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 2023 • Asie • Jour 2

Code sujet **23-NSIJ2JA1** 3 heures 30

Proposition de corrigé pédagogique • FormaV

Se repérer dans le corrigé

Exercice 1 — Protocoles RIP et OSPF, adressage IPv4 et système d’exploitation
11 réponses • sujet p. 2–4

Exercice 2 — Bases de données relationnelles et langage SQL
7 réponses • sujet p. 5–6

Exercice 3 — Algorithmes de parcours et déplacements sur un plateau
7 réponses • sujet p. 7–8

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

Protocoles RIP et OSPF, adressage IPv4 et système d'exploitation

A

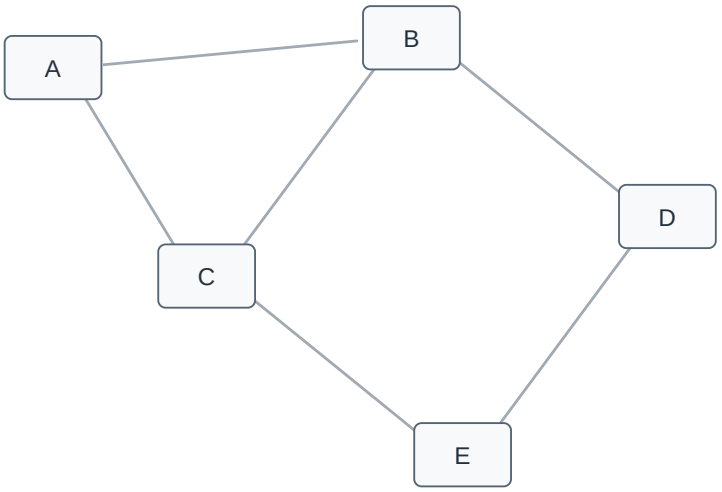
Question 1.

Sujet p. 2

> À l'aide des tables de routage des routeurs A, B, C et D, schématiser le réseau composé des cinq routeurs A, B, C, D et E.

Dans une table de routage, une destination à distance 1 est directement reliée au routeur. On lit ainsi les liaisons A–B et A–C, B–C et B–D, C–E et D–E. Les distances supérieures à 1 sont cohérentes avec ces liaisons : par exemple, depuis A, D est atteint en deux sauts par B.

Réseau des routeurs A, B, C, D et E



Les liaisons sont déduites des destinations accessibles en un saut dans les tables de routage.

RÉPONSE À RETENIR

Le réseau comporte les liaisons A–B, A–C, B–C, B–D, C–E et D–E.

Question 2.

Sujet p. 2

> En déduire une table de routage possible pour le routeur E.

Les tables indiquent que E est directement relié à C et à D : ces destinations sont donc à distance 1 depuis E. A est à deux sauts par C. B est également à deux sauts, par C ou par D ; on choisit ici C pour la table proposée. Comme dans les tables fournies, la destination E elle-même n'est pas inscrite.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Destination	Passe par	Distance
A	C	2
B	C	2
C	C	1
D	D	1

Question 3.

Sujet p. 2

> La connexion entre le routeur A et C est coupée. Déterminer alors la nouvelle table de routage du routeur A.

Après la coupure de A–C, A reste relié directement à B. Depuis B, C et D sont chacun à un saut : ils sont donc à deux sauts de A. E est accessible en trois sauts, soit par A–B–C–E, soit par A–B–D–E. La table ci-dessous choisit le passage par C pour E.

RÉPONSE À RETENIR

Destination	Passe par	Distance
B	B	1
C	B	2
D	B	2
E	B	3

B

Question 1.

Sujet p. 3

> Recopier et compléter le tableau des coûts des liaisons Ethernet, Fast-Ethernet et Fibre. Le coût d'une liaison est donné par la formule coût = $10^8/d$, où d correspond au débit exprimé en bits par seconde. Les débits indiqués dans le tableau sont 10^7, 10^8 et 10^9 bits par seconde ; le coût de la Fibre est déjà indiqué.

On applique la formule coût = 10^8 / d, avec le débit d exprimé en bits par seconde. Pour Ethernet, le coût vaut 10^8 / 10^7 = 10. Pour Fast-Ethernet, il vaut 10^8 / 10^8 = 1. Pour la Fibre, il vaut 10^8 / 10^9 = 0,1, ce qui correspond à la valeur déjà inscrite dans le tableau du sujet.

RÉPONSE À RETENIR

Type de liaison	Ethernet	Fast-Ethernet	Fibre
Débit	10^7	10^8	10^9
Coût	10	1	0,1

FormaV - document d'entraînement non officiel

Question 2.

Sujet p. 3

> On souhaite acheminer un paquet de données depuis le routeur R1 au routeur R4. Donner la liste des six chemins possibles que peut emprunter ce paquet.

On énumère les chemins simples, c'est-à-dire sans repasser par un routeur déjà traversé. En suivant les liaisons de la figure, R1 peut partir vers R2, R3 ou R5. En poursuivant sans revenir sur ses pas et jusqu'à R4, on obtient les six chemins ci-dessous.

RÉPONSE À RETENIR

1. R1–R3–R4
2. R1–R3–R5–R4
3. R1–R5–R4
4. R1–R5–R3–R4
5. R1–R2–R3–R4
6. R1–R2–R3–R5–R4

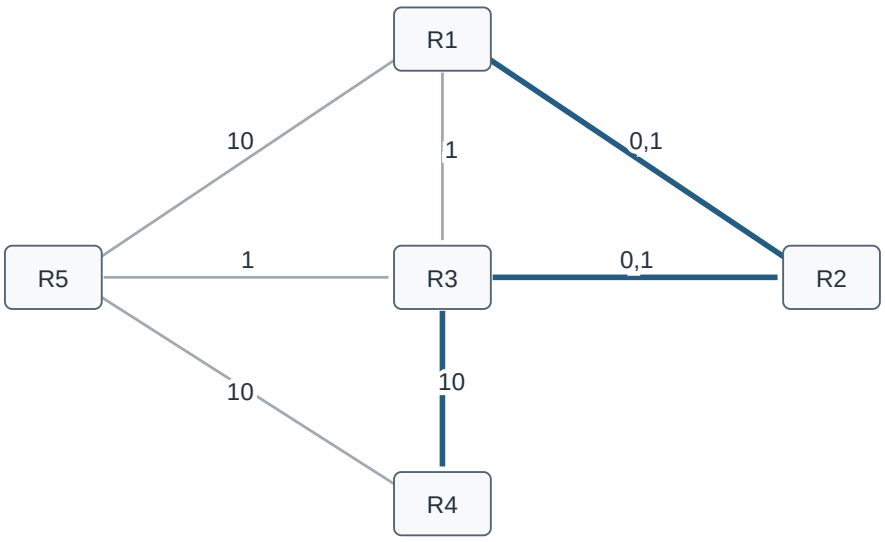
Question 3.

Sujet p. 3

> Déterminer le coût total de chacun de ces chemins puis en déduire le chemin respectant le protocole OSPF.

Le coût total d'un chemin est la somme des coûts de ses liaisons. D'après la figure et le tableau complété, une liaison Ethernet coûte 10, une liaison Fast-Ethernet coûte 1 et une liaison Fibre coûte 0,1. On calcule chaque somme et OSPF retient le chemin dont le coût total est minimal.

Réseau OSPF et chemin de coût minimal



Les poids correspondent aux coûts des liaisons indiqués dans la figure. Le chemin mis en évidence est R1–R2–R3–R4, de coût 10,2.

FormaV - document d'entraînement non officiel

RÉPONSE À RETENIR

Chemin	Calcul du coût	Coût total
R1–R3–R4	1 + 10	11
R1–R3–R5–R4	1 + 1 + 10	12
R1–R5–R4	10 + 10	20
R1–R5–R3–R4	10 + 1 + 10	21
R1–R2–R3–R4	0,1 + 0,1 + 10	10,2
R1–R2–R3–R5–R4	0,1 + 0,1 + 1 + 10	11,2

Le plus petit coût est 10,2. Le chemin choisi par OSPF est donc R1–R2–R3–R4.

C

Question 1.

Sujet p. 4

> L'adresse IPv4 du PC1 est 172.20.14.26/16. Donner l'adresse de ce réseau.

Le préfixe /16 indique que les 16 premiers bits, soit les deux premiers octets, forment la partie réseau. Pour obtenir l'adresse du réseau, on conserve donc 172.20 et on met à zéro les deux octets de la partie hôte.

RÉPONSE À RETENIR

L'adresse du réseau est **172.20.0.0**, soit **172.20.0.0/16** en notation CIDR.

Question 2.

Sujet p. 4

> En déduire le nombre de machines pouvant être connectées à ce réseau.

Il reste 32 − 16 = 16 bits pour la partie hôte, soit 2^{16} adresses. Dans le décompte usuel d'un réseau IPv4, l'adresse du réseau et l'adresse de diffusion ne sont pas attribuables à des machines : on les retranche.

RÉPONSE À RETENIR

Le nombre de machines pouvant être adressées est $2^{16} - 2$ = **65 534**.

Question 3.

Sujet p. 4

> L'interface du routeur a pour adresse IPv4 172.20.14.254/16. Donner une adresse IPv4 possible pour la machine PC3 appartenant à ce réseau.

PC3 doit appartenir au réseau 172.20.0.0/16 : ses deux premiers octets doivent donc être 172.20. Il faut choisir une adresse d'hôte qui ne soit ni l'adresse du réseau, ni l'adresse de diffusion, ni une adresse déjà utilisée. Par exemple, 172.20.14.27 est différente de l'adresse de PC1 (172.20.14.26) et de celle de l'interface du routeur (172.20.14.254). On suppose, comme dans toute proposition d'adresse, qu'aucune autre machine n'utilise déjà cette adresse.

RÉPONSE À RETENIR

Une adresse possible pour PC3 est **172.20.14.27/16**.

Question 4.

Sujet p. 4

> L'une de ces machines a pour adresse IPv4 172.20.14.16. Déterminer l'écriture binaire de l'adresse de cette machine.

On convertit chaque octet en binaire sur 8 bits : 172 = 10101100, 20 = 00010100, 14 = 00001110 et 16 = 00010000. On assemble ces quatre groupes dans le même ordre que les octets de l'adresse IPv4.

RÉPONSE À RETENIR

L'écriture binaire de 172.20.14.16 est **10101100.00010100.00001110.00010000**.

Question 5.

Sujet p. 4

> L'utilisateur du PC2 souhaiterait connaître l'adresse IPv4 de sa machine. Donner la commande du terminal du système d'exploitation du PC2 permettant de l'obtenir.

Le système d'exploitation de PC2 n'est pas précisé, donc la commande dépend de celui qui est installé. Sous Windows, la commande usuelle est `ipconfig`. Sous Linux, la commande `ip addr` affiche les adresses des interfaces ; il faut repérer celle utilisée par PC2.

RÉPONSE À RETENIR

Sous Windows : `ipconfig`. Sous Linux : `ip addr`.

Exercice 2

3 points

Bases de données relationnelles et langage SQL

1

Question a.

Sujet p. 5, 6

> Représenter par un schéma relationnel les relations candidats et etablissements en précisant les types pour les attributs.

On relève dans les commandes de création chaque attribut et son type. Les attributs déclarés comme clés primaires sont `identifiant` dans `candidats` et `id_etab` dans `etablissements`. La déclaration de clé étrangère de `candidats` relie `id_etab` à `etablissements.id_etab`.

Relation	Schéma et types
candidats	`identifiant` INTEGER (clé primaire), `prenom` TEXT, `nom` TEXT, `email` TEXT, `telephone` TEXT, `moyenne` REAL, `id_etab` INTEGER (clé étrangère)
etablissements	`id_etab` INTEGER (clé primaire), `etablissement` TEXT, `adresse` TEXT, `mail` TEXT, `proviseur` TEXT

Les attributs `identifiant`, `prenom`, `nom` et `id_etab` de `candidats` sont déclarés `NOT NULL`. Dans `etablissements`, `id_etab` est déclaré `NOT NULL`.

> **RÉPONSE À RETENIR**
>
> **Schéma relationnel :** `candidats(identifiant INTEGER [PK], prenom TEXT, nom TEXT, email TEXT, telephone TEXT, moyenne REAL, id_etab INTEGER [FK])` et `etablissements(id_etab INTEGER [PK], etablissement TEXT, adresse TEXT, mail TEXT, proviseur TEXT)`.

Question b.

Sujet p. 5, 6

> Indiquer, en justifiant, quelle est la clé étrangère de la relation candidats.

La création de la relation `candidats` indique explicitement `FOREIGN KEY("id_etab") REFERENCES "etablissements"("id_etab")`. La clé étrangère est donc `candidats.id_etab` : elle référence la clé primaire `etablissements.id_etab`. Dans les données fournies, par exemple, l'identifiant 2 correspond à Acamus.

> **RÉPONSE À RETENIR**
>
> **Clé étrangère :** `candidats.id_etab`, qui référence `etablissements.id_etab`.

Question c.

Sujet p. 5, 6

> Écrire une requête permettant d'ajouter le candidat présentant les renseignements suivants : identifiant 8 ; email jacques.haddy@gmail.com ; moyenne 9.75 ; nom HADDY ; prénom Jacques ; téléphone 98 76 54 32 10 ; établissement Acamus.

La relation `candidats` stocke l'identifiant de l'établissement dans `id_etab` : il faut donc utiliser l'identifiant de l'établissement Acamus, qui vaut 2 dans la relation `etablissements`. La requête insère les renseignements donnés, dont l'identifiant 8. Le téléphone est écrit comme une chaîne, car son type est `TEXT`.

```
INSERT INTO candidats (identifiant, prenom, nom, email, telephone, moyenne, id_etab)
VALUES (8, 'Jacques', 'HADDY', 'jacques.haddy@gmail.com', '98 76 54 32 10', 9.75, 2);
```

> **RÉPONSE À RETENIR**
>
> La requête ajoute Jacques HADDY avec l'identifiant 8, la moyenne 9.75 et la référence à Acamus, représentée par `id_etab = 2`.

2

Question a.

Sujet p. 6

> Écrire ce qu'affiche la requête suivante appliquée aux relations précédentes : SELECT prenom, nom FROM candidats WHERE moyenne > 12.5 ;

La condition `moyenne > 12.5` conserve les lignes dont la moyenne est strictement supérieure à 12.5. Dans la relation fournie, il s'agit de Lydie HECTOR (18.5), Mathis PARIS (19.5) et Roman LAMI (19.5). Kevin APPA, par exemple, n'est pas retenu car sa moyenne est 12.2.

> **RÉPONSE À RETENIR**
>
> La requête affiche :

prenom	nom
Lydie	HECTOR
Mathis	PARIS
Roman	LAMI

Question b.

Sujet p. 6

> Écrire une requête permettant d'afficher les noms, prénoms et moyennes des candidats venant de l'établissement dont l'id_etab vaut 1.

Comme `id_etab` figure dans `candidats`, on peut filtrer directement cette relation sur la valeur 1, puis sélectionner les trois attributs demandés.

```
SELECT nom, prenom, moyenne
FROM candidats
WHERE id_etab = 1;
```

Dans les données du sujet, les candidats concernés sont Kevin APPA, Claire TOU et Mathis PARIS.

RÉPONSE À RETENIR

nom	prenom	moyenne
APPA	Kevin	12.2
TOU	Claire	9.5
PARIS	Mathis	19.5

Question c.

Sujet p. 6

> Écrire ce qu'affiche la requête suivante : SELECT nom, moyenne FROM candidats WHERE moyenne < 16 ORDER BY nom ;

La condition `moyenne < 16` retient Kevin APPA (12.2), Paul TARDI (11.5), Pierre PASTA (11.5) et Claire TOU (9.5). La clause `ORDER BY nom` trie ensuite ces lignes par ordre croissant du nom : APPA, PASTA, TARDI, TOU.

RÉPONSE À RETENIR

nom	moyenne
APPA	12.2
PASTA	11.5
TARDI	11.5
TOU	9.5

Question 3.

Sujet p. 6

> Écrire une requête donnant les noms des établissements et de leurs proviseurs classés par ordre croissant des notes.

La relation `etablissements` ne comporte pas d'attribut de note. J'interprète donc « notes » comme les moyennes des candidats, stockées dans `candidats.moyenne`. Il faut joindre les deux relations grâce à la correspondance entre les attributs `id_etab`, puis trier selon la moyenne. La requête affiche une ligne par candidat : un établissement peut donc apparaître plusieurs fois.

```
SELECT etablissements.etablissement, etablissements.proviseur
FROM candidats
JOIN etablissements ON candidats.id_etab = etablissements.id_etab
ORDER BY candidats.moyenne;
```


FormaV - document d'entraînement non officiel

Les moyennes des lignes obtenues sont, dans l'ordre : 9.5, 11.5, 11.5, 12.2, 18.5, 19.5 et 19.5. En cas d'égalité, cette requête ne fixe pas l'ordre entre les lignes de même moyenne.

RÉPONSE À RETENIR

etablissement	proviseur
Vinci	Gerard LECHEF
Brassens	Philippe STAR
Brassens	Philippe STAR
Vinci	Gerard LECHEF
Acamus	Camille PREMIERE
Vinci	Gerard LECHEF
Acamus	Camille PREMIERE

Les deux lignes à 11.5 correspondent à Paul TARDI et Pierre PASTA, et les deux lignes à 19.5 à Mathis PARIS et Roman LAMI. L'ordre au sein de chacune de ces égalités n'est pas déterminé par la requête.

FormaV - document d'entraînement non officiel

Exercice 3

5 points

Algorithmes de parcours et déplacements sur un plateau

1

Question a.

Sujet p. 7

> Donner sans le justifier le plus haut score possible, si le plateau est défini par la liste [-2, 4, -1, -2, 1, 4].

Depuis le départ, on peut avancer d'une case ou en sauter une. Pour vérifier le maximum, on calcule le meilleur score permettant d'atteindre chaque case, puis l'arrivée. Pour la liste $(-2, 4, -1, -2, 1, 4)$, les meilleurs scores aux six cases sont respectivement $-2, 4, 3, 2, 4, 8$: à chaque étape, on ajoute la valeur de la case au meilleur des scores des deux cases précédentes (ou du départ, au début). L'arrivée peut être atteinte depuis les deux dernières cases ; le meilleur score est donc $\max(4, 8) = 8$.

RÉPONSE À RETENIR

Le plus haut score possible est **8**.

Question b.

Sujet p. 7

> Si le plateau ne contient que des valeurs positives, quelle est la meilleure stratégie ?

Les valeurs étant toutes positives, passer par une case supplémentaire ajoute un nombre positif au score. La stratégie optimale est donc de ne sauter aucune case.

RÉPONSE À RETENIR

Il faut passer par toutes les cases, en se déplaçant toujours sur la case juste à droite.

2

Question a.

Sujet p. 7

> Écrire une fonction parcours1(e) qui prend en paramètre une liste e représentant le plateau et renvoie le score réalisé si le joueur choisit comme stratégie de passer par toutes les cases.

La stratégie consiste à additionner chaque valeur de la liste une fois. La boucle visite chaque élément, donc le temps d'exécution est proportionnel à la longueur de la liste.

```
def parcours1(e):
    score = 0
    for valeur in e:
        score = score + valeur
    return score
```

Pour le plateau de l'exemple, la somme vaut $(4 - 3 - 1 + 3 - 2 - 3 = -2)$. La fonction renvoie donc -2.

RÉPONSE À RETENIR

La fonction est celle donnée ci-dessus ; pour le plateau $[4, -3, -1, 3, -2, -3]$, elle renvoie **-2**.

FormaV - document d'entraînement non officiel

Question b.

Sujet p. 7

> Pour la question suivante, on suppose que le nombre de cases est impair. Écrire une fonction parcours2(e) qui prend en paramètre une liste e représentant le plateau et renvoie le score réalisé si le joueur choisit comme stratégie de ne faire que des sauts.

Un saut fait avancer de deux cases. En partant du départ, on atterrit donc sur les valeurs d'indices 1, 3, 5, etc. (indices Python commençant à 0). Si la longueur est impaire, ces sauts successifs atteignent l'arrivée après la dernière case visitée.

```
def parcours2(e):
    score = 0
    for i in range(1, len(e), 2):
        score = score + e[i]
    return score
```

Par exemple, pour le plateau impair $[4, -3, -1, 3, -2]$, les sauts font gagner -3 puis 3, avant d'atteindre l'arrivée. Le score est $-3 + 3 = 0$. Le plateau de l'exemple imprimé dans l'énoncé a six cases : il ne satisfait pas l'hypothèse d'imparité de cette question.

> **RÉPONSE À RETENIR**
>
> La fonction renvoie la somme des valeurs d'indices impairs. Pour $[4, -3, -1, 3, -2]$, elle renvoie **0**.

Question 3.

Sujet p. 7, 8

> On modélise les choix du joueur par une chaîne de caractères qui contient les caractères « D » (le joueur se déplace sur la case juste à droite) ou « XS » (le joueur saute une case toujours sur la droite). Écrire une fonction score(plateau, deplacements) qui reçoit en paramètre un plateau sous forme de liste et un choix de déplacements donné sous forme d'une chaîne de caractères et retourne le score réalisé par ce choix. Dès que l'arrivée est atteinte, le score est renvoyé. On considère que la chaîne de caractères permet d'atteindre la case arrivée.

On représente le départ par la position -1 et les cases du plateau par les positions 0 à $\text{len}(plateau) - 1$. Le caractère « D » avance d'une position ; la paire « XS » avance de deux positions. À chaque arrivée sur une case, on ajoute sa valeur. L'exemple "$DDXSDDD$" se lit comme « D », « D », « XS », « D », « D », « D » : il visite les valeurs 4, -3, 3, -2 et -3, puis atteint l'arrivée.

FormaV - document d'entraînement non officiel

```
def score(plateau, deplacements):
    position = -1
    total = 0
    i = 0

    while position < len(plateau):
        if deplacements[i] == "D":
            position = position + 1
            i = i + 1
        else:
            # Le déplacement "XS" avance de deux cases.
            position = position + 2
            i = i + 2

        if position < len(plateau):
            total = total + plateau[position]

    return total
```

Pour l'exemple du sujet, le calcul est $(4 - 3 + 3 - 2 - 3 = -1)$. La boucle s'arrête dès que la position atteint l'arrivée, sans ajouter de valeur pour celle-ci. Elle suppose, comme l'énoncé, que la chaîne est bien formée et permet d'atteindre l'arrivée.

> **RÉPONSE À RETENIR**
>
> Pour $score([4, -3, -1, 3, -2, -3], "DDXSDDD")$, le résultat est **-1**.

Question 4.

Sujet p. 8

> On souhaite écrire une fonction meilleur(plateau) qui génère 1 000 déplacements dont la longueur et le nombre de cases du plateau permettent d'assurer d'arriver au bout et qui renvoie le meilleur score sur ces 1 000 déplacements. Recopier puis compléter la fonction fournie.

La boucle doit réaliser exactement 1 000 essais ; on utilise donc `range(1000)`. Chaque score est comparé au meilleur score rencontré et le maximum est conservé. Initialiser avec `parcours1(plateau)` est valide puisque passer par toutes les cases constitue un parcours possible.

Il y a toutefois une incohérence dans l'énoncé tel qu'il est imprimé : il demande de générer une chaîne de longueur égale au nombre de cases, alors que le départ est une case distincte avant les valeurs du plateau et que l'arrivée est après celles-ci. Pour un plateau de longueur m, il faut avancer de $m + 1$ positions du départ à l'arrivée. Comme « D » avance d'une position et « XS » de deux pour deux caractères, une chaîne de longueur m ne garantit pas d'atteindre l'arrivée. La complétion ci-dessous utilise donc `len(plateau) + 1` ; c'est la correction nécessaire pour respecter le dessin et la fonction `score`.

```
def meilleur(plateau):
    score_max = parcours1(plateau)
    for i in range(1000):
        d = genere(len(plateau) + 1)
        s = score(plateau, d)
        if s > score_max:
            score_max = s
    return score_max
```

Avec le plateau de six valeurs de l'exemple, la chaîne générée doit avoir une longueur de 7 pour garantir l'arrivée depuis le départ. Le score initial vaut -2 ; le score maximal observé est ensuite mis à jour seulement lorsqu'un essai dépasse cette valeur. La fonction renvoie le meilleur score parmi le parcours initial et les 1 000 essais.

> **RÉPONSE À RETENIR**
>
> Les compléments sont `range(1000)`, `len(plateau) + 1` et `s > score_max`, avec la correction de longueur expliquée ci-dessus : la longueur `len(plateau)` indiquée dans l'énoncé ne garantit pas l'arrivée si le départ est distinct de la première case.

Question 5.

Sujet p. 8

> Écrire la fonction genere(n) de la question 3. Elle reçoit en paramètre un entier naturel n non nul et renvoie une chaîne de caractères de longueur n ne contenant que les caractères « D » et « XS », choisis de manière aléatoire, et permettant d'atteindre la case arrivée. Pour cela, on pourra utiliser la fonction randint du module random, qui reçoit deux entiers a et b et renvoie un entier de l'intervalle [a,b], a et b inclus.

On construit la chaîne en choisissant des déplacements de sorte que le nombre de caractères restant à produire diminue à chaque étape. « D » utilise un caractère et « XS » en utilise deux. Quand il ne reste qu'un caractère à produire, le choix « XS » est impossible ; on doit donc ajouter « D ». Chaque déplacement fait avancer d'autant de positions qu'il consomme de caractères. Ainsi, la chaîne renvoyée a exactement n caractères et correspond à une avancée totale de n positions.

```
from random import randint

def genere(n):
    deplacements = ""
    restant = n

    while restant > 0:
        if restant == 1:
            deplacements = deplacements + "D"
            restant = restant - 1
        elif randint(0, 1) == 0:
            deplacements = deplacements + "D"
            restant = restant - 1
        else:
            deplacements = deplacements + "XS"
            restant = restant - 2

    return deplacements
```

Par exemple, pour $n = 8$, les choix « D », « XS », « XS », « D », « D », puis « D » donnent `"DXSXSDDD"`. Sa longueur est $(1 + 2 + 2 + 1 + 1 + 1 = 8)$, et les déplacements font avancer de huit positions. Pour garantir l'arrivée sur un plateau de m valeurs avec le départ distinct figuré dans le sujet, il faut appeler cette fonction avec $m + 1$; cela met en évidence la même incohérence de longueur que pour la question 4.

> **RÉPONSE À RETENIR**
>
> La fonction renvoie une chaîne de longueur exactement n composée des commandes « D » et « XS ». Elle garantit une avancée totale de n positions ; pour un plateau de m cases où le départ est distinct, il faut donc utiliser `genere(m + 1)` pour garantir l'arrivée.

© 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.