MODULE CS54
Examen du 16 décembre 2021 (1h)

Module

N'essayez pas nécessairement de tout faire. Lisez bien l'intégralité du sujet avant de choisir vos exercices. La notation tiendra compte de la validité des réponses, mais aussi de la présentation et de la clarté de la rédaction.

Documents interdits.

★ **Questions de cours.** (4 pts)

▷ **Question 1:** Définissez en français (sans équation) les notations O, Ω et Θ utilisées pour dénoter la complexité algorithmique en insistant sur leurs relations les unes avec les autres 3 pt).

▷ **Question 2:** À quelles classes de complexité (en notation Θ) appartiennent les algorithmes 1 et 2 suivants ? (1pt)

algorithme 1

```
k=8
pour i = n à 1 faire
  pour j = k à n faire
     x += 3
```

algorithme 2

```
pour i = 1 à n faire
  pour j = n/2 à i faire
    x += 3
pour i = 1 à n faire
  y = x + 5
```

★ **Exercice 1: Code récursif mystère** (6 pts).
Considérez le code mystère ci-contre.

```
def puzzle(i : int, j: int) -> int:
    if (i == 0):
        return 0
    elif ((j % 2) == 1):
        return j+puzzle(i-1,j+1)
    else:
        return puzzle(i   ,j+1)

def myst(i : int) -> int :
        return puzzle(i,1)
```

▷ **Question 1:** Explicitez les appels récursifs (dessinez l'arbre de Noël des appels) effectués pour `myst(4)` (1 pt).

▷ **Question 2:** Quelle somme cette fonction calcule-t-elle ? Exprimez le calcul réalisé par la fonction sous forme d'un $\sum_{?}^{?}$ et explicitez à quoi correspond ce calcul (1 pt).

▷ **Question 3:** Calculez le résultat de la fonction myst pour i=1, i=2, i=3, i=4, i=5 et i=6. Que semble calculer cette fonction (en plus de la somme vue plus haut) ? (0.5 pt)

▷ **Question 4:** Quelle est la complexité algorithmique de `puzzle` (en nombre d'appels récursifs) ? (0.5 pt)

▷ **Question 5:** Dérécursivez la fonction *puzzle* en appliquant les méthodes vues en cours (en une ou plusieurs étapes). Explicitez ce que vous faites et pourquoi (3 pts).

★ **Exercice 2: Fiboscopie** (10 pts).

Edouard Zeckendorf (1901-1983), était un médecin belge, mathématicien amateur de génie, père du théorème éponyme. Son théorème dit que tout entier naturel positif peut se décomposer en une somme de termes **tous différents** de la suite de Fibonacci. Par exemple, l'entier $123 = 89 + 21 + 8 + 5$ (rappel des premiers termes de la séquence de Fibonacci : $0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89$). Il peut exister plusieurs décompositions d'un même entier (123 résulte également de la somme suivante : $55 + 34 + 21 + 13$ ou $89 + 21 + 13$).

▷ **Question 1:** Construisez un algorithme qui, étant donné un entier n, renvoie une liste de termes (tous différents) de la séquence de Fibonacci dont la somme est égale à n. Votre algorithme devra également expliciter le calcul des termes de la suite de Fibonacci. Si votre algorithme renvoie la séquence la plus courte possible, vous bénéficierez d'un bonus de 3 pts (6-9 pts).

▷ **Question 2:** Donnez la complexité de l'intégralité du calcul de la séquence (1 pt)