Série 3

1 Quel est le bon algorithme ?

Lequel des quatre algorithmes récursifs suivants permet de calculer la somme des n premiers nombres pairs ? (ex : si $n = 4$, alors la sortie doit valoir $2 + 4 + 6 + 8 = 20$). Expliquez pourquoi les autres ne fonctionnent pas.

algo1
entrée : *nombre entier positif ou nul n*
sortie : *nombre entier positif ou nul s*

Si $n = 0$
  Sortir : 0
Sortir : **algo1**$(n-2) + n$

algo2
entrée : *nombre entier positif ou nul n*
sortie : *nombre entier positif ou nul s*

Si $n = 0$
  Sortir : 0
Sortir : **algo2**$(2n-2) + 2n$

algo3
entrée : *nombre entier positif ou nul n*
sortie : *nombre entier positif ou nul s*

Si $n = 0$
  Sortir : 0
Sortir : **algo3**$(n-1) + 2n$

algo4
entrée : *nombre entier positif ou nul n*
sortie : *nombre entier positif ou nul s*

Si $n = 0$
  Sortir : 0
Sortir : $2 \times$ (**algo4**$(n-1) + n$)

2 Mystère 1

mystère
entrée : *liste ordonnée L de nombres entiers, n taille de la liste, x nombre entier*
sortie : *valeur binaire oui/non*

$a \longleftarrow 1$
$b \longleftarrow n$
Tant que $b \geq a$
  $j \longleftarrow \lfloor \frac{a+b}{2} \rfloor$
  Si $x = L(j)$
    Sortir : *oui*
  Sinon
    Si $L(j) < x$
      $a \longleftarrow j + 1$
    Sinon
      $b \longleftarrow j - 1$
Sortir : *non*

a) Si $L = (2, 2, 2, 2, 2, 2, 2, 2)$, $n = 8$ et $x = 1$, quelle est la sortie de l'algorithme et combien de fois la boucle "Tant que" est-elle exécutée ?

b) En général, quelle est la sortie de l'algorithme et quelle est sa complexité temporelle en fonction de la taille n de la liste L ? (utiliser la notation $\Theta(\cdot)$)

3 Au temps des Grecs

Proposez une version récursive de l'algorithme d'Euclide, qui calcule le plus grand diviseur commun de deux entiers naturels a et b : (*Note :* $a \bmod b$ désigne le reste de la division euclidienne de a par b)

algorithme d'Euclide

entrée : *a, b deux entiers positifs*
sortie : *pgcd(a,b)*

Tant que $b \neq 0$
  $temp \longleftarrow a \bmod b$
  $a \longleftarrow b$
  $b \longleftarrow temp$
Sortir : a

4 Mystère 2

On considère l'algorithme suivant :

algo mystère

entrée : *nombre entier strictement positif n*
sortie : *nombre entier s*

Si $n = 1$ ***or*** $n = 2$
  Sortir : *1*
$s \longleftarrow 0$
Pour *i allant de* 1 *à* $n - 2$
  $s \longleftarrow s + algo\_mystère(i)$
Sortir : s

Pour une valeur de $n \geq 3$, que vaut la sortie de cet algorithme ?

- ☐ 2^{n-2}
- ☐ $F(n-1)$
- ☐ 2^{n-3}
- ☐ $F(n-2)$

5 Mystère 3

On considère l'algorithme suivant :

mystère

entrée : *tableau tab de taille n*
sortie : *tableau res*

Si $n \leq 1$
  Sortir : tab
$mid \longleftarrow \lfloor n/2 \rfloor$
$left \longleftarrow mystère(tab[1 : mid])$
$right \longleftarrow mystère(tab[mid + 1 : n])$
$res \longleftarrow fusion(left,right)$
Sortir : res

On suppose que la fonction `fusion` combine deux tableaux triés de tailles n_1 et n_2 en temps linéaire $O(n_1 + n_2)$.

Les notations de sous-tableaux signifient :

$tab[i : j]$ est le sous-tableau contenant les éléments d'indices i à j,

Quelle est la complexité en pire cas de l'algorithme `mystère` en fonction de n ?

- ☐ $O(n)$
- ☐ $O(n \log n)$
- ☐ $O(n^2)$
- ☐ $O(2^n)$

6 Fonction `machin`

On considère le code suivant :

machin

entrée : *nombre entier n*
sortie : *nombre entier res*

Si $n = 0$
  Sortir : *0*
$res \longleftarrow (n \bmod 10) + machin(\lfloor n/10 \rfloor)$
Sortir : res

On calcule ensuite :

Afficher : machin(1234)

Quelle est la sortie du programme ?

- ☐ 4
- ☐ 123
- ☐ 10
- ☐ 0