Partie A

Dans le thème “La photographie numérique” du programme de SNT, l’une des capacités attendues est de “traiter par programme une image pour la transformer en agissant sur les trois composantes de ses pixels.”.

On donne ci-dessous un extrait du programme de SNT (thème photographie numérique) :

Contenus	Capacités attendues
Photosites, pixels, résolution (du capteur, de l’image), profondeur de couleur	Distinguer les photosites du capteur et les pixels de l’image en comparant les résolutions du capteur et de l’image selon les réglages de l’appareil.
Métadonnées EXIF	Retrouver les métadonnées d’une photographie.
Traitement d’image	Traiter par programme une image pour la transformer en agissant sur les trois composantes de ses pixels.
Rôle des algorithmes dans les appareils photo numériques	Expliciter des algorithmes associés à la prise de vue. Identifier les étapes de la construction de l’image finale.

1. En vous aidant de la documentation de la bibliothèque Python Pillow donnée dans l’annexe 1, écrivez un programme Python permettant de transformer une image couleur au format JPEG (nom : “`photo.jpg`” ; définition de l’image en pixels : 800 x 600) en image en niveaux de gris de même format puis de l’afficher.
2. Proposez une activité destinée à vos élèves de seconde leur permettant d’écrire (ou de compléter) un programme Python ayant pour but de modifier “une image pour la transformer en agissant sur les trois composantes de ses pixels.”
3. Proposez une correction, à destination des élèves, de l’activité que vous aurez rédigée à la question 2.

Partie B

L’étude des “types et valeurs de base” est au programme de la spécialité NSI en classe de première :

Contenus	Capacités attendues	Commentaires
Écriture d’un entier positif dans une base b ≥ 2	Passer de la représentation d’une base dans une autre.	Les bases 2, 10 et 16 sont privilégiées.
Représentation binaire	Évaluer le nombre de bits	Il s’agit de décrire les tailles

d'un entier relatif	nécessaires à l'écriture en base 2 d'un entier, de la somme ou du produit de deux nombres entiers. Utiliser le complément à 2.	courantes des entiers (8, 16, 32 ou 64 bits). Il est possible d'évoquer la représentation des entiers de taille arbitraire de Python.
Représentation approximative des nombres réels : notion de nombre flottant	Calculer sur quelques exemples la représentation de nombres réels : 0.1, 0.25 ou 1/3.	0.2 + 0.1 n'est pas égal à 0.3. Il faut éviter de tester l'égalité de deux flottants. Aucune connaissance précise de la norme IEEE-754 n'est exigible.
Valeurs booléennes : 0, 1. Opérateurs booléens : and, or, not. Expressions booléennes	Dresser la table d'une expression booléenne.	Le ou exclusif (xor) est évoqué. Quelques applications directes comme l'addition binaire sont présentées. L'attention des élèves est attirée sur le caractère séquentiel de certains opérateurs booléens.
Représentation d'un texte en machine. Exemples des encodages ASCII, ISO-8859-1, Unicode	Identifier l'intérêt des différents systèmes d'encodage. Convertir un fichier texte dans différents formats d'encodage.	Aucune connaissance précise des normes d'encodage n'est exigible.

4. Vous trouverez dans l'annexe 2 un exercice donné à des élèves de première NSI dans le cadre d'une évaluation. Proposez une correction de cet exercice à destination de vos élèves.
5. Expliquez le principe de la représentation en machine des nombres à virgule flottante, par exemple en vous appuyant sur la norme IEEE-754.
6. Expliquez la phrase suivante du programme : "0.2 + 0.1 n'est pas égal à 0.3".

Partie C

L'annexe 3 est un extrait de la note de service qui précise les modalités de l'épreuve pratique de l'enseignement de spécialité NSI au baccalauréat.

7. Voici l'énoncé de l'exercice 1 d'un sujet tiré de la banque 2022 :

   Écrire une fonction maxi qui prend en paramètre une liste tab de nombres entiers et qui renvoie un couple donnant le plus grand élément de cette liste ainsi que l'indice de la première apparition de ce maximum dans la liste.

   Exemple : >>> maxi([1,5,6,9,1,2,3,7,9,8])
   (9,3)

   Proposez un corrigé de cet exercice.

8. Voici l'énoncé de l'exercice 2 d'un sujet tiré de la banque 2022 :

   La fonction recherche prend en paramètres deux chaines de caractères gene et seq_adn et renvoie True si on retrouve gene dans seq_adn et False sinon. Compléter le code Python ci-dessous pour qu'il implémente la fonction recherche.

```
def recherche(gene, seq_adn):
    n = len(seq_adn)
    g = len(gene)
    i = ...
    trouve = False
    while i < ... and trouve == ... :
        j = 0
        while j < g and gene[j] == seq_adn[i+j]:
            ...
        if j == g:
            trouve = True
        ...
    return trouve
```

   Proposez un corrigé de cet exercice.

9. Dans le cadre de la préparation de vos élèves de première à cette épreuve pratique, vous élaborez le contenu d'une séance de Travaux Pratiques qui s'appuiera sur la résolution d'un sujet. Rédigez un sujet type « épreuve pratique », pour vos élèves de première NSI en respectant les préconisations de la note de service dont l'extrait est en annexe 3.

   Le premier exercice devra porter sur la partie suivante du programme de première de l'enseignement de spécialité NSI Rédigez un exercice de type "Exercice I". Cet exercice devra porter sur la recherche dichotomique.

   On donne ci-dessous un extrait du programme de première NSI :

Contenus	Capacités attendues	Commentaires
Recherche dichotomique dans un tableau trié	Montrer la terminaison de la recherche dichotomique à l'aide d'un variant de boucle.	Des assertions peuvent être utilisées. La preuve de la correction peut être présentée par le professeur.

10. Proposez une correction pour l'exercice proposé à la question 9
11. Rédigez un exercice de type "Exercice II". Cet exercice devra porter sur la partie "dictionnaire" du programme de première.

    On donne ci-dessous un extrait du programme de première NSI :

Contenus	Capacités attendues	Commentaires
	Construire une entrée de	Il est possible de présenter les

Dictionnaires par clés et valeurs	dictionnaire. Itérer sur les éléments d'un dictionnaire.	données EXIF d'une image sous la forme d'un enregistrement. En Python, les p-uplets nommés sont implémentés par des dictionnaires. Utiliser les méthodes keys(), values () et items ().

12. Proposez une correction pour l'exercice rédigé à la question 11.

Partie D

Le tri par insertion et le tri par sélection sont au programme de la spécialité NSI en classe de première.

Contenus	Capacités attendues	Commentaires
Tris par insertion, par sélection de l'enseignement de spécialité NSI en classe de terminale :	Écrire un algorithme de tri. Décrire un invariant de boucle qui prouve la correction des tris par insertion, par sélection.	La terminaison de ces algorithmes est à justifier. On montre que leur coût est quadratique dans le pire cas.

13. Expliquez, comme vous le feriez devant une classe de première, le principe du tri par insertion.
14. Expliquez, comme vous le feriez devant une classe de première, le principe du tri par sélection.
15. Montrez que la complexité en temps dans le pire des cas de l'algorithme du tri par insertion est quadratique.
16. Démontrez la correction de l'algorithme du tri par insertion à l'aide d'un invariant de boucle.
17. Écrivez une fonction Python *tri_selection* qui prend en paramètre une liste d'entiers *L*. Cette fonction devra trier la liste passée en paramètre en utilisant l'algorithme du tri par sélection.

L'étude de la méthode "diviser pour régner" est au programme de l'enseignement de spécialité NSI en classe de terminale :

Contenus	Capacités attendues	Commentaires
Méthode « diviser pour régner ».	Écrire un algorithme utilisant la méthode « diviser pour régner ».	La rotation d'une image bitmap d'un quart de tour avec un coût en mémoire constant est un bon exemple. L'exemple du tri fusion permet également d'exploiter la récursivité et d'exhiber un algorithme de coût en $n.\log_2(n)$ dans les pires des cas.

Il est précisé dans la rubrique “commentaires”, que “l’exemple du tri fusion permet également d’exploiter la récursivité et d’exhiber un algorithme de coût en $n.\log_2(n)$ dans les pires des cas.”.

18. Expliquez, comme vous le feriez devant une classe de terminale, le principe du tri-fusion.
19. Expliquez, comme vous le feriez devant une classe de terminale, le coût en $n.\log_2(n)$ dans le pire des cas du tri-fusion.
20. À la suite d’une activité sur le tri-fusion, une élève vient vous dire qu’elle a regardé une vidéo consacrée au tri-rapide (quicksort) et qu’elle a l’impression que cet algorithme ressemble au tri-fusion étudié en classe, comment lui expliquez-vous les points communs et les différences entre ces deux algorithmes ?

Partie E

L’étude de la récursivité est au programme de terminale de la spécialité NSI.

Contenus	Capacités attendues	Commentaires
Récursivité	Écrire un programme récursif. Analyser le fonctionnement d’un programme récursif.	Des exemples relevant de domaines variés sont à privilégier

Après avoir proposé à vos élèves un premier exemple de fonction récursive avec la fonction permettant de calculer la factorielle d’un entier, vous décidez de faire travailler vos élèves sur la suite de Fibonacci. On rappelle que cette suite est définie par :

$F_0 = 0$
$F_1 = 1$
Pour tout $n \geq 2$, $F_n = F_{n-1} + F_{n-2}$

21. Écrivez une fonction Python récursive *fib* qui prend en paramètre un entier *n* et qui renvoie le terme F_n de la suite de Fibonacci.
22. Un élève a réussi à écrire la fonction *fib* et s’interroge sur le fait que l’exécution de *fib(120)* “fasse planter l’ordinateur”, que pouvez-vous lui répondre ?

Plus tard dans l’année, vous décidez de reprendre cet exemple de la suite de Fibonacci pour illustrer le début de votre cours sur la programmation dynamique.

On donne ci-dessous un extrait du programme de l’enseignement de spécialité NSI en Terminale :

Contenus	Capacités attendues	Commentaires
Programmation dynamique.	Utiliser la programmation dynamique pour écrire un algorithme.	Les exemples de l’alignement de séquences ou du rendu de monnaie peuvent être présentés. La discussion sur le coût en mémoire peut être développée.

23. Expliquez, en quelques lignes, le principe de la programmation dynamique.

24. Utilisez la programmation dynamique afin d'améliorer l'efficacité de la fonction écrite à la question 21. Vous écrirez deux programmes Python : un premier programme qui utilisera l'approche ascendante (bottom-up) et un deuxième programme qui utilisera l'approche descendante (top-down).

Après ce premier exemple d'utilisation de la programmation dynamique, vous décidez de faire travailler vos élèves sur le problème du rendu de monnaie. Comme ce problème a déjà été traité en première dans le cadre du cours sur les algorithmes gloutons, vous décidez de donner un exercice à vos élèves de terminale afin de voir ce qu'ils ont retenu de la résolution du problème du rendu de monnaie par une méthode gloutonne.

On donne ci-dessous un extrait du programme de l'enseignement de spécialité NSI en première :

Contenus	Capacités attendues	Commentaires
Algorithmes gloutons	Résoudre un problème grâce à un algorithme glouton.	Exemples : problèmes du sac à dos ou du rendu de monnaie. Les algorithmes gloutons constituent une méthode algorithmique parmi d'autres qui seront vues en terminale.

25. Expliquez, en quelques lignes, le principe d'un algorithme glouton.
26. Voici une fonction Python *rendu* proposée par un élève de terminale permettant de résoudre le problème du rendu de monnaie en utilisant une méthode gloutonne. Cette fonction prend en paramètre la somme à rendre (en euro) et renvoie le nombre minimum de pièces qu'il est possible de rendre. Quel est, selon vous, le problème qui risque de se poser lors de l'exécution de ce programme ? Proposez une modification afin de supprimer ce problème.

```
def rendu(s):
    index = 0
    tab_piece = [2, 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01]
    somme_rendu = 0
    nbre_piece = 0
    while somme_rendu != s :
        somme_a_rendre = s - somme_rendu
        if somme_a_rendre < tab_piece[index]:
            index = index + 1
        else :
            somme_rendu = somme_rendu + tab_piece[index]
            nbre_piece = nbre_piece + 1
    return nbre_piece
```

27. Pour les questions suivantes, on supprime la pièce de 1 centime du tableau *tab_piece* (valeur 0.01). Donnez une valeur de *s* (paramètre de la fonction *rendu*) pour laquelle la méthode gloutonne ne permet pas de trouver une solution alors que cette solution existe.
28. Suite aux constatations faites à la question 27, vous décidez d'abandonner la méthode gloutonne pour résoudre le problème du rendu de monnaie et d'écrire à la place une fonction récursive *rendu_rec* qui prend en paramètre la somme à rendre *s*

(en centimes) et qui renvoie le nombre minimum de pièces qu'il est possible de rendre (cette fonction doit trouver la solution, si cette solution existe). Écrivez cette fonction en Python.

29. En utilisant la programmation dynamique, améliorer l'efficacité du programme écrit à la question 28.

Partie F

Vous trouverez en annexe 4 l'énoncé d'un exercice sur les arbres binaires donné à des élèves de terminale NSI lors d'une évaluation.

On donne ci-dessous des extraits du programme de l'enseignement de spécialité NSI de terminale :

Contenus	Capacités attendues	Commentaires
Arbres : structures hiérarchiques. Arbres binaires : nœuds, racines, feuilles, sous-arbres gauches, sous-arbres droits.	Arbres : structures hiérarchiques. Arbres binaires : nœuds, racines, feuilles, sous-arbres gauches, sous-arbres droits.	On fait le lien avec la rubrique « algorithmique ».

Contenus	Capacités attendues	Commentaires
Algorithmes sur les arbres binaires et sur les arbres binaires de recherche.	Calculer la taille et la hauteur d'un arbre. Parcourir un arbre de différentes façons (ordres infixe, préfixe ou suffixe ; ordre en largeur d'abord). Rechercher une clé dans un arbre de recherche, insérer une clé.	Une structure de données récursive adaptée est utilisée. L'exemple des arbres permet d'illustrer la programmation par classe. La recherche dans un arbre de recherche équilibré est de coût logarithmique.

30. Proposez un corrigé de cet exercice à destination de vos élèves.
31. Proposez un barème sur 10 points pour cet exercice en expliquant vos choix.
32. On donne dans l'annexe 5 un extrait d'une copie d'élève. Corrigez cet extrait de copie, la copie corrigée sera rendue à l'élève.
33. Globalement, cet exercice est difficile. Proposez, en les justifiant, au moins deux modifications permettant de le rendre plus accessible.

ANNEXES

Annexe 1

Documentation succincte de la bibliothèque Pillow.

La bibliothèque Pillow propose la classe Image. La méthode open de la classe Image permet d'obtenir une instance de la classe Image. Cette méthode open prend en paramètre l'url d'un "fichier image".

Exemple :

```
from PIL import Image
img = Image.open("chat.jpg")
```

L'attribut size de la classe Image permet d'obtenir la dimension ou taille de l'image (en pixel) sous la forme d'un tuple.

Exemple :

```
from PIL import Image
img = Image.open("chat.jpg")
res = img.size
# le contenu de la variable res est le tuple (800,600) pour une image de
# taille 800 pixels par 600 pixels
```

La méthode `getpixel` de la classe Image renvoie un tuple correspondant aux composantes "rouge, vert, bleu" du pixel de coordonnées (x,y). Cette méthode prend en paramètre un tuple contenant les coordonnées (x,y) du pixel considéré.

On rappelle que le pixel de coordonnées (0,0) correspond au pixel en haut à gauche de l'image.

Exemple :

```
from PIL import Image
img = Image.open("chat.jpg")
r, v, b = img.getpixel((100,150))
# si le pixel de coordonnées (100,150) est rouge
# on obtiendra r = 255, v = 0, b = 0
```

La méthode `putpixel` de la classe Image permet de modifier la couleur du pixel de coordonnées (x,y). Cette méthode prend en paramètres 2 tuples : un premier tuple correspondant aux coordonnées du pixel que l'on souhaite modifier, un deuxième tuple correspondant à la nouvelle couleur du pixel de coordonnées (x,y).

Exemple :

```
from PIL import Image
img = Image.open("chat.jpg")
img.putpixel((100,150), (0,255,0))
# le pixel de coordonnées (100,150) est désormais vert
```

La méthode `show` de la classe image permet d'afficher l'image qui vient d'être modifiée (par exemple à l'aide de la méthode putpixel).

Exemple :

```
from PIL import Image
img = Image.open("chat.jpg")
img.putpixel((100,150), (0,255,0))
img.show()
```

Annexe 2

Exercice proposé à des élèves de première de spécialité NSI lors d'une évaluation :

Exercice 2

1. Convertissez 27_{10} en base 2.
2. Convertissez 1000101_2 en base 10.
3. Convertissez 44_{10} en base 16.
4. Convertissez $A1_{16}$ en base 10.
5. Convertissez 100110_2 en base 16.
6. Convertissez $F3_{16}$ en base 2.
7. En utilisant le complément à 2 sur 8 bits, convertissez -9_{10} en base 2.
8. En utilisant le complément à 2 sur 8 bits, convertissez 10010110_2 en base 10.
9. Convertissez $1,0100_2$ en base 10.
10. Convertissez $3,125_{10}$ en base 2.

Annexe 3

NOR : MENE2001797N
Extrait de la note de service n° 2020-030 du 11-2-2020
MENJS - DGESCO A2-1

[…]

2. Partie pratique

Durée : 1 heure

Modalités

La partie pratique consiste en la résolution de deux exercices sur ordinateur, chacun étant noté sur 4 points.

Le candidat est évalué sur la base d'un dialogue avec un professeur-examinateur. Un examinateur évalue au maximum quatre élèves. L'examinateur ne peut pas évaluer un élève qu'il a eu en classe durant l'année en cours.

L'évaluation de cette partie se déroule au cours du deuxième trimestre pendant la période de l'épreuve écrite de spécialité.

- Premier exercice

Le premier exercice consiste à programmer un algorithme figurant explicitement au programme, ne présentant pas de difficulté particulière, dont on fournit une spécification. Il s'agit donc de restituer un algorithme rencontré et travaillé à plusieurs reprises en cours de formation. Le sujet peut proposer un jeu de test avec les réponses attendues pour permettre au candidat de vérifier son travail.

- Deuxième exercice

Pour le second exercice, un programme est fourni au candidat. Cet exercice ne demande pas l'écriture complète d'un programme, mais permet de valider des compétences de programmation suivant des modalités variées : le candidat doit, par exemple, compléter un programme « à trous » afin de répondre à une spécification donnée, ou encore compléter un programme pour le documenter, ou encore compléter un programme en ajoutant des assertions, etc.

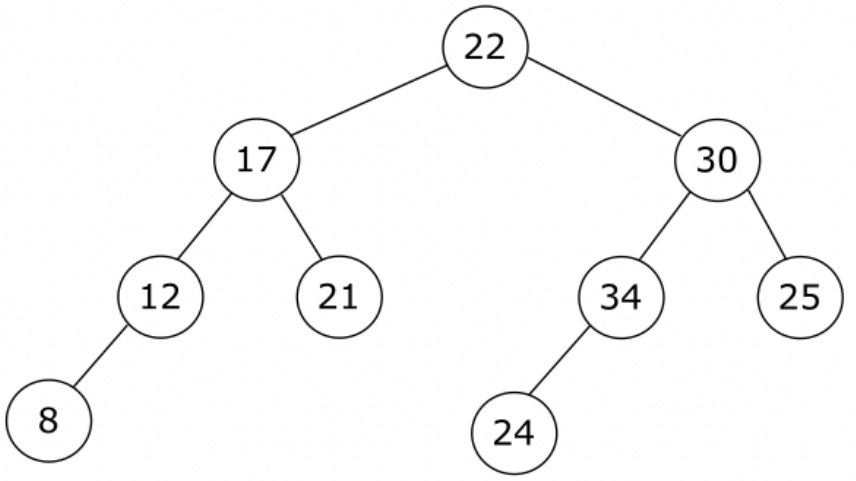
[…]

Annexe 4

Extrait du sujet d'une évaluation de terminale

Exercice III

Cet exercice porte sur les arbres binaires et les arbres binaires de recherche.
Dans cet exercice, on considérera que la racine d'un arbre binaire a une profondeur de 1.



Arbre 1

PARTIE A

1. Quelle est la taille de l'arbre 1 ?
2. Quelle est la hauteur de l'arbre 1 ?
3. L'arbre 1 est-il un arbre binaire de recherche ? Justifiez votre réponse.
4. Parcourir l'arbre 1 dans l'ordre suffixe

PARTIE B

Dans la suite de cet exercice, on utilisera la classe ArbreBinaire donnée ci-dessous.

```
class ArbreBinaire:

    def __init__(self, valeur):
        self.valeur = valeur
        self.enfant_gauche = None
        self.enfant_droit = None

    def insert_gauche(self, valeur):
        if self.enfant_gauche == None:
            self.enfant_gauche = ArbreBinaire(valeur)
        else:
            new_node = ArbreBinaire(valeur)
            new_node.enfant_gauche = self.enfant_gauche
            self.enfant_gauche = new_node
```

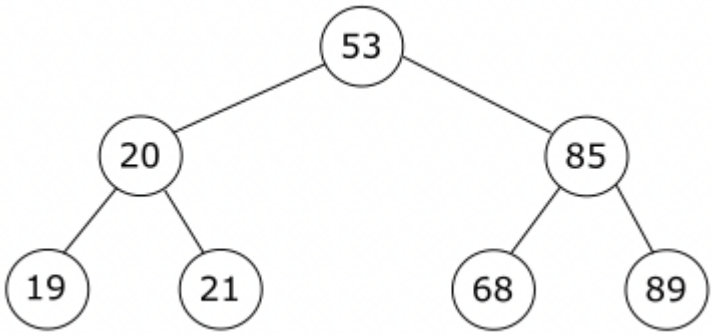
```
def insert_droit(self, valeur):
    if self.enfant_droit == None:
        self.enfant_droit = ArbreBinaire(valeur)
    else:
        new_node = ArbreBinaire(valeur)
        new_node.enfant_droit = self.enfant_droit
        self.enfant_droit = new_node

def get_valeur(self):
    return self.valeur

def get_gauche(self):
    return self.enfant_gauche

def get_droit(self):
    return self.enfant_droit
```

5. En utilisant la classe `ArbreBinaire`, implémenter en Python l'arbre binaire donné ci-dessous (arbre 2)



arbre 2

6. Écrire la fonction *taille* qui prend en paramètre une instance T de la classe `ArbreBinaire` et qui renvoie la taille de l'arbre T.
7. Écrire la fonction *hauteur* qui prend en paramètre une instance T de la classe `ArbreBinaire` et qui renvoie la hauteur de l'arbre T.
8. Écrire la fonction *parcours_infixe* qui prend en paramètre une instance T de la classe `ArbreBinaire` et qui permet d'afficher dans la console le parcours de l'arbre binaire T dans l'ordre infixe
9. Écrire la fonction *arbre_recherche* qui prend en paramètres une instance T de la classe `ArbreBinaire` (cette instance représentant un arbre binaire de recherche dont les valeurs des nœuds sont des entiers) et un entier k. Cette fonction renvoie True si k appartient à T et False dans le cas contraire.

Annexe 5 **À RENDRE AVEC LA COPIE**

Extrait d'une copie d'élève

Exercice III

PARTIE A

1. 9

2. 3

3. Oui, cet arbre binaire est bien un arbre binaire de recherche, car tous les nœuds situés à gauche de la racine sont plus petits que 22 et tous les nœuds situés à droite de la racine sont plus grands que 22.

4. 22 - 17 - 12 - 8 - 21 - 30 - 34 - 24 - 25

PARTIE B

5.
```
arbre = ArbreBinaire(53)
arbre.insert_gauche(20)
arbre.insert_droit(85)
n_20 = arbre.get_gauche()
n_85 = arbre.get_droit()
n_20.insert_gauche(19)
n_20.insert_droit(21)
n_85.insert_gauche(68)
n_85.insert_droit(89)
```

6.
```
def taille(T):
    if T==None:
        return 0
    else :
        return 1 + taille(T-1)
```

7.

```
def hauteur(T):
    if T == None:
        return 0
    else :
        return 1 + hauteur(T.get_gauche()) + hauteur(T.get_droit())
```

8.

```
def parcours_infixe(T):
    if T == None:
        return 0
    else :
        print(T)
        parcours_infixe(T.get_gauche())
        parcours_infixe(T.get_droit())
```

9.

```
def arbre_recherche(T,k):
    while T != None:
        if T.get_valeur() == k:
            return True
        elif T.get_valeur() < k:
            T = T.get_gauche()
        else :
            T = T.get_droit()
    return False
```