BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

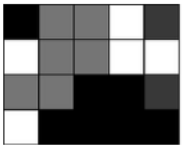
Sujet n°1

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

On considère dans ce sujet des images en niveaux de gris, c'est-à-dire composées de pixels décrits par une valeur entre 0 et 255 représentant l'intensité du niveau de gris, allant de noir pour la valeur 0 à blanc pour la valeur 255. Une image est alors vue comme une liste de valeurs entre 0 et 255 données par les valeurs des pixels ligne par ligne.



$$\begin{bmatrix} 0 & 128 & 128 & 255 & 64 \\ 255 & 128 & 128 & 255 & 255 \\ 128 & 128 & 0 & 0 & 64 \\ 255 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 1. Exemple d'image en niveaux de gris et sa représentation en tableau de valeurs qui, une fois aplatie, devient `[0, 128, 128, 255, 64, 255, 128, 128, 255, 255, 128, 128, 0, 0, 64, 255, 0, 0, 0, 0]` et de laquelle on peut retrouver l'image en connaissant sa largeur.

Les images manipulées sont des dessins ou des schémas présentant de grandes zones d'un même gris, l'objectif de ce problème est d'étudier une manière de les représenter efficacement en tirant parti de cette information. Pour cela, on remarque que les aplats font apparaître des valeurs qui se répètent dans les niveaux de gris des pixels, on va donc remplacer chaque suite de valeurs identiques par un couple (compte, valeur) indiquant le nombre de fois qu'une valeur est répétée ainsi que cette valeur. La liste de couples est elle-même aplatie, c'est-à-dire représentée par une liste de longueur paire `[compte1, valeur1, compte2, valeur2, ...]`. On appelle codage RLE de l'image cette nouvelle liste, cela vient de l'anglais *run-length encoding*, car une suite de valeur identique est appelée *run*.

Exemple: La liste `[4, 4, 4, 0, 5, 5]` présente trois fois de suite la valeur `4`, ce qui correspond au couple `(3, 4)`, une fois la valeur `0` donc le couple `(1, 0)` et enfin deux fois la valeur `5`, donc le couple `(2, 5)`. La liste admet alors pour codage RLE la nouvelle liste `[3, 4, 1, 0, 2, 5]`. Ici, les deux listes sont de même longueur, mais si la liste initiale était `[0, 0, 0, 0, 0]` on aurait obtenu la liste `[5, 0]` plus courte.

Question 1

Déterminer si la liste obtenue par codage RLE est forcément de longueur inférieure ou égale à la liste de départ.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

En étudiant bien la fonction `codage_rle` qui réalise le codage, écrire le corps de la fonction `decodage_rle` qui réalise le décodage d'une liste. Des tests sont fournis dans la fonction `test_codage`, on pourra les compléter.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

Pour tester le codage sur une image, on peut utiliser la fonction fournie `encoder_decoder_image` qui effectue le codage puis le décodage d'une image pour l'enregistrer à nouveau dans un fichier. Utiliser cette fonction sur les images `bac_nsi_32.png` et `bac_nsi_256.png` et observer la différence de comportement.

Question 4

Le problème précédent est lié au fait que sur des grandes images, il est possible d'avoir plus de 255 pixels de la même couleur. Proposer une démarche de résolution de ce problème qui modifie les fonctions d'encodage et de décodage, puis l'implémenter.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `rle.py` ;
- des images au format `png` en niveaux de gris auquel le sujet fait référence.

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `pillow`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°2

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Des écarts de salaires subsistent entre les femmes et les hommes, même à poste équivalent. En France, l'écart de salaire moyen est encore d'environ 15 % en 2023 selon l'INSEE.

Afin d'observer cet écart et ces conséquences, on dispose de jeux de données représentant les employés d'une entreprise. Chaque jeu de données est une liste de dictionnaires où chaque dictionnaire représente un employé avec les champs suivants et le type des valeurs associées :

- `'experience'` (int, en années, indiquant l'expérience professionnelle) ;
- `'etudes'` (int en années, indiquant le nombre d'années d'étude après l'obtention du baccalauréat) ;
- `'sexe'` (`str`, `'F'` ou `'M'`) ;
- `'salaire'` (`int`, en euros).

Deux jeux de données vous sont fournis, un premier jeu de données de test dans le fichier donnees.py dont le contenu est reproduit ci-dessous et un jeu de données plus complet de 2000 employés dans le fichier donnees_completes.py.

```
employes = [
    {'experience': 5, 'etudes': 3, 'sexe': 'F', 'salaire': 2400},
    {'experience': 3, 'etudes': 3, 'sexe': 'M', 'salaire': 2550},
    {'experience': 5, 'etudes': 5, 'sexe': 'F', 'salaire': 2500},
    {'experience': 3, 'etudes': 5, 'sexe': 'M', 'salaire': 2800},
    {'experience': 2, 'etudes': 5, 'sexe': 'F', 'salaire': 2300},
    {'experience': 2, 'etudes': 3, 'sexe': 'M', 'salaire': 2700}
]
```

Le fichier `analyse.py` présente des éléments d'analyse de ces jeux de données qui vont être complétés et améliorés dans les questions qui suivent.

Question 1

Écrire le code de la fonction salaire_moyen_condition qui prend en paramètres un tableau d'employés dans le format décrit ci-dessus, le nom d'un des trois champs 'experience', 'etudes' ou 'sexe' ainsi qu'une valeur et qui renvoie un nombre flottant (type `float`) correspondant au salaire moyen des employés dont la valeur associée au champ est la valeur fournie.
La fonction doit renvoyer `None` s'il n'y a pas d'employés ayant la valeur recherchée pour le champ donné.
Ainsi, un appel à `salaire_moyen_condition(employes, 'sexe', 'F')` perme-

ttra de renvoyer le salaire moyen des femmes employées.
Une fonction de test sur le jeu de données de test est fournie. Déterminer le salaire moyen des femmes et des hommes pour le jeu de données complet.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire en Python une fonction nommée effectif_par_sexe qui prend en paramètre un tableau non vide d'employés et qui renvoie un dictionnaire ayant deux clés 'F' et 'H' associées respectivement à l'effectif des femmes et à l'effectif des hommes employés.
Par exemple, avec le tableau `employes` précédent :

```
>>> effectif_par_sexe(employes)
{'F':3, 'M':3)
```

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On définit l'écart de salaire moyen en pourcentage comme :

$$\text{écart} = \frac{\text{salaire moyen hommes} - \text{salaire moyen femmes}}{\text{salaire moyen hommes}} \times 100$$

Une fonction `calcule_ecart_sexe` est écrite dans le fichier `analyse.py`

Question 3

Expliquer pourquoi le code de cette fonction est incorrect et proposer quelques tests simples sous forme d'assertions qui permettent de mettre ce ou ces problèmes en évidence :

- vérifier que le résultat est None dans le cas où un seul sexe est présent ;
- vérifier qu'un écart de salaires exprimé en pourcentage soit toujours compris entre 0 et 100.

Proposer une version corrigée de la fonction `ecart_salaire` qui valide ces tests et renvoie le bon écart. En déduire l'écart de salaire moyen dans les données complètes.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

Afin de proposer un salaire d'embauche à un nouvel employé, le service informatique de l'entreprise a proposé d'utiliser l'algorithme des k-plus proches voisins et renvoyer la moyenne des salaires des trois employés aux caractéristiques les plus proches.
La fonction `salaire_par_proximite` effectue ce calcul pour faire une proposition.

Tester et comparer les salaires proposés aux deux futurs employés suivants :

```
{'experience': 3, 'etudes': 3, 'sexe': 'F'}
{'experience': 3, 'etudes': 3, 'sexe': 'M'}
```

Identifier la source des écarts entre les deux propositions de salaire dans le programme et la corriger.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier fourni

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- Une version PDF de l'énoncé ;
- Un code source à analyser et compléter `analyse.py` ;
- un jeu de données de tests `donnees.py` ;
- un jeu de données complet de 2000 employés `donnees_completes.py`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

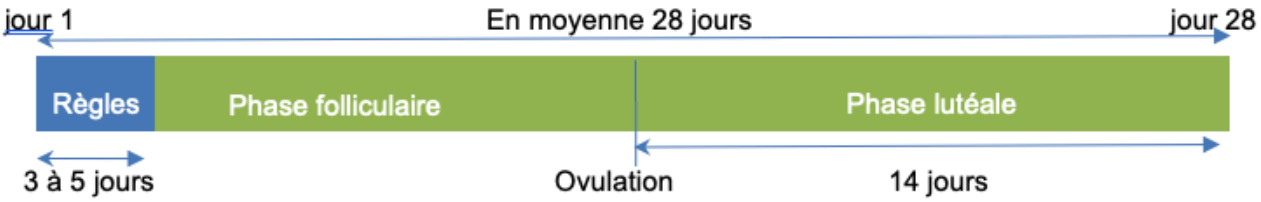
Sujet n°3

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Le cycle menstruel désigne l'ensemble des transformations physiologiques cycliques qui se répètent en moyenne tous les 28 jours chez une femme, de la puberté à la ménopause. Il est constitué d'une succession de deux phases séparées par l'ovulation. Avant l'ovulation, on définit la phase folliculaire, après l'ovulation, on définit la phase lutéale. Par convention, le cycle menstruel commence le premier jour des règles. L'ovulation se réalise toujours 14 jours avant le début des menstruations. Le schéma ci-dessous résume ce cycle.



Ce sujet propose ainsi de concevoir une application permettant de calculer et de prédire les différentes étapes du cycle menstruel à partir des dates fournies préalablement. On suppose un cycle *régulier de 28 jours* et on distingue les quatre phases suivantes, d'une façon simplifiée :

Phases d'un cycle menstruel

Phase	Numéro	Jours du cycle	Description
Règles	1	1 à 5	Écoulement menstruel
Phase folliculaire	2	6 à 13	Développement d'un follicule dans l'ovaire
Ovulation	3	14	Libération de l'ovocyte 1
Phase lutérale	4	15 à 28	Développement de la muqueuse utérine

Dans ce sujet, une date se représente par le tuple d'entiers `(jour, mois, annee)`, et l'on suppose toujours qu'elle est correctement formée et correspond à une date valide du calendrier grégorien. Par exemple, la date 7 septembre 2025 peut être représentée par le tuple `(7, 9, 2025)`.

Une année est dite bissextile lorsqu'elle comporte 366 jours au lieu de 365. Une année est bissextile si elle est divisible par 4 à l'exception des années divisibles par 100 sauf si ce sont des multiples de 400.

Par exemple :

- 2024 est une année bissextile car elle est divisible par 4 et pas par 100 ;
- 2100 n'est pas une année bissextile car elle est divisible par 4 et par 100 mais pas par 400 ;
- 2000 est une année bissextile car elle est divisible par 4, par 100 et par 400.

Question 1

Écrire en Python une fonction nommée `est_bissextile` qui prend en paramètre un entier correspondant à une année et qui renvoie un booléen indiquant si elle est bissextile, en appliquant la règle donnée ci-dessus.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire en Python une fonction nommée `determiner_phase` qui prend en paramètre un entier(compris entre 1 et 28 inclus) qui correspond au jour d'un cycle et qui renvoie un entier correspondant au numéro de la phase associée. À l'aide d'une assertion, on garantira que l'entier donné en argument est compris entre 1 et 28 inclus.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

La fonction ajouter_jours, dont le code est déjà fourni, prend en paramètres une date et un entier représentant un nombre de jours. Elle renvoie la nouvelle date obtenue après ajout de ces jours.
Compléter la fonction test_ajouter_jours en ajoutant au moins trois autres tests pertinents. Pour chaque test ajouté, une brève justification doit être donnée afin d'expliquer pourquoi ce cas est important à vérifier.
Dans le but d'ajouter l'ensemble des dates de début de règles sur un agenda en ligne pour une année donnée, on souhaite créer un fichier au format iCalendar. La structure d'un tel fichier pour un calendrier est la suivante :

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID: *le nom du calendrier*
*une suite d'événements*
END :VCALENDAR
```

Chaque événement d'une journée est lui-même décrit par une entrée de la forme suivante :

```
BEGIN :VEVENT
DTSTART: la date JJ/MM/AAAA écrite sous la forme AAAAMMJJ
SUMMARY : la description de l'événement
END :VEVENT
```

On complète les nombres strictement inférieurs à 10 par un 0 pour s'assurer que la valeur de DTSTART soit toujours de longueur 8. Ainsi, la date du 3 juillet 2026 s'écrit sous la forme DTSTART : 20260703.

Question 4

La fonction `calendrier_``cycles``,`présente dans le fichier fourni, prend en paramètre une date correspondant au premier jour des dernières règles et renvoie la liste chronologique des dates de début de règles qui se présentent dans les 100 jours suivant cette date, date incluse, au format iCalendar sous la forme d'une chaîne de caractères.
Observer avec la fonction test_calendrier_cycles que le calendrier renvoyé par la fonction calendrier_cycles n'est pas dans un format valide.
Identifier le problème dans la fonction calendrier_cycles, proposer une démarche de résolution et la mettre en œuvre.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `cycle_menstruel.py`.

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `ics`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°4

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Une équipe d'agronomes souhaite étudier la croissance de différentes plantes cultivées en serre afin de déterminer l'impact des conditions environnementales sur leur développement. Chaque jour, les chercheurs mesurent la hauteur de chaque plante, la température ambiante et l'humidité de la serre.

L'objectif de ce sujet est de concevoir un programme permettant d'analyser ces données afin de dégager des tendances sur la croissance des plantes.

Deux fichiers Python sont fournis :

- `plantes.py` qui contient la description des différentes plantes cultivées ;
- `mesures.py` qui contient les relevés journaliers effectués dans la serre.

Le fichier `plantes.py` présente les plantes étudiées sous la forme de liste d'objet plantes. La chaque objet de type plantes est une instance de la classe `Plante`.

Chaque instance de la classe plante contient les attributs suivant :

- son `nom` ;
- son `espece` ;
- sa durée moyenne de `croissance` (en jours) ;
- sa `taille` moyenne (en cm) ;
- son type d'`exposition` à la lumière parmi `"ombre"`, `"mi-ombre"` ou `"plein soleil"`.

Le fichier `mesures.py` contient les données collectées pendant plusieurs jours sous la forme d'une liste de dictionnaires. Chaque dictionnaire correspond à une mesure quotidienne. Chaque mesure comporte les champs suivants :

- `plante` nom d'une plante figurant dans plantes.py ;
- `jour` entier représentant le numéro du jour de culture ;
- `hauteur` entier correspondant à la hauteur de la plante en centimètres ;
- `temperature` entier représentant la température moyenne du jour (en °C) ;
- `humidite` entier représentant le taux d'humidité (en %) mesuré dans la serre.

Question 1

Écrire une fonction croissance_moyenne(plantes) qui prend en paramètre une liste d'instances de la classe Plante et renvoie la moyenne des durées de croissance de l'ensemble de ces plantes (en jours). Si la liste fournie est vide, la fonction doit renvoyer None.
Écrire au moins deux tests pour valider le bon fonctionnement de cette fonction, dont une traitant le cas d'une liste vide.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire une fonction dictionnaire_mesure(plantes, mesures) qui prend en paramètre la liste des plantes et la liste des mesures. Elle doit renvoyer un dictionnaire où chaque clé est le nom d'une plante (présente dans la liste plantes), et chaque valeur associée est la liste des mesures concernant cette plante spécifique.
Si une plante de la liste ne possède aucune mesure associée, la liste correspondante dans le dictionnaire devra être vide. Concevoir une série de tests pertinente pour vérifier le bon comportement de cette fonction.
Exemple de retour :

```
{
    "plante1": [
        {
            "plante": "plante1",
            "jour": 1,
            "hauteur": 100,
            "temperature": 20,
            "humidite": 100,
        },
        {
            "plante": "plante1",
            "jour": 2,
            "hauteur": 101,
            "temperature": 20,
            "humidite": 100,
        },
    ],
    "plante2": [
        {
            "plante": "plante2",
            "jour": 1,
            "hauteur": 10,
            "temperature": 20,
            "humidite": 20,
        }
    ],
}
```

Afin de ne conserver que les données correspondant à des conditions climatiques idéales, un chercheur a rédigé la fonction `purger_mesures_extremes(liste_mesures)`. Cette fonction, fournie dans le fichier `culture.py`, est censée retirer de la liste toutes les mesures ayant été

prises à une température strictement inférieure à 20°C ou strictement supérieure à 25°C.

Cependant, lorsqu'on exécute cette fonction sur les données, on constate qu'un certain nombre de relevés pris à des températures extrêmes sont toujours présents dans la liste finale.

Question 3

Exécuter le test de la fonction `test_purger` et analyser son code pour identifier la source de cette erreur logique.

Question 4

Proposer une version corrigée de la fonction répondant parfaitement à l'objectif.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier :

Le dossier fourni au candidat comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un fichier `culture.py` à compléter ;
- un jeu de données de plantes `plantes.py` ;
- un jeu de données de mesures `mesures.py`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°5

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

On s'intéresse ici à la notion d'**empreinte carbone**, qui correspond à la production de CO2 (un gaz à effet de serre) imputable à un individu ou à un groupe pendant une année, exprimé en kilogrammes. Dans le cadre de la lutte contre le réchauffement climatique, le site `nosgestesclimat.fr` propose un simulateur permettant d'obtenir une estimation de son empreinte carbone en répondant à des questions sur sa situation et sa consommation. Les résultats sont compilés dans un dictionnaire qui associe à des catégories (logement, alimentation etc.) l'empreinte carbone associée.

Une utilisatrice prénommée Ada a réalisé une simulation de son empreinte carbone, dont les résultats vous sont donnés dans les fichiers `empreinte_ada.json` sous format brute et `empreinte_ada_agr.json` sous forme agrégée. Le JSON est un format de fichier permettant de stocker des listes ou des dictionnaires dont les clés soient des chaînes de caractères. Le module `json` permet de convertir des valeurs Python en `json` et réciproquement.

Dans un premier temps, on considère les résultats sous forme agrégée représentés par un simple dictionnaire associant à chaque catégorie un entier représentant l'empreinte carbone correspondante (en kilogrammes de CO2).

```
{
   "Logement": 2660,
   "Alimentation": 1500,
   ...
}
```

Question 1

Compléter le corps de la fonction total_simple. Ajouter un test permettant d'afficher l'empreinte carbone totale d'Ada, en utilisant les fonctions fournies pour accéder au fichier `empreinte_ada_aggr.json`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Le fichier réel obtenu par Ada est `empreinte_ada.json` qui contient une imbrication de dictionnaires qui permet de préciser les postes de consommation. Par exemple :

```
{
    "Logement": {
        "Energie": {
            "Electricité": 206,
            "Cuisson": 105,
```

```
            "Chauffage individuel": 1500
        },
        "Construction": 650,
        "Location": 37,
        "Ameublement": 162
    },
    ...
}
```

Question 2

En utilisant la fonction `est_dictionnaire` qui teste la nature d'une valeur, écrire le corps de la fonction récursive `total_rec` qui calcule la somme des valeurs numériques présentes dans des dictionnaires imbriqués. Des tests sont fournis dans la fonction `test_total_rec`, on pourra les compléter par un cas plus proche de la structure présente dans `empreinte_ada.json`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Ada souhaite intégrer une fonctionnalité d'alerte. Il s'agit d'identifier si une source d'émission spécifique dépasse un certain seuil jugé critique.

Une fonction nommée `alerte_valeur_aberrante(empreinte, limite)` a été rédigée à cet effet et figure dans le fichier Python fourni. Elle est censée parcourir l'ensemble du dictionnaire, y compris les sous-catégories, et renvoyer la valeur booléenne `True` dès qu'une valeur strictement supérieure à la limite est rencontrée.

Question 3

Lorsqu'on exécute la fonction `alerte_valeur_aberrante` sur le dictionnaire complet d'Ada avec une limite fixée à 1000, elle ne détecte aucune valeur aberrante, alors que le poste "Chauffage individuel" s'élève à 1500.
Expliquer précisément l'origine de cette erreur de conception et proposer une version corrigée de la fonction.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

Afin de prévenir toute régression future sur la fonction `alerte_valeur_aberrante` corrigée, il convient de définir une stratégie de validation robuste.
Proposer un jeu de tests pertinent pour cette fonction. Pour chaque cas de test envisagé, préciser la structure du dictionnaire fourni en entrée, le résultat attendu et la particularité algorithmique que ce test permet de vérifier.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `empreinte.py` ;
- les deux fichiers `empreinte_ada.json` et `empreinte_ada_agr.json`.

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `json`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°6

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Dans le but de proposer une alternative aux sodas et d'encourager la consommation de fruits, la maison des lycéens (MDL) d'un lycée souhaite proposer des recettes de smoothies. Les fruits ne sont pas toujours tous disponibles et, par conséquent, les recettes ne sont pas toujours toutes possibles. Pour proposer une carte dynamique, l'équipe de la MDL fait appel aux élèves de NSI pour proposer une solution sous la forme d'un programme Python.

Chaque recette est composée de trois fruits comme précisé dans le tableau suivant :

Recette	Fruits
Tropical	Mangue, Ananas, Banane
Rouge	Fraise, Framboise, Cerise
Vert	Kiwi, Pomme verte, Menthe
Agrume	Orange, Citron, Pamplemousse
Exotique	Papaye, Fruit de la passion, Noix de coco
Tropical citron	Mangue, Ananas, Citron
Rouge kiwi	Fraise, Framboise, Kiwi
Exotique rouge	Papaye, Fraise, Fruit de la passion
Vert citron	Kiwi, Pomme verte, Citron
Soleil couchant	Mangue, Fraise, Pamplemousse

La classe `Boutique_smoothie` permet de créer des boutiques avec la liste des fruits du jour disponibles. Elle permet d'afficher les recettes possibles et les alternatives aux recettes qui ne peuvent pas être réalisées.

Une alternative est un smoothie réalisable qui partage un maximum de fruits communs avec le smoothie d'origine.

Par exemple, la recette *Tropical Citron* est une alternative à la recette *Tropical* s'il n'y a pas de banane parmi les fruits disponibles.

Première partie

On souhaite connaître les smoothies réalisables avec les fruits disponibles.

La liste des fruits disponibles est initialisée lors de la création de l'objet `Boutique_smoothie`.

Une recette est possible si tous ses ingrédients sont disponibles.

Exemple :

```
b = Boutique_smoothie(["Mangue", "Pamplemousse", "Ananas", "Banane"])
b.smoothie_possible("Tropical")
>>> True

b.smoothie_possible("Soleil couchant")
>>> False
```

Question 1

Écrire en Python une fonction nommée `smoothie_possible` qui prend en paramètre une chaîne de caractères représentant le nom d'un smoothie et qui renvoie un booléen qui indique s'il est possible de réaliser ce smoothie.

Question 2

Écrire en Python une fonction nommée `liste_smoothies_possibles` qui ne prend pas de paramètre et qui renvoie la liste des recettes possibles.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Seconde partie

Les alternatives permettent de proposer à un utilisateur qui souhaiterait un smoothie non disponible des smoothies de compositions proches.

Une fonction `score_proximité` permet de renvoyer le score de proximité entre deux smoothies. Ce score est calculé en comptant le nombre de fruits communs.

Question 3

Pour tester la fonction `score_proximité`, compléter la fonction `test_score_proximité` avec des tests pertinents.

La fonction `plus_proche_possible` renvoie le dictionnaire correspondant au smoothie possible le plus proche du smoothie passé en paramètre.

Question 4

Observer avec la fonction `test_plus_proche_possible` que le smoothie renvoyé n'est pas le bon.

Identifier le problème dans la fonction `plus_proche_possible`. Proposer une démarche de résolution et la mettre en œuvre.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 5

Créer une boutique avec les ingrédients suivants : Mangue, Ananas, Banane, Fraise, Citron, Kiwi et Pomme verte.
Utiliser la fonction `affichage_possibles` pour préciser quel smoothie sera proposé si on souhaite un smoothie *Agrume*.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ smoothie.py.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°7

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Les coccinelles jouent un rôle essentiel dans la régulation naturelle des pucerons, dont elles sont les principaux prédateurs. Dans une exploitation de tomates sous serre, un producteur est confronté à une prolifération de pucerons. Afin de limiter l'usage de traitements chimiques, il choisit une solution de lutte biologique consistant à introduire des coccinelles.

Il souhaite disposer d'un modèle numérique simplifié permettant d'étudier l'évolution de la population de coccinelles, leur consommation de pucerons, leur reproduction et leur mortalité.

Le fichier `coccinelles.py` contient :

- Une classe `Coccinelle` dont les objets possèdent les attributs :
  - `age` : âge de la coccinelle (en jours) ;
  - `esperance_de_vie` : durée de vie maximale (en jours) ;
  - `sexe` : sexe de la coccinelle ("male" ou "femelle") ;
  - `niv_nutrition` : nombre de jours consécutifs durant lesquels la coccinelle s'est suffisamment nourrie.
- Une fonction `evolution(population, nb_proies)` simulant l'évolution de l'écosystème sur une journée.

Rappel : Le module `random` permet de générer des valeurs aléatoires.

- `random.randint(a, b)` renvoie un entier aléatoire compris entre a et b inclus.
- `random.random()` renvoie un nombre flottant aléatoire compris entre `0` inclus et `1` exclu.

Question 1

On souhaite observer le comportement du modèle sur une courte période.
Créer une population initiale contenant 3 coccinelles (2 femelles et 1 mâle), toutes âgées de 10 jours et ayant un niveau de nutrition de 2. Le nombre initial de pucerons est fixé à 200.
Écrire une séquence d'instructions (utilisant une boucle) permettant de simuler l'évolution de ce petit écosystème sur 5 jours consécutifs, en appelant la fonction `evolution`.
Afficher le nombre de coccinelles et de pucerons à la fin de chaque journée.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire une fonction `simulation_simple(population, nb_proies)` qui automatise ce processus sur une durée maximale de 30 jours.
Cette fonction doit s'interrompre prématurément si la population de coccinelles ou de pucerons tombe à zéro. Elle doit renvoyer un triplet (tuple) contenant : le nombre final de coccinelles, le nombre final de pucerons, et le nombre de jours effectivement simulés.
Tester cette fonction avec la même population initiale que la question précédente face à 1000 pucerons.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

Écrire la documentation et les commentaires de la méthode `chasser`.

Après une première analyse des résultats, le producteur se rend compte que le modèle actuel présente des limites biologiques importantes :

1. le modèle suppose que les coccinelles peuvent se reproduire dès leur naissance, alors qu'en réalité, la reproduction n'est possible qu'à partir de 20 jours de vie ;
2. le modèle ne tient pas compte des conséquences mortelles d'un manque de nourriture. Lorsqu'une coccinelle atteint un niveau de nutrition de 0, elle a en réalité 1 chance sur 3 (environ 33 % de probabilité) de ne pas survivre à la journée.

Question 4

Modifier les méthodes `reproduction` et `a_survecu` de la classe `Coccinelle` afin d'y intégrer ces deux nouvelles règles biologiques (la maturité sexuelle et l'impact mortel du manque de nourriture).
Tester à nouveau la simulation globale pour observer les changements.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat contient :

- une version PDF de l'énoncé ;
- un fichier source `coccinelles.py` ;

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°8

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

En informatique, utiliser des nombres flottants pour manipuler des valeurs monétaires est une erreur de conception classique. Les ordinateurs utilisant le système binaire (base 2), certains nombres décimaux comme 0.1 ne peuvent pas être représentés de manière exacte et génèrent une infinité de décimales dans leur représentation binaire flottante (`0.00011001100110011...`).

Lors de calculs financiers, ces erreurs d'arrondi s'accumulent et faussent les bilans comptables.

Question 1

La chaîne de restauration "RESTO NSI" comprend 1 000 restaurants qui délivrent chacun 500 menus par jour. Un menu est composé d'une entrée à 2.27 €, d'un plat à 5.19 € et d'un dessert à 1.81 €.
Écrire, dans le fichier `addition_BCD.py`, une fonction `calcul_recettes()` qui additionne le prix de chaque menu vendu dans la journée en utilisant une boucle.
Afficher le résultat de cette fonction. Sachant que la valeur théorique exacte est de 4 635 000 €, justifier le comportement observé.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Historiquement, pour pallier ce problème dans les calculatrices et les systèmes financiers, on utilise le système BCD (Binary Coded Decimal). Le principe est de coder chaque chiffre du nombre décimal séparément sur 4 bits (un quartet).

La virgule n'étant pas codée, on utilise la convention monétaire "virgule implicite deux rangs avant la fin". Il faut donc au minimum 3 quartets pour représenter une somme.

Montant en euros	Représentation BCD (listes de chaînes)
59.00	['0101', '1001', '0000', '0000']
1.75	['0001', '0111', '0101']
0.23	['0000', '0010', '0011']

Le fichier `addition_BCD.py` contient des fonctions permettant de manipuler ces données.

Question 2

Écrire la fonction `convertir_BCD_vers_decimal(liste_quartets)` qui prend en paramètre une liste de chaînes de caractères représentant des quartets BCD, et renvoie la valeur décimale correspondante (de type float).
Ajouter une assertion pour vérifier que `convertir_BCD_vers_decimal(['0001', '0011', '0101', '0110'])` renvoie bien la valeur `13.56`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Afin de réaliser une addition de deux nombres donnés en BCD, on additionne les nombres quartet par quartet, de droite à gauche.

Si le résultat d'une addition binaire de quartets est supérieur ou égal à 10 (soit '1010' en binaire) ou s'il génère une retenue, le format BCD n'est plus valide. Il faut alors appliquer une correction en ajoutant 6 (soit '0110') à ce quartet et propager la retenue.

Question 3

La fonction `additionner_nombres_format_BCD(a, b)` fournie dans le fichier réalise cette addition. L'évaluation de l'appel `additionner_nombres_format_BCD('27', '35')` devrait correspondre à 62, mais la liste renvoyée est fausse.
Analyser le code fourni. Identifier l'oubli de l'étape de correction dans l'algorithme, puis insérer un appel à la fonction `corriger_BCD` (déjà fournie) au bon endroit pour résoudre ce problème. Refaire le test pour valider la réparation.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

Tester maintenant l'addition de 23 et de 4 avec votre code et décrire ce que vous observez.
Modifier la fonction `aligner_quartets(q1, q2)` pour qu'elle ajoute des quartets `'0000'` au début du nombre le plus court jusqu'à ce que les deux listes aient la même longueur et effectuer à nouveau des tests.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- Une version PDF de l'énoncé ;
- Un code source `addition_BCD.py`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°9

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 5 pages numérotées de 1 / 5 à 5 / 5
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Les fichiers OBJ constituent l'un des formats les plus utilisés dans le domaine de la modélisation 3D.

Ils servent à décrire une forme géométrique en listant ses sommets (les points dans l'espace) et ses faces (les surfaces reliant ces points). Grâce à ce principe, il est possible de représenter aussi bien des objets simples, comme un cube, que des modèles complexes issus de logiciels de modélisation.

Ce format est particulièrement apprécié parce qu'il est simple à lire, facile à manipuler et compatible avec la majorité des outils 3D. Cette simplicité vient du fait qu'un fichier OBJ n'est rien d'autre qu'un fichier texte, dans lequel on décrit ligne par ligne les différents éléments : le nom de l'objet (o), les sommets (v), les faces (f)…

Voici un exemple du contenu d'un fichier OBJ permettant de construire une seule face d'un cube.

```
o cube
v 0.0 0.0 0.0
v 0.0 1.0 0.0
v 1.0 1.0 0.0
v 1.0 0.0 0.0
f 1 2 3 4
```

Figure 1 – Contenu d'un fichier OBJ

Représentation de l'information

Pour manipuler ces fichiers OBJ, nous avons à disposition trois classes Python :

- La classe `Sommet` : elle représente un sommet, défini par trois entiers précisant les coordonnées x, y et z du sommet.
- La classe `Face` : elle représente une face d'un élément 3D, définie par une liste d'indices. Cet indice identifie un sommet présent dans la liste des sommets d'un `Objet3D`.
- La classe `Objet3D` : elle représente un élément 3D, défini par une liste d'objets de type `Sommet` et une liste d'objets de type Face.

Calcul du volume d'un élément 3D

On s'intéresse au calcul du volume d'un cube afin d'obtenir une estimation du temps d'impression à l'aide d'une imprimante 3D. Pour cela, il faut récupérer la longueur d'une arête du cube, à l'aide de ses différents sommets.

Pour calculer la longueur d'une arête, il suffit de calculer la distance entre deux sommets adjacents, c'est-à-dire deux sommets reliés par une arête.

Question 1

Écrire la méthode `distance` de la classe `Sommet`. Elle prend en paramètre un objet de type `Sommet`. La méthode renvoie la distance entre l'objet courant et l'objet de type `Sommet` passé en paramètre.
On rappelle que la distance entre deux points A et B dans un repère choisi s'obtient grâce à la formule

$$\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}$$

avec $(x_A; y_A; z_A)$ les coordonnées du point A et $(x_B; y_B; z_B)$ les coordonnées du point B.
Pour calculer le résultat d'une racine carrée en Python, on peut utiliser la fonction `sqrt` du module `math`.

Il faut maintenant trouver deux sommets adjacents au sein de la liste de sommets de notre objet 3D. Cette liste de sommets est présente au sein de la classe `Objet3D` grâce à l'attribut sommets. Pour savoir si deux sommets sont adjacents, on dispose d'une méthode `est_adjacent` dans la classe `Sommet`.

Question 2

Écrire la méthode `trouver_sommets_adjacents` de la classe `Objet3D`. Elle renvoie un tuple composé de deux sommets adjacents, `None` si aucun sommet n'est adjacent.
Cette méthode doit parcourir l'ensemble des couples possibles entre deux sommets à l'aide de deux boucles imbriquées et tester si les sommets du couple sont adjacents. Il faudra donc utiliser la méthode est_adjacent présente dans la classe Sommet. La méthode renvoie le premier couple de sommets adjacents trouvé.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On dispose d'une fonction volume_cube présente dans le fichier main.py. Le dictionnaire `parametres_imprimante` contient des informations concernant une imprimante 3D.

- La valeur associée à la clé remplissage précise le taux de remplissage du plastique lors de l'impression. Il est représenté sous la forme d'un entier, précisant un pourcentage.
- La valeur associée à la clef vitesse_extrusion précise la vitesse d'impression de l'imprimante. Elle est représentée sous la forme d'un entier, précisant la vitesse en mm^3/s.

Question 3

Écrire une fonction `estimation_impression`. Elle prend en paramètre le volume réel d'un élément 3D sous la forme d'un entier et un dictionnaire contenant les informations d'une imprimante 3D. La fonction renvoie le temps total d'impression sous la forme d'un flottant, exprimé en seconde.

- Dans un premier temps, il faut calculer le volume d'impression de l'objet, c'est-à-dire en multipliant le volume réel par le taux de remplissage.
- Dans un second temps, il faut calculer le temps d'impression en secondes. On rappelle que le temps s'obtient en divisant le volume d'impression de l'objet par la vitesse d'extrusion de l'imprimante.

Manipulation d'éléments 3D

On propose un programme permettant de construire un objet de type Objet3D composé d'une seule face carrée (4 sommets, 1 face).

```
from Objet3D import Objet3D

objet = Objet3D()
objet.ajouter_sommet(0, 0, 0)
objet.ajouter_sommet(0, 1, 0)
objet.ajouter_sommet(1, 1, 0)
objet.ajouter_sommet(1, 0, 0)
objet.ajouter_face([1, 2, 3, 4])

objet.afficher()
```

L'appel à la méthode `ajouter_face` permet d'ajouter une face à notre élément 3D. Elle prend en paramètre une liste d'entiers représentant l'ordre d'ajout des sommets dans notre objet 3D. Par conséquent, l'entier 1 dans la liste [1, 2, 3, 4] correspond au premier sommet ajouté dans l'objet 3D, dont la position est (0, 0, 0).

Question 4

Modifier le programme présent dans le fichier main.py afin d'instancier et afficher la pyramide présentée ci-dessous.

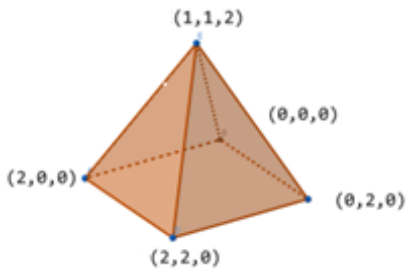


Figure 2 – Représentation d'une pyramide

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On dispose d'une méthode exporter présente dans la classe `Objet3D`. Cette méthode exporte l'ensemble des sommets et des faces de l'élément 3D selon le format d'un fichier OBJ. Cette méthode prend en paramètre une chaîne de caractères correspondant au nouveau nom du fichier exporté.

On souhaite agrandir ou rétrécir un élément 3D selon un rapport (agrandissement ou réduction), et pouvoir ensuite exporter ce nouvel objet. On dispose dans la classe Objet3D de la méthode transformer, qui permet de transformer l'instance courante selon le rapport passé en paramètre.

Question 5

Analyser le fonctionnement de la méthode transformer. Proposer une amélioration afin qu'elle ne modifie plus l'instance courante. La méthode doit désormais renvoyer un nouvel objet résultant de la transformation, sans modifier l'instance d'origine.
Tester la nouvelle méthode sur la pyramide de la question 4 avec un rapport de 2, puis l'afficher pour observer l'agrandissement.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- Plusieurs fichiers de classe Python : `Objet3D.py`, `Face.py`, `Sommet.py`
- Un fichier `main.py`

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `matplotlib`, `math`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°10

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Les compteurs d'eau modernes permettent de mesurer automatiquement la consommation d'un foyer. Ils enregistrent, chaque heure, plusieurs informations.

Les données sont fournies sous forme d'une liste de dictionnaires. On considère que cette liste est triée par jour et heure croissant. Chaque mesure est un dictionnaire contenant :

- `"jour"` : le jour de la mesure (chaîne de caractères au format "AAAA-MM-JJ") ;
- `"heure"` : l'heure de la mesure (chaîne de caractères au format "HH:MM") ;
- `"chaude"` : le volume d'eau chaude consommé en litres depuis la mesure précédente (entier) ;
- `"froide"` : le volume d'eau froide consommé en litres depuis la mesure précédente (entier).

Exemple de mesure :

```
{"jour": "2025-02-04", "heure": "08:00", "chaude": 5, "froide": 8}
```

La consommation totale d'une mesure est la somme de l'eau chaude et de l'eau froide.

L'objectif de ce sujet est d'écrire plusieurs fonctions manipulant ces données, puis d'analyser et de corriger une fonction existante qui contient une erreur.

Question 1

Écrire une fonction `total_conso` qui prend en paramètres : - `donnees` : une liste de mesures - `jour` : une chaîne représentant le jour (ex. "2025-02-04")
et renvoie la consommation totale d'eau (somme de l'eau chaude et de l'eau froide) de toutes les mesures pour ce jour.
Par convention, si aucune mesure n'existe pour ce jour, la fonction renvoie `None`. Par exemple :
"' python »> total_conso(donnees, "2025-02-04") 33 »> total_conso(donnees, "2025-12-25") »>

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

On considère que la nuit, quand tout le monde dort, la consommation d'eau n'est pas censée être supérieure à zéro pendant plusieurs heures consécutives. Une fuite est donc suspectée lorsqu'il y a au moins 3 mesures consécutives entre 00:00 et 05:00 inclus où la consommation totale est toujours non nulle (eau chaude + eau froide > 0).
Écrire une fonction `fuite_possible` qui renvoie `True` si une fuite est possible ce jour-là, `False` sinon. Cette fonction prend en paramètres : - `donnees` : une liste de mesures - `jour` : une chaîne de caractères représentant la date, au format "AAAA-MM-JJ"
Cet exemple renverrait `False`, car il n'y a pas trois mesures consécutives non nulles :

Heure	00:00	01:00	02:00	03:00
Consommation	5	3	0	3

Cet exemple renverrait `True`, car il y a trois mesures consécutives non nulles :

Heure	00:00	01:00	02:00	03:00
Consommation	2	1	1	0

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

Le fichier `analyse_eau.py` contient une fonction appelée `lissage_conso` censée calculer une moyenne sur 3 valeurs pour lisser les mesures de la consommation.
Pour les cas particuliers : - Premier élément : faire la moyenne du premier et du deuxième - Dernier élément : faire la moyenne du dernier et de l'avant-dernier - Éléments intermédiaires : faire la moyenne de trois valeurs (précédente, actuelle, suivante)
La fonction doit toujours renvoyer une liste de même taille que la liste d'origine.
Pour chaque valeur, on fait la moyenne avec ses voisins (précédent et suivant). Cependant, cette fonction contient une erreur.
Expliquer pourquoi la fonction `lissage_conso`, testée avec la liste suivante, présente un résultat incorrect, et proposer une correction.

```
test = [10, 20, 30, 40, 50]
```

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

La fonction `lissage_conso` prend en compte les cas limites dans lesquels la liste fournie contient seulement deux valeurs. Identifier un autre cas limite qui n'est pas pris en compte par la fonction, et proposer une solution pour y remédier.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- `analyse_eau.py` : fonction à corriger + squelette des autres fonctions.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°11

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Le renard est un animal qui peut habiter dans plusieurs types d'habitats pouvant être des plaines, des montagnes, des environnements ruraux, périurbains, voire urbains. La loi Biodiversité 2016 ainsi que l'arrêté du 3 aout 2023 prévoient que le renard n'est plus un animal "Nuisible" mais "Susceptible d'être nuisible".

Malgré la loi, le renard est souvent chassé des zones où il pourrait normalement évoluer et réguler la faune. Pour éviter certaines dérives, il est recommandé de surveiller la population de renards dans les zones concernées et donc prédire si un renard peut habiter cette zone.

Pour la prédiction des zones habitables d'un renard, on considère les caractéristiques suivantes :

- la végétation
- la proximité de l'eau
- la densité urbaine
- la disponibilité de proies
- la présence du renard

Ces caractéristiques sont toutes mesurées sur une échelle de 1 à 10 sauf pour la présence qui est évaluée à 1 si c'est vrai et 0 en cas d'absence du renard.

Pour évaluer la possibilité qu'un renard puisse habiter une zone prédéfinie, on utilisera la méthode des k plus proches voisins.

Le jeu de données est fourni dans le fichier donnees_habitats.py dont une partie du contenu est ci-après :

```
zones_connues = [
    {'vegetation': 9, 'proximite_eau': 6, 'densite_urbaine': 0,
    'disponibilite_proies': 4, 'presence_renard': 1},
    {'vegetation': 10, 'proximite_eau': 5, 'densite_urbaine': 9,
    'disponibilite_proies': 10, 'presence_renard': 0}
]
```

Le fichier prediction_habitat.py contient des fonctions qui seront nécessaires à l'évaluation de ces zones qui devront être complétées, modifiées ou à implémenter.

Si h_a correspond aux cinq valeurs $(v_a, p_a, de_a, di_a, pr_a)$ et h_b aux valeurs $(v_b, p_b, de_b, di_b, pr_b)$, une "distance entre h_a et h_b est donnée par la formule

$$d = \sqrt{(v_a - v_b)^2 + (p_a - p_b)^2 + (de_a - de_b)^2 + (di_a - di_b)^2 + (pr_a - pr_b)^2}$$

Question 1

Écrire le code de la fonction `distance` qui prend en paramètres deux habitats sous la forme de dictionnaires et renvoie la valeur “distance” entre ces deux habitats définie ci-dessus.
On rappelle que la racine carrée se calcule avec la fonction `sqrt` du module `math`.

Question 2

Écrire le code de la fonction `distance_d_un_habitat` qui prend en paramètres un habitat sous la forme de dictionnaire et une liste d’habitats sous la forme de liste de dictionnaires. La fonction doit renvoyer une liste de tuples où chaque tuple contient : - la distance entre l’habitat fourni et un habitat de la liste donnée; - le dictionnaire représentant l’habitat.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

Tester la fonction `distance_d_un_habitat` avec l’habitat nouveau et la liste d’habitats fournis, en affichant les 3 premiers tuples de la liste. Les résultats attendus sont indiqués ci-dessous.

```
(7.211102550927978, {'vegetation': 9, 'proximite_eau': 6,
 ↪  'densite_urbaine': 0,
  'disponibilite_proies': 4, 'presence_renard': 1})
(8.660254037844387, {'vegetation': 10, 'proximite_eau': 5,
 ↪  'densite_urbaine': 9,
  'disponibilite_proies': 10, 'presence_renard': 0})
(5.196152422706632, {'vegetation': 8, 'proximite_eau': 5,
 ↪  'densite_urbaine': 1,
  'disponibilite_proies': 6, 'presence_renard': 0})
```

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

La fonction `presence_renard` renvoie `True` s’il y a un renard qui habite dans au moins la moitié des k habitats traités, `False` sinon.

Question 4

La fonction `presence_renard` contient une erreur de traitement des tuples.
Corriger la fonction `presence_renard`.

Question 5

L’habitat `zones_connues` proposé est-il susceptible ou non de contenir une population de renards ? Expliquer en utilisant la fonction précédente avec plusieurs valeurs pour k.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- Une version PDF de l'énoncé ;
- un code source de départ `prediction_habitat.py` ;
- un jeu de données `donnees_habitats.py` qui contient toutes les zones à exploiter.

Préparation de l'environnement

Pour faire fonctionner le code fourni, les bibliothèques suivantes doivent être présentes : math et les données du fichier donnees_habitats.py doivent être importées.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°12

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Le renard, longtemps considéré comme nuisible, est aujourd'hui de plus en plus protégé pour son rôle dans la régulation de la biodiversité. Afin d'aider à la réhabilitation des individus blessés ou orphelins, un refuge de protection a été construit. La personne en charge de l'infrastructure souhaite réaliser une base de données en CSV et Python pour stocker les informations essentielles sur les renards pris en charge.

Deux entités distinctes sont représentées. La première entité est le Renard. Un renard est défini par un identifiant de type entier, un nom sous forme de chaîne de caractères, un poids en kilogrammes de type flottant, ainsi qu'une date d'arrivée représentée par une chaîne de caractères au format AAAA-MM-JJ. La seconde entité est le Refuge. Un refuge est défini par son nom, son adresse postale, et une liste regroupant les objets de type Renard qu'il héberge.

Toutes les données relatives aux animaux sont fournies dans le fichier `donnees_renards.csv`, structuré au format CSV avec le point-virgule comme séparateur.

Extrait d'informations fournies dans le fichier `donnees_renards.csv` :

id	nom	poids	date_arrivee
101	Édgar	6.5	2023-01-15
102	César	5.8	2023-02-10
103	Gérard	7.2	2023-03-05
104	Sybille	4.9	2024-11-20

Le fichier `gestion_refuge.py` comporte des éléments à compléter pour définir la classe `Renard`.

Question 1

Écrire le code du constructeur `__init__` de la classe `Renard`.

Question 2

Écrire le code de la méthode `__str__` de la classe `Renard` qui renvoie une chaîne de caractères qui présente l'animal sous le format précis : `"Renard ID [id] - [Nom]`

`(Arrivé le [date_arrivee])"`.
Tester ensuite cette classe en instanciant un renard dans une variable `renard1` ayant pour identifiant 200, se nommant Oscar, ayant un poids de 5.1 kg et étant arrivé le 1er janvier 2026. Afficher les informations de cette instance dans la console.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Le fichier `gestion_refuge.py` comporte une classe Refuge. Une méthode `importer_donnees` y est pré-écrite pour lire le fichier CSV et peupler le refuge.

Question 3

L'exécution de la méthode `importer_donnees` provoque une erreur logique lors de l'utilisation ultérieure des données, notamment lors de la manipulation du poids et de l'identifiant des renards. Identifier la source de cette erreur dans la lecture des données brutes, proposer une correction du code de la méthode, puis tester cette correction en instanciant le refuge "SOS Goupil" et en y important le fichier `donnees_renards.csv`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Le refuge utilise ces données pour surveiller la santé des animaux. Les vétérinaires considèrent qu'un renard est peu corpulent si son poids est strictement inférieur à 6.0 kg. La classe `Refuge` dispose de deux méthodes nommées `lister_peu_corpulents` et `pourcentage_peu_corpulents` pour effectuer ce suivi.

Question 4

Exécuter les deux méthodes d'analyse de la corpulence sur l'instance de votre refuge. Justifier le pourcentage obtenu en isolant et en affichant le nombre de renards peu corpulents par rapport au nombre total de renards hébergés.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `gestion_refuge.py` ;
- un fichier CSV `donnees_renards.csv` contenant toutes les données des renards.

Préparation de l’environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : math et csv.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

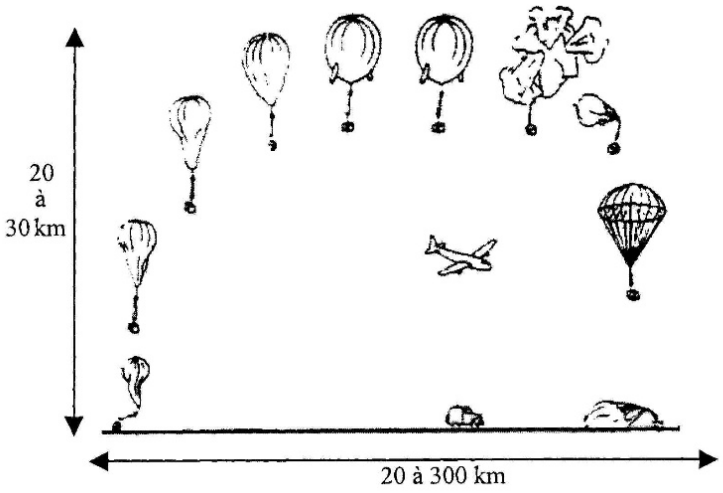
Sujet n°13

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

On considère dans ce sujet un fichier csv de données climatiques de Météo France obtenues par ballon sonde. Il est gonflé à l'aide d'un gaz léger (hélium) et est équipé d'instruments pour acquérir les paramètres tels que la température en kelvins, l'altitude en mètres et sa géolocalisation. Lorsqu'il est lâché, ce ballon sonde s'élève et collecte des données tout au long de sa trajectoire, puis éclate à une altitude appropriée ce qui entraîne sa chute au sol avec un parachute pour amortir sa vitesse de descente. Ces données relevées sont essentielles pour les prévisions météorologiques car elles fournissent des informations sur les conditions atmosphériques à différentes altitudes. Pour simplifier l'étude, on ne dispose que de données mesurées durant la montée.



Source : CNES-Planète sciences

Ce sujet propose de concevoir un script Python qui exploite les données météorologiques relevées avec :

- la conversion des valeurs de températures dans une unité plus courante ;
- une recherche de basse température et d'altitude(s) correspondante(s) ;
- la génération d'un fichier d'extension `kml` (Keyhole Markup Language) compatible avec de nombreuses applications de cartographie.

On donne une première fonction Python `recupere_donnees_fichier_csv(nom_fichier)` qui permet d'ouvrir le fichier csv, de supprimer les en-têtes et de récupérer les données relevées sous forme de 4 listes.

Question 1

En utilisant cette fonction `recupere_donnees_fichier_csv`, écrire la ou les lignes de code qui récupèrent les données relevées par le ballon sonde dans 4 listes différentes (`altitudes`, `temperatures`, `longitudes` et `latitudes`).

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

La température en kelvins correspond à la température en degrés celsius à laquelle on ajoute la valeur 273,15.

Question 2

Écrire en Python une fonction nommée `conversion_K_en_C` qui prend en paramètre une liste de températures en kelvins (K) et retourne cette même liste de températures, mais converties en degrés Celsius (°C). Les valeurs de températures doivent être arrondies à 1 chiffre après la virgule (commande utilisable : `round(valeur, nb chiffres après la virgule)`). Écrire une ligne de code permettant de tester la fonction proposée.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On souhaite disposer d'une fonction nommée `altitude_la_plus_froide` qui prend en paramètres une liste d'altitudes, ainsi qu'une liste des températures en degrés Celsius. Cette fonction doit renvoyer la température la plus froide, ainsi qu'une liste contenant la ou les altitudes correspondantes.

Par exemple, avec les deux listes `altitudes` et `temperatures` suivantes :

Exemple 1 :

```
>>> altitudes = [7000, 10125, 13896, 14211]
>>> temperatures = [-35.2, -52.1, -57.4, -57.4]
>>> altitude_la_plus_froide(altitudes,temperatures)
(-57.4, [13896, 14211])
```

Exemple 2 :

```
>>> altitudes = [6000, 7250, 11542, 15214, 17300]
>>> temperatures = [-33.7, -45, -53, -58.5, -60.1]
>>> altitude_la_plus_froide(altitudes,temperatures)
(-60.1, [17300])
```

Question 3

Proposer une écriture de la fonction `altitude_la_plus_froide`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On donne une deuxième fonction Python `genere_kml` qui prend en paramètres 2 listes (`liste_longitudes` et `liste_latitudes`), qui sont les données relevées de géolocalisation

du ballon sonde, et les utilise pour générer un fichier d'extension kml. Ce format normalisé kml utilise des balises ouvrantes et fermantes.

Question 4

Observer le corps de la fonction `genere_kml`, puis, à l'aide d'une assertion, insérer une ligne de code qui garantit que les deux listes `liste_longitudes` et `liste_latitudes` sont de même longueur.

Question 5

Écrire une ligne de code qui appelle la fonction `genere_kml` avec les deux listes de longitudes et latitudes obtenues à la question 1, puis ouvrir le fichier kml généré dans le même répertoire que le fichier Python.

Après analyse, il s'avère que ce fichier kml généré pose des problèmes de compatibilité avec certains logiciels de cartographie. En effet, la balise kml n'a pas été fermée en toute fin de fichier (`</kml>` absente).

Question 6

Proposer une amélioration du code visant à répondre à cette difficulté.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `etude_climatique.py` ;
- Un fichier des données météorologiques au format csv nommé `releves_ballon_sonde.csv`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°14

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Lors de la construction d'un bâtiment, d'un lieu culturel ou sportif, le respect des normes de sécurité amène à se poser la question du nombre judicieux de sorties, de leurs emplacements et du temps nécessaire pour l'évacuation totale des occupants.

Ce sujet propose de finaliser une application permettant de simuler l'évacuation d'une pièce rectangulaire. Cette pièce sera une instance de la classe `Piece` dont le code est dans le fichier `simulation_evacuation.py` du dossier fourni. Le constructeur de cette classe permet de définir la profondeur et la largeur de la pièce.

La méthode `ajouter_occupants(i, j, nb)` permet d'ajouter jusqu'à nb occupants dans la case située ligne i et colonne j, sachant que le nombre d'occupants d'une case est obligatoirement compris entre 0 et 5.

La méthode `ajouter_sortie(direction, position)` permet d'ajouter une sortie à la pièce bien que, pour l'instant, seules les directions "N" (pour le nord) et "O" (pour l'ouest) soient prises en compte. Lors de l'affichage d'une pièce, les sorties sont représentées par la lettre P.

Voici un exemple d'utilisation de cette classe. Le programme

```
p1 = Piece(5, 7)
p1.ajouter_occupants(2, 0, 4)
p1.ajouter_occupants(3, 4, 1)
p1.ajouter_occupants(0, 5, 2)
p1.ajouter_sortie("N", 5)
print(p1)
```

produit l'affichage console

```
             P
[0, 0, 0, 0, 2, 0]
[0, 0, 0, 0, 0, 0]
[4, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0]
[0, 0, 0, 0, 0, 0]
```

La méthode `alerter` permet de simuler une alerte : chaque occupant essaie de se rapprocher d'une sortie en se déplaçant d'une case (vers le nord, le sud, l'est ou l'ouest) ; chaque sortie ne laisse passer qu'une seule personne par alerte.

Voici, par exemple, trois alertes successives sur la pièce précédente. Il n'est pas nécessaire de comprendre, ni de modifier, le code de cette méthode.

```
                P                           P                           P
[0, 0, 0, 0, 0, 1, 0]   [0, 0, 0, 0, 0, 0, 0]   [0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0]   [0, 4, 0, 0, 0, 0, 0]   [0, 0, 4, 0, 0, 1, 0]
[0, 4, 0, 0, 0, 1, 0]   [0, 0, 0, 0, 0, 1, 0]   [0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0]   [0, 0, 0, 0, 0, 0, 0]   [0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0]   [0, 0, 0, 0, 0, 0, 0]   [0, 0, 0, 0, 0, 0, 0]
```

Question 1

Écrire le corps de la méthode `nb_occupants_restants` de la classe `Piece`. Comme son nom l'indique, cette méthode doit renvoyer le nombre d'occupants restants dans la pièce.
La fonction `test_nb_occupants_restants` présente dans le fichier `simulation_evacuation.py` vous permettra d'effectuer une première série de tests.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire le corps de la fonction `evacuation` afin qu'elle simule l'évacuation complète de la pièce et renvoie le nombre de tours nécessaire. On pourra, dans cette fonction, faire appel à la méthode `alerter` qui simule un tour et renvoie True si des déplacements ont pu avoir lieu, False sinon. En complément de la pièce à évacuer, la fonction `evacuation` a un paramètre silencieux dont la valeur par défaut est True. Si ce paramètre vaut False, l'état de la pièce doit être affiché à chaque tour dans la console.
La fonction `test_evacuation` présente dans le fichier `simulation_evacuation.py` vous permettra d'effectuer une série de tests.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

Modifier la méthode `ajouter_sortie(direction, position)` afin qu'il soit aussi possible d'ajouter une sortie dans les directions qui ne sont pour l'instant pas prises en compte : "S" (pour le sud) et "E" (pour l'est).
La fonction `test_ajouter_sortie` vous permettra d'effectuer une première série de tests. Vous vérifierez également qu'il est maintenant possible d'ajouter des sorties dans les quatre directions via l'interface homme machine (IHM), sans modifier le code de celle-ci. Lorsqu'une pièce a été créée dans l'IHM, un clic en périphérie de cette pièce déclenche automatiquement un appel à la méthode `ajouter_sortie` et fait apparaître la porte ajoutée. Cependant, une seule porte sera utilisée lors des alertes tant que la question suivante n'aura pas été traitée.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On s'aperçoit que, lorsqu'une pièce possède plusieurs sorties, seule la première est utilisée par les occupants. Le problème vient de la méthode `choix_sortie(i, j)` qui renvoie la sortie à

utiliser pour une personne positionnée sur la ligne i et la colonne j.

Question 4

Identifier l'erreur logique et la variable non définie dans le code de cette méthode, puis effectuer les corrections nécessaires afin qu'elle renvoie la sortie la plus proche.
La fonction `test_choix_sortie` vous permettra d'effectuer une première série de tests.
Vous poursuivrez vos tests avec l'IHM (sans la modifier).

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants : - une version PDF de l'énoncé ; - un code source à compléter et corriger `simulation_evacuation.py` ; - un programme `IHM_evacuation.py` permettant d'ouvrir une IHM qui facilitera les tests de l'élève, à utiliser sans modification.

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `random`, `copy`, `tkinter`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°15

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 6 pages numérotées de 1 / 6 à 6 / 6
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Cadre

Un cabinet vétérinaire utilise une base de données pour assurer le suivi et la gestion des clients et animaux qui y sont suivis, dans le but d'améliorer la qualité des soins et de fournir des services utiles, que ce soit pour le vétérinaire ou pour ses clients.

Objectif

La vaccination régulière des chats (une fois par an) étant importante mais souvent oubliée par les propriétaires, le vétérinaire souhaiterait pouvoir envoyer des messages de rappel de vaccination par SMS, deux mois avant l'échéance (la date à laquelle la vaccination devrait avoir lieu).

Il dispose pour cela d'une plateforme d'envoi de SMS, mais il est nécessaire de préparer les données et les messages à envoyer.

Description de la base de données

Le logiciel mis en œuvre par le cabinet utilise 3 tables pour enregistrer les animaux, leur propriétaire et les consultations.

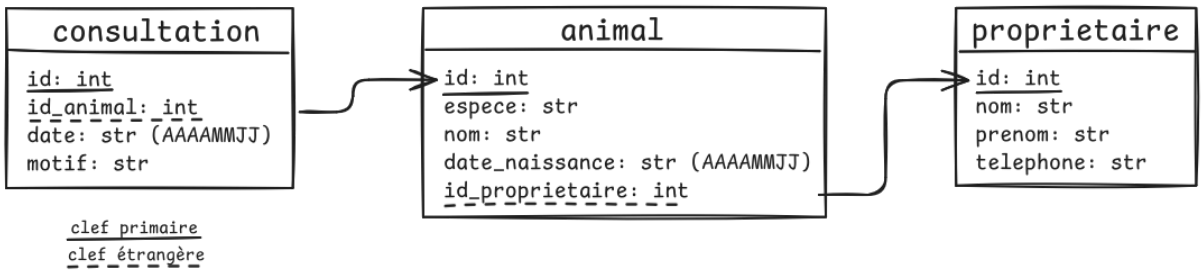


Schéma de la base de données

Extraits de la table proprietaire

id	nom	prenom	telephone
1	Duhamel	François	06-57-84-52-41
2	Dumont	Marcelle	0648747223
3	Joubert	Anouk	0 4 76 33 78 15

Extraits de la table consultation

id	id_animal	date	motif
1	1	20171229	vaccination
25	4	20050621	bilan
34	5	20131106	maladie

Extraits de la table animal

id	espece	nom	date_naissance	id_proprietaire
1	oiseau	Flocon	20171111	1
2	lapin	Roudoudou	20210901	2
3	chat	Mistral	20251206	2

Extraits de la base de données

Le système de gestion de base de données utilisé est SQLite, et il est possible d'interagir avec la base de données en utilisant le module sqlite, disponible dans la bibliothèque standard python.

Rappels techniques

La connexion à la base de données se fait en précisant l'emplacement du fichier de la base de données. Par exemple, si le fichier cabinet.sqlite se trouve dans le répertoire courant, on écrit :

```
DB_PATH = "cabinet.sqlite"
conn = sqlite3.connect(DB_PATH)
```

Une fois que l'on dispose d'une connexion (variable conn), on utilise un curseur (variable cursor) pour exécuter des requêtes sur la base de données et éventuellement récupérer les résultats de la requête.

```
cursor = conn.cursor()
resultats = cursor.execute(
    "SELECT id, nom FROM animal ORDER BY nom LIMIT 10"
)
# resultats permet de récupérer les lignes de manière
# progressive, sous forme d'une liste itérable de tuples
# correspondant aux champs demandés
# Pour accéder aux champs sélectionnés
for r in resultats:
# les valeurs renvoyées par le SELECT sont
# accessibles par leur position
    print("id:", r[0], ", nom:", r[1])
```

Lorsqu'un paramètre doit être donné dans une requête SQL, il est fourni sous forme d'un ? dans le texte de la requête SQL, et passé en argument à la fonction exécute sous forme d'un tuple (chaque ? correspond à un élément du tuple, dans l'ordre). Par exemple, pour récupérer les chats du propriétaire d'id 2 :

```
resultat = cursor.execute(
    "SELECT nom FROM animal "
    " WHERE espece = 'chat' AND id_proprietaire= ?",
    (2,)
)
```

Les clauses de langage SQL à utiliser sont : SELECT, FROM, JOIN, ON, WHERE, ORDER BY

Numéros de téléphone

La plateforme d'envoi de SMS requiert un numéro de téléphone au format 06XXXXXXXX ou 07XXXXXXXX, 10 chiffres au total, sans espace, ni point, ni autre caractère. Malheureusement, les téléphones ont été enregistrés dans la base sans distinguer les téléphones mobiles des téléphones fixes, et en utilisant différents styles comme "(0)6 12 34 56 78" ou "01.23.45.67.89".

Rappel : on peut utiliser la méthode `isdigit` de la classe `str` pour déterminer si un caractère est un chiffre

```
>>> '2'.isdigit()
True
>>> 'C'.isdigit()
False
```

Question 1

Écrire une fonction python `normalisation_tel` qui prend un numéro de téléphone tel de type str en argument et qui renvoie le numéro de téléphone nettoyé en ne gardant que les chiffres. Exemples :

```
normalisation_tel("(0)2 12 99 90 12") # renvoie "0212999012"
normalisation_tel("0.6.12.99.90.12") # renvoie "0612999012"
normalisation_tel("0-6-12-9-90-1") # renvoie "06129901"
```

Une fonction `test_normalisation_tel` est fournie.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Validation des numéros de téléphone

La fonction `validation_tel(tel)` prend comme argument un numéro de téléphone nettoyé, et renvoie un booléen selon que ce numéro est un numéro de téléphone portable valide ou non.

Question 2

Écrire un jeu de tests permettant de vérifier le bon fonctionnement de la fonction.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Détermination de la liste des chats à vacciner

La vaccination se faisant tous les ans, le vétérinaire souhaite envoyer trois rappels aux propriétaires. Chaque rappel sera donc envoyé aux propriétaires des chats ayant été vaccinés pour la **dernière** fois dans l'intervalle J - 13 mois à J - 10 mois.

Pour cela on commencera par déterminer quels chats ont été vaccinés il y a **moins de 13 mois**, à l'aide de la fonction `consultation_vaccination_chat(date)`.

Puis à l'aide de la fonction `derniere_vaccination(consultations)`, on déterminera la date de **dernière** vaccination de ces chats.

Enfin, dans la dernière étape du projet, se fera le filtrage des chats ayant été vaccinés pour la **dernière** fois il y a plus de 10 mois.

Interrogation de la base de données

Pour interroger la base de données, on utilise la fonction `consultation_vaccination_chat(date)` qui doit récupérer toutes les consultations de motif vaccination des chats (espece : chat) du cabinet, pour des dates supérieures à une date, date, exprimée au format AAAAMMJJ (ce format permet de comparer des dates en utilisant l'ordre alphabétique).

On souhaite récupérer les champs suivants : - id de l'animal ; - nom de l'animal ; - telephone du propriétaire ; - date de la consultation.

triés par le champ id d'animal puis date de consultation croissants.

Question 3

En vous inspirant de la fonction `proprietaires_animaux_nes_apres(date)`, écrire la fonction la fonction `consultation_vaccination_chat(date)`.
Des tests sont fournis dans la fonction `test_consultation_vaccination_chat`, votre fonction doit les passer.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Détermination de la date de dernière vaccination

Pour déterminer la date de dernière vaccination, on utilise la fonction `derniere_vaccination` qui prend en paramètre une liste de consultations avec les champs précédents (id de l'animal, nom de l'animal, id_proprietaire de l'animal, date de la consultation). Elle renvoie un dictionnaire associant comme clef l'id de chaque chat avec les informations de sa **dernière** (plus récente) consultation.

Exemple : pour Plume et Gollum

```
(16, 'Plume', '0.6.36.96.89.83', '20241024'),
(16, 'Plume', '0.6.36.96.89.83', '20251125'),
(17, 'Gollum', '0.6.36.96.89.83', '20250113'),
```

La dernière consultation de Plume est le 20251125, et pour Gollum, c'est le 20250113 donc `derniere_vaccination` doit donc renvoyer le dictionnaire :

```
{
16: (16, 'Plume', '0.6.36.96.89.83', '20251125'),
17: (17, 'Gollum', '0.6.36.96.89.83', '20250113')
}
```

Cependant, la fonction ne produit pas les résultats attendus : les tests fournis dans la fonction `test_derniere_vaccination` échouent.

Question 4

Expliquer le problème rencontré, puis comment corriger cette fonction pour qu'elle passe les tests avec succès.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `veto.py`.
- une base de données `cabinet.sqlite`

Préparation de l’environnement

Environnement de gestion des bases de données : SQLite

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `sqlite3`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°16

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Conçues par le climatologue Ed Hawkins en 2018, les *warming stripes* ("bandes de réchauffement") permettent de représenter visuellement l'évolution des températures au cours du temps par rapport à une moyenne historique.

Le fichier `datas.csv` est issu du site de l'agence américaine d'observation océanique et atmosphérique (NOAA). Il contient les écarts de températures mondiales par rapport à la moyenne du 20ème siècle (1901-2000), pour chaque année de 1851 à 2025.

Par exemple, la ligne `2020,1.01` signifie qu'en 2020, la température moyenne mondiale (terres et océans) était supérieure de 1,01 °C par rapport à la moyenne de référence.

L'image `warming_stripes.png` représente les *warming stripes* obtenues à partir des données présentes dans le fichier `data.csv`. Les couleurs varient du bleu au rouge, les tons de bleu indiquant des températures inférieures à la moyenne de référence et les tons de rouge indiquant des températures supérieures à cette même moyenne.

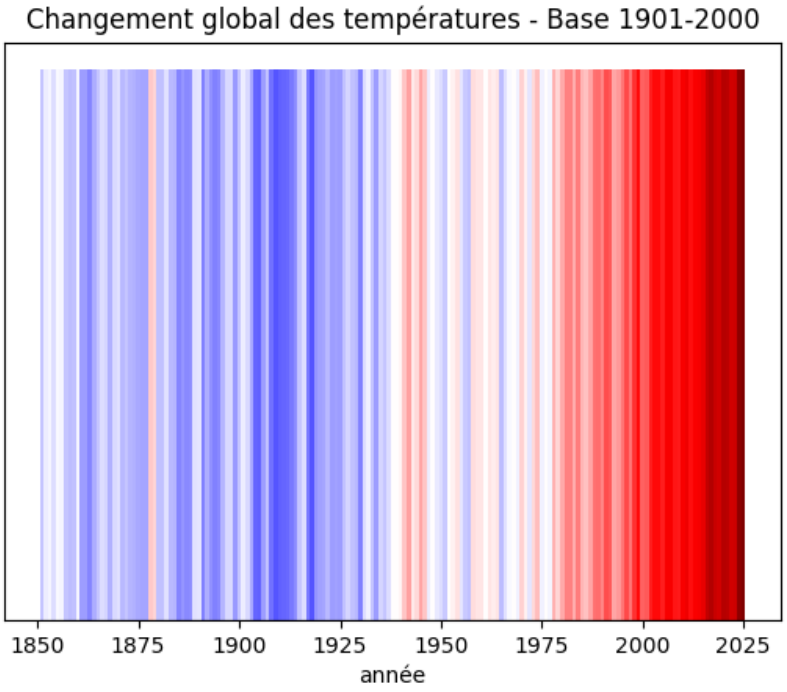


Figure 1. Les *warming stripes* obtenues à partir du fichier data.csv.

La fonction `charger(fichier)`, dont le code est fourni dans le script `warming_stripes.py`, lit ce fichier et renvoie une liste de dictionnaires. Chaque dictionnaire représente une année avec les clés `'année'` (type entier) et `'écart'` (type flottant).

Question 1

Écrire la fonction `ecart_temperature(datas, annee)` qui prend en paramètres la liste des données et une année cible. Elle doit renvoyer l'écart de température correspondant à cette année, ou `None` si l'année n'est pas présente dans les données.
Rédiger ensuite un jeu de tests (via des assertions) permettant de vérifier le bon fonctionnement de votre fonction, y compris pour une année hors du jeu de données.

Question 2

En utilisant la fonction `derniere_annee_ecart_negatif` (déjà fournie et qui utilise votre fonction), déterminer quelle a été la dernière année où la température mondiale était inférieure à la moyenne de référence.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

La fonction `prevision(datas, annee_cible, n)`, dont le code est fourni, effectue une régression linéaire sur les n dernières années pour estimer l'écart de température d'une année future. C'est-à-dire que cette fonction calcule une droite proche des points des n dernières années et renvoie l'ordonnée du point sur cette droite dont l'abscisse est l'année cible.

Question 3

En exécutant `prevision(datas, 2040, 20)`, le résultat obtenu semble absurde compte tenu du réchauffement actuel.
Cette absurdité provient d'une erreur de logique qui s'est glissée dans la fonction annexe `moyenne_ecarts`. Analyser le code de cette fonction, identifier l'erreur, et la corriger. Que devient alors la prévision pour 2040 ?

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

On souhaite générer la représentation graphique des *warming stripes*. Dans la fonction `graphique(datas)`, le code pour gérer les couleurs (du bleu pour le froid au rouge pour le chaud) est déjà préparé.
Compléter le code de la fonction en générant les listes `annees` (pour l'axe des abscisses) et `ordonnees`.
Indication : Pour obtenir des bandes colorées de hauteur uniforme couvrant tout le graphique, la liste `ordonnees` devra contenir la valeur `1` pour chaque année traitée.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `warming_stripes.py` ;
- un fichier `datas.csv` contenant les écarts de températures constatés entre 1851 et 2025 par rapport à la moyenne 1901-2000 ;
- une image `warming_stripes.png` représentant les *warming stripes* obtenues à partir du fichier `datas.csv`.

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `matplotlib`, `csv`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°17

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Un club de handball professionnel souhaite mieux comprendre et gérer ses dépenses et recettes annuelles. Les dirigeants veulent identifier les postes de dépenses les plus importants et s'assurer de l'exactitude de leur solde budgétaire de fin d'année.

On dispose de jeux de données sous forme de fichiers CSV représentant les mouvements budgétaires du club. Chaque mouvement est décrit par un enregistrement comportant les attributs suivants :

- 'type' : chaîne de caractères dont les valeurs possibles sont 'dépense' ou 'recette' ;
- 'catégorie' : chaîne de caractères dont les valeurs possibles sont 'subventions', 'sponsoring', 'billetterie', 'marketing', 'salaires', 'déplacements', 'fonctionnement', 'cotisations' ou 'autres_charges' ;
- 'montant' : nombre flottant positif représentant la somme en euros ;
- 'mois' : nombre entier compris entre 1 et 12.

Une fois chargé par le script `analyse_budget.py`, ces jeux de données donnent une liste de dictionnaires mouvements.

Deux jeux de données sont fournis :

- un jeu de test défini directement dans le code dans la variable `mouvements_test` donne la liste de dictionnaires reproduite ci-dessous :

```
mouvements_test = [
    {'type': 'recette', 'catégorie': 'cotisations', 'montant': 1200.0,
 ↪  'mois': 1},
    {'type': 'recette', 'catégorie': 'billetterie', 'montant': 300.0,
 ↪  'mois': 6},
    {'type': 'dépense', 'catégorie': 'fonctionnement', 'montant': 450.0,
 ↪  'mois': 6},
    {'type': 'dépense', 'catégorie': 'déplacements', 'montant': 200.0,
 ↪  'mois': 12},
    {'type': 'dépense', 'catégorie': 'salaires', 'montant': 1500.0,
 ↪  'mois': 12},
    {'type': 'recette', 'catégorie': 'subventions', 'montant': 800.0,
 ↪  'mois': 12}
]
```

- un jeu complet `budget_complet.csv` de plus de 2000 mouvements.

Le fichier `analyse_budget.py` contient plusieurs fonctions d'analyse que le sujet vise à compléter.

Question 1

Écrire la fonction `total_par_type(mouvements, type_mouvement)` qui prend en paramètres une liste de mouvements et une chaîne de caractères (parmi 'dépense' ou 'recette'). Cette fonction doit renvoyer la somme totale des montants correspondant à ce type précis. La fonction renverra `0` si aucun mouvement ne correspond.
Créer ensuite une fonction test_total() contenant au moins deux assertions pour valider le bon fonctionnement de votre code sur le jeu de données mouvements_test (par exemple, le total des recettes attendu est de 2300.0).

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

La fonction `solde_annuel(mouvements)`, déjà codée dans le fichier fourni, est censée calculer le bilan financier de l'année en faisant la somme des soldes de chaque mois (recettes totales moins dépenses totales). Si le solde annuel est positif, le club dégage des bénéfices. S'il est négatif, le club est en déficit

Question 2

Calculer manuellement le solde annuel attendu pour la liste mouvements_test. Écrire ensuite une fonction `test_solde_annuel()` contenant une assertion qui vérifie que la fonction `solde_annuel(mouvements_test)` renvoie bien ce résultat théorique. Exécuter ce test.

L'exécution du test précédent lève une erreur : le résultat renvoyé par la fonction solde_annuel ne correspond pas à la réalité comptable.

Question 3

Analyser le code de la fonction solde_annuel pour identifier la source de cette erreur logique, expliquer pourquoi certains mouvements ont été ignorés, puis proposer une correction.
Appliquer ensuite votre fonction corrigée sur le fichier `budget_complet.csv` pour annoncer le véritable solde du club au professeur.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier fourni

Le dossier mis à disposition du candidat contient :

- une version PDF de cet énoncé ;
- un code source à analyser et compléter : `analyse_budget.py` ;
- un jeu de données complet plus de 2000 mouvements : `budget_complet.csv`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°18

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

En Polynésie française, un réseau de bouées océanographiques mesure quotidiennement la température de surface de l'océan dans quatre archipels : **Société, Tuamotu, Marquises et Australes**. Ces données permettent de documenter le réchauffement climatique dans le Pacifique Sud.

Vous êtes chargé(e) de développer des outils d'analyse pour traiter ces relevés et détecter les anomalies thermiques.

On vous fournit le fichier `analyse_temperatures_polynesie.py` dans lequel figure les données à utiliser, ainsi que les fonctions à écrire, compléter ou corriger.

Question 1

Écrire une fonction `temperature_moyenne(zone, donnees)` qui :

- Prend en paramètres une chaîne `zone` (nom d'un archipel) et un tableau `donnees` de dictionnaires représentant les relevés
- Renvoie la température moyenne (float) pour cette zone
- Renvoie `None` si la zone n'a aucun relevé

Exemple :

```
>>> donnees = [
    {'date': '2020-01-15', 'zone': 'Societe', 'temperature': 28.5},
    {'date': '2020-01-16', 'zone': 'Societe', 'temperature': 29.0},
    {'date': '2020-01-15', 'zone': 'Tuamotu', 'temperature': 27.5}
]
>>> temperature_moyenne('Societe', donnees)
28.75
>>> temperature_moyenne('Marquises', donnees)
>>>
```

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire une fonction `detecter_anomalies(zone, seuil, donnees)` qui :

- Prend en paramètres une `zone`, un `seuil` (float), et la liste `donnees`
- Calcule la température moyenne de la zone

- Renvoie la liste des dates où la température s'écarte de plus de `seuil` degrés (en valeur absolue) de cette moyenne
- Renvoie une liste vide si la zone n'existe pas

Exemple avec les données de l'exemple précédent :

```
>>> detecter_anomalies('Societe', 3.0, donnees)
['2020-02-10', '2020-03-05']
```

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Dans le fichier `analyse_temperatures_polynesie.py` figure le code de la fonction `evolution_par_decennie(zone, donnees)` censée calculer l'évolution des températures moyennes par décennie (2010, 2020).

Question 3

Compléter les trois fonctions de test, situées à la fin du fichier, pour la fonction `evolution_par_decennie` en utilisant le jeu de données fourni.

1. **test_zone_inexistante()** : Tester une zone qui n'existe pas
2. **test_une_seule_decennie()** : Tester une zone avec données sur une seule décennie
3. **test_plusieurs_decennies()** : Tester une zone avec données sur plusieurs décennies

Vos tests doivent permettre d'identifier le bug présent dans le code.

Question 4

Après avoir identifié le bug grâce à vos tests, corriger la fonction `evolution_par_decennie`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé
- un fichier python nommé `analyse_temperatures_polynesie.py` contenant tous les éléments nécessaires et qu'il s'agira de compléter.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°19

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Dans une commune d'une île polynésienne, les services communaux doivent surveiller et gérer les réserves d'eau de plusieurs réservoirs alimentant différents districts. Les apports d'eau dans les réservoirs proviennent uniquement de précipitations ou de captages naturels. La commune dispose d'un système de redistribution de l'eau entre réservoirs par réseaux de canalisations ou par camions citernes sinon.

L'objectif est d'aider à :

- analyser les volumes d'eau disponibles,
- prévoir les besoins,
- proposer un programme de gestion existant.

Vous êtes chargé(e) de participer à ce développement. Ce suivi permet d'anticiper des risques de sécheresse, de planifier la redistribution en cas de restriction d'eau.

On représente un réservoir d'eau par un dictionnaire avec les champs suivants et le type des valeurs associées :

- `"nom"` (`str` indiquant le nom du réservoir) ;
- `"capacite"` (`int` indiquant la capacité maximale, en litres, d'eau que le réservoir peut contenir) ;
- `"volume"` (`int` indiquant le volume d'eau en litres actuellement disponible dans le réservoir) ;
- `"district"` (`str` indiquant le lieu desservi par le réservoir).

Plusieurs réservoirs peuvent alimenter un même district.

Le fichier `donnees.py`, dont le contenu est partiellement reproduit ci-dessous, contient les informations sur les réservoirs de la commune.

```
reservoirs = [
    {"nom": "Nuiavai", "capacite": 100000,
     "volume": 55000, "district": "Tepua"},
    {"nom": "Farepape", "capacite": 120000,
     "volume": 45000, "district": "Fare"},
    ...
]
```

Question 1

Un réservoir est considéré comme en pénurie si son taux de remplissage est strictement inférieur à 20 %.
Écrire en Python une fonction nommée `est_en_penurie` qui prend en paramètres une liste de réservoirs et le nom d'un réservoir. On supposera que le nom donné correspond à un réservoir présent dans les données. La fonction renverra un booléen indiquant si ce réservoir est en pénurie.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire en Python une fonction nommée `volume_par_district` qui prend en paramètres une liste de réservoirs et qui renvoie un dictionnaire ayant pour clés chaque nom différent de district associés au volume total d'eau en litres disponible dans ce district.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Le fichier `gestion_eau.py` fourni contient les fonctions : `volume_moyen`, `taux_remplissage`, `listes_des_districts` et `reservoirs_par_district`.

Question 3

La fonction volume_moyen est censée renvoyer le volume moyen d'eau disponible sur l'ensemble des réservoirs de la commune. Le code de cette fonction est incorrect.
Proposer quelques tests simples, sous forme d'assertions qui permettent de mettre ce ou ces problèmes en évidence :

- vérifier que la liste contient au moins un réservoir ;
- vérifier qu'une moyenne doit être strictement inférieure ou égale à la plus grande capacité parmi les réservoirs de la liste ;
- vérifier que la fonction renvoie le bon résultat dans un cas simple (par exemple deux réservoirs avec le même volume).

Proposer une version corrigée de la fonction volume_moyen qui valide ces tests et renvoie la valeur correcte.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

Même si le volume moyen d'eau disponible au niveau de l'ensemble des réservoirs de la commune est satisfaisant, certains districts peuvent présenter un niveau d'eau disponible

nettement insuffisant.
Proposer une adaptation du code fourni permettant d'identifier ces districts vulnérables (par exemple en considérant que leur volume moyen est inférieur à 80 % du volume moyen global, cette valeur pouvant être introduite en paramètre) puis proposer une stratégie de gestion permettant d'améliorer la situation de ces districts (sans l'implémenter).

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `gestion_eau.py` ;
- un fichier de données `donnees.py`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°20

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Le numérique représente aujourd'hui environ 4 % des émissions mondiales de gaz à effet de serre, soit davantage que le transport aérien civil. Une part importante de cette empreinte provient de la fabrication des équipements, mais l'usage quotidien (streaming vidéo, envoi d'emails, stockage cloud) y contribue également de manière significative.

On souhaite développer une application permettant d'estimer l'empreinte carbone d'un utilisateur en fonction de ses usages numériques. Les émissions sont exprimées en gramme de CO2 équivalent (gCO2e).

On dispose des données suivantes concernant les émissions moyennes de différentes activités numériques :

Activité	Émission unitaire	Unité
Email sans pièce jointe	4	gCO2e par email
Email avec pièce jointe (1 Mo)	19	gCO2e par email
Streaming vidéo SD	36	gCO2e par heure
Streaming vidéo HD	100	gCO2e par heure
Recherche web	7	gCO2e par recherche
Stockage cloud	10	gCO2e par Go par mois

Un utilisateur est représenté par un dictionnaire contenant ses usages mensuels. Par exemple :

```
utilisateur1 = {
    'emails_simples': 150,
    'emails_pj': 20,
    'streaming_sd': 10, # en heures
    'streaming_hd': 25, # en heures
    'recherches': 500,
    'stockage_cloud': 15 # en Go
}
```

Question 1

Dans le fichier code_empreinte.py, écrire en Python le code de la fonction `calculer_empreinte(utilisateur)` qui prend en paramètre un dictionnaire représentant les usages

d'un utilisateur (au format décrit précédemment) et qui renvoie l'empreinte carbone totale mensuelle en gCO2e qui est de type entier.
Par exemple, `calculer_empreinte(utilisateur1)` renvoie la valeur 7490.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire en Python le code de la fonction `classer_par_impact(utilisateur)` qui prend en paramètre un dictionnaire représentant les usages d'un utilisateur et qui renvoie un dictionnaire contenant trois clés 'fort', 'moyen' et 'faible', chacune associée à une liste des noms des activités correspondantes selon les critères suivants :

- impact 'fort' : émissions ≥ 1000 gCO2e
- impact 'moyen' : 200 gCO2e ≤ émissions < 1000 gCO2e
- impact 'faible' : émissions < 200 gCO2e

Par exemple, pour l'utilisateur `utilisateur2` ayant : - streaming_hd : 15 (1500 gCO2e) → impact 'fort' - emails_simples : 100 (400 gCO2e) → impact 'moyen' - recherches : 10 (70 gCO2e) → impact 'faible'
La fonction doit renvoyer dans ce cas :

```
{
 'fort': ['streaming_hd'],
 'moyen': ['emails_simples'],
 'faible': ['recherches']
}
```

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

La fonction `comparer(utilisateur1, utilisateur2)`, dont le code est fourni, prend en paramètres deux dictionnaires `utilisateur1` et `utilisateur2` représentant les usages de deux utilisateurs. Elle renvoie un dictionnaire associant à chaque activité la différence des émissions entre les deux utilisateurs. Cette différence est définie comme l'émission de l'`utilisateur2` moins celle de l'`utilisateur1`. Si une activité est absente chez un utilisateur, on considère que son émission vaut 0.

Question 3

Compléter le code de la fonction test_comparer en ajoutant au moins deux tests pertinents qui permettent de vérifier différents aspects du comportement de la fonction. Pour chaque test ajouté, une brève justification doit être donnée afin d'expliquer pourquoi ce cas est important à vérifier. D'autres utilisateurs peuvent être créés lors de l'écriture des tests.

Pour mieux percevoir l'écart entre les usages de deux utilisateurs, on utilise la formule ci-dessous :

$$\text{pourcentage d'écart} = \frac{\text{émission2} - \text{émission1}}{\text{émission1}} \times 100$$

La fonction `comparer_v2(utilisateur1, utilisateur2)`, dont le code est fourni, prend en paramètres deux dictionnaires représentant les usages de deux utilisateurs. Elle renvoie un dictionnaire associant à chaque activité le pourcentage d'écart expliqué ci-dessus.

Par exemple, `comparer_v2(utilisateur1, utilisateur2)` renvoie :

```
{
   'emails_pj': -100.0,
   'emails_simples': -33.33333333333333,
   'recherches': -98.0,
   'stockage_cloud': -100.0,
   'streaming_hd': -40.0,
   'streaming_sd': -100.0
}
```

Dans certains cas, on constate que cette fonction provoque une erreur.

Question 4

Identifier dans quels cas l'erreur se produit et proposer une correction.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `code_empreinte.py`

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°21

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Les cartes mémoire (ou *flashcards*) sont des supports de révision comportant au recto une question et au verso la réponse. Elles permettent des révisions actives très efficaces.

On souhaite développer une application utilisant le système des boîtes de *Leitner*, qui permet de gérer ces cartes en se basant sur un principe de répétition espacée.

Dans cette méthode, chaque carte possède un niveau d'avancement (de 0 à 4). Plus le niveau est élevé, plus le délai avant la prochaine révision est grand. Les délais sont donnés par le tableau suivant :

Niveau de la carte	Délai avant révision
0	1 jour
1	3 jours
2	7 jours
3	15 jours
4	30 jours

En pratique, la méthode fonctionne ainsi : lorsque l'utilisateur révise une carte, s'il répond correctement, la carte passe au niveau supérieur (sans dépasser le niveau 4 maximum). S'il se trompe, la carte retombe immédiatement au niveau 0, quel que soit son niveau précédent. La prochaine date de révision est ensuite calculée en ajoutant le délai du nouveau niveau à la date du jour.

On modélise ce fonctionnement à l'aide de la Programmation Orientée Objet. Dans le fichier `cartes.py`, la classe `Carte` a été commencée.

Question 1

Écrire la méthode `traiter_reponse(self, succes)` qui prend en paramètre un booléen `succes` (`True` si l'utilisateur a bien répondu, `False` sinon). Cette méthode doit mettre à jour l'attribut `self.niveau` de la carte selon les règles de Leitner énoncées accessible dans la liste globale `DELAIS`, puis calculer et mettre à jour l'attribut `self.date_prochaine`.
Indication : Pour ajouter des jours à la date d'aujourd'hui, on utilisera la fonction fournie

`date_future(nb_jours)` qui renvoie la date située `nb_jours` après aujourd'hui.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

On considère que la base de révision, le *paquet* de cartes, est une liste d'instances de la classe `Carte`.

Question 2

Écrire une fonction `extraire_cartes_du_jour(paquet, date_jour)` qui prend en paramètres une liste de cartes `paquet` et une date de référence `date_jour` et qui renvoie une nouvelle liste contenant uniquement les cartes dont la `date_prochaine` est inférieure ou égale à `date_jour`.
On admet qu'on peut comparer des dates avec les opérateurs usuels `<`, `<=`, `==`, `>=` et `>`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Afin d'aider l'étudiant à cibler ses lacunes, on souhaite extraire du paquet les cartes qui lui posent le plus de problèmes, c'est-à-dire celles dont le niveau est le plus bas parmi toutes les cartes du paquet.

La fonction `extraire_cartes_a_renforcer(paquet)` a été rédigée dans ce but. Cependant, elle contient une faille logique.

Question 3

Exécuter la fonction `test_renforcement()` fournie. Observer le résultat affiché dans la console et constater l'incohérence.
Analyser le code de la fonction `extraire_cartes_a_renforcer(paquet)`, identifier la source de cette erreur logique, puis corriger le code afin qu'il ne renvoie que les cartes possédant rigoureusement le niveau minimum.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `cartes.py`.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°22

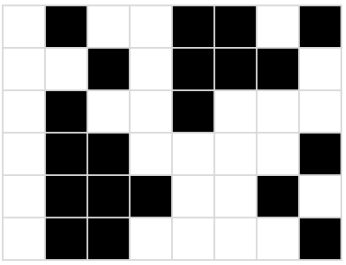
DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

Un QR code dans sa version simplifiée est une image constituée de carrés noirs disposés sur un fond blanc. Ces carrés définissent l'information que contient le code et seront convertis en une chaîne de caractères lors du déchiffrement du code par un appareil.

Prenons par exemple un code de 6x8 carrés, chacun des 48 carrés est une case qui est soit noire et représente un bit de valeur 1, soit blanche et représente un bit de valeur 0. Chaque ligne de 8 cases est représentée par un tuple de 8 bits et le QR code entier par une liste de tuples.



⇒

```
[ ( 0,1,0,0,1,1,0,1, ) ,
  ( 0,0,1,0,1,1,1,0, ) ,
  ( 0,1,0,0,1,0,0,0, ) ,
  ( 0,1,1,0,0,0,0,1, ) ,
  ( 0,1,1,1,0,0,1,0  ) ,
  ( 0,1,1,0,0,0,0,1  ) ]
```

Figure 1. Exemple de QR code simplifié et sa représentation en liste de tuples

Le décodage du QR code s'effectue alors en deux étapes :

- Chaque tuple est vu comme la représentation binaire d'un entier naturel en base 10.

  Par exemple, le tuple (0,1,1,0,0,0,0,1) représente le nombre binaire 01100001 qui vaut 97 en base 10.

- Chaque entier obtenu est ensuite associé à un caractère selon une table de correspondance. On utilisera la table des codes **ASCII** (Figure 2) qui fournit un caractère unique pour chaque entier compris entre 0 et 127.

  Par exemple, l'entier 97 code le caractère a.

Code	Caractère	Code	Caractère	Code	Caractère	Code	Caractère	Code	Caractère	Code	Caractère	Code	Caractère	Code	Caractère
0	NUL	16	DLE	32	Space	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(	56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41	)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[	107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	|
13	CR	29	GS	45	-	61	=	77	M	93	]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

Figure 2. Table des codes **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange).

La liste de tuples représentant le QR code, devient donc une liste d'entiers puis une chaîne de caractères c'est-à-dire l'information du QR code.

Question 1

Écrire une fonction en Python nommée `bin2dec` qui prend en paramètre un tuple représentant un nombre binaire et qui renvoie l'entier naturel en base 10 correspondant.
À l'aide des informations ci-dessus, déterminer la chaîne de caractères contenue dans le QR code de la figure 1 pour découvrir le nom de l'inventeur de ce système de codage.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire une fonction en Python nommée `qrcode2dec` qui prend en paramètre une liste de tuples représentant un QR code et qui renvoie une liste d'entiers décimaux correspondant à chacune des lignes du QR code.
Proposer un test de `qrcode2dec` qui utilisera la représentation du QR code de la Figure 1 fourni dans le module `ascii.py`.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

La table **ASCII** est ici implémentée dans le dictionnaire `dict_ascii` du module `ascii.py`.
Il est utilisé par la fonction fournie `dec2str` qui prend en paramètre une liste d'entiers et renvoie une chaîne formée des caractères correspondants dans la table **ASCII**.
Exécuter la fonction fournie `test_dec2str` et observer les résultats affichés. Identifier le problème et proposer une modification de la fonction dec2str pour l'éviter.
Après modification, la fonction `dec2str` devra toujours renvoyer une chaîne lisible.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 4

On souhaite maintenant réaliser l'opération inverse : générer un QR code à partir d'un texte. La fonction `str2qrcode(message)` a été rédigée dans ce but. Elle parcourt les caractères du message, retrouve leur code ASCII, le convertit en binaire et génère le tuple correspondant. Cependant, en exécutant cette fonction sur la chaîne contenue dans le QR code la figure 1, on obtient un résultat qui n'est pas exactement le QR code de la figure 1.
Analyser le code de la fonction `str2qrcode`. Identifier la source de ce problème, puis proposer une modification du code afin de garantir l'obtention d'un QR code simplifié valide.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `qrcode.py` ;
- un module `ascii.py` contenant le dictionnaire de conversion `dict_ascii` et des données de tests.

BACCALAURÉAT

SESSION 2026

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°23

DURÉE DE L'ÉPREUVE : 1 heure

Le sujet comporte 4 pages numérotées de 1 / 4 à 4 / 4
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Cette situation d'évaluation comporte ce document ainsi que des fichiers de codes et de données présents sur l'ordinateur à la disposition du candidat. Le candidat doit restituer ce document avant de sortir de la salle d'examen. Le candidat doit agir en autonomie et faire preuve d'initiative tout au long de l'épreuve.
En cas de difficulté, le candidat peut solliciter l'examinateur afin de lui permettre de continuer la tâche. Des moments privilégiés pour solliciter l'examinateur sont indiqués dans le document sous la forme d'appels professeur.
L'examinateur peut intervenir à tout moment, s'il le juge utile.

L'entreprise *PlovaNuviam* développe un module d'acquisition *DEMETER*, de transmission et de traitement de données environnementales (température, humidité…) pour la prévention de maladies dans les cultures agricoles comme les vignes.

Plusieurs fois par heure, chaque module mesure la température et l'humidité et transmet les données par ondes radio. Ces données sont réceptionnées sur un serveur dont le rôle est de vérifier, d'analyser et de traiter les informations.

Le but de ce sujet est de finaliser la programmation du serveur et de répondre à certaines problématiques.

Un code Python à corriger et à compléter ainsi qu'un jeu de données sont fournis avec ce document.

Description des données envoyées

Les données mesurées par chacun des modules *DEMETER* sont envoyées par trames de 40 bits, dont la forme est la suivante :

ID (8 bits)	Clé de sécurité (8 bits)	Température (12 bits)	Humidité (8 bits)	Bits de contrôle (4 bits)
00101010	11001000	010010001100	01100010	1101

Pour une trame donnée notée `trame` :

- les 8 premiers bits `trame[0:8]` correspondent à l'identifiant (ID) du module en binaire classique ;
- les 8 bits suivants `trame[8:16]` sont une clé de sécurité ;
- les 12 bits suivants `trame[16:28]` encodent la température. Pour l'obtenir en °C, il faut convertir cette valeur binaire en décimal, lui soustraire 900, puis diviser le tout par 10. Par exemple, `010010001100` vaut 1164 en décimal, ce qui donne (1164 - 900) / 10 = 26.4 °C ;
- les 8 bits suivants `trame[28:36]` encodent l'humidité en pourcentage. Attention, elle est codée en Décimal Codé Binaire (BCD) : chaque chiffre décimal est codé séparément sur 4 bits. Par exemple, 01100010 se sépare en 0110 (6) et 0010 (2), ce qui donne 62 %. Exception : 100% est codé `10100000` ;

- les 4 derniers bits `trame[36:40]` sont des bits de parité permettant de vérifier l'intégrité des 4 premiers blocs. Le bit de parité vaut 0 si le nombre de '1' dans le bloc correspondant est pair, sinon il vaut 1. Par exemple, pour l'ID 00101010, le nombre de '1' est impair (3), donc le premier bit de contrôle est 1.

Indication : en Python, l'expression `int(chaine, 2)` prend une chaîne de caractère contenant l'écriture binaire d'un entier et renvoie cet entier.

Question 1

Dans le fichier `transmission.py`, écrire en autonomie la fonction `decoder_temperature(trame)` qui prend en paramètre une chaîne de caractères représentant une trame et renvoie la température sous forme de flottant. Écrire ensuite la fonction `decoder_humidite(trame)` qui renvoie l'humidité sous forme d'entier.
Écrire des tests pour vérifier que ces fonctions renvoient bien 26.4 et 62 avec la trame d'exemple de l'énoncé.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 2

Écrire la fonction `est_valide(trame)` qui prend une trame en paramètre et renvoie `True` si les 4 bits de contrôle correspondent bien à la parité des 4 blocs de données, et `False` sinon.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Question 3

Le fichier `data.txt` contient une série de plus de 300 transmissions réelles.
Dans le fichier `analyse.py`, une fonction d'analyse a été écrite pour automatiser le traitement et l'affichage des données valides.
Exécuter le programme et constater qu'il produit un message d'erreur.
Analyser ce plantage. Identifier sa cause en observant, entre autres, le contenu du fichier `data.txt`.

Question 4

Proposer des corrections de la classe `Transmission` permettant de la rendre plus robuste afin de pouvoir effectuer l'analyse.

Appeler le professeur pour lui présenter votre réponse ou en cas de difficulté.

Description du dossier

Le dossier fourni au candidat sur l'ordinateur comporte les éléments suivants :

- une version PDF de l'énoncé ;
- un code source de départ `transmission.py`, `analyse.py` ;
- un fichier `data.txt`.

Préparation de l'environnement

Pour faire fonctionner le code fourni dans le dossier, les bibliothèques suivantes doivent être présentes : `matplotlib`