_palindrome(53235)`, l'exécution s'arrête sur le `raise` et affiche un `TypeError` accompagné du message explicite.

> **RÉPONSE À RETENIR**
>
> Insérer au début de la fonction le test du type et le `raise TypeError` montrés ci-dessus, avant l'appel à `len`.

Question 4.

Sujet p. 1

> Écrire le code d'une fonction itérative (non récursive) est_palindrome qui prend en paramètre une chaîne de caractères et renvoie un booléen égal à True si cette chaîne est un palindrome, False sinon.

On compare les caractères situés aux deux extrémités, puis on rapproche les indices vers le centre. Une différence suffit à conclure que la chaîne n'est pas un palindrome. Si les indices se rejoignent sans différence, tous les couples comparés sont égaux.

Exemple avec `kayak` : les caractères d'indices 0 et 4 sont `k` et `k`, puis ceux d'indices 1 et 3 sont `a` et `a`. La comparaison s'achève au centre, et la fonction renvoie `True`. Une chaîne vide ou composée d'un seul caractère renvoie aussi `True`, car la boucle ne s'exécute pas.

À chaque tour, l'écart entre les deux indices diminue : la boucle termine. Le temps d'exécution est proportionnel à la longueur de la chaîne dans le pire cas, et l'espace supplémentaire utilisé est constant.

RÉPONSE À RETENIR

```
def est_palindrome(chaine):
    gauche = 0
    droite = len(chaine) - 1

    while gauche < droite:
        if chaine[gauche] != chaine[droite]:
            return False
        gauche += 1
        droite -= 1

    return True
```

Exercice 2

5 points

Bases de données, SQL et réseaux

1. Étude du schéma relationnel

Question 1. (a)

Sujet p. 2, 3

> Pour chaque attribut de la relation cage, spécifier son type en utilisant le tableau des types fourni.

On attribue le type entier à l'identifiant numérique et un type texte aux deux attributs descriptifs. Le tableau donne `VARCHAR(t)` pour un texte de longueur variable, où `t` est une longueur maximale suffisante.

Les extraits montrent que `taille_cage` contient « petit », « moyen » ou « grand » et que `secteur_cage` contient notamment « rongeur ». Les longueurs exactes nécessaires pour toutes les valeurs possibles ne sont pas précisées ; les tailles ci-dessous sont donc des choix adaptés aux valeurs visibles.

RÉPONSE À RETENIR

Attribut	Type possible	Justification
num_cage	INT	Identifiant numérique entier.
taille_cage	VARCHAR(5)	Les valeurs visibles ont au plus 5 caractères.
secteur_cage	VARCHAR(7)	Les valeurs visibles ont au plus 7 caractères.

Les longueurs proposées sont des choix suffisants pour les valeurs de l'extrait ; elles ne sont pas imposées par l'énoncé.

Question 1. (b)

Sujet p. 2, 3

> Préciser, pour la relation reservation, le nom de la clé primaire pouvant être utilisée.

Une clé primaire identifie de façon unique chaque ligne d'une relation. Le numéro attribué à chaque réservation est l'attribut prévu pour distinguer ses enregistrements.

RÉPONSE À RETENIR

On peut utiliser `num_reservation` comme clé primaire de la relation `reservation`.

Question 1. (c)

Sujet p. 2, 3

> Indiquer, pour la relation reservation, la ou les clés étrangères (ou secondaires) et en indiquer l'utilité.

Une clé étrangère référence une clé primaire d'une autre relation. Elle établit un lien entre les tables et permet de vérifier que l'enregistrement référencé existe.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

- `num_client` référence `client.num_client` : il associe la réservation à un client.
- `num_animal` référence `animal.num_animal` : il associe la réservation à un animal.
- `num_cage` référence `cage.num_cage` : il associe la réservation à une cage.

Ces clés étrangères contribuent à assurer la cohérence des références dans la base de données.

2. Requêtes

Question 2. (a)

Sujet p. 2, 3

> Indiquer le résultat de l'exécution de la requête suivante : SELECT nom_animal FROM animal WHERE categorie_animal = 'chien';

La requête sélectionne le nom de chaque animal dont la catégorie est `'chien'`. Elle ne comporte pas `DISTINCT` : si deux lignes ont le même nom, ce nom apparaît deux fois. Dans l'extrait, les chiens sont Api et les deux animaux nommés Rex.

RÉPONSE À RETENIR

nom_animal
Api
Rex
Rex

Question 2. (b)

Sujet p. 2, 3

> Écrire une requête SQL permettant d'afficher les noms de tous les clients dont l'animal a occupé la cage numéro 23.

On relie les clients aux réservations à l'aide de leur numéro de client, puis on filtre les réservations qui utilisent la cage 23. `DISTINCT` évite de répéter le nom d'un client ayant plusieurs réservations dans cette cage.

```
SELECT DISTINCT c.nom_client
FROM client AS c
JOIN reservation AS r ON r.num_client = c.num_client
WHERE r.num_cage = 23;
```

Dans l'extrait fourni, les réservations de la cage 23 sont celles du client 345. Son nom, dans l'extrait de la table `client`, est Morel.

RÉPONSE À RETENIR

nom_client
Morel

FormaV - document d'entraînement non officiel

Question 2. (c)

Sujet p. 2, 3

> Un nouvel animal doit être enregistré dans la base de données, qui contient actuellement 491 animaux. Il s'appelle Suki, c'est un chat de petite taille dont le propriétaire a déjà été enregistré sous le numéro 342. Écrire la requête SQL permettant d'insérer ces nouvelles données dans la base de données.

La nouvelle ligne concerne la relation `animal`, dont les cinq attributs sont fournis dans le schéma. Puisque la base contient 491 animaux, on peut proposer le numéro 492 si les numéros sont attribués séquentiellement et si 492 est libre. Le nombre total d'animaux ne garantit pas à lui seul que ce numéro soit disponible : dans le cas contraire, il faut utiliser un numéro libre.

```
INSERT INTO animal (num_animal, nom_animal, categorie_animal, taille_animal, num_client)
VALUES (492, 'Suki', 'chat', 'petit', 342);
```

RÉPONSE À RETENIR

Cette requête insère Suki comme chat de petite taille, rattaché au client 342, avec le numéro 492 sous l'hypothèse qu'il est disponible.

3. Programmation Python

Question 3. (a)

Sujet p. 3

> À partir de la liste reservation et de la fonction mystere fournie, indiquer l'affichage produit par l'exécution de la ligne de code print(mystere(reservation, '2022-08-23')).

La fonction parcourt les réservations dans l'ordre et ajoute le numéro du client lorsque la date de début est exactement `'2022-08-23'`. Cette date apparaît dans les réservations 44 et 47, dont les numéros de client sont respectivement 26 et 345.

RÉPONSE À RETENIR

L'affichage produit est `[26, 345]`.

Question 3. (b)

Sujet p. 4

> Le responsable de la pension veut obtenir le nombre de réservations effectuées pour un numéro de client donné. Écrire les lignes de code après la ligne 6 de la fonction nombre_reservation afin de respecter sa spécification : table est une liste de dictionnaires représentant les réservations ; numero_client est un entier représentant le numéro du client concerné ; la valeur renvoyée est un entier donnant le nombre d'occurrences du numéro du client concerné.

On initialise un compteur à zéro, puis on examine chaque réservation. On l'incrémente si le champ `num_client` est égal au numéro recherché. Après la boucle, le compteur contient le nombre de réservations concernées.

```
def nombre_reservation(table, numero_client):
    compteur = 0
    for ligne in table:
        if ligne['num_client'] == numero_client:
            compteur += 1
    return compteur
```


FormaV - document d'entraînement non officiel

Avec les données du sujet et `numero_client = 345`, la réservation 44 ne correspond pas, tandis que les réservations 45, 46, 47 et 48 correspondent. Le compteur final vaut donc 4.

Trace pour numero_client = 345

Réservation	num_client	Compteur après traitement
44	26	0
45	345	1
46	345	2
47	345	3
48	345	4

Le compteur augmente d'une unité pour chaque réservation dont num_client vaut 345.

> **RÉPONSE À RETENIR**
>
> La fonction renvoie `4` pour le client 345 dans la liste fournie.

4. Protocole OSPF

Question 4. (a)

Sujet p. 4

> Pour passer de L1 à L2, le chemin R1 - R2 - R5 utilisant la fibre a le coût le plus faible. Calculer ce coût.

La formule est `C = 10^8 / d`, avec la bande passante `d` en bit/s. Sur le trajet R1–R2–R5, la liaison R1–R2 est en Fast-Ethernet : `d = 100 Mbit/s = 10^8 bit/s`, donc son coût vaut `10^8 / 10^8 = 1`. La liaison R2–R5 est en fibre : `d = 1 Gbit/s = 10^9 bit/s`, donc son coût vaut `10^8 / 10^9 = 0,1`. Le coût d'un chemin est la somme des coûts de ses liaisons.

> **RÉPONSE À RETENIR**
>
> Le coût du chemin R1–R2–R5 est **1 + 0,1 = 1,1**.

Question 4. (b)

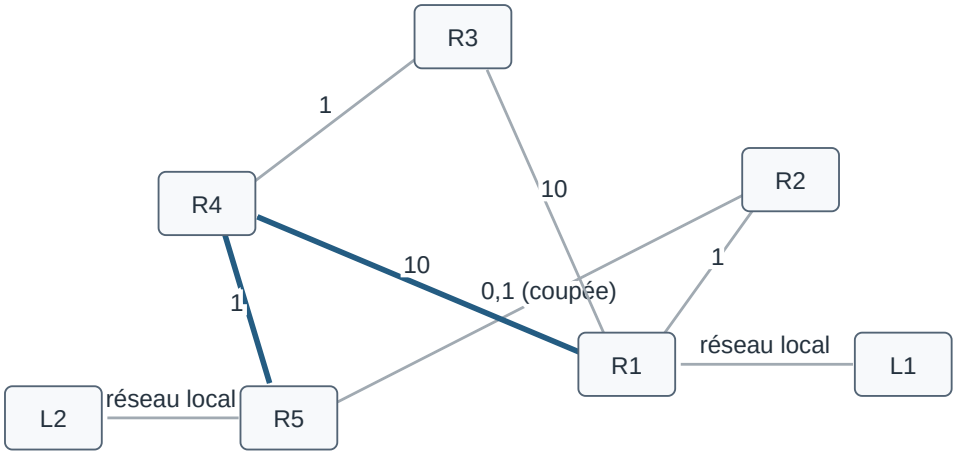
Sujet p. 4

> La liaison entre R2 et R5 a été coupée en raison de travaux. Déterminer la route permettant de relier le réseau L1 au réseau L2 selon le protocole OSPF. Justifier.

Après la coupure, la liaison R2–R5 ne peut plus être empruntée. Les coûts sont de 1 pour une liaison Fast-Ethernet de 100 Mbit/s et de 10 pour une liaison Ethernet de 10 Mbit/s. Le trajet R1–R4–R5 coûte donc `10 + 1 = 11` : R1–R4 est en Ethernet et R4–R5 en Fast-Ethernet. L'autre trajet simple possible, R1–R3–R4–R5, coûte `10 + 1 + 1 = 12`. Le passage par R2 ne permet plus de rejoindre R5. Les coûts étant positifs, un détour comportant une boucle ne peut pas améliorer le trajet.

FormaV - document d'entraînement non officiel

Réseau OSPF après la coupure de R2–R5



Coûts des liaisons entre routeurs : 1 pour Fast-Ethernet, 10 pour Ethernet et 0,1 pour la fibre. La liaison R2–R5 est coupée ; la route de coût minimal R1–R4–R5 est mise en évidence.

RÉPONSE À RETENIR

La route retenue est **L1–R1–R4–R5–L2**, soit R1–R4–R5 entre les routeurs. Son coût est **11**, inférieur au coût 12 de R1–R3–R4–R5.

FormaV - document d'entraînement non officiel

Exercice 3

4 points

Piles, arbres binaires et algorithmique

Question 1.

Sujet p. 5

> Donner la valeur de l'expression suivante écrite en notation polonaise inversée : 15 5 − 4 12 + × 2 /.

En notation polonaise inversée, on traite les opérateurs dans l'ordre où ils apparaissent, en appliquant chacun aux deux valeurs qui le précèdent immédiatement :

1. `15 5 −` donne `15 − 5 = 10`.
2. `4 12 +` donne `4 + 12 = 16`.
3. Il reste `10 16 × 2 /` : `10 × 16 = 160`, puis `160 / 2 = 80`.

RÉPONSE À RETENIR

La valeur de l'expression est **80**.

Question 2. — un parcours en largeur

Sujet p. 5

> Pour l'arbre binaire fourni, indiquer lequel des quatre parcours permet d'obtenir la succession des symboles correspondant à la notation polonaise inversée 20 11 3 + 5 2 + / −. Proposition : un parcours en largeur.

Un parcours en largeur visite les nœuds niveau par niveau. Pour l'arbre donné, il commence par la racine `−`, puis visite `20` et `/`. La succession demandée commence par `20` et se termine par l'opérateur racine `−` : elle ne peut donc pas être obtenue par un parcours en largeur.

RÉPONSE À RETENIR

Non. Un parcours en largeur ne donne pas la notation polonaise inversée demandée.

Question 2. — un parcours en profondeur préfixe

Sujet p. 5

> Pour l'arbre binaire fourni, indiquer lequel des quatre parcours permet d'obtenir la succession des symboles correspondant à la notation polonaise inversée 20 11 3 + 5 2 + / −. Proposition : un parcours en profondeur préfixe.

Un parcours en profondeur préfixe visite d'abord la racine, puis le sous-arbre gauche et le sous-arbre droit. Il commence donc ici par `−`, tandis que la succession demandée commence par `20`.

RÉPONSE À RETENIR

Non. Le parcours préfixe ne donne pas la succession demandée.

Question 2. — un parcours en profondeur infixe

Sujet p. 5

> Pour l'arbre binaire fourni, indiquer lequel des quatre parcours permet d'obtenir la succession des symboles correspondant à la notation polonaise inversée 20 11 3 + 5 2 + / −. Proposition : un parcours en profondeur infixe.

Un parcours en profondeur infixe visite le sous-arbre gauche, la racine, puis le sous-arbre droit. Sur l'arbre donné, il produit `20 − 11 + 3 / 5 + 2`, différent de la succession demandée `20 11 3 + 5 2 + / −`.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Non. Le parcours infixe ne donne pas la succession demandée.

Question 2. — un parcours en profondeur postfixe (ou suffixe)

Sujet p. 5

> Pour l'arbre binaire fourni, indiquer lequel des quatre parcours permet d'obtenir la succession des symboles correspondant à la notation polonaise inversée 20 11 3 + 5 2 + / −. Proposition : un parcours en profondeur postfixe (ou suffixe).

Un parcours en profondeur postfixe visite le sous-arbre gauche, puis le sous-arbre droit, et enfin la racine. Le sous-arbre gauche fournit `20`. Le sous-arbre droit fournit `11 3 + 5 2 + /`. La racine − est visitée en dernier : on obtient exactement la succession demandée.

RÉPONSE À RETENIR

Oui. C'est le parcours en profondeur postfixe (ou suffixe) qui donne `20 11 3 + 5 2 + / −`.

Question 3. (a) — LIFO

Sujet p. 6

> Parmi les acronymes LIFO et FIFO, indiquer lequel permet de décrire, de manière générale, la structure de pile et donner la signification des quatre lettres de l'acronyme retenu. Proposition : LIFO.

Dans une pile, le dernier élément empilé est le premier dépilé. Cette règle est décrite par l'acronyme LIFO.

RÉPONSE À RETENIR

Oui : LIFO signifie *Last In, First Out*, soit « dernier entré, premier sorti ».

Question 3. (a) — FIFO

Sujet p. 6

> Parmi les acronymes LIFO et FIFO, indiquer lequel permet de décrire, de manière générale, la structure de pile et donner la signification des quatre lettres de l'acronyme retenu. Proposition : FIFO.

FIFO désigne une structure où le premier élément entré est le premier sorti. C'est le principe d'une file, et non celui d'une pile.

RÉPONSE À RETENIR

Non. FIFO signifie *First In, First Out* (« premier entré, premier sorti »). Une pile est décrite par LIFO.

Question 3. (b) — nombre d'éléments

Sujet p. 6

> L'état de la pile P à la fin de l'exécution du quatrième tour de la boucle pour est représenté dans le sujet. Indiquer le nombre d'éléments contenus dans la pile P.

À la fin du quatrième tour, la pile représentée contient deux arbres : un arbre de racine + et de feuilles `2` et `3`, ainsi qu'un arbre réduit à la feuille `6`. Chaque arbre empilé compte comme un élément.

RÉPONSE À RETENIR

La pile P contient **2 éléments**.

FormaV - document d'entraînement non officiel

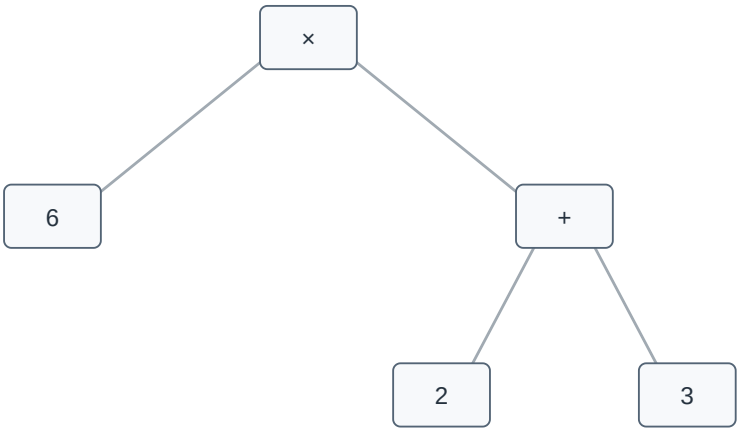
Question 3. (b) — état au cinquième tour

Sujet p. 6

> En appliquant l'algorithme à la liste [6, 2, 3, +, ×, 94, 1, −, +], dessiner l'état de la pile P à la fin de l'exécution du cinquième tour de la boucle pour. L'état de la pile à la fin du quatrième tour est représenté dans le sujet.

Au cinquième tour, le symbole lu est ×. L'algorithme dépile d'abord l'arbre de racine +, qui devient le sous-arbre droit, puis l'arbre réduit à `6`, qui devient le sous-arbre gauche. Il construit et empile ensuite l'arbre de racine ×. La pile contient alors un seul élément.

Élément de la pile P après le cinquième tour



P contient un seul élément : l'arbre représenté.

RÉPONSE À RETENIR

À la fin du cinquième tour, P contient un seul arbre, de racine ×, avec `6` à gauche et l'arbre de racine + à droite.

Question 3. (c)

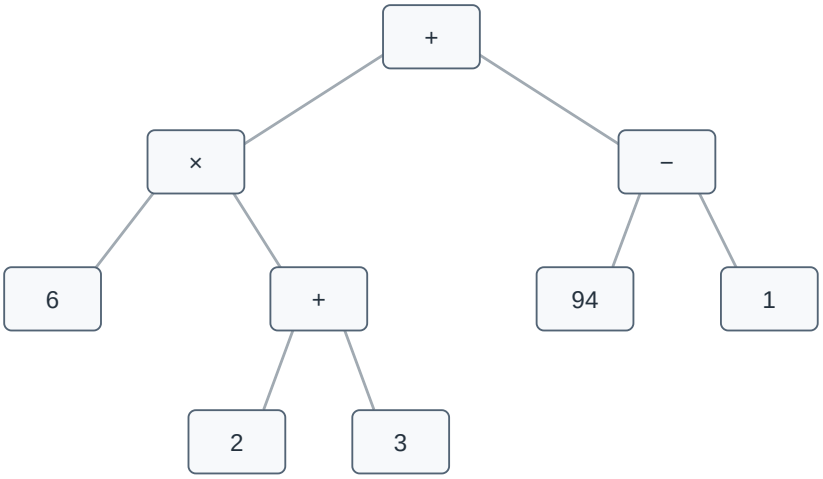
Sujet p. 6

> Dessiner l'arbre binaire représentant l'expression mathématique dont la notation polonaise inversée est la liste [6, 2, 3, +, ×, 94, 1, −, +].

On construit l'arbre en lisant la liste de gauche à droite. Les symboles `2 3 +` forment un arbre de racine +. Le symbole × réunit ensuite la feuille `6` à gauche et cet arbre à droite. Puis `94 1 −` forme un arbre de racine −. Enfin, le dernier + devient la racine, avec le produit à gauche et la différence à droite.

FormaV - document d'entraînement non officiel

Arbre binaire associé à la notation polonaise inversée



Les fils de gauche et de droite correspondent respectivement aux sous-arbres gauche et droit.

RÉPONSE À RETENIR

L'arbre représente `6 × (2 + 3) + (94 − 1)`.

Question 4.

Sujet p. 6, 7

Compléter le code Python de la fonction evaluer. Cette fonction prend en paramètre un arbre binaire arb représentant une expression mathématique écrite en notation polonaise inversée et renvoie la valeur de cette expression.

Une feuille a deux sous-arbres vides : on renvoie alors directement la valeur de sa racine. Dans les autres cas, la racine est un opérateur. On évalue récursivement les sous-arbres gauche et droit, puis on applique l'opérateur aux résultats, en conservant l'ordre pour la soustraction et la division.

RÉPONSE À RETENIR

```
def evaluer(arb):
    if est_vide(gauche(arb)) and est_vide(droit(arb)):
        res = racine(arb)
    elif racine(arb) == '+':
        res = evaluer(gauche(arb)) + evaluer(droit(arb))
    elif racine(arb) == '-':
        res = evaluer(gauche(arb)) - evaluer(droit(arb))
    elif racine(arb) == '*':
        res = evaluer(gauche(arb)) * evaluer(droit(arb))
    else:
        res = evaluer(gauche(arb)) / evaluer(droit(arb))
    return res
```

Pour l'arbre de la question 3(c), le résultat est `6 × (2 + 3) + (94 − 1) = 30 + 93 = 123`.

Sujet : https://aeif.fr/wp-content/uploads/2025/07/Liban_mars_2023_1.pdf
© FormaV EI • Proposition de correction, distincte d'un corrigé officiel.

FormaV - document d'entraînement non 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.